NEC

Preliminary User's Manual

V850ES/SG2™

32-Bit Single-Chip Microcontroller

Hardware

μPD703260	μPD703270	μPD703280
μPD703260Y	μPD703270Y	μPD703280Y
μPD703261	μPD703271	μPD703281
μPD703261Y	μPD703271Y	μPD703281Y
μPD703262	μPD703272	μPD703282
μPD703262Y	μPD703272Y	μPD703282Y
μPD703263	μPD703273	μPD703283
μPD703263Y	μPD703273	μPD703283Y
μPD70F3261	μPD70F3271	μPD70F3281
μPD70F3261Y	μPD70F3271Y	μPD70F3281Y
μPD70F3263	μPD70F3273	μPD70F3283
μPD70F3263Y	μPD70F3273Y	μPD70F3283Y

Document No. U16541EJ1V0UM00 (1st edition)
Date Published February 2003 N CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

Purchase of NEC Electronics I^2C components conveys a license under the Philips I^2C Patent Rights to use these components in an I^2C system, provided that the system conforms to the I^2C Standard Specification as defined by Philips.

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

License no needed: μPD70F3261, 70F3261Y, 70F3263, 70F3263Y, 70F3271, 70F3271Y, 70F3273, 70F3273Y, 70F3281, 70F3281Y, 70F3283, 70F3283Y

The customer must judge the need for a license:

μPD703260, 703260Y, 703261, 703261Y, 703262, 703262Y, 703263, 703263Y, 703270, 703270Y, 703271, 703271Y, 703272, 703272Y, 703273, 703273Y, 703280, 703280Y, 703281, 703281Y, 703282, 703282Y, 703283, 703283Y

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

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

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

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

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

(Note)

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

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

M5 02. 11-1

Regional Information

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

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

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

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

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 01
Fax: 0211-65 03 327

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
  Fax: 091-504 28 60

- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
  Fax: 01-30-67 58 99

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

- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
  Fax: 040-244 45 80

- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
  Fax: 08-63 80 388

- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133
  Fax: 01908-670-290

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 Shanghai, Ltd.
Shanghai, P.R. China
Tel: 021-6841-1138
Fax: 021-6841-1137

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC Electronics Singapore Pte. Ltd.
Novena Square, Singapore
Tel: 6253-8311
Fax: 6250-3583

J02.11

PREFACE

Readers

This manual is intended for users who wish to understand the functions of the V850ES/SG2 and design application systems using these products.

Purpose

This manual is intended to give users an understanding of the hardware functions of the V850ES/SG2 shown in the Organization below.

Organization

This manual is divided into two parts: Hardware (this manual) and Architecture (**V850ES Architecture User's Manual**).

Hardware	Architecture
• Pin functions	• Data types
• CPU function	• Register set
• On-chip peripheral functions	• Instruction format and instruction set
• Flash memory programming	• Interrupts and exceptions
	• Pipeline operation

How to Read This Manual

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

To understand the details of an instruction function
→Refer to the **V850ES Architecture User's Manual** available separately.

Register format
→The name of the bit whose number is in angle brackets (<>) in the figure of the register format of each register is defined as a reserved word in the device file.

To understand the overall functions of the V850ES/SG2
→Read this manual according to 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 address:	Higher addresses on the top and lower addresses on the bottom
Note:	Footnote for item marked with **Note** in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numeric representation:	Binary ... xxxx or xxxxB
	Decimal ... xxxx
	Hexadecimal ... xxxxH
Prefix indicating power of 2 (address space, memory capacity):	K (kilo): $2^{10} = 1,024$
	M (mega): $2^{20} = 1,024^2$
	G (giga): $2^{30} = 1,024^3$

Related Documents

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

Documents related to V850ES/SG2

Document Name	Document No.
V850ES Architecture User's Manual	U15943E
V850ES/SG2 Hardware User's Manual	This manual

Documents related to development tools

Document Name		Document No.
IE-V850ES-G1 (In-Circuit Emulator)		To be prepared
IE-703288-G1-EM1 (In-Circuit Emulator Option Board)		To be prepared
CA850 Ver. 2.50 C Compiler Package	Operation	U16053E
	C Language	U16054E
	PM Plus	U16055E
	Assembly Language	U16042E
ID850 Ver. 2.50 Integrated Debugger	Operation	U16217E
SM850 Ver. 2.50 Systerm Simulator	Operation	U15182E
SM850 Ver. 2.00 or Later System Simulator	External Part User Open Interface Specification	U14873E
RX850 Ver. 3.13 or Later Real-Time OS	Basics	U13430E
	Installation	U13410E
	Technical	U13431E
RX850 Pro Ver. 3.15 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-FP4 Flash Memory Programmer		U15260E

CONTENTS

LIST OF FIGURES (1/9)

LIST OF FIGURES (2/9)

LIST OF FIGURES (3/9)

LIST OF FIGURES (4/9)

LIST OF FIGURES (5/9)

LIST OF FIGURES (6/9)

LIST OF FIGURES (7/9)

LIST OF FIGURES (8/9)

LIST OF FIGURES (9/9)

LIST OF TABLES (1/4)

LIST OF TABLES (2/4)

LIST OF TABLES (3/4)

LIST OF TABLES (4/4)

CHAPTER 1 INTRODUCTION

The V850ES/SG2 is one of the products in the NEC Electronics V850 Series™ of single-chip microcontrollers designed for low-power operation for real-time control applications.

1.1 General

The V850ES/SG2 is a 32-bit single-chip microcontroller that includes the V850ES CPU core and peripheral functions such as ROM/RAM, a timer/counter, serial interfaces, an A/D converter, and a D/A converter. Some models of the V850ES/SG2 are provided with IEBus® (Inter Equipment Bus®) or CAN (Control Area Network) as an automotive LAN.

In addition to high real-time response characteristics and 1-clock-pitch basic instructions, the V850ES/SG2 features multiply instructions, saturated operation instructions, bit manipulation instructions, etc., realized by a hardware multiplier, as optimum instructions for digital servo control applications. Moreover, as a real-time control system, the V850ES/SG2 enables an extremely high cost-performance for applications that require a low power consumption, such as audio and car audio.

Table 1-1 lists the products of the V850ES/SG2.

Table 1-1. V850ES/SG2 Product List

Function / Part Number	ROM Type	ROM Size	RAM Size	I^2C	IEBus	CAN	Maskable Interrupts External	Maskable Interrupts Internal	Non-maskable Interrupts
μPD703260	Mask ROM	256 KB	24 KB	None	None	None	8	47	2
μPD703260Y				On-chip					
μPD703261		384 KB	32 KB	None					
μPD703261Y				On-chip					
μPD70F3261	Flash memory			None					
μPD70F3261Y				On-chip					
μPD703262	Mask ROM	512 KB	40 KB	None					
μPD703262Y				On-chip					
μPD703263		640 KB	48 KB	None					
μPD703263Y				On-chip					
μPD70F3263	Flash memory			None					
μPD70F3263Y				On-chip					
μPD703270	Mask ROM	256 KB	24 KB	None	On-chip			51	
μPD703270Y				On-chip					
μPD703271		384 KB	32 KB	None					
μPD703271Y				On-chip					
μPD70F3271	Flash memory			None					
μPD70F3271Y				On-chip					
μPD703272	Mask ROM	512 KB	40 KB	None					
μPD703272Y				On-chip					
μPD703273		640 KB	48 KB	None					
μPD703273Y				On-chip					
μPD70F3273	Flash memory			None					
μPD70F3273Y				On-chip					
μPD703280	Mask ROM	256 KB	24 KB	None	None	On-chip		51	
μPD703280Y				On-chip					
μPD703281		384 KB	32 KB	None					
μPD703281Y				On-chip					
μPD70F3281	Flash memory			None					
μPD70F3281Y				On-chip					
μPD703282	Mask ROM	512 KB	40 KB	None					
μPD703282Y				On-chip					
μPD703283		640 KB	48 KB	None					
μPD703283Y				On-chip					
μPD70F3283	Flash memory			None					
μPD70F3283Y				On-chip					

Remark The part numbers of the V850ES/SG2 are shown as follows in this manual.

- Mask ROM version
  μPD703260, 703260Y, 703261, 703261Y, 703262, 703262Y, 703263, 703263Y, 703270, 703270Y, 703271, 703271Y, 703272, 703272Y, 703273, 703273Y, 703280, 703280Y, 703281, 703281Y, 703282, 703282Y, 703283, 703283Y
- Flash memory version
  μPD70F3261, 70F3261Y, 70F3263, 70F3263Y, 70F3271, 70F3271Y, 70F3273, 70F3273Y, 70F3281, 70F3281Y, 70F3283, 70F3283Y
- I^2C bus version (Y version)
  μPD703260Y, 703261Y, 703262Y, 703263Y, 703270Y, 703271Y, 703272Y, 703273Y, 703280Y, 703281Y, 703282Y, 703283Y, 70F3261Y, 70F3263Y, 70F3271Y, 70F3273Y, 70F3281Y, 70F3283Y
- General-purpose version
  μPD703260, 703260Y, 703261, 703261Y, 703262, 703262Y, 703263, 703263Y, 70F3261, 70F3261Y, 70F3263, 70F3263Y
- IEBus controller version
  μPD703270, 703270Y, 703271, 703271Y, 703272, 703272Y, 703273, 703273Y, 70F3271, 70F3271Y, 70F3273, 70F3273Y
- CAN controller version
  μPD703280, 703280Y, 703281, 703281Y, 703282, 703282Y, 703283, 703283Y, 70F3281, 70F3281Y, 70F3283, 70F3283Y

1.2 Features

- Number of instructions: 83
- Minimum instruction execution time: 50 ns (operating with main clock (fxx) of 20 MHz)
- General-purpose registers: 32 bits × 32 registers
- Instruction set:
  - Signed multiplication (16 × 16 → 32): 1 to 2 clocks
  - Signed multiplication (32 × 32 → 64): 1 to 5 clocks
  - Saturated operations (overflow and underflow detection functions included)
  - 32-bit shift instruction: 1 clock
  - Bit manipulation instructions
  - Load/store instructions with long/short format
- Memory space:
  - 64 MB of linear address space (for programs and data)
  - External expansion: Up to 16 MB (including 1 MB used as internal ROM/RAM)
  - Programmable wait function
  - Idle state insertion function
- External bus interface:
  - Separate bus/multiplexed bus output selectable
  - 8/16 bit data bus sizing function
  - Wait function
    - Programmable wait function
    - External wait function
  - Idle state function
  - Bus hold function
- Internal memory:
  - RAM: 24/32/40/48 KB (see **Table 1-1**)
  - Mask ROM: 256/384/512/640 KB (see **Table 1-1**)
  - Flash memory: 384/640 KB (see **Table 1-1**)
- Interrupts and exceptions:
  - Non-maskable interrupts: 2 sources
  - Maskable interrupts: 55/59 sources (see **Table 1-1**)
  - Software exceptions: 32 sources
  - Exception trap: 2 sources
- I/O lines:
  - I/O ports: 84
- Timer/counter:
  - 16-bit interval timer M (TMM): 1 channel
  - 16-bit timer/event counter P (TMP): 6 channels
  - 16-bit timer/event counter Q (TMQ): 1 channel
- Real-time output port: 6 bits × 1 channel
- Watch timer: 1 channel
- Watchdog timer: 1 channel
- Serial interface (SIO):
  - Asynchronous serial interface A (UARTA)
  - 3-wire variable-length serial interface B (CSIB)
  - I^2C bus interface (I^2C)
    - UARTA/CSIB: 1 channel
    - UARTA/I^2C: 2 channels
    - CSIB/I^2C: 1 channel
    - CSIB: 3 channels
- IEBus controller: 1 channel (IEBus controller version only)
- CAN controller: 1 channel (CAN controller version only)
- A/D converter: 10-bit resolution: 12 channels
- D/A converter: 8-bit resolution: 2 channels
- DMA controller: 4 channels

- ROM correction: 4 correction addresses specifiable
- Clock generator: During main clock or subclock operation
  7-level CPU clock (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, f_{XT})
  Clock-through mode/PLL mode selectable
- Ring-OSC: 200 kHz (TYP.)
- Power-save functions: HALT/IDLE1/IDLE2/software STOP/subclock/sub-IDLE mode
- Package: 100-pin plastic QFP (14×20)
  100-pin plastic LQFP (fine pitch) (14×14)

1.3 Application Fields

Audio, car audio

1.4 Ordering Information

Part Number	Package	Internal ROM
μPD703260GF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703260GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703260YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703260YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703261GF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703261GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)
μPD703261YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703261YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)
μPD703262GF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703262GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703262YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703262YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703263GF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703263GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD703263YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703263YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD703270GF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703270GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703270YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703270YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703271GF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703271GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)
μPD703271YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703271YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)
μPD703272GF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703272GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703272YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703272YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703273GF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703273GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD703273YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703273YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD703280GF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703280GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703280YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703280YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703281GF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703281GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)

Remark xxx indicates ROM code suffix.

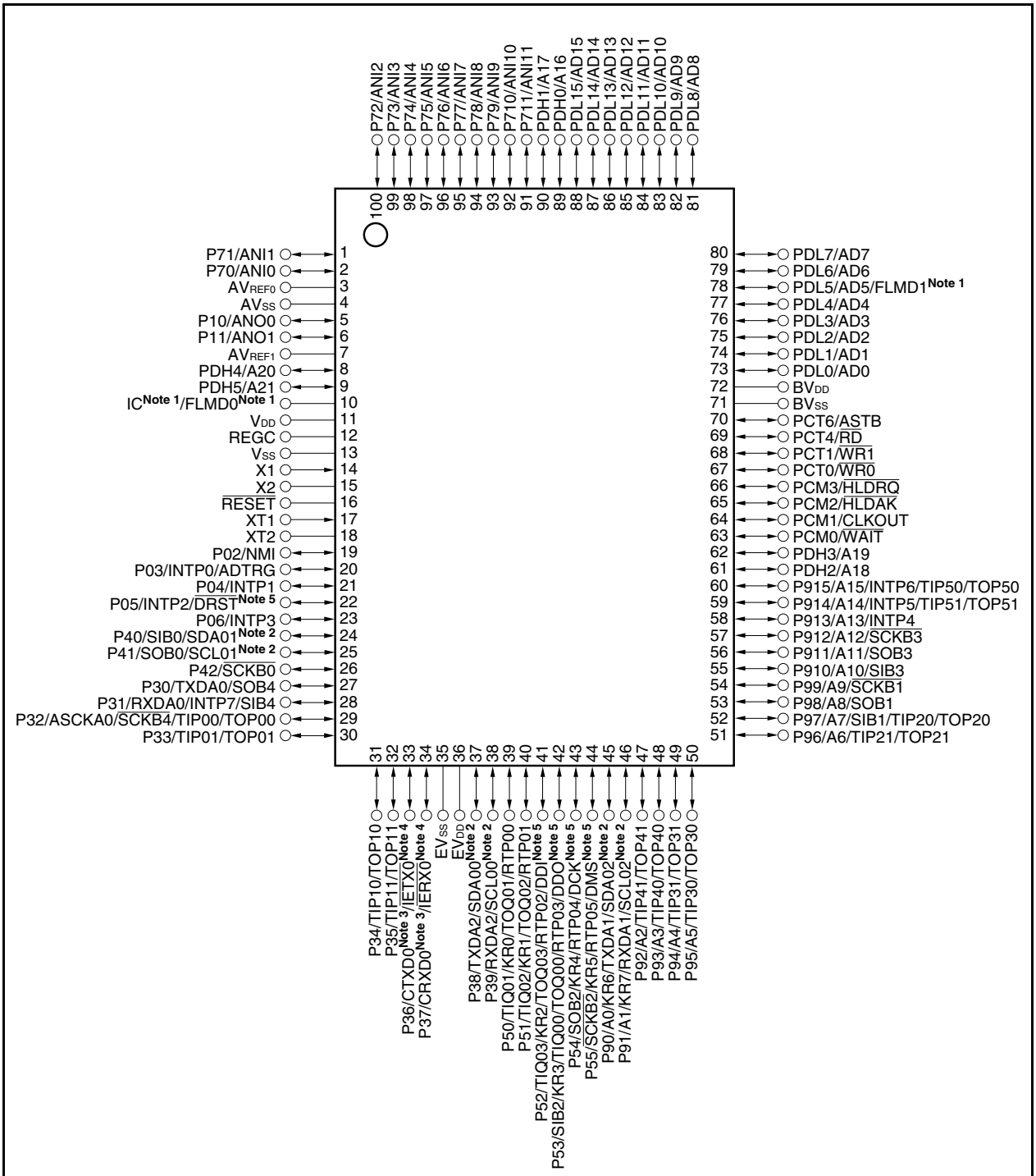
Part Number	Package	Internal ROM
μPD703281YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	384 KB (mask ROM)
μPD703281YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (mask ROM)
μPD703282GF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703282GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703282YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	512 KB (mask ROM)
μPD703282YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	512 KB (mask ROM)
μPD703283GF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703283GC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD703283YGF-xxx-3BA	100-pin plastic QFP (14 × 20)	640 KB (mask ROM)
μPD703283YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (mask ROM)
μPD70F3261GF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3261GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3261YGF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3261YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3263GF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3263GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)
μPD70F3263YGF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3263YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)
μPD70F3271GF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3271GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3271YGF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3271YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3273GF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3273GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)
μPD70F3273YGF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3273YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)
μPD70F3281GF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3281GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3281YGF-3BA	100-pin plastic QFP (14 × 20)	384 KB (flash memory)
μPD70F3281YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	384 KB (flash memory)
μPD70F3283GF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3283GC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)
μPD70F3283YGF-3BA	100-pin plastic QFP (14 × 20)	640 KB (flash memory)
μPD70F3283YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	640 KB (flash memory)

Remark xxx indicates ROM code suffix.

1.5 Pin Configuration (Top View)

100-pin plastic QFP (14 × 20)

μPD703260GF-xxx-3BA	μPD703270GF-xxx-3BA	μPD703280GF-xxx-3BA
μPD703260YGF-xxx-3BA	μPD703270YGF-xxx-3BA	μPD703280YGF-xxx-3BA
μPD703261GF-xxx-3BA	μPD703271GF-xxx-3BA	μPD703281GF-xxx-3BA
μPD703261YGF-xxx-3BA	μPD703271YGF-xxx-3BA	μPD703281YGF-xxx-3BA
μPD703262GF-xxx-3BA	μPD703272GF-xxx-3BA	μPD703282GF-xxx-3BA
μPD703262YGF-xxx-3BA	μPD703272YGF-xxx-3BA	μPD703282YGF-xxx-3BA
μPD703263GF-xxx-3BA	μPD703273GF-xxx-3BA	μPD703283GF-xxx-3BA
μPD703263YGF-xxx-3BA	μPD703273YGF-xxx-3BA	μPD703283YGF-xxx-3BA
μPD70F3261GF-3BA	μPD70F3271GF-3BA	μPD70F3281GF-3BA
μPD70F3261YGF-3BA	μPD70F3271YGF-3BA	μPD70F3281YGF-3BA
μPD70F3263GF-3BA	μPD70F3273GF-3BA	μPD70F3283GF-3BA
μPD70F3263YGF-3BA	μPD70F3273YGF-3BA	μPD70F3283YGF-3BA



Notes 1. IC: Directly connect this pin to V_{SS} (mask ROM version only).
FLMD0, FLMD1: Connect these pins to V_{SS} in the normal mode (flash memory version only).

2. SCL00 to SCL02 and SDA00 to SDA02 are valid only in the I^2C bus version (Y version).

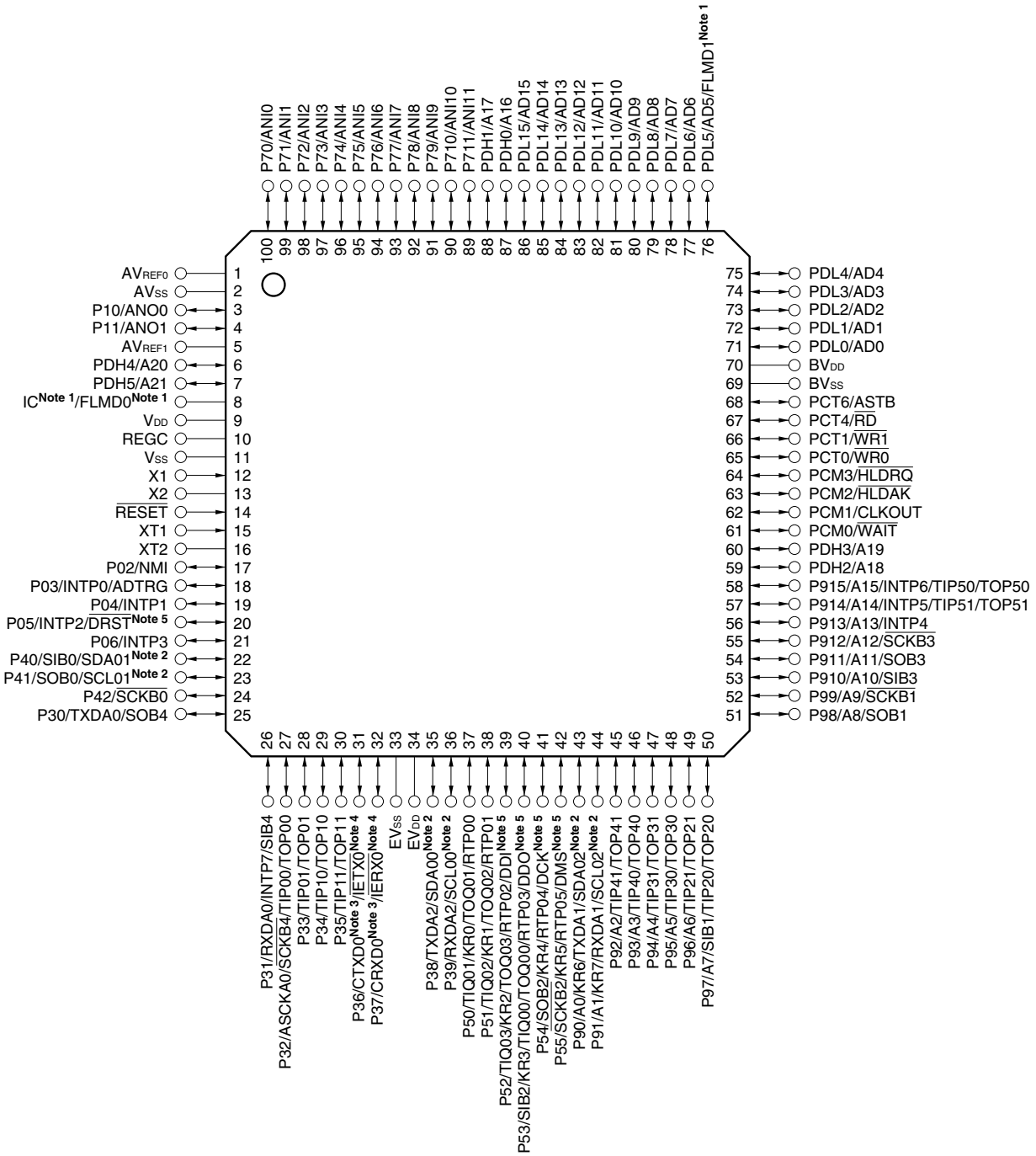
3. CTXD0 and CRXD0 are valid only in the CAN controller version.

4. $\overline{IETX0}$ and $\overline{IERX0}$ are valid only in the IEBus controller version.

5. $\overline{DRST}$, DDI, DDO, DCK, and DMS are valid only in the flash memory version.

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

μPD703260GC-xxx-8EU	μPD703270GC-xxx-8EU	μPD703280GC-xxx-8EU
μPD703260YGC-xxx-8EU	μPD703270YGC-xxx-8EU	μPD703280YGC-xxx-8EU
μPD703261GC-xxx-8EU	μPD703271GC-xxx-8EU	μPD703281GC-xxx-8EU
μPD703261YGC-xxx-8EU	μPD703271YGC-xxx-8EU	μPD703281YGC-xxx-8EU
μPD703262GC-xxx-8EU	μPD703272GC-xxx-8EU	μPD703282GC-xxx-8EU
μPD703262YGC-xxx-8EU	μPD703272YGC-xxx-8EU	μPD703282YGC-xxx-8EU
μPD703263GC-xxx-8EU	μPD703273GC-xxx-8EU	μPD703283GC-xxx-8EU
μPD703263YGC-xxx-8EU	μPD703273YGC-xxx-8EU	μPD703283YGC-xxx-8EU
μPD70F3261GC-8EU	μPD70F3271GC-8EU	μPD70F3281GC-8EU
μPD70F3261YGC-8EU	μPD70F3271YGC-8EU	μPD70F3281YGC-8EU
μPD70F3263GC-8EU	μPD70F3273GC-8EU	μPD70F3283GC-8EU
μPD70F3263YGC-8EU	μPD70F3273YGC-8EU	μPD70F3283YGC-8EU



Notes 1. IC: Directly connect this pin to V_{SS} (mask ROM version only).
FLMD0, FLMD1: Connect these pins to V_{SS} in the normal mode (flash memory version only).

2. SCL00 to SCL02 and SDA00 to SDA02 are valid only in the I^2C bus version (Y version).

3. CTXD0 and CRXD0 are valid only in the CAN controller version.

4. $\overline{\text{IETX0}}$ and $\overline{\text{IERX0}}$ are valid only in the IEBus controller version.

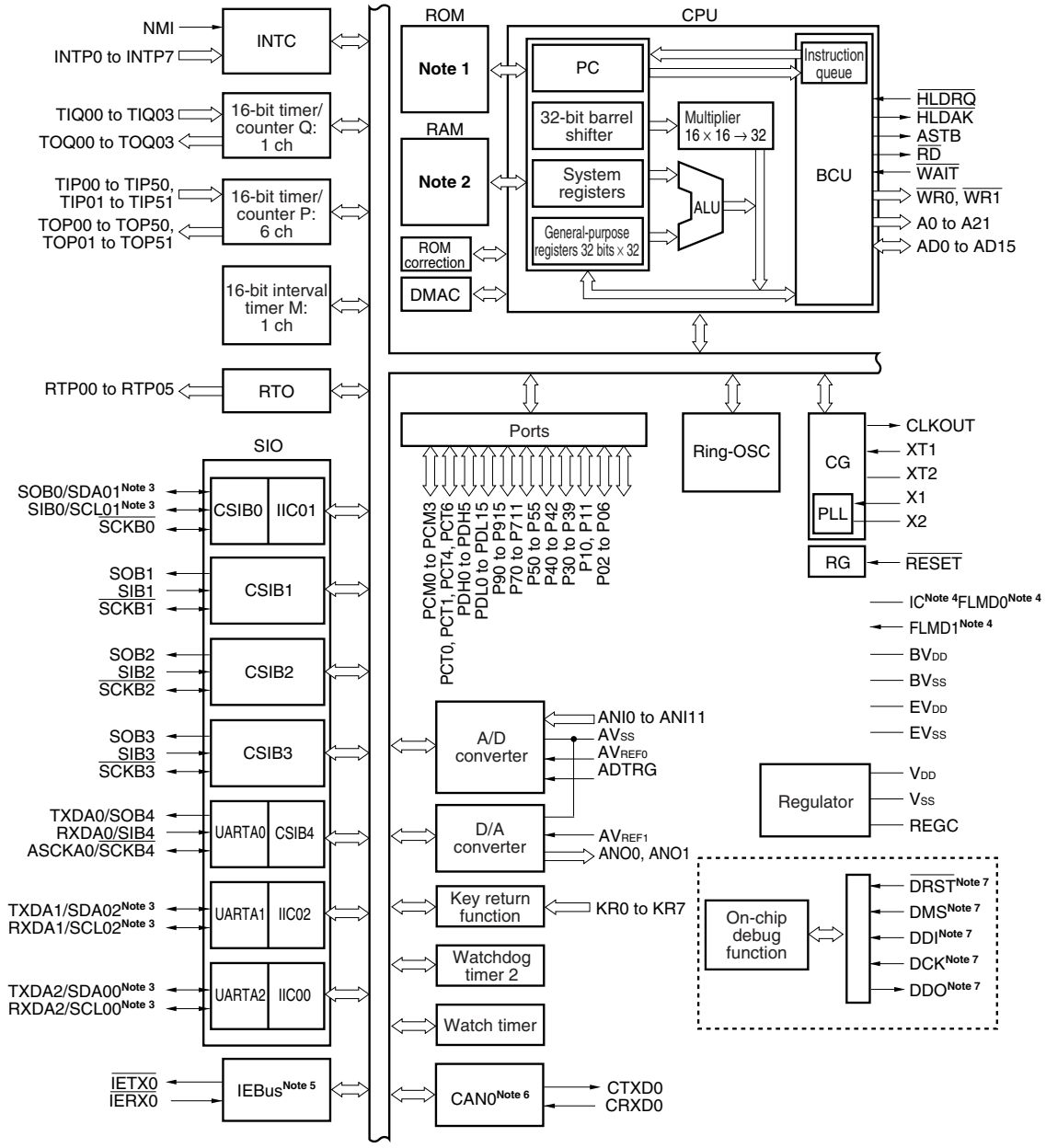
5. $\overline{\text{DRST}}$, DDI, DDO, DCK, and DMS are valid only in the flash memory version.

Pin names

A0 to A21:	Address bus	PCM0 to PCM3:	Port CM
AD0 to AD15:	Address/data bus	PCT0, PCT1,	
ADTRG:	A/D trigger input	PCT4, PCT6:	Port CT
ANI0 to ANI11:	Analog input	PDH0 to PDH5:	Port DH
ANO0, ANO1:	Analog output	PDL0 to PDL15:	Port DL
ASCKA0:	Asynchronous serial clock	$\overline{RD}$:	Read strobe
ASTB:	Address strobe	REGC:	Regulator control
AV_{REF0}, AV_{REF1}:	Analog reference voltage	$\overline{RESET}$:	Reset
AV_{SS}:	Analog V_{SS}	RTP00 to RTP05:	Real-time output port
BV_{DD}:	Power supply for bus interface	RXDA0 to RXDA2:	Receive data
BV_{SS}:	Ground for bus interface	$\overline{SCKB0}$ to $\overline{SCKB4}$:	Serial clock
CLKOUT:	Clock output	SCL00 to SCL02:	Serial clock
CRXD0:	CAN receive data	SDA00 to SDA02:	Serial data
CTXD0:	CAN transmit data	SIB0 to SIB4:	Serial input
DCK:	Debug clock	SOB0 to SOB4:	Serial output
DDI:	Debug data input	TIP00, TIP01,	
DDO:	Debug data output	TIP10, TIP11,	
DMS:	Debug mode select	TIP20, TIP21,	
$\overline{DRST}$:	Debug reset	TIP30, TIP31,	
EV_{DD}:	Power supply for port	TIP40, TIP41,	
EV_{SS}:	Ground for port	TIP50, TIP51,	
FLMD0, FLMD1:	Flash programming mode	TIQ00 to TIQ03:	Timer input
$\overline{HLDAK}$:	Hold acknowledge	TOP00, TOP01,	
$\overline{HLDRQ}$:	Hold request	TOP10, TOP11,	
IC:	Internally connected	TOP20, TOP21,	
$\overline{IERX0}$:	IEBus receive data	TOP30, TOP31,	
$\overline{IETX0}$:	IEBus transmit data	TOP40, TOP41,	
INTP0 to INTP7:	Interrupt request from peripherals	TOP50, TOP51,	
KR0 to KR7:	Key return	TOQ00 to TOQ03:	Timer output
NMI:	Non-maskable interrupt request	TXDA0 to TXDA2:	Transmit data
P02 to P06:	Port 0	V_{DD}:	Power supply
P10, P11:	Port 1	V_{SS}:	Ground
P30 to P39:	Port 3	$\overline{WAIT}$:	Wait
P40 to P42:	Port 4	$\overline{WR0}$:	Lower byte write strobe
P50 to P55:	Port 5	$\overline{WR1}$:	Upper byte write strobe
P70 to P711:	Port 7	X1, X2:	Crystal for main clock
P90 to P915:	Port 9	XT1, XT2:	Crystal for subclock

1.6 Function Block Configuration

1.6.1 Internal block diagram



Notes 1. 256/384/512/640 KB (mask ROM) (see **Table 1-1**)
384/640 KB (flash memory) (see **Table 1-1**)

2. 24/32/40/48 KB (see **Table 1-1**)

3. I^2C bus version (Y version) only

4. IC: Mask ROM version only
FLMD0, FLMD1: Flash memory version

5. IEBus controller version only

6. CAN controller version only

7. Flash memory version only

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 a multiplier (16 bits × 16 bits → 32 bits) and a barrel shifter (32 bits) contribute to faster 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 is a 640/512/384/256 KB mask ROM or flash memory mapped to addresses 0000000H to 009FFFFH/0000000H to 007FFFFH/0000000H to 005FFFFH/0000000H to 003FFFFH. It can be accessed from the CPU in one clock during instruction fetch.

(4) RAM

This is a 48/40/32/24 KB RAM mapped to addresses 3FF3000H to 3FFEFFFH/3FF5000H to 3FFEFFFH/3FF7000H to 3FFEFFFH/3FF9000H to 3FFEFFFH. It can be accessed from the CPU in one clock during data access.

(5) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP0 to INTP7) 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: one for the main clock (f_{XX}) and one for the subclock (f_X). It generates seven types of clocks (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, f_{XT}), and supplies one of them as the operating clock for the CPU (f_{CPU}).

(7) Ring-OSC

A ring oscillator (Ring-OSC) is provided on chip. The oscillation frequency is 200 kHz (TYP.). Ring-OSC supplies the clock for watchdog timer 2 and timer M.

(8) Timer/counter

Six-channel 16-bit timer/event counter P (TMP), one-channel 16-bit timer/event counter Q (TMQ), and one-channel 16-bit interval timer M (TMM), are provided on chip.

(9) Watch timer

This timer counts the reference time period (0.5 s) for counting the clock (the 32.768 kHz from the subclock or the 32.768 kHz f_{BRG} from prescaler 3). The watch timer can also be used as an interval timer for the main clock.

(10) Watchdog timer 2

A watchdog timer is provided on chip to detect inadvertent program loops, system abnormalities, etc.
Either the Ring-OSC, the main clock, or the subclock can be selected as the source clock.
Watchdog timer 2 generates a non-maskable interrupt request signal (INTWDT2) or a system reset signal (WDT2RES) after an overflow occurs.

(11) Serial interface (SIO)

The V850ES/SG2 includes three kinds of serial interfaces: asynchronous serial interface A (UARTA), 3-wire variable-length serial interface B (CSIB), and an I^2C bus interface (I^2C). These interfaces which can use up to seven channels at the same time. One of these channels is switchable between UARTA and CSIB, another two channels are switchable between UARTA and I^2C, and another one is switchable between CSIB and I^2C.
In the case of UARTA, data is transferred via the TXDA0 to TXDA2 pins and RXDA0 to RXDA2 pins.
In the case of CSIB, data is transferred via the SOB0 to SOB4 pins, SIB0 to SIB4 pins, and $\overline{\text{SCKB0}}$ to $\overline{\text{SCKB4}}$ pins.
In the case of I^2C, data is transferred via the SDA00 to SDA02 and SCL00 to SCL02 pins.
A dedicated baud rate generator is provided on chip for UARTA.

(12) IEBus controller

The IEBus controller is a small-scale digital data transmission system for transferring data between units.
The IEBus controller is provided only in the IEBus controller version (see **Table 1-1**).

(13) CAN controller

The CAN controller is a small-scale digital data transmission system for transferring data between units.
The CAN controller is provided only in the CAN controller version (see **Table 1-1**).

(14) 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.

(15) D/A converter

A two-channel, 8-bit-resolution D/A converter that uses the R-2R ladder method is provided on chip.

(16) DMA controller

A 4-channel DMA controller is provided on chip. 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.

(17) ROM correction

A ROM correction function that replaces part of a program in the mask ROM with a program in the internal RAM is provided. Up to four correction addresses can be specified.

(18) Key interrupt function

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to key input pins (8 channels).

(19) Real-time output function

The real-time output function transfers preset 6-bit data to output latches upon the occurrence of an external trigger signal or a timer compare register match signal.

(20) CRC function

A CRC operation circuit that generates 16-bit CRC (Cyclic Redundancy Check) code upon setting of 8-bit data is provided on chip.

(21) On-chip debug function

An on-chip debug function via an N-wire-type in-circuit emulator that uses the JTAG (Joint Test Action Group) communication specifications is provided. Switching between the normal port function and on-chip debugging function is done with the control pin input level and the on-chip debug mode setting register (OCDM).

On-chip debug function is provided only in the flash memory version.

(22) Port

There are general-purpose port functions and control pin functions, as listed below.

Port	I/O	Port Function	Control Function
P0	5-bit I/O	General-purpose port	NMI, external interrupt, A/D converter trigger, debug reset
P1	2-bit I/O		D/A converter analog output
P3	10-bit I/O		External interrupt, serial interface, timer I/O, CAN data I/O, IEBus data I/O
P4	3-bit I/O		Serial interface
P5	6-bit I/O		Timer I/O, real-time output, key interrupt input, serial interface
P7	12-bit I/O		A/D converter analog input
P9	16-bit I/O		External address bus, serial interface, key interrupt input, timer I/O, external interrupt
PCM	4-bit I/O		External bus interface
PCT	4-bit I/O		External bus interface
PDH	6-bit I/O		External address bus
PDL	16-bit I/O		External address/data bus

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

The names and functions of the pins of the V850ES/SG2 pin are described below.

There are three types of pin I/O buffer power supplies: AV_{REF0}, AV_{REF1}, BV_{DD}, and EV_{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 Pin
AV_{REF0}	Port 7
AV_{REF1}	Port 1
BV_{DD}	Port CM, port CT, port DH (bits 0 to 3), port DL
EV_{DD}	Port 0, port 3, port 4, port 5, port 9, port DH (bits 4, 5), $\overline{RESET}$

(1) Port pins

(1/3)

Pin Name	I/O	Function	Alternate Function
P02	I/O	Port 0 5-bit I/O port Input/output can be specified in 1-bit units.	NMI
P03			INTP0/ADTRG
P04			INTP1
P05			INTP2/$\overline{DRST}$Note 1
P06			INTP3
P10	I/O	Port 1 2-bit I/O port Input/output can be specified in 1-bit units.	ANO0
P11			ANO1
P30	I/O	Port 3 10-bit I/O port Input/output can be specified in 1-bit units.	TXDA0/SOB4
P31			RXDA0/INTP7/SIB4
P32			ASCKA0/$\overline{SCKB4}$/TIP00/TOP00
P33			TIP01/TOP01
P34			TIP10/TOP10
P35			TIP11/TOP11
P36			CTXD0Note 2/$\overline{IETX0}$Note 3
P37			CRXD0Note 2/$\overline{IERX0}$Note 3
P38			TXDA2/SDA00Note 4
P39			RXDA2/SCL00Note 4

Notes 1. Flash memory version only

2. CAN controller version only

3. IEBus controller version only

4. I^2C bus version (Y version) only

(2/3)

Pin Name	I/O	Function	Alternate Function
P40	I/O	Port 4 3-bit I/O port Input/output can be specified in 1-bit units.	SIB0/SDA01[Note 1]
P41			SOB0/SCL01[Note 1]
P42			$\overline{\text{SCKB0}}$
P50	I/O	Port 5 6-bit I/O port Input/output can be specified in 1-bit units.	TIQ01/KR0/TOQ01/RTP00
P51			TIQ02/KR1/TOQ02/RTP01
P52			TIQ03/KR2/TOQ03/RTP02/DDI[Note 2]
P53			SIB2/KR3/TIQ00/TOQ00/RTP03/DDO[Note 2]
P54			SOB2/KR4/RTP04/DCK[Note 2]
P55			$\overline{\text{SCKB2}}$/KR5/RTP05/DMS[Note 2]
P70	I/O	Port 7 12-bit I/O port Input/output can be specified in 1-bit units.	ANI0
P71			ANI1
P72			ANI2
P73			ANI3
P74			ANI4
P75			ANI5
P76			ANI6
P77			ANI7
P78			ANI8
P79			ANI9
P710			ANI10
P711			ANI11
P90	I/O	Port 9 16-bit I/O port Input/output can be specified in 1-bit units.	A0/KR6/TXDA1/SDA02[Note 1]
P91			A1/KR7/RXDA1/SCL02[Note 1]
P92			A2/TIP41/TOP41
P93			A3/TIP40/TOP40
P94			A4/TIP31/TOP31
P95			A5/TIP30/TOP30
P96			A6/TIP21/TOP21
P97			A7/SIB1/TIP20/TOP20
P98			A8/SOB1
P99			A9/$\overline{\text{SCKB1}}$
P910			A10/SIB3
P911			A11/SOB3
P912			A12/$\overline{\text{SCKB3}}$
P913			A13/INTP4
P914			A14/INTP5/TIP51/TOP51
P915			A15/INTP6/TIP50/TOP50

Notes 1. I^2C bus version (Y version) only

2. Flash memory version only

(3/3)

Pin Name	I/O	Function	Alternate Function
PCM0	I/O	Port CM 4-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{WAIT}}$
PCM1			CLKOUT
PCM2			$\overline{\text{HLDAK}}$
PCM3			$\overline{\text{HLDRQ}}$
PCT0	I/O	Port CT 4-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{WR0}}$
PCT1			$\overline{\text{WR1}}$
PCT4			$\overline{\text{RD}}$
PCT6			ASTB
PDH0	I/O	Port DH 6-bit I/O port Input/output can be specified in 1-bit units.	A16
PDH1			A17
PDH2			A18
PDH3			A19
PDH4			A20
PDH5			A21
PDL0	I/O	Port DL 16-bit I/O port Input/output can be specified in 1-bit units.	AD0
PDL1			AD1
PDL2			AD2
PDL3			AD3
PDL4			AD4
PDL5			AD5/FLMD1[Note]
PDL6			AD6
PDL7			AD7
PDL8			AD8
PDL9			AD9
PDL10			AD10
PDL11			AD11
PDL12			AD12
PDL13			AD13
PDL14			AD14
PDL15			AD15

Note Flash memory version only

(2) Non-port pins

(1/4)

Pin Name	I/O	Function	Alternate Function
A0	Output	Address bus for external memory (when using separate bus)	P90/KR6/TDXA1/SDA02[Note 1]
A1			P91/KR7/RXDA1/SCL02[Note 1]
A2			P92/TIP41/TOP41
A3			P93/TIP40/TOP40
A4			P94/TIP31/TOP31
A5			P95/TIP30/TOP30
A6			P96/TIP21/TOP21
A7			P97/SIB1/TIP20/TOP20
A8			P98/SOB1
A9			P99/$\overline{\text{SCKB1}}$
A10			P910/SIB3
A11			P911/SOB3
A12			P912/$\overline{\text{SCKB3}}$
A13			P913/INTP4
A14			P914/INTP5/TIP51/TOP51
A15			P915/INTP6/TIP50/TOP50
A16 to A21	Output	Address bus for external memory	PDH0 to PDH5
AD0 to AD4	I/O	Address bus/data bus for external memory	PDL0 to PDL4
AD5			PDL5/FLMD1[Note 2]
AD6 to AD15			PDL6 to PDL15
ADTRG	Input	A/D converter external trigger input	P03/INTP0
ANI0	Input	Analog voltage input for A/D converter	P70
ANI1			P71
ANI2			P72
ANI3			P73
ANI4			P74
ANI5			P75
ANI6			P76
ANI7			P77
ANI8			P78
ANI9			P79
ANI10			P710
ANI11			P711
ANO0	Output	Analog voltage output for D/A converter	P10
ANO1			P11
ASCKA0	Input	UARTA0 baud rate clock input	P32/$\overline{\text{SCKB4}}$/TIP00/TOP00
ASTB	Output	Address strobe signal output for external memory	PCT6
AV_{REF0}	–	Reference voltage input for A/D converter (same potential as V_{DD})	–
AV_{REF1}		Reference voltage input for D/A converter (same potential as V_{DD})	–
AV_{SS}	–	Ground potential for A/D and D/A converters (same potential as V_{SS})	–

Notes 1. I^2C bus version (Y version) only

2. Flash memory version only

(2/4)

Pin Name	I/O	Function	Alternate Function
BV_{DD}	–	Positive power supply for bus interface and alternate-function ports	–
BV_{SS}	–	Ground potential for bus interface and alternate-function ports	–
CLKOUT	Output	Internal system clock output	PCM1
CRXD0[Note 1]	Input	CAN receive data input	P37/$\overline{IERX0}$[Note 2]
CTXD0[Note 1]	Output	CAN transmit data output	P36/$\overline{IETX0}$[Note 2]
DCK[Note 3]	Input	Debug clock input	P54/SOB2/KR4/RTP04
DDI[Note 3]	Input	Debug data input	P52/TIQ03/KR2/TOQ03/RTP02
DDO[Note 3]	Output	Debug data output	P53/SIB2/KR3/TIQ00/TOQ00/ RTP03
DMS[Note 3]	Input	Debug mode select	P55/$\overline{SCKB2}$/KR5/RTP05
$\overline{DRST}$[Note 3]	Input	Debug reset input	P05/INTP2
EV_{DD}	–	Positive power supply for external (same potential as V_{DD})	–
EV_{SS}	–	Ground potential for external (same potential as V_{SS})	–
FLMD0[Note 3]	Input	Flash programming mode setting pin	–
FLMD1[Note 3]			PDL5/AD5
$\overline{HLDAK}$	Output	Bus hold acknowledge output	PCM2
$\overline{HLDRQ}$	Input	Bus hold request input	PCM3
IC[Note 4]	–	Internally connected	–
$\overline{IERX0}$[Note 2]	Input	IEBus receive data input	P37/CRXD0[Note 1]
$\overline{IETX0}$[Note 2]	Output	IEBus transmit data output	P36/CTXD0[Note 1]
INTP0	Input	External interrupt request input (maskable, analog noise elimination[Note 5])	P03/ADTRG
INTP1			P04
INTP2			P05/$\overline{DRST}$[Note 3]
INTP3			P06
INTP4			P913/A13
INTP5			P914/A14/TIP51/TOP51
INTP6			P915/A15/TIP50/TOP50
INTP7			P31/RXDA0/SIB4
KR0	Input	Key interrupt input	P50/TIQ01/TOQ01/RTP00
KR1			P51/TIQ02/TOQ02/RTP01
KR2			P52/TIQ03/TOQ03/RTP02/DDI[Note 3]
KR3			P53/SIB2/TIQ00/TOQ00/RTP03/ DDO[Note 3]
KR4			P54/SOB2/RTP04/DCK[Note 3]
KR5			P55/$\overline{SCKB2}$/RTP05/DMS[Note 3]
KR6			P90/A0/TXDA1/SDA02[Note 6]
KR7			P91/A1/RXDA1/SCL02[Note 6]

Notes 1. CAN controller version only

2. IEBus controller version only

3. Flash memory version only

4. Mask ROM version only

5. An analog noise elimination function or digital noise elimination function is selectable for the INTP3 pin.

6. I^2C bus version (Y version) only

(3/4)

Pin Name	I/O	Function	Alternate Function
NMI	Input	External interrupt input (non-maskable, analog noise elimination)	P02
$\overline{RD}$	Output	Read strobe signal output for external memory	PCT4
REGC	–	Connection of regulator output stabilization capacitance	–
$\overline{RESET}$	Input	System reset input	–
RTP00	Output	Real-time output port	P50/TIQ01/KR0/TOQ01
RTP01			P51/TIQ02/KR1/TOQ02
RTP02			P52/TIQ03/KR2/TOQ03/DDI[Note 1]
RTP03			P53/SIB2/KR3/TIQ00/TOQ00/ DDO[Note 1]
RTP04			P54/SOB2/KR4/DCK[Note 1]
RTP05			P55/$\overline{SCKB2}$/KR5/DMS[Note 1]
RXDA0	Input	Serial receive data input (UARTA0)	P31/INTP7/SIB4
RXDA1		Serial receive data input (UARTA1)	P91/A1/KR7/SCL02[Note 2]
RXDA2		Serial receive data input (UARTA2)	P39/SCL00[Note 2]
$\overline{SCKB0}$	I/O	Serial clock I/O (CSIB0)	P42
$\overline{SCKB1}$		Serial clock I/O (CSIB1)	P99/A9
$\overline{SCKB2}$		Serial clock I/O (CSIB2)	P55/KR5/RTP05/DMS[Note 1]
$\overline{SCKB3}$		Serial clock I/O (CSIB3)	P912/A12
$\overline{SCKB4}$		Serial clock I/O (CSIB4)	P32/ASCKA0/TIP00/TOP00
SCL00[Note 2]	I/O	Serial clock I/O (I^2C00)	P39/RXDA2
SCL01[Note 2]		Serial clock I/O (I^2C01)	P41/SOB0
SCL02[Note 2]		Serial clock I/O (I^2C02)	P91/A1/KR7/RXDA1
SDA00[Note 2]	I/O	Serial transmit/receive data I/O (I^2C00)	P38/TXDA2
SDA01[Note 2]		Serial transmit/receive data I/O (I^2C01)	P40/SIB0
SDA02[Note 2]		Serial transmit/receive data I/O (I^2C02)	P90/A0/KR6/TXDA1
SIB0	Input	Serial receive data input (CSIB0)	P40/SDA01[Note 2]
SIB1		Serial receive data input (CSIB1)	P97/A7/TIP20/TOP20
SIB2		Serial receive data input (CSIB2)	P53/KR3/TIQ00/TOQ00/RTP03/ DDO[Note 1]
SIB3		Serial receive data input (CSIB3)	P910/A10
SIB4		Serial receive data input (CSIB4)	P31/RXDA0/INTP7
SOB0	Output	Serial transmit data output (CSIB0)	P41/SCL01[Note 2]
SOB1		Serial transmit data output (CSIB1)	P98/A8
SOB2		Serial transmit data output (CSIB2)	P54/KR4/RTP04/DCK[Note 1]
SOB3		Serial transmit data output (CSIB3)	P911/A11
SOB4		Serial transmit data output (CSIB4)	P30/TXDA0
TIP00	Input	External event/clock input (TMP0)	P32/ASCKA0/$\overline{SCKB4}$/TOP00
TIP01		External event/clock input (TMP0)	P33/TOP01
TIP10		External event/clock input (TMP1)	P34/TOP10
TIP11		External event/clock input (TMP1)	P35/TOP11
TIP20		External event/clock input (TMP2)	P97/A7/SIB1/TOP20
TIP21		External event/clock input (TMP2)	P96/A6/TOP21

Notes 1. Flash memory version only

2. I^2C bus version (Y version) only

(4/4)

Pin Name	I/O	Function	Alternate Function
TIP30	Input	External event/clock input (TMP3)	P95/A5/TOP30
TIP31		External event/clock input (TMP3)	P94/A4/TOP31
TIP40		External event/clock input (TMP4)	P93/A3/TOP40
TIP41		External event/clock input (TMP4)	P92/A2/TOP41
TIP50		External event/clock input (TMP5)	P915/A15/INTP6/TOP50
TIP51		External event/clock input (TMP5)	P914/A14/INTP5/TOP51
TIQ00	Input	External event/clock input (TMQ0)	P53/SIB2/KR3/TOQ00/RTP03 /DDO[Note 1]
TIQ01		External event/clock input (TMQ0)	P50/KR0/TOQ01/RTP00
TIQ02		External event/clock input (TMQ0)	P51/KR1/TOQ02/RTP01
TIQ03		External event/clock input (TMQ0)	P52/KR2/TOQ03/RTP02/DDI[Note 1]
TOP00	Output	Timer output (TMP0)	P32/ASCKA0/$\overline{\text{SCKB4}}$/TIP00
TOP01		Timer output (TMP0)	P33/TIP01
TOP10		Timer output (TMP1)	P34/TIP10
TOP11		Timer output (TMP1)	P35/TIP11
TOP20		Timer output (TMP2)	P97/A7/SIB1/TIP20
TOP21		Timer output (TMP2)	P96/A6/TIP21
TOP30		Timer output (TMP3)	P95/A5/TIP30
TOP31		Timer output (TMP3)	P94/A4/TIP31
TOP40		Timer output (TMP4)	P93/A3/TIP40
TOP41		Timer output (TMP4)	P92/A2/TIP41
TOP50		Timer output (TMP5)	P915/A15/INTP6/TIP50
TOP51		Timer output (TMP5)	P914/A14/INTP5/TIP51
TOQ00	Output	Timer output (TMQ0)	P53/SIB2/KR3/TIQ00/RTP03/ DDO[Note 1]
TOQ01		Timer output (TMQ0)	P50/TIQ01/KR0/RTP00
TOQ02		Timer output (TMQ0)	P51/RTP01/KR1/TIQ02
TOQ03		Timer output (TMQ0)	P52/TIQ03/KR2/RTP02/DDI[Note 1]
TXDA0	Output	Serial transmit data output (UARTA0)	P30/SOB4
TXDA1		Serial transmit data output (UARTA1)	P90/A0/KR6/SDA02[Note 2]
TXDA2		Serial transmit data output (UARTA2)	P38/SDA00[Note 2]
V_{DD}	–	Positive power supply for internal	–
V_{SS}	–	Ground potential for internal	–
$\overline{\text{WAIT}}$	Input	External wait input	PCM0
$\overline{\text{WR0}}$	Output	Write strobe for external memory (lower 8-bits)	PCT0
$\overline{\text{WR1}}$		Write strove for external memory (higher 8 bits)	PCT1
X1	Input	Connection of resonator for main clock	–
X2	–		–
XT1	Input	Connection of resonator for subclock	–
XT2	–		–

Notes 1. Flash memory version only

2. I^2C bus version (Y version) only

2.2 Pin States

The operation states of pins in the various modes are described below.

Table 2-2. Pin Operation States in Various Modes

Bus Control Pin	Reset	HALT Mode During DMA Transfer	IDLE1, IDLE2 Mode, Software STOP Mode	Idle State[Note 2]	Bus Hold
AD0 to AD15	Hi-Z[Note 1]	Operating	Hi-Z	Held	Hi-Z
A0 to A21					
$\overline{WAIT}$			–	–	–
CLKOUT			L	Operating	Operating
$\overline{WR0}$, $\overline{WR1}$			H	H	Hi-Z
$\overline{RD}$					
ASTB					
$\overline{HLDAK}$					L
$\overline{HLDRQ}$			–	–	Operating

Notes 1. The bus control pin is shared with a port pin, so it is initialized to the input mode (port mode).

2. The state of the pins in the idle state inserted following the T3 state is shown.

Remark Hi-Z: High impedance
Held: The state during the immediately preceding external bus cycle is held.
L: Low-level output
H: High-level output
–: Input without sampling (not acknowledged)

2.3 Description of Pin Functions

(1) P02 to P05 (Port 0) … 3-state I/O

P02 to P05 function as a 5-bit I/O port for which input and output can be specified in 1-bit units.

In addition to I/O port pins, these pins can also be used as an NMI input, external interrupt request signal inputs, the external trigger for the A/D converter, and debug reset input. The port or control mode can be selected for each bit, and a pin's valid edge is specified by the INTR0 and INTF0 registers.

Normal output and N-ch open-drain output can be selected for P02 to P05.

(a) Port mode

P02 to P05 can be set to input or output in 1-bit units using port mode register 0 (PM0).

(b) Control mode

(i) NMI (non-maskable interrupt request) … Input

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

(ii) INTP0 to INTP3 (interrupt request from peripherals) … Input

These are external interrupt request signal input pins.

(iii) ADTRG (A/D trigger input) … Input

This is the A/D converter's external trigger input pin. This pin is controlled by A/D converter mode register 0 (ADA0M0).

(iv) $\overline{\text{DRST}}$ (debug reset) … Input

This is the debug reset input pin. This is a negative logic signal that initializes the on-chip debug circuit asynchronously. When set to low level, it resets/disables the on-chip debug circuit. When not using the debug function, set this pin to low level (DRST is valid only in the flash memory version).

(2) P10, P11 (Port 1) … 3-state I/O

P10 and P11 function as a 2-bit I/O port for which input and output can be specified in 1-bit units.

In addition to I/O pins, these pins can also be used as the analog output pins for the A/D converter in the control mode. When using these pins as analog output pins, set them in the input mode. At this time, do not read the port.

(a) Port mode

P10 and P11 can be set to input or output in 1-bit units using port mode register 1 (PM1).

(b) Control mode

(i) ANO0, ANO1 (analog output) … Output

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

(3) P30 to P39 (port 3) … 3-state I/O

P30 to P39 function as a 10-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O port pins, these pins can also be used as external interrupt request signal inputs, serial interface I/O, timer/counter I/O, CAN data I/O, and IEBus data I/O. The port or control mode can be selected for each bit, and the valid edge for P31 is specified by the INTR3 and INTF3 registers.

Normal output and N-ch open-drain output can be selected for P30 to P39.

(a) Port mode

P30 to P39 can be set to input or output in 1-bit units using port mode register 3 (PM3).

(b) Control mode

(i) SIB4 (serial input 1) … Input
This is the serial receive data input pin for CSIB4.

(ii) SOB4 (serial output) … Output
This is the serial transmit data output pin for CSIB4.

(iii) $\overline{\text{SCKB4}}$ (serial clock) … 3-state I/O
This is the serial clock I/O pin for CSIB4.

(iv) RXDA0, RXDA2 (receive data) … Input
These are the serial receive data input pins for UARTA0 and UARTA2.

(v) TXDA0, TXDA2 (transmit data) … Output
These are the serial transmit data output pins for UARTA0 and UARTA2.

(vi) ASCKA0 (asynchronous serial clock) … Input
This is the serial baud rate input pin for UARTA0.

(vii) INTP7 (interrupt request from peripherals) … Input
This is the external interrupt request signal input pin.

(viii) TIP00, TIP01, TIP10, TIP11 (timer input) … Input
These are the external count clock input pins for timers P0 and P1.

(ix) TOP00, TOP01, TOP10, TOP11 (timer output) … Output
These are the pulse signal output pins for timers P0 and P1.

(x) SDA00 (serial data) … Input
This is the serial transmit/receive data I/O pin for I^2C00 (SDA00 is valid only in the I^2C bus version (Y version)).

(xi) SCL00 (serial clock) … I/O
This is the serial clock I/O pin for I^2C00 (SCL00 is valid only in the I^2C bus version (Y version)).

(xii) CRXD0 (CAN receive data) … Input
This is the receive data input pin for CAN0 (CRXD0 is valid only in the CAN controller version).

(xiii) CTXD0 (CAN transmit data) … Output
This is the transmit data output pin for CAN0 (CTXD0 is valid only in the CAN controller version).

(xiv) $\overline{\text{IERX0}}$ (IEBus receive data) … Input
This is the receive data input pin for IEBus ($\overline{\text{IERX0}}$ is valid only in the IEBus controller version).

(xv) $\overline{\text{IETX0}}$ (IEBus transmit data) … Output
This is the transmit data output pin for IEBus ($\overline{\text{IETX0}}$ is valid only in the IEBus controller version).

(4) P40 to P42 (port 4) … 3-state I/O

P40 to P42 function as a 3-bit I/O port for which input and output can be set in 1-bit units.
In addition to I/O port pins, these pins can also be used as serial interface I/O. The port or control mode can be selected for each bit.
Normal output and N-ch open-drain output can be selected for P40 to P42.

(a) Port mode

P40 to P42 can be set to input or output in 1-bit units using port mode register 4 (PM4).

(b) Control mode

(i) SIB0 (serial input) … Input
This is the serial receive data input pin for CSIB0.

(ii) SOB0 (serial output) … Output
This is the serial transmit data output pin for CSIB0.

(iii) $\overline{\text{SCKB0}}$ (serial clock) … 3-state I/O
This is the serial clock I/O pin for CSIB0.

(iv) SDA01 (serial data) … I/O
This is the serial transmit/receive data I/O pin for I^2C00 (SDA01 is valid only in the I^2C bus version (Y version)).

(v) SCL01 (serial clock) … I/O
This is the serial clock I/O pin for I^2C00 (SCL01 is valid only in the I^2C bus version (Y version)).

(5) P50 to P55 (port 5) … 3-state I/O

P50 to P55 function as a 6-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O port pins, these pins can also be used as serial interface I/O, timer/counter I/O, real-time output, debug function I/O, and key interrupt input function. The port or control mode can be selected for each bit.

Normal output and N-ch open-drain output can be selected for P50 to P55.

(a) Port mode

P50 to P55 can be set to input or output in 1-bit units using port mode register 5 (PM5).

(b) Control mode

(i) SIB2 (serial input) … Input
This is the serial receive data input pin for CSIB2.

(ii) SOB2 (serial output) … Output
This is the serial transmit data output pin for CSIB2.

(iii) $\overline{\text{SCKB2}}$ (serial clock) … 3-state I/O
This is the serial clock I/O pin for CSIB2.

(iv) RTP00 to RTP05 (real-time output port) … Output
These are real-time output port.

(v) KR0 to KR5 (key return) … Input
These are the key interrupt input pins. In the input port mode, the operation is specified by the key return mode register (KRM).

(vi) TIQ00, TIQ01, TIQ02, TIQ03 (timer input) … Input
These are the external count clock input pins for timer Q0.

(vii) TOQ00, TOQ01, TOQ02, TOQ03 (timer output) … Output
These are the pulse signal output pins for timer Q0.

(viii) DDI (debug data input) … Input
This is the debug data input pin for the on-chip debug circuit.

(ix) DDO (debug data output) … Output
This is the debug data output pin for the on-chip debug circuit.

(x) DCK (debug clock input) … Input
This is the debug clock input pin for the on-chip debug circuit.

(xi) DMS (debug mode select) … Input
This is the debug mode select pin for the on-chip debug circuit.

(6) P70 to P711 (port 7) … 3-state I/O

P70 to P711 function as a 12-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O port pins, these pins can also be used as analog input pins for the A/D converter in the control mode. When using these pins as analog output pins, set them in the input mode. At this time, do not read the port.

(a) Port mode

P70 to P711 can be set to input or output in 1-bit units using port mode register 7 (PM7).

(b) Control mode

P70 to P711 function alternatively with ANI0 to ANI11.

(i) ANI0 to ANI11 (analog input) … Input
These are the analog input pins for the A/D converter.

(7) P90 to P915 (port 9) … 3-state I/O

P90 to P915 function as a 16-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O port pins, these pins can also be used as serial interface I/O, timer/counter I/O, an address bus when externally expanding memory, external interrupt request signal inputs, and the key interrupt input function. The port or control mode can be selected for each bit.

Normal output and N-ch open-drain output can be selected for P90 to P915.

(a) Port mode

P90 to P915 can be set to input or output in 1-bit units using port mode register 9 (PM9) (to use these pins as the A0 to A15 pins, the mode must be changed in 16-bit units).

(b) Control mode

(i) SIB1, SIB3 (serial input) … Input
These are the serial receive data input pins for CSIB1 and CSIB3.

(ii) SOB1, SOB3 (serial output) … Output
These are the serial transmit data output pins for CSIB1 and CSIB3.

(iii) $\overline{\text{SCKB1}}$, $\overline{\text{SCKB3}}$ (serial clock) … 3-state I/O
These are the serial clock I/O pins for CSIB1 and CSIB3.

(iv) RXDA1 (receive data) … Input
This is the serial receive data input pin for UARTA1.

(v) TXDA1 (transmit data) … Output
This is the serial transmit data output pin for UARTA1.

(vi) TIP20, TIP21, TIP30, TIP31, TIP40, TIP41, TIP50, TIP51 (timer input) … Input
These are the external count clock input pins for timers P2, P3, P4, and P5.

(vii) TOP20, TOP21, TOP30, TOP31, TOP40, TOP41, TOP50, TOP51 (timer output) … Output
These are the pulse signal output pins for timers P2, P3, P4, and P5.

(viii) A0 to A15 (address bus) … Output

These are 16-bit address output pins used during external access. The output changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. When the bus cycle becomes inactive, these pins hold the address of the immediately preceding bus cycle.

(ix) INTP4 to INTP6 (interrupt request from peripherals) … Input

These are external interrupt request signal input pins.

(iix) KR6, KR7 (key return) … Input

These are key interrupt input pins. The operation is specified by the key return mode register (KRM) in the input port mode.

(xi) SDA02 (serial data) … I/O

This is the serial transmit/receive data I/O pin for I^2C02 (SDA02 is valid only in the I^2C bus version (Y version)).

(xii) SCL02 (serial clock) … I/O

This is the serial clock data I/O pin for I^2C02 (SCL02 is valid only in the I^2C bus version (Y version)).

(8) PCM0 to PCM3 (port CM) … 3-state I/O

PCM0 to PCM3 function as a 4-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O pins, these pins can also be used as the bus hold signal I/O in the control mode, bus clock output, and the control signal ($\overline{\text{WAIT}}$) that inserts waits into the bus cycle.

(a) Port mode

PCM0 to PCM3 can be set to input or output in 1-bit units using port mode register CM (PMCM).

(b) Control mode

(i) $\overline{\text{HLDAK}}$ (hold acknowledge) … Output

This is an output pin for the acknowledge signal that indicates the high-impedance status for the address bus, data bus, and control bus when the V850ES/SG2 receives a bus hold request.

The address bus, data bus, and control bus are high impedance while this signal is active.

(ii) $\overline{\text{HLDRQ}}$ (hold request) … Input

This is an input pin by which an external device requests the V850ES/SG2 to release the address bus, data bus, and control bus release requests. This pin accepts asynchronous input for CLKOUT. When this pin is made active, the V850ES/SG2 sets the address bus, data bus, and control bus to high impedance upon the end of the bus cycle currently being executed, or immediately if no bus cycle is being executed, and the $\overline{\text{HLDAK}}$ signal is then made active and the bus is released.

(iii) CLKOUT (clock output) … Output

This pin outputs internally generated bus clocks.

(iv) $\overline{\text{WAIT}}$ (wait) … Input

This is a control signal input pin that inserts data wait states in the bus cycle. Data can be input to this pin asynchronous to the CLKOUT signal. In the multiplexed mode, this pin is sampled at the falling edge of the CLKOUT signal in the T2 and TW states of the bus cycle. In the separate mode, it is sampled at the rising edge of the CLKOUT signal immediately after the T1 and TW states of the bus cycle. A wait state may not be inserted if the setup/hold time of the sampling timing is not satisfied.

On/off switching of the wait function is performed using port mode control register CM (PMCCM).

(9) PCT0, PCT1, PCT4, PCT6 (port CT) … 3-state I/O

PCT0, PCT1, PCT4, and PCT6 function as a 4-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O pins, these pins can also be used as control signal output pins for external memory expansion in the control mode.

(a) Port mode

PCT0, PCT1, PCT4, and PCT6 can be set to input or output in 1-bit units using port mode register CT (PMCT).

(b) Control mode

(i) $\overline{\text{WR0}}$ (lower byte write strobe) … Output

This is the write strobe signal output pin for the lower data of the external 16-bit data bus.

(ii) $\overline{\text{WR1}}$ (upper byte write strobe) … Output

This is the write strobe signal output pin for the higher data of the external 16-bit data bus.

(iii) $\overline{\text{RD}}$ (read strobe) … Output

This is the read strobe signal output pin for the external 16-bit data bus.

(iv) ASTB (address strobe) … Output

This is the output pin for the latch strobe signal for the external address bus. Output becomes low level in synchronization with the falling edge of the clock during the T1 state of the bus cycle, and becomes high level in synchronization with the falling edge of the clock during the T3 state of the bus cycle. Output becomes high level when the bus cycle is inactive.

(10) PDH0 to PDH5 (port DH) … 3-state I/O

PDH0 to PDH5 function as a 6-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O port pins, these pins can also be used as an address bus during external memory expansion in the control mode.

(a) Port mode

PDH0 to PDH5 can be set to input or output in 1-bit units using port mode register DH (PMDH).

(b) Control mode

(i) A16 to A21 (address bus) … Output

These are 6-bit address output pins used by the address bus during external access. The output changes in synchronization with the rising edge of the clock during the T1 state of the bus cycle. When the bus cycle becomes inactive, these pins hold the address of the immediately preceding bus cycle.

(11) PDL0 to PDL15 (port DL) … 3-state I/O

PDL0 to PDL15 function as a 16-bit I/O port for which input and output can be set in 1-bit units.

In addition to I/O pins, these pins can also be used as a time division address/data bus (AD0 to AD15) during external memory expansion.

Moreover, during flash memory programming (input of high level to FLMD0), PDL5/AD5 function as the FLMD1 pin. At this time, be sure to input a low level to the FLMD1 pin.

(a) Port mode

PDL0 to PDL15 can be set to input or output in 1-bit units using port mode register DL (PMDL).

(b) Control mode

(i) AD0 to AD15 (address/data bus) … 3-state I/O

These form an address/data multiplexed bus during external access. In the multiplexed bus mode, they function as address output or data I/O, and in the separate bus mode, they function as data I/O.

(12) $\overline{\text{RESET}}$ (reset) … Input

$\overline{\text{RESET}}$ is a signal that is input asynchronously and has a low level width regardless of the status of the operating clock. When this signal is input, a system reset is executed with a higher priority than all other operations.

In addition to being used for ordinary initializations/start operations, this signal can also be used to release a standby mode (HALT, IDLE1, IDLE2, software STOP).

(13) X1, X2 (crystal for main clock)

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

(14) XT1, XT2 (crystal for subclock)

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

(15) AV_{SS} (ground for analog)

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

(16) AV_{REF0} (analog reference voltage) … Input

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

It is also used to supply a reference voltage to the A/D converter.

(17) AV_{REF1} (analog reference voltage) … Input

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

It is also used to supply a reference voltage to the D/A converter.

(18) EV_{DD} (power supply for port)

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

(19) EV_{SS} (ground for port)

This is the ground pin for I/O ports and alternate-function pins.

(20) V_{DD} (power supply)

This is the positive power supply pin. Connect all the V_{DD} pins to a positive power supply.

(21) V_{SS} (ground)

This is the ground pin. Connect all the V_{SS} pins to ground.

(22) FLMD0, FLMD1 (flash programming mode)

These are the positive power supply pins for the flash memory programming mode.

In the normal operation mode, connect these pins to V_{SS}.

(23) BV_{DD} (power supply for bus interface)

This is the positive power supply pin for the bus interface.

(24) BV_{SS} (ground for bus interface)

This is the ground pin for the bus interface.

(25) IC (internally connect)

This is an internally connected pin. In the normal operation mode, directly connect this pin to V_{SS}.

(26) REGC (regulator control) … Input

This is the capacitor pin for the regulator.

2.4 Pin I/O Circuit Types, I/O Buffer Power Supplies and Handling of Unused Pins

(1/3)

Pin	Alternate Function	I/O Circuit Type	Recommended Connection
P02	NMI	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P03	INTP0/ADTRG		
P04	INTP1		
P05	INTP2/$\overline{\text{DRST}}$[Note 1]	10-N	Input: Independently connect to EV_{SS} via a resistor. Fixing to V_{DD} level is prohibited. Output: Leave open. Internally pull-down after reset.
P06	INTP3	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P10	ANO0	12-D	Input: Independently connect to AV_{REF1} or AV_{SS} via a resistor. Output: Leave open.
P11	ANO1		
P30	TXDA0/SOB4	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P31	RXDA0/INTP7/SIB4		
P32	ASCKA0/$\overline{\text{SCKB4}}$/TIP00		
P33	TIP01/TOP01		
P34	TIP10/TOP10		
P35	TIP11/TOP11		
P36	CTXD0[Note 2]/$\overline{\text{IETX0}}$[Note 3]		
P37	CRXD0[Note 2]/$\overline{\text{IERX0}}$[Note 3]		
P38	TXDA2/SDA00[Note 4]		
P39	RXDA2/SCL00[Note 4]		
P40	SIB0/SDA01[Note 4]	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P41	SOB0/SCL01[Note 4]		
P42	$\overline{\text{SCKB0}}$		
P50	TIQ01/KR0/TOQ01/RTP00	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P51	TIQ02/KR1/TOQ02/RTP01		
P52	TIQ03/KR2/TOQ03/RTP02/DDI[Note 1]		
P53	SIB2/KR3/TIQ00/TOQ00/RTP03/DDO[Note 1]		
P54	SOB2/KR4/RTP04/DCK[Note 1]		
P55	$\overline{\text{SCKB2}}$/KR5/RTP05/DMS[Note 1]		

Notes 1. Flash memory version only

2. CAN controller version only

3. IEBus controller version only

4. I^2C version (Y version) only

(2/3)

Pin	Alternate Function	I/O Circuit Type	Recommended Connection
P70 to P711	ANI0 to ANI11	11-F	Input: Independently connect to AV_{REF0} or AV_{SS} via a resistor. Output: Leave open.
P90	A0/KR6/TDXA1/SDA02[Note 1]	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P91	A1/KR7/RXDA1/SCL02[Note 1]		
P92	A2/TIP41/TOP41		
P93	A3/TIP40/TOP40		
P94	A4/TIP31/TOP31		
P95	A5/TIP30/TOP30		
P96	A6/TIP21/TOP21		
P97	A7/SIB1/TIP20/TOP20		
P98	A8/SOB1		
P99	A9/$\overline{\text{SCKB1}}$		
P910	A10/SIB3		
P911	A11/SOB3		
P912	A12/$\overline{\text{SCKB3}}$		
P913	A13/INTP4		
P914	A14/INTP5/TIP51/TOP51		
P915	A15/INTP6/TIP50/TOP50		
PCM0	$\overline{\text{WAIT}}$	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PCM1	CLKOUT		
PCM2	$\overline{\text{HLDAK}}$		
PCM3	$\overline{\text{HLDRQ}}$		
PCT0, PCT1	$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PCT4	$\overline{\text{RD}}$		
PCT6	ASTB		
PDH0 to PDH3	A16 to A19	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PDH4, PDH5	A20, A21	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PDL0 to PDL4	AD0 to AD4	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PDL5	AD5/FLMD1[Note 2]		Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
PDL6 to PDL15	AD6 to AD15		Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.

Notes 1. I^2C version (Y version) only

2. Flash memory version only

(3/3)

Pin	Alternate Function	I/O Circuit Type	Recommended Connection
AV_{REF0}	–	–	Directly connect to V_{DD}.
AV_{REF1}	–	–	Directly connect to V_{DD}.
AV_{SS}	–	–	Directly connect to V_{SS}.
BV_{DD}	–		
BV_{SS}	–		
EV_{DD}	–	–	–
EV_{SS}	–	–	–
FLMD0[Note 1]	–	–	Directly connect to V_{SS} in a mode other than the flash programming mode.
IC[Note 2]	–	–	Directly connect to V_{SS}.
REGC	–	–	Connect regulator output stabilization capacitance.
$\overline{RESET}$	–	2	–
V_{DD}	–	–	–
V_{SS}	–	–	–
X1	–	–	–
X2	–	–	–
XT1	–	16	Connect to V_{SS} via a resistor.
XT2	–	16	Leave open.

Notes 1. Flash memory version only

2. Mask ROM version only

Figure 2-1. Pin I/O Circuits (1/2)

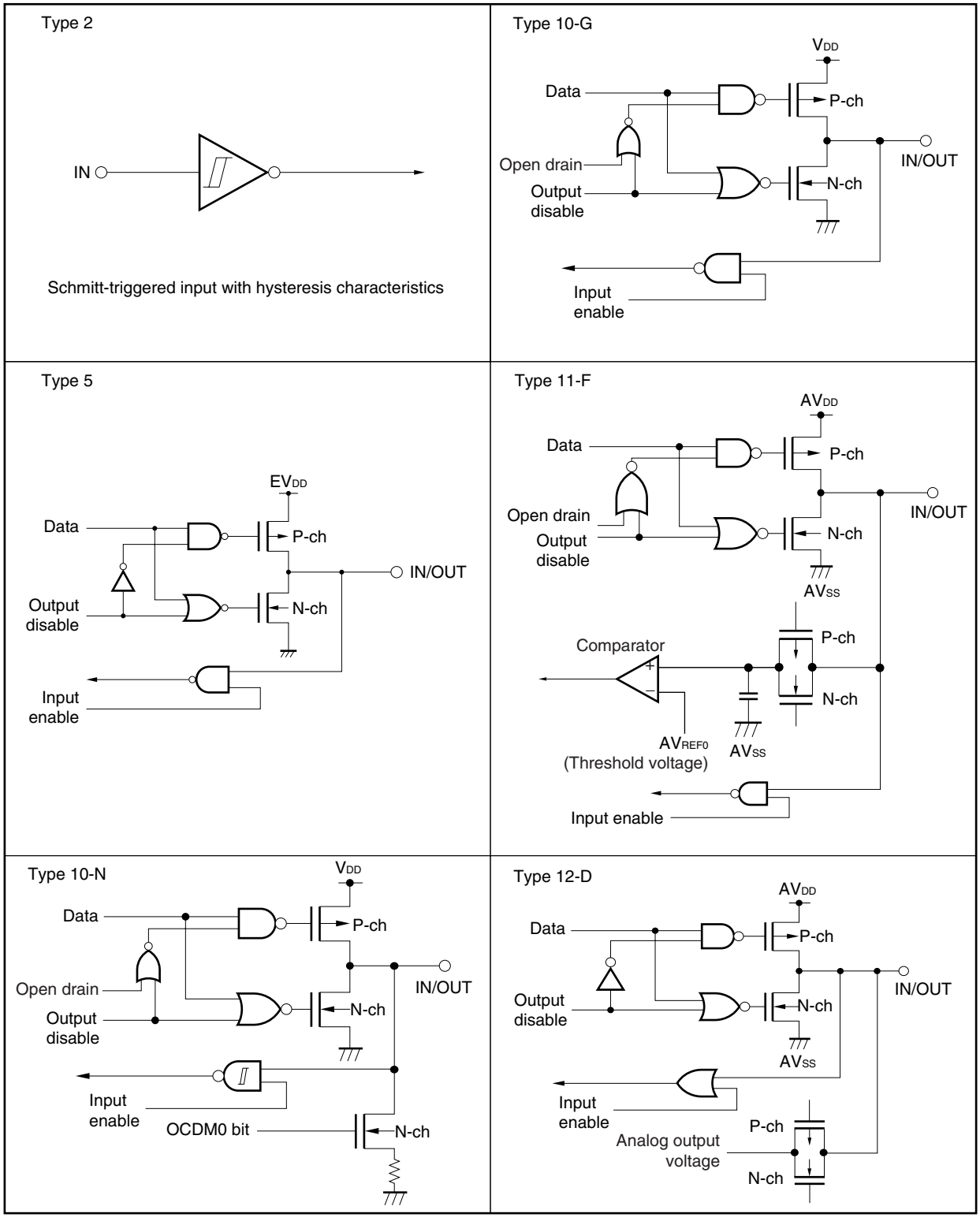
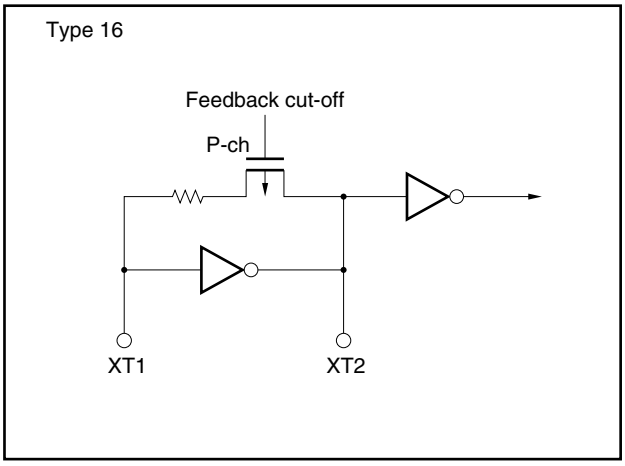


Figure 2-1. Pin I/O Circuits (2/2)



CHAPTER 3 CPU FUNCTION

The CPU of the V850ES/SG2 is based on RISC architecture and executes almost all instructions with one clock by using a 5-stage pipeline.

3.1 Features

○ Minimum instruction execution time: 50 ns (at 20 MHz operation: 3.0 to 3.6 V)
30.5 ns (with subclock (f_{XT} = 32.768 kHz operation))

○ Memory space Program space: 64 MB linear
Data space: 4 GB linear

○ General-purpose registers: 32 bits × 32 registers

○ Internal 32-bit architecture

○ 5-stage pipeline control

○ Multiplication/division instruction

○ Saturation operation instruction

○ 32-bit shift instruction: 1 clock

○ Load/store instruction with long/short format

○ Four types of bit manipulation instructions
- SET1
- CLR1
- NOT1
- TST1

3.2 CPU Register Set

The registers of the V850ES/SG2 can be classified into two types: general-purpose program registers and dedicated system registers. All the registers are 32 bits wide.

For details, refer to the **V850ES Architecture User's Manual**.

(1) Program register set

31	0
r0	(Zero register)
r1	(Assembler-reserved register)
r2	
r3	(Stack pointer (SP))
r4	(Global pointer (GP))
r5	(Text pointer (TP))
r6	
r7	
r8	
r9	
r10	
r11	
r12	
r13	
r14	
r15	
r16	
r17	
r18	
r19	
r20	
r21	
r22	
r23	
r24	
r25	
r26	
r27	
r28	
r29	
r30	(Element pointer (EP))
r31	(Link pointer (LP))

31	0
PC	(Program counter)

(2) System register set

31	0
EIPC	(Interrupt status saving register)
EIPSW	(Interrupt status saving register)

FEPC	(NMI status saving register)
FEPSW	(NMI status saving register)

ECR	(Interrupt source register)

PSW	(Program status word)

CTPC	(CALLT execution status saving register)
CTPSW	(CALLT execution status saving register)

DBPC	(Exception/debug trap status saving register)
DBPSW	(Exception/debug trap status saving register)

CTBP	(CALLT base pointer)

3.2.1 Program register set

The program registers include general-purpose registers and a program counter.

(1) General-purpose registers (r0 to r31)

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

However, r0 and r30 are implicitly used by instructions and care must be exercised when these registers are used. r0 always holds 0 and is used for an operation that uses 0 or addressing of offset 0. r30 is used by the SLD and SST instructions as a base pointer when these instructions access the memory. r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler. When using these registers, save their contents for protection, and then restore the contents after using the registers. r2 is sometimes used by the real-time OS. If the real-time OS does not use r2, it can be used as a register for variables.

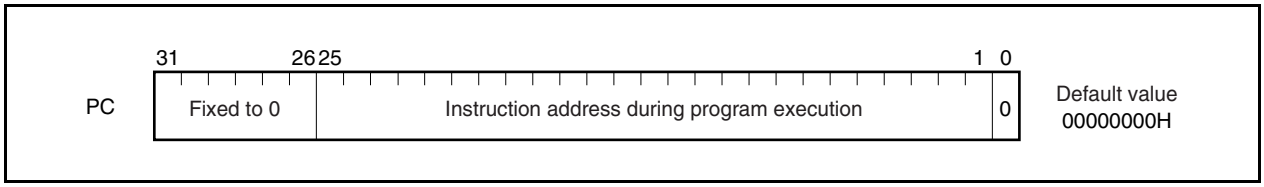
Table 3-1. Program Registers

Name	Usage	Operation
r0	Zero register	Always holds 0.
r1	Assembler-reserved register	Used as working register to create 32-bit immediate data
r2	Register for address/data variable (if real-time OS does not use r2)	
r3	Stack pointer	Used to create a stack frame when a function is called
r4	Global pointer	Used to access a global variable in the data area
r5	Text pointer	Used as register that indicates the beginning of a text area (area where program codes are located)
r6 to r29	Register for address/data variable	
r30	Element pointer	Used as base pointer to access memory
r31	Link pointer	Used when the compiler calls a function
PC	Program counter	Holds the instruction address during program execution

(2) Program counter (PC)

The program counter holds the instruction address during program execution. The lower 32 bits of this register are valid. Bits 31 to 26 are fixed to 0. A carry from bit 25 to 26 is ignored even if it occurs.

Bit 0 is fixed to 0. This means that execution cannot branch to an odd address.



3.2.2 System register set

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

These registers can be read or written by using system register load/store instructions (LDSR and STSR), using the system register numbers listed below.

Table 3-2. System Register Numbers

Register Number	System Register Name	Operand Specification	
		LDSR Instruction	STSR Instruction
0	Interrupt status saving register (EIPC)[Note 1]	√	√
1	Interrupt status saving register (EIPSW)[Note 1]	√	√
2	NMI status saving register (FEPC)	√	√
3	NMI status saving register (FEPSW)	√	√
4	Interrupt source register (ECR)	×	√
5	Program status word (PSW)	√	√
6 to 15	Reserved for future function expansion (operation is not guaranteed if these registers are accessed)	×	×
16	CALLT execution status saving register (CTPC)	√	√
17	CALLT execution status saving register (CTPSW)	√	√
18	Exception/debug trap status saving register (DBPC)	√[Note 2]	√
19	Exception/debug trap status saving register (DBPSW)	√[Note 2]	√
20	CALLT base pointer (CTBP)	√	√
21 to 31	Reserved for future function expansion (operation is not guaranteed if these registers are accessed)	×	×

Notes 1. Because only one set of these registers is available, the contents of this register must be saved by program if multiple interrupts are enabled.

2. These registers can be accessed only when the DBTRAP instruction is executed.

Caution Even if EIPC or FEPC, or bit 0 of CTPC is set to 1 by the LDSR instruction, bit 0 is ignored when execution is returned to the main routine by the RETI instruction after interrupt servicing (this is because bit 0 of the PC is fixed to 0). Set an even value to EIPC, FEPC, and CTPC (bit 0 = 0).

Remark √: Can be accessed
×: Access prohibited

(1) Interrupt status saving registers (EIPC and EIPSW)

EIPC and EIPSW are used to save the status when an interrupt occurs.

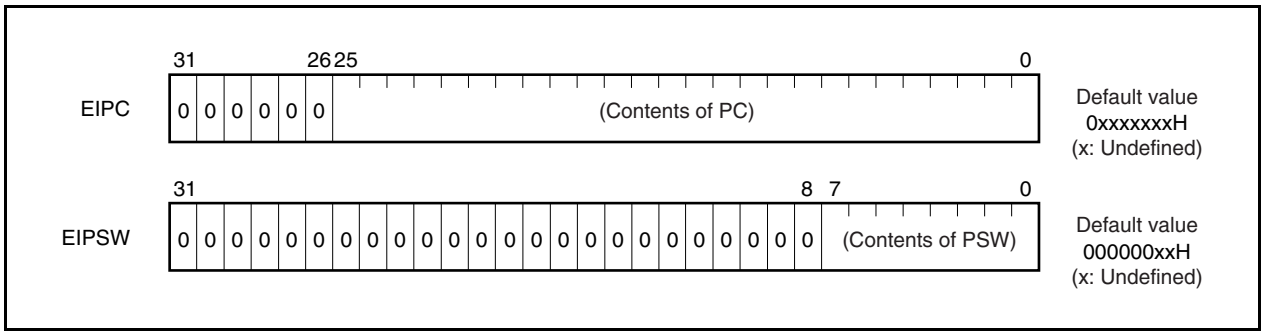
If a software exception or a maskable interrupt occurs, the contents of the program counter (PC) are saved to EIPC, and the contents of the program status word (PSW) are saved to EIPSW (these contents are saved to the NMI status saving registers (FEPC and FEPSW) if a non-maskable interrupt occurs).

The address of the instruction next to the instruction under execution, except some instructions, is saved to EIPC when a software exception or a maskable interrupt occurs.

The current contents of the PSW are saved to EIPSW.

Because only one set of interrupt status saving registers is available, the contents of these registers must be saved by program when multiple interrupts are enabled.

Bits 31 to 26 of EIPC and bits 31 to 8 of EIPSW are reserved for future function expansion (these bits are always fixed to 0).



(2) NMI status saving registers (FEPC and FEPSW)

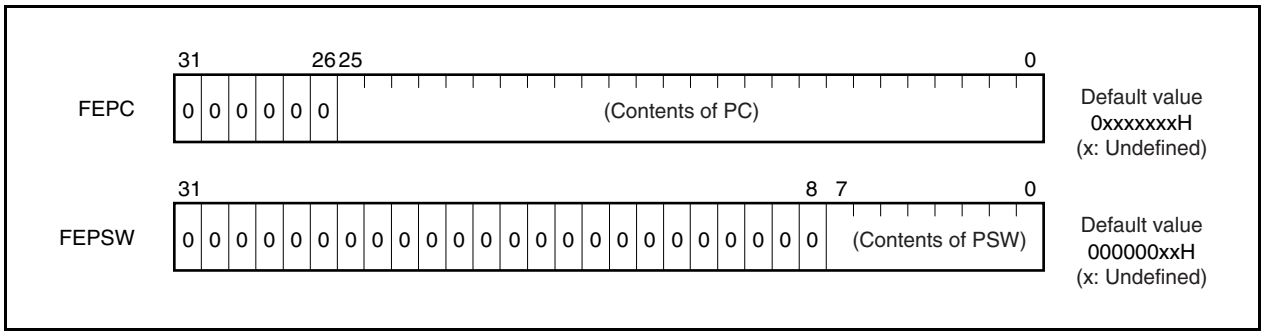
FEPC and FEPSW are used to save the status when a non-maskable interrupt (NMI) occurs.

If an NMI occurs, the contents of the program counter (PC) are saved to FEPC, and those of the program status word (PSW) are saved to FEPSW.

The address of the instruction next to the one of the instruction under execution, except some instructions, is saved to FEPC when an NMI occurs.

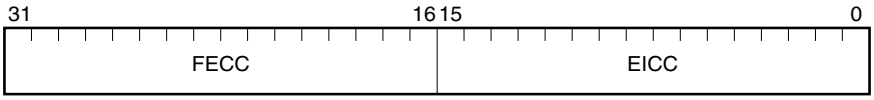
The current contents of the PSW are saved to FEPSW.

Bits 31 to 26 of FEPC and bits 31 to 8 of FEPSW are reserved for future function expansion (these bits are always fixed to 0).



(3) Interrupt source register (ECR)

The interrupt source register (ECR) holds the source of an exception or interrupt if an exception or interrupt occurs. This register holds the exception code of each interrupt source. Because this register is a read-only register, data cannot be written to this register using the LDSR instruction.



Bit position	Bit name	Meaning
31 to 16	FECC	Exception code of non-maskable interrupt (NMI)
15 to 0	EICC	Exception code of exception or maskable interrupt

(4) Program status word (PSW)

The program status word (PSW) is a collection of flags that indicate the status of the program (result of instruction execution) and the status of the CPU.

If the contents of a bit of this register are changed by using the LDSR instruction, the new contents are validated immediately after completion of LDSR instruction execution. If the ID flag is set to 1, however, interrupt request acknowledgement is disabled even while the LDSR instruction is being executed.

Bits 31 to 8 of this register are reserved for future function expansion (these bits are fixed to 0).

(1/2)

Bit position	Flag name	Meaning
31 to 8	RFU	Reserved field. Fixed to 0.
7	NP	Indicates that a non-maskable interrupt (NMI) is being serviced. This bit is set to 1 when an NMI request is acknowledged, disabling multiple interrupts. 0: NMI is not being serviced. 1: NMI is being serviced.
6	EP	Indicates that an exception is being processed. This bit is set to 1 when an exception occurs. Even if this bit is set, interrupt requests are acknowledged. 0: Exception is not being processed. 1: Exception is being processed.
5	ID	Indicates whether a maskable interrupt can be acknowledged. 0: Interrupt enabled 1: Interrupt disabled
4	SAT[Note]	Indicates that the result of a saturation operation has overflowed and is saturated. Because this is a cumulative flag, it is set to 1 when the result of a saturation operation instruction is saturated, and is not cleared to 0 even if the subsequent operation result is not saturated. Use the LDSR instruction to clear this bit. This flag is neither set to 1 nor cleared to 0 by execution of an arithmetic operation instruction. 0: Not saturated 1: Saturated
3	CY	Indicates whether a carry or a borrow occurs as a result of an operation. 0: Carry or borrow does not occur. 1: Carry or borrow occurs.
2	OV[Note]	Indicates whether an overflow occurs during operation. 0: Overflow does not occur. 1: Overflow occurs.
1	S[Note]	Indicates whether the result of an operation is negative. 0: The result is positive or 0. 1: The result is negative.
0	Z	Indicates whether the result of an operation is 0. 0: The result is not 0. 1: The result is 0.

Remark Also read **Note** on the next page.

(2/2)

Note The result of the operation that has performed saturation processing is determined by the contents of the OV and S flags. The SAT flag is set to 1 only when the OV flag is set to 1 when a saturation operation is performed.

Status of operation result	Flag status			Result of operation of saturation processing
	SAT	OV	S	
Maximum positive value is exceeded	1	1	0	7FFFFFFFH
Maximum negative value is exceeded	1	1	1	80000000H
Positive (maximum value is not exceeded)	Holds value before operation	0	0	Operation result itself
Negative (maximum value is not exceeded)			1	

(5) CALLT execution status saving registers (CTPC and CTPSW)

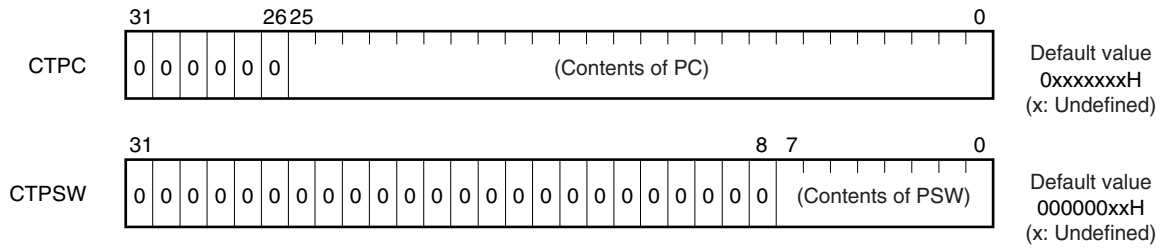
CTPC and CTPSW are CALLT execution status saving registers.

When the CALLT instruction is executed, the contents of the program counter (PC) are saved to CTPC, and those of the program status word (PSW) are saved to CTPSW.

The contents saved to CTPC are the address of the instruction next to CALLT.

The current contents of the PSW are saved to CTPSW.

Bits 31 to 26 of CTPC and bits 31 to 8 of CTPSW are reserved for future function expansion (fixed to 0).



(6) Exception/debug trap status saving registers (DBPC and DBPSW)

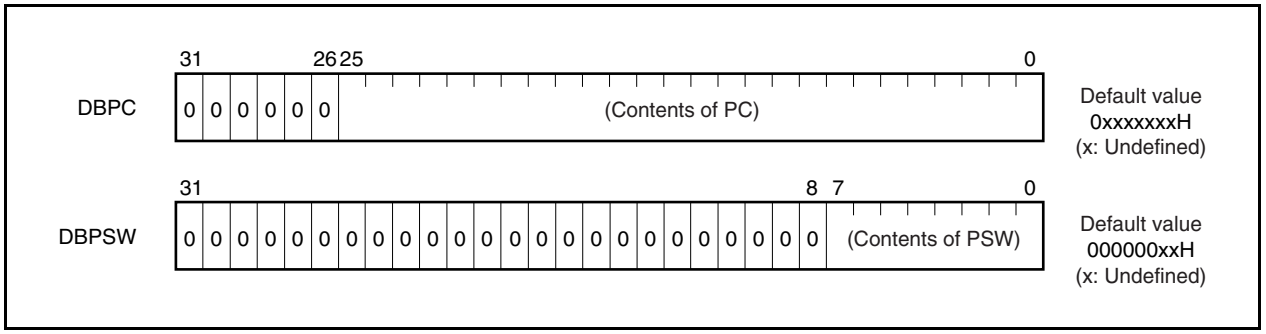
DBPC and DBPSW are exception/debug trap status registers.

If an exception trap or debug trap occurs, the contents of the program counter (PC) are saved to DBPC, and those of the program status word (PSW) are saved to DBPSW.

The contents to be saved to DBPC are the address of the instruction next to the one that is being executed when an exception trap or debug trap occurs.

The current contents of the PSW are saved to DBPSW.

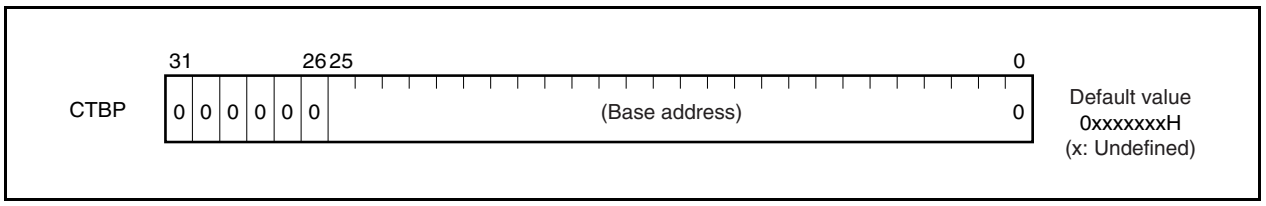
Bits 31 to 26 of DBPC and bits 31 to 8 of DBPSW are reserved for future function expansion (fixed to 0).



(7) CALLT base pointer (CTBP)

The CALLT base pointer (CTBP) is used to specify a table address or generate a target address (bit 0 is fixed to 0).

Bits 31 to 26 of this register are reserved for future function expansion (fixed to 0).



3.3 Operation Modes

3.3.1 Operation modes

The V850ES/SG2 has the following operation modes.

(1) Normal operation mode

In this mode, each pin related to the bus interface is set to the port mode after system reset has been released. Execution branches to the reset entry address of the internal ROM, and then instruction processing is started. By setting the PMCDH, PMCDL, PMCCM, and PMCCT registers to the control mode using software, an external device can be connected to the external memory area.

(2) Flash memory programming mode

In this mode, the internal flash memory can be programmed by using a flash programmer.

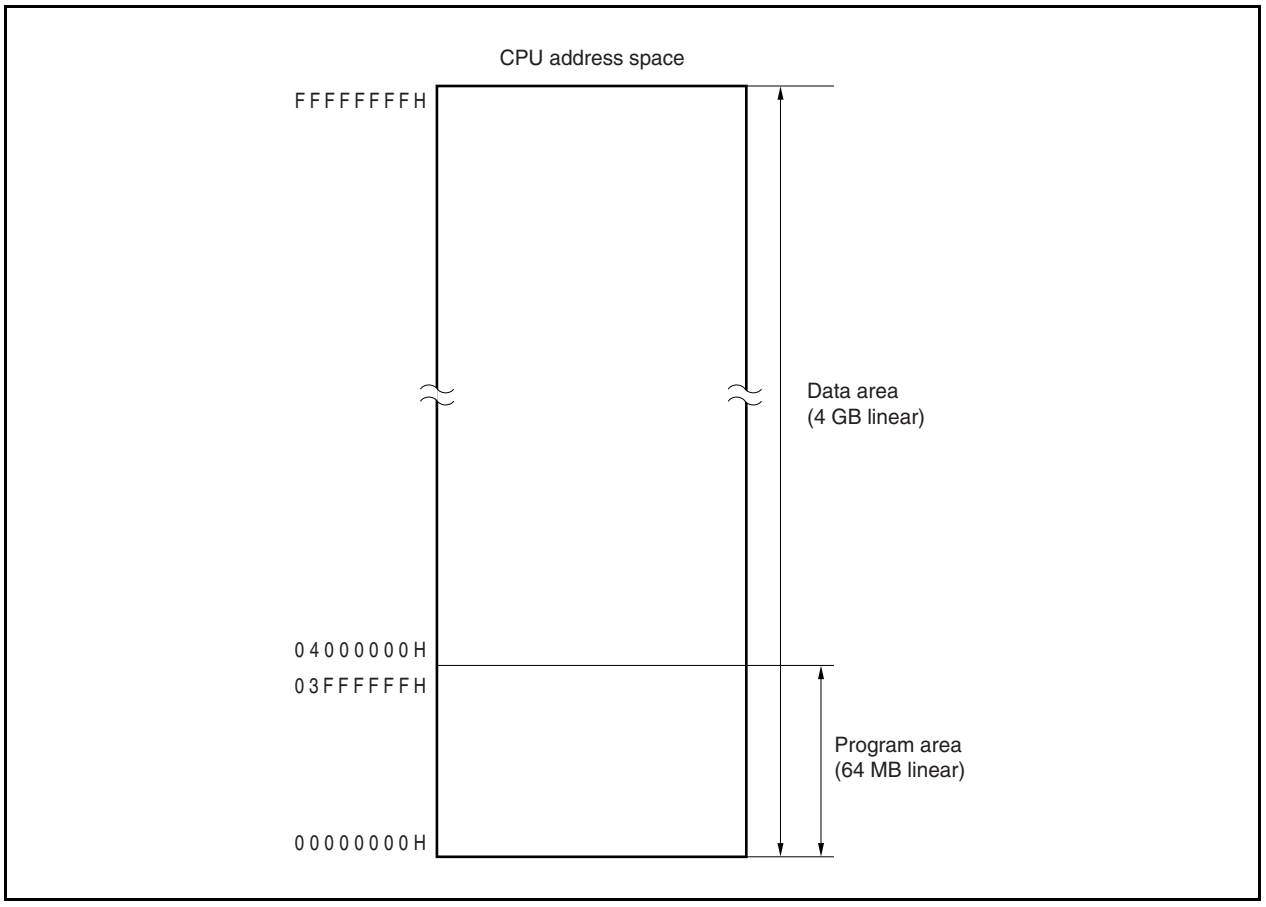
3.4 Address Space

3.4.1 CPU address space

The CPU of the V850ES/SG2 has 32-bit architecture and supports up to 4 GB of linear address space (data space) for operand addressing (data access). It also supports up to 64 MB of linear address space (program space) for instruction addressing. Note, however, that both the program and data spaces have areas that are prohibited from being used. For details, see **Figure 3-2**.

Figure 3-1 shows the CPU address space.

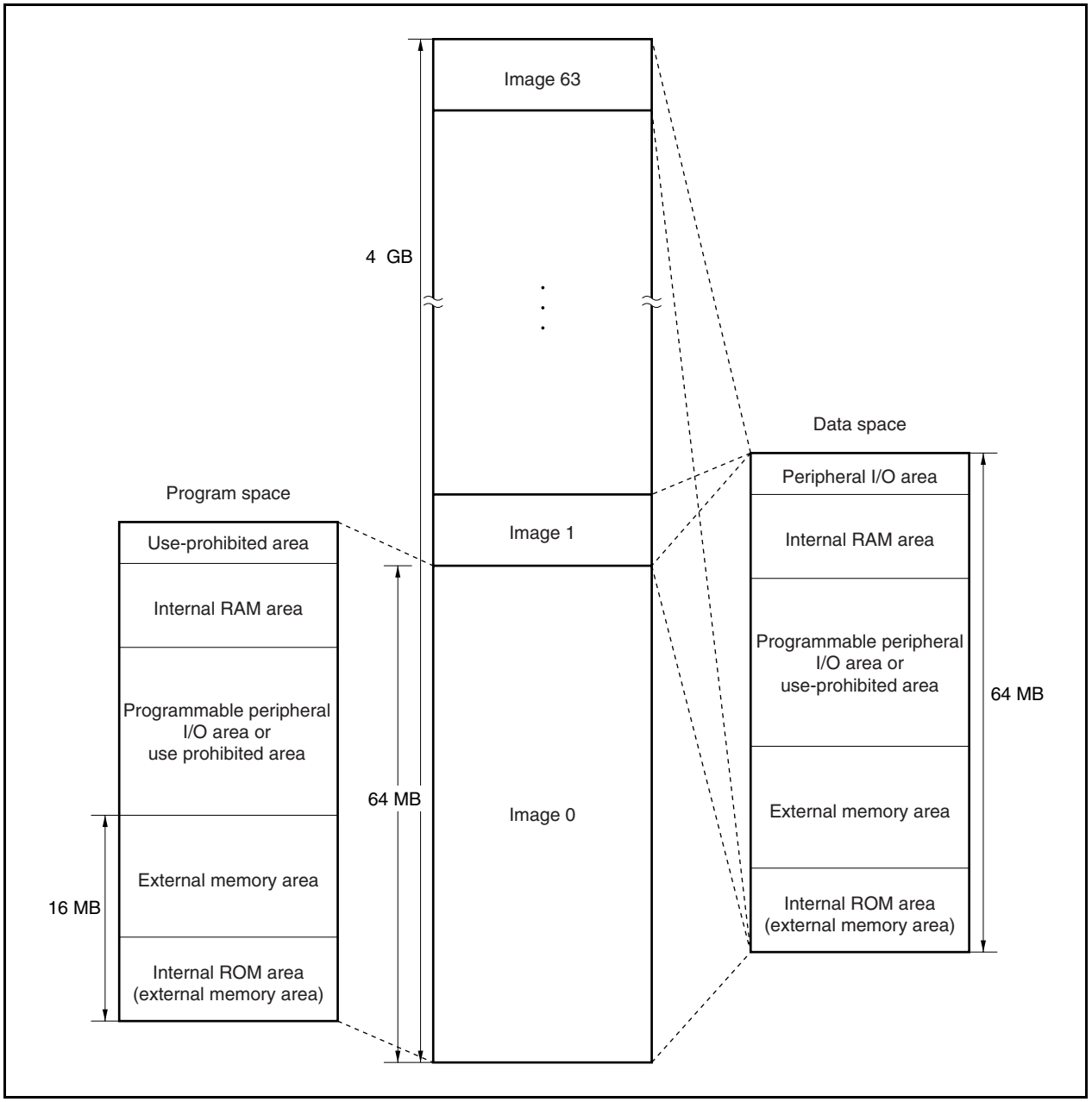
Figure 3-1. CPU Address Space



3.4.2 Image

For addressing instruction addresses, up to 16 MB of external memory area, internal ROM area, and internal RAM area in an area of up to 16 MB of linear address space (program space) is supported. Up to 4 GB of linear address space (data space) is supported for operand addressing (data access). In the 4 GB address space, it seems that there are sixty-four 64 MB physical address spaces. This means that the same 64 MB physical address space is accessed, regardless of the values of bits 31 to 26.

Figure 3-2. Image on Address Space



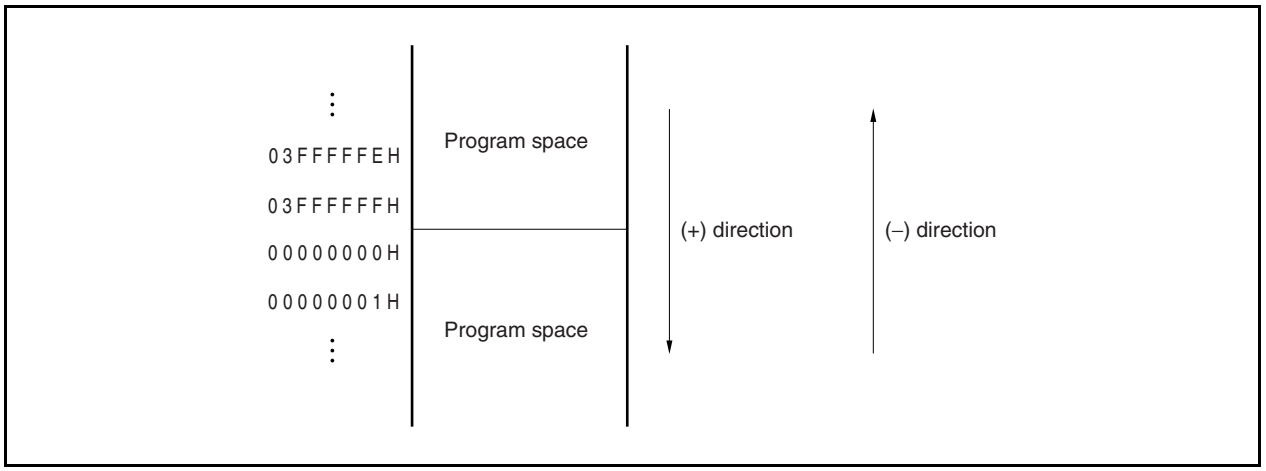
3.4.3 Wraparound of CPU address space

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0 and only the lower 26 bits are valid. The higher 6 bits ignore a carry or borrow from bit 25 to 26 during branch address calculation.

Therefore, the lowest address of the program space, 00000000H, and the highest address, 03FFFFFFH, are contiguous addresses. That the lowest address and the highest address of the program space are contiguous in this way is called wraparound.

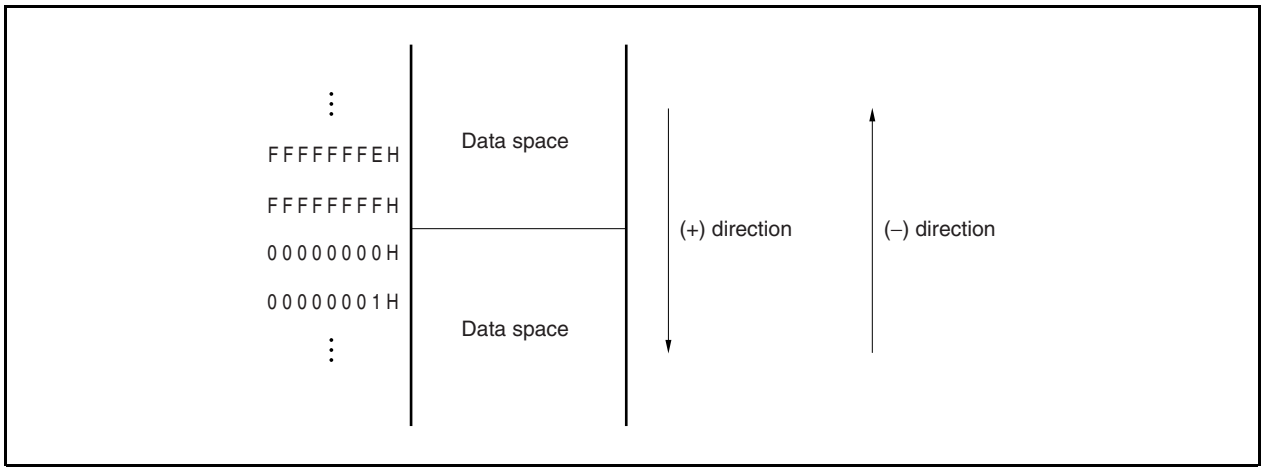
Caution Because the 4 KB area of addresses 03FFF000H to 03FFFFFFH is an on-chip peripheral I/O area, instructions cannot be fetched from this area. Therefore, do not execute an operation in which the result of a branch address calculation affects this area.



(2) Data space

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

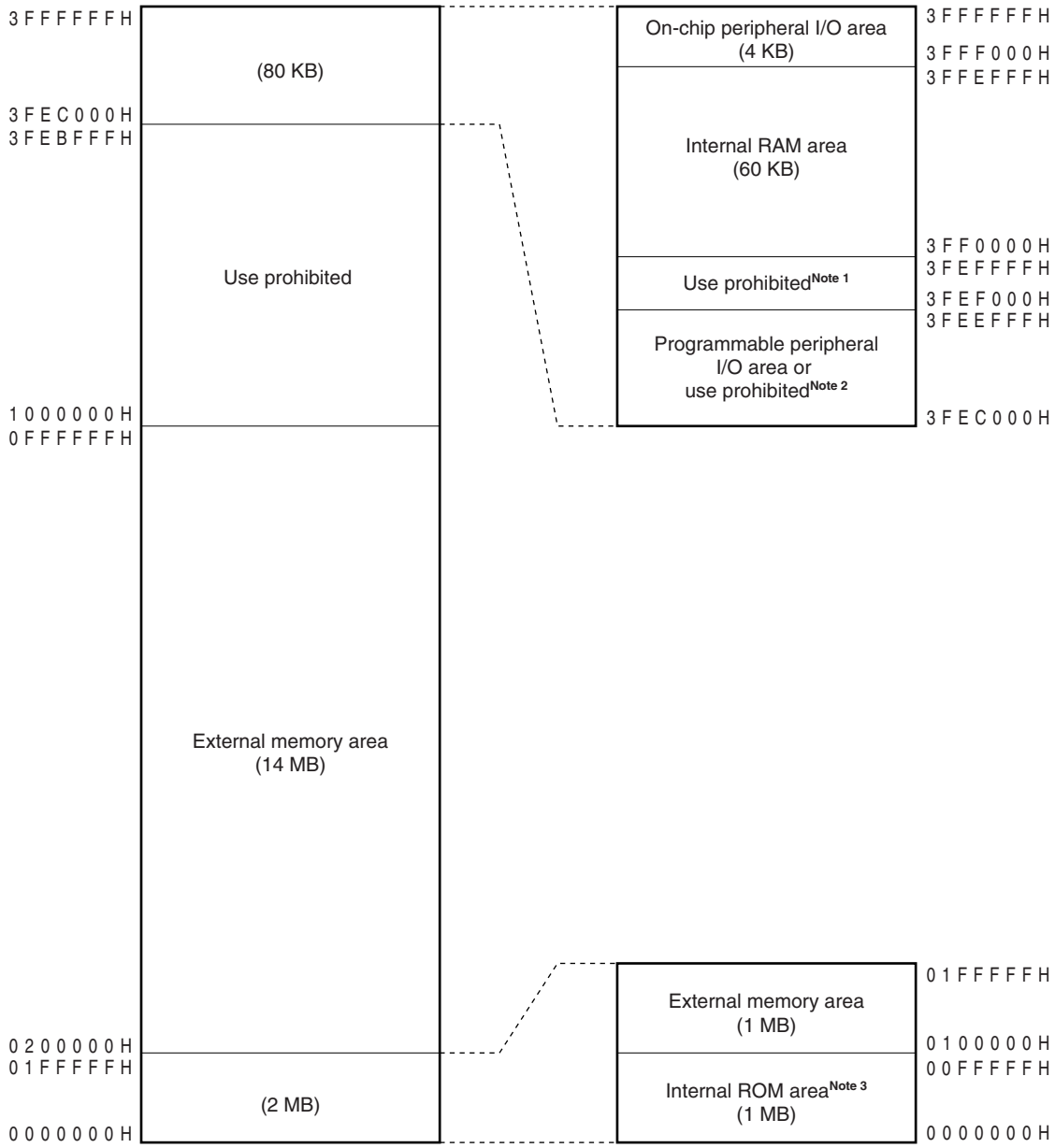
Therefore, the lowest address of the data space, 00000000H, and the highest address, FFFFFFFFH, are contiguous, and wraparound occurs at the boundary of these addresses.



3.4.4 Memory map

The areas shown in Figure 3-3 are reserved in the V850ES/SG2.

Figure 3-3. Data Memory Map (Physical Addresses)

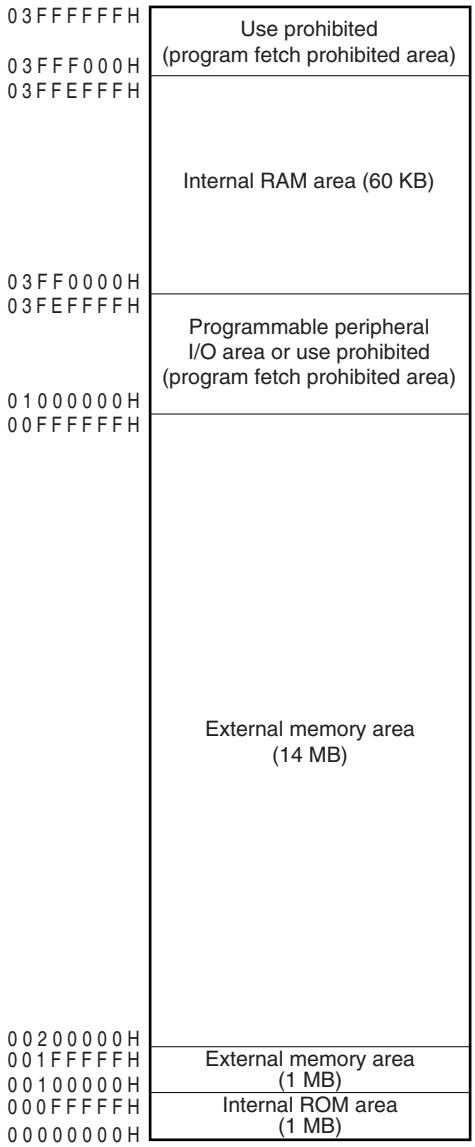


Notes 1. Use of addresses 3FEF000H to 3FEFFFFH is prohibited because these addresses are in the same area as the on-chip peripheral I/O area.

2. Addresses 3FEC000H to 3FECBFFH are allocated to addresses 3FEC000H to 3FEEFFFH of the CAN controller version as a programmable peripheral I/O area. Use of these addresses in a version without a CAN controller is prohibited.

3. Fetch access and read access to addresses 0000000H to 00FFFFFH is made to the internal ROM area. However, data write access to these addresses is made to the external memory area.

Figure 3-4. Program Memory Map



Remark Instructions can be executed to the external memory area without execution branching from the internal ROM area to the external memory area.

3.4.5 Areas

(1) Internal ROM/internal flash memory area

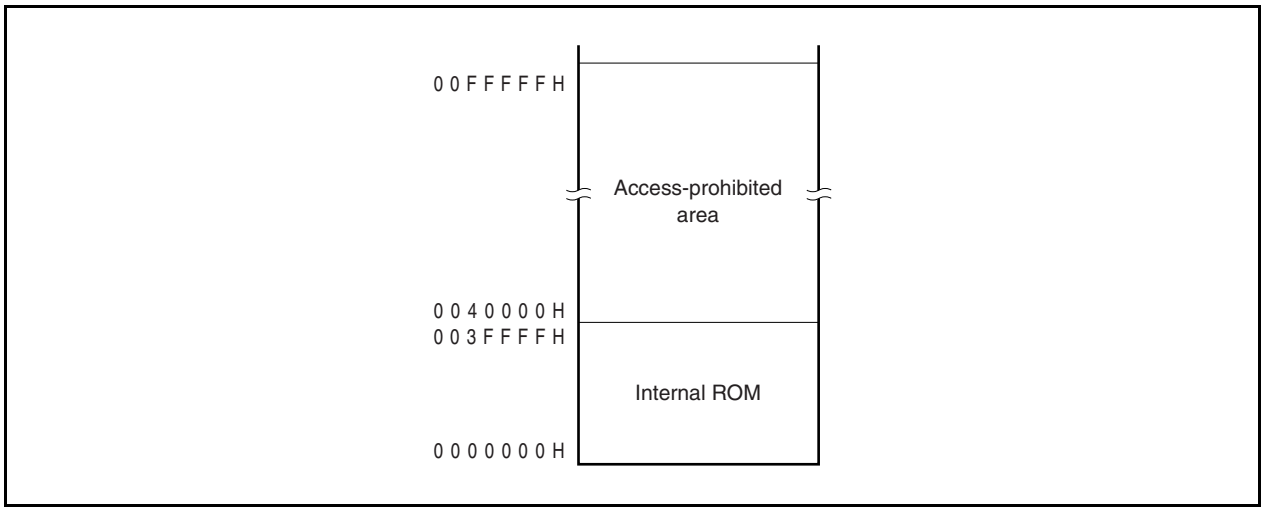
Up to 1 MB is reserved as an internal ROM/internal flash memory area.

(a) Internal ROM (256 KB)

256 KB are allocated to addresses 0000000H to 003FFFFH of the following versions.
Accessing addresses 0040000H to 00FFFFFH is prohibited.

- μPD703260, 703260Y, 703270, 703270Y, 703280, 703280Y

Figure 3-5. Internal ROM Area (256 KB)

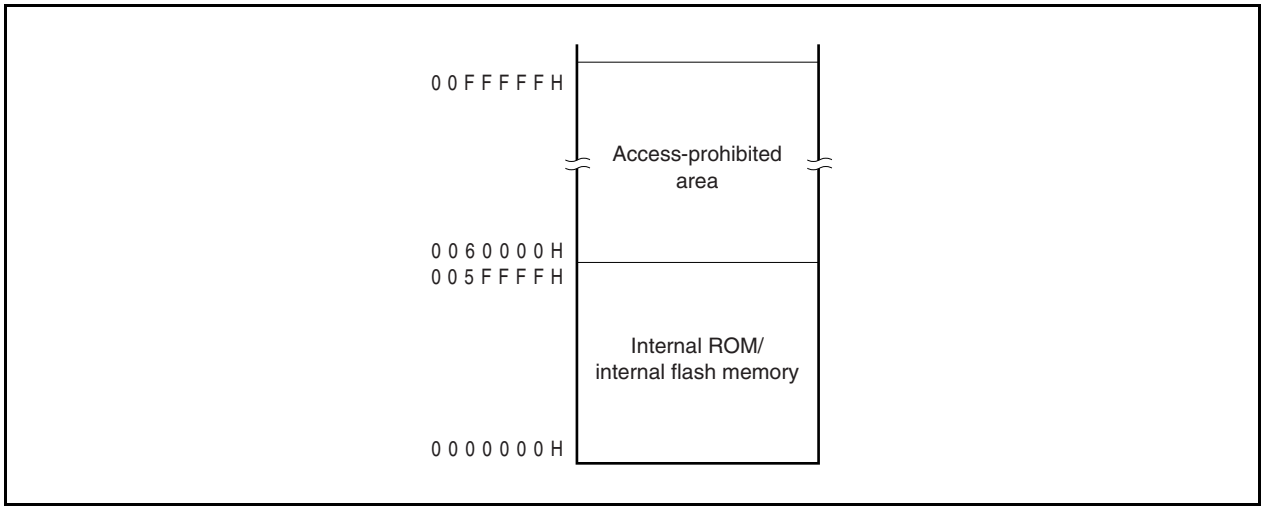


(b) Internal ROM/internal flash memory area (384 KB)

384 KB are allocated to addresses 0000000H to 005FFFFH of the following versions.
Accessing addresses 0060000H to 00FFFFFH is prohibited.

- μPD703261, 703261Y, 703271, 703271Y, 703281, 703281Y, 70F3261, 70F3261Y, 70F3271, 70F3271Y, 70F3281, 70F3281Y

Figure 3-6. Internal ROM/Internal Flash Memory Area (384 KB)

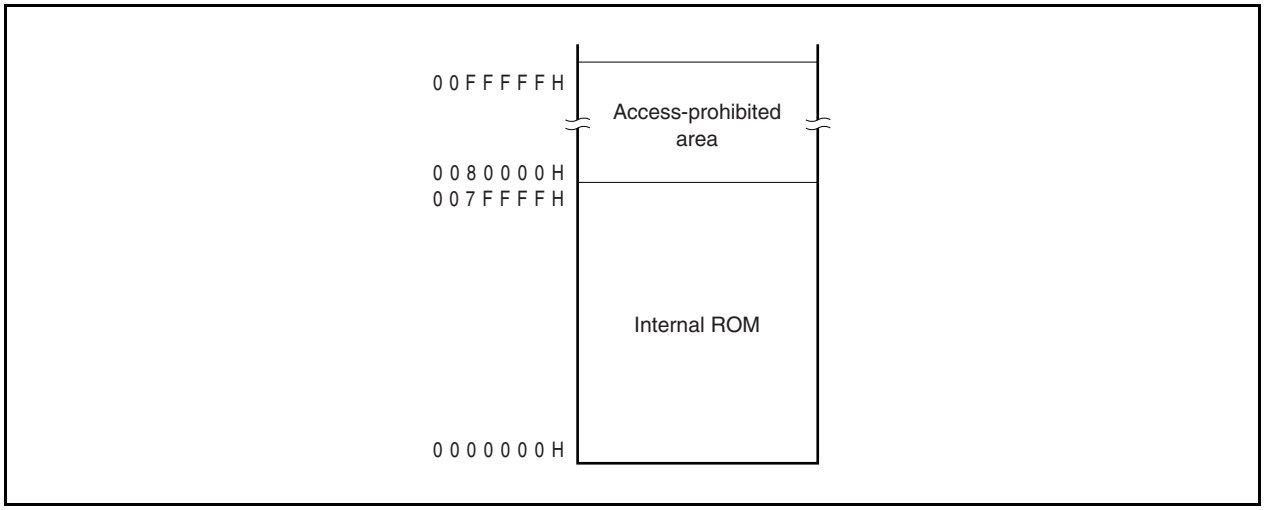


(c) Internal ROM (512 KB)

512 KB are allocated to addresses 0000000H to 007FFFFH of the following versions.
Accessing addresses 0080000H to 00FFFFFH is prohibited.

- μPD703262, 703262Y, 703272, 703272Y, 703282, 703282Y,

Figure 3-7. Internal ROM Area (512 KB)



(d) Internal ROM/internal flash memory area (640 KB)

640 KB are allocated to addresses 0000000H to 009FFFFH of the following versions.
Accessing addresses 00A0000H to 00FFFFFH is prohibited.

- μPD703263, 703263Y, 703273, 703273Y, 703283, 703283Y, 70F3263, 70F3263Y, 70F3273, 70F3273Y, 70F3283, 70F3283Y

Figure 3-8. Internal ROM/Internal Flash Memory Area (640 KB)

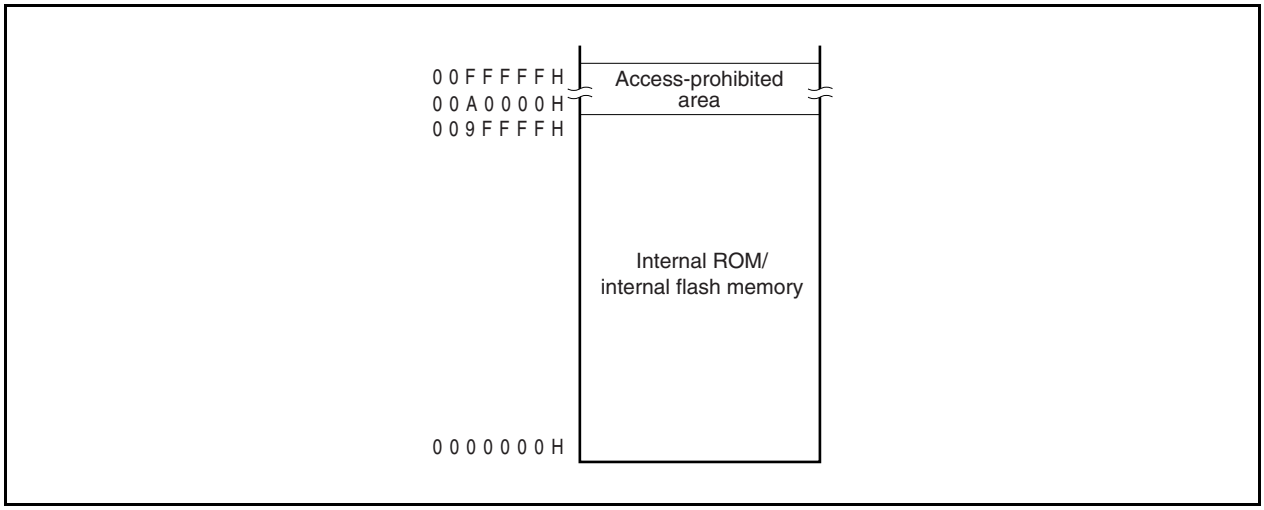


Table 3-3. Interrupt/Exception Table

First Address of Interrupt/ Exception Table	Interrupt/ Exception Source	First Address of Interrupt/ Exception Table	Interrupt/ Exception Source
00000000H	RESET	00000230H	INTTP4CC0
00000010H	NMI	00000240H	INTTP4CC1
00000020H	INTWDT2	00000250H	INTTP5OV
0000004nH	TRAP0n (n = 0 to F)	00000260H	INTTP5CC0
0000005nH	TRAP1n (n = 0 to F)	00000270H	INTTP5CC1
00000060H	ILGOP/DBG0	00000280H	INTTM0EQ0
00000080H	INTLVI	00000290H	INTCB0R/INTIIC1[Note 1]
00000090H	INTP0	000002A0H	INTCB0T
000000A0H	INTP1	000002B0H	INTCB1R
000000B0H	INTP2	000002C0H	INTCB1T
000000C0H	INTP3	000002D0H	INTCB2R
000000D0H	INTP4	000002E0H	INTCB2T
000000E0H	INTP5	000002F0H	INTCB3R
000000F0H	INTP6	00000300H	INTCB3T
00000100H	INTP7	00000310H	INTUA0R/INTCB4R
00000110H	INTTQ0OV	00000320H	INTUA0T/INTCB4T
00000120H	INTTQ0CC0	00000330H	INTUA1R/INTIIC2[Note 1]
00000130H	INTTQ0CC1	00000340H	INTUA1T
00000140H	INTTQ0CC2	00000350H	INTUA2R/INTIIC0[Note 1]
00000150H	INTTQ0CC3	00000360H	INTUA2T
00000160H	INTTP0OV	00000370H	INTAD
00000170H	INTTP0CC0	00000380H	INTDMA0
00000180H	INTTP0CC1	00000390H	INTDMA1
00000190H	INTTP1OV	000003A0H	INTDMA2
000001A0H	INTTP1CC0	000003B0H	INTDMA3
000001B0H	INTTP1CC1	000003C0H	INTKR
000001C0H	INTTP2OV	000003D0H	INTWTI
000001D0H	INTTP2CC0	000003E0H	INTWT
000001E0H	INTTP2CC1	000003F0H	INTC0ERR[Note 2]/INTERR[Note 3]
000001F0H	INTTP3OV	00000400H	INTC0WUP[Note 2]/INTSTA[Note 3]
00000200H	INTTP3CC0	00000410H	INTC0REC[Note 2]/INTIE1[Note 3]
00000210H	INTTP3CC1	00000420H	INTC0TRX[Note 2]/INTIE2[Note 3]
00000220H	INTTP4OV	–	–

Notes 1. I^2C bus version only

2. CAN controller version only

3. IEBus controller version only

(2) Internal RAM area

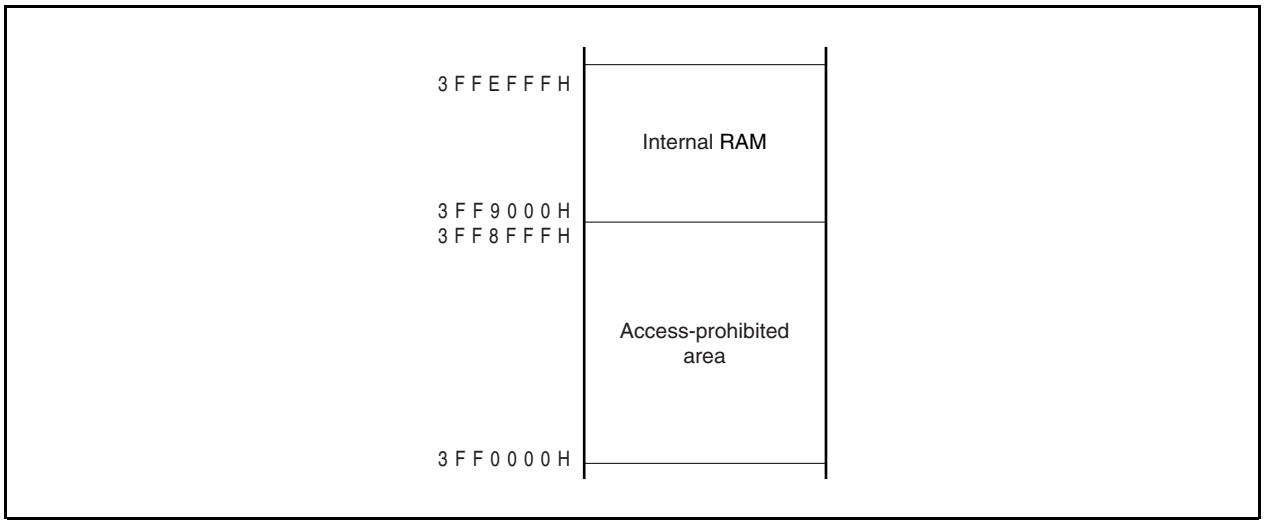
Up to 60 KB are reserved as the internal RAM area.

(a) Internal RAM (24 KB)

24 KB are allocated to addresses 3FF9000H to 3FFEFFFH of the following versions.
Accessing addresses 3FF0000H to 3FF8FFFH is prohibited.

- μPD703260, 703260Y, 703270, 703270Y, 703280, 703280Y

Figure 3-9. Internal RAM Area (24 KB)

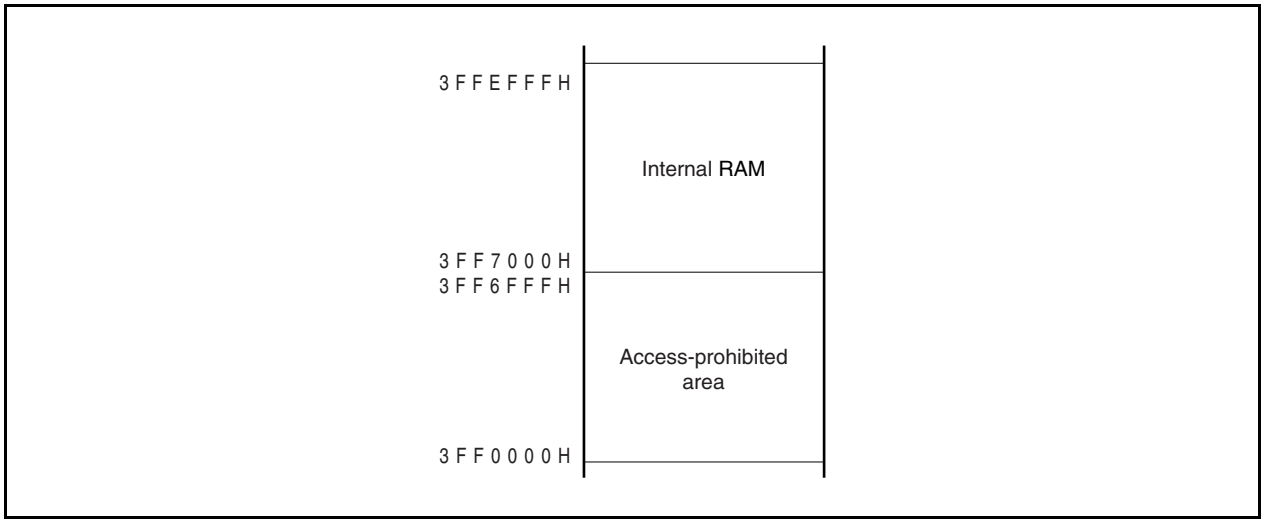


(b) Internal RAM (32 KB)

32 KB are allocated to addresses 3FF7000H to 3FFEFFFH of the following versions.
Accessing addresses 3FF0000H to 3FF6FFFH is prohibited.

- μPD703261, 703261Y, 703271, 703271Y, 703281, 703281Y, 70F3261, 70F3261Y, 70F3271, 70F3271Y, 70F3281, 70F3281Y

Figure 3-10. Internal RAM Area (32 KB)

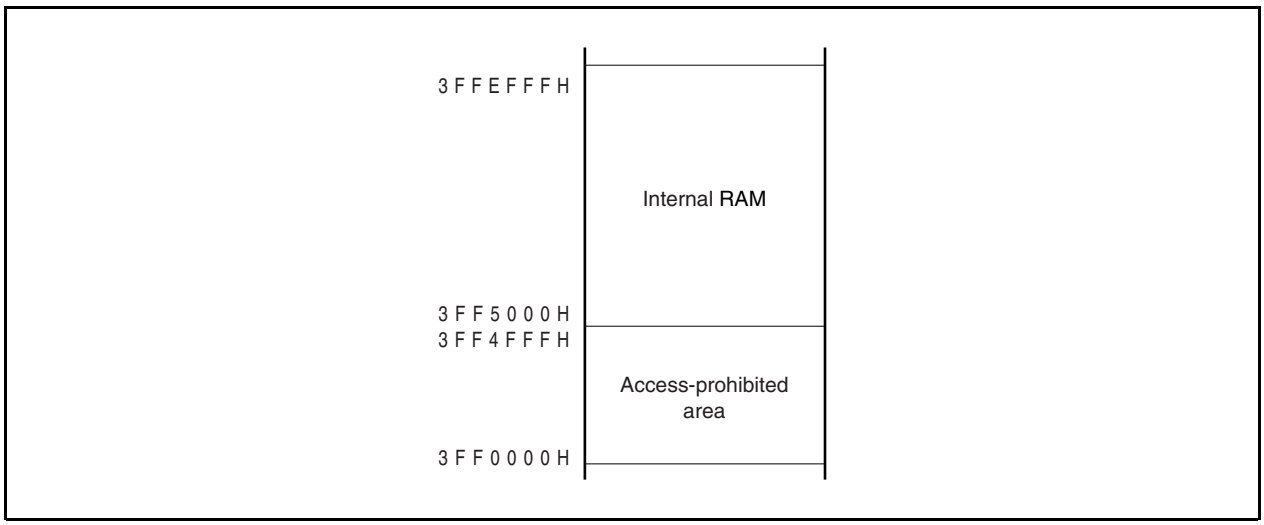


(c) Internal RAM (40 KB)

40 KB are allocated to addresses 3FF5000H to 3FFEFFFH of the following versions.
Accessing addresses 3FF0000H to 3FF4FFFH is prohibited.

- μPD703262, 703262Y, 703272, 703272Y, 703282, 703282Y,

Figure 3-11. Internal RAM Area (40 KB)

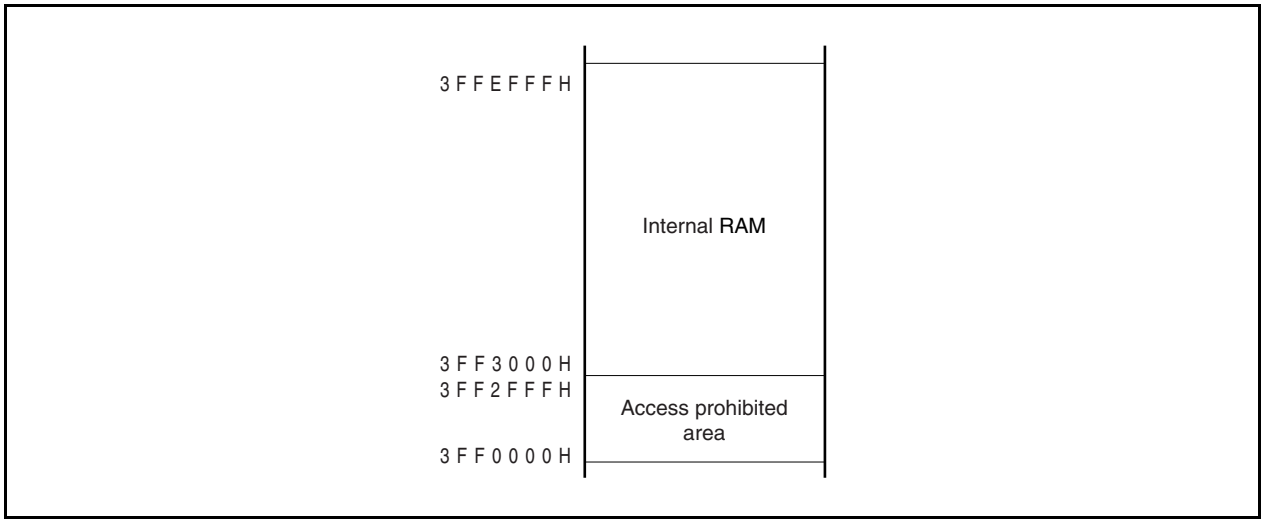


(d) Internal RAM area (48 KB)

48 KB are allocated to addresses 3FF3000H to 3FFEFFFH of the following versions.
Accessing addresses 3FF0000H to 3FF2FFFH is prohibited.

- μPD703263, 703263Y, 703273, 703273Y, 703283, 703283Y, 70F3263, 70F3263Y, 70F3273, 70F3273Y, 70F3283, 70F3283Y

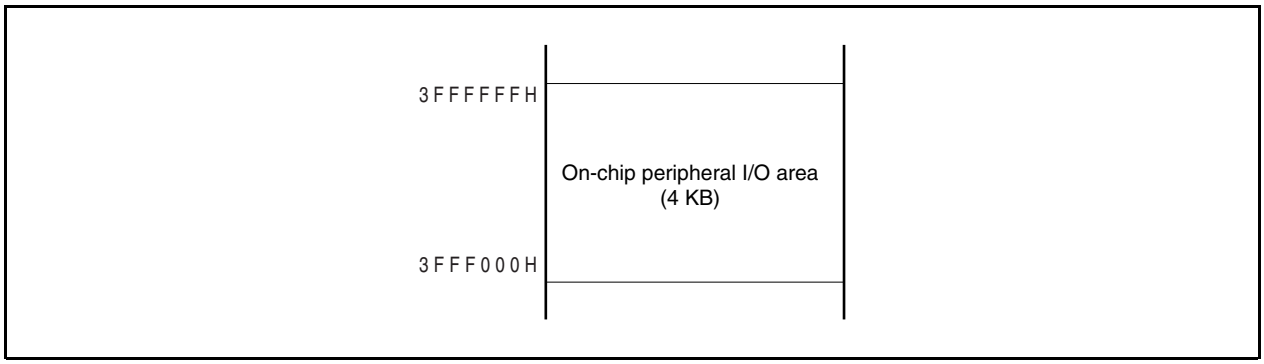
Figure 3-12. Internal RAM Area (48 KB)



(3) On-chip peripheral I/O area

4 KB of addresses 3FFF000H to 3FFFFFFH are reserved as the on-chip peripheral I/O area.

Figure 3-13. On-Chip Peripheral I/O Area



Peripheral I/O registers that have functions to specify the operation mode for and monitor the status of the on-chip peripheral I/O are mapped to the on-chip peripheral I/O area. Program cannot be fetched from this area.

Cautions 1. When a register is accessed in word units, a word area is accessed twice in halfword units in the order of lower area and higher area, with the lower 2 bits of the address ignored.

2. If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits are undefined when the register is read, and data is written to the lower 8 bits.

3. Addresses not defined as registers are reserved for future expansion. The operation is undefined and not guaranteed when these addresses are accessed.

(4) External memory area

15 MB (0100000H to 0FFFFFFH) are allocated as the external memory area. For details, see **CHAPTER 5 BUS CONTROL FUNCTION**.

Caution The V850ES/SG2 has 22 address pins (A0 to A21), so the external memory area appears as a repeated 4 MB image. In this case, it is necessary that $EV_{DD} = BV_{DD} = V_{DD}$.

3.4.6 Recommended use of address space

The architecture of the V850ES/SG2 requires that a register that serves as a pointer be secured for address generation when operand data in the data space is accessed. The address stored in this pointer ±32 KB can be directly accessed by an instruction for operand data. Because the number of general-purpose registers that can be used as a pointer is limited, however, by keeping the performance from dropping during address calculation when a pointer value is changed, as many general-purpose registers as possible can be secured for variables, and the program size can be reduced.

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0, and only the lower 26 bits are valid. Regarding the program space, therefore, a 64 MB space of contiguous addresses starting from 00000000H unconditionally corresponds to the memory map.

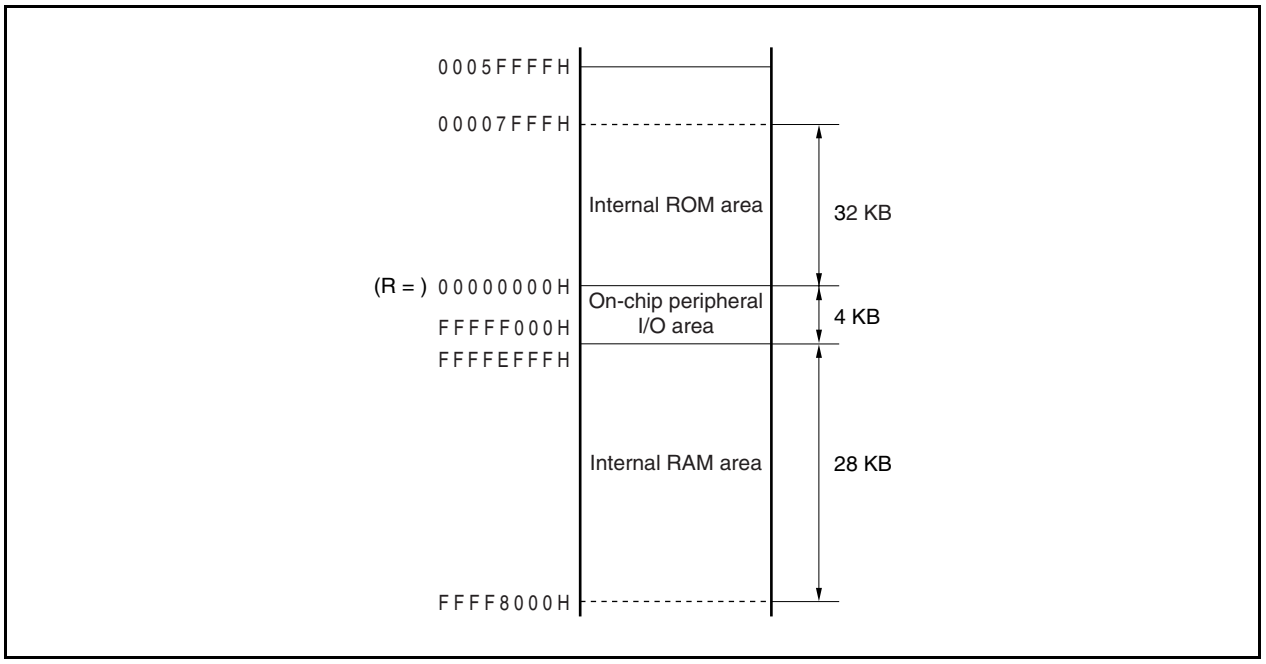
To use the internal RAM area as the program space, access following addresses.

RAM Size	Access Address
48 KB	3FF3000H to 3FFEFFFH
40 KB	3FF5000H to 3FFEFFFH
32 KB	3FF7000H to 3FFEFFFH
24 KB	3FF9000H to 3FFEFFFH

(2) Data space

With the V850ES/SG2, it seems that there are sixty-four 64 MB address spaces on the 4 GB CPU address space. Therefore, the least significant bit (bit 25) of a 26-bit address is sign-extended to 32 bits and allocated as an address.

Example: μPD703261, 703261Y

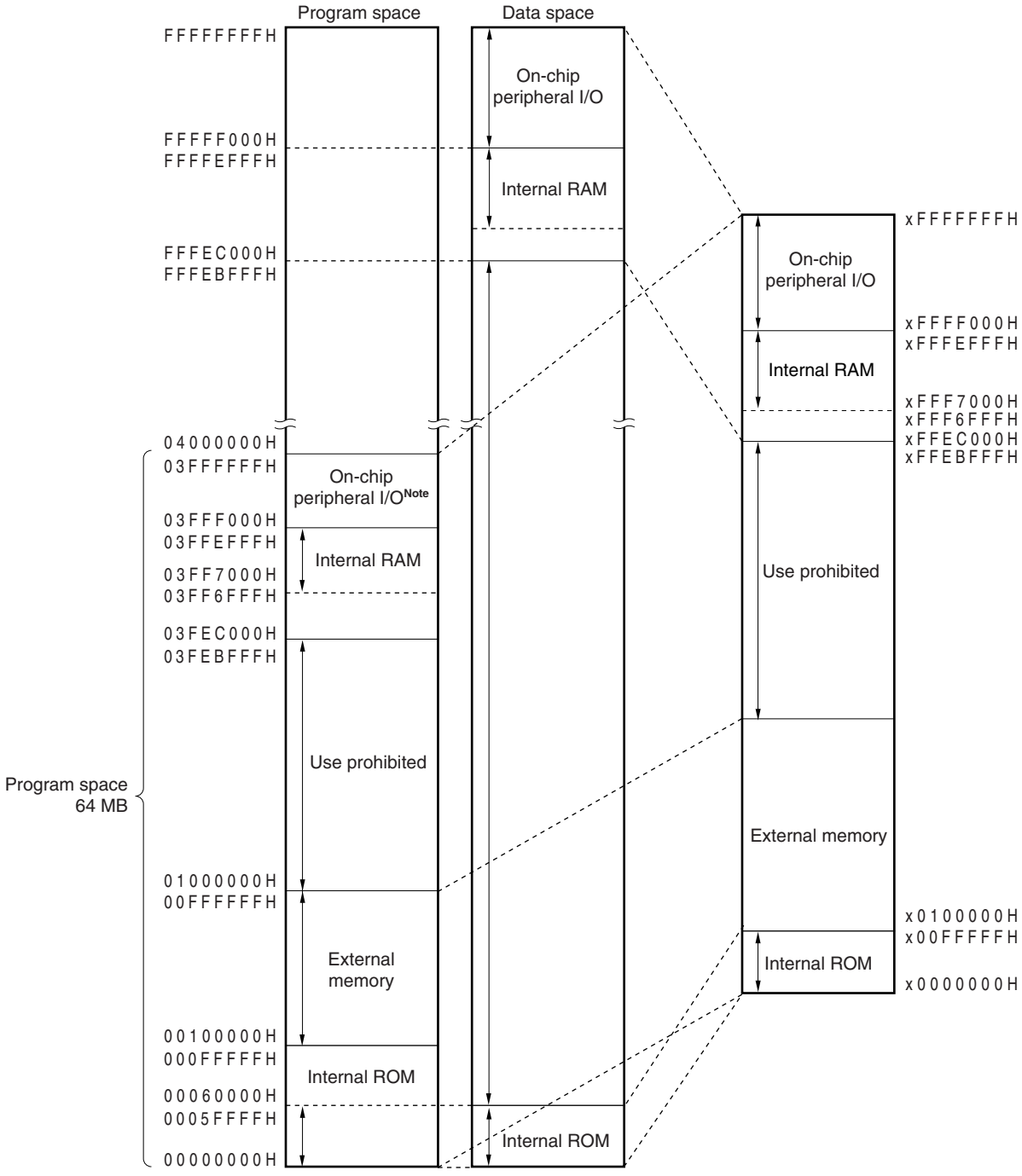


(a) Application example of wraparound

If R = r0 (zero register) is specified for the LD/ST disp16 [R] instruction, a range of addresses 00000000H ±32 KB can be addressed by sign-extended disp16. All the resources, including the internal hardware, can be addressed by one pointer.

The zero register (r0) is a register fixed to 0 by hardware, and practically eliminates the need for registers dedicated to pointers.

Figure 3-14. Recommended Memory Map



Note Access to this area is prohibited. To access the on-chip peripheral I/O in this area, specify addresses FFFF000H to FFFFFFFH.

Remarks 1. ↕ indicates the recommended area.

2. This figure is the recommended memory map of the μPD703261 and 703261Y.

3.4.7 Peripheral I/O registers

(1/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF004H	Port DL	PDL	R/W			√	Undefined
FFFFF004H	Port DLL	PDLL		√	√		Undefined
FFFFF005H	Port DLH	PDLH		√	√		Undefined
FFFFF006H	Port DH	PDH		√	√		Undefined
FFFFF00AH	Port CT	PCT		√	√		Undefined
FFFFF00CH	Port CM	PCM		√	√		Undefined
FFFFF024H	Port mode register DL	PMDL				√	FFFFH
FFFFF024H	Port mode register DLL	PMDLL		√	√		FFH
FFFFF025H	Port mode register DLH	PMDLH		√	√		FFH
FFFFF026H	Port mode register DH	PMDH		√	√		FFH
FFFFF02AH	Port mode register CT	PMCT		√	√		FFH
FFFFF02CH	Port mode register CM	PMCM		√	√		FFH
FFFFF044H	Port mode control register DL	PMCDL				√	0000H
FFFFF044H	Port mode control register DLL	PMCDLL		√	√		00H
FFFFF045H	Port mode control register DLH	PMCDLH		√	√		00H
FFFFF046H	Port mode control register DH	PMCDH		√	√		00H
FFFFF04AH	Port mode control register CT	PMCCT		√	√		00H
FFFFF04CH	Port mode control register CM	PMCCM		√	√		00H
FFFFF064H	Peripheral I/O area select control register	BPC[Note]				√	0000H
FFFFF066H	Bus size configuration register	BSC				√	5555H
FFFFF06EH	System wait control register	VSWC			√		77H
FFFFF080H	DMA source address register 0L	DSA0L				√	Undefined
FFFFF082H	DMA source address register 0H	DSA0H				√	Undefined
FFFFF084H	DMA destination address register 0L	DDA0L				√	Undefined
FFFFF086H	DMA destination address register 0H	DDA0H				√	Undefined
FFFFF088H	DMA source address register 1L	DSA1L				√	Undefined
FFFFF08AH	DMA source address register 1H	DSA1H				√	Undefined
FFFFF08CH	DMA destination address register 1L	DDA1L				√	Undefined
FFFFF08EH	DMA destination address register 1H	DDA1H				√	Undefined
FFFFF090H	DMA source address register 2L	DSA2L				√	Undefined
FFFFF092H	DMA source address register 2H	DSA2H				√	Undefined
FFFFF094H	DMA destination address register 2L	DDA2L				√	Undefined
FFFFF096H	DMA destination address register 2H	DDA2H				√	Undefined
FFFFF098H	DMA source address register 3L	DSA3L				√	Undefined
FFFFF09AH	DMA source address register 3H	DSA3H				√	Undefined
FFFFF09CH	DMA destination address register 3L	DDA3L				√	Undefined
FFFFF09EH	DMA destination address register 3H	DDA3H				√	Undefined
FFFFF0C0H	DMA transfer count register 0	DBC0				√	Undefined
FFFFF0C2H	DMA transfer count register 1	DBC1				√	Undefined
FFFFF0C4H	DMA transfer count register 2	DBC2				√	Undefined
FFFFF0C6H	DMA transfer count register 3	DBC3				√	Undefined

Note CAN controller version only

(2/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF0D0H	DMA addressing control register 0	DADC0	R/W				0000H
FFFFF0D2H	DMA addressing control register 1	DADC1				√	0000H
FFFFF0D4H	DMA addressing control register 2	DADC2				√	0000H
FFFFF0D6H	DMA addressing control register 3	DADC3				√	0000H
FFFFF0E0H	DMA channel control register 0	DCHC0		√	√		00H
FFFFF0E2H	DMA channel control register 1	DCHC1		√	√		00H
FFFFF0E4H	DMA channel control register 2	DCHC2		√	√		00H
FFFFF0E6H	DMA channel control register 3	DCHC3		√	√		00H
FFFFF100H	Interrupt mask register 0	IMR0				√	FFFFH
FFFFF100H	Interrupt mask register 0L	IMR0L		√	√		FFH
FFFFF101H	Interrupt mask register 0H	IMR0H		√	√		FFH
FFFFF102H	Interrupt mask register 1	IMR1				√	FFFFH
FFFFF102H	Interrupt mask register 1L	IMR1L		√	√		FFH
FFFFF103H	Interrupt mask register 1H	IMR1H		√	√		FFH
FFFFF104H	Interrupt mask register 2	IMR2				√	FFFFH
FFFFF104H	Interrupt mask register 2L	IMR2L		√	√		FFH
FFFFF105H	Interrupt mask register 2H	IMR2H		√	√		FFH
FFFFF106H	Interrupt mask register 3	IMR3				√	FFFFH
FFFFF106H	Interrupt mask register 3L	IMR3L		√	√		FFH
FFFFF107H	Interrupt mask register 3H	IMR3H		√	√		FFH
FFFFF110H	Interrupt control register	LVIIC		√	√		47H
FFFFF112H	Interrupt control register	PIC0		√	√		47H
FFFFF114H	Interrupt control register	PIC1		√	√		47H
FFFFF116H	Interrupt control register	PIC2		√	√		47H
FFFFF118H	Interrupt control register	PIC3		√	√		47H
FFFFF11AH	Interrupt control register	PIC4		√	√		47H
FFFFF11CH	Interrupt control register	PIC5		√	√		47H
FFFFF11EH	Interrupt control register	PIC6		√	√		47H
FFFFF120H	Interrupt control register	PIC7		√	√		47H
FFFFF122H	Interrupt control register	TQ0OVIC		√	√		47H
FFFFF124H	Interrupt control register	TQ0CCIC0		√	√		47H
FFFFF126H	Interrupt control register	TQ0CCIC1		√	√		47H
FFFFF128H	Interrupt control register	TQ0CCIC2		√	√		47H
FFFFF12AH	Interrupt control register	TQ0CCIC3		√	√		47H
FFFFF12CH	Interrupt control register	TP0OVIC		√	√		47H
FFFFF12EH	Interrupt control register	TP0CCIC0		√	√		47H
FFFFF130H	Interrupt control register	TP0CCIC1		√	√		47H
FFFFF132H	Interrupt control register	TP1OVIC		√	√		47H
FFFFF134H	Interrupt control register	TP1CCIC0		√	√		47H
FFFFF136H	Interrupt control register	TP1CCIC1		√	√		47H
FFFFF138H	Interrupt control register	TP2OVIC		√	√		47H
FFFFF13AH	Interrupt control register	TP2CCIC0		√	√		47H
FFFFF13CH	Interrupt control register	TP2CCIC1		√	√		47H

(3/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF13EH	Interrupt control register	TP3OVIC	R/W	√	√		47H
FFFFF140H	Interrupt control register	TP3CCIC0		√	√		47H
FFFFF142H	Interrupt control register	TP3CCIC1		√	√		47H
FFFFF144H	Interrupt control register	TP4OVIC		√	√		47H
FFFFF146H	Interrupt control register	TP4CCIC0		√	√		47H
FFFFF148H	Interrupt control register	TP4CCIC1		√	√		47H
FFFFF14AH	Interrupt control register	TP5OVIC		√	√		47H
FFFFF14CH	Interrupt control register	TP5CCIC0		√	√		47H
FFFFF14EH	Interrupt control register	TP5CCIC1		√	√		47H
FFFFF150H	Interrupt control register	TM0EQIC0		√	√		47H
FFFFF152H	Interrupt control register	CB0RIC/IICIC1 Note 1		√	√		47H
FFFFF154H	Interrupt control register	CB0TIC		√	√		47H
FFFFF156H	Interrupt control register	CB1RIC		√	√		47H
FFFFF158H	Interrupt control register	CB1TIC		√	√		47H
FFFFF15AH	Interrupt control register	CB2RIC		√	√		47H
FFFFF15CH	Interrupt control register	CB2TIC		√	√		47H
FFFFF15EH	Interrupt control register	CB3RIC		√	√		47H
FFFFF160H	Interrupt control register	CB3TIC		√	√		47H
FFFFF162H	Interrupt control register	UA0RIC/CB4RIC		√	√		47H
FFFFF164H	Interrupt control register	UA0TIC/CB4TIC		√	√		47H
FFFFF166H	Interrupt control register	UA1RIC/IICIC2 Note 1		√	√		47H
FFFFF168H	Interrupt control register	UA1TIC		√	√		47H
FFFFF16AH	Interrupt control register	UA2RIC/IICIC0 Note 1		√	√		47H
FFFFF16CH	Interrupt control register	UA2TIC		√	√		47H
FFFFF16EH	Interrupt control register	ADIC		√	√		47H
FFFFF170H	Interrupt control register	DMAIC0		√	√		47H
FFFFF172H	Interrupt control register	DMAIC1		√	√		47H
FFFFF174H	Interrupt control register	DMAIC2		√	√		47H
FFFFF176H	Interrupt control register	DMAIC3		√	√		47H
FFFFF178H	Interrupt control register	KRIC		√	√		47H
FFFFF17AH	Interrupt control register	WTIIC		√	√		47H
FFFFF17CH	Interrupt control register	WTIC		√	√		47H
FFFFF17EH	Interrupt control register	ERRIC0 Note 2 / ERRIC Note 3		√	√		47H
FFFFF180H	Interrupt control register	WUPIC0 Note 2 / STAIC Note 3		√	√		47H
FFFFF182H	Interrupt control register	RECIC0 Note 2 / IEIC1 Note 3		√	√		47H
FFFFF184H	Interrupt control register	TRXIC0 Note 2 / IEIC2 Note 3		√	√		47H

Notes 1. I^2C bus version (Y version) only

2. CAN controller version only

3. IEBus controller version only

(4/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF1FAH	In-service priority register	ISPR	R	√	√		00H
FFFFF1FCH	Command register	PRCMD	W		√		Undefined
FFFFF1FEH	Power save control register	PSC	R/W	√	√		00H
FFFFF200H	A/D converter mode register 0	ADA0M0		√	√		00H
FFFFF201H	A/D converter mode register 1	ADA0M1		√	√		00H
FFFFF202H	A/D converter channel specification register	ADA0S		√	√		00H
FFFFF203H	A/D converter mode register 2	ADA0M2		√	√		00H
FFFFF204H	Power-fail compare mode register	ADA0PFM		√	√		00H
FFFFF205H	Power-fail compare threshold value register	ADA0PFT		√	√		00H
FFFFF210H	A/D conversion result register 0	ADA0CR0	R			√	Undefined
FFFFF211H	A/D conversion result register 0H	ADA0CRH0			√		Undefined
FFFFF212H	A/D conversion result register 1	ADA0CR1				√	Undefined
FFFFF213H	A/D conversion result register 1H	ADA0CRH1			√		Undefined
FFFFF214H	A/D conversion result register 2	ADA0CR2				√	Undefined
FFFFF215H	A/D conversion result register 2H	ADA0CRH2			√		Undefined
FFFFF216H	A/D conversion result register 3	ADA0CR3				√	Undefined
FFFFF217H	A/D conversion result register 3H	ADA0CRH3			√		Undefined
FFFFF218H	A/D conversion result register 4	ADA0CR4				√	Undefined
FFFFF219H	A/D conversion result register 4H	ADA0CRH4			√		Undefined
FFFFF21AH	A/D conversion result register 5	ADA0CR5				√	Undefined
FFFFF21BH	A/D conversion result register 5H	ADA0CRH5			√		Undefined
FFFFF21CH	A/D conversion result register 6	ADA0CR6				√	Undefined
FFFFF21DH	A/D conversion result register 6H	ADA0CRH6			√		Undefined
FFFFF21EH	A/D conversion result register 7	ADA0CR7				√	Undefined
FFFFF21FH	A/D conversion result register 7H	ADA0CRH7			√		Undefined
FFFFF220H	A/D conversion result register 8	ADA0CR8				√	Undefined
FFFFF221H	A/D conversion result register 8H	ADA0CRH8			√		Undefined
FFFFF222H	A/D conversion result register 9	ADA0CR9				√	Undefined
FFFFF223H	A/D conversion result register 9H	ADA0CRH9			√		Undefined
FFFFF224H	A/D conversion result register 10	ADA0CR10				√	Undefined
FFFFF225H	A/D conversion result register 10H	ADA0CRH10			√		Undefined
FFFFF226H	A/D conversion result register 11	ADA0CR11				√	Undefined
FFFFF227H	A/D conversion result register 11H	ADA0CRH11			√		Undefined
FFFFF280H	D/A converter conversion value setting register 0	DA0CS0	R/W		√		00H
FFFFF281H	D/A converter conversion value setting register 1	DA0CS1			√		00H
FFFFF282H	D/A converter mode register	DA0M		√	√		00H
FFFFF300H	Key return mode register	KRM		√	√		00H
FFFFF308H	Selector operation control register	SELCNT		√	√		00H
FFFFF310H	CRC input register	CRCIN			√		00H
FFFFF312H	CRC data register	CRCD				√	0000H
FFFFF318H	Noise elimination control register	NFC			√		00H
FFFFF320H	BRG1 prescaler mode register	PRSM1			√		00H

(5/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF321H	BRG1 prescaler compare register	PRSCM1	R/W		√		00H
FFFFF324H	BRG2 prescaler mode register	PRSM2			√		00H
FFFFF325H	BRG2 prescaler compare register	PRSCM2			√		00H
FFFFF328H	BRG3 prescaler mode register	PRSM3			√		00H
FFFFF329H	BRG3 prescaler compare register	PRSCM3			√		00H
FFFFF340H	IIC division clock select register	OCKS0 Note 1			√		00H
FFFFF344H	IIC division clock select register	OCKS1 Note 1			√		00H
FFFFF348H	IEBus clock select register	OCKS2 Note 2			√		00H
FFFFF360H	IEBus control register	BCR Note 2		√	√		00H
FFFFF361H	IEBus power save register	PSR Note 2		√	√		00H
FFFFF362H	IEBus slave status register	SSR Note 2	R		√		81H
FFFFF363H	IEBus unit status register	USR Note 2			√		00H
FFFFF364H	IEBus interrupt status register	ISR Note 2	R/W	√	√		00H
FFFFF365H	IEBus error status register	ESR Note 2		√	√		00H
FFFFF366H	IEBus unit address register	UAR Note 2				√	0000H
FFFFF368H	IEBus slave address register	SAR Note 2				√	0000H
FFFFF36AH	IEBus partner address register	PAR Note 2	R			√	0000H
FFFFF36CH	IEBus receive slave address register	RSA Note 2				√	0000H
FFFFF36EH	IEBus control data register	CDR Note 2	R/W		√		00H
FFFFF36FH	IEBus telegraph length register	DLR Note 2			√		01H
FFFFF370H	IEBus data register	DR Note 2			√		00H
FFFFF371H	IEBus field status register	FSR Note 2	R		√		00H
FFFFF372H	IEBus success count register	SCR Note 2			√		01H
FFFFF373H	IEBus communication count register	CCR Note 2			√		20H
FFFFF400H	Port 0	P0	R/W	√	√		Undefined
FFFFF402H	Port 1	P1		√	√		Undefined
FFFFF406H	Port 3	P3				√	Undefined
FFFFF406H	Port 3L	P3L		√	√		Undefined
FFFFF407H	Port 3H	P3H		√	√		Undefined
FFFFF408H	Port 4	P4		√	√		Undefined
FFFFF40AH	Port 5	P5		√	√		Undefined
FFFFF40EH	Port 7L	P7L		√	√		Undefined
FFFFF40FH	Port 7H	P7H		√	√		Undefined
FFFFF412H	Port 9	P9				√	Undefined
FFFFF412H	Port 9L	P9L		√	√		Undefined
FFFFF413H	Port 9H	P9H		√	√		Undefined
FFFFF420H	Port mode register 0	PM0		√	√		FFH
FFFFF422H	Port mode register 1	PM1		√	√		FFH
FFFFF426H	Port mode register 3	PM3				√	FFFFH
FFFFF426H	Port mode register 3L	PM3L		√	√		FFH
FFFFF427H	Port mode register 3H	PM3H		√	√		

Notes 1. I^2C bus version (Y version) only

2. IEBus controller version only

(6/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF428H	Port mode register 4	PM4	R/W	√	√		FFH
FFFFF42AH	Port mode register 5	PM5		√	√		FFH
FFFFF42EH	Port mode register 7L	PM7L		√	√		FFH
FFFFF42FH	Port mode register 7H	PM7H		√	√		FFH
FFFFF432H	Port mode register 9	PM9				√	FFFFH
FFFFF432H	Port mode register 9L	PM9L		√	√		FFH
FFFFF433H	Port mode register 9H	PM9H		√	√		FFH
FFFFF440H	Port mode control register 0	PMC0		√	√		00H
FFFFF446H	Port mode control register 3	PMC3				√	0000H
FFFFF446H	Port mode control register 3L	PMC3L		√	√		00H
FFFFF447H	Port mode control register 3H	PMC3H		√	√		00H
FFFFF448H	Port mode control register 4	PMC4		√	√		00H
FFFFF44AH	Port mode control register 5	PMC5		√	√		00H
FFFFF452H	Port mode control register 9	PMC9				√	0000H
FFFFF452H	Port mode control register 9L	PMC9L		√	√		00H
FFFFF453H	Port mode control register 9H	PMC9H		√	√		00H
FFFFF460H	Port function control register 0	PFC0		√	√		00H
FFFFF466H	Port function control register 3	PFC3				√	0000H
FFFFF466H	Port function control register 3L	PFC3L		√	√		00H
FFFFF467H	Port function control register 3H	PFC3H		√	√		00H
FFFFF468H	Port function control register 4	PFC4		√	√		00H
FFFFF46AH	Port function control register 5	PFC5		√	√		00H
FFFFF472H	Port function control register 9	PFC9				√	0000H
FFFFF472H	Port function control register 9L	PFC9L		√	√		00H
FFFFF473H	Port function control register 9H	PFC9H		√	√		00H
FFFFF484H	Data wait control register 0	DWC0				√	7777H
FFFFF488H	Address wait control register	AWC				√	FFFFH
FFFFF48AH	Bus cycle control register	BCC				√	AAAAH
FFFFF540H	TMQ0 control register 0	TQ0CTL0		√	√		00H
FFFFF541H	TMQ0 control register 1	TQ0CTL1		√	√		00H
FFFFF542H	TMQ0 I/O control register 0	TQ0IOC0		√	√		00H
FFFFF543H	TMQ0 I/O control register 1	TQ0IOC1		√	√		00H
FFFFF544H	TMQ0 I/O control register 2	TQ0IOC2		√	√		00H
FFFFF545H	TMQ0 option register	TQ0OPT0		√	√		00H
FFFFF546H	TMQ0 capture/compare register 0	TQ0CCR0				√	0000H
FFFFF548H	TMQ0 capture/compare register 1	TQ0CCR1				√	0000H
FFFFF54AH	TMQ0 capture/compare register 2	TQ0CCR2				√	0000H
FFFFF54CH	TMQ0 capture/compare register 3	TQ0CCR3				√	0000H
FFFFF54EH	TMQ0 read buffer register	TQ0CNT	R			√	0000H

(7/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF590H	TMP0 control register 0	TP0CTL0	R/W	√	√		00H
FFFFF591H	TMP0 control register 1	TP0CTL1		√	√		00H
FFFFF592H	TMP0 I/O control register 0	TP0IOC0		√	√		00H
FFFFF593H	TMP0 I/O control register 1	TP0IOC1		√	√		00H
FFFFF594H	TMP0 I/O control register 2	TP0IOC2		√	√		00H
FFFFF595H	TMQ0 option register	TP0OPT0		√	√		00H
FFFFF596H	TMQ0 capture/compare register 0	TP0CCR0				√	0000H
FFFFF598H	TMQ0 capture/compare register 1	TP0CCR1				√	0000H
FFFFF59AH	TMP0 counter register	TP0CNT	R			√	0000H
FFFFF5A0H	TMP1 control register 0	TP1CTL0	R/W	√	√		00H
FFFFF5A1H	TMP1 control register 1	TP1CTL1		√	√		00H
FFFFF5A2H	TMP1 I/O control register 0	TP1IOC0		√	√		00H
FFFFF5A3H	TMP1 I/O control register 1	TP1IOC1		√	√		00H
FFFFF5A4H	TMP1 I/O control register 2	TP1IOC2		√	√		00H
FFFFF5A5H	TMP1 option register	TP1OPT0		√	√		00H
FFFFF5A6H	TMP1 capture/compare register 0	TP1CCR0				√	0000H
FFFFF5A8H	TMP1 capture/compare register 1	TP1CCR1				√	0000H
FFFFF5AAH	TMP1 counter register	TP1CNT	R			√	0000H
FFFFF5B0H	TMP2 control register 0	TP2CTL0	R/W	√	√		00H
FFFFF5B1H	TMP2 control register 1	TP2CTL1		√	√		00H
FFFFF5B2H	TMP2 I/O control register 0	TP2IOC0		√	√		00H
FFFFF5B3H	TMP2 I/O control register 1	TP2IOC1		√	√		00H
FFFFF5B4H	TMP2 I/O control register 2	TP2IOC2		√	√		00H
FFFFF5B5H	TMP2 option register	TP2OPT0		√	√		00H
FFFFF5B6H	TMP2 capture/compare register 0	TP2CCR0				√	0000H
FFFFF5B8H	TMP2 capture/compare register 1	TP2CCR1				√	0000H
FFFFF5BAH	TMP2 counter register	TP2CNT	R			√	0000H
FFFFF5C0H	TMP3 control register 0	TP3CTL0	R/W	√	√		00H
FFFFF5C1H	TMP3 control register 1	TP3CTL1		√	√		00H
FFFFF5C2H	TMP3 I/O control register 0	TP3IOC0		√	√		00H
FFFFF5C3H	TMP3 I/O control register 1	TP3IOC1		√	√		00H
FFFFF5C4H	TMP3 I/O control register 2	TP3IOC2		√	√		00H
FFFFF5C5H	TMP3 option register	TP3OPT0		√	√		00H
FFFFF5C6H	TMP3 capture/compare register 0	TP3CCR0				√	0000H
FFFFF5C8H	TMP3 capture/compare register 1	TP3CCR1				√	0000H
FFFFF5CAH	TMP3 counter register	TP3CNT	R			√	0000H
FFFFF5D0H	TMP4 control register 0	TP4CTL0	R/W	√	√		00H
FFFFF5D1H	TMP4 control register 1	TP4CTL1		√	√		00H
FFFFF5D2H	TMP4 I/O control register 0	TP4IOC0		√	√		00H
FFFFF5D3H	TMP4 I/O control register 1	TP4IOC1		√	√		00H
FFFFF5D4H	TMP4 I/O control register 2	TP4IOC2		√	√		00H
FFFFF5D5H	TMP4 option register	TP4OPT0		√	√		00H
FFFFF5D6H	TMP4 capture/compare register 0	TP4CCR0				√	0000H
FFFFF5D8H	TMP4 capture/compare register 1	TP4CCR1				√	0000H

(8/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits				Default Value
				1	8	6	32	
FFFFF5DAH	TMP4 counter register	TP4CNT	R			√		0000H
FFFFF5E0H	TMP5 control register 0	TP5CTL0	R/W	√	√			00H
FFFFF5E1H	TMP5 control register 1	TP5CTL1		√	√			00H
FFFFF5E2H	TMP5 I/O control register 0	TP5IOC0		√	√			00H
FFFFF5E3H	TMP5 I/O control register 1	TP5IOC1		√	√			00H
FFFFF5E4H	TMP5 I/O control register 2	TP5IOC2		√	√			00H
FFFFF5E5H	TMP5 option register	TP5OPT0		√	√			00H
FFFFF5E6H	TMP5 capture/compare register 0	TP5CCR0				√		0000H
FFFFF5E8H	TMP5 capture/compare register 1	TP5CCR1				√		0000H
FFFFF5EAH	TMP5 counter register	TP5CNT	R			√		0000H
FFFFF680H	Watch timer operation mode register	WTM	R/W	√	√			00H
FFFFF690H	TMM0 control register 0	TM0CTL0		√	√			00H
FFFFF694H	TMM0 compare register 0	TM0CMP0				√		0000H
FFFFF6C0H	Oscillation stabilization time select register	OSTS			√			06H
FFFFF6C1H	PLL lockup time specification register	PLLS			√			03H
FFFFF6D0H	Watchdog timer mode register 2	WDTM2		√	√			67H
FFFFF6D1H	Watchdog timer enable register	WDTE			√			9AH
FFFFF6E0H	Real-time output buffer register 0L	RTBL0		√	√			00H
FFFFF6E2H	Real-time output buffer register 0H	RTBH0		√	√			00H
FFFFF6E4H	Real-time output port mode register 0	RTPM0		√	√			00H
FFFFF6E5H	Real-time output port control register 0	RTPC0		√	√			00H
FFFFF706H	Port function control expansion register 3L	PFCE3L		√	√			00H
FFFFF70AH	Port function control expansion register 5	PFCE5		√	√			00H
FFFFF712H	Port function control expansion register 9	PFCE9				√		0000H
FFFFF712H	Port function control expansion register 9L	PFCE9L		√	√			00H
FFFFF713H	Port function control expansion register 9H	PFCE9H		√	√			00H
FFFFF802H	System status register	SYS		√	√			00H
FFFFF80CH	Ring-OSC mode register	RCM		√	√			00H
FFFFF810H	DMA trigger factor register 0	DTFR0		√	√			00H
FFFFF812H	DMA trigger factor register 1	DTFR1		√	√			00H
FFFFF814H	DMA trigger factor register 2	DTFR2		√	√			00H
FFFFF816H	DMA trigger factor register 3	DTFR3		√	√			00H
FFFFF820H	Power save mode register	PSMR		√	√			00H
FFFFF822H	Clock control register	CKC		√	√			0AH
FFFFF824H	Lock register	LOCKR	R		√			00H
FFFFF828H	Processor clock control register	PCC	R/W	√	√			03H
FFFFF82CH	PLL control register	PLLCTL		√	√			01H
FFFFF82EH	CPU operation clock status registe	CCLS	R	√	√			00H
FFFFF840H	Correction address register 0	CORAD0	R/W				√	00000000H
FFFFF840H	Correction address register 0L	CORAD0L				√		0000H
FFFFF842H	Correction address register 0H	CORAD0H				√		0000H

(9/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits				Default Value
				1	8	16	32	
FFFFF844H	Correction address register 1	CORAD1	R/W				√	00000000H
FFFFF844H	Correction address register 1L	CORAD1L				√		0000H
FFFFF846H	Correction address register 1H	CORAD1H				√		0000H
FFFFF848H	Correction address register 2	CORAD2					√	00000000H
FFFFF848H	Correction address register 2L	CORAD2L				√		0000H
FFFFF84AH	Correction address register 2H	CORAD2H				√		0000H
FFFFF84CH	Correction address register 3	CORAD3					√	00000000H
FFFFF84CH	Correction address register 3L	CORAD3L				√		0000H
FFFFF84EH	Correction address register 3H	CORAD3H				√		0000H
FFFFF870H	Clock monitor mode register	CLM		√	√			00H
FFFFF880H	Correction control register	CORCN		√	√			0000H
FFFFF888H	Reset source flag register	RESF		√	√			00H
FFFFF890H	Low-voltage detection register	LVIM		√	√			00H
FFFFF891H	Low-voltage detection level select register	LVIS			√			00H
FFFFF892H	Internal RAM data status register	RAMS		√	√			01H
FFFFF8B0H	Prescaler mode register	PRSM0			√			0000H
FFFFF8B1H	Prescaler compare register	PRSCM0			√			00H
FFFFF9FCH	On-chip debug alternate-function pin setting register	OCDM[Note]		√	√			01H
FFFFFA00H	UARTA0 control register 0	UA0CTL0		√	√			00H
FFFFFA01H	UARTA0 control register 1	UA0CTL1			√			01H
FFFFFA02H	UARTA0 control register 2	UA0CTL2			√			10H
FFFFFA03H	UARTA0 option control register 0	UA0OPT0		√	√			00H
FFFFFA04H	UARTA0 status register	UA0STR		√	√			FFH
FFFFFA06H	UARTA0 receive data register	UA0RX	R		√			14H
FFFFFA07H	UARTA0 transmit data register	UA0TX	R/W		√			00H
FFFFFA10H	UARTA1 control register 0	UA1CTL0		√	√			FFH
FFFFFA11H	UARTA1 control register 1	UA1CTL1			√			FFH
FFFFFA12H	UARTA1 control register 2	UA1CTL2			√			10H
FFFFFA13H	UARTA1 option control register 0	UA1OPT0		√	√			00H
FFFFFA14H	UARTA1 status register	UA1STR		√	√			FFH
FFFFFA16H	UARTA1 receive data register	UA1RX	R		√			14H
FFFFFA17H	UARTA1 transmit data register	UA1TX	R/W		√			00H
FFFFFA20H	UARTA2 control register 0	UA2CTL0		√	√			FFH
FFFFFA21H	UARTA2 control register 1	UA2CTL1			√			FFH
FFFFFA22H	UARTA2 control register 2	UA2CTL2			√			10H
FFFFFA23H	UARTA2 option control register 0	UA2OPT0		√	√			00H
FFFFFA24H	UARTA2 status register	UA2STR		√	√			FFH
FFFFFA26H	UARTA2 receive data register	UA2RX	R		√			FFH
FFFFFA27H	UARTA2 transmit data register	UA2TX	R/W		√			14H
FFFFFC00H	External interrupt falling edge specification register 0	INTF0		√	√			00H
FFFFFC06H	External interrupt falling edge specification register 3L	INTF3L		√	√			00H
FFFFFC13H	External interrupt falling edge specification register 9H	INTF9H		√	√			00H

Note Flash memory version only

(10/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFFC20H	External interrupt rising edge specification register 0	INTR0	R/W	√	√		00H
FFFFFC26H	External interrupt rising edge specification register 3L	INTR3L		√	√		00H
FFFFFC33H	External interrupt rising edge specification register 9H	INTR9H		√	√		00H
FFFFFC60H	Port function control register 0	PF0		√	√		00H
FFFFFC66H	Port function control register 3	PF3				√	0000H
FFFFFC66H	Port function control register 3L	PF3L		√	√		00H
FFFFFC67H	Port function control register 3H	PF3H		√	√		00H
FFFFFC68H	Port function control register 4	PF4		√	√		00H
FFFFFC6AH	Port function control register 5	PF5		√	√		00H
FFFFFC72H	Port function control register 9	PF9				√	0000H
FFFFFC72H	Port function control register 9L	PF9L		√	√		00H
FFFFFC73H	Port function control register 9H	PF9H		√	√		00H
FFFFFD00H	CSIB0 control register 0	CB0CTL0		√	√		01H
FFFFFD01H	CSIB0 control register 1	CB0CTL1		√	√		00H
FFFFFD02H	CSIB0 control register 2	CB0CTL2			√		00H
FFFFFD03H	CSIB0 status register	CB0STR		√	√		00H
FFFFFD04H	CSIB0 receive data register	CB0RX	R			√	0000H
FFFFFD04H	CSIB0 receive data register L	CB0RXL			√		00H
FFFFFD06H	CSIB0 transmit data register	CB0TX	R/W			√	01H
FFFFFD06H	CSIB0 transmit data register L	CB0TXL			√		00H
FFFFFD10H	CSIB1 control register 0	CB1CTL0		√	√		00H
FFFFFD11H	CSIB1 control register 1	CB1CTL1		√	√		00H
FFFFFD12H	CSIB1 control register 2	CB1CTL2			√		0000H
FFFFFD13H	CSIB1 status register	CB1STR		√	√		00H
FFFFFD14H	CSIB1 receive data register	CB1RX	R			√	0000H
FFFFFD14H	CSIB1 receive data register L	CB1RXL			√		00H
FFFFFD16H	CSIB1 transmit data register	CB1TX	R/W			√	0000H
FFFFFD16H	CSIB1 transmit data register L	CB1TXL			√		00H
FFFFFD20H	CSIB2 control register 0	CB2CTL0		√	√		01H
FFFFFD21H	CSIB2 control register 1	CB2CTL1		√	√		00H
FFFFFD22H	CSIB2 control register 2	CB2CTL2			√		00H
FFFFFD23H	CSIB2 status register	CB2STR		√	√		00H
FFFFFD24H	CSIB2 receive data register	CB2RX	R			√	0000H
FFFFFD24H	CSIB2 receive data register L	CB2RXL			√		00H
FFFFFD26H	CSIB2 transmit data register	CB2TX	R/W			√	0000H
FFFFFD26H	CSIB2 transmit data register L	CB2TXL			√		00H
FFFFFD30H	CSIB3 control register 0	CB3CTL0		√	√		01H
FFFFFD31H	CSIB3 control register 1	CB3CTL1		√	√		00H
FFFFFD32H	CSIB3 control register 2	CB3CTL2			√		00H
FFFFFD33H	CSIB3 status register	CB3STR		√	√		00H
FFFFFD34H	CSIB3 receive data register	CB3RX	R			√	0000H
FFFFFD34H	CSIB3 receive data register L	CB3RXL			√		00H

(11/11)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFFD36H	CSIB3 transmit data register	CB3TX	R/W			√	0000H
FFFFFD36H	CSIB3 transmit data register L	CB3TXL			√		00H
FFFFFD40H	CSIB4 control register 0	CB4CTL0		√	√		01H
FFFFFD41H	CSIB4 control register 1	CB4CTL1		√	√		00H
FFFFFD42H	CSIB4 control register 2	CB4CTL2			√		00H
FFFFFD43H	CSIB4 status register	CB4STR		√	√		00H
FFFFFD44H	CSIB4 receive data register	CB4RX	R			√	0000H
FFFFFD44H	CSIB4 receive data register L	CB4RXL			√		00H
FFFFFD46H	CSIB4 transmit data register	CB4TX	R/W			√	0000H
FFFFFD46H	CSIB4 transmit data register L	CB4TXL			√		00H
FFFFFD80H	IIC shift register 0	IIC0Note			√		00H
FFFFFD82H	IIC control register 0	IICC0Note		√	√		00H
FFFFFD83H	Slave address register 0	SVA0Note			√		00H
FFFFFD84H	IIC clock select register 0	IICCL0Note		√	√		00H
FFFFFD85H	IIC function expansion register 0	IICX0Note		√	√		00H
FFFFFD86H	IIC status register 0	IICS0Note	R	√	√		00H
FFFFFD8AH	IIC flag register 0	IICF0Note	R/W	√	√		00H
FFFFFD90H	IIC shift register 1	IIC1Note			√		00H
FFFFFD92H	IIC control register 1	IICC1Note		√	√		00H
FFFFFD93H	Slave address register 1	SVA1Note			√		00H
FFFFFD94H	IIC clock select register 1	IICCL1Note		√	√		00H
FFFFFD95H	IIC function expansion register 1	IICX1Note		√	√		00H
FFFFFD96H	IIC status register 1	IICS1Note	R	√	√		00H
FFFFFD9AH	IIC flag register 1	IICF1Note	R/W	√	√		00H
FFFFFDA0H	IIC shift register 2	IIC2Note			√		00H
FFFFFDA2H	IIC control register 2	IICC2Note		√	√		00H
FFFFFDA3H	IIC slave address register 2	SVA2Note			√		00H
FFFFFDA4H	IIC clock select register 2	IICCL2Note		√	√		00H
FFFFFDA5H	IIC function expansion register 2	IICX2Note		√	√		00H
FFFFFDA6H	IIC status register 2	IICS2Note	R	√	√		00H
FFFFFDAAH	IIC flag register 2	IICF2Note	R/W	√	√		00H
FFFFFDBEH	External bus interface mode control register	EXIMC		√	√		00H

Note I^2C bus version (Y version) only

3.4.8 Programmable peripheral I/O registers

The peripheral I/O area select control register (BPC) is used for programmable peripheral I/O register area selection.

(1) Peripheral I/O area select control register (BPC)

The BPC register can be read or written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Default value
BPC	PA15	0	PA13	PA12	PA11	PA10	PA09	PA08	PA07	PA06	PA05	PA04	PA03	PA02	PA01	PA00	FFFFF064H	0000H

Bit Position	Bit Name	Function
15	PA15	Enables/disables usage of programmable peripheral I/O area
13 to 0	PA13 to PA00	Specify an address in programmable peripheral I/O area (correspond to A27 to A14, respectively).

PA15	Usage of programmable peripheral I/O area
0	Usage of programmable peripheral I/O area disabled
1	Usage of programmable peripheral I/O area disabled

Caution When setting the PA15 bit to 1, be sure to set the BPC register to 8FFBH.
When clearing the PA15 bit to 0, be sure to set the BPC register to 0000H.

For a list of the programmable peripheral I/O register areas, see **Table 19-17 Control Register Access Type** and **Table 19-18 Message Buffer Access Type**.

3.4.9 Special registers

Special registers are registers that are protected from being written with illegal data due to a program hang-up. V850ES/SG2 has the following eight (seven in mask ROM version) special registers.

- Power save control register (PSC)
- Clock control register (CKC)
- Processor clock control register (PCC)
- Clock monitor mode register (CLM)
- Reset source flag register (RESF)
- Low-voltage detection register (LVIM)
- Internal RAM data status register (RAMS)
- On-chip debug mode setting register (OCDM) (flash memory version)

In addition, a command register (PRCDM) is provided to protect against a write access to the special registers so that the application system does not inadvertently stop due to a program hang-up. A write access to the special registers is made in a specific sequence, and an illegal store operation is reported to the system status register (SYS).

(1) Setting data to special registers

Set data to the special registers in the following sequence.

<1> Disable DMA operation.
<2> Prepare data to be set to the special register in a general-purpose register.
<3> Write the data prepared in <2> to the command register (PRCMD).
<3> Write the setting data to the special register (by using the following instructions).
- Store instruction (ST/SST instruction)
- Bit manipulation instruction (SET1/CLR1/NOT1 instruction)

<5> to <9> Insert NOP instructions (5 instructions).
<10> Enable DMA operation if necessary.

[Example] With PSC register (setting standby mode)

```
   ST.B r11, PSMR[r0]    ; Set PSMR register (setting IDLE1, IDLE2, and software STOP modes).
<1>CLR1 0, DCHCn[r0]     ; Disable DMA operation.  n = 0 to 3
<2>MOV0x02, r10
<3>ST.B r10, PRCMD[r0]   ; Write PRCMD register.
<4>ST.B r10, PSC[r0]     ; Set PSC register.
<5>NOP                   ; Dummy instruction
<6>NOP                   ; Dummy instruction
<7>NOP                   ; Dummy instruction
<8>NOP                   ; Dummy instruction
<9>NOP                   ; Dummy instruction
<10>SET1 0, DCHCn[r0]    ; Enable DMA operation.  n = 0 to 3
(next instruction)
```

There is no special sequence to read a special register.

Cautions 1. When a store instruction is executed to store data in the command register, interrupts are not acknowledged. This is because it is assumed that steps <3> and <4> above are performed by successive store instructions. If another instruction is placed between <3> and <4>, and if an interrupt is acknowledged by that instruction, the above sequence may not be established, causing malfunction.

2. Although dummy data is written to the PRCMD register, use the same general-purpose register used to set the special register (<4> in Example) to write data to the PRCMD register (<3> in Example). The same applies when a general-purpose register is used for addressing.

3. Five NOP instructions or more must be inserted immediately after setting the IDLE1 mode, IDLE2 mode, or software STOP mode (by setting the STP bit of the PSC register to 1).

(2) Command register (PRCMD)

The command register (PRCMD) is an 8-bit register that protects the registers that may seriously affect the application system from being written, so that the system does not inadvertently stop due to a program hang-up. The first write access to a special register is valid after data has been written in advance to the PRCMD register. In this way, the value of the special register can be rewritten only in a specific sequence, so as to protect the register from an illegal write access.

The PRCMD register is write-only, in 8-bit units (undefined data is read when this register is read).

After reset: Undefined W Address: FFFFF1FCH

	7	6	5	4	3	2	1	0
PRCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0

(3) System status register (SYS)

Status flags that indicate the operation status of the overall system are allocated to this register.
This register can be read or written in 8-bit or 1-bit units.

After reset: 00H R/W Address: FFFFF802H

	7	6	5	4	3	2	1	<0>
SYS	0	0	0	0	0	0	0	PRERR

PRERR	Detects protection error
0	Protection error did not occur
1	Protection error occurred

The PRERR flag operates under the following conditions.

(a) Set condition (PRERR flag = 1)

(i) When data is written to a special register without writing anything to the PRCMD register (when <4> is executed without executing <3> in **3.4.9 (1) Setting data to special registers**)

(ii) When data is written to an on-chip peripheral I/O register other than a special register (including execution of a bit manipulation instruction) after writing data to the PRCMD register (if <4> in **3.4.9 (1) Setting data to special registers** is not the setting of a special register)

Remark Even if an on-chip peripheral I/O register is read (except by a bit manipulation instruction) between an operation to write the PRCMD register and an operation to write a special register, the PRERR flag is not set, and the set data can be written to the special register.

(b) Clear condition (PRERR flag = 0)

(i) When 0 is written to the PRERR flag of the SYS register

(ii) When the system is reset

Cautions 1. If 0 is written to the PRERR bit of the SYS register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is cleared to 0 (the write access takes precedence).

2. If data is written to the PRCMD register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is set to 1.

3.4.10 Notes

Be sure to set the following register first when using the V850ES/SG2.

- System wait control register (VSWC)
- On-chip debug mode register (OCDM) (flash memory version only)

After setting the OCDM register, set the VSWC register, and then set the other registers as necessary.

When using the external bus, set each pin to the control mode by using the port-related registers after setting the above register.

(1) System wait control register (VSWC)

The VSWC register controls wait of bus access to the on-chip peripheral I/O registers.

Three clocks are required to access an on-chip peripheral I/O register (without a wait cycle). The V850ES/SG2 requires wait cycles according to the operating frequency. Set the following value to the VSWC register in accordance with the frequency used.

The VSWC register can be read or written in 8-bit units (address: FFFFF06EH, default value: 77H).

Operating Frequency (f_{CLK})	Set Value of VSWC	Number of Waits
32 kHz ≤ f_{CLK} < 16.6 MHz	00H	0 (no wait)
16.6 MHz ≤ f_{CLK} ≤ 20 MHz	01H	1

Remark When any of the following registers is accessed, the register access is kept waiting if the hardware changes the contents of the register at the same time as the CPU accesses the register. Consequently, it may take a longer time than usual to access an on-chip peripheral I/O register.

Peripheral Function	Target Register Name
Timer P (n = 0 to 5)	TPnCCR0, TPnCCR1, TPnCNT
Timer Q	TQ0CCR0, TQ0CCR1, TQ0CCR2, TQ0CCR3, TQ0CNT
Watchdog timer 2	WDTM2
Real-time output function	RTBL0, RTBH0
I^2C bus (n = 0 to 2)	IICSn
A/D converter (n = 0 to 11)	ADAM0, ADA0CRn, ADA0CRnH
CRC function	CRCD
CAN controller	Each control register, each message buffer register

(2) On-chip debug mode register (OCDM) (flash memory version only)

The OCDM register is used to switch between the normal operation mode and the on-chip debug mode. This register is a special register (see **3.4.9 Special registers**). Writing is possible only using a specific sequence so as to not overwrite setting contents by mistake due to inadvertent program loops, etc.

When the OCDM0 bit is set to 1 and the $\overline{\text{DRST}}$ pin input is high level, the on-chip debug mode is selected. Since after reset the initial value of the OCDM0 bit is 1, when not using the on-chip debug function, it is necessary to clear (0) the OCDM0 bit and maintain the $\overline{\text{DRST}}$ pin at low level until the OCDM0 bit is cleared (the DRST pin has a pull-down resistor (30 kΩ typ.) in the buffer and therefore does not have to be fixed to low level from the external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0)).

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

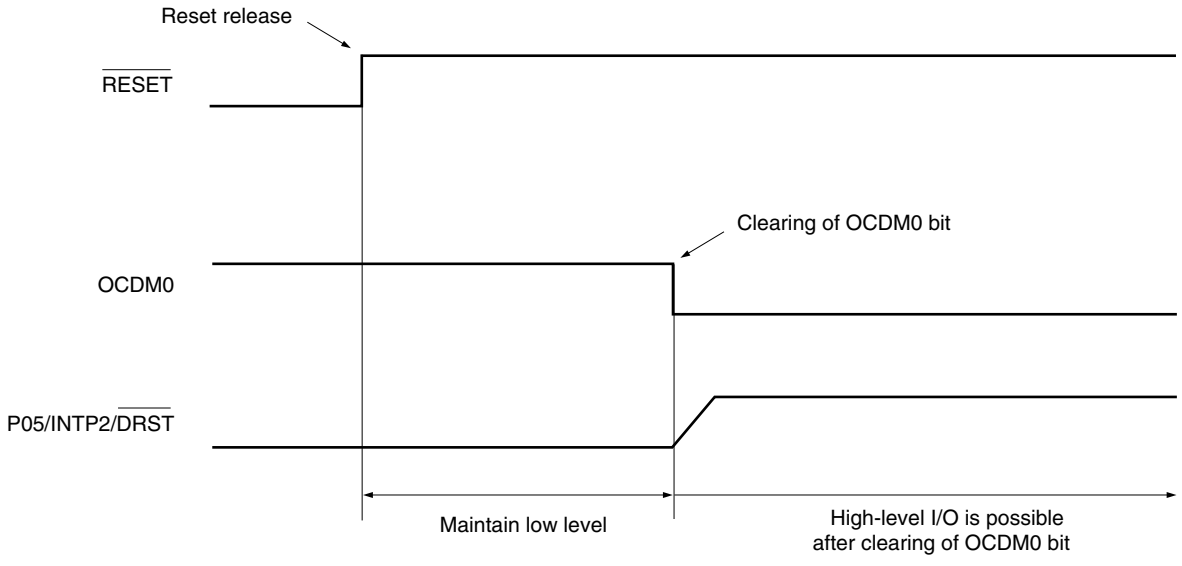
After reset: 01H[Note] R/W Address: FFFFF9FCH

	7	6	5	4	3	2	1	<0>
OCDM	0	0	0	0	0	0	0	OCDM0

OCDM0	Operation mode
0	Normal operation mode
1	$\overline{\text{DRST}}$ pin = low level: Normal operation mode $\overline{\text{DRST}}$ pin = high level: On-chip debug mode

Note This register holds the value 01H when reset is executed by the $\overline{\text{RESET}}$ pin, or the value of the OCDM register when reset is executed by the WDT2RES signal, clock monitor (CLM), or low-voltage detector (LVI).

Figure 3-15. Timing When On-Chip Debug Function Is Not Used



CHAPTER 4 PORT FUNCTIONS

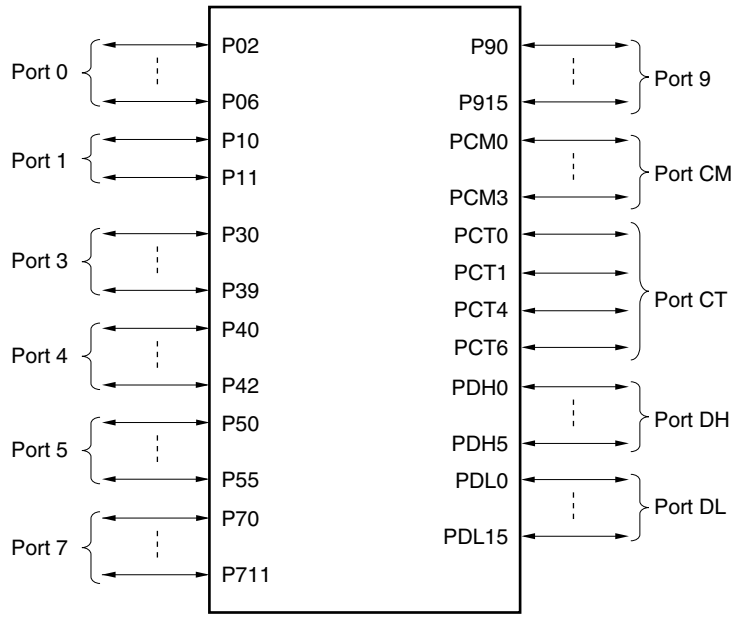
4.1 Features

- I/O ports: 84
- Other peripheral function I/O pins can be alternatively used
- Input/output specifiable in 1-bit units

4.2 Basic Port Configuration

The V850ES/SG2 features a total of 84 I/O ports consisting of ports 0, 1, 3, 4, 5, 7, 9, CM, CT, DH, and DL. The port configuration is shown below.

Figure 4-1. Port Configuration Diagram



Caution Ports 0, 3, 4, 5, and 9 are 5 V tolerant.

Table 4-1. I/O Buffer Power Supplies for Pins

Power Supply	Corresponding Pins
AV_{REF0}	Port 7
AV_{REF1}	Port 0
BV_{DD}	Port CM, port CT, port DH (bits 0 to 3), port DL
EV_{DD}	Port 0, port 3, port 4, port 5, port 9, port DH (bits 4, 5), $\overline{RESET}$

4.3 Port Configuration

Table 4-2. Port Configuration

Item	Configuration
Control register	Port mode register n (PMn: n = 0, 1, 3, 4, 5, 7, 9, CM, CT, DH, DL) Port mode control register n (PMCn: n = 0, 3, 4, 5, 9, CM, CT, DH, DL) Port function control register n (PFCn: n = 0, 3, 4, 5, 9) Port function control expansion register n (PFCEn: n = 3, 5, 9) Port function register n (PFn: n = 0, 3, 4, 5, 9) External interrupt falling edge specification register n (INTFn: n = 0, 3, 9) External interrupt rising edge specification register n (INTRn: n = 0, 3, 9)
Ports	I/O: 84

4.3.1 Notes on setting port pins

(1) Set the registers of a port in the following sequence.

<1> Set port function control register n (PFCn) and port function control expansion register n (PFCEn).
<2> Set port mode control register n (PMCn).
<3> Set external interrupt falling edge specification register n (INTFn) and external interrupt rising edge specification register n (INTRn).

If the PFCn and PFCEn registers are set after the PMCn register is set, an unexpected peripheral function may be selected while the PFCn and PFCEn registers are being set.

(2) The PFnm bit of the PFn register is valid only in the output mode (PMnm bit of PMn register = 0). In the input mode (PMn bit = 1), the value of the PFnm bit is not reflected in the buffer.

4.3.2 Port 0

Port 0 is a 5-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port 0 functions

- ○ Port input/output data specifiable in 1-bit units
  Specification made by port register 0 (P0)
- ○ Port input/output specifiable in 1-bit units
  Specification made by port mode register 0 (PM0)
- ○ Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register 0 (PMC0)
- ○ Control mode 1/control mode 2 specifiable in 1-bit units
  Specification made by port function control register 0 (PFC0)
- ○ N-ch open-drain specifiable in 1-bit units
  Specification made by port function register 0 (PF0)
- ○ Valid edge of external interrupts (alternate function) specifiable in 1-bit units
  Specification made by external interrupt falling edge specification register 0 (INTF0) and external interrupt rising edge specification register 0 (INTR0)

Port 0 includes the following alternate-function pins.

Table 4-3. Port 0 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 0	P02	NMI	I/O	Selectable as N-ch open-drain output
	P03	INTP0/ADTRG		
	P04	INTP1		
	P05	INTP2/$\overline{\text{DRST}}$Note		
	P06	INTP3		

Note The P05 pin's alternate function is a pin for on-chip debugging (flash memory version only). After external reset, the P05/INTP2/$\overline{\text{DRST}}$ pin is initialized to the on-chip debugging pin ($\overline{\text{DRST}}$). When using the P05 pin as a port pin and not as an on-chip debugging pin, the following handling is required.

<1> Clear (0) the OCDM0 bit of the OCDM register (special register).

If a high level is input to the $\overline{\text{DRST}}$ pin when the on-chip debug function is not used and before the above processing is performed, malfunction (CPU deadlock) may occur. Exercise care in handling the P05 pin. (The P05 pin has a pull-down resistor (30 kΩ typ.) in the buffer and does not have to be fixed to low level by an external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).)

(2) Registers

(a) Port register 0 (P0)

The P0 register is an 8-bit register that controls pin level read and output level write. This register can be read or written in 8-bit or 1-bit units.

After reset: Undefined R/W Address: FFFFF400H

	7	6	5	4	3	2	1	0
P0	0	P06	P05	P04	P03	P02	0	0

P0n	Output data control (in output mode) (n = 2 to 6)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port 0 (P0) is read, the pin levels at this time are read. Writing to P0 writes the values to that port. This does not affect the input pins.

Output mode: When port 0 (P0) is read, the P0 values are read. Writing to P0 writes the values to that port, and those values are immediately output.

2. The value of P0 is undefined (pin input level) when it is read in the input mode after reset. When P0 is read in the output mode, 00H (output latch value) is read.

(b) Port mode register 0 (PM0)

The PM0 register is an 8-bit register that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF420H

	7	6	5	4	3	2	1	0
PM0	1	PM06	PM05	PM04	PM03	PM02	1	1

PM0n	I/O mode control (n = 2 to 6)
0	Output mode
1	Input mode

(c) Port mode control register 0 (PMC0)

The PMC0 register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF440H

	7	6	5	4	3	2	1	0
PMC0	0	PMC06	PMC05	PMC04	PMC03	PMC02	0	0

PMC06	Specification of P06 pin operation mode
0	I/O port
1	INTP3 input

PMC05	Specification of P05 pin operation mode
0	I/O port
1	INTP2 input

PMC04	Specification of P04 pin operation mode
0	I/O port
1	INTP1 input

PMC03	Specification of P03 pin operation mode
0	I/O port
1	INTP0/ADTRG input

PMC02	Specification of P02 pin operation mode
0	I/O port
1	NMI input

Caution The P05/INTP2/$\overline{\text{DRST}}$ pin becomes the $\overline{\text{DRST}}$ pin regardless of the value of the PMC05 bit when the OCDM0 bit of the OCDM register = 1.

(d) Port function control register 0 (PFC0)

The PFC0 register is an 8-bit register that specifies control mode 1 or control mode 2.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF460H

	7	6	5	4	3	2	1	0
PFC0	0	0	0	0	PFC03	0	0	0

PFC03	Specification of P03 pin operation mode in control mode
0	INTP0 input
1	ADTRG input

(e) Port function register 0 (PF0)

The PF0 register is an 8-bit register that specifies normal output or N-ch open-drain output.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFFC60H

	7	6	5	4	3	2	1	0
PF0	0	PF06	PF05	PF04	PF03	PF02	0	0

PF0n	Control of normal output or N-ch open-drain output (n = 2 to 6)
0	Normal output
1	N-ch open drain output

(f) External interrupt falling edge specification register 0 (INTF0)

The INTF0 register is an 8-bit register that specifies detection of the falling edge for the external interrupt pin.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF0n bit = INTR0n bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.

However, noise elimination by digital delay is selectable for INTP3.

After reset: 00H R/W Address: FFFFFC00H

	7	6	5	4	3	2	1	0
INTF0	0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0

Remark For details on valid edge specification, see **Table 4-4**.

(g) External interrupt rising edge specification register 0 (INTR0)

The INTR0 register is an 8-bit register that specifies detection of the rising edge for the external interrupt pin.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF0n bit = INTR0n bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.
However, noise elimination by digital delay is selectable for INTP3.

After reset: 00H R/W Address: FFFFFC20H

	7	6	5	4	3	2	1	0
INTR0	0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0

Remark For details on valid edge specification, see **Table 4-4**.

Table 4-4. Valid Edge Specification

INTF0n	INTR0n	Valid Edge Specification (n = 2 to 6)
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Remark n = 2: Control of NMI pin
n = 3 to 6: Control of INTP0 to INTP3 pins

4.3.3 Port 1

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

(1) Port 1 functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register 1 (P1)
- Port input/output specification in 1-bit units
  Specification made by port mode register 1 (PM1)

Port 1 includes the following alternate-function pins.

Table 4-5. Port 1 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 1	P10	ANO0	Output	–
	P11	ANO1		–

(2) Registers

(a) Port register 1 (P1)

The P1 register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF402H

	7	6	5	4	3	2	1	0
P1	0	0	0	0	0	0	P11	P10

P1n	Output data control (in output mode) (n = 0, 1)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port 1 (P1) is read, the pin levels at this time are read. Writing to P1 writes the values to that port. This does not affect the input pins.

Output mode: When port 1 (P1) is read, the P1 values are read. Writing to P1 writes the values to that port, and those values are immediately output.

2. The value of P1 is undefined (pin input level) when it is read in the input mode after reset. When P1 is read in the output mode, 00H (output latch value) is read.

(b) Port mode register 1 (PM1)

The PM1 register is an 8-bit register that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF422H

	7	6	5	4	3	2	1	0
PM1	1	1	1	1	1	1	PM11	PM10

PM1n	I/O mode control (n = 0, 1)
0	Output mode
1	Input mode

Caution When using P10 and P11 as alternate functions (ANO0, ANO1), set the PM1 register = FFH.

4.3.4 Port 3

Port 3 is a 10-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port 3 functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register 3 (P3)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register 3 (PM3)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register 3 (PMC3)
- Control mode specifiable in 1-bit units
  Specification made by port function control register 3 (PFC3) and port function control expansion register 3L (PFCE3L)
- N-ch open-drain specifiable in 1-bit units
  Specification made by port function register 3 (PF3)
- Valid edge of external interrupts (alternate function) specifiable in 1-bit units
  Specification made by external interrupt falling edge specification register 3L (INTF3L) and external interrupt rising edge specification register 3L (INTR3L)

Port 3 includes the following alternate-function pins.

Table 4-6. Port 3 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 3	P30	TXDA0/SOB4	I/O	Selectable as N-ch open-drain output
	P31	RXDA0/INTP7/SIB4		
	P32	ASCKA0/$\overline{\text{SCKB4}}$/TIP00/TOP00		
	P33	TIP01/TOP01		
	P34	TIP10/TOP10		
	P35	TIP11/TOP11		
	P36	CTXD0[Note 1]/$\overline{\text{IETX0}}$[Note 2]		
	P37	CRXD0[Note 1]/$\overline{\text{IERX0}}$[Note 2]		
	P38	TXDA2/SDA00[Note 3]		
	P39	RXDA2/SCL00[Note 3]		

Notes 1. CAN controller version only

2. IEBus controller version only

3. I^2C bus version (Y version) only

(2) Registers

(a) Port register 3 (P3)

The P3 register is a 16-bit register that controls pin level read and output level write.

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

However, when using the higher 8 bits of the P3 register as the P3H register and the lower 8 bits as the P3L register, P3 can be read or written in 8-bit or 1-bit units.

After reset: Undefined R/W Address: P3 FFFFF406H, P3L FFFFF406H, P3H FFFF407H

	15	14	13	12	11	10	9	8
P3 (P3H[Note])	0	0	0	0	0	0	P39	P38
	7	6	5	4	3	2	1	0
(P3L)	P37	P36	P35	P34	P33	P32	P31	P30

P3n	Output data control (in output mode) (n = 0 to 9)
0	Outputs 0
1	Outputs 1

Note To read/write bits 8 to 15 of the P3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the P3H register.

Remarks 1. Input mode: When port 3 (P3) is read, the pin levels at this time are read. Writing to P3 writes the values to that port. This does not affect the input pins.

Output mode: When port 3 (P3) is read, the P3 values are read. Writing to P3 writes the values to that port, and those values are immediately output.

2. The value of P3 is undefined (pin input level) when it is read in the input mode after reset. When P3 is read in the output mode, 0000H (output latch value) is read.

(b) Port mode register 3 (PM3)

The PM3 register is a 16-bit register that specifies input or output mode.

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

However, when using the higher 8 bits of the PM3 register as the PM3H register and the lower 8 bits as the PM3L register, PM3 can be read or written in 8-bit or 1-bit units.

Reset input sets this register to FFFFH.

After reset: FFFFH R/W Address: PM3 FFFFF426H, PM3L FFFFF426H, PM3H FFFFF427H

	15	14	13	12	11	10	9	8
PM3 (PM3H[Note])	1	1	1	1	1	1	PM39	PM38
	7	6	5	4	3	2	1	0
(PM3L)	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30

PM3n	I/O mode control (n = 0 to 9)
0	Output mode
1	Input mode

Note To read/write bits 8 to 15 of the PM3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PM3H register.

(c) Port mode control register 3 (PMC3)

The PMC3 register is a 16-bit register that specifies port or control mode.

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

However, when using the higher 8 bits of the PMC3 register as the PMC3H register and the lower 8 bits as the PMC3L register, PMC3 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PMC3 FFFFF446H,
PMC3L FFFFF446H, PMC3H FFFFF447H

	15	14	13	12	11	10	9	8
PMC3 (PMC3H[Note])	0	0	0	0	0	0	PMC39	PMC38

	7	6	5	4	3	2	1	0
(PMC3L)	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

PMC39	Specification of P39 pin operation mode in control mode
0	I/O port
1	RXDA2/SCL00 I/O

PMC38	Specification of P38 pin operation mode in control mode
0	I/O port
1	TXDA2/SDA00 I/O

PMC37	Specification of P37 pin operation mode in control mode
0	I/O port
1	CRXD0/$\overline{\text{IERX0}}$ input

PMC36	Specification of P36 pin operation mode in control mode
0	I/O port
1	CTXD0/$\overline{\text{IETX0}}$ output

PMC35	Specification of P35 pin operation mode in control mode
0	I/O port
1	TIP11/TOP11 I/O

PMC34	Specification of P34 pin operation mode in control mode
0	I/O port
1	TIP10/TOP10 I/O

PMC33	Specification of P33 pin operation mode in control mode
0	I/O port
1	TIP01/TOP01 I/O

PMC32	Specification of P32 pin operation mode in control mode
0	I/O port
1	ASCKA0/$\overline{\text{SCKB4}}$/TIP00/TOP00 I/O

PMC31	Specification of P31 pin operation mode in control mode
0	I/O port
1	RXDA0/SIB4/INTP7 input

PMC30	Specification of P30 pin operation mode in control mode
0	I/O port
1	TXDA0/SOB4 output

Note To read/write bits 8 to 15 of the PMC3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMC3H register.

(d) Port function control register 3 (PFC3)

The PFC3 register is a 16-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

However, when using the higher 8 bits of the PFC3 register as the PFC3H register and the lower 8 bits as the PFC3L register, PFC3 can be read or written in 8-bit and 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PFC3 FFFFF466H, PFC3L FFFFF466H, PFC3L FFFFF467H

	15	14	13	12	11	10	9	8
PFC3 (PFC3H[Note])	0	0	0	0	0	0	PFC39	PFC38
	7	6	5	4	3	2	1	0
(PFC3L)	PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

Note To read/write bits 8 to 15 of the PFC3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFC3H register.

Remark For details on control mode specification, see **4.3.4 (2) (f) P3 pin control mode settings**.

(e) Port function control expansion register 3L (PFCE3L)

The PFCE3L register is an 8-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF706H

	7	6	5	4	3	2	1	0
PFCE3L	0	0	0	0	0	PFCE32	0	0

Remark For details on control mode specification, see **4.3.4 (2) (f) P3 pin control mode settings**.

(f) P3 pin control mode settings

PFC39	Specification of P39 Pin Control Mode
0	RXDA2 input
1	SCL00 input

PFC38	Specification of P38 Pin Control Mode
0	TXDA2 output
1	SDA00 I/O

PFC37	Specification of P37 Pin Control Mode
0	CRXD0 input
1	$\overline{\text{IERX0}}$ input

PFC36	Specification of P36 Pin Control Mode
0	CTXD0 output
1	$\overline{\text{IETX0}}$ output

PFC35	Specification of P35 Pin Control Mode
0	TIP11 input
1	TOP11 output

PFC34	Specification of P34 Pin Control Mode
0	TIP10 input
1	TOP10 output

PFC33	Specification of P33 Pin Control Mode
0	TIP01 input
1	TOP01 output

PFCE32	PFC32	Specification of P32 Pin Control Mode
0	0	ASCKA0 input
0	1	$\overline{\text{SCKB4}}$ I/O
1	0	TIP00 input
1	1	TOP00 output

PFC31	Specification of P31 Pin Control Mode
0	RXDA0/INTP7[Note] input
1	SIB4 input

PFC30	Specification of P30 Pin Control Mode
0	TXDA0 output
1	SOB4 output

Note The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin. (Clear the INTF31 bit of the INTF3L register and the INTR31 bit of the INTR3L register to 0.) When using the pin as the INTP7 pin, stop UARTA0 reception. (Clear the UA0RXE bit of the UA0CTL0 register to 0.)

(g) Port function register 3 (PF3)

The PF3 register is a 16-bit register that specifies normal output or N-ch open-drain output.

This register can be read or written in 16-bit units. However, when using the higher 8 bits of the PF3 register as the PF3H register and the lower 8 bits as the PF3L register, PF3 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PF3 FFFFFC66H, PF3L FFFFFC66H, PF3H FFFFFC67H

	15	14	13	12	11	10	9	8
PF3 (PF3H[Note])	0	0	0	0	0	0	PF39	PF38

	7	6	5	4	3	2	1	0
(PF3L)	PF37	PF36	PF35	PF34	PF33	PF32	PF31	PF30

PF3n	Control of normal output or N-ch open-drain output (n = 0 to 9)
0	Normal output
1	N-ch open-drain output

Note To read/write bits 8 to 15 of the PF3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PF3H register.

(h) External interrupt falling edge specification register 3L (INTF3L)

The INTF3L register is an 8-bit register that specifies detection of the falling edge for the external interrupt pin.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF31 bit = INTR31 bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.

After reset: 00H R/W Address: FFFFFC06H

	7	6	5	4	3	2	1	0
INTF3L	0	0	0	0	0	0	INTF31	0

Remark For details on valid edge specification, see **Table 4-7**.

(i) External interrupt rising edge specification register 3L (INTR3L)

The INTR3L register is an 8-bit register that specifies detection of the rising edge for the external interrupt pin.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF31 bit = INTR31 bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.

After reset: 00H R/W Address: INTR3: FFFFFC26H

	7	6	5	4	3	2	1	0
INTR3L	0	0	0	0	0	0	INTR31	0

Remark For details on valid edge specification, see **Table 4-7**.

Table 4-7. Valid Edge Specification

INTF31	INTR31	Valid Edge Specification
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution When not using the pin as INTP7 pin, be sure to set INTF31 bit = INTR31 bit = 0.

4.3.5 Port 4

Port 4 is a 3-bit port that controls I/O in 1-bit units.

(1) Port 4 functions

- Port input/output specifiable in 1-bit units
  Specification made by port register 4 (P4)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register 4 (PM4)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register 4 (PMC4)
- Control mode 1/control mode 2 specifiable in 1-bit units
  Specification made by port function control register 4 (PFC4)
- N-ch open-drain specifiable in 1-bit units
  Specification made by port function register 4 (PF4)

Port 4 includes the following alternate-function pins.

Table 4-8. Port 4 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 4	P40	SIB0/SDA01[Note]	I/O	Selectable as N-ch open-drain output
	P41	SOB0/SCL01[Note]		
	P42	$\overline{\text{SCKB0}}$		

Note I^2C bus version (Y version) only

(2) Registers

(a) Port register (P4)

The P4 register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF408H

	7	6	5	4	3	2	1	0
P4	0	0	0	0	0	P42	P41	P40

P4n	Output data control (in output mode) (n = 0 to 2)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port 4 (P4) is read, the pin levels at this time are read. Writing to P4 writes the values to that port. This does not affect the input pins.

Output mode: When port 4 (P4) is read, the P4 values are read. Writing to P4 writes the values to that port, and those values are immediately output.

2. The value of P4 is undefined (pin input level) when it is read in the input mode after reset. When P4 is read in the output mode, 00H (output latch value) is read.

(b) Port mode register 4 (PM4)

The PM4 register is an 8-bit register that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF428H

	7	6	5	4	3	2	1	0
PM4	1	1	1	1	1	PM42	PM41	PM40

PM4n	I/O mode control (n = 0 to 2)
0	Output mode
1	Input mode

(c) Port mode control register 4 (PMC4)

The PMC4 register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

PMC42	Specification of P42 pin operation mode
0	I/O port
1	$\overline{\text{SCKB0}}$ I/O

PMC41	Specification of P41 pin operation mode
0	I/O port
1	SCB0/SCL01 I/O

PMC40	Specification of P40 pin operation mode
0	I/O port
1	SB0/SDA01 I/O

(d) Port function control register 4 (PFC4)

The PFC4 register is an 8-bit register that specifies control mode 1 or control mode 2.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PFC41	Specification of P41 pin operation mode in control mode
0	SCB0 output
1	SCL01 I/O

PFC40	Specification of P40 pin operation mode in control mode
0	SIB0 input
1	SDA01 I/O

(e) Port function register 4 (PF4)

The PF4 register is an 8-bit register that specifies normal output or N-ch open-drain output.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFFC68H

	7	6	5	4	3	2	1	0
PF4	0	0	0	0	0	PF42	PF41	PF40

PF4n	Control of normal output or N-ch open-drain output (n = 0 to 2)
0	Normal output
1	N-ch open-drain output

4.3.6 Port 5

Port 5 is a 6-bit port that controls I/O in 1-bit units.

(1) Port 5 functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register 5 (P5)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register 5 (PM5)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register 5 (PMC5)
- Control mode specifiable in 1-bit units
  Specification made by port function control register 5 (PFC5) and port function control expansion register 5 (PFCE5)
- N-ch open-drain specifiable in 1-bit units
  Specification made by port function register 5 (PF5)

Port 5 includes the following alternate-function pins.

Table 4-9. Port 0 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 5	P50	TIQ01/KR0/TOQ01/RTP00	I/O	Selectable as N-ch open-drain output
	P51	TIQ02/KR1/TOQ02/RTP01		
	P52	TIQ03/KR2/TOQ03/RTP02/DDI[Note]		
	P53	SIB2/KR3/TIQ00/TOQ00/RTP03/DDO[Note]		
	P54	SOB2/KR4/RTP04/DCK[Note]		
	P55	$\overline{\text{SCKB2}}$/KR5/RTP05/DMS[Note]		

Note The DDI, DDO, DCK, and DMS pins are pins for on-chip debugging (flash memory version only). After external reset, when not using these pins as on-chip debugging pins and using them instead as a port, the following handling is required.

- Input a low level to the P05/INTP2/$\overline{\text{DRST}}$ pin.
- Set the ODCM0 bit of the ODCM register (special register). The handling in this case is as follows.
  <1> Clear (0) the OCDM0 bit of the OCDM register.
  <2> Fix the P05/INTP2/$\overline{\text{DRST}}$ pin input to low level until the handling in <1> above is completed.

If a high level is input to the $\overline{\text{DRST}}$ pin when the on-chip debug function is not used and before the above processing is performed, malfunction (CPU deadlock) may occur. Exercise care in handling the P05 pin. (The P05 pin has a pull-down resistor (30 kΩ typ.) in the buffer and does not have to be fixed to low level by an external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).)

(2) Registers

(a) Port register 5 (P5)

The P5 register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF40AH

	7	6	5	4	3	2	1	0
P5	0	0	P55	P54	P53	P52	P51	P50

P5n	Output data control (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port 5 (P5) is read, the pin levels at this time are read. Writing to P5 writes the values to that port. This does not affect the input pins.

Output mode: When port 5 (P5) is read, the P5 values are read. Writing to P5 writes the values to that port, and those values are immediately output.

2. The value of P5 is undefined (pin input level) when it is read in the input mode after reset. When P5 is read in the output mode, 00H (output latch value) is read.

(b) Port mode register 5 (PM5)

The PM5 register is an 8-bit register that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF42AH

	7	6	5	4	3	2	1	0
PM5	1	1	PM55	PM54	PM53	PM52	PM51	PM50

PM5n	I/O mode control (n = 0 to 5)
0	Output mode
1	Input mode

(c) Port mode control register 5 (PMC5)

The PMC5 register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF44AH

	7	6	5	4	3	2	1	0
PMC5	0	0	PMC55	PMC54	PMC53	PMC52	PMC51	PMC50

PMC55	Specification of P55 pin operation mode
0	I/O port
1	$\overline{\text{SCKB2}}$/KR5/RTP05 I/O

PMC54	Specification of P54 pin operation mode
0	I/O port
1	SOB2/KR4/RTP04 I/O

PMC53	Specification of P53 pin operation mode
0	I/O port
1	SB2/KR3/TIQ00/TOQ00/RTP03 I/O

PMC52	Specification of P52 pin operation mode
0	I/O port
1	TIQ03/KR2/TOQ03/RTP02 I/O

PMC51	Specification of P51 pin operation mode
0	I/O port
1	TIQ02/KR1/TOQ02/RTP01 I/O

PMC50	Specification of P50 pin operation mode
0	I/O port
1	TIQ01/KR0/TOQ01/RTP00 I/O

(d) Port function control register 5 (PFC5)

The PFC5 register is an 8-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF46AH

	7	6	5	4	3	2	1	0
PFC5	0	0	PFC55	PFC54	PFC53	PFC52	PFC51	PFC50

Remark For details on control mode specification, see **4.3.6 (2) (f) P5 pin control mode settings**.

(e) Port function control expansion register 5 (PFCE5)

The PFCE5 register is an 8-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF70AH

	7	6	<5>	<4>	<3>	<2>	<1>	<0>
PFCE5	0	0	PFCE55	PFCE54	PFCE53	PFCE52	PFCE51	PFCE50

Remark For details on control mode specification, see **4.3.6 (2) (f) P5 pin control mode settings**.

(f) P5 pin control mode settings

PFCE55	PFC55	Specification of P55 Pin Control Mode
0	0	$\overline{\text{SCKB2}}$ I/O
0	1	KR5 input
1	0	Setting prohibited
1	1	RTP05 output

PFCE54	PFC54	Specification of P54 Pin Control Mode
0	0	SOB2 output
0	1	KR4 input
1	0	Setting prohibited
1	1	RTP04 output

PFCE53	PFC53	Specification of P53 Pin Control Mode
0	0	SIB2 input
0	1	TIQ00/KR3[Note] input
1	0	TOQ00 output
1	1	RTP03 output

PFCE52	PFC52	Specification of P52 Pin Control Mode
0	0	Setting prohibited
0	1	TIQ03/KR2[Note] input
1	0	TOQ03 input
1	1	RTP02 output

PFCE51	PFC51	Specification of P51 Pin Control Mode
0	0	Setting prohibited
0	1	TIQ02/KR1[Note] input
1	0	TOQ02 output
1	1	RTP01 output

PFCE50	PFC50	Specification of P50 Pin Control Mode
0	0	Setting prohibited
0	1	TIQ01/KR0[Note] input
1	0	TOQ01 output
1	1	RTP00 output

Note The KRn pin and TIQ0m pin are alternate-function pins. When using the pin as the TIQ0m pin, disable KRn pin key return detection, which is the alternate function. (Clear the KRMn bit of the KRM register to 0.) Also, when using the pin as the KRn pin, disable TIQ0m pin edge detection, which is the alternate function (n = 0 to 3, m = 0 to 3).

Pin Name	Use as TIQ0m Pin	Use as KRn Pin
KR0/TIQ01	KRM0 bit of KRM register = 0	TQ0TIG2, TQ0TIG3 bits of TQ0IOC1 register = 0
KR1/TIQ02	KRM1 bit of KRM register = 0	TQ0TIG4, TQ0TIG5 bits of TQ0IOC1 register = 0
KR2/TIQ03	KRM2 bit of KRM register = 0	TQ0TIG6, TQ0TIG7 bits of TQ0IOC1 register = 0
KR3/TIQ00	KRM3 bit of KRM register = 0	TQ0TIG0, TQ0TIG1 bits of TQ0IOC1 register = 0 TQ0EES0, TQ0EES1 bits of TQ01OC2 register = 0 TQ0ETS0, TQ0ETS1 bits of TQ01OC2 register = 0

(g) Port function register 5 (PF5)

The PF5 register is an 8-bit register that specifies normal output or N-ch open-drain output.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFFC6AH

	7	6	5	4	3	2	1	0
PF5	0	0	PF55	PF54	PF53	PF52	PF51	PF50

PF5n	Control of normal output or N-ch open-drain output (n = 0 to 5)
0	Normal output
1	N-ch open-drain output

4.3.7 Port 7

Port 7 is a 12-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port 7 functions

- Port input/output data specifiable in 1-bit units
  Specification made by port 7 register (P7)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register 7 (PM7)

Port 7 includes the following alternate-function pins.

Table 4-10. Port 7 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 7	P70	ANI0	I/O	–
	P71	ANI1		
	P72	ANI2		
	P73	ANI3		
	P74	ANI4		
	P77	ANI5		
	P76	ANI6		
	P77	ANI7		
	P78	ANI8		
	P79	ANI9		
	P710	ANI10		
	P711	ANI11		

(2) Registers

(a) Port register 7H, port register 7L (P7H, P7L)

The P7H and P7L registers are 8-bit registers that control pin level read and output level write.

These registers can be read or written in 8-bit or 1-bit units.

16-bit access is not possible.

After reset: Undefined R Address: P7H FFFFF40EH, P7L FFFFF40FH

	7	6	5	4	3	2	1	0
P7H	0	0	0	0	P711	P710	P79	P78

	7	6	5	4	3	2	1	0
P7L	P77	P76	P75	P74	P73	P72	P71	P70

P7n	Output data control (in output mode) (n = 0 to 11)
0	Outputs 0
1	Outputs 1

Caution Do not read the P7H and P7L registers during A/D conversion.

Remarks 1. Input mode: When ports 7H and 7L (P7H, P7L) are read, the pin levels at this time are read. Writing to P7H and P7L writes the values to that port. This does not affect the input pins.

Output mode: When ports 7H and 7L (P7H, P7L) are read, the P7H and P7L values are read. Writing to P7H and P7L writes the values to that port, and those values are immediately output.

2. The values of P7H and P7H are undefined (pin input level) when they are read in the input mode after reset. When P7H and P7L are read in the output mode, 00H (output latch value) is read.

(b) Port mode register 7H, port mode register 7L (PM7H, PM7L)

The PM7H and PM7L registers are 8-bit registers that specify input or output mode.

These registers can be read or written in 8-bit or 1-bit units.

16-bit access is not possible.

Reset input sets these registers to FFH.

After reset: FFH R/W Address: PM7H FFFFF42EH, PM7L FFFFF42FH

	7	6	5	4	3	2	1	0
PM7H	1	1	1	1	PM711	PM710	PM79	PM78
PM7L	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70

PM7n	I/O mode control (n = 0 to 11)
0	Output mode
1	Input mode

Caution When using P7n as its alternate function (ANIn), set PM7n to 1.

4.3.8 Port 9

Port 9 is a 16-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port 9 functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register 9 (P9)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register 9 (PM9)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register 9 (PMC9)
- Control mode specifiable in 1-bit units
  Specification made by port function control register 9 (PFC9) and port function control expansion register 9 (PFCE9)
- N-ch open-drain specifiable in 1-bit units
  Specification made by port function register 9 (PF9)
- External interrupt (alternate function) valid edge specifiable in 1-bit units
  Specification made by external interrupt falling edge specification register 9H (INTF9H) and external interrupt rising edge specification register 9H (INTR9H)

Port 9 includes the following alternate-function pins.

Table 4-11. Port 9 Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port 9	P90	A0/KR6/TDXA1/SDA02[Note]	I/O	Selectable as N-ch open-drain output
	P91	A1/KR7/RXDA1/SCL02[Note]		
	P92	A2/TIP41/TOP41		
	P93	A3/TIP40/TOP40		
	P94	A4/TIP31/TOP31		
	P95	A5/TIP30/TOP30		
	P96	A6/TIP21/TOP21		
	P97	A7/SIB1/TIP20/TOP20		
	P98	A8/SOB1		
	P99	A9/$\overline{\text{SCKB1}}$		
	P910	A10/SIB3		
	P911	A11/SOB3		
	P912	A12/$\overline{\text{SCKB3}}$		
	P913	A13/INTP4		
	P914	A14/INTP5/TIP51/TOP51		
	P915	A15/INTP6/TIP50/TOP50		

Note I^2C bus version (Y version) only

(2) Registers

(a) Port register 9 (P9)

The P9 register is a 16-bit register that controls pin level read and output level write.

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

However, when using the higher 8 bits of the P9 register as the P9H register and the lower 8 bits as the P9L register, P9 can be read or written in 8-bit or 1-bit units.

After reset: Undefined R/W Address: P9 FFFFF412H,
P9L FFFFF412H, P9H FFFFF413H

	15	14	13	12	11	10	9	8
P9 (P9H[Note])	P915	P914	P913	P912	P911	P910	P99	P98

	7	6	5	4	3	2	1	0
(P9L)	P97	P96	P95	P94	P93	P92	P91	P90

P9n	Output data control (in output mode) (n = 0 to 15)
0	Outputs 0
1	Outputs 1

Note To read/write bits 8 to 15 of the P9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the P9H register.

Remarks 1. Input mode: When port 9 (P9) is read, the pin levels at this time are read. Writing to P9 writes the values to that port. This does not affect the input pins.

Output mode: When port 9 (P9) is read, the P9 values are read. Writing to P9 writes the values to that port, and those values are immediately output.

2. The value of P9 is undefined (pin input level) when it is read in the input mode after reset. When P9 is read in the output mode, 0000H (output latch value) is read.

(b) Port mode register 9 (PM9)

The PM9 register is a 16-bit register that specifies input or output mode.

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

However, when using the higher 8 bits of the PM9 register as the PM9H register and the lower 8 bits as the PM9L register, PM9 can be read or written in 8-bit and 1-bit units.

Reset input sets this register to FFFFH.

After reset: FFFFH R/W Address: PM9 FFFFF432H, PM9L FFFFF432H, PM9H FFFFF433H

	15	14	13	12	11	10	9	8
PM9 (PM9H[Note])	PM915	PM914	PM913	PM912	PM911	PM910	PM99	PM98

	7	6	5	4	3	2	1	0
(PM9L)	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90

PM9n	I/O mode control (n = 0 to 15)
0	Output mode
1	Input mode

Note To read/write bits 8 to 15 of the PM9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PM9H register.

(c) Port mode control register 9 (PMC9)

The PMC9 register is a 16-bit register that specifies port or control mode.

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

However, when using the higher 8 bits of the PMC9 register as the PMC9H register and the lower 8 bits as the PMC9L register, PMC9 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

(1/2)

After reset: 0000H R/W Address: PMC9 FFFFF452H,
PMC9L FFFFF452H, PMC9H FFFFF453H

	15	14	13	12	11	10	9	8
PMC9 (PMC9H[Note])	PMC915	PMC914	PMC913	PMC912	PMC911	PMC910	PMC99	PMC98

	7	6	5	4	3	2	1	0
(PMC9L)	PMC97	PMC96	PMC95	PMC94	PMC93	PMC92	PMC91	PMC90

PMC915	Specification of P915 pin operation mode
0	I/O port
1	A15/INTP6/TIP50/TOP50 I/O

PMC914	Specification of P914 pin operation mode
0	I/O port
1	A14/INTP5/TIP51/TOP51 I/O

PMC913	Specification of P913 pin operation mode
0	I/O port
1	A13/INTP4 I/O

PMC912	Specification of P912 pin operation mode
0	I/O port
1	A12/$\overline{\text{SCKB3}}$ I/O

PMC911	Specification of P911 pin operation mode
0	I/O port
1	A11/SOB3 I/O

PMC910	Specification of P910 pin operation mode
0	I/O port
1	A10/SIB3 I/O

PMC99	Specification of P99 pin operation mode
0	I/O port
1	A9/$\overline{\text{SCKB1}}$ I/O

Note To read/write bits 8 to 15 of the PMC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMC9H register.

(2/2)

PMC98	Specification of P98 pin operation mode
0	I/O port
1	A8/SOB1 output

PMC97	Specification of P97 pin operation mode
0	I/O port
1	A7/SIB1/TIP20/TOP20 I/O

PMC96	Specification of P96 pin operation mode
0	I/O port
1	A6/TIP21/TOP21 output

PMC95	Specification of P95 pin operation mode
0	I/O port
1	A5/TIP30/TOP30 I/O

PMC94	Specification of P94 pin operation mode
0	I/O port
1	A4/TIP31/TOP31 I/O

PMC93	Specification of P93 pin operation mode
0	I/O port
1	A3/TIP40/TOP40 I/O

PMC92	Specification of P92 pin operation mode
0	I/O port
1	A2/TIP41/TOP41 I/O

PMC91	Specification of P91 pin operation mode
0	I/O port
1	A1/KR7/RXDA1/SCL02 I/O

PMC90	Specification of P90 pin operation mode
0	I/O port
1	A0/KR6/TXDA1/SDA02 I/O

Caution Only when using the A0 to A15 pins as the alternate functions of the P90 to P915 pins, set all 16 bits of the PMC9 register to FFFFH at once.

(d) Port function control register 9 (PFC9)

The PFC9 register is a 16-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

However, when using the higher 8 bits of the PFC9 register as the PFC9H register and the lower 8 bits as the PFC9L register, PFC9 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

Caution When performing separate address bus output (A0 to A15), set the PMC9 register to FFFFH for all 16 bits at once after clearing the PFC9 register to 0000H.

After reset: 0000H R/W Address: PFC9 FFFFF472H, PFC9L FFFFF472H, PFC9H FFFFF473H

	15	14	13	12	11	10	9	8
PFC9 (PFC9H[Note])	PFC915	PFC914	PFC913	PFC912	PFC911	PFC910	PFC99	PFC98

	7	6	5	4	3	2	1	0
(PFC9L)	PFC97	PFC96	PFC95	PFC94	PFC93	PFC92	PFC91	PFC90

Note To read/write bits 8 to 15 of the PFC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFC9H register.

Remark For details on control mode specification, see **4.3.8 (2) (f) P9 pin control mode settings**.

(e) Port function control expansion register 9 (PFCE9)

The PFCE9 register is a 16-bit register that specifies control mode 1, control mode 2, control mode 3, or control mode 4.

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

However, when using the higher 8 bits of the PFCE9 register as the PFCE9H register and the lower 8 bits as the PFCE9L register, PFCE9 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PFCE9 FFFFF712H, PFCE9L FFFFF712H, PFCE9H FFFFF713H

	15	14	13	12	11	10	9	8
PFCE9 (PFCE9H[Note])	PFCE915	PFCE914	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
(PFCE9L)	PFCE97	PFCE96	PFCE95	PFCE94	PFCE93	PFCE92	PFCE91	PFCE90

Note To read/write bits 8 to 15 of the PFCE9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFCE9H register.

Remark For details on control mode specification, see **4.3.8 (2) (f) P9 pin control mode settings**.

(f) P9 pin control mode settings

PFCE915	PFC915	Specification of P915 Pin Control Mode
0	0	A15 output
0	1	INTP6 input
1	0	TIP50 input
1	1	Setting prohibited

PFCE914	PFC914	Specification of P914 Pin Control Mode
0	0	A14 output
0	1	INTP5 input
1	0	TIP51 input
1	1	TOP51 output

PFC913	Specification of P913 Pin Control Mode
0	A13 output
1	INTP4 input

PFC912	Specification of P912 Pin Control Mode
0	A12 output
1	$\overline{\text{SCKB3}}$ I/O

PFC911	Specification of P911 Pin Control Mode
0	A11 output
1	SOB3 output

PFC910	Specification of P910 Pin Control Mode
0	A10 output
1	SIB3 input

PFC99	Specification of P99 Pin Control Mode
0	A9 output
1	$\overline{\text{SCKB1}}$ I/O

PFC98	Specification of P98 Pin Control Mode
0	A8 output
1	SOB1 output

PFCE97	PFC97	Specification of P97 Pin Control Mode
0	0	A7 output
0	1	SIB1 input
1	0	TIP20 input
1	1	TOP20 output

PFCE96	PFC96	Specification of P96 Pin Control Mode
0	0	A6 output
0	1	Setting prohibited
1	0	TIP21 input
1	1	TOP21 output

PFCE95	PFC95	Specification of P95 Pin Control Mode
0	0	A5 output
0	1	TIP30 input
1	0	TOP30 output
1	1	Setting prohibited

PFCE94	PFC94	Specification of P94 Pin Control Mode
0	0	A4 output
0	1	TIP31 input
1	0	TOP31 output
1	1	Setting prohibited

PFCE93	PFC93	Specification of P93 Pin Control Mode
0	0	A3 output
0	1	TIP40 input
1	0	TOP40 output
1	1	Setting prohibited

PFCE92	PFCE92	Specification of P92 Pin Control Mode
0	0	A2 output
0	1	TIP41 input
1	0	TOP41 output
1	1	Setting prohibited

PFCE91	PFCE91	Specification of P91 Pin Control Mode
0	0	A1 output
0	1	KR7 input
1	0	RXDA1/KR7 input
1	1	SCL02 I/O

PFCE90	PFC90	Specification of P90 Pin Control Mode
0	0	A0 output
0	1	KR6 input
1	0	TXDA1 output
1	1	SDA02 I/O

(g) Port function register 9 (PF9)

The PF9 register is a 16-bit register that specifies normal output or N-ch open-drain output.

This register can be read or written in 16-bit units. However, when using the higher 8 bits of the PF9 register as the PF9H register and the lower 8 bits as the PF9L register, PF9 can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PF3 FFFFFC72H, PF9L FFFFFC72H, PF9H FFFFFC73H

	15	14	13	12	11	10	9	8
PF9 (PF9H[Note])	PF915	PF914	PF913	PF912	PF911	PF910	PF99	PF98

	7	6	5	4	3	2	1	0
(PF9L)	PF97	PF96	PF95	PF94	PF93	PF92	PF91	PF90

PF9n	Control of normal output or N-ch open-drain output (n = 0 to 15)
0	Normal output
1	N-ch open-drain output

Note To read/write bits 8 to 15 of the PF9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PF9H register.

(h) External interrupt falling edge specification register 9H (INTF9H)

The INTF9H register is an 8-bit register that specifies external interrupt pin falling edge detection.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF9n bit = INTR9n bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.

After reset: 00H R/W Address: FFFFFC13H

	15	14	13	12	11	10	9	8
INTF9H	INTF915	INTF914	INTF913	0	0	0	0	0

Remark For details on valid edge specification, see **Table 4-12**.

(i) External interrupt rising edge specification register 9H (INTR9H)

The INTR9H register is an 8-bit register that specifies external interrupt pin rising edge detection.

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

Reset input clears this register to 00H.

Cautions 1. When switching from the external interrupt function (alternate function) to the port function, edge detection may occur. Therefore, set the port mode after setting INTF9n bit = INTR9n bit = 0.

2. An on-chip circuit for eliminating noise through analog delay is provided for external interrupt input.

After reset: 00H R/W Address: FFFFFC33H

	15	14	13	12	11	10	9	8
INTR9H	INTR915	INTR914	INTR913	0	0	0	0	0

Remark For details on valid edge specification, see **Table 4-12**.

Table 4-12. Valid Edge Specification

INTF9n	INTR9n	Valid Edge Specification (n = 13 to 15)
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution When not using the pins as INTP4 to INTP6 pins, be sure to set INTF9n bit = INTR9n bit = 0.

Remark n = 13 to 15: Control of INTP4 to INTP6 pins

4.3.9 Port CM

Port CM is a 4-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port CM functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register CM (PCM)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register CM (PMCM)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register (PMCCM)

Port CM includes the following alternate-function pins.

Table 4-13. Port CM Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port CM	PCM0	$\overline{\text{WAIT}}$	I/O	–
	PCM1	CLKOUT		
	PCM2	$\overline{\text{HLDAK}}$		
	PCM3	$\overline{\text{HLDRQ}}$		

(2) Registers

(a) Port register CM (PCM)

The PCM register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF00CH

	7	6	5	4	3	2	1	0
PCM	0	0	0	0	PCM3	PCM2	PCM1	PCM0

PCMn	Output data control (in output mode) (n = 0 to 3)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port CM (PCM) is read, the pin levels at this time are read. Writing to PCM writes the values to that port. This does not affect the input pins.

Output mode: When port CM (PCM) is read, the PCM values are read. Writing to PCM writes the values to that port, and those values area immediately output.

2. The value of PCM is undefined (pin input level) when it is read in the input mode after reset. When PCM is read in the output mode, 00H (output latch value) is read.

(b) Port mode register CM (PMCM)

The PMCM register is an 8-bit register that specifies the input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF02CH

	7	6	5	4	3	2	1	0
PMCM	1	1	1	1	PMCM3	PMCM2	PMCM1	PMCM0

PMCMn	I/O mode control (n = 0 to 3)
0	Output mode
1	Input mode

(c) Port mode control register CM (PMCCM)

The PMCCM register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF04CH

	7	6	5	4	3	2	1	0
PMCCM	0	0	0	0	PMCCM3	PMCCM2	PMCCM1	PMCCM0

PMCCM3	Specification of PCM3 pin operation mode
0	I/O port
1	$\overline{\text{HLDRQ}}$ input

PMCCM2	Specification of PCM2 pin operation mode
0	I/O port
1	$\overline{\text{HLDAK}}$ output

PMCCM1	Specification of PCM1 pin operation mode
0	I/O port
1	CLKOUT output

PMCCM0	Specification of PCM0 pin operation mode
0	I/O port
1	$\overline{\text{WAIT}}$ input

4.3.10 Port CT

Port CT is a 4-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port CT functions

- ○ Port input/output data specifiable in 1-bit units
  Specification made by port register CT (PCT)
- ○ Port input/output specifiable in 1-bit units
  Specification made by port mode register CT (PMCT)
- ○ Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register CT (PMCCT)

Table 4-14. Port CT Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port CT	PCT0	$\overline{WR0}$	I/O	–
	PCT1	$\overline{WR1}$		
	PCT4	$\overline{RD}$		
	PCT6	ASTB		

(2) Registers

(a) Port register CT (PCT)

The PCT register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF00AH

	7	6	5	4	3	2	1	0
PCT	0	PCT6	0	PCT4	0	0	PCT1	PCT0

PCTn	Output data control (in output mode) (n = 0, 1, 4, 6)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port CT (PCT) is read, the pin levels at this time are read. Writing to PCT writes the values that port. This does not affect the input pins.

Output mode: When port CT (PCT) is read, the PCT values are read. Writing to PCT writes the values to that port, and those values are immediately output.

2. The value of PCT is undefined (pin input level) when it is read in the input mode after reset. When PCT is read in the output mode, 00H (output latch value) is read.

(b) Port mode register CT (PMCT)

The PMCT register is an 8-bit port that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF02AH

	7	6	5	4	3	2	1	0
PMCT	1	PMCT6	1	PMCT4	1	1	PMCT1	PMCT0

PMCTn	I/O mode control (n = 0, 1, 4, 6)
0	Output mode
1	Input mode

(c) Port mode control register CT (PMCCT)

The PMCCT register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF04AH

	7	6	5	4	3	2	1	0
PMCCT	0	PMCCT6	0	PMCCT4	0	0	PMCCT1	PMCCT0

PMCCT6	Specification of PCT6 pin operation mode
0	I/O port
1	ASTB output

PMCCT4	Specification of PCT4 pin operation mode
0	I/O port
1	$\overline{RD}$ output

PMCCT1	Specification of PCT1 pin operation mode
0	I/O port
1	$\overline{WR1}$ output

PMCCT0	Specification of PCT0 pin operation mode
0	I/O port
1	$\overline{WR0}$ output

4.3.11 Port DH

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

(1) Port DH functions

- Port input/output data specifiable in 1-bit units
  Specification made by port register DH (PDH)
- Port input/output specifiable in 1-bit units
  Specification made by port mode register DH (PMDH)
- Port mode/control mode (alternate functions) specifiable in 1-bit units
  Specification made by port mode control register DH (PMCDH)

Port DH includes the following alternate-function pins.

Table 4-15. Port DH Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port DH	PDH0	A16	I/O	–
	PDH1	A17		
	PDH2	A18		
	PDH3	A19		
	PDH4	A20		
	PDH5	A21		

(2) Registers

(a) Port register DH (PDH)

The PDH register is an 8-bit register that controls pin level read and output level write.

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

After reset: Undefined R/W Address: FFFFF006H

	7	6	5	4	3	2	1	0
PDH	0	0	PDH5	PDH4	PDH3	PDH2	PDH1	PDH0

PDHn	Output data control (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

Remarks 1. Input mode: When port DH (PDH) is read, the pin levels at this time are read. Writing to PDH writes the values to that port. This does not affect the input pins.

Output mode: When port DH (PDH) is read, the PDH values are read. Writing to PDH writes the values to that port, and those the values are immediately output.

2. The value of PDH is undefined (pin input level) when it is read in the input mode after reset. When PDH is read in the output mode, 00H (output latch value) is read.

(b) Port mode register DH (PMDH)

The PMDH is an 8-bit register that specifies input or output mode.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: FFFFF026H

	7	6	5	4	3	2	1	0
PMDH	1	1	PMDH5	PMDH4	PMDH3	PMDH2	PMDH1	PMDH0

PMDHn	I/O mode control (n = 0 to 5)
0	Output mode
1	Input mode

(c) Port mode control register DH (PMCDH)

The PMCDH register is an 8-bit register that specifies port or control mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF046H

	7	6	5	4	3	2	1	0
PMCDH	0	0	PMCDH5	PMCDH4	PMCDH3	PMCDH2	PMCDH1	PMCDH0

PMCDHn	Specification of PDHn pin operation mode (n = 0 to 5)
0	I/O port
1	Am output (address bus output) (m = 16 to 21)

4.3.12 Port DL

Port DL is a 16-bit port for which I/O settings can be controlled in 1-bit units.

(1) Port DL functions

- ○ Port input/output data specifiable in 1-bit units
  Specification made by port register DL (PDL)
- ○ Port input/output specifiable in 1-bit units
  Specification made by port mode register DL (PMDL)
- ○ Port mode/control mode (alternate functions) specifiable
  Specification made by port mode control register DL (PMCDL)

Port DL includes the following alternate-function pins.

Table 4-16. Port DL Alternate-Function Pins

Pin Name		Alternate-Function Pin Name	I/O	Remark
Port DL	PDL0	AD0	I/O	–
	PDL1	AD1		
	PDL2	AD2		
	PDL3	AD3		
	PDL4	AD4		
	PDL5	AD5/FLMD1[Note]		
	PDL6	AD6		
	PDL7	AD7		
	PDL8	AD8		
	PDLDL	AD9		
	PDL10	AD10		
	PDL11	AD11		
	PDL12	AD12		
	PDL13	AD13		
	PDL14	AD14		
	PDL15	AD15		

Note Since this pin is set in the flash programming mode, it does not need to be manipulated with the port control register. For details, see **CHAPTER 28 FLASH MEMORY**.

(2) Registers

(a) Port register DL (PDL)

The PDL register is a 16-bit register that controls pin level read and output level write.

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

However, when using the higher 8 bits of the PDL register as the PDLH register and the lower 8 bits as the PDLL register, PDL can be read or written in 8-bit or 1-bit units.

After reset: Undefined R/W Address: PDL FFFFF004H, PDLL FFFFF004H, PDLH FFFFF005H

	15	14	13	12	11	10	9	8
PDL (PDLH[Note])	PDL15	PDL14	PDL13	PDL12	PDL11	PDL10	PDL9	PDL8

	7	6	5	4	3	2	1	0
(PDLL)	PDL7	PDL6	PDL5	PDL4	PDL3	PDL2	PDL1	PDL0

PDLn	Output data control (in output mode) (n = 0 to 15)
0	Outputs 0
1	Outputs 1

Note To read/write bits 8 to 15 of the PDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PDLH register.

Remarks 1. Input mode: When port DL (PDL) is read, the pin levels at this time are read. Writing to PDL writes the values to that port. This does not affect the input pins.

Output mode: When port DL (PDL) is read, the PDL values are read. Writing to PDL writes the values to that port, and those values are immediately output.

2. The value of PDL is undefined (pin input level) when it is read in the input mode after reset. When PDL is read in the output mode, 0000H (output latch value) is read.

(b) Port mode register DL (PMDL)

The PMDL register is a 16-bit register that specifies input or output mode.

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

However, when using the higher 8 bits of the PMDL register as the PMDLH register and the lower 8 bits as the PMDLL register, PMDL can be read or written in 8-bit or 1-bit units.

Reset input sets this register to FFFFH.

After reset: FFFFH R/W Address: PMDL FFFFF024H, PMDLL FFFFF024H, PMDLH FFFFF025H

	15	14	13	12	11	10	9	8
PMDL (PMDLH[Note])	PMDL15	PMDL14	PMDL13	PMDL12	PMDL11	PMDL10	PMDL9	PMDL8

	7	6	5	4	3	2	1	0
(PMDLL)	PMDL7	PMDL6	PMDL5	PMDL4	PMDL3	PMDL2	PMDL1	PMDL0

PMDLn	I/O mode control (n = 0 to 15)
0	Output mode
1	Input mode

Note To read/write bits 8 to 15 of the PMDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMDLH register.

(c) Port mode control register DL (PMCDL)

The PMCDL register is a 16-bit register that specifies port or control mode.

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

However, when using the higher 8 bits of the PMCDL register as the PMCDLH register and the lower 8 bits as the PMCDLL register, PMCDL can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: PMCDL FFFFF044H, PMCDLL FFFFF044H, PMCDLH FFFFF045H

	15	14	13	12	11	10	9	8
PMCDL (PMCDLH[Note])	PMCDL15	PMCDL14	PMCDL13	PMCDL12	PMCDL11	PMCDL10	PMCDL9	PMCDL8

	7	6	5	4	3	2	1	0
(PMCDLL)	PMCDL7	PMCDL6	PMCDL5	PMCDL4	PMCDL3	PMCDL2	PMCDL1	PMCDL0

PMCDLn	Specification of PDLn pin operation mode (n = 0 to 15)
0	I/O port
1	ADn I/O (address/data bus I/O)

Note To read/write bits 8 to 15 of the PMCDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMCDLH register.

Caution When the SMSEL bit of the EXIMC register = 1 (separate mode) and the BS30 to BS00 bits of the BSC register = 0 (8-bit bus width), do not specify the AD8 to AD15 pins.

Table 4-17. Using Port Pin as Alternate-Function Pin (1/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P02	NMI	Input	P02 = Setting not required	PM02 = Setting not required	PMC02 = 1	–	–	INTR02 (INTR0), INTF02 (INTF0)
P03	INTP0	Input	P03 = Setting not required	PM03 = Setting not required	PMC03 = 1	–	PFC03 = 0	INTR03 (INTR0), INTF03 (INTF0)
	ADTRG	Input	P03 = Setting not required	PM03 = Setting not required	PMC03 = 1	–	PFC03 = 1	
P04	INTP1	Input	P04 = Setting not required	PM04 = Setting not required	PMC04 = 1	–	–	INTR04 (INTR0), INTF04 (INTF0)
P05	INTP2	Input	P05 = Setting not required	PM05 = Setting not required	PMC05 = 1	–	–	INTR05 (INTR0), INTF05 (INTF0)
	$\overline{\text{DRST}}$[Note]	Input	P05 = Setting not required	PM05 = Setting not required	PMC05 = Setting not required	–	–	OCDM0 (OCDM) = 1
P06	INTP3	Input	P06 = Setting not required	PM06 = Setting not required	PMC06 = 1	–	–	INTR06 (INTR0), INTF06 (INTF0)
P10	ANO0	Output	P10 = Setting not required	PM10 = 1	–	–	–	
P11	ANO1	Output	P11 = Setting not required	PM11 = 1	–	–	–	

Note The P05 pin's alternate function is a pin for on-chip debugging (flash memory version only). After external reset, the P05/INTP2/$\overline{\text{DRST}}$ pin is initialized to the on-chip debugging pin ($\overline{\text{DRST}}$). When using the P05 pin as a port pin and not as an on-chip debugging pin, the following handling is required.

<1> Clear (0) the OCDM0 bit of the OCDM register (special register).

If a high level is input to the $\overline{\text{DRST}}$ pin when the on-chip debug function is not used and before the above processing is performed, malfunction (CPU deadlock) may occur. Exercise care in handling the P05 pin. (The P05 pin has a pull-down resistor (30 kΩ typ.) in the buffer and does not have to be fixed to low level by an external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).)

Caution When using the P10 and P11 pins as alternate functions (ANO0, ANO1 pin), set the PM1 register to FFH.

Table 4-17. Using Port Pin as Alternate-Function Pin (2/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P30	TXDA0	Output	P30 = Setting not required	PM30 = Setting not required	PMC30 = 1	–	PFC30 = 0	
	SOB4	Output	P30 = Setting not required	PM30 = Setting not required	PMC30 = 1	–	PFC30 = 1	
P31	RXDA0	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	**Note 1**, PFC31 = 0	
	INTP7	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	**Note 1**, PFC31 = 0	INTR31 (INTR3), INTF31 (INTF3)
	SIB4	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	PFC31 = 1	
P32	ASCKA0	Input	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 0	PFC32 = 0	
	$\overline{\text{SCKB4}}$	I/O	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 0	PFC32 = 1	
	TIP00	Input	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 1	PFC32 = 0	
	TOP00	Output	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 1	PFC32 = 1	
P33	TIP01	Input	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	PFCE33 = 0	PFC33 = 0	
	TOP01	Output	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	PFCE33 = 0	PFC33 = 1	
P34	TIP10	Input	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	PFCE34 = 0	PFC34 = 0	
	TOP10	Input	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	PFCE34 = 0	PFC34 = 1	
P35	TIP11	Input	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	PFCE35 = 0	PFC35 = 0	
	TOP11	Output	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	PFCE35 = 0	PFC35 = 1	
P36	CTXD0**Note 2**	Output	P36 = Setting not required	PM36 = Setting not required	PMC36 = 1	–	PFC36 = 0	
	$\overline{\text{IETX0}}$**Note 3**	Output	P36 = Setting not required	PM36 = Setting not required	PMC36 = 1	–	PFC36 = 1	
P37	CRXD0**Note 2**	Input	P37 = Setting not required	PM37 = Setting not required	PMC37 = 1	–	PFC37 = 0	
	$\overline{\text{IERX0}}$**Note 3**	Input	P37 = Setting not required	PM37 = Setting not required	PMC37 = 1	–	PFC37 = 1	
P38	TXDA2	Output	P38 = Setting not required	PM38 = Setting not required	PMC38 = 1	–	PFC38 = 0	
	SDA00**Note 4**	I/O	P38 = Setting not required	PM38 = Setting not required	PMC38 = 1	–	PFC38 = 1	

Notes 1. The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin. (Clear the INTF31 bit of the INTF3 register and the INTR31 bit of the INRT3 register to 0.) When using the pin as the INTP7 pin, stop UARTA0 reception. (Clear the UA0RXE bit of the UA0CTL0 register to 0.)

2. CAN controller version only

3. IEBus controller version only

4. I^2C bus version (Y version) only

Table 4-17. Using Port Pin as Alternate-Function Pin (3/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P39	RXDA2	Input	P39 = Setting not required	PM39 = Setting not required	PMC39 = 1	–	PFC39 = 0	
	SCL00Note	I/O	P39 = Setting not required	PM39 = Setting not required	PMC39 = 1	–	PFC39 = 1	
P40	SIB0	Input	P40 = Setting not required	PM40 = Setting not required	PMC40 = 1	–	PFC40 = 0	
	SDA01Note	I/O	P40 = Setting not required	PM40 = Setting not required	PMC40 = 1	–	PFC40 = 1	
P41	SOB0	Output	P41 = Setting not required	PM41 = Setting not required	PMC41 = 1	–	PFC41 = 0	
	SCL01Note	I/O	P41 = Setting not required	PM41 = Setting not required	PMC41 = 1	–	PFC41 = 1	
P42	$\overline{\text{SCKB0}}$	I/O	P42 = Setting not required	PM42 = Setting not required	PMC42 = 1	–	–	
P50	TIQ01	Input	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFCE50 = 0	PFC50 = 1	KRM0 (KRM) = 0
	KR0	Input	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFCE50 = 0	PFC50 = 1	TQ0TIG2, TQ0TIG3 (TQ0IOC1) = 0
	TOQ01	Output	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFCE50 = 1	PFC50 = 0	
	RTP00	Output	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFCE50 = 1	PFC50 = 1	
P51	TIQ02	Input	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFCE51 = 0	PFC51 = 1	KRM1 (KRM) = 0
	KR1	Input	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFCE51 = 0	PFC51 = 1	TQ0TIG4, TQ0TIG5 (TQ0IOC1) = 0
	TOQ02	Output	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFCE51 = 1	PFC51 = 0	
	RTP01	Output	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFCE51 = 1	PFC51 = 1	

Note I^2C bus version (Y version) only

Table 4-17. Using Port Pin as Alternate-Function Pin (4/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P52	TIQ03	Input	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFCE52 = 0	PFC52 = 1	KRM2 (KRM) = 0
	KR2	Input	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFCE52 = 0	PFC52 = 1	TQ0TIG6, TQ0TIG7 (TQ0I0C1) = 0
	TOQ03	Output	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFCE52 = 1	PFC52 = 0	
	RTP02	Output	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFCE52 = 1	PFC52 = 1	
	DDI[Note]	Input	P52 = Setting not required	PM52 = Setting not required	PMC52 = Setting not required	–	–	OCDM0 (OCDM) = 1
P53	SIB2	Input	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 0	PFC53 = 0	
	TIQ00	Input	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 0	PFC53 = 1	KRM3 (KRM) = 0
	KR3	Input	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 0	PFC53 = 1	TQ0TIG0, TQ0TIG1 (TQ0IOC1) = 0, TQ0EES0, TQ0EES1 (TQ0IOC2) = 0, TQ0ETS0, TQ0ETS1 (TQ0IOC2) = 0
	TOQ00	Input	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 1	PFC53 = 0	
	RTP03	Output	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 1	PFC53 = 1	
	DDO[Note]	Output	P53 = Setting not required	PM53 = Setting not required	PMC53 = Setting not required	–	–	OCDM0 (OCDM) = 1

Note The DDI and DDO pins are pins for on-chip debugging (flash memory version only). After external reset, when not using these pins as on-chip debugging pins and using them instead as port pins, the following handling is required.

- Input a low level to the P05/INTP2/$\overline{\text{DRST}}$ pin.
- Set the ODCM0 bit of the ODCM register (special register). The handling in this case is as follows.
  <1> Clear (0) the OCDM0 bit of the OCDM register.
  <2> Fix the P05/INTP2/$\overline{\text{DRST}}$ pin input to low level until the handling in <1> above is completed.

If a high level is input to the $\overline{\text{DRST}}$ pin when the on-chip debug function is not used and before the above processing is performed, malfunction (CPU deadlock) may occur. Exercise care in handling the P05 pin. (The P05 pin has a pull-down resistor (30 kΩ typ.) in the buffer and does not have to be fixed to low level by an external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).)

Table 4-17. Using Port Pin as Alternate Function-Pin (5/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P54	SOB2	Output	P54 = Setting not required	PM54 = Setting not required	PMC54 = 1	PFCE54 = 0	PFC54 = 0	
	KR4	Input	P54 = Setting not required	PM54 = Setting not required	PMC54 = 1	PFCE54 = 0	PFC54 = 1	
	RTP04	Output	P54 = Setting not required	PM54 = Setting not required	PMC54 = 1	PFCE54 = 1	PFC54 = 1	
	DCK[Note]	Input	P54 = Setting not required	PM54 = Setting not required	PMC54 = Setting not required	–	–	OCDM0 (OCDM) = 1
P55	$\overline{\text{SCKB2}}$	I/O	P55 = Setting not required	PM55 = Setting not required	PMC55 = 1	PFCE55 = 0	PFC55 = 0	
	KR5	Input	P55 = Setting not required	PM55 = Setting not required	PMC55 = 1	PFCE55 = 0	PFC55 = 1	
	RTP05	Output	P55 = Setting not required	PM55 = Setting not required	PMC55 = 1	PFCE55 = 1	PFC55 = 1	
	DMS[Note]	Input	P55 = Setting not required	PM55 = Setting not required	PMC55 = Setting not required	–	–	OCDM0 (OCDM) = 1

Note The DCK and DMS pins are pins for on-chip debugging (flash memory version only). After external reset, when not using these pins as on-chip debugging pins and using them instead as port pins, the following handling is required.

- Input a low level to the P05/INTP2/$\overline{\text{DRST}}$ pin.
- Set the ODCM0 bit of the ODCM register (special register). The handling in this case is as follows.
  <1> Clear (0) the OCDM0 bit of the OCDM register.
  <2> Fix the P05/INTP2/$\overline{\text{DRST}}$ pin input to low level until the handling in <1> above is completed.

If a high level is input to the $\overline{\text{DRST}}$ pin when the on-chip debug function is not used and before the above processing is performed, malfunction (CPU deadlock) may occur. Exercise care in handling the P05 pin. (The P05 pin has a pull-down resistor (30 kΩ typ.) in the buffer and does not have to be fixed to the low level by an external source. The pull-down resistor is disconnected when the OCDM0 bit is cleared (0).)

Table 4-17. Using Port Pin as Alternate-Function Pin (6/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P70	ANI0	Input	P70 = Setting not required	PM70 = 1	–	–	–	
P71	ANI1	Input	P71 = Setting not required	PM71 = 1	–	–	–	
P72	ANI2	Input	P72 = Setting not required	PM72 = 1	–	–	–	
P73	ANI3	Input	P73 = Setting not required	PM73 = 1	–	–	–	
P74	ANI4	Input	P74 = Setting not required	PM74 = 1	–	–	–	
P75	ANI5	Input	P75 = Setting not required	PM75 = 1	–	–	–	
P76	ANI6	Input	P76 = Setting not required	PM76 = 1	–	–	–	
P77	ANI7	Input	P77 = Setting not required	PM77 = 1	–	–	–	
P78	ANI8	Input	P78 = Setting not required	PM78 = 1	–	–	–	
P79	ANI9	Input	P79 = Setting not required	PM79 = 1	–	–	–	
P710	ANI10	Input	P710 = Setting not required	PM710 = 1	–	–	–	
P711	ANI11	Input	P711 = Setting not required	PM711 = 1	–	–	–	
P90	A0	Output	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 0	PFC90 = 0	**Note 1**
	KR6	Input	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 0	PFC90 = 1	
	TXDA1	Output	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 1	PFC90 = 0	
	SDA02Note 2	I/O	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 1	PFC90 = 1	
P91	A1	Output	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 0	PFC91 = 0	**Note 1**
	KR7	Input	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 0	PFC91 = 1	
	RXDA1/KR7	Input	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 1	PFC91 = 0	
	SCL02Note 2	I/O	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 1	PFC91 = 1	
P92	A2	Output	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 0	PFC92 = 0	**Note 1**
	TIP41	Input	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 0	PFC92 = 1	
	TOP41	Output	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 1	PFC92 = 0	

Notes 1. When setting pins A0 to A15 as the alternate function, set the PMC9 register to FFFFH for all 16 bits at once.

2. I^2C bus version (Y version) only

Table 4-17. Using Port Pin as Alternate-Function Pin (7/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P93	A3	Output	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 0	PFC93 = 0	**Note**
	TIP40	Input	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 0	PFC93 = 1	
	TOP40	Output	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 1	PFC93 = 0	
P94	A4	Output	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 0	PFC94 = 0	**Note**
	TIP31	Input	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 0	PFC94 = 1	
	TOP31	Output	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 1	PFC94 = 0	
P95	A5	Output	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 0	PFC95 = 0	**Note**
	TIP30	Input	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 0	PFC95 = 1	
	TOP30	Output	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 1	PFC95 = 0	
P96	A6	Output	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 0	PFC96 = 0	**Note**
	TIP21	Input	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 1	PFC96 = 0	
	TOP21	Output	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 1	PFC96 = 1	
P97	A7	Output	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 0	PFC97 = 0	**Note**
	SIB1	Input	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 0	PFC97 = 1	
	TIP20	Input	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 1	PFC97 = 0	
	TOP20	Output	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 1	PFC97 = 1	
P98	A8	Output	P98 = Setting not required	PM98 = Setting not required	PMC98 = 1	–	PFC98 = 0	**Note**
	SOB1	Output	P98 = Setting not required	PM98 = Setting not required	PMC98 = 1	–	PFC98 = 1	
P99	A9	Output	P99 = Setting not required	PM99 = Setting not required	PMC99 = 1	–	PFC99 = 0	**Note**
	$\overline{\text{SCKB1}}$	I/O	P99 = Setting not required	PM99 = Setting not required	PMC99 = 1	–	PFC99 = 1	
P910	A10	Output	P910 = Setting not required	PM910 = Setting not required	PMC910 = 1	–	PFC910 = 0	**Note**
	SIB3	Input	P910 = Setting not required	PM910 = Setting not required	PMC910 = 1	–	PFC910 = 1	
P911	A11	Output	P911 = Setting not required	PM911 = Setting not required	PMC911 = 1	–	PFC911 = 0	**Note**
	SOB3	Output	P911 = Setting not required	PM911 = Setting not required	PMC911 = 1	–	PFC911 = 1	

Note When setting pins A0 to A15 as the alternate function, set the PMC9 register to FFFFH for all 16 bits at once.

Table 4-17. Using Port Pin as Alternate-Function Pin (8/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P912	A12	Output	P912 = Setting not required	PM912 = Setting not required	PMC912 = 1		PFC912 = 0	**Note 1**
	$\overline{\text{SCKB3}}$	I/O	P912 = Setting not required	PM912 = Setting not required	PMC912 = 1		PFC912 = 1	
P913	A13	Output	P913 = Setting not required	PM913 = Setting not required	PMC913 = 1		PFC913 = 0	**Note 1**
	INTP4**Note 2**	Input	P913 = Setting not required	PM913 = Setting not required	PMC913 = 1		**Note 2**, PFC913 = 1	INTR913 (INTR9H), INTF913 (INTF9H)
P914	A14	Output	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 0	PFC914 = 0	**Note 1**
	INTP5**Note 2**	Input	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 0	**Note 2**, PFC914 = 1	INTR914 (INTR9H), INTF914 (INTF9H)
	TIP51	Input	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 1	PFC914 = 0	
	TOP51	Output	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 1	PFC914 = 1	
P915	A15	Output	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 0	PFC915 = 0	**Note 1**
	INTP6**Note 2**	Input	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 0	**Note 2**, PFC915 = 1	INTR915 (INTR9H), INTF915 (INTF9H)
	TIP50	Input	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 1	PFC915 = 0	
	TOP50	Output	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 1	PFC915 = 1	

Notes 1. When setting pins A0 to A15 as the alternate function, set the PMC9 register to FFFFH for all 16 bits at once.

2. When not using the pins as the INTP4 to INTP6 pins, disable edge detection (clear the INTF9n bit of the INTF9H register and INTR9n bit of the INTR9H register to 0 (n = 13 to 15)).

Table 4-17. Using Port Pin as Alternate-Function Pin (9/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
PCM0	$\overline{\text{WAIT}}$	Input	PCM0 = Setting not required	PMCM0 = Setting not required	PMCCM0 = 1	–	–	
PCM1	CLKOUT	Output	PCM1 = Setting not required	PMCM1 = Setting not required	PMCCM1 = 1	–	–	
PCM2	$\overline{\text{HLDAK}}$	Output	PCM2 = Setting not required	PMCM2 = Setting not required	PMCCM2 = 1	–	–	
PCM3	$\overline{\text{HLDRQ}}$	Input	PCM3 = Setting not required	PMCM3 = Setting not required	PMCCM3 = 1	–	–	
PCT0	$\overline{\text{WR0}}$	Output	PCT0 = Setting not required	PMCT0 = Setting not required	PMCCT0 = 1	–	–	
PCT1	$\overline{\text{WR1}}$	Output	PCT1 = Setting not required	PMCT1 = Setting not required	PMCCT1 = 1	–	–	
PCT4	$\overline{\text{RD}}$	Output	PCT4 = Setting not required	PMCT4 = Setting not required	PMCCT4 = 1	–	–	
PCT6	ASTB	Output	PCT6 = Setting not required	PMCT6 = Setting not required	PMCCT6 = 1	–	–	
PDH0	A16	Output	PDH0 = Setting not required	PMDH0 = Setting not required	PMCDH0 = 1	–	–	
PDH1	A17	Output	PDH1 = Setting not required	PMDH1 = Setting not required	PMCDH1 = 1	–	–	
PDH2	A18	Output	PDH2 = Setting not required	PMDH2 = Setting not required	PMCDH2 = 1	–	–	
PDH3	A19	Output	PDH3 = Setting not required	PMDH3 = Setting not required	PMCDH3 = 1	–	–	
PDH4	A20	Output	PDH4 = Setting not required	PMDH4 = Setting not required	PMCDH4 = 1	–	–	
PDH5	A21	Output	PDH5 = Setting not required	PMDH5 = Setting not required	PMCDH5 = 1	–	–	

Note When setting pins A0 to A15 as the alternate function, set the PMC9 register to FFFFH for all 16 bits at once.

Table 4-17. Using Port Pin as Alternate-Function Pin (10/10)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
PDL0	AD0	I/O	PDL0 = Setting not required	PMDL0 = Setting not required	PMCDL0 = 1	–	–	
PDL1	AD1	I/O	PDL1 = Setting not required	PMDL1 = Setting not required	PMCDL1 = 1	–	–	
PDL2	AD2	I/O	PDL2 = Setting not required	PMDL2 = Setting not required	PMCDL2 = 1	–	–	
PDL3	AD3	I/O	PDL3 = Setting not required	PMDL3 = Setting not required	PMCDL3 = 1	–	–	
PDL4	AD4	I/O	PDL4 = Setting not required	PMDL4 = Setting not required	PMCDL4 = 1	–	–	
PDL5	AD5	I/O	PDL5 = Setting not required	PMDL5 = Setting not required	PMCDL5 = 1	–	–	
	FLMD1[Note]	Input	PDL5 = Setting not required	PMDL5 = Setting not required	Setting not required	–	–	
PDL6	AD6	I/O	PDL6 = Setting not required	PMDL6 = Setting not required	PMCDL6 = 1	–	–	
PDL7	AD7	I/O	PDL7 = Setting not required	PMDL7 = Setting not required	PMCDL7 = 1	–	–	
PDL8	AD8	I/O	PDL8 = Setting not required	PMDL8 = Setting not required	PMCDL8 = 1	–	–	
PDL9	AD9	I/O	PDL9 = Setting not required	PMDL9 = Setting not required	PMCDL9 = 1	–	–	
PDL10	AD10	I/O	PDL10 = Setting not required	PMDL10 = Setting not required	PMCDL10 = 1	–	–	
PDL11	AD11	I/O	PDL11 = Setting not required	PMDL11 = Setting not required	PMCDL11 = 1	–	–	
PDL12	AD12	I/O	PDL12 = Setting not required	PMDL12 = Setting not required	PMCDL12 = 1	–	–	
PDL13	AD13	I/O	PDL13 = Setting not required	PMDL13 = Setting not required	PMCDL13 = 1	–	–	
PDL14	AD14	I/O	PDL14 = Setting not required	PMDL14 = Setting not required	PMCDL14 = 1	–	–	
PDL15	AD15	I/O	PDL15 = Setting not required	PMDL15 = Setting not required	PMCDL15 = 1	–	–	

Note Since this pin is set in the flash programming mode, it does not need to be manipulated with the port control register. For details, see **CHAPTER 28 FLASH MEMORY**.

4.4 Port Function Operation

The port operation differs according to the I/O mode settings, as follows.

4.4.1 Write to I/O ports

(1) Output mode

Values are written to output latches using transfer instructions. Moreover, the output latch contents are output from the pin. Once data has been written to an output latch, it is held until data is newly written to that output latch.

(2) I/O mode

Values are written to output latches using transfer instructions. However, since the output buffer is off, the status of the pin does not change.

Once data has been written to an output latch, it is held until data is newly written to that output latch.

Caution In the case of a 1-bit memory manipulation instruction, the manipulation target is 1 bit, but the port is accessed in 8-bit units. Therefore, for ports where input and output are mixed, the contents of output latches of pins specified for input other than the manipulation target bit become undefined.

4.4.2 Read from I/O port

(1) Output mode

The contents of the output latch are read using a transfer instruction. The contents of the output latch do not change.

(2) Input mode

The pin status is read using a transfer instruction. The contents of the output latch do not change.

4.4.3 I/O port calculation

(1) Output port

Calculation for the output latch contents is performed and the result is written to the output latch. The output latch contents are output from the pin.

Once data has been written to an output latch, it is held until data is newly written to that output latch.

(2) Input mode

The output latch contents become undefined. However, since the output buffer is off, the pin status does not change.

Caution In the case of a 1-bit memory manipulation instruction, the manipulation target is 1 bit, but the port is accessed in 8-bit units. Therefore, for ports where input and output are mixed, the contents of output latches of pins specified for input other than the manipulation target bit become undefined.

CHAPTER 5 BUS CONTROL FUNCTION

The V850ES/SG2 is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

5.1 Features

- ○ Output is selectable from a multiplexed bus with a minimum of 3 bus cycles and a separate bus with a minimum of 2 bus cycles.
- ○ 8-bit/16-bit data bus selectable
- ○ Wait function
  - Programmable wait function of up to 7 states
  - External wait function using $\overline{\text{WAIT}}$ pin
- ○ Idle state function
- ○ Bus hold function
- ○ Up to 4 MB of physical memory connectable

5.2 Bus Control Pins

The pins used to connect an external device are listed in the table below.

Table 5-1. Bus Control Pins (Multiplexed Bus)

Bus Control Pin	Alternate-Function Pin	I/O	Function
AD0 to AD15	PDL0 to PDL15	I/O	Address/data bus
A16 to A21	PDH0 to PDH5	Output	Address bus
$\overline{\text{WAIT}}$	PCM0	Input	External wait control
CLKOUT	PCM1	Output	Internal system clock
$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	PCT0, PCT1	Output	Write strobe signal
$\overline{\text{RD}}$	PCT4	Output	Read strobe signal
ASTB	PCT6	Output	Address strobe signal
$\overline{\text{HLDRQ}}$	PCM3	Input	Bus hold control
$\overline{\text{HLDAK}}$	PCM2	Output	

Table 5-2. External Control Pins (Separate Bus)

Bus Control Pin	Alternate-Function Pin	I/O	Function
AD0 to AD15	PDL0 to PDL15	I/O	Data bus
A0 to A15	P90 to P915	Output	Address bus
A16 to A21	PDH0 to PDH5	Output	Address bus
$\overline{\text{WAIT}}$	PCM0	Input	External wait control
CLKOUT	PCM1	Output	Internal system clock
$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	PCT0, PCT1	Output	Write strobe signal
$\overline{\text{RD}}$	PCT4	Output	Read strobe signal
$\overline{\text{HLDRQ}}$	PCM3	Input	Bus hold control
$\overline{\text{HLDAK}}$	PCM2	Output	

5.2.1 Pin status when internal ROM, internal RAM, or on-chip peripheral I/O is accessed

Table 5-3. Pin Status When Internal ROM, Internal RAM, or On-Chip Peripheral I/O Is Accessed

Access Destination	Address Bus	Data Bus	Control Signal
Internal ROM	Undefined	Hi-Z	Inactive
Internal RAM	Undefined	Hi-Z	Inactive
On-chip peripheral I/O	**Note**	Hi-Z	Inactive

Note When an on-chip peripheral I/O is accessed, the address bus outputs the address of the on-chip peripheral I/O that is accessed.

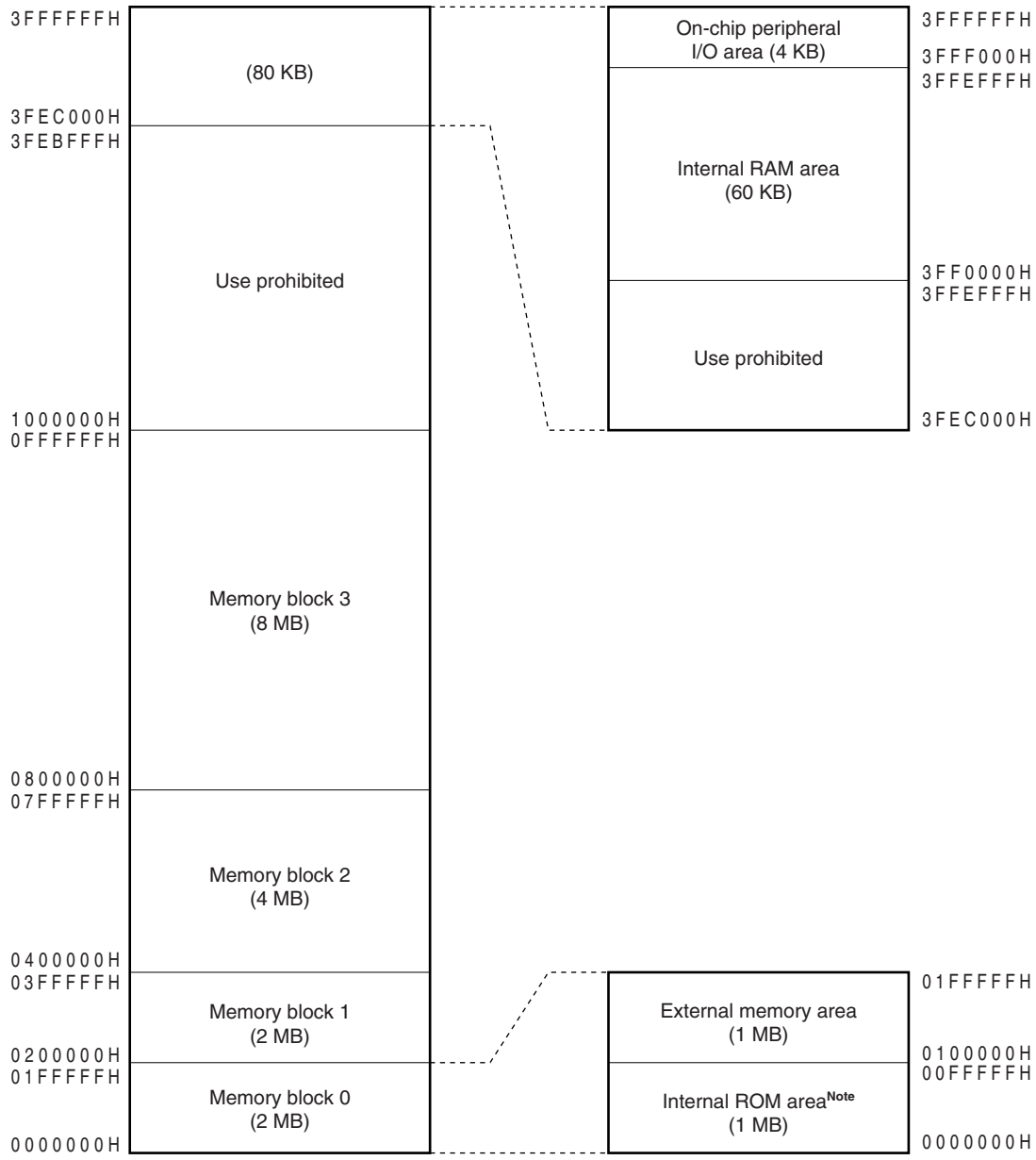
5.2.2 Pin status in each operation mode

For the pin status of the V850ES/SG2 in each operation mode, see **2.2 Pin Status**.

5.3 Memory Block Function

The 64 MB memory space is divided into memory blocks of (lower) 2 MB, 2 MB, 4 MB, and 8 MB. The programmable wait function and bus cycle operation mode for each of these blocks can be independently controlled in one-block units.

Figure 5-1. Data Memory Map



Note This area is an external memory area in the case of a data write access.

5.4 External Bus Interface Mode Control Function

The V850ES/SG2 includes the following two external bus interface modes.

- Multiplexed bus mode
- Separate bus mode

These two modes can be selected by using the external bus interface mode control register (EXIMC).

(1) External bus interface mode control register (EXIMC)

The EXIMC register can be read or written in 8-bit or 1-bit units.

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFFFBEH

	7	6	5	4	3	2	1	0
EXIMC	0	0	0	0	0	0	0	SMSEL

SMSEL	Mode selection
0	Multiplexed bus mode
1	Separate bus mode

Caution Set the EXIMC register from the internal ROM or internal RAM area before making an external access.

After setting the EXIMC register, be sure to insert a NOP instruction.

5.5 Bus Access

5.5.1 Number of clocks for access

The following table shows the number of basic clocks required for accessing each resource.

Bus Cycle Type \ Area (Bus Width)	Internal ROM (32 Bits)	Internal RAM (32 Bits)	External Memory (16 Bits)
Instruction fetch (normal access)	1	1 Note 1	3 + n Note 2
Instruction fetch (branch)	2	1	3 + n Note 2
Operand data access	3	1	3 + n Note 2

Notes 1. 2 if a conflict with a data access occurs.

2. 2 + n clocks (n: Number of wait states) when the separate bus mode is selected.

Remark Unit: Clocks/access

5.5.2 Bus size setting function

The bus size of each external memory area selected by memory block n can be set (to 8 bits or 16 bits) by using the bus size configuration register (BSC).

The external memory area (01000000H to 0FFFFFFH) of the V850ES/SG2 is selected by memory blocks 0 to 3.

(1) Bus size configuration register (BSC)

The BSC register can be read or written in 16-bit units.

Reset input sets this register to 5555H.

Caution Write to the BSC register after reset, and then do not change the set values. Also, do not access an external memory area other than the one for this initialization routine until the initial settings of the BSC register are complete. However, external memory areas whose initial settings are complete may be accessed.

After reset: 5555H R/W Address: FFFFF066H

	15	14	13	12	11	10	9	8
BSC	0	1	0	1	0	1	0	1

7	6	5	4	3	2	1	0
0	BS30	0	BS20	0	BS10	0	BS00

Memory block 3 Memory block 2 Memory block 1 Memory block 0

BSn0	Data bus width of memory block n space (n = 0 to 3)
0	8 bits
1	16 bits

Caution Be sure to set bits 14, 12, 10, and 8 to 1, and clear bits 15, 13, 11, 9, 7, 5, 3, and 1 to 0.

5.5.3 Access by bus size

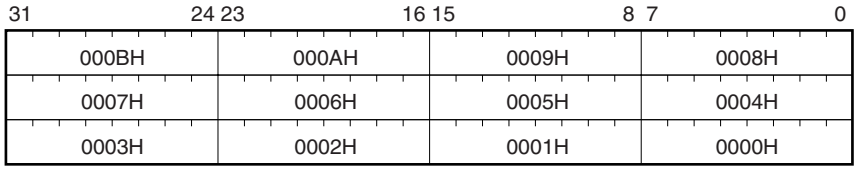
The V850ES/SG2 accesses the on-chip peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The bus size is as follows.

- The bus size of the on-chip peripheral I/O is fixed to 16 bits.
- The bus size of the external memory is selectable from 8 bits or 16 bits (by using the BSC register).

The operation when each of the above is accessed is described below. All data is accessed starting from the lower side.

The V850ES/SG2 supports only the little-endian format.

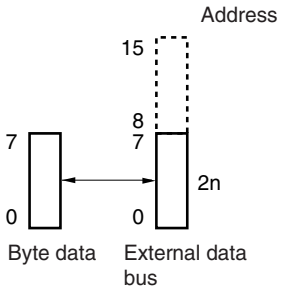
Figure 5-2. Little-Endian Address in Word



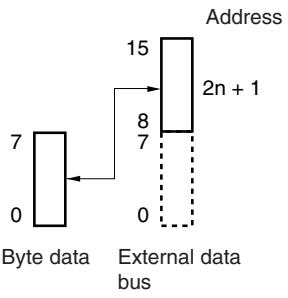
(1) Byte access (8 bits)

(a) 16-bit data bus width

<1> Access to even address (2n)

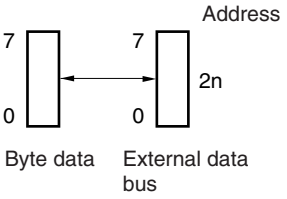


<2> Access to odd address (2n + 1)

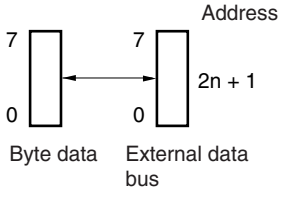


(b) 8-bit data bus width

<1> Access to even address (2n)



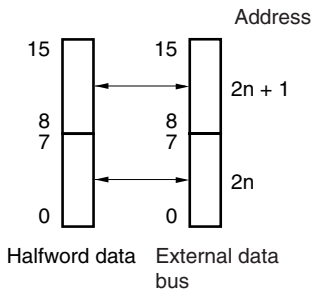
<2> Access to odd address (2n + 1)



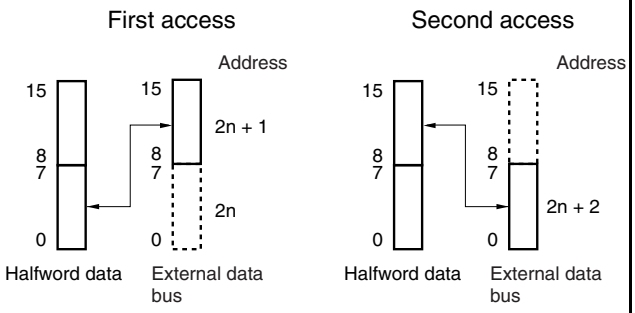
(2) Halfword access (16 bits)

(a) With 16-bit data bus width

<1> Access to even address (2n)

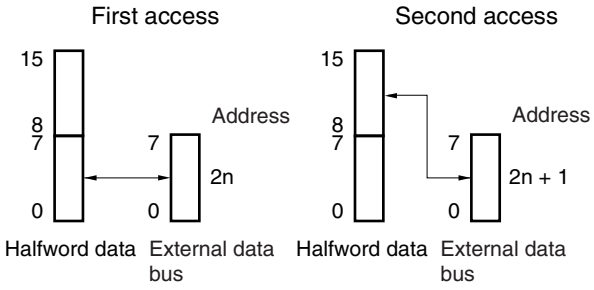


<2> Access to odd address (2n + 1)

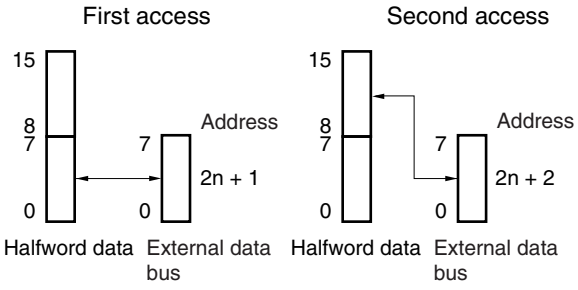


(b) 8-bit data bus width

<1> Access to even address (2n)



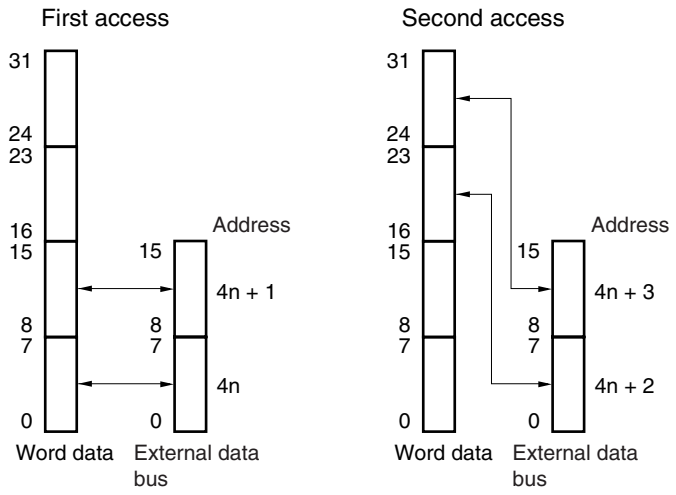
<2> Access to odd address (2n + 1)



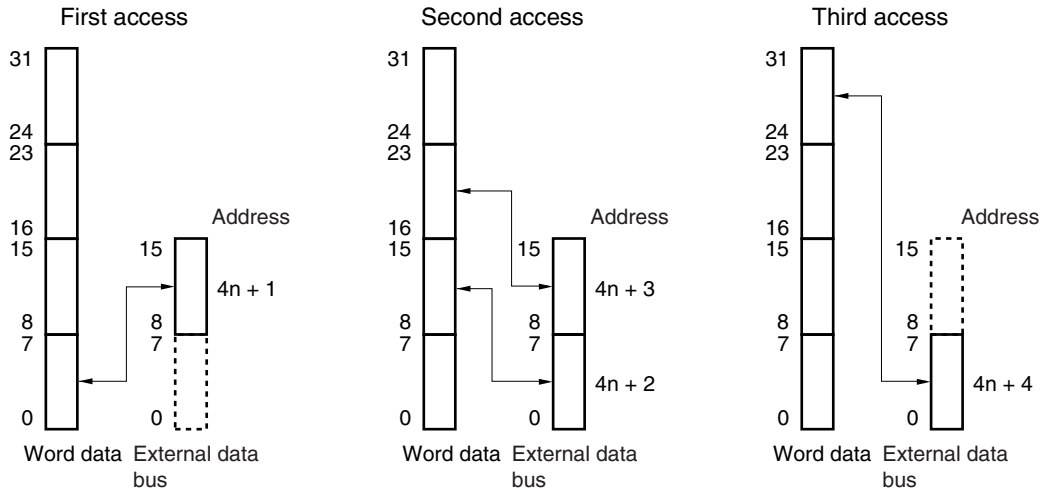
(3) Word access (32 bits)

(a) 16-bit data bus width (1/2)

<1> Access to address (4n)

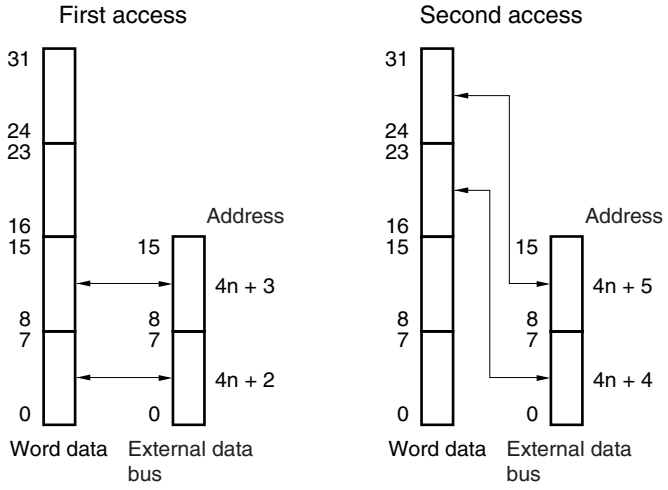


<2> Access to address (4n + 1)

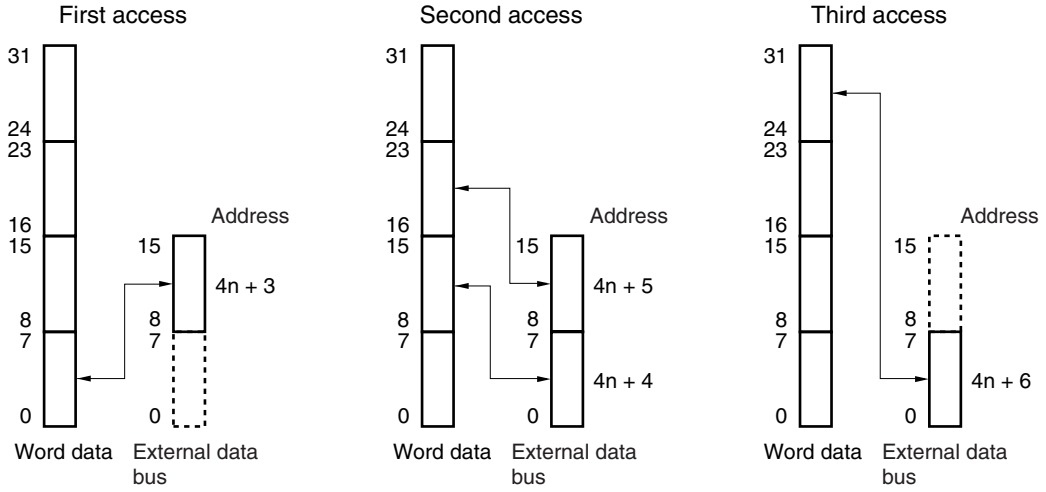


(a) 16-bit data bus width (2/2)

<3> Access to address (4n + 2)

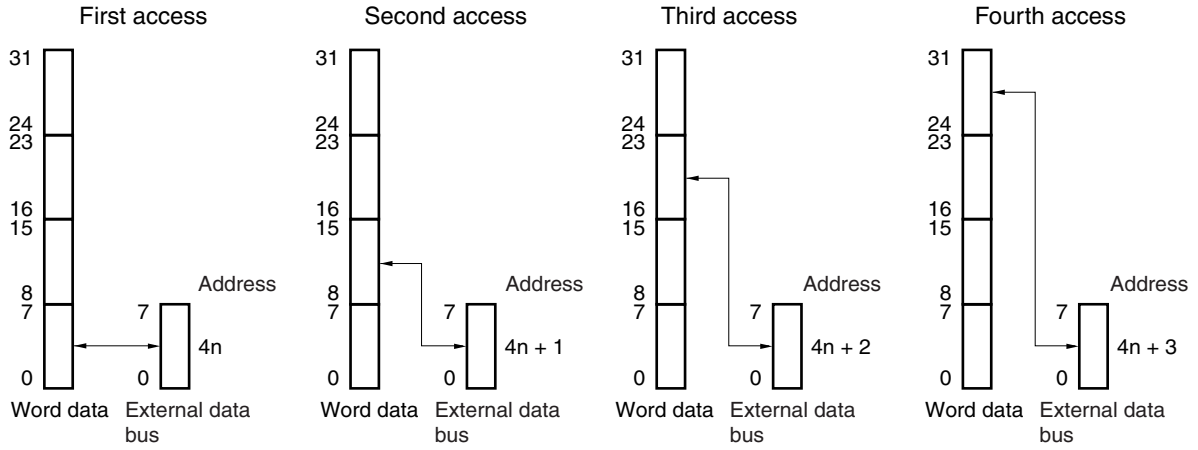


<4> Access to address (4n + 3)

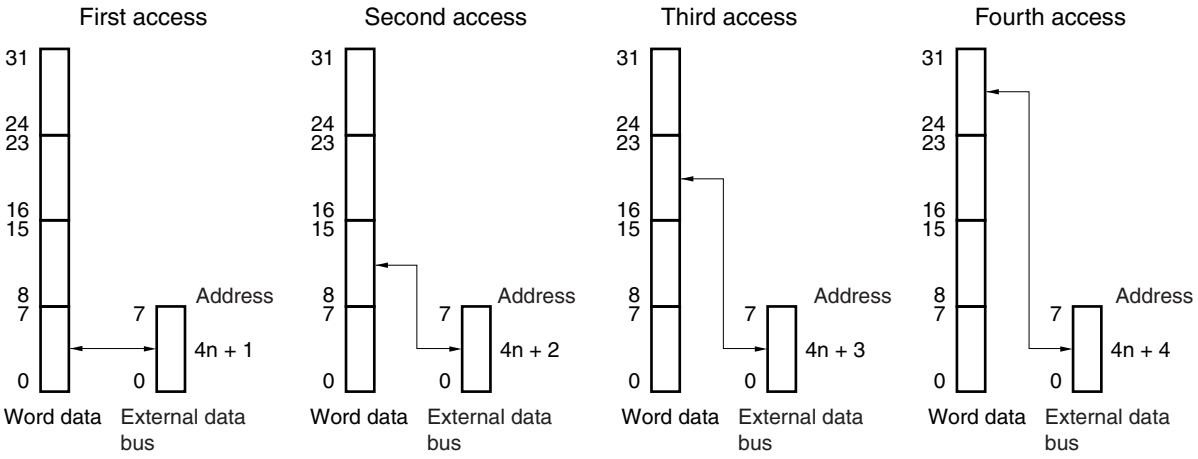


(b) 8-bit data bus width (1/2)

<1> Access to address (4n)

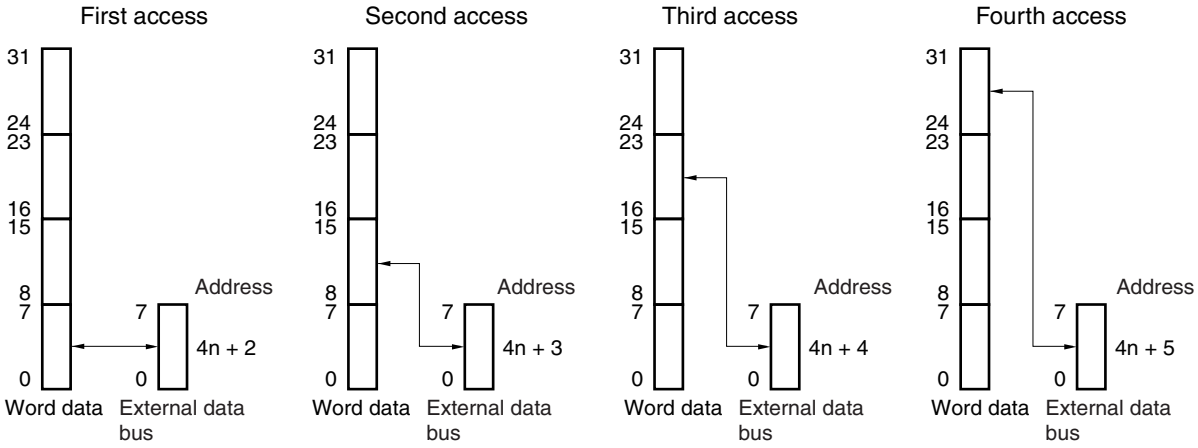


<2> Access to address (4n + 1)

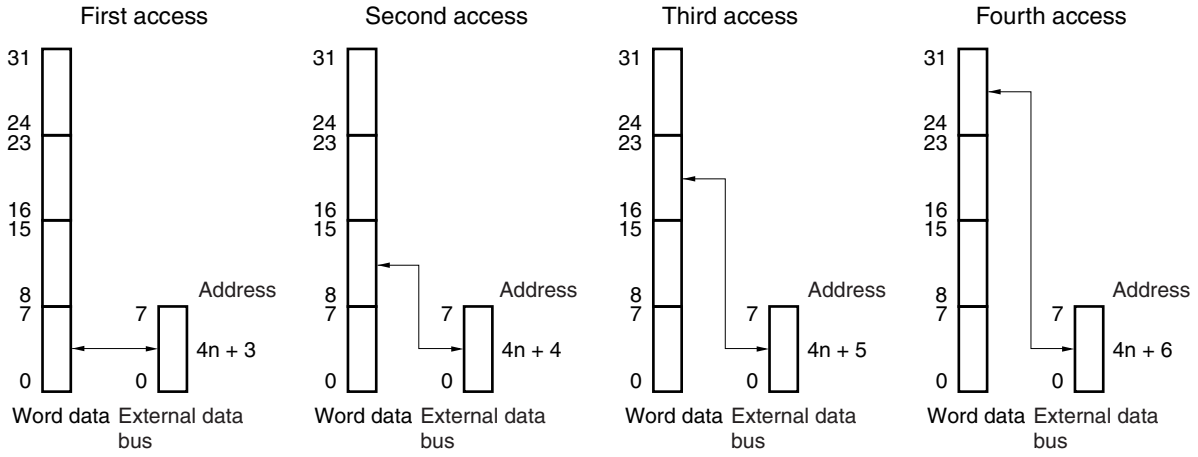


(b) 8-bit data bus width (2/2)

<3> Access to address (4n + 2)



<4> Access to address (4n + 3)



5.6 Wait Function

5.6.1 Programmable wait function

(1) Data wait control register 0 (DWC0)

To realize interfacing with a low-speed memory or I/O, up to seven data wait states can be inserted in the bus cycle that is executed for each memory block space.

The number of wait states can be programmed by using data wait control register 0 (DWC0). Immediately after system reset, 7 data wait states are inserted for all the blocks.

The DWC0 register can be read or written in 16-bit units.

Reset input sets this register to 7777H.

Cautions 1. The internal ROM and internal RAM areas are not subject to programmable wait, and are always accessed without a wait state. The on-chip peripheral I/O area is also not subject to programmable wait, and only wait control from each peripheral function is performed.

2. Write to the DWC0 register after reset, and then do not change the set values. Also, do not access an external memory area other than the one for this initialization routine until the initial settings of the DWC0 register are complete. However, external memory areas whose initial settings are complete may be accessed.

After reset: 7777H R/W Address: FFFFF484H

	15	14	13	12	11	10	9	8
DWC0	0	DW32	DW31	DW30	0	DW22	DW21	DW20
		Memory block 3				Memory block 2		

	7	6	5	4	3	2	1	0
	0	DW12	DW11	DW10	0	DW02	DW01	DW00
		Memory block 1				Memory block 0		

DWn2	DWn1	DWn0	Number of wait states inserted in memory block n space (n = 0 to 3)
0	0	0	None
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Caution Be sure to clear bits 15, 11, 7, and 3 to 0.

5.6.2 External wait function

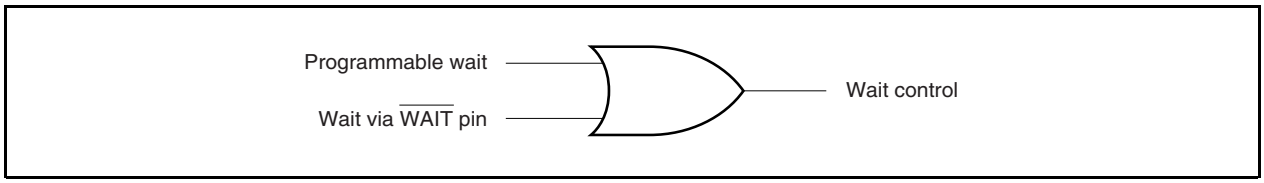
To synchronize an extremely slow external device, I/O, or asynchronous system, any number of wait states can be inserted in the bus cycle by using the external wait pin ($\overline{\text{WAIT}}$).

Access to each area of the internal ROM, internal RAM, and on-chip peripheral I/O is not subject to control by the external wait function, in the same manner as the programmable wait function.

The $\overline{\text{WAIT}}$ 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 in the multiplexed bus mode. In the separate bus mode, it is sampled at the rising edge of the clock immediately after the T1 and TW states of the bus cycle. If the setup/hold time of the sampling timing is not satisfied, a wait state is inserted in the next state, or not inserted at all.

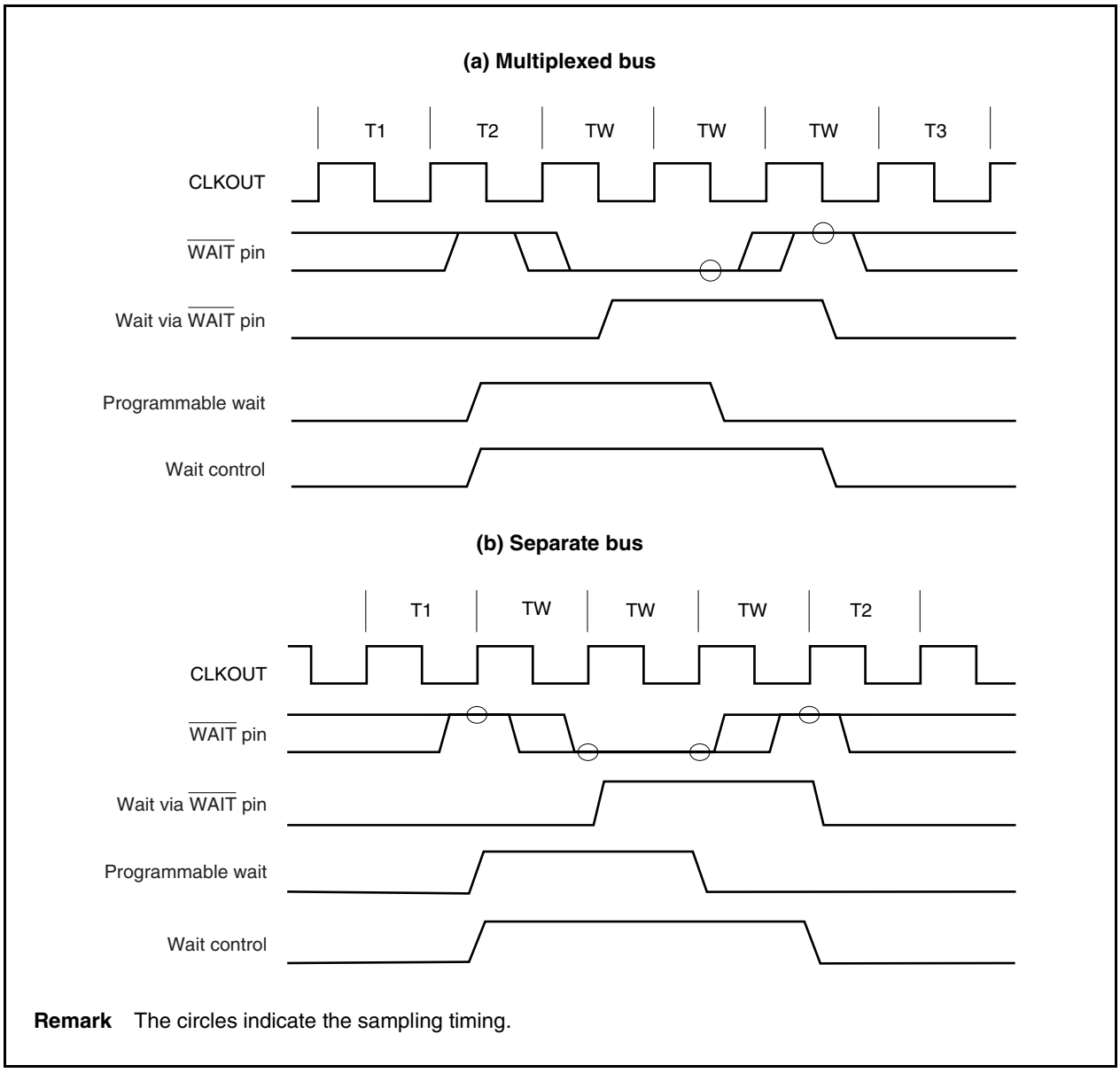
5.6.3 Relationship between programmable wait and external wait

Wait cycles are inserted as the result of an OR operation between the wait cycles specified by the set value of the programmable wait and the wait cycles controlled by the $\overline{\text{WAIT}}$ pin. In other words, the number of wait cycles is determined by the side with the greatest number of cycles.



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

Figure 5-3. Inserting Wait Example



Remark The circles indicate the sampling timing.

5.6.4 Programmable address wait function

Address-setup or address-hold waits to be inserted in each bus cycle can be set by using the address wait control register (AWC). Address wait insertion is set for each memory block area (memory blocks 0 to 3).

If an address setup wait is inserted, it seems that the high-clock period of the T1 state is extended by 1 clock. If an address hold wait is inserted, it seems that the low-clock period of the T1 state is extended by 1 clock.

(1) Address wait control register (AWC)

The AWC register can be read or written in 16-bit units.

Reset input sets this register to FFFFH.

After reset: FFFFH R/W Address: FFFFF488H

	15	14	13	12	11	10	9	8
AWC	1	1	1	1	1	1	1	1

7	6	5	4	3	2	1	0
AHW3	ASW3	AHW2	ASW2	AHW1	ASW1	AHW0	ASW0
Memory block 3		Memory block 2		Memory block 1		Memory block 0	

AHWn	Specifies insertion of address hold wait (n = 0 to 3)
0	Not inserted
1	Inserted

ASWn	Specifies insertion of address setup wait (n = 0 to 3)
0	Not inserted
1	Inserted

Caution Be sure to set bits 15 to 8 to 1.

5.7 Idle State Insertion Function

To facilitate interfacing with low-speed memories, one idle state (TI) can be inserted after the T3 state in the bus cycle that is executed for each space selected by the chip select function in the multiplexed address/data bus mode. In the separate bus mode, one idle state (TI) can be inserted after the T2 state. By inserting an idle state, the data output float delay time of the memory can be secured during read access (an idle state cannot be inserted during write access).

Whether the idle state is to be inserted can be programmed by using the bus cycle control register (BCC).

An idle state is inserted for all the areas immediately after system reset.

(1) Bus cycle control register (BCC)

The BCC register can be read or written in 16-bit units.

Reset input sets this register to AAAAH.

Cautions 1. The internal ROM, internal RAM, and on-chip peripheral I/O areas are not subject to idle state insertion.

2. Write to the BCC register after reset, and then do not change the set values. Also, do not access an external memory area other than the one for this initialization routine until the initial settings of the BCC register are complete. However, external memory areas whose initial settings are complete may be accessed.

After reset: AAAAH R/W Address: FFFFF48AH

	15	14	13	12	11	10	9	8
BCC	1	0	1	0	1	0	1	0

7	6	5	4	3	2	1	0
BC31	0	BC21	0	BC11	0	BC01	0

Memory block 3 Memory block 2 Memory block 1 Memory block 0

BCn1	Specifies insertion of idle state (n = 0 to 3)
0	Not inserted
1	Inserted

Caution Be sure to set bits 15, 13, 11, and 9 to 1, and clear bits 14, 12, 10, 8, 6, 4, 2, and 0 to 0.

5.8 Bus Hold Function

5.8.1 Functional outline

The $\overline{\text{HLDAK}}$ and $\overline{\text{HLDRQ}}$ functions are valid if the PCM2 and PCM3 pins are set in the control mode.

When the $\overline{\text{HLDRQ}}$ pin is asserted (low level), indicating that another bus master has requested bus mastership, the external address/data bus goes into a high-impedance state and is released (bus hold status). If the request for the bus mastership is cleared and the $\overline{\text{HLDRQ}}$ pin is deasserted (high level), driving these pins is started again.

During the bus hold period, execution of the program in the internal ROM and internal RAM is continued until a peripheral I/O register or the external memory is accessed.

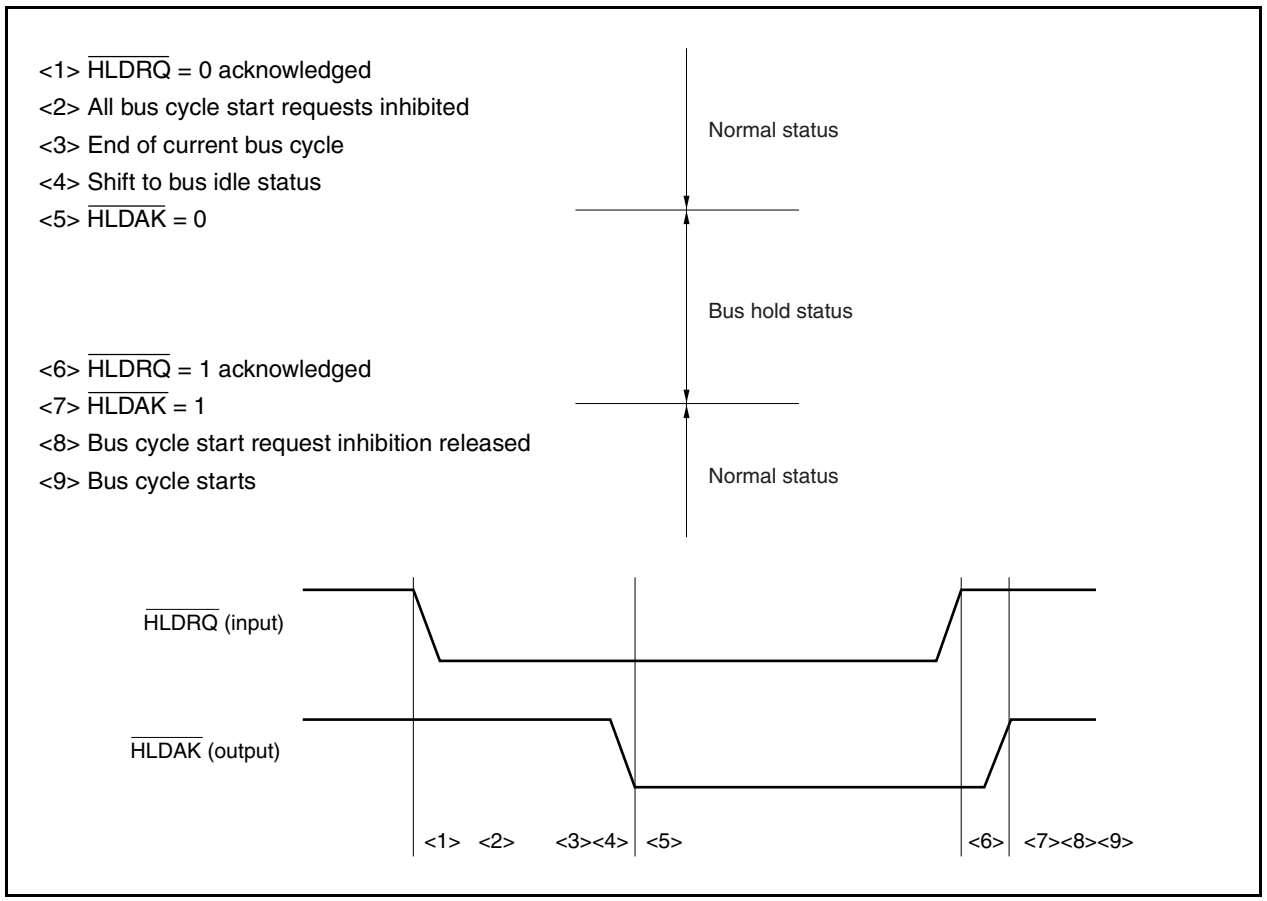
The bus hold status is indicated by assertion of the $\overline{\text{HLDAK}}$ pin (low level). The bus hold function enables the configuration of multi-processor type systems in which two or more bus masters exist.

Note that the bus hold request is not acknowledged during a multiple-access cycle initiated by the bus sizing function or a bit manipulation instruction.

Status	Data Bus Width	Access Type	Timing in Which Bus Hold Request Is Not Acknowledged
CPU bus lock	16 bits	Word access to even address	Between first and second access
		Word access to odd address	Between first and second access
			Between second and third access
		Halfword access to odd address	Between first and second access
	8 bits	Word access	Between first and second access
			Between second and third access
			Between third and fourth access
		Halfword access	Between first and second access
Read-modify-write access of bit manipulation instruction	–	–	Between read access and write access

5.8.2 Bus hold procedure

The bus hold status transition procedure is shown below.



5.8.3 Operation in power save mode

Because the internal system clock is stopped in the software STOP, IDLE1, and IDLE2 modes, the bus hold status is not entered even if the $\overline{\text{HLDRQ}}$ pin is asserted.

In the HALT mode, the $\overline{\text{HLDAK}}$ pin is asserted as soon as the $\overline{\text{HLDRQ}}$ pin has been asserted, and the bus hold status is entered. When the $\overline{\text{HLDRQ}}$ pin is later deasserted, the $\overline{\text{HLDAK}}$ pin is also deasserted, and the bus hold status is cleared.

5.9 Bus Priority

Bus hold, instruction fetch (branch), instruction fetch (successive), and operand data accesses are executed in the external bus cycle.

Bus hold has the highest priority, followed by operand data access, instruction fetch (branch), and instruction fetch (successive).

An instruction fetch may be inserted between the read access and write access in a read-modify-write access.

If an instruction is executed for two or more accesses, an instruction fetch and bus hold are not inserted between accesses due to bus size limitations.

Table 5-4. Bus Priority

Priority	External Bus Cycle	Bus Master
High ↕ Low	Bus hold	External device
	DMA transfer	DMAC
	Operand data access	CPU
	Instruction fetch (branch)	CPU
	Instruction fetch (successive)	CPU

5.10 Boundary Operation Conditions

5.10.1 Program space

(1) If a branch instruction exists at the upper limit of the internal RAM area, a prefetch operation straddling over the on-chip peripheral I/O area (invalid fetch) does not occur.

(2) Instruction execution to the external memory area cannot be continued without a branch from the internal ROM area to the external memory area.

5.10.2 Data space

The V850ES/SG2 has an address misalign function.

With this function, data can be placed at all addresses, regardless of the format of the data (word data or halfword data). However, if the word data or halfword data is not aligned at the boundary, a bus cycle is generated at least twice, causing the bus efficiency to drop.

(1) Halfword-length data access

A byte-length bus cycle is generated twice if the least significant bit of the address is 1.

(2) Word-length data access

(a) A byte-length bus cycle, halfword-length bus cycle, and byte-length bus cycle are generated in that order if the least significant bit of the address is 1.

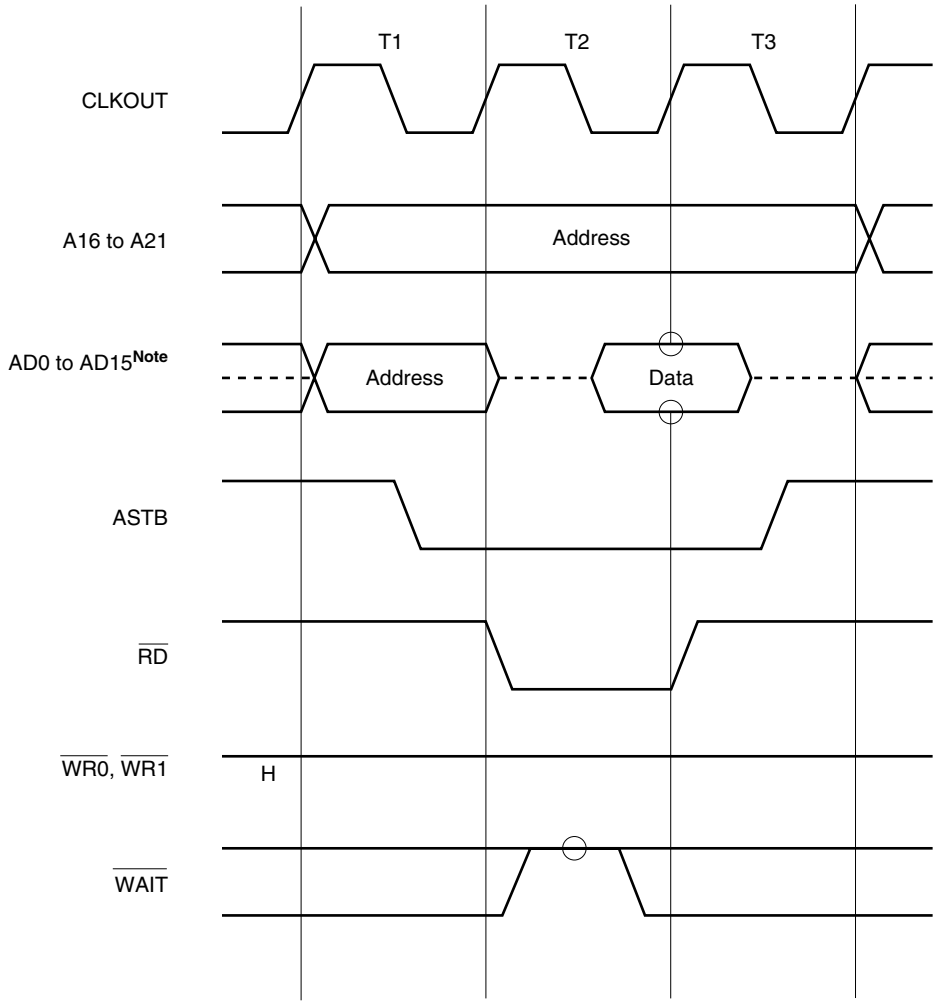
(b) A halfword-length bus cycle is generated twice if the lower 2 bits of the address are 10.

5.11 Bus Timing

5.11.1 Multiplexed bus

(1) Read cycle

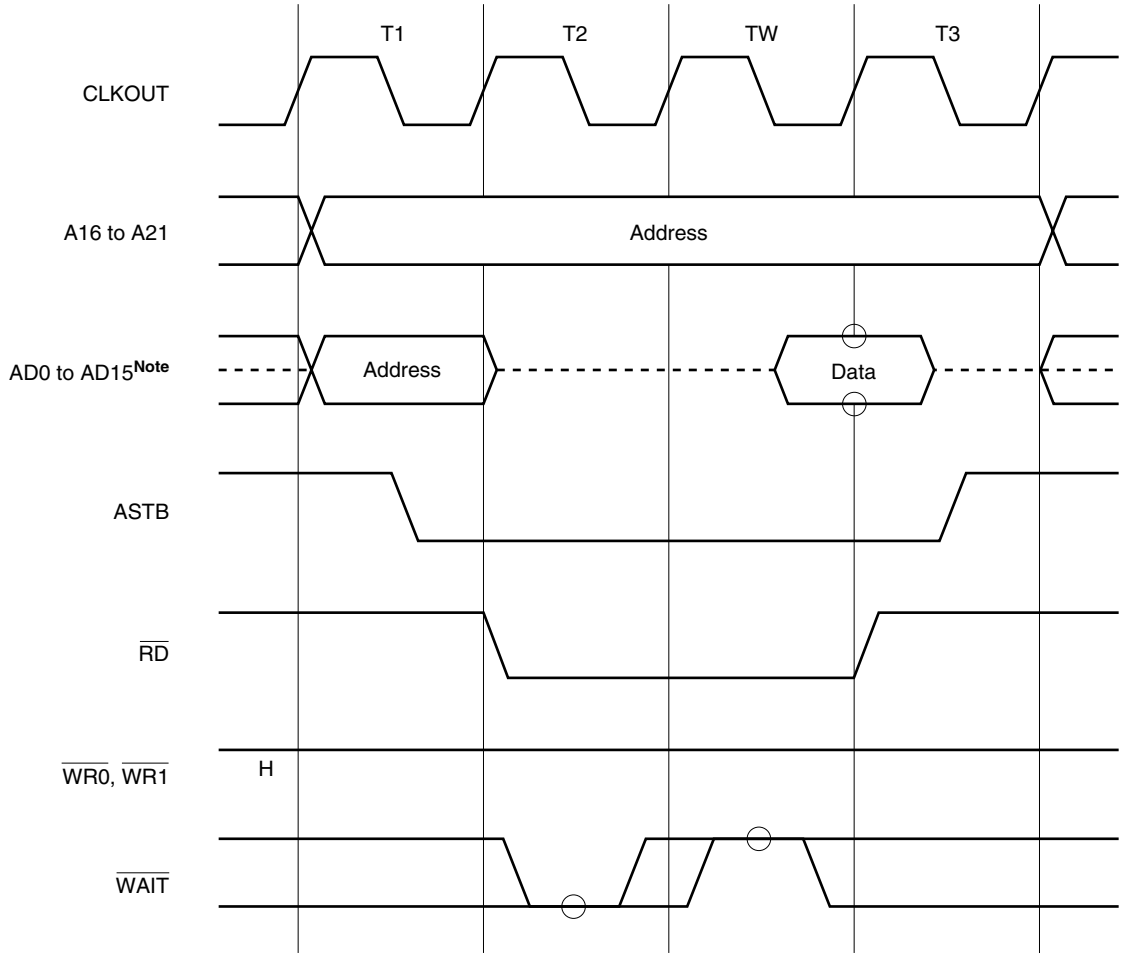
Figure 5-4. Basic Bus Cycle



Note AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

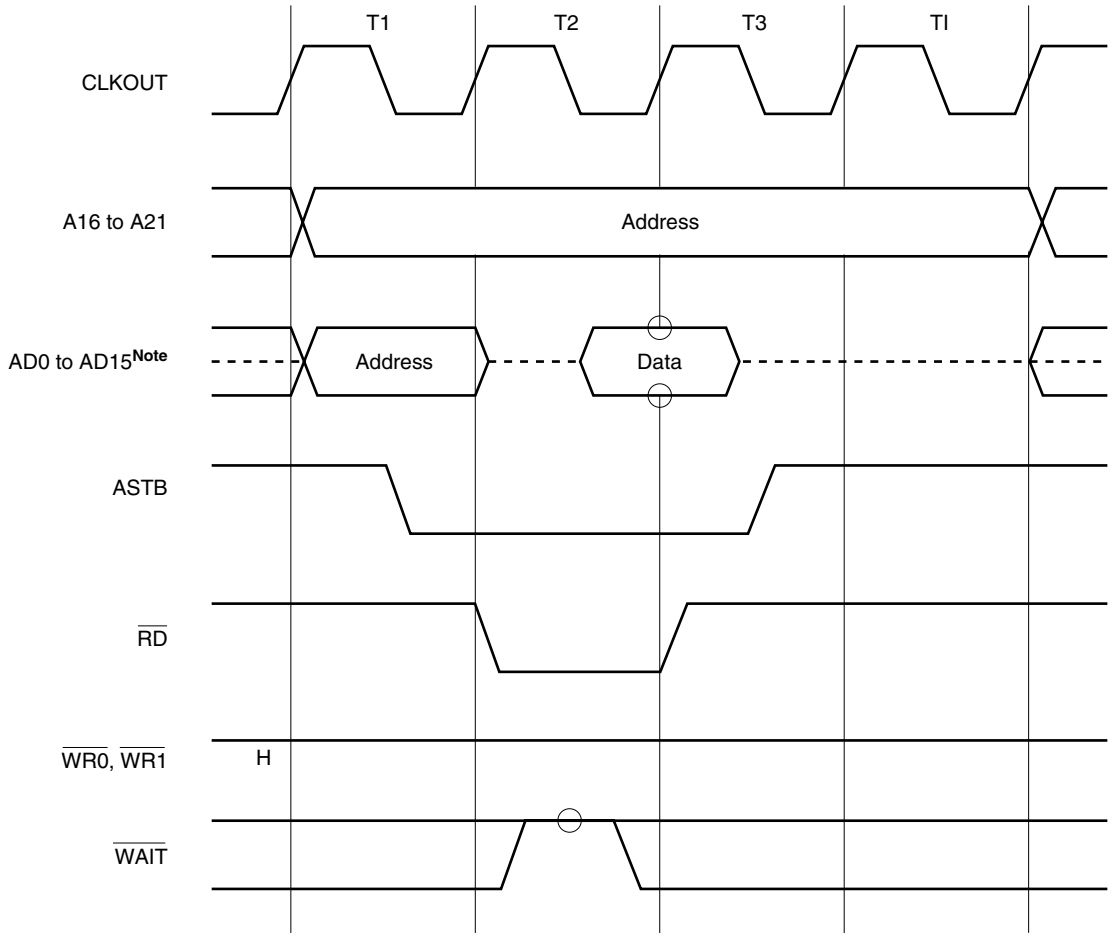
Figure 5-5. When Wait State (1 Wait) Is Inserted



Note AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

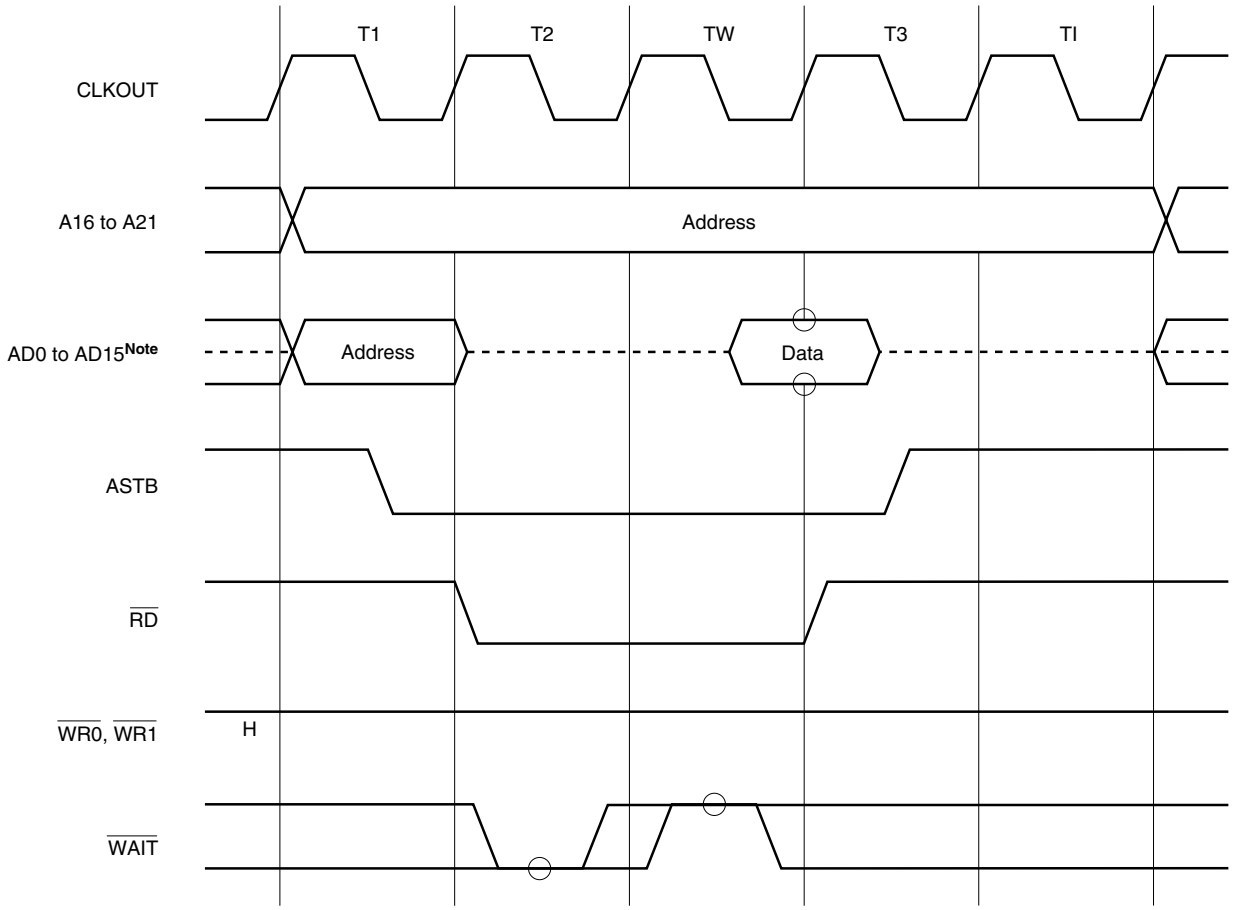
Figure 5-6. When Idle State Is Inserted



Note AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

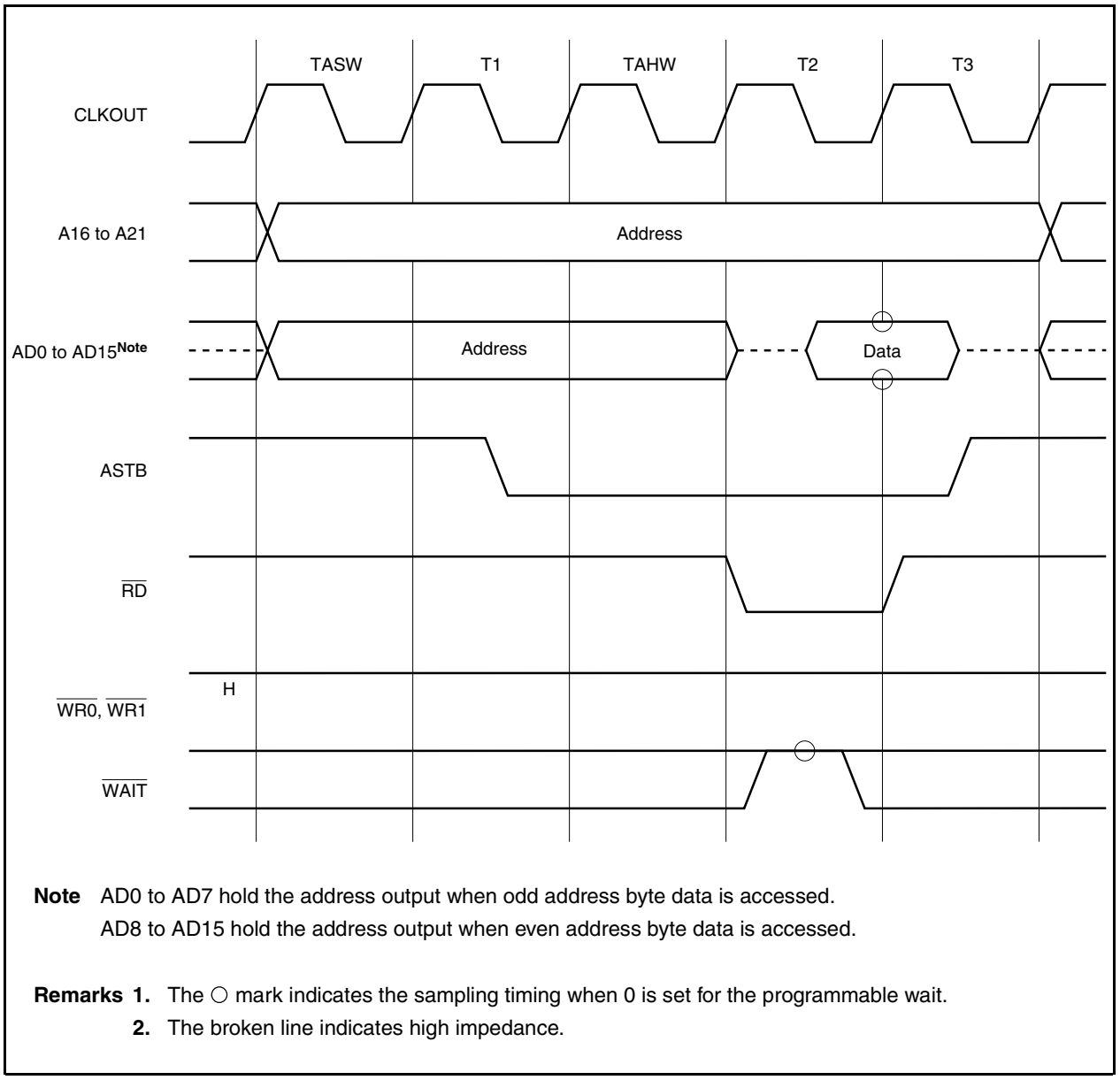
Figure 5-7. When Wait State (1 Wait) and Idle State Are Inserted



Note AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

Figure 5-8. When Address Wait State Is Inserted

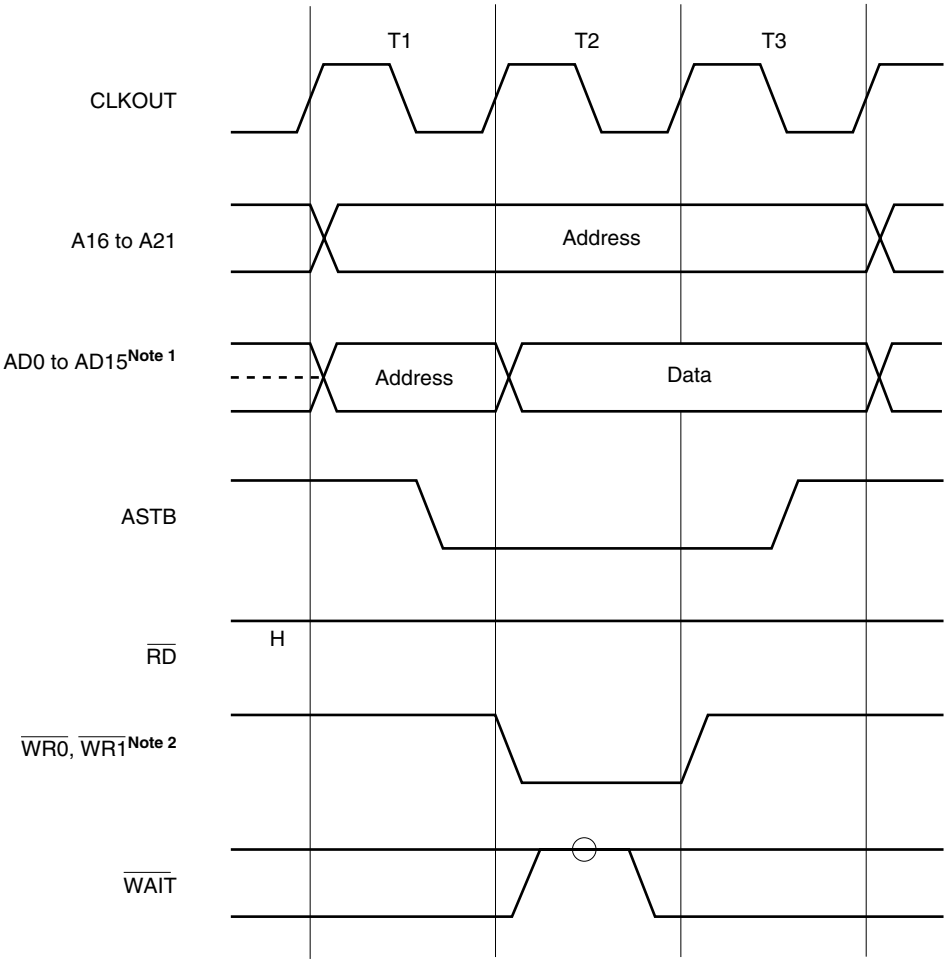


Note AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

(2) Write cycle

Figure 5-9. Basic Bus Cycle



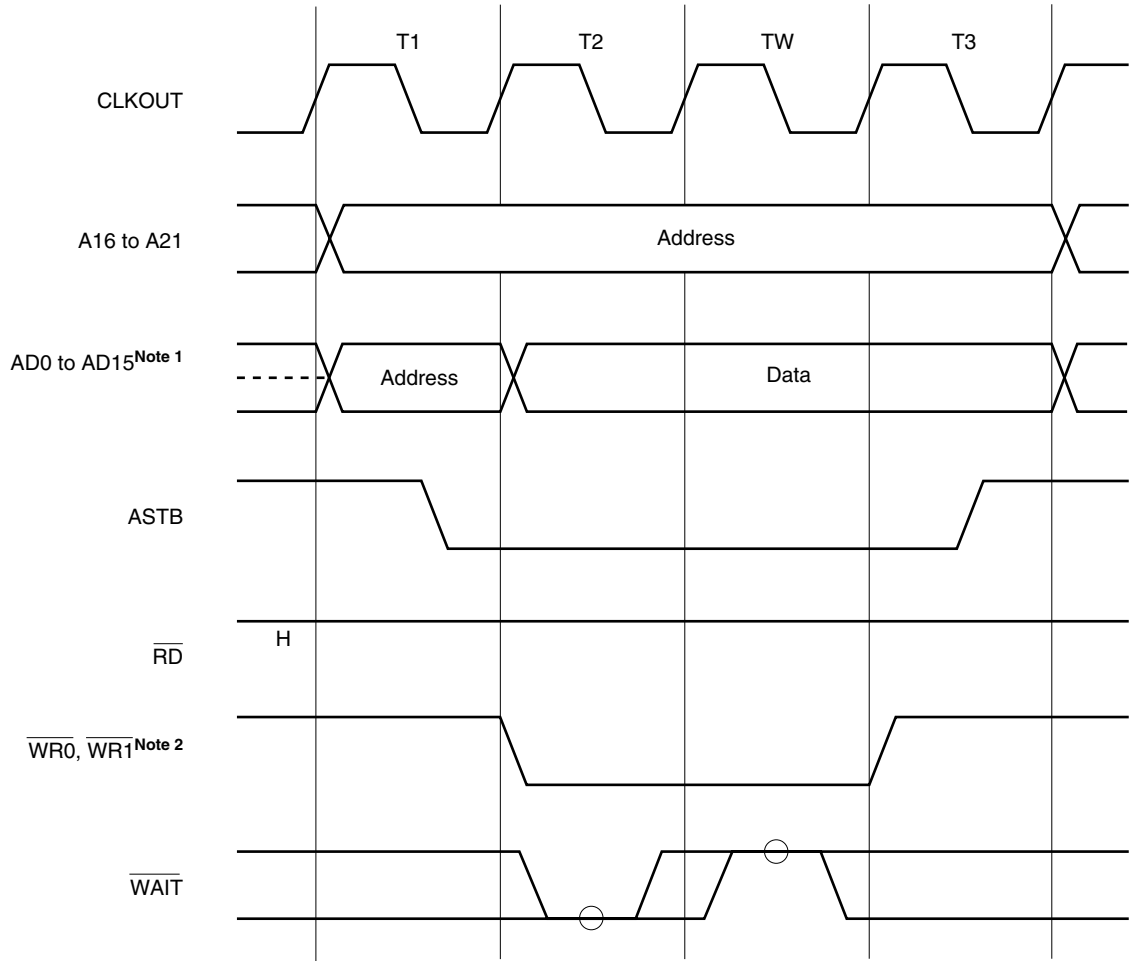
Notes 1. AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

Figure 5-10. When Wait State (1 Wait) Is Inserted



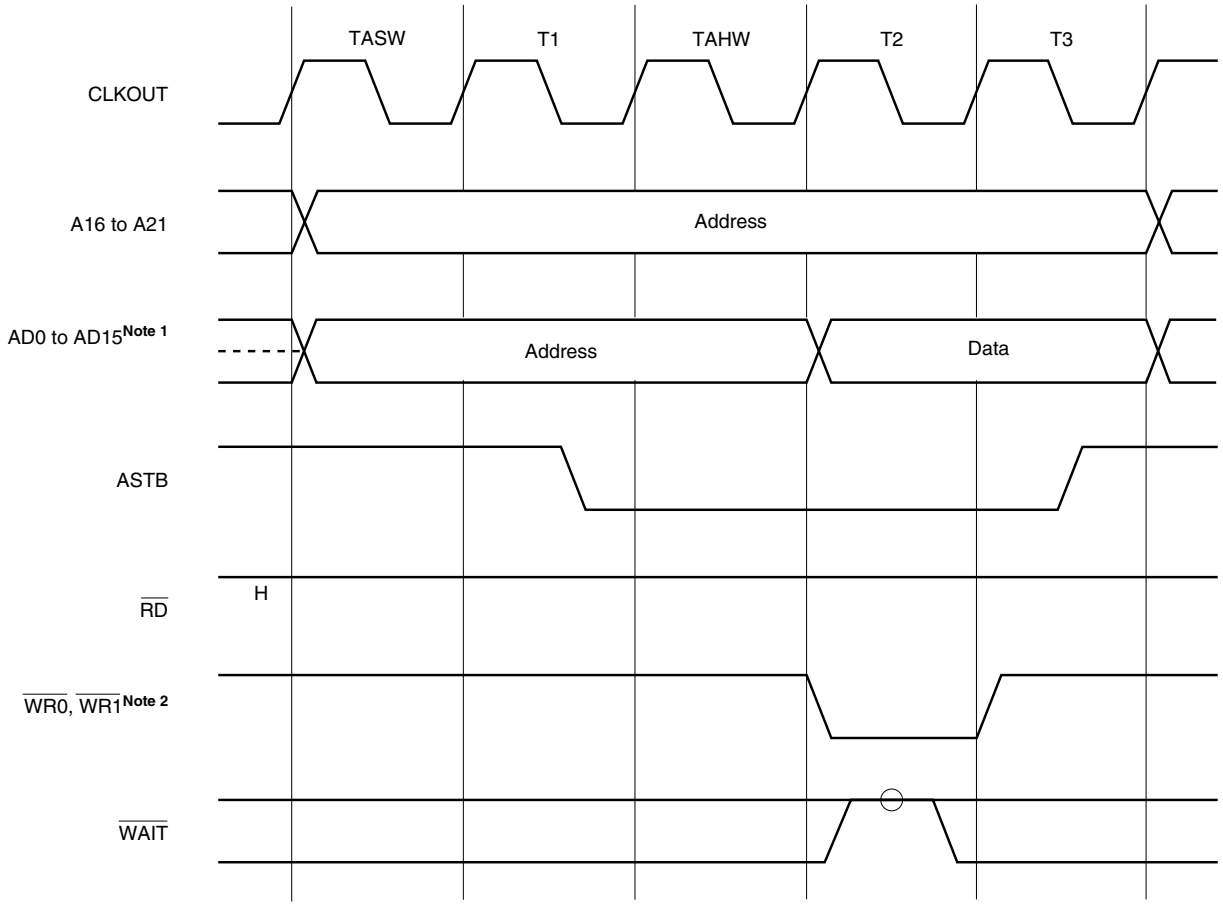
Notes 1. AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

Figure 5-11. When Address Wait State Is Inserted



Notes 1. AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

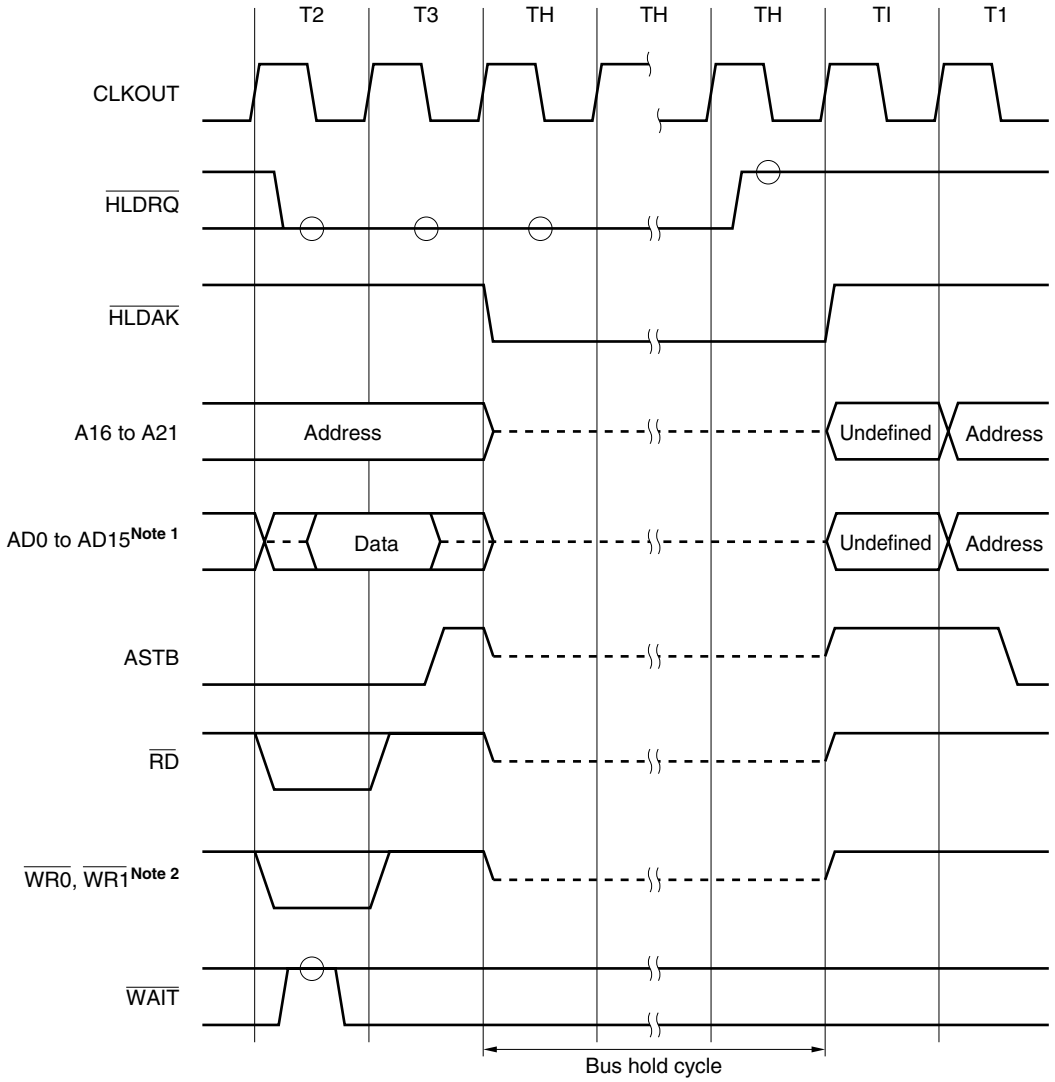
2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

(3) Bus hold cycle

Figure 5-12. Bus Hold Cycle



Notes 1. AD0 to AD7 hold the address output when odd address byte data is accessed.
AD8 to AD15 hold the address output when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. Upon detection of a low level in the T2 and T3 states of $\overline{HLDRQ}$ (sampling timing), the operation moves on to the bus hold cycle after the T3 state ends. Thereafter, upon detection of a low level or high level in the TH state (sampling timing), the bus hold status is maintained after the TH state ends, or the bus cycle is restarted.

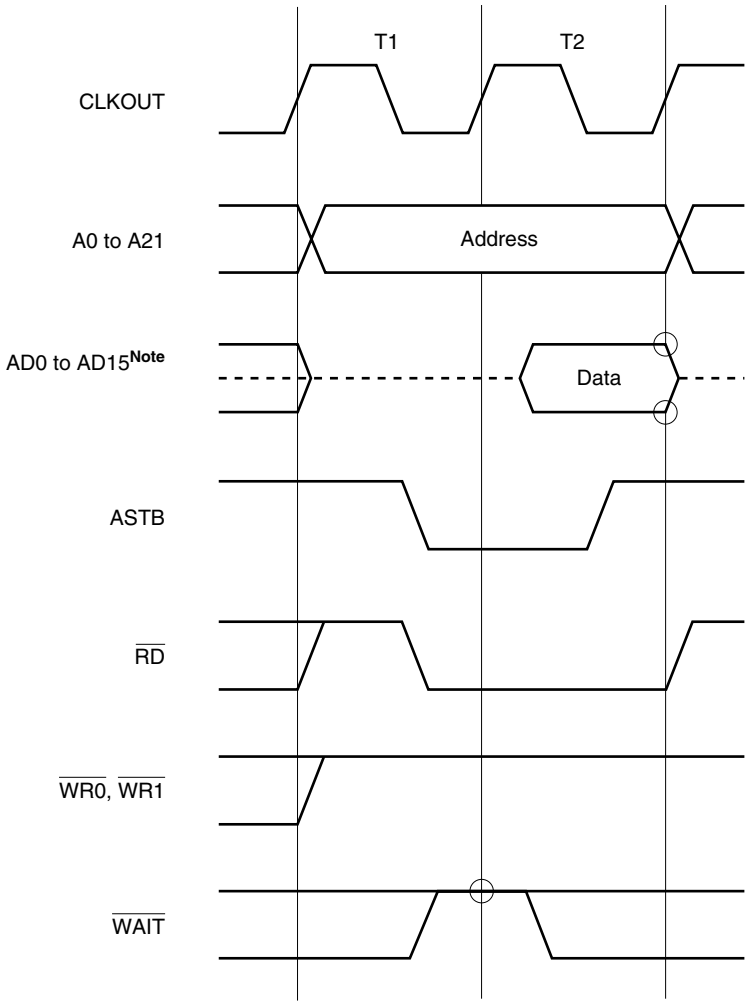
2. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

3. The broken line indicates high impedance.

5.11.2 Separate bus

(1) Read cycle

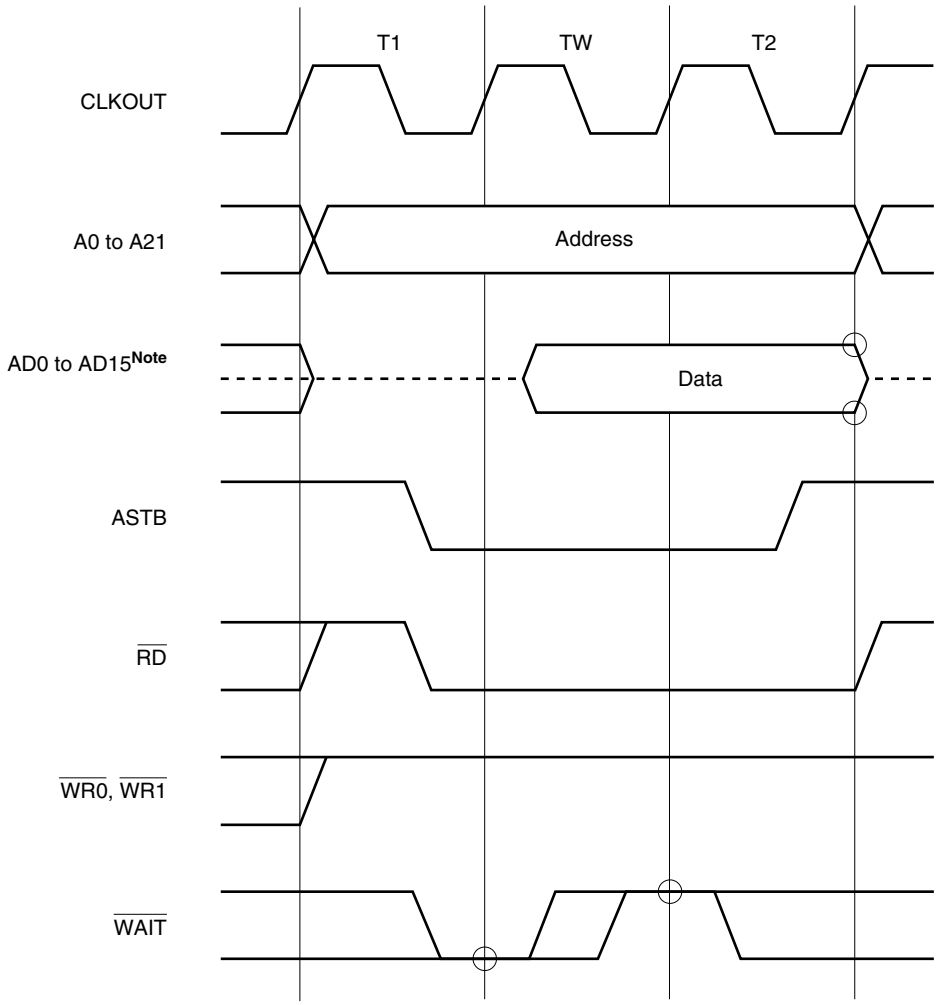
Figure 5-13. Basic Bus Cycle



Note AD0 to AD7 go into a high-impedance state when odd address byte data is accessed.
AD8 to AD15 go into a high-impedance state when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

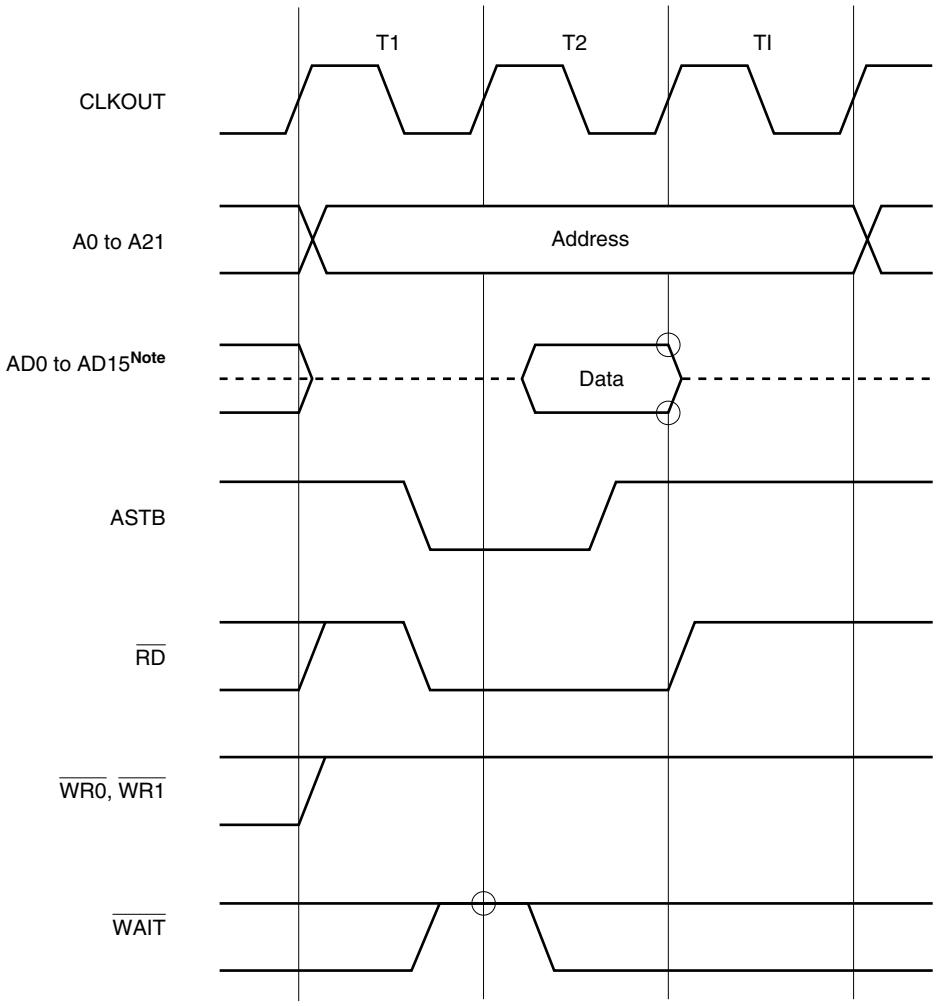
Figure 5-14. When Wait State (1 Wait) Is Inserted



Note AD0 to AD7 go into a high-impedance state when odd address byte data is accessed.
AD8 to AD15 go into a high-impedance state when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

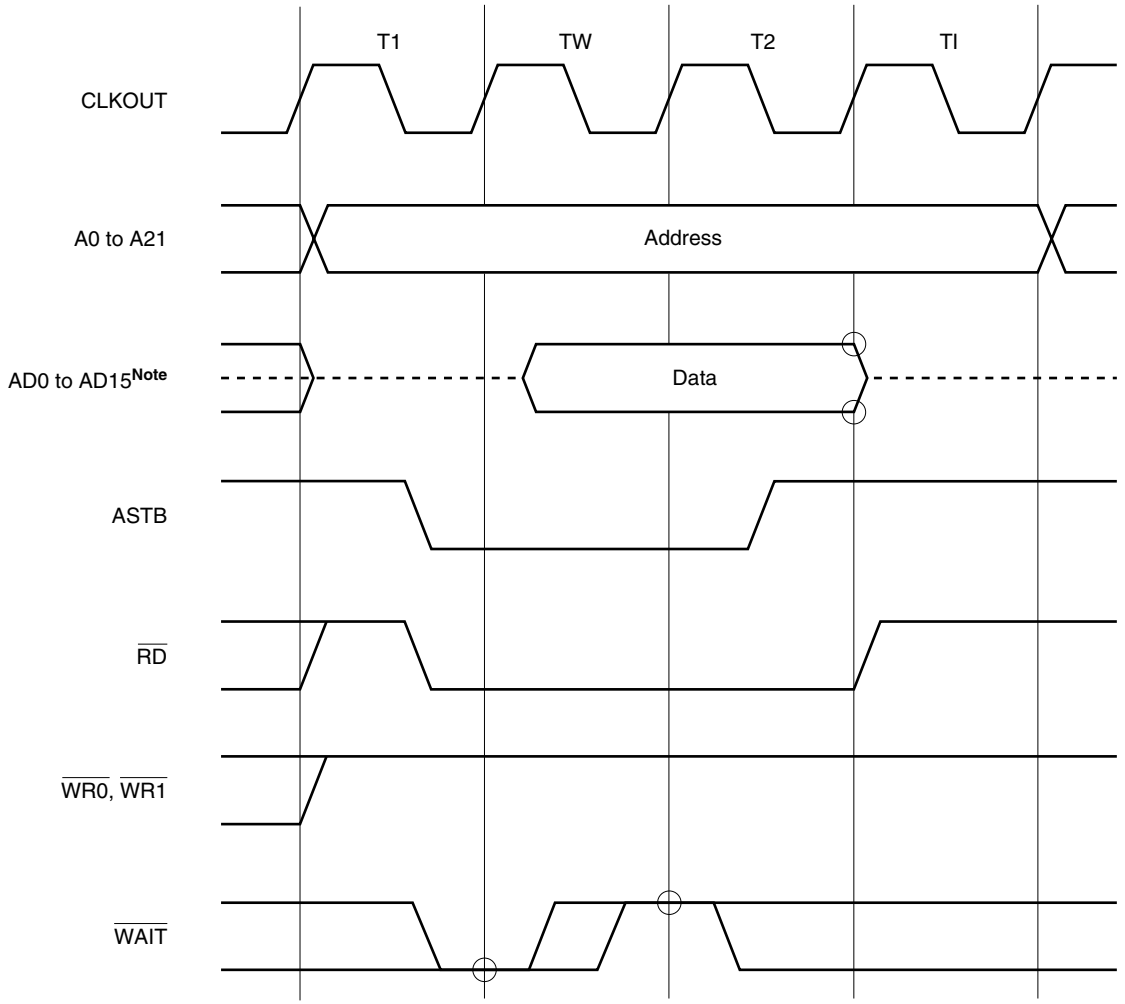
Figure 5-15. When Idle State Is Inserted



Note AD0 to AD7 go into a high-impedance state when odd address byte data is accessed.
AD8 to AD15 go into a high-impedance state when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

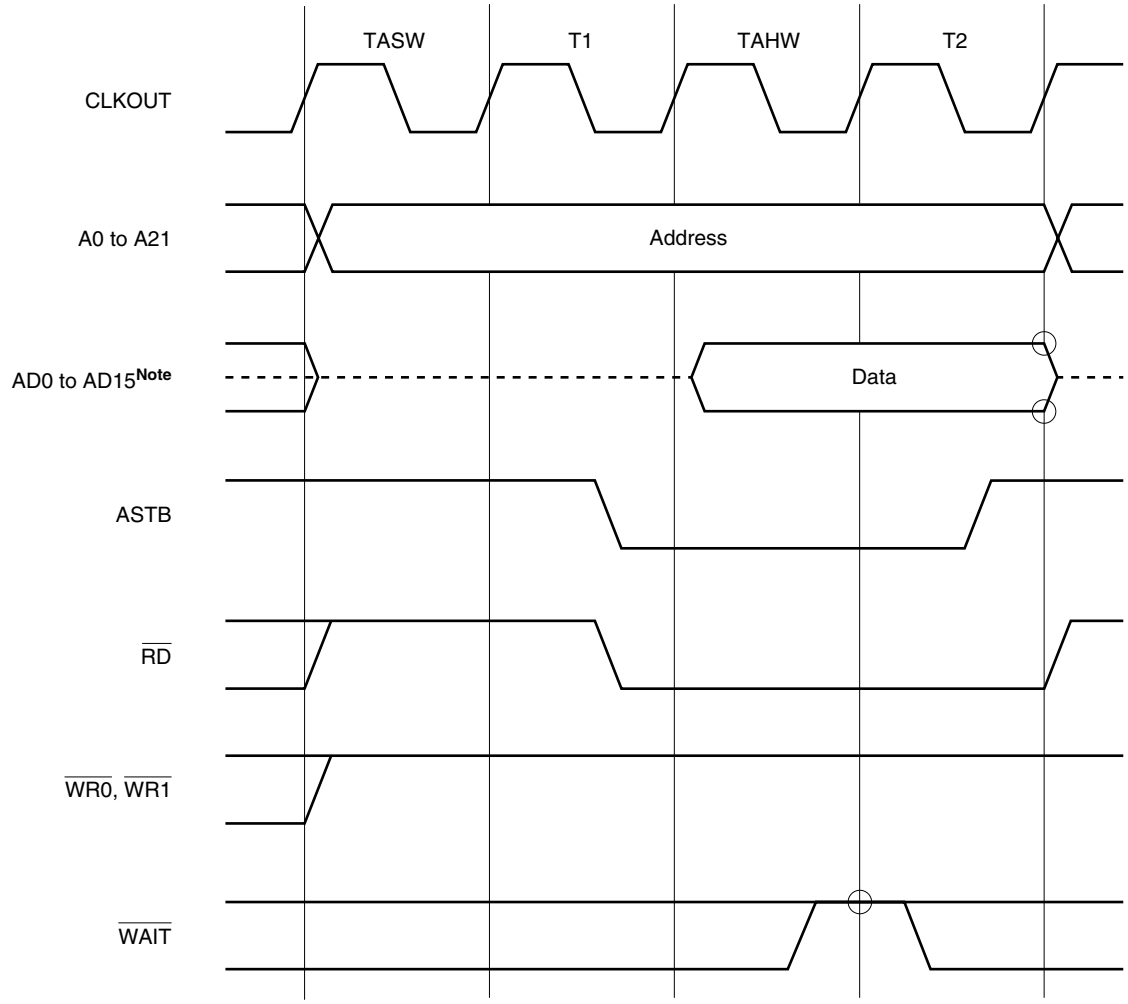
Figure 5-16. When Wait State (1 Wait) and Idle State Are Inserted



Note AD0 to AD7 go into a high-impedance state when odd address byte data is accessed.
AD8 to AD15 go into a high-impedance state when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.
2. The broken line indicates high impedance.

Figure 5-17. When Address Wait State Is Inserted



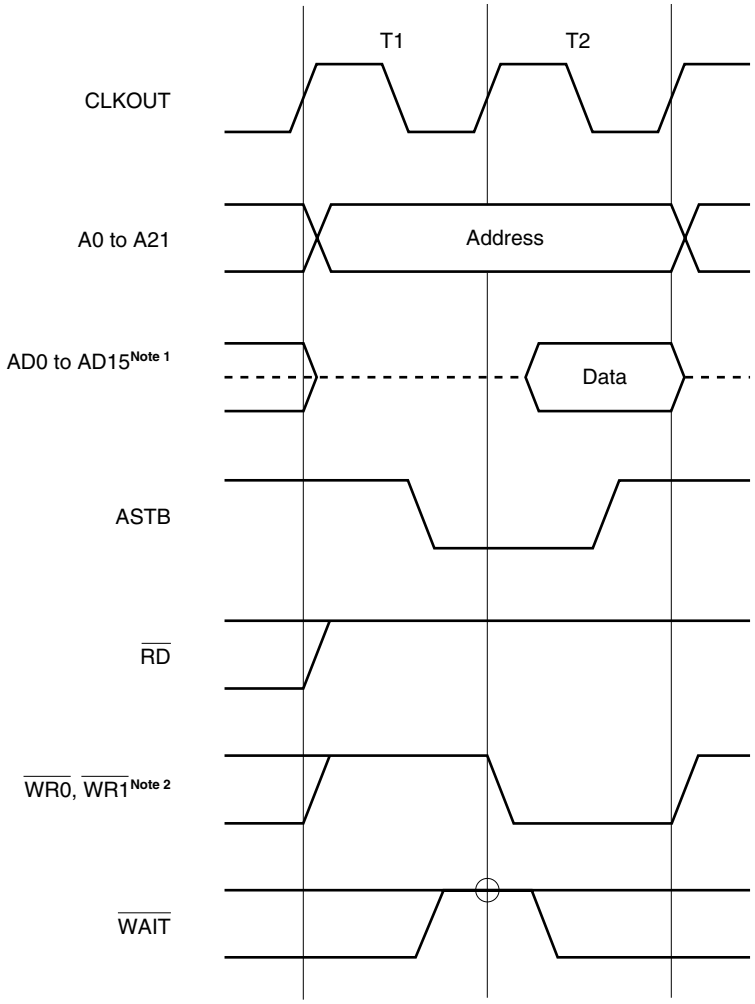
Note AD0 to AD7 go into a high-impedance state when odd address byte data is accessed.
AD8 to AD15 go into a high-impedance state when even address byte data is accessed.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

(2) Write cycle

Figure 5-18. Basic Bus Cycle



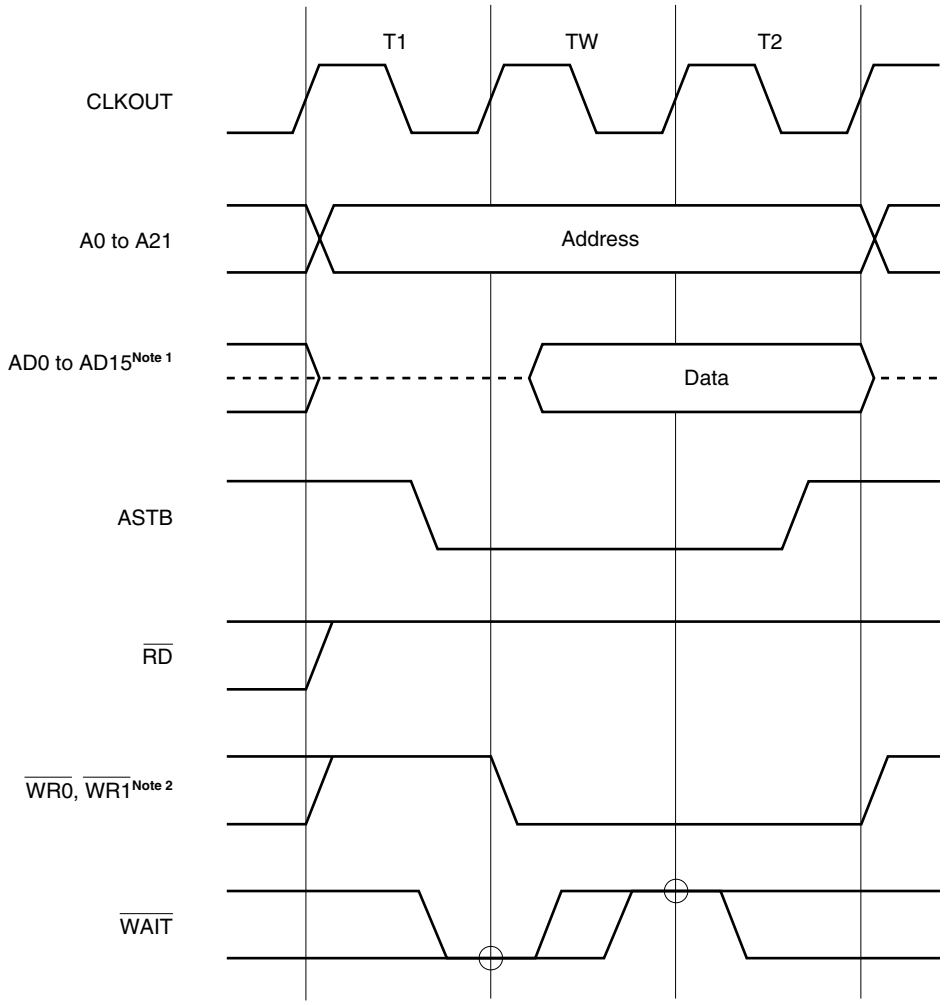
Notes 1. AD0 to AD7 become undefined when odd address byte data is accessed.
AD8 to AD15 become undefined when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

Figure 5-19. When Wait State (1 Wait) Is Inserted



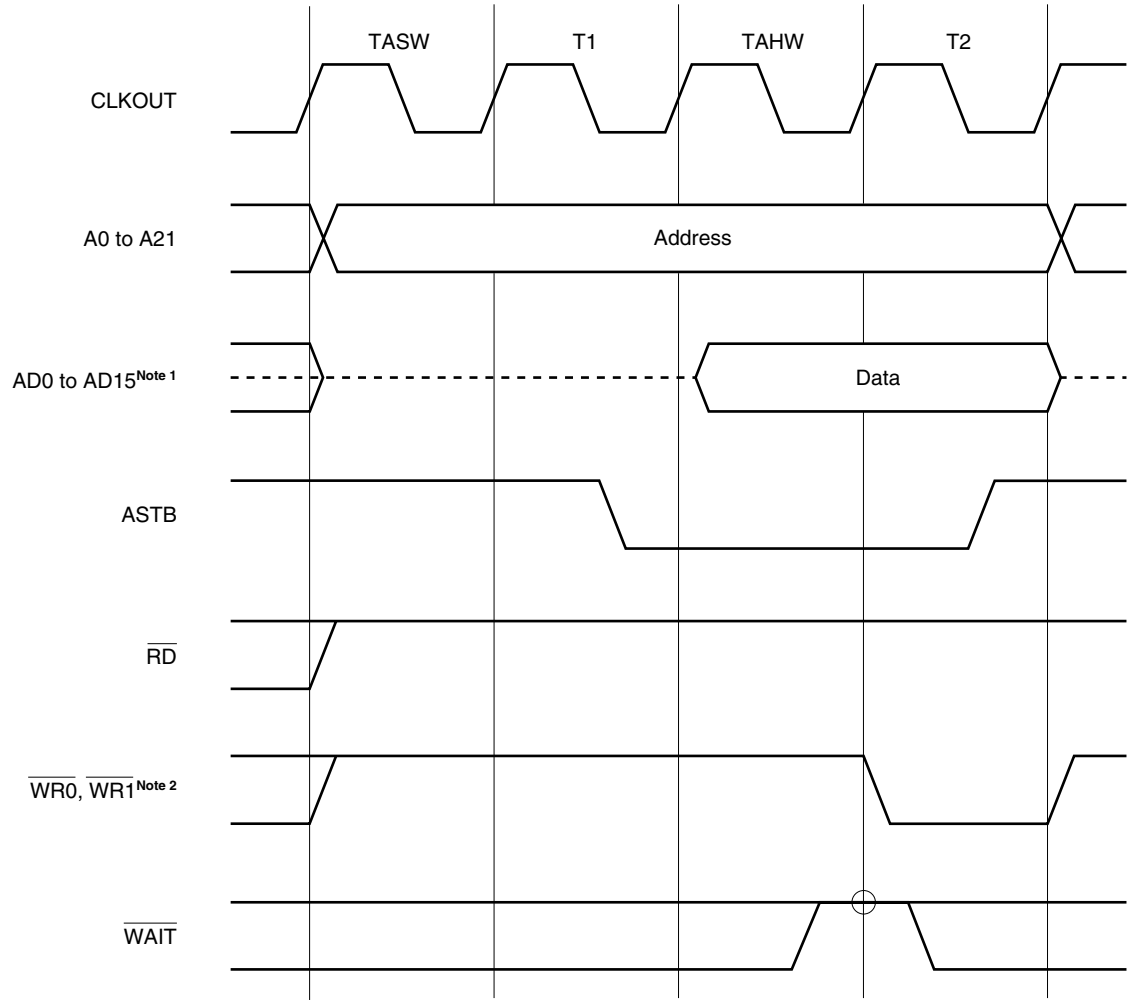
Notes 1. AD0 to AD7 become undefined when odd address byte data is accessed.
AD8 to AD15 become undefined when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

Figure 5-20. When Address Wait State Is Inserted



Notes 1. AD0 to AD7 become undefined when odd address byte data is accessed.
AD8 to AD15 become undefined when even address byte data is accessed.

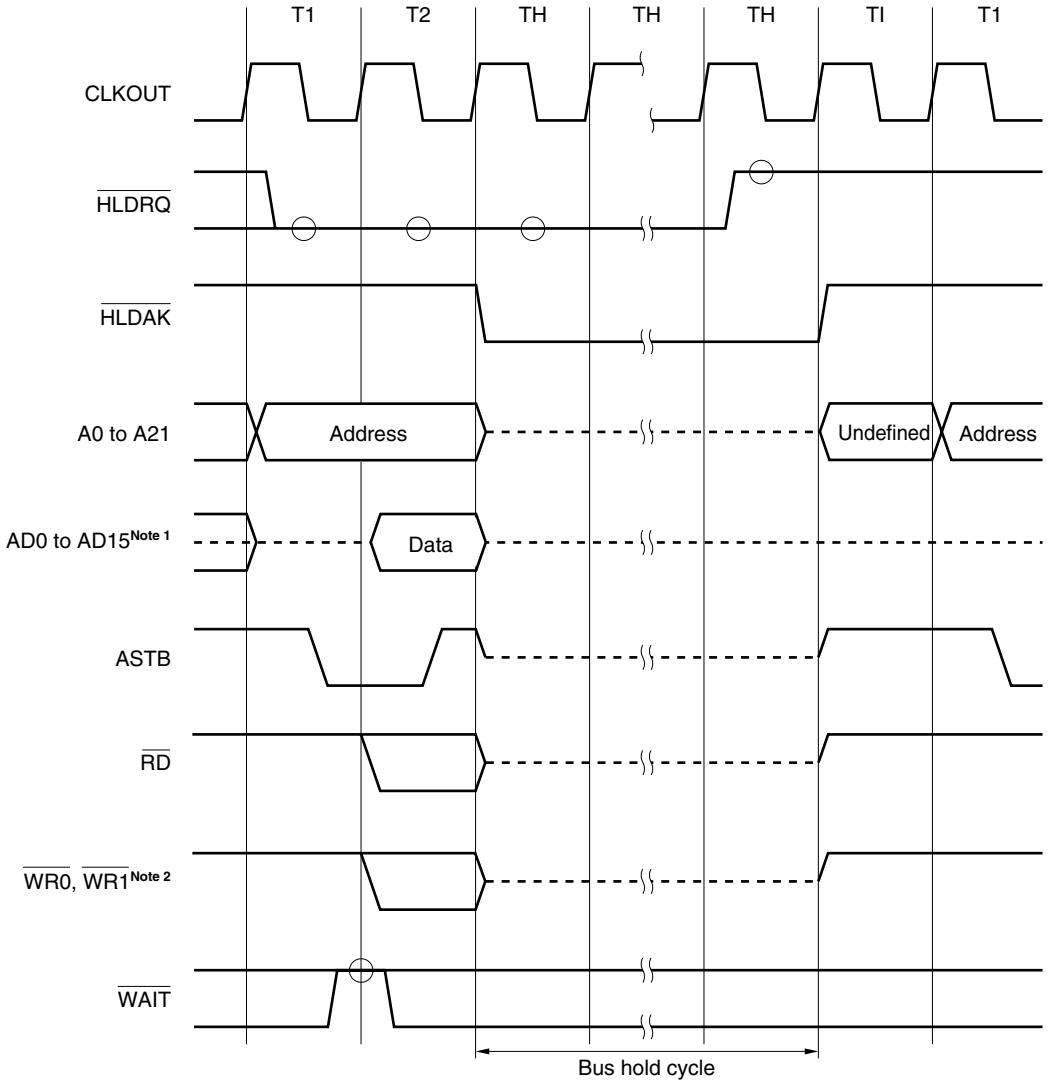
2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

2. The broken line indicates high impedance.

(3) Bus hold cycle

Figure 5-21. Bus Hold Cycle



Notes 1. AD0 to AD7 become undefined when odd address byte data is accessed.
AD8 to AD15 become undefined when even address byte data is accessed.

2. $\overline{WR0}$ and $\overline{WR1}$ output a low level as shown in the above timing chart when target data access is performed. At all other times, these pins output a high level.

Remarks 1. Upon detection of a low level in the T2 and T3 states of $\overline{HLDRQ}$ (sampling timing), the operation moves on to the bus hold cycle after the T3 state ends. Thereafter, upon detection of a low level or high level in the TH state (sampling timing), the bus hold status is maintained after the TH state ends, or the bus cycle is restarted.

2. The ○ mark indicates the sampling timing when 0 is set for the programmable wait.

3. The broken line indicates high impedance.

CHAPTER 6 CLOCK GENERATION FUNCTION

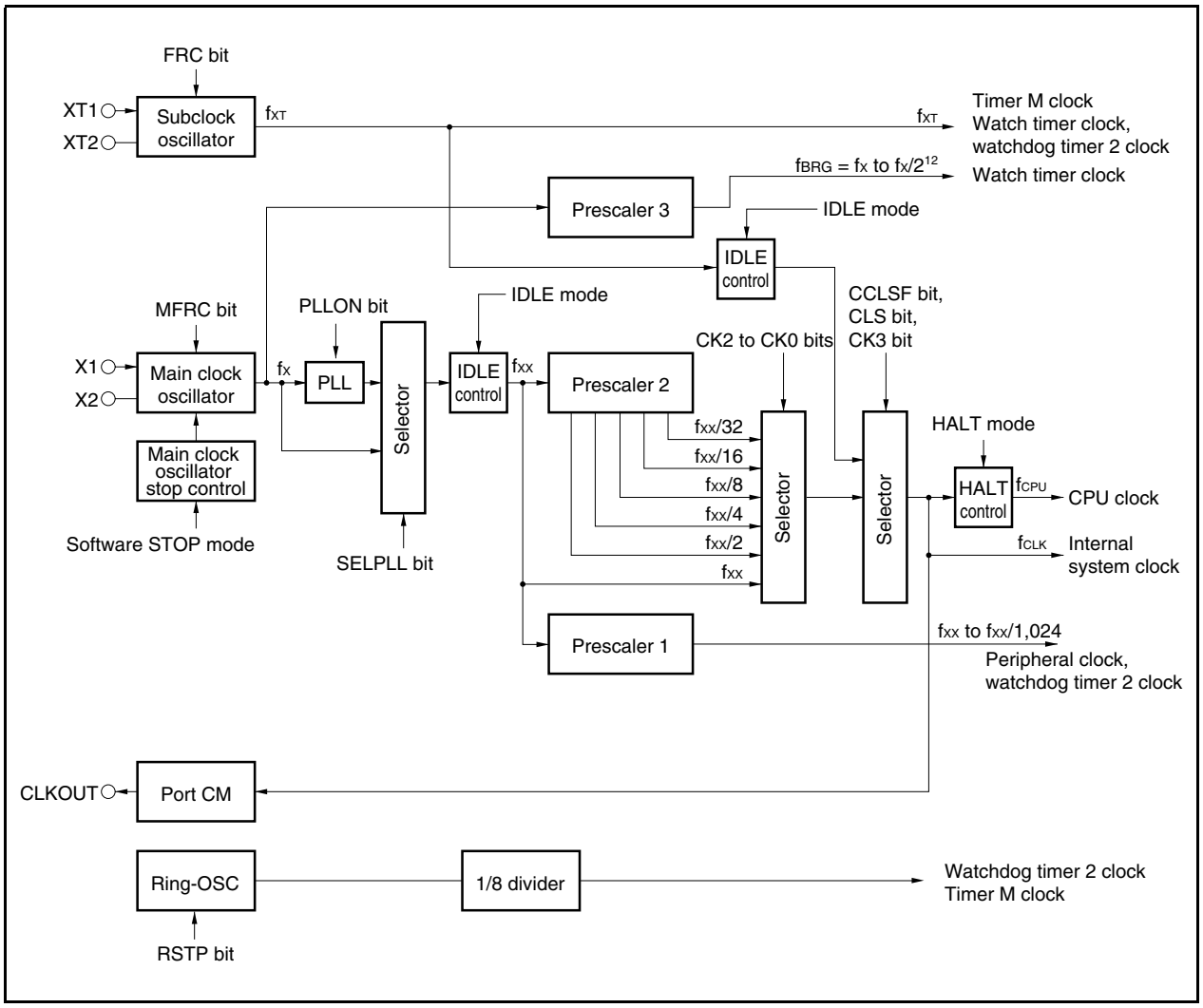
6.1 Overview

The following clock generation functions are available.

- ○ Main clock oscillator
  - In clock-through mode
    f_X = 2.5 to 10 MHz (f_{XX} = 2.5 to 10 MHz)
  - In PLL mode
    f_X = 2.5 to 5 MHz (f_{XX} = 10 to 20 MHz)
- ○ Subclock oscillator (sub-resonator)
  - 32.768 kHz
- ○ Multiply (×4/×8) function by PLL (Phase Locked Loop)
  - Clock-through mode/PLL mode selectable
- ○ Ring OSC
  - f_R = 200 kHz (TYP.)
- ○ Internal system clock generation
  - 7 steps (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, f_{XT})
- ○ Peripheral clock generation
- ○ Clock output function

6.2 Configuration

Figure 6-1. Clock Generator



(1) Main clock oscillator

The main resonator oscillates the following frequencies (fx).

- In clock-through mode
  fx = 2.5 to 10 MHz (internal fxx = 2.5 to 10 MHz)
- In PLL mode
  fx = 2.5 to 5 MHz (internal fxx = 10 to 20 MHz)

(2) Subclock oscillator

The sub-resonator oscillates a frequency of 32.768 kHz (fXT).

(3) Main clock oscillator stop control

This circuit generates a control signal that stops oscillation of the main clock oscillator.

Oscillation of the main clock oscillator is stopped in the software STOP mode or when the MCK bit of the PCC register = 1 (valid only when the CLS bit of the PCC register = 1).

(4) Ring-OSC

Outputs a frequency (f_R) of 200 kHz (TYP.).

(5) Prescaler 1

This prescaler generates the clock (f_{XX} to $f_{XX}/1,024$) to be supplied to the following on-chip peripheral functions: TMP0 to TMP5, TMQ, TMM, CSIB0 to CSIB4, UARTA0 to UARTA2, I^2C00 to I^2C02, ADC, DAC, and WDT2

(6) Prescaler 2

This circuit divides the CPU clock (f_{CPU}) and main clock (f_{XX}).

The clock generated by prescaler 2 (f_{XX} to $f_{XX}/32$) is supplied to the selector that generates the internal system clock (f_{CLK}).

f_{CLK} is the clock supplied to the INTC, ROM correction, ROM, and RAM blocks, and can be output from the CLKOUT pin.

(7) Prescaler 3

This circuit divides the clock generated by the main clock oscillator (f_X) to a specific frequency (32.768 kHz) and supplies that clock to the watch timer block.

For details, see **CHAPTER 9 WATCH TIMER**.

(8) PLL

This circuit multiplies the clock generated by the main clock oscillator (f_X) by 4 or 8.

It operates in two modes: clock-through mode in which f_X is output as is, and PLL mode in which a multiplied clock is output. These modes can be selected by using the SELPLL bit of the PLL control register (PLLCTL).

Whether the clock is multiplied by 4 or 8 is selected by the CKDIV0 bit of the CKC register, and PLL is started or stopped by the PLLON bit of the PLLCTL register.

6.3 Control Registers

(1) Processor clock control register (PCC)

The PCC register is a special register. Data can be written to this register only in combination of specific sequences (see **3.4.9 Special registers**).

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

Reset input sets this register to 03H.

(1/2)

After reset: 03H R/W Address: FFFFF828H

	7	<6>	5	<4>	<3>	2	1	0
PCC	FRC	MCK	MFRC	CLS[Note]	CK3	CK2	CK1	CK0

FRC	Use of subclock on-chip feedback resistor
0	Used
1	Not used

MCK	Operation of main clock
0	Operating
1	Stopped

- Even if the MCK bit is set to 1 while the system is operating with the main clock as the CPU clock, the operation of the main clock does not stop. It stops after the CPU clock has been changed to the subclock.
- When the main clock is stopped and the device is operating on the subclock, clear the MCK bit to 0 and wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.

MFRC	Use of main clock on-chip feedback resistor
0	Used
1	Not used

CLS	Status of CPU clock (f_{CPU})
0	Main clock operation
1	Subclock operation

Note The CLS bit is a read-only bit.

(2/2)

CK3	CK2	CK1	CK0	Clock selection (f_{CLK}/f_{CPU})
0	0	0	0	f_{XX}
0	0	0	1	$f_{XX}/2$
0	0	1	0	$f_{XX}/4$
0	0	1	1	$f_{XX}/8$
0	1	0	0	$f_{XX}/16$
0	1	0	1	$f_{XX}/32$
0	1	1	×	Setting prohibited
1	×	×	×	f_{XT}

Cautions 1. Do not change the CPU clock (by using the CK3 to CK0 bits) while CLKOUT is being output.

2. Use a bit manipulation instruction to manipulate the CK3 bit. When using an 8-bit manipulation instruction, do not change the set values of the CK2 to CK0 bits.

Remark ×: don't care

(a) Example of setting main clock operation → subclock operation

<1> CK3 bit ← 1: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<2> Subclock operation: It takes up to the following number of instructions after the CK3 bit is set until the subclock operation is started.

f_{CPU}/f_{XT} of main clock

Therefore, read the CLS bit to check if the subclock operation has started.

<3> MCK bit ← 0: Clear the MCK bit to 0 only when stopping the main clock.

(b) Example of setting subclock operation → main clock operation

<1> MCK bit ← 1: Main clock oscillation starts.

<2> Insert wait cycles by program and wait until the oscillation of the main clock has stabilized.

<3> CK3 bit ← 0: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<4> Main clock operation: It takes up to the following number of instructions after the CK3 bit is set until the main clock operation specified by the CK2 to CK0 bits is started.

Max.: (1/subclock frequency)

Therefore, read the CLS bit to check if the subclock operation has started.

(2) Power save control register (PSC)

The PSC register is a special register. Data can be written to this register only in a combination of specific sequences (see **3.4.9 Special registers**).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF1FEH

	7	<6>	<5>	<4>	3	2	<1>	0
PSC	0	NMI1M	NMI0M	INTM	0	0	STP	0

NMI1M	Standby mode release control upon occurrence of INTWDT2
0	Standby mode release by INTWDT2 enabled
1	Standby mode release by INTWDT2 disabled

NMI0M	Standby mode release control by NMI pin input
0	Standby mode release by NMI pin input enabled
1	Standby mode release by NMI pin input disabled

INTM	Standby mode release control by maskable interrupt request
0	Standby mode release by maskable interrupt request enabled
1	Standby mode release by maskable interrupt request disabled

STP	Standby mode setting
0	Normal mode
1	Standby mode

Cautions 1. When setting the IDLE1, IDLE2 mode (including sub-IDLE mode)/software STOP mode, set the STP bit to 1 after setting the PSM0 and PSM1, bits of the PSMR register.

2. The standby modes controlled by the STP bit include the IDLE1, IDLE2, software STOP, and sub-IDLE modes.

(3) Power save mode register (PSMR)

The PSMR register is an 8-bit register that controls the operation status in the power save mode and the clock operation.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF820H

	7	6	5	4	3	2	<1>	<0>
PSMR	0	0	0	0	0	0	PSM1	PSM0

PSM1	PSM0	Specification of operation in software standby mode
0	0	IDLE1 mode
0	1	Software STOP mode
1	0	IDLE2 mode
1	1	Software STOP mode

Cautions 1. Be sure to clear bits 2 to 7 to 0.

2. The PSM0 and PSM1 bits are valid only when the STP bit of the PSC register is 1.

Remark IDLE1: In this mode, all operations except the oscillator operation and some other circuits (flash memory and PLL) are stopped.

After the IDLE1 mode is released, the normal operation mode is restored without needing to secure the oscillation stabilization time, like the HALT mode.

IDLE2: In this mode, all operations except the oscillator operation are stopped.

After the IDLE2 mode is released, the normal operation mode is restored following the lapse of the setup time specified by the OSTS register (flash memory and PLL).

STOP: In this mode, all operations except the subclock oscillator operation are stopped.

After the software STOP mode is released, the normal operation mode is restored following the lapse of the oscillation stabilization time specified by the OSTS register.

(4) Ring OSC mode register (RCM)

The RCM register is an 8-bit register that sets the operation mode of Ring OSC.

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

Reset input clears this register to 00H.

After reset: 01H R/W After reset: FFFFF806H

	7	6	5	4	3	2	1	<0>
RCM	0	0	0	0	0	0	0	RSTOP

RSTOP	Operation/stop of Ring-OSC
0	Ring-OSC operating
1	Ring-OSC stopped

(5) CPU operation clock status register (CCLS)

The CCLS register indicates the status of the CPU operation clock.

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

Reset input clears this register to 00H.

After reset: 00H R Address: FFFFF82EH

	7	6	5	4	3	2	1	0
CCLS	0	0	0	0	0	0	0	CCLSF

CCLSF	CPU operation clock status
0	Operating on main clock (f_X) or subclock (f_{XT}).
1	Operating on Ring-OSC (f_{XR}).

(6) Oscillation stabilization time select register (OSTS)

The OSTS register selects the oscillation stabilization time following reset or release of the STOP mode.

See **12.3 (1) Oscillation stabilization time select register (OSTS)**.

6.4 Operation

6.4.1 Operation of each clock

The following table shows the operation status of each clock.

Table 6-1. Operation Status of Each Clock

	PLL Register								
	CLK Bit = 0, MCK Bit = 0					CLS Bit = 1, MCK Bit = 0		CLS Bit = 1, MCK Bit = 1	
	<1>	<2>	<3>	<4>	<5>	<6>	<7>	<6>	<7>
Main clock oscillator (f_X)	×	O	O	O	×	O	O	×	×
Subclock oscillator (f_{XT})	O	O	O	O	O	O	O	O	O
CPU clock (f_{CPU})	×	×	×	×	×	O	×	O	×
Internal system clock (f_{CLK})	×	×	O	×	×	O	×	O	×
Peripheral clock (f_{XX} to f_{XX}/1,024)	×	×	O	×	×	O	×	×	×
WT clock (main)	×	O	O	O	×	O	O	×	×
WT clock (sub)	O	O	O	O	O	O	O	O	O
WDT2 clock (ring)	×	O	O	O	O	O	O	O	O
WDT2 clock (main)	×	×	O	×	×	O	×	×	×
WDT2 clock (sub)	O	O	O	O	O	O	O	O	O

Remark CLS bit: Bit 4 of the processor clock control register (PCC)
MCK bit: Bit 6 of the PCC register
O: Operable
×: Stopped
<1>: $\overline{\text{RESET}}$ pin input
<2>: During oscillation stabilization time count
<3>: HALT mode
<4>: IDLE1, IDLE2 mode
<5>: Software STOP mode
<6>: Subclock operation mode
<7>: Sub-IDLE mode

6.4.2 Clock output function

The clock output function is used to output the internal system clock (f_{CLK}) from the CLKOUT pin.

The internal system clock (f_{CLK}) is selected by using the CK3 to CK0 bits of the processor clock control register (PCC).

The CLKOUT pin functions alternately as the PCM1 pin and functions as a clock output pin if so specified by the control register of port CM.

The status of the CLKOUT pin is the same as the internal system clock in Table 6-1 and the pin can output the clock when it is in the operable status. It outputs a low level in the stopped status. However, the alternate-function pin (PCM1: input mode) is selected in <1> and <2> after the $\overline{\text{RESET}}$ signal has been input. Consequently, the CLKOUT pin goes into a high-impedance state.

6.5 PLL Function

6.5.1 Overview

The PLL function is used to output the operating clock of the CPU and peripheral macro at a frequency 4 or 8 times higher than the oscillation frequency, and select the clock-through mode.

When PLL function is used: Input clock = 2.5 to 5 MHz (output: 10 to 20 MHz)
Clock-through mode: Input clock = 2.5 to 10 MHz (output: 2.5 to 10 MHz)

6.5.2 Control registers

(1) PLL control register (PLLCTL)

The PLLCTL register is an 8-bit register that controls the PLL function.
This register can be read or written in 8-bit or 1-bit units.
Reset input sets this register to 01H.

After reset: 01H R/W Address: FFFFF82CH

	7	6	5	4	3	2	<1>	<0>
PLLCTL	0	0	0	0	0	0	SELPLL	PLLON

PLLON	PLL operation stop register
0	PLL stopped
1	PLL operating (After PLL operation starts, a lockup time is required for frequency stabilization)

SELPLL	CPU operation clock selection register
0	Clock-through mode
1	PLL mode

Cautions 1. When the PLLON bit is cleared to 0, the SELPLL bit is automatically cleared to 0 (clock-through mode).

2. The SELPLL bit can be set to 1 only when the PLL clock frequency is stabilized. If not (unlocked), "0" is written to the SELOLL bit if data is written to it.

(2) Clock control register (CKC)

The CKC register controls the internal system clock in the PLL mode.

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

Reset input sets this register to 0AH.

After reset: 0AH R/W Address: FFFFF822H

	7	6	5	4	3	2	1	0
CKC	0	0	0	0	1	0	1	CKDIV0

CKDIV0	Internal system clock (f_{XX}) in PLL mode
0	$f_{XX} = 4 \times f_X$ (f_X = 2.5 to 5.0 MHz)
1	$f_{XX} = 8 \times f_X$ (f_X = 2.5 MHz)

Caution The PLL mode cannot be used at f_X = 5.0 to 10.0 MHz.

Remark Both the CPU clock and peripheral clock are divided by the CKC register, but only the CPU clock is divided by the PCC register.

(3) Lock register (LOCKR)

Phase lock occurs at a given frequency following power application or immediately after the software STOP mode is released, and the time required for stabilization is the lockup time (frequency stabilization time). This time until stabilization is called the lockup status, and the stabilized state is called the locked status.

The lock register (LOCKR) includes a LOCK bit that reflects the PLL frequency stabilization status.

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

Reset input clears this register to 00H.

After reset: 00H R Address: FFFFF824H

	7	6	5	4	3	2	1	<0>
LOCKR	0	0	0	0	0	0	0	LOCK

LOCK	PLL lock status check
0	Locked status
1	Unlocked status

Caution The LOCK register does not reflect the lock status of the PLL in real time. The set/reset conditions are as follows.

[Set conditions]

- Upon system reset[Note]
- In IDLE2 or software STOP mode
- Upon setting of PLL stop (clearing of PLLON bit of PLLCTL register to 0)
- Upon stopping main clock and using CPU with subclock (setting of CK3 bit of PCC register to 1 and setting of MCK bit of same register to 1)

Note This register is set to 01H by reset and cleared to 00H after the reset has been released and the oscillation stabilization time has elapsed.

[Reset conditions]

- Upon overflow of oscillation stabilization time following reset release (OSTS register default time (see **12.3 (1) Oscillation stabilization time select register (OSTS)**))
- Upon oscillation stabilization timer overflow (time set by OSTS register) following software STOP mode release, when the software STOP mode was set in the PLL operating status
- Upon PLL lockup time timer overflow (time set by PLLS register) when the PLLON bit of the PLLCTL register is changed from 0 to 1

(4) PLL lockup time specification register (PLLS)

The PLLS register is an 8-bit register used to select the PLL lockup time when the PLLON bit of the PLLCTL register is changed from 0 to 1.

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

Reset input sets this register to 03H.

After reset: 03H R/W Address: FFFFF6C1H

	7	6	5	4	3	2	1	0
PLLS	0	0	0	0	0	0	PLLS1	PLLS0

PLLS1	PLLS0	Selection of PLL lockup time
0	0	$2^{10}/f_X$
0	1	$2^{11}f_X$
1	0	$2^{12}/f_X$
1	1	$2^{13}/f_X$ (default value)

Caution Set so that the interval time is 800 μs or longer.

6.5.3 Usage

(1) To use PLL

- After the $\overline{\text{RESET}}$ signal has been released, the PLL operates (PLLON bit = 1), but because the default mode is the clock-through mode (SELPLL bit = 0), select the PLL mode (SELPLL bit = 1).
- To set the IDLE1 or IDLE2 mode, first select the clock-through mode and then stop the PLL. To return from the IDLE1 or IDLE2 mode, first enable PLL operation (PLLON bit = 1), and then select the PLL mode (SELPLL bit = 1).
- To enable PLL operation, first set the PLLON bit to 1, and then set the SELPLL bit to 1 after the LOCK bit = 0. To stop the PLL, first select the clock-through mode (SELPLL bit = 0), wait for 8 clocks or more, and then stop the PLL (PLLON bit = 0).

(2) When PLL is not used

- The clock-through mode (SELPLL bit = 0) is selected after the $\overline{\text{RESET}}$ signal has been released, but the PLL is operating (PLLON bit = 1) and must therefore be stopped (PLLON bit = 0).

CHAPTER 7 16-BIT TIMER/EVENT COUNTER P

7.1 Features

Timer P (TMP) is a 16-bit timer/event counter that can be used in various ways.
TMP can perform the following operations.

- PWM output
- Interval timer
- External event counter (operation not possible when clock is stopped)
- One-shot pulse output
- Pulse width measurement

7.2 Function Outline

- Capture trigger input signal × 2
- External trigger input signal × 1
- Clock select × 8
- External event count input × 1
- Readable counter × 1
- Capture/compare reload register × 2
- Capture/compare match interrupt × 2
- Timer output (TOPn0, TOPn1) × 2

7.3 Configuration

TMP includes the following hardware.

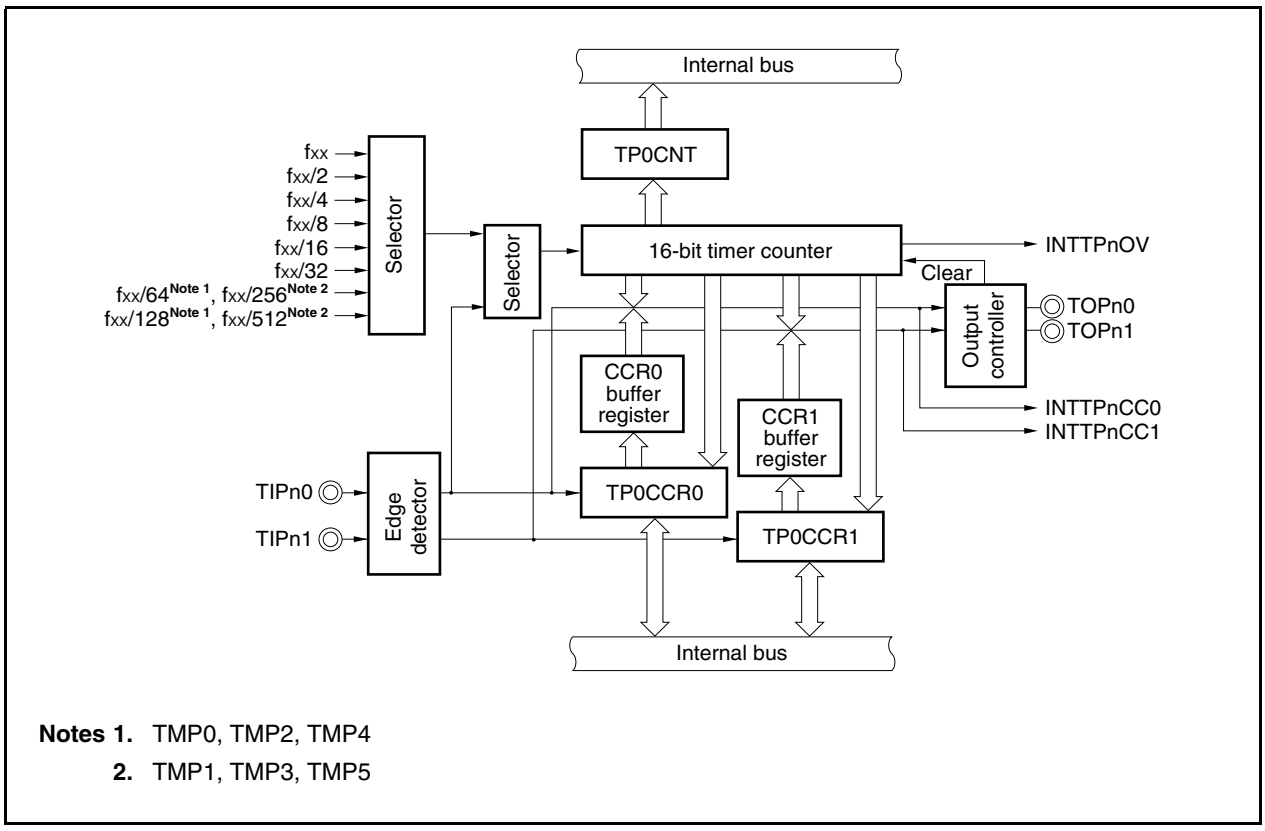
Table 7-1. Configuration of TMP0 to TMP5

Item	Configuration
Timer register	16-bit counter
Registers	TMPn capture/compare registers 0, 1 (TPnCCR0, TPnCCR1) TMPn counter register (TPnCNT) CCR0 buffer register, CCR1 buffer register
Timer input	2 × 4 (TIPn0[Note], TIPn1)
Timer output	2 × 4 (TOPn0, TOPn1)
Control registers	TMPn control registers 0, 1 (TPnCTL0, TPnCTL1) TMPn I/O control registers 0 to 2 (TPnIOC0 to TPnIOC2) TMPn option registers 0, 1 (TPnOPT0, TPnOPT1)

Note TIPn0 is multiplexed with a capture trigger input signal, external trigger input signal, and external event input signal.

Remark n = 0 to 5

Figure 7-1. Block Diagram of Timer P



Notes 1. TMP0, TMP2, TMP4

2. TMP1, TMP3, TMP5

(1) TMPn capture/compare register 0 (TPnCCR0)

The TPnCCR0 register is a 16-bit register that functions both as a capture register and as a compare register.

Whether this register functions as a capture register or as a compare register can be controlled with the TPnCCSS0 bit of the TPnOPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TPnCCR0 register is a compare register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: TP0CCR0 FFFFF596H, TP1CCR0 FFFFF5A6H,
TP2CCR0 FFFFF5B6H, TP3CCR0 FFFFF5C6H,
TP4CCR0 FFFFF5D6H, TP5CCR0 FFFFF5E6H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPnCCR0																

(n = 0 to 5)

- Use as a compare register

  TPnCCR0 can be rewritten when TPnCE = 1

  The rewrite method is as follows.

TMP Operation Mode	Method of Writing TPnCCR0 Register
PWM output mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse output mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as capture register

  The counter value is saved to TPnCCR0 upon capture trigger (TIPn0) input edge detection.

(2) TMPn capture/compare register 1 (TPnCCR1)

The TPnCCR1 register is a 16-bit register that functions both as a capture register and as a compare register.

Whether this register functions as a capture register or as a compare register can be controlled with the TPnCCS1 bit of the TPnOPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

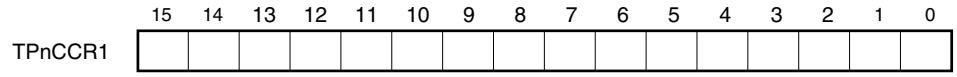
In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TPnCCR1 register is a reload register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: TP0CCR1 FFFFF598H, TP1CCR1 FFFFF5A8H, TP2CCR1 FFFFF5B8H, TP3CCR1 FFFFF5C8H, TP4CCR1 FFFFF5D8H, TP5CCR1 FFFFF5E8H



TPnCCR1

(n = 0 to 5)

- Use as a compare register

  TPnCCR1 can be rewritten when TPnCE = 1

  The timing at which the TPnCCR1 rewrite values become valid when TPnCE = 1 is as follows.

TMP Operation Mode	Method of Writing TPnCCR1 Register
PWM output mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse output mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as a capture register

  The counter value is saved to TPnCCR1 upon capture trigger (TIPn1) input edge detection.

(3) TMPn counter register (TPnCNT)

The TPnCNT register is a read buffer register that can read 16-bit counter values.

This register is read-only, in 16-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R Address: TP0CNT FFFFF59AH, TP1CNT FFFFF5AAH,
TP2CNT FFFFF5BAH, TP3CNT FFFFF5CAH,
TP4CNT FFFFF5DAH, TP5CNT FFFFF5EAH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPnCNT																

(n = 0 to 5)

7.4 Control Registers

(1) TMPn control register 0 (TPnCTL0)

The TPnCTL0 register is an 8-bit register that controls the operation of timer P.

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

Reset input clears this register to 00H.

The same value can always be written to the TPnCTL0 register by software.

After reset: 00H R/W Address: TP0CTL0 FFFFF590H, TP1CTL0 FFFFF5A0H, TP2CTL0 FFFFF5B0H, TP3CTL0 FFFFF5C0H, TP4CTL0 FFFFF5D0H, TP5CTL0 FFFFF5E0H

TPnCTL0 (n = 0 to 5)

<7>	6	5	4	3	2	1	0
TPnCE	0	0	0	0	TPnCKS2	TPnCKS1	TPnCKS0

TPnCE	Timer Pn operation control
0	Internal operating clock operation disabled (TMPn reset asynchronously)
1	Internal operating clock operation enabled

Internal operating clock control and TMPn asynchronous reset are performed with the TPnCE bit. When the TPnCE bit is cleared to 0, the internal operating clock of TMPn stops (fixed to low level) and TMPn is reset asynchronously.
When the TPnCE bit is set to 1, the internal operating clock is enabled and count-up operation starts within 2 input clocks after the TPnCE bit was set to 1.

TPnCKS2	TPnCKS1	TPnCKS0	Internal count clock selection	
			n = 0, 2, 4	n = 1, 3, 5
0	0	0	f_{XX}	
0	0	1	$f_{XX}/2$	
0	1	0	$f_{XX}/4$	
0	1	1	$f_{XX}/8$	
1	0	0	$f_{XX}/16$	
1	0	1	$f_{XX}/32$	
1	1	0	$f_{XX}/64$	$f_{XX}/256$
1	1	1	$f_{XX}/128$	$f_{XX}/512$

Caution Set the TPnCKS2 to TPnCKS0 bits when TPnCE = 0. When the value of the TPnCE bit is changed from 0 to 1, the TPnCKS2 to TPnCKS0 bits can be set simultaneously.

(2) TMPn control register 1 (TPnCTL1)

The TPnCTL1 register is an 8-bit register that controls the operation of timer P.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: TP0CTL1 FFFFF591H, TP1CTL1 FFFFF5A1H, TP2CTL1 FFFFF5B1H, TP3CTL1 FFFFF5C1H, TP4CTL1 FFFFF5D1H, TP5CTL1 FFFFF5E1H

	7	<6>	<5>	4	3	2	1	0
TPnCTL1	0	TPnEST	TPnEEE	0	0	TPnMD2	TPnMD1	TPnMD0

(n = 0 to 5)

TPnEST	Software trigger control
0	No operation
1	In one-shot pulse mode: One-shot pulse software trigger
	In external trigger pulse output mode: Pulse output software trigger
The TPnEST bit functions as a software trigger in the one-shot pulse mode and the external trigger pulse output mode. (This bit is invalid even if it is controlled in any other mode.) TPnEST functions as a software trigger if it is set to 1 when TPnCE = 1. Therefore, be sure to set TPnEST to 1 when TPnCE = 1. The TIPn0 pin is used for an external trigger. The read value of the TPnEST bit is always 0.	

TPnEEE	Count clock selection
0	Use the internal clock (clock selected with bits TPnCKS2 to TPnCKS0)
1	Use external clock (TIPn0 input edge)
The valid edge when TPnEET = 1 (external clock: TIPn0) is specified with bits TPnEES1 and TPnEES0.	

TPnMD2	TPnMD1	TPnMD0	Timer mode selection
0	0	0	Interval timer mode
0	0	1	External event count mode
0	1	0	External trigger pulse output mode
0	1	1	One-shot pulse mode
1	0	0	PWM mode
1	0	1	Free-running mode
1	1	0	Pulse width measurement mode
1	1	1	Setting prohibited

Caution Set the TPnEEE and TPnMD2 to TPnMD0 bits when TPnCE = 0. (The same value can be written when TPnCE = 1.) The operation is not guaranteed when rewriting is performed when TPnCE = 1. If rewriting was mistakenly performed, set TPnCE = 0 and then set the bits again.

(3) TMPn I/O control register 0 (TPnIOC0)

The TPnIOC0 register is an 8-bit register that controls the timer output (TOPn0, TOPn1).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: TP0IOC0 FFFFF592H, TP1IOC0 FFFFF5A2H, TP2IOC0 FFFFF5B2H, TP3IOC0 FFFFF5C2H, TP4IOC0 FFFFF5D2H, TP5IOC0 FFFFF5E2H

TPnIOC0 (n = 0 to 5)

7	6	5	4	<3>	<2>	<1>	<0>
0	0	0	0	TPnOL1	TPnOE1	TPnOL0	TPnOE0

TPnOL1	TOPn1 output level setting
0	Normal output
1	Inverted output

TPnOE1	TOPn1 output setting
0	Timer output prohibited (low level and high level are output from the TOPn1 pin when TPnOL1 = 0 and TPnOL1 = 1, respectively).
1	Timer output enabled (A pulse is output from the TOPn1 pin.)

TPnOL0	TOPn0 output level setting
0	Normal output
1	Inverted output

TPnOE0	TOPn0 output setting
0	Timer output prohibited (low level and high level are output from the TOPn0 pin when TPnOL0 = 0 and TPnOL0 = 1, respectively).
1	Timer output enabled (A pulse is output from the TOPn0 pin.)

Caution Rewrite the TPnOL1, TPnOE1, TPnOl0, and TPnOE0 bits when TPnCE = 0. (The same value can be written when TPnCE = 1.) If rewriting was mistakenly performed, set TPnCE = 0 and then set the bits again.

(4) TMPn I/O control register 1 (TPnIOC1)

The TPnIOC1 register is an 8-bit register that controls the valid edge for the external input signals (TIPn0, TIPn1).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: TP0IOC1 FFFFF593H, TP1IOC1 FFFFF5A3H, TP2IOC1 FFFFF5B3H, TP3IOC1 FFFFF5C3H, TP4IOC1 FFFFF5D3H, TP5IOC1 FFFFF5E3H

	7	6	5	4	3	2	1	0
TPnIOC1	0	0	0	0	TPnIS3	TPnIS2	TPnIS1	TPnIS0

(n = 0 to 5)

TPnIS3	TPnIS2	Capture input (TIPn1) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TPnIS1	TPnIS0	Capture input (TIPn0) valid edge detection
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TPnIS3 to TPnIS0 bits when TPnCE = 0. (The same value can be written when TPnCE = 1.) If rewriting was mistakenly performed, set TPnCE = 0 and then set the bits again.

2. The TPnIS3 to TPnIS0 bits are valid only in the free-running mode and the pulse width measurement mode. In all other modes, a capture operation is not possible.

(5) TMPn I/O control register 2 (TPnIOC2)

The TPnIOC2 register is an 8-bit register that controls the valid edge of the external event count input signal (TIPn0) and external trigger input signal (TIPn0).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: TP0IOC2 FFFFF594H, TP1IOC2 FFFFF5A4H, TP2IOC2 FFFFF5B4H, TP3IOC2 FFFFF5C4H, TP4IOC2 FFFFF5D4H, TP5IOC2 FFFFF5E4H

	7	6	5	4	3	2	1	0
TPnIOC2	0	0	0	0	TPnEES1	TPnEES0	TPnETS1	TPnETS0

(n = 0 to 5)

TPnEES1	TPnEES0	External event count input valid edge setting
0	0	No edge detection (external event count invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TPnETS1	TPnETS0	External trigger input valid edge detection
0	0	No edge detection (external trigger invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TPnEES1, TPnEES0, TPnEST1, and TPnEST0 bits when TPnCE = 0. (The same value can be written when TPnCE = 1.) If rewriting was mistakenly performed, set TPnCE = 0 and then set the bits again.

2. The TPnEES1 and TPnEES0 bits are valid only when TPnEEE = 1 or when the external event count mode (TPnMD2 to TPnMD0 = 001 of the TIPnCTL1 register) has been set.

(6) TMPn option register 0 (TPnOPT0)

The TPnOPT0 register is an 8-bit register used to set the capture/compare operation and detect overflow.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: TP0OPT0 FFFFF595H, TP1OPT0 FFFFF5A5H, TP2OPT0 FFFFF5B5H, TP3OPT0 FFFFF5C5H, TP4OPT0 FFFFF5D5H, TP5OPT0 FFFFF5E5H

	7	6	<5>	<4>	3	2	1	<0>
TPnOPT0 (n = 0 to 5)	0	0	TPnCCS1	TPnCCS0	0	0	0	TPnOVF

TPnCCS1	TPnCCR1 register capture/compare selection
0	Compare register selection
1	Capture register selection
The TPnCCS1 bit setting is valid only in the free-running mode.	

TPnCCS0	TPnCCR0 register capture/compare selection
0	Compare register selection
1	Capture register selection
The TPnCCS0 bit setting is valid only in the free-running mode.	

TPnOVF	Timer P overflow detection
Set (1)	Overflow occurrence
Reset (0)	TPnOVF bit 0 write or TPnCE = 0
• The TPnOVF bit is reset when the 16-bit counter value overflows from FFFFH to 0000H in the free-running mode or the pulse width measurement mode. • An interrupt request signal (INTTPnOV) is generated at the same time that the TPnOVF bit is set (1). The INTTPnOV signal is not generated in modes other than the free-running mode and the pulse width measurement mode. • The TPnOVF bit is not cleared even when the TPnOVF bit and the TPnOPT0 register are read when TPnOVF = 1. • The TPnOVF bit can be both read and written, but 1 cannot be written to the TPnOVF bit from the CPU. Writing 1 has no influence on the operation of timer P.	

Caution Rewrite the TPnCCS1 and TPnCCS0 bits when TPnCE = 0. (The same value can be written when TPnCE = 1.) If rewriting was mistakenly performed, set TPnCE = 0 and then set the bits again.

7.5 Operation

Timer P can perform the following operations.

Operation	TPnEST (Software Trigger Bit)	TIPn0 (External Trigger Input)	Capture/Compare Write	Compare Write
Interval timer mode	Invalid	Invalid	Compare only	Anytime write
External event count mode[Note 1]	Invalid	Invalid	Compare only	Anytime write
External trigger pulse output mode[Note 2]	Valid	Valid	Compare only	Reload
One-shot pulse output mode[Note 2]	Valid	Valid	Compare only	Anytime write
PWM mode	Invalid	Invalid	Compare only	Reload
Free-running mode	Invalid	Invalid	Capture/compare switching enabled	Anytime write
Pulse width measurement mode[Note 2]	Invalid	Invalid	Capture only	Not applicable

Notes 1. To use the external event count function, specify that the edge of the TIP00 capture input is not detected (by clearing the TP0IS1 and TP0IS0 bits of the TP0IOC1 register to “00”).

2. When using the external trigger pulse output mode, one-shot pulse mode, and pulse width measurement mode, select a count clock (by clearing the TPnEEE bit of the TPnCTL1 register to 0).

Remark n = 0 to 5

7.5.1 Anytime write and reload

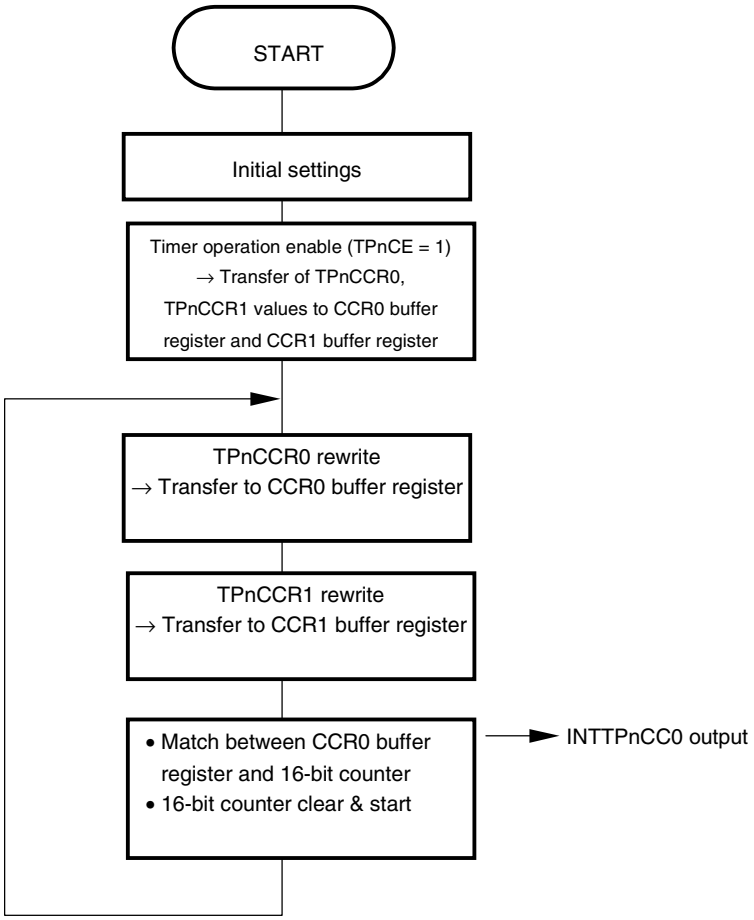
TPnCCR0 and TPnCCR1 register rewrite is possible for timer P during timer operation (TPnCE = 1), but the write method (anytime write, reload) differs depending on the mode.

(1) Anytime write

When the TPnCCRm register is written during timer operation, the write data is transferred at that time to the CCRm buffer register and used as the 16-bit counter comparison value.

Remark n = 0 to 5
m = 0, 1

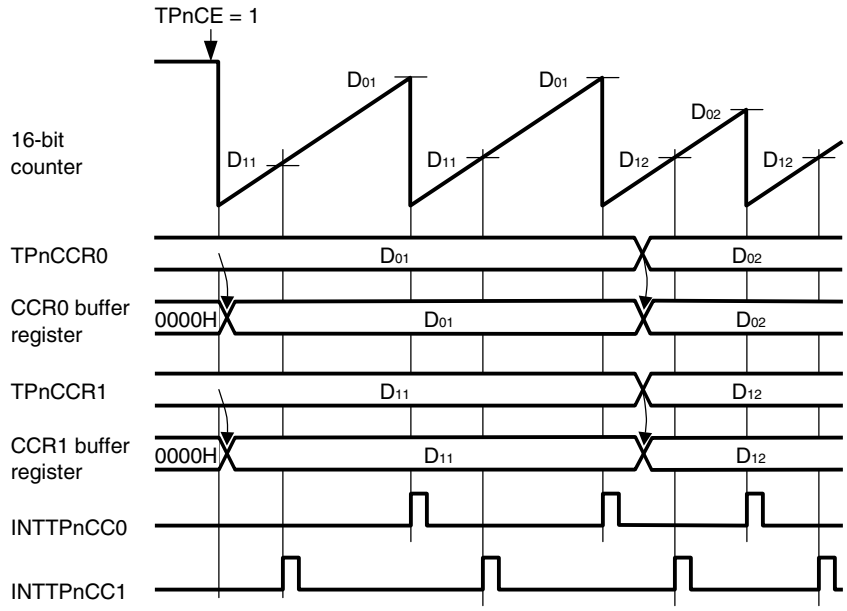
Figure 7-2. Flowchart of Basic Operation for Anytime Write



Remarks 1. The above flowchart illustrates an example of the operation in the interval timer mode.

2. n = 0 to 5

Figure 7-3. Timing Diagram for Anytime Write



Remarks 1. D_{01}, D_{02}: Setting values of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TPnCCR1 register (0000H to FFFFH)

2. The above timing chart illustrates an example of the operation in the interval timer mode.

3. n = 0 to 5

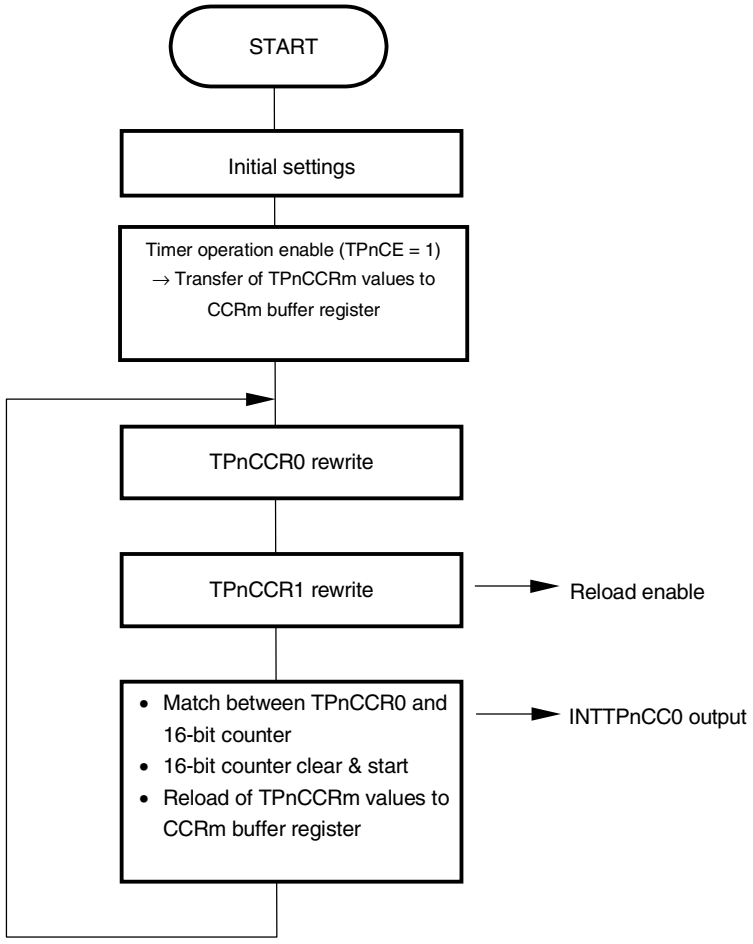
(2) Reload

When the TPnCCR0 and TPnCCR1 registers is written during timer operation via the CCRm buffer register, the write data is used as the 16-bit counter comparison value. The TPnCCR0 register and the TPnCCR1 register can be rewritten when TPnCE = 1.

In order for the setting value when the TPnCCR0 register and the TPnCCR1 register are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to the CCRm buffer register), it is necessary to rewrite TPnCCR0 and then write to the TPnCCR1 register before the 16-bit counter value and the TPnCCR0 register value match. Thereafter, the values of the TPnCCR0 and the TPnCCR1 register are reloaded upon TPnCCR0 register match.

Whether to enable or disable the next reload timing is controlled by writing to the TPnCCR1 register. Thus even when wishing only to rewrite the value of the TPnCCR0 register, also write the same value to the TPnCCR1 register.

Figure 7-4. Flowchart of Basic Operation for Reload

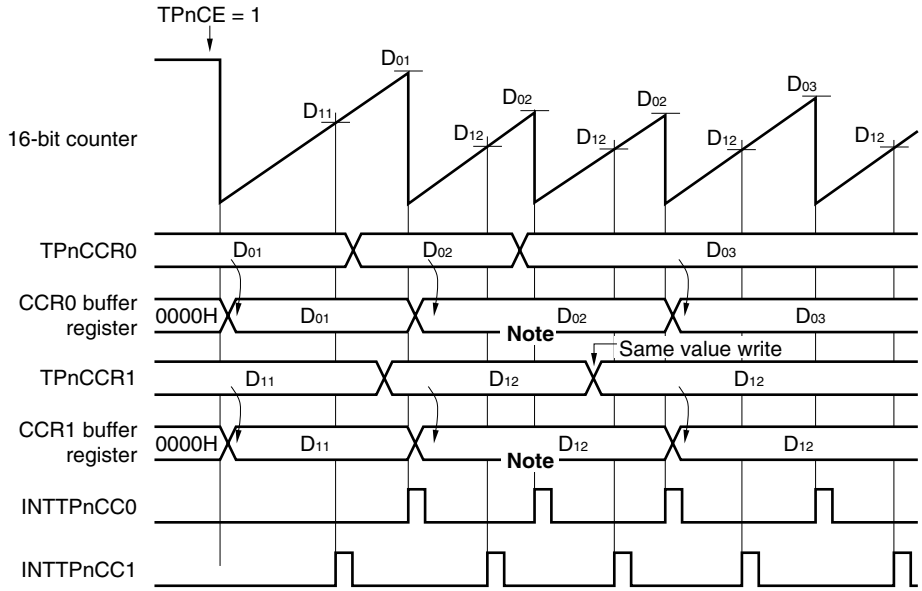


Caution Writing to the TPnCCR1 register includes enabling of reload. Thus, rewrite the TPnCCR1 register after rewriting the TPnCCR0 register.

Remarks 1. The above flowchart illustrates an example of the operation in the PWM mode.

2. n = 0 to 5, m = 0, 1

Figure 7-5. Timing Chart for Reload



Note Reload is not performed because the TPnCCR1 register was not rewritten.

Remarks 1. D_{01}, D_{02}, D_{03}: Setting value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting value of TPnCCR1 register (0000H to FFFFH)

2. The above flowchart illustrates the operation in the PWM mode as an example.

3. n = 0 to 5

7.5.2 Interval timer mode (TPnMD2 to TPnMD0 = 000)

In the interval timer mode, an interrupt request signal (INTTPnCC0) is output upon a match between the setting value of the TPnCCR0 register and the value of the 16-bit counter, and the 16-bit counter is cleared. The TPnCCR0 register can be rewritten when TPnCE = 1, and when a value is set to the TPnCCR0 register with a write instruction from the CPU, it is transferred to the CCR0 buffer register through anytime write, and is used as the value for comparison with the 16-bit counter value.

In the interval timer mode, the 16-bit counter is cleared only upon a match between the value of the 16-bit counter and the value of the CCR0 buffer register.

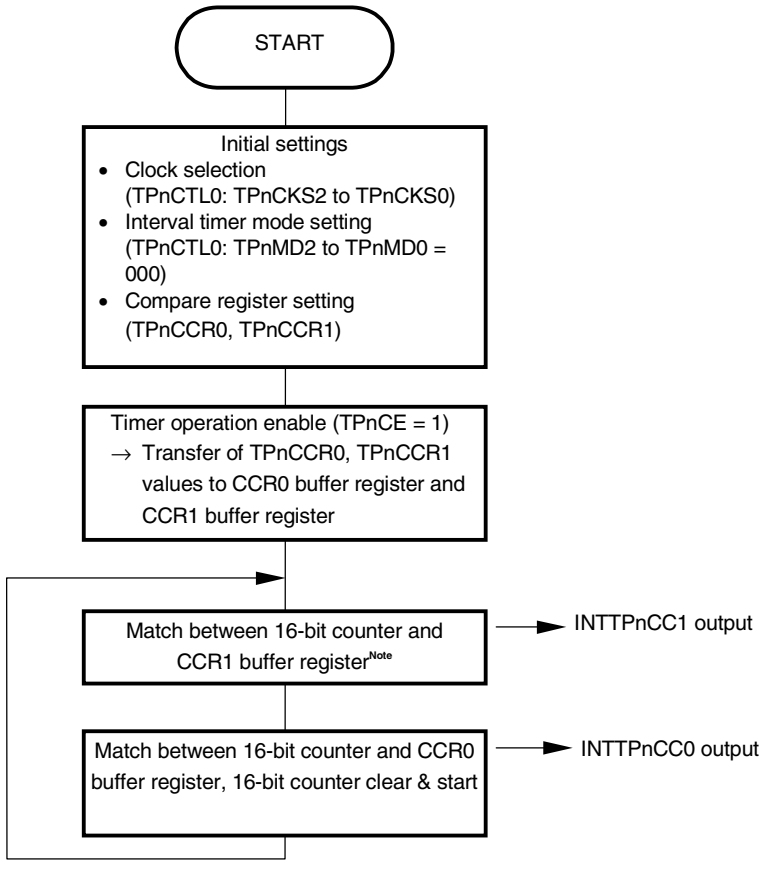
16-bit counter clearing using the TPnCCR1 register is not performed. However, the setting value of the TPnCCR1 register is transferred to the CCR1 buffer register and compared with the value of the 16-bit counter, and an interrupt request (INTTPnCC1) is output if these values match.

Moreover, TOPnm pin output is also possible by setting the TPnOEm bit to 1.

When the TPnCCR1 register is not used, it is recommended to set FFFFH as the setting value for the TPnCCR1 register.

Remark n = 0 to 5,
m = 0, 1

Figure 7-6. Flowchart of Basic Operation in Interval Timer Mode

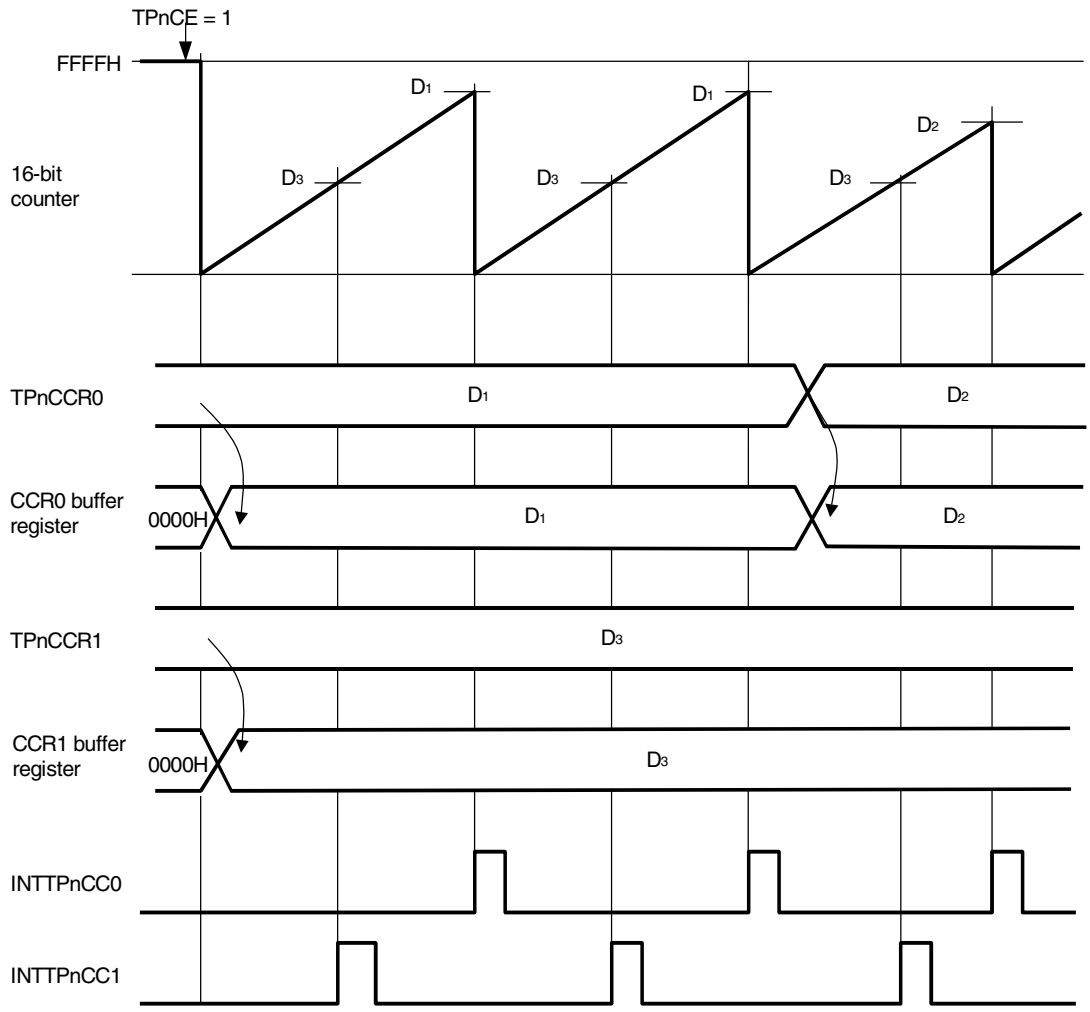


Note The 16-bit counter is not cleared upon a match between the 16-bit counter and TPnCCR1.

Remark n = 0 to 5

Figure 7-7. Basic Operation Timing in Interval Timer Mode (1/2)

(a) $D_1 > D_2 > D_3$; rewrite of TPnCCR0 register only; no TOPn0, TOPn1 output



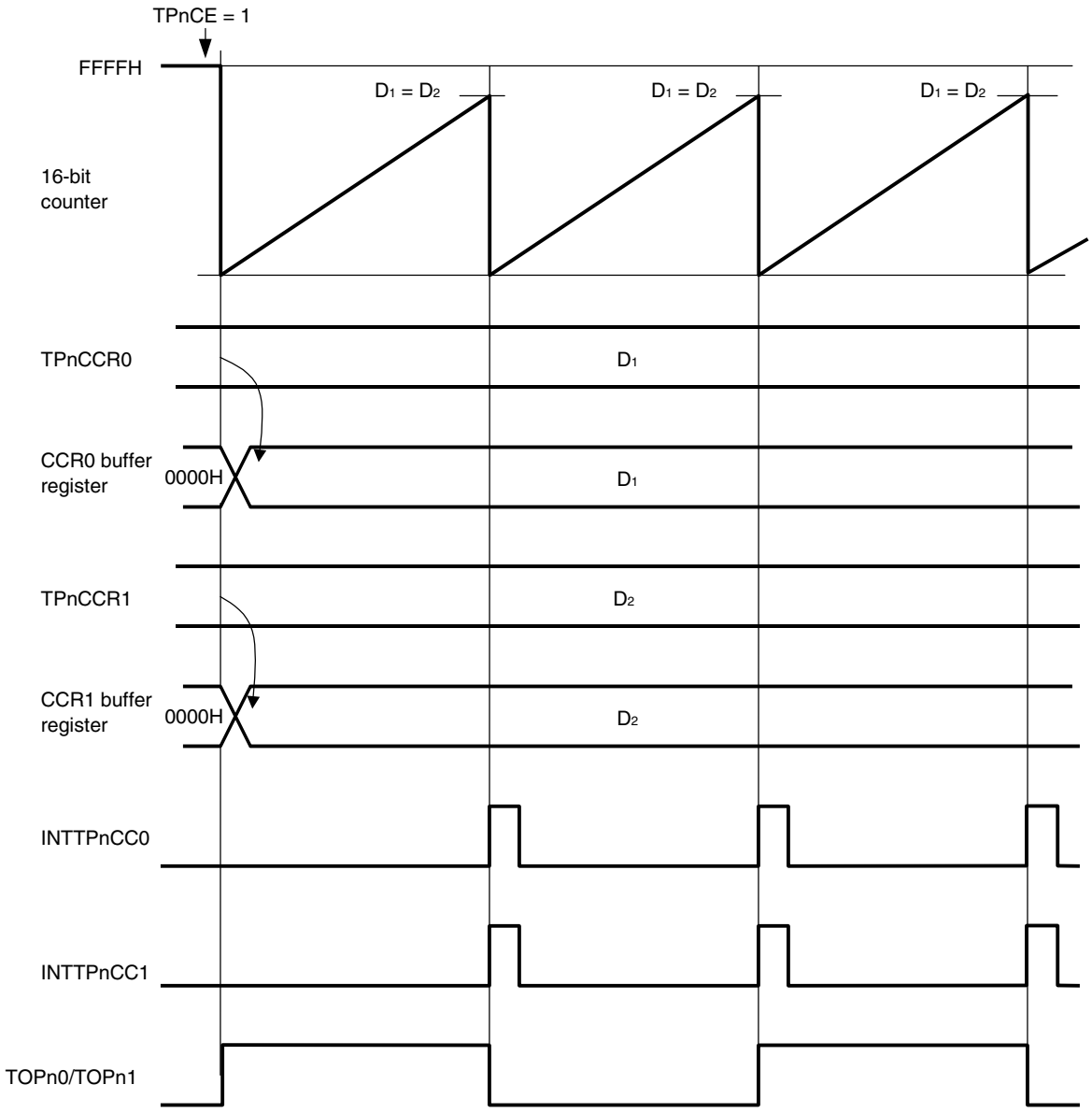
Remarks 1. D_1, D_2: Setting values of TPnCCR0 register (0000H to FFFFH)
D_3: Setting value of TPnCCR1 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 5

Figure 7-7. Basic Operation Timing in Interval Timer Mode (2/2)

(b) $D_1 = D_2$; no TPnCCR0, TPnCCR1 rewrite; TOPn1 output



Remarks 1. D_1: Setting value of TPnCCR0 register (0000H to FFFFH)
D_2: Setting value of TPnCCR1 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 5

7.5.3 External event count mode (TPnMD2 to TPnMD0 = 001)

In the external event count mode, external event count input (TIPn0 pin input) is used as a count-up signal. When the external event count mode is set, count-up is performed using external event count input (TIPn0 pin input), regardless of the setting of the TPnEEE bit of the TPnCTL0 register.

In the external event count mode, a match interrupt request (INTTPnCC0) is output upon a match between the setting value of the TPnCCR0 register and the value of the 16-bit counter, and the 16-bit counter is cleared.

When a value is set to the TPnCCR0 register with a write instruction from the CPU, it is transferred to the CCR0 buffer register through anytime write, and is used as the value for comparison with the 16-bit counter value.

In the external event count mode, the 16-bit counter is cleared only upon a match between the value of the 16-bit counter and the value of the CCR0 buffer register.

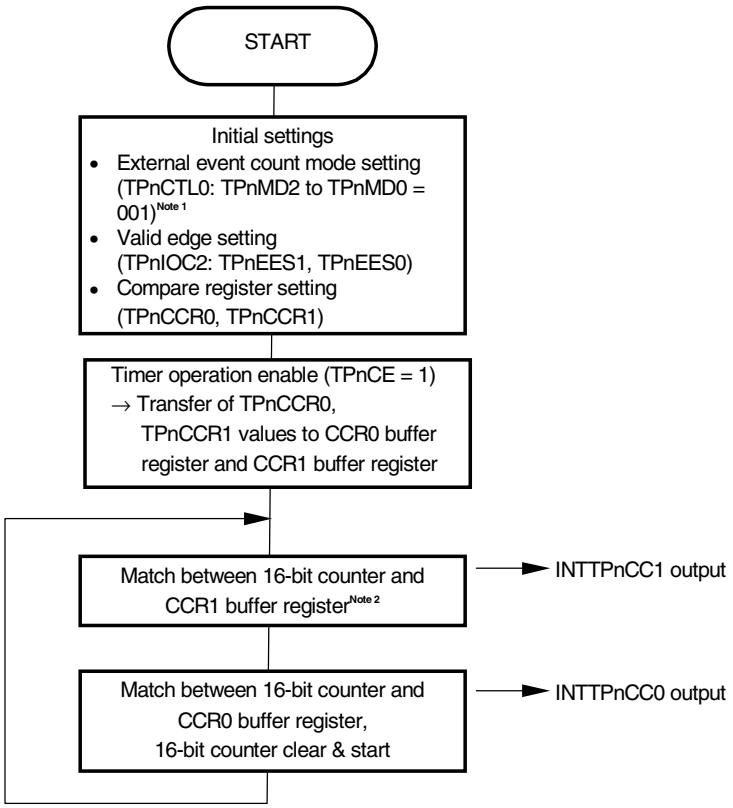
16-bit counter clearing using the TPnCCR1 register is not performed. However, the setting value of the TPnCCR1 register is transferred to the CCR1 buffer register and compared with the value of the 16-bit counter, and an interrupt request (INTTPnCC1) is output if these values match.

Moreover, TOPnm pin output is also possible by setting the TPnOEm bit to 1.

The TPnCCR0 register can be rewritten when TPnCE = 1. When the TPnCCR1 register is not used, it is recommended to set FFFFH as the setting value for the TPnCCR1 register.

Remark n = 0 to 5
m = 0, 1

Figure 7-8. Flowchart of Basic Operation in External Event Count Mode



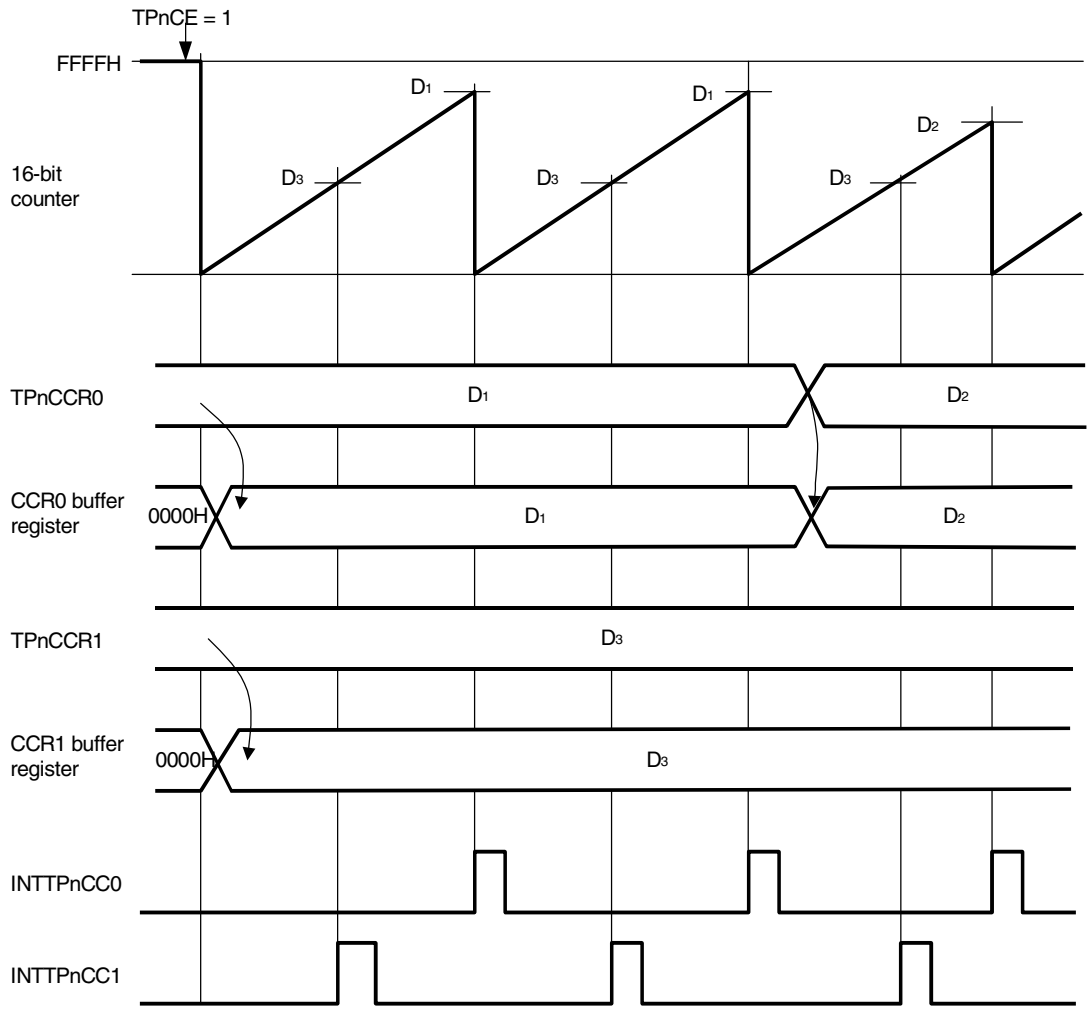
Notes 1. Selection of the TPnEEE bit has no influence.

2. The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCR1 buffer register.

Remark n = 0 to 5

Figure 7-9. Basic Operation Timing in External Event Count Mode (1/2)

(a) $D_1 > D_2 > D_3$; rewrite of TPnCCR0 only; no TOPn0, TOPn1 output



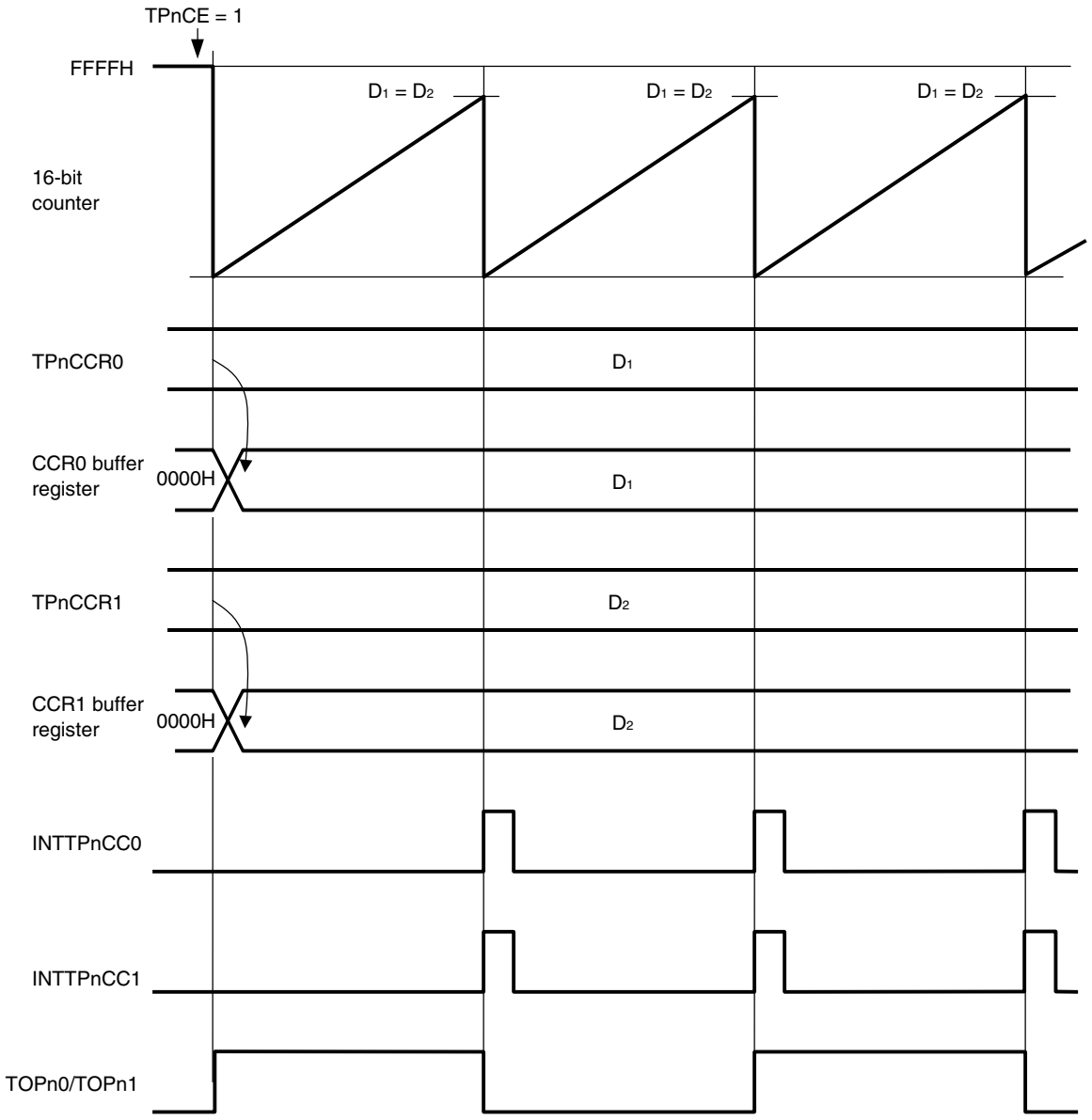
Remarks 1. D_1, D_2: Setting values of TPnCCR0 register (0000H to FFFFH)
D_3: Setting value of TPnCCR1 register (0000H to FFFFH)

2. Event count = (Dn + 1)

3. n = 0 to 5

Figure 7-9. Basic Operation Timing in External Event Count Mode (2/2)

(b) $D_1 = D_2$; no TPnCCR0, TPnCCR1 rewrite; TOPn0, TOPn1 output



Remarks 1. D_1: Setting value of TPnCCR0 register (0000H to FFFFH)
D_2: Setting value of TPnCCR1 register (0000H to FFFFH)

2. Event count = (Dn + 1)

3. n = 0 to 5

7.5.4 External trigger pulse mode (TPnMD2 to TPnMD0 = 010)

In the external trigger pulse mode, setting TPnCE = 1 causes external trigger input (TIPn0 pin input) wait with the 16-bit counter stopped at FFFFH. The count-up operation starts upon detection of the external trigger input (TIPn0 pin input) edge.

Regarding TOPn1 output control, the reload register (TPnCCR1) is used as the duty setting register and the compare register (TPnCCR0) is used as the cycle setting register.

The TPnCCR0 register and the TPnCCR1 register can be rewritten when TPnCE = 1.

In order for the setting value when the TPnCCR0 register and the TPnCCR1 register are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to the CCRm buffer register), it is necessary to rewrite TPnCCR0 and then write to the TPnCCR1 register before the 16-bit counter value and the TPnCCR0 register value match. Thereafter, the values of the TPnCCR0 and the TPnCCR1 register are reloaded upon a TPnCCR0 register match.

Whether to enable or disable the next reload timing is controlled by writing to the TPnCCR1 register. Thus even when wishing only to rewrite the value of the TPnCCR0 register, also write the same value to the TPnCCR1 register.

Reload is disabled even when only the TPnCCR0 register is rewritten. To stop timer P, set TPnCE = 0. If the external trigger (TIPn0 pin input) edge is detected several times in the external trigger pulse mode, the 16-bit counter is cleared at the edge detection timing and count-up starts.

To realize the same function (software trigger pulse mode) as external trigger pulse mode using a software trigger instead of external trigger input (TIPn0 pin input), set the TPnEST bit of the TPnCTL1 register to 1 so that the software trigger is output. The external trigger pulse waveform is output from TOPn1. The TOPn0 pin performs toggle output upon a match between the TPnCCR0 register and the 16-bit counter.

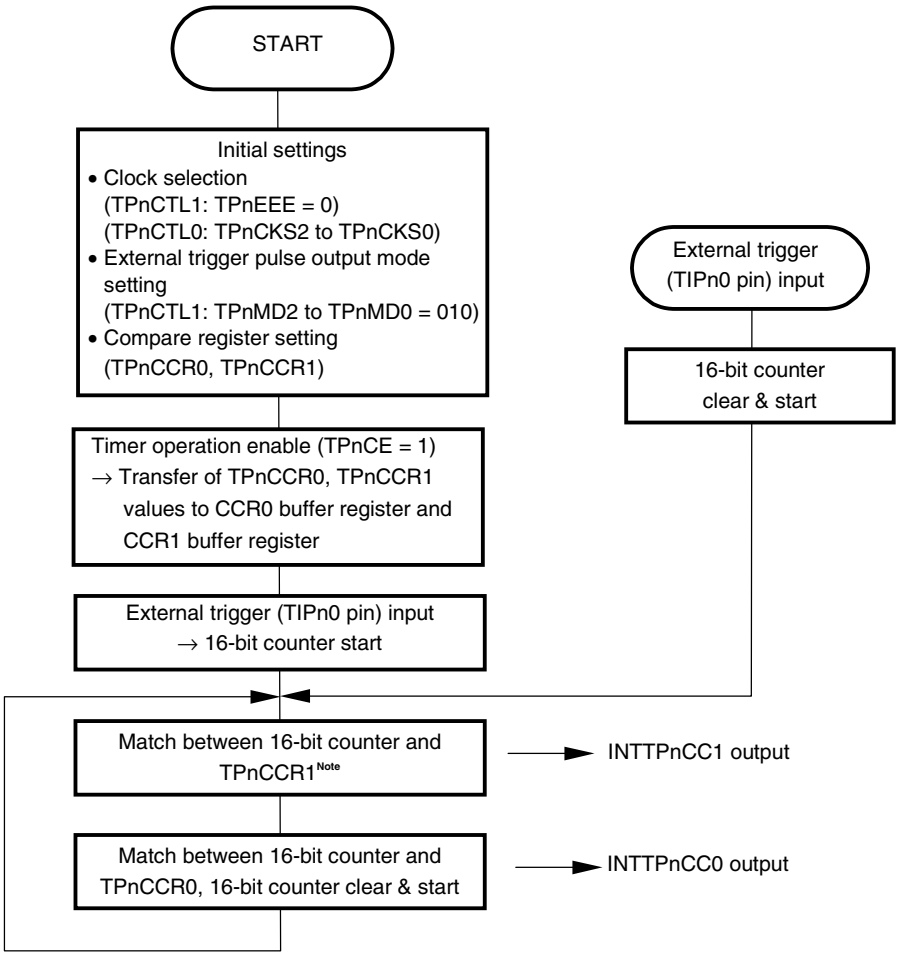
Since the TPnCCR0 register and the TPnCCR1 register have their function fixed to that of a compare register in the external trigger pulse mode, they cannot be used for capture operation in this mode.

Caution In the external trigger pulse mode, select the internal clock (TPnEEE bit of TPnCTL1 register = 0) for the count clock.

Remarks 1. For the reload operation when TPnCCR0 and TPnCCR1 are rewritten during timer operation, refer to **7.5.6 PWM mode**.

2. n = 0 to 5,
m = 0, 1

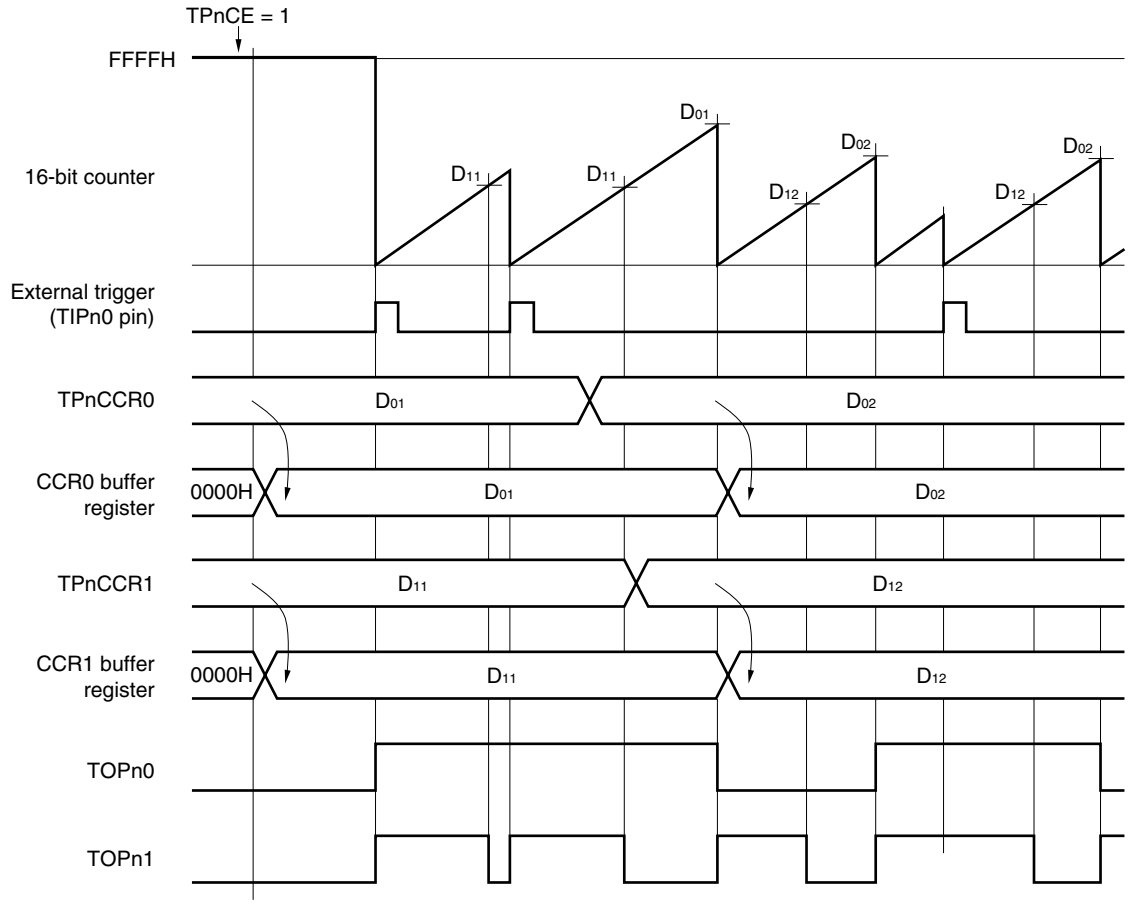
Figure 7-10. Flowchart of Basic Operation in External Trigger Pulse Output Mode



Note The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCR1 buffer register.

Remark n = 0 to 5

Figure 7-11. Basic Operation Timing in External Trigger Pulse Output Mode



Remarks 1. D_{01}, D_{02}: Setting value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting value of TPnCCR1 register (0000H to FFFFH)

2. TOPn1 output duty = (Setting value of TPnCCR1 register)/(Setting value of TP0CCR0 register)
TOPn1 output cycle = (Setting value of TPnCCR0 register) × (Count clock cycle)

3. n = 0 to 5

7.5.5 One-shot pulse mode (TPnMD2 to TPnMD0 = 011)

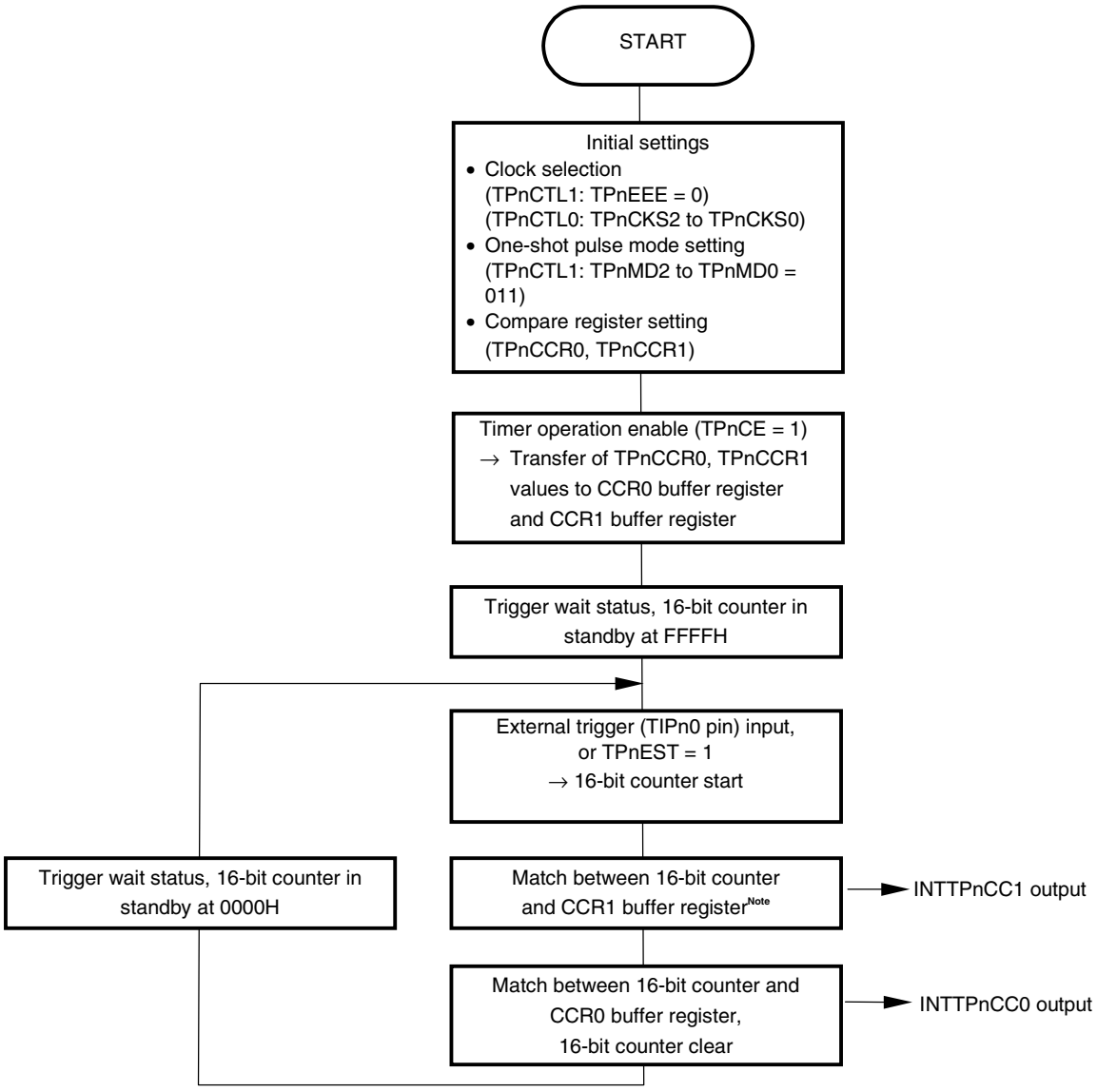
In the one-shot pulse mode, setting TPnCE = 1 causes TPnEST bit setting (1) or TIPn0 pin edge detection trigger wait with the 16-bit counter held at FFFFH. The 16-bit counter starts counting up upon trigger input, and upon a match between the value of the 16-bit counter and the value of the CCR1 buffer register transferred from the TPnCR1 register, TOPn1 becomes high level; Upon a match between the value of the 16-bit counter and the value of the CCR0 register transferred from the TPnCCR0 register, TOPn1 becomes low level and the 16-bit counter is cleared to 0000H and stops. Any trigger input past the first one during 16-bit counter operation is ignored. Be sure to input the second and subsequent triggers when the 16-bit counter has stopped at 0000H. In the one-shot pulse mode, the TPnCCR0 and TPnCCR1 registers can be rewritten when TPnCE = 1. The setting values rewritten to the TPnCCR0 and TPnCCR1 registers become valid following execution of a write instruction from the CPU, at which time they are transferred to the CCR0 buffer register and the CCR0 buffer register through anytime write, and become the values for comparison with the 16-bit counter value. The one-shot pulse waveform is output from the TOPn1 pin. The TOPn0 pin performs toggle output upon a match between the 16-bit counter and the TPnCCR0 register.

Since the TPnCCR0 and TPnCCR1 registers have their function fixed to that of a compare register in the one-shot pulse mode, they cannot be used for capture operation in this mode.

Caution In the one-shot pulse mode, select the internal clock (TPnEEE bit of TPnCTL1 register = 0) for the count clock.

Remark n = 0 to 5

Figure 7-12. Flowchart of Basic Operation in One-Shot Pulse Mode

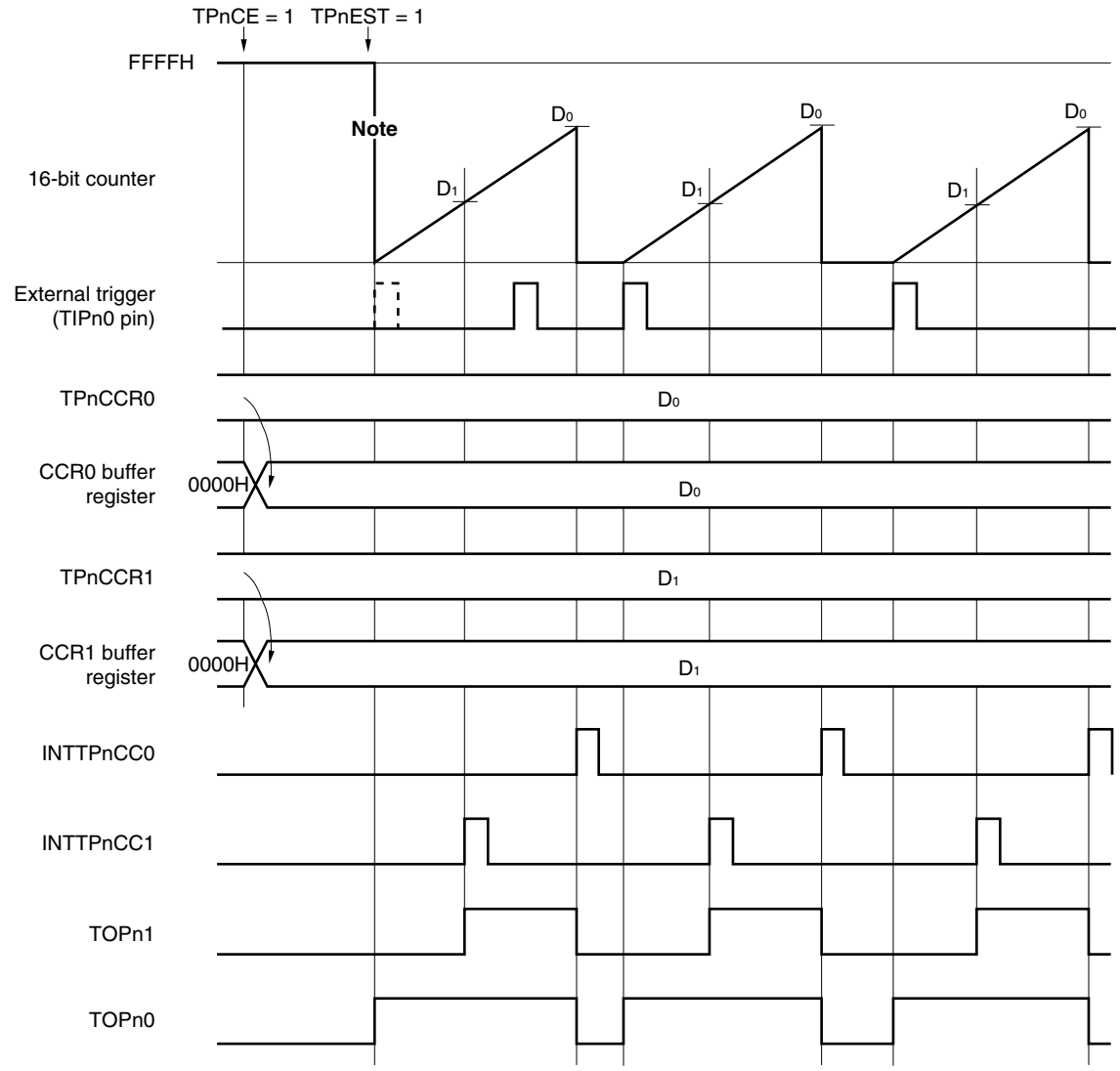


Note The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCR1 buffer register.

Caution The 16-bit counter is not cleared and trigger input is ignored even if trigger input is performed during the count-up operation of the 16-bit counter.

Remark n = 0 to 5

Figure 7-13. Timing of Basic Operation in One-Shot Pulse Mode



Note The 16-bit counter starts counting up when either TPnEST = 1 is set or TIPn0 is input.

Remarks 1. D_0: Setting value of TPnCCR0 register (0000H to FFFFH)
D_1: Setting value of TPnCCR1 register (0000H to FFFFH)

2. n = 0 to 5

7.5.6 PWM mode (TPnMD2 to TPnMD0 = 110)

In the PWM mode, TMPn capture/compare register 1 (TPnCCR1) is used as the duty setting register and TMPn capture/compare register 0 (TPnCCR0) is used as the cycle setting register.

Variable duty PWM is output by setting these two registers and operating the timer.

The TPnCCR0 register and the TPnCCR1 register can be rewritten when TPnCE = 1.

In order for the setting value when the TPnCCR0 register and the TPnCCR1 register are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to CCR0 buffer register or CCR1 buffer register), it is necessary to rewrite TPnCCR0 and then write to the TPnCCR1 register before the 16-bit counter value and the TPnCCR0 register value match. Thereafter, the values of the TPnCCR0 register and the TPnCCR1 register are reloaded upon a TPnCCR0 register match.

Whether to enable or disable the next reload timing is controlled by writing to the TPnCCR1 register. Thus even when wishing only to rewrite the value of the TPnCCR0 register, also write the same value to the TPnCCR1 register.

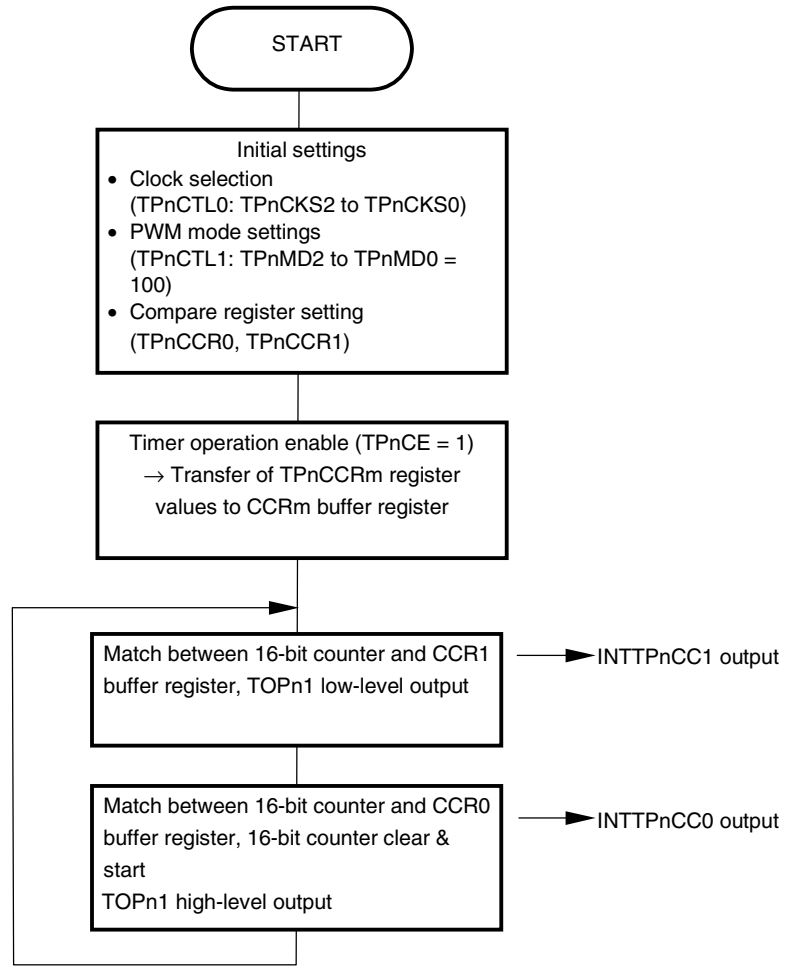
Reload is disabled even when only the TPnCCR0 register is rewritten. To stop timer P, set TPnCE = 0. PWM waveform output is performed from the TOPn1 pin. The TOPn0 pin performs toggle output upon a match between the 16-bit counter and the TPnCCR0 register.

Since the TPnCCR0 and TPnCCR1 registers have their function fixed that of a compare register in the PWM mode, they cannot be used for capture operation in this mode.

Remark n = 0 to 5

Figure 7-14. Flowchart of Basic Operation in PWM Mode (1/2)

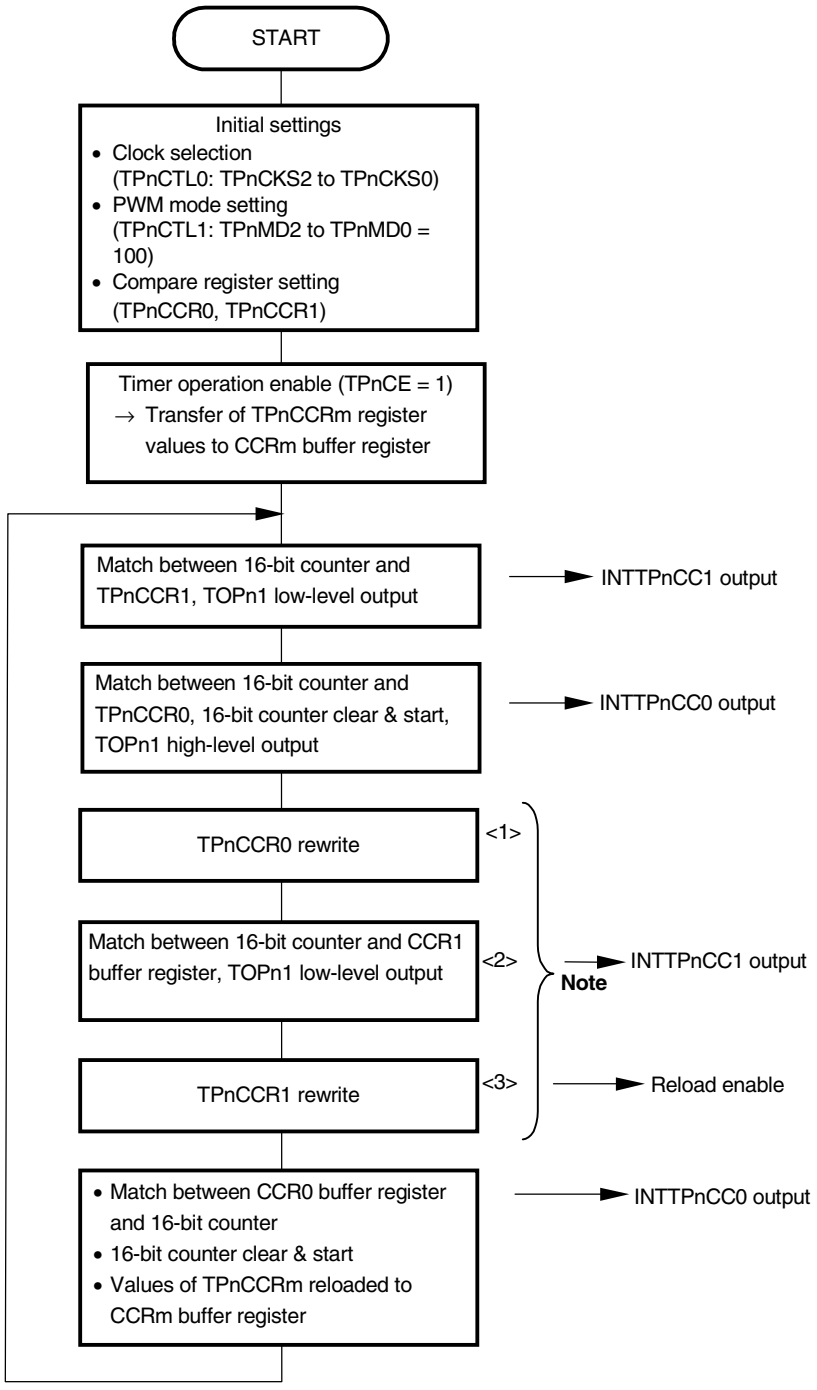
(a) Values of TPnCCR0, TPnCCR1 registers not rewritten during timer operation



Remark n = 0 to 5,
m = 0, 1

Figure 7-14. Flowchart of Basic Operation in PWM Mode (2/2)

(b) Values of TPnCCR0, TPnCCR1 registers rewritten during timer operation

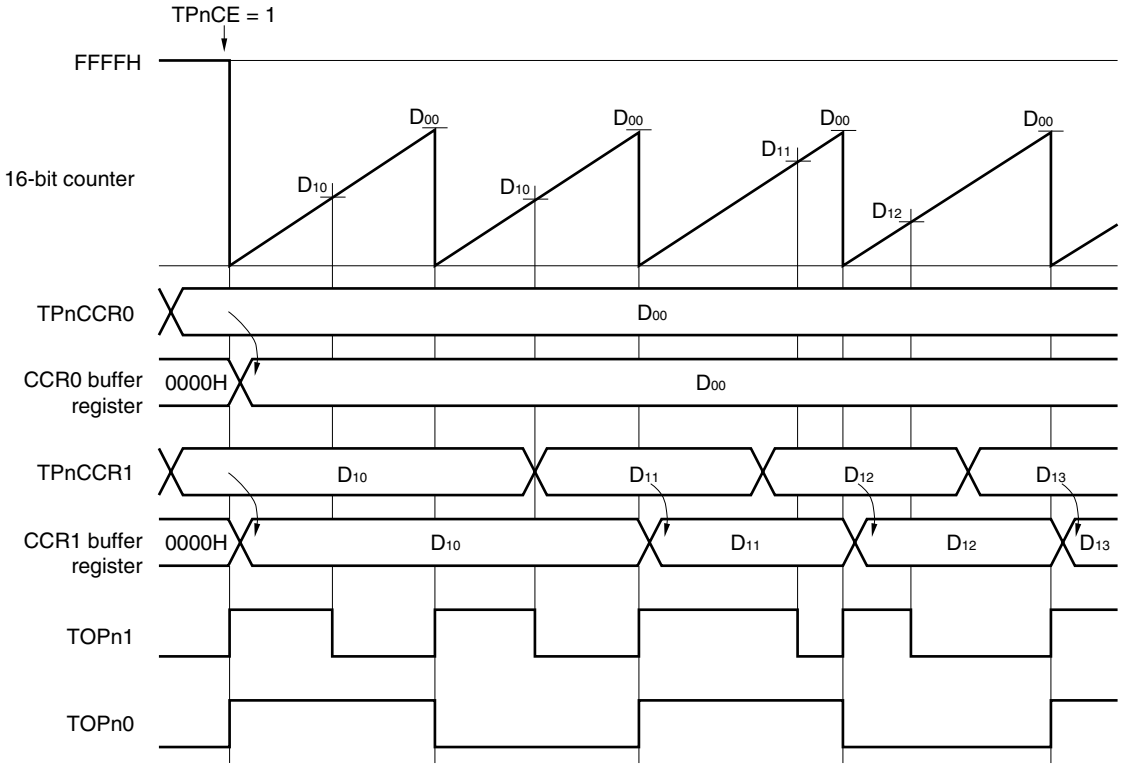


Note The timing of <2> in the above flowchart may differ depending on the rewrite timing of steps <1> and <3> and the value of TPnCCR1, but make sure that step <3> comes after step <1>.

Remark n = 0 to 5
m = 0, 1

Figure 7-15. Basic Operation Timing in PWM Mode (1/2)

(a) TPnCCR1 value rewritten



Remarks 1. D_{00}: Setting value of TPnCCR0 register (0000H to FFFFH)

D_{10}, D_{11}, D_{12}, D_{13}: Setting values of TPnCCR1 register (0000H to FFFFH)

2. TOPn1 output duty = (Setting value of TPnCCR1 register)/(Setting value of TP0CCR0 register)

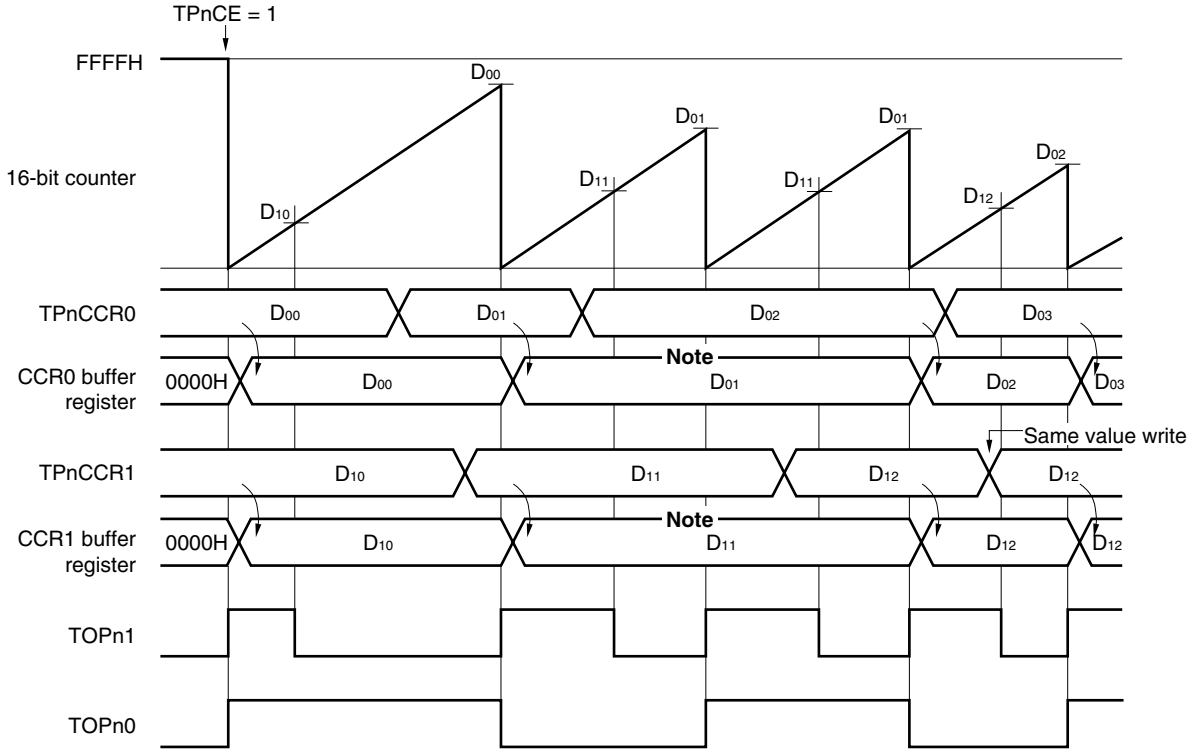
TOPn1 output cycle = (Setting value of TPnCCR0 register) × (Count clock cycle)

TOPn0 output toggle width = (Setting value of TPnCCR0 register + 1) × (Count clock cycle)

3. n = 0 to 5

Figure 7-15. Basic Operation Timing in PWM Mode (2/2)

(b) TPnCCR0, TPnCCR1 values rewritten



Note Reload is not performed because the TPnCCR1 register was not rewritten.

Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Setting values of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Setting values of TPnCCR1 register (0000H to FFFFH)

2. TOPn1 output duty = (Setting value of TPnCCR1 register)/(Setting value of TP0CCR0 register)
TOPn1 output cycle = (Setting value of TPnCCR0 register)/(Count clock cycle)
TOPn0 output toggle width = (Setting value of TPnCCR0 register + 1) × (Count clock cycle)

3. n = 0 to 5

7.5.7 Free-running mode (TPnMD2 to TPnMD0 = 101)

In the free-running mode, both the interval function and the compare function can be realized by operating the 16-bit counter as a free-running counter and selecting capture/compare operation with the TPnCCS1 and TPnCCS0 bits.

The settings of the TPnCCS1 and TPnCCS0 bits of the TPnOPT0 register are valid only in the free-running mode.

TPnCCS1	Operation
0	Use TPnCCR1 register as compare register
1	Use TPnCCR1 register as capture register

TPnCCS0	Operation
0	Use TPnCCR0 register as compare register
1	Use TPnCCR0 register as capture register

- Using TPnCCR1 register as compare register

  An interrupt is output upon a match between the 16-bit counter and the CCR1 buffer register in the free-running mode (interval function).

  Rewrite during compare timer operation is enabled and performed with anytime write. (Once the compare value has been written, synchronization with the internal clock is done and this value is used as the 16-bit counter comparison value.)

  When timer output (TOPn1) has been enabled, TOPn1 performs toggle output upon a match between the 16-bit counter and the CCR1 buffer register.

- Using TPnCCR1 register as capture register

  The value of the 16-bit counter is saved to the TPnCCR1 register upon TIPn1 pin edge detection.

- Using TPnCCR0 register as compare register

  An interrupt is output upon a match between the 16-bit counter and the CCR0 buffer register in the free-running mode (interval function).

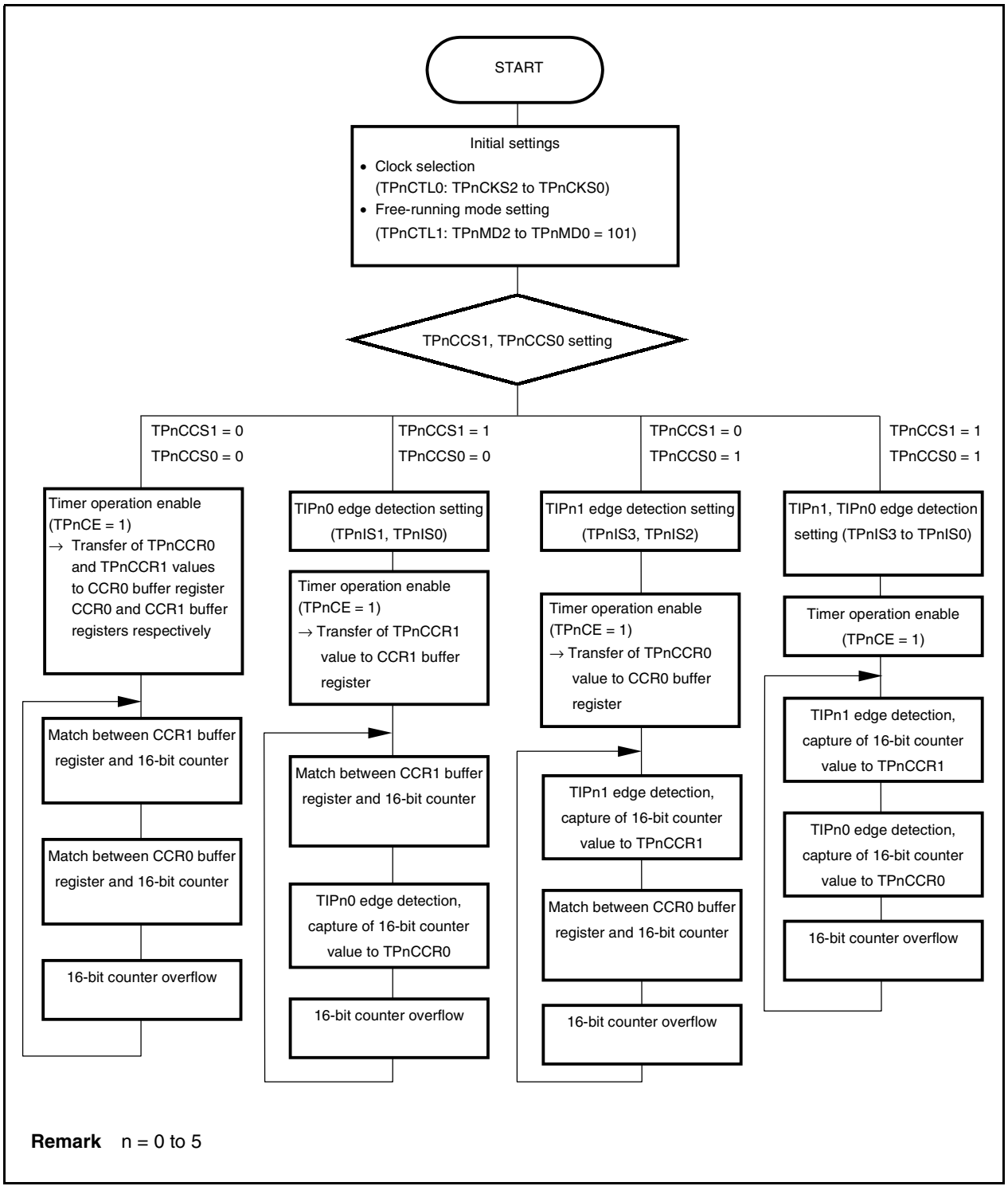
  Rewrite during compare timer operation is enabled and performed with anytime rewrite.

  When timer output (TOPn0) has been enabled, TOPn0 performs toggle output upon a match between the 16-bit counter and the CCR0 buffer register.

- Using TPnCCR0 register as capture register

  The value of the 16-bit counter is saved to the TPnCCR0 register upon TIPn0 pin edge detection.

Figure 7-16. Flowchart of Basic Operation in Free-Running Mode



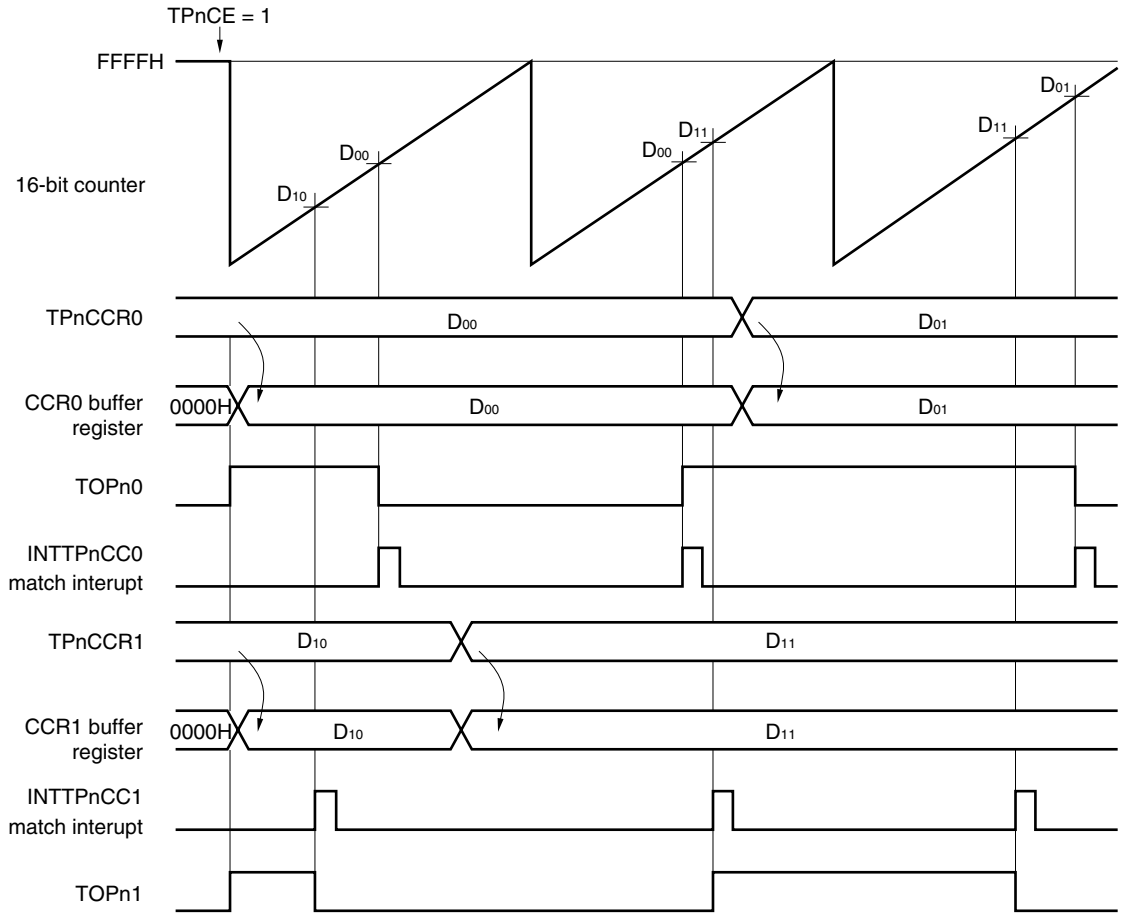
Remark n = 0 to 5

(1) TPnCCS1 = 0, TPnCCS0 = 0 settings (interval function description)

When TPnCE = 1 is set, the 16-bit counter counts from 0000H to FFFFH and the free-running count-up operation continues until TPnCE = 0 is set. In this mode, when a value is written to the TPnCCR0 and TPnCCR1 registers, they are transferred to the CCR0 buffer register and the CCR1 buffer register (anytime write). In this mode, no one-shot pulse is output even when a one-shot pulse trigger is input. Moreover, when TPnOEm = 1 is set, TOPnm performs toggle output upon a match between the 16-bit counter and the CCRm buffer register.

Remark n = 0 to 5,
m = 0, 1

Figure 7-17. Basic Operation Timing in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 0)



Remarks 1. D_{00}, D_{01}: Setting values of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}: Setting values of TPnCCR1 register (0000H to FFFFH)

2. TOPn0 output toggle width = (Setting value of TPnCCR0 register) × (Count clock cycle)
TOPn1 output toggle width = (Setting value of TPnCCR1 register)

3. TOPnm output rises to the high level when counting is started.

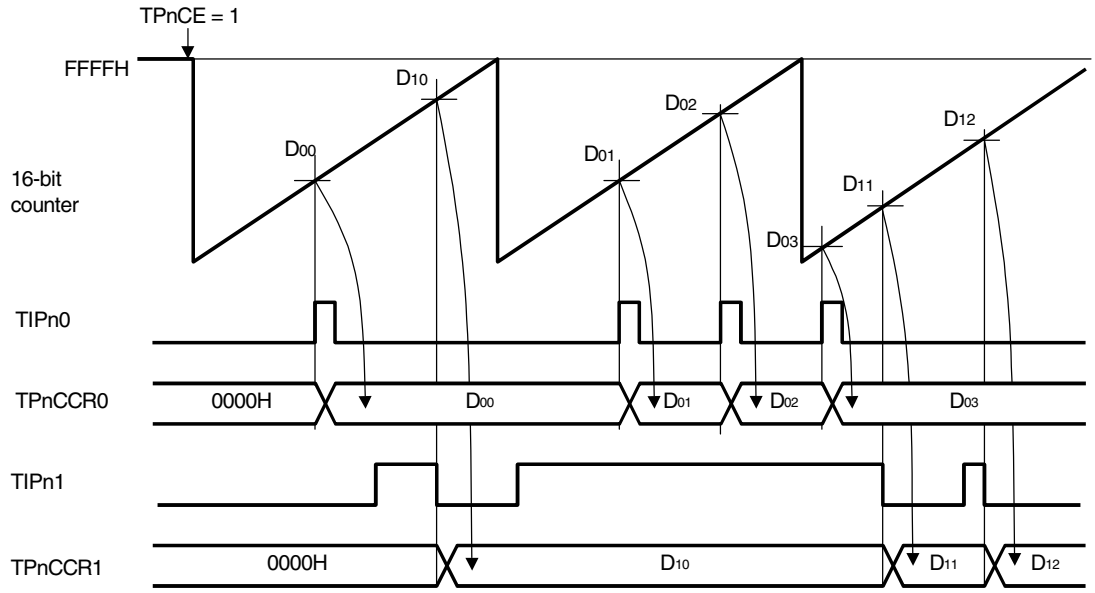
4. n = 0 to 5,
m = 0, 1

(2) TPnCCS1 = 1, TPnCCS0 = 1 settings (capture function description)

When TPnCE = 1, the 16-bit counter counts from 0000H to FFFFH and free-running count-up operation continues until TPnCE = 0 is set. During this time, values are captured by capture trigger operation and are written to the TPnCCR0 and TPnCCR1 registers.

Regarding capture in the vicinity of overflow (FFFFH), judgment is made using the overflow flag (TPnOVF). However, if overflow occurs twice (2 or more free-running cycles), the capture trigger interval cannot be judged with the TPnOVF flag. In this case, the system should be revised.

Figure 7-18. Basic Operation Timing in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 1)



Remarks 1. D_{00}, D_{01}: Values captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}: Values captured to TPnCCR1 register (0000H to FFFFH)

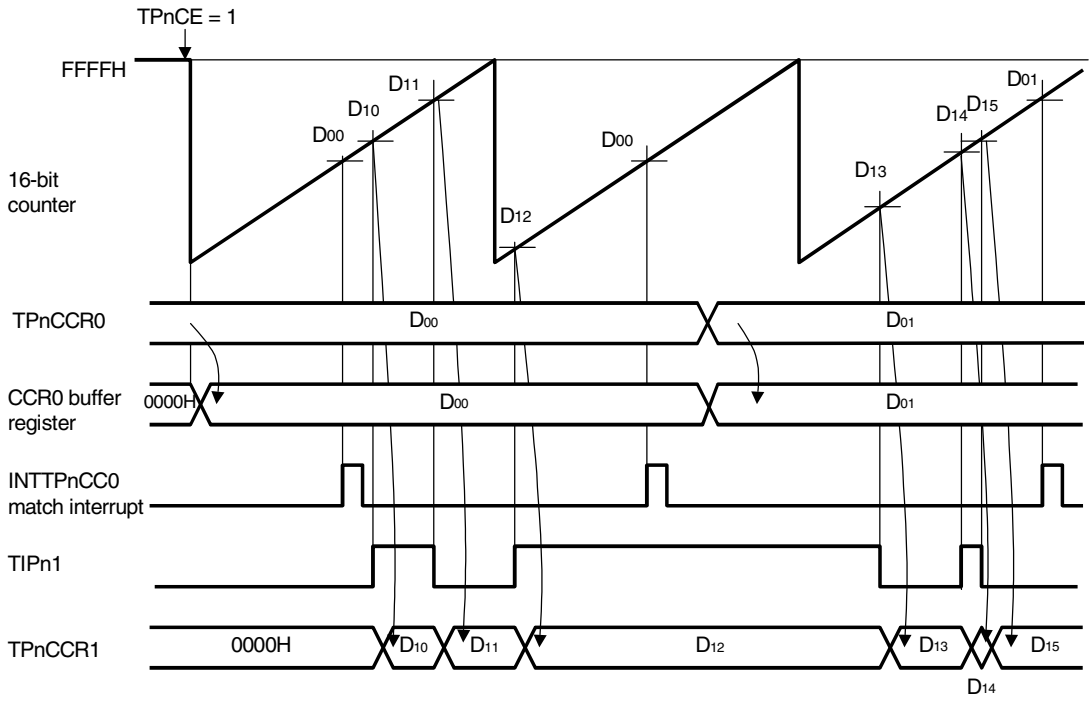
2. TIPn0: Set to rising edge detection (TPnIS1, TPnIS0 = 01)
TIPn1: Set to falling edge detection (TPnIS3, TPnIS2 = 10)

3. n = 0 to 5

(3) TPnCCS1 = 1, TPnCCS0 = 0 settings

When TPnCE = 1 is set, the counter counts from 0000H to FFFFH and free-running count-up operation continues until TPnCE = 0 is set. The TPnCCR0 register is used as a compare register. An interrupt signal is output upon a match between the value of the 16-bit counter and the setting value transferred to the CCR0 buffer register from the TPnCCR0 register as an interval function. Even if TPnOE1 = 1 is set to realize the capture function, the TPnCCR1 register cannot control TOPn1.

Figure 7-19. Basic Operation Timing in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 0)



Remarks 1. D_{00}, D_{01}: Setting values of TPnCCR0 register (0000H to FFFFH)

D_{10}, D_{11}, D_{12}, D_{13}, D_{14}, D_{15}: Values captured to TPnCCR1 register (0000H to FFFFH)

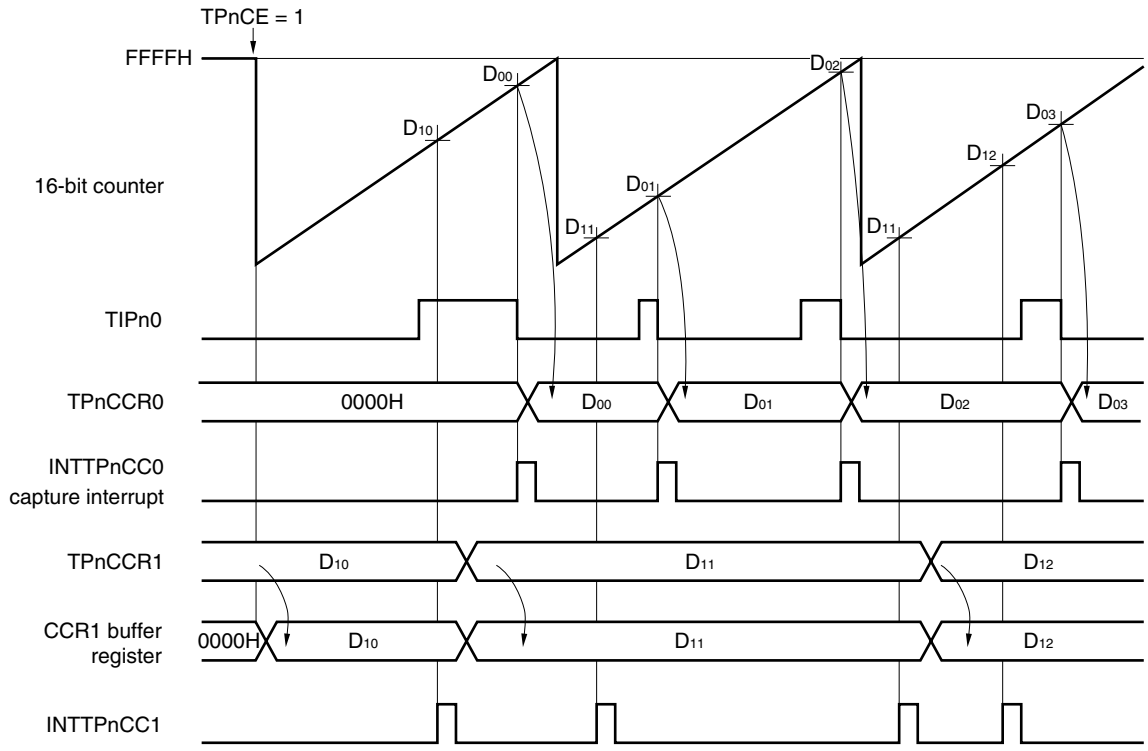
2. TIPn1: Set to detection of both rising and falling edges (TPnIS3, TPnIS2 = 11)

3. n = 0 to 5

(4) TPnCCS1 = 0, TPnCCS0 = 1 settings

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH and free-running count-up operation continues until TPnCE = 0 is set. The TPnCCR1 register is used as a compare register. An interrupt signal is output upon a match between the value of the 16-bit counter and the setting value of the TPnCCR1 register as an interval function. When TPnOE1 = 1 is set, TOPn1 performs toggle output upon mach between the value of the 16-bit counter and the setting value of the TPnCCR1 register.

Figure 7-20. Basic Operation Timing in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 1)



Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Values captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Setting value of TPnCCR1 register (0000H to FFFFH)

2. TIPn0: Set to falling edge detection (TPnIS1, TPnIS0 = 10)

3. n = 0 to 5

(5) Overflow flag

When the counter overflows from FFFFH to 0000H in the free-running mode, the overflow flag (TPnOVF) is set to 1 and an overflow interrupt (INTTPnOV) is output.

Be sure to confirm that the overflow flag (TPnOVF) is set to “1” when the overflow interrupt (INTTPnOV) has occurred.

The overflow flag is cleared by writing 0 from the CPU.

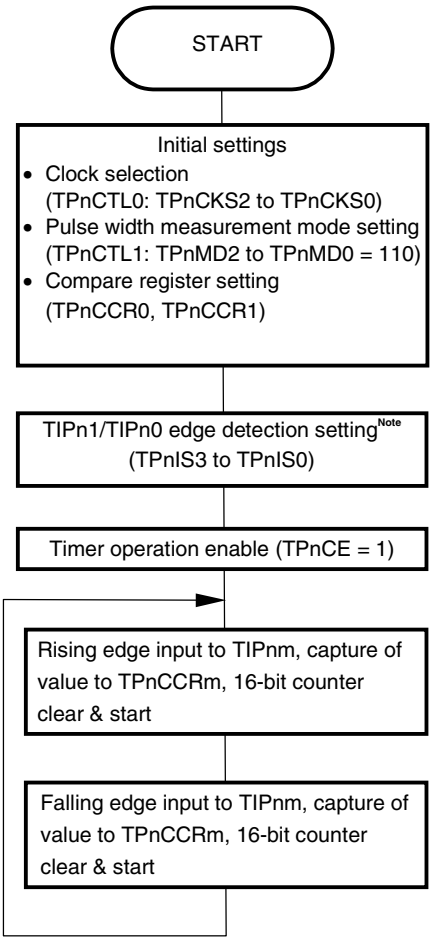
7.5.8 Pulse width measurement mode (TPnMD2 to TPnMD0 = 110)

In the pulse width measurement mode, free-running count is performed, and upon detection of both the rising and falling edges of TIPn0 pin, the 16-bit counter value is saved to capture register 0 (TPnCCR0) and the 16-bit counter is cleared to 0000H. The external input pulse width can be measured as a result.

However, when measuring a large pulse width that exceeds 16-bit counter overflow, perform judgment with the overflow flag. Since measurement of pulses for which overflow occurs twice or more is not possible, adjust the operating frequency of the 16-bit counter. The value of the 16-bit counter is also saved to capture register 1 (TPnCCR1) and the 16-bit counter cleared upon edge detection of the TIPn1 pin.

Caution In the pulse width measurement mode, select the internal clock (TPnEEE of TPnCTL1 register = 0).

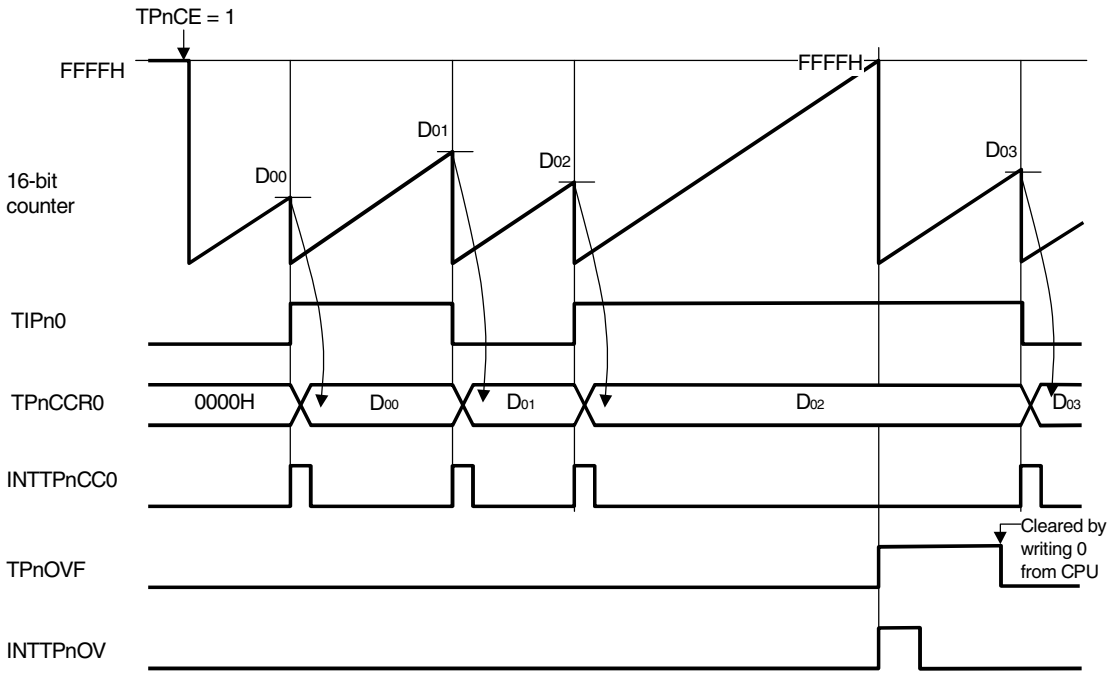
Figure 7-21. Flowchart of Basic Operation in Pulse Width Measurement Mode



Note External pulse input is possible both for TIPn0 and TIPn1, but only one can be selected.
Specify “both the rising and the falling edges” for edge detection. Specify the edge of the external input pulse that is not used as “No edge to be detected”.

Remark n = 0 to 5,
m = 0, 1

Figure 7-22. Basic Operation Timing in Pulse Width Measurement Mode



Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Values captured to TPnCCR0 register (0000H to FFFFH)

2. TIPn0: Set to detection of both rising and falling edges (TPnIS1, TPnIS0 = 11)

3. n = 0 to 5

CHAPTER 8 16-BIT TIMER/EVENT COUNTER Q

8.1 Features

Timer Q (TMQ) is a 16-bit timer/event counter that can be used in various ways.
TMQ can perform the following operations.

- PWM output
- Interval timer
- External event count (operation not possible when clock is stopped)
- One-shot pulse output
- Pulse width measurement function

8.2 Function Outline

- Capture trigger input signal × 4
- External trigger input signal × 1
- Clock select × 8
- External event count input × 1
- Readable counter × 1
- Capture/compare reload register × 4
- Capture/compare match interrupt × 4
- Timer output (TOQ00 to TOQ04) × 4

8.3 Configuration

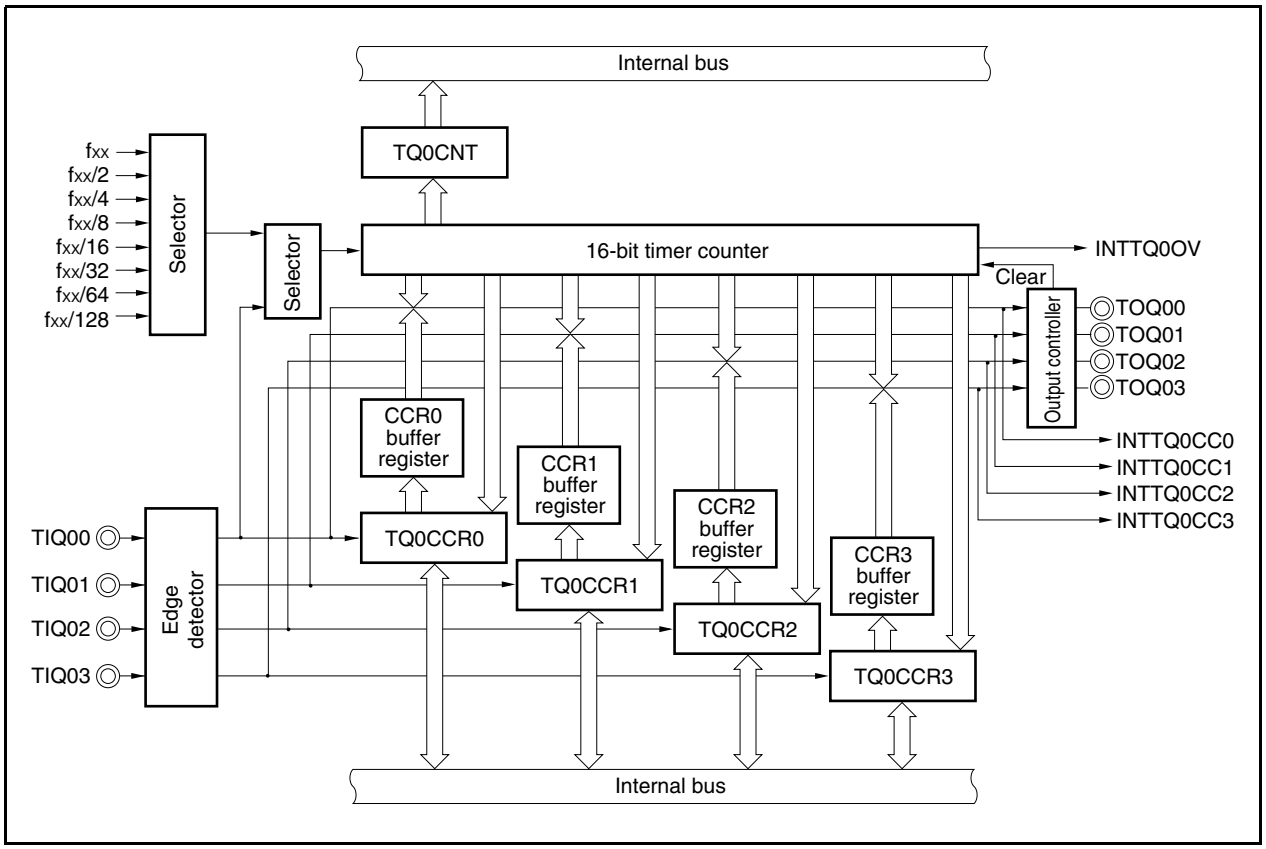
TMQ includes the following hardware.

Table 8-1. TMQ Configuration

Item	Configuration
Timer register	16-bit counter × 1
Registers	TMQ0 timer capture/compare registers 0 to 3 (TQ0CCR0 to TQ0CCR3) TMQ0 read buffer register (TQ0CNT) CCR0 buffer register to CCR3 buffer register
Timer input	4 (TIQ00[Note] to TIQ03)
Timer output	4 (TOQ00 toTOQ03)
Control registers	TMQ0 control registers 0, 1 (TQ0CTL0, TQ0CTL1) TMQ0 I/O control registers 0 to 2 (TQ0IOC0 to TQ0IOC2) TMQ0 option register 0 (TQ0OPT0)

Note TIQ00 is multiplexed with a capture trigger input signal, external trigger input signal, and external event input signal.

Figure 8-1. Timer Q Block Diagram



(1) TMQ0 capture/compare register 0 (TQ0CCR0)

The TQ0CCR0 register is a 16-bit register that functions both as a capture register and as a compare register. Whether this register functions as a capture register or as a compare register can be controlled with the TQ0CCS0 bit of the TQ0OPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

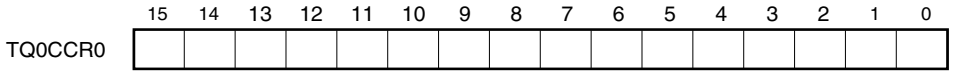
In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TQ0CCR0 register is a compare register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: FFFFF546H



- Use as a compare register
  TQ0CCR0 can be rewritten when TQ0CE = 1
  The rewrite method is as follows.

TMQ Operation Mode	Method of Writing TQ0CCR0 Register
PWM mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as capture register
  The counter value is saved to TQ0CCR0 upon capture trigger (TIQ00) input edge detection.

(2) TMQ0 capture/compare register 1 (TQ0CCR1)

The TQ0CCR1 register is a 16-bit register that functions both as a capture register and as a compare register. Whether this register functions as a capture register or as a compare register can be controlled with the TQ0CCS1 bit of the TQ0OPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TQ0CCR1 register is a reload register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: FFFFF548H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TQ0CCR1																

- Use as a compare register
  TQ0CCR1 can be rewritten when TQ0CE = 1
  The rewrite method is as follows.

TMQ Operation Mode	Method of Writing TQ0CCR1 Register
PWM mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as a capture register
  The counter value is saved to TQ0CCR1 upon capture trigger (TIQ01) input edge detection.

(3) TMQ0 capture/compare register 2 (TQ0CCR2)

The TQ0CCR2 register is a 16-bit register that functions both as a capture register and as a compare register.

Whether this register functions as a capture register or as a compare register can be controlled with the TQ0CCS2 bit of the TQ0OPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TQ0CCR2 register is a compare register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: FFFFF54AH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TQ0CCR2																

- Use as a compare register

  TQ0CCR2 can be rewritten when TQ0CE = 1

  The rewrite method is as follows.

TMQ Operation Mode	Method of Writing TQ0CCR2 Register
PWM mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as capture register

  The counter value is saved to TQ0CCR2 upon capture trigger (TIQ02) input edge detection.

(4) TMQ0 capture/compare register 3 (TQ0CCR3)

The TQ0CCR3 register is a 16-bit register that functions both as a capture register and as a compare register.

Whether this register functions as a capture register or as a compare register can be controlled with the TQ0CCS3 bit of the TQ0OPT0 register, but only in the free-running mode.

In the pulse width measurement mode, this register can be used as a dedicated capture register (the compare function cannot be used.)

In modes other than the free-running mode and pulse width measurement mode, this register is used as a dedicated compare register.

In the initial setting, the TQ0CCR3 register is a compare register.

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

Reset input clears this register to 0000H.

After reset: 0000H R/W Address: FFFFF54CH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TQ0CCR3																

- Use as a compare register
  TQ0CCR3 can be rewritten when TQ0CE = 1
  The rewrite method is as follows.

TMQ Operation Mode	Method of Writing TQ0CCR3 Register
PWM mode, external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse mode, interval timer mode	Anytime write
Pulse width measurement mode	Cannot be used because dedicated capture register

- Use as capture register
  The counter value is saved to TQ0CCR3 upon capture trigger (TIQ03) input edge detection.

(5) TMQ0 read buffer register 0 (TQ0CNT)

The TQ0CNT register is a read buffer register that can read 16-bit counter values.

This register is read-only, in 16-bit units.

Reset input clears this register to 0000H.

After reset: 0000H R Address: FFFFF54EH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TQ0CNT																

8.4 Control Registers

(1) TMQ0 control register 0 (TQ0CTL0)

The TQ0CTL0 register is an 8-bit register that controls the operation of timer Q.

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

Reset input clears this register to 00H.

The same value can always be written to the TQ0CTL0 register by software.

After reset: 00H R/W Address: FFFFF540H

	<7>	6	5	4	3	2	1	0
TQ0CTL0	TQ0CE	0	0	0	0	TQ0CKS2	TQ0CKS1	TQ0CKS0

TQ0CE	Timer Qn operation control
0	Internal operating clock operation disabled (TMQ0 reset asynchronously)
1	Internal operating clock operation enabled

Internal operating clock control and TMQ0 asynchronous reset are performed with the TQ0CE bit. When the TQ0CE bit is cleared to 0, the internal operating clock of TMQ0 stops (fixed to low level) and TMQ0 is reset asynchronously.
When the TQ0CE bit is set to 1, the internal operating clock is enabled and count-up operation starts within 2 input clocks after the TQ0CE bit was set to 1.

TQ0CKS2	TQ0CKS1	TQ0CKS0	Internal count clock selection
0	0	0	f_{XX}
0	0	1	$f_{XX}/2$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/8$
1	0	0	$f_{XX}/16$
1	0	1	$f_{XX}/32$
1	1	0	$f_{XX}/64$
1	1	1	$f_{XX}/128$

Caution Set the TQ0CKS2 to TQ0CKS0 bits when TQ0CE = 0. When the value of the TQ0CE bit is changed from 0 to 1, the TQ0CKS2 to TQ0CKS0 bits can be set simultaneously.

(2) TMQ0 control register 1 (TQ0CTL1)

After reset: 00H R/W Address: FFFFF541H

	7	<6>	<5>	4	3	2	1	0
TQ0CTL1	0	TQ0EST	TQ0EEE	0	0	TQ0MD2	TQ0MD1	TQ0MD0

TQ0EST	Software trigger control
0	No operation
1	In one-shot pulse mode: One-shot pulse software trigger
	In external trigger pulse output mode: Pulse output software trigger

The TQ0EST bit functions as a software trigger in the one-shot pulse mode and the external trigger pulse output mode. (This bit is invalid even if it is controlled in any other mode.) TQ0EST functions as a software trigger if it is set to 1 when TQ0CE = 1. Therefore, be sure to set TQ0EST to 1 when TQ0CE = 1.
The TIQ00 pin is used for an external trigger.
The read value of the TQ0EST bit is always 0.

TQ0EEE	Count clock selection
0	Use the internal clock (clock selected with bits TQ0CKS2 to TQ0CKS0)
1	Use the clock input from the TIQ00 pin

The valid edge when TQ0EET = 1 (the clock input from the TIQ00 pin) is specified with bits TQ0EES1 and TQ0EES0.

TQ0MD2	TQ0MD1	TQ0MD0	Timer mode selection
0	0	0	Interval timer mode
0	0	1	External event count mode
0	1	0	External trigger pulse output mode
0	1	1	One-shot pulse mode
1	0	0	PWM mode
1	0	1	Free-running mode
1	1	0	Pulse width measurement mode
1	1	1	Setting prohibited

Caution Set the TQ0EEE and TQ0MD2 to TQ0MD0 bits when TQ0CE = 0. (The same value can be written when TQ0CE = 1.) The operation is not guaranteed when rewriting is performed when TQ0CE = 1. If rewriting was mistakenly performed, set TQ0CE = 0 and then set the bits again.

(3) TMQ0 I/O control register 0 (TQ0IOC0)

The TQ0IOC0 register is an 8-bit register that controls the timer output.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF542H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
TQ0IOC0	TQ0OL3	TQ0OE3	TQ0OL2	TQ0OE2	TQ0OL1	TQ0OE1	TQ0OL0	TQ0OE0

TQ0OLn	Timer output level setting (n = 0 to 3)
0	Normal output
1	Inverted output

TQ0OEn	Timer output setting (n = 0 to 3)
0	Timer output prohibited (Low level and high level are output from TOQ0n pin when TQ0OLn = 0 and TQ0OLn = 1, respectively.)
1	Timer output enabled (A pulse is output from the TOQ0n pin.)

Caution Rewrite the TQ0OLn and TQ0OEn bits when TQ0CE = 0. (The same value can be written when TQ0CE = 1.) If rewriting was mistakenly performed, set TQ0CE = 0 and then set the bits again.

(4) TMQ0 I/O control register 1 (TQ0IOC1)

The TQ0IOC1 register is an 8-bit register that controls the valid edge for the external input signals (TIQ00 to TIQ03).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF593H

	7	6	5	4	3	2	1	0
TQ0IOC1	TQ0IS7	TQ0IS6	TQ0IS5	TQ0IS4	TQ0IS3	TQ0IS2	TQ0IS1	TQ0IS0

TQ0IS7	TQ0IS6	Capture input (TIQ03) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TQ0IS5	TQ0IS4	Capture input (TIQ02) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TQ0IS3	TQ0IS2	Capture input (TIQ01) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TQ0IS1	TQ0IS0	Capture input (TIQ00) valid edge detection
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TQ0IS3 to TQ0IS0 bits when TQ0CE = 0. (The same value can be written when TQ0CE = 1.) If rewriting was mistakenly performed, set TQ0CE = 0 and then set the bits again.

2. The TQ0IS7 to TQ0IS0 bits are valid only in the free-running mode and pulse width measurement mode. In the other modes, the capture operation is not performed.

(5) TMQ0 I/O control register 2 (TQ0IOC2)

The TQ0IOC2 register is an 8-bit register that controls the valid edge for external event counter input signal (TIQ00) and external trigger input signal (TIQ00).

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF544H

	7	6	5	4	3	2	1	0
TQ0IOC2	0	0	0	0	TQ0EES1	TQ0EES0	TQ0ETS1	TQ0ETS0

TQ0EES1	TQ0EES0	External event count input (TIQ00) valid edge setting
0	0	No edge detection (external event count invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TQ0ETS1	TQ0ETS0	External trigger input (TIQ00) valid edge detection
0	0	No edge detection (external trigger invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TQ0EESn and TQ0ESTn bits when TQ0CE = 0. (The same value can be written when TQ0CE = 1.) If rewriting was mistakenly performed, set TQ0CE = 0 and then set the bits again.

2. The TQnEES1 and TQnEES0 bits are valid only when the TQnEEE bit = 1 or when the external event count mode is set (TQ0MD2 to TQ0MD0 of TQ0CTL1 register = 001).

(6) TMQ0 option register 0 (TQ0OPT0)

The TQ0OPT0 register is an 8-bit register used to set the capture/compare operation and detect overflow.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF545H

	7	6	5	4	3	2	1	<0>
TQ0OPT0	TQ0CCS3	TQ0CCS2	TQ0CCS1	TQ0CCS0	0	0	0	TQ0OVF

TQ0CCSn	TQ0CCRn register capture/compare selection (n = 0 to 3)
0	Compare register selection
1	Capture register selection
The TQ0CCSn bit setting is valid only in the free-running mode.	

TQ0OVF	Timer Q overflow detection
Set (1)	Overflow occurrence
Reset (0)	TQ0OVF bit 0 write or TQ0CE = 0
• The TQ0OVF bit is reset when the 16-bit counter value overflows from FFFFH to 0000H in the free-running mode or the pulse width measurement mode. • An interrupt request signal (INTTQ0OV) is generated at the same time that the TQ0OVF bit is set (1). The INTTQ0OV signal is not generated in modes other than the free-running mode and the pulse width measurement mode. • The TQ0OVF bit is not cleared even when the TQ0OVF bit and the TQ0OPT0 register are read when TQ0OVF = 1. • The TQ0OVF bit can be both read and written, but 1 cannot be written to the TQ0OVF bit from the CPU. Writing 1 has no influence on the operation of timer Q.	

Caution Rewrite the TQ0CCS3 to TQ0CCS0 bits when TQ0CE = 0. (The same value can be written when TQ0CE = 1.) If rewriting was mistakenly performed, set TQ0CE = 0 and then set the bits again.

8.5 Operation

Timer Q can perform the following operations.

Operation	TQ0EST (Software Trigger Bit)	TIQ00 (External Trigger Input)	Capture/Compare Write	Compare Write
Interval timer mode	Invalid	Invalid	Compare only	Anytime write
External event count mode[Note 1]	Invalid	Invalid	Compare only	Anytime write
External trigger pulse output mode[Note 2]	Valid	Valid	Compare only	Reload
One-shot pulse output mode[Note 2]	Valid	Valid	Compare only	Anytime write
PWM mode	Invalid	Invalid	Compare only	Reload
Free-running mode	Invalid	Invalid	Capture/compare switching enabled	Anytime write
Pulse width measurement mode[Note 2]	Invalid	Invalid	Capture only	Not applicable

Notes 1. To use the external event count function, specify that the edge of the TIQ00 capture input is not detected (by clearing the TQ0IS1 and TQ0IS0 bits of the TQ0IOC1 register to “00”).

2. When using the external trigger pulse output mode, one-shot pulse mode, or pulse width measurement mode, select the internal clock as the count clock (by setting the TQ0EEE bit of the TQ0CTL1 register to 1).

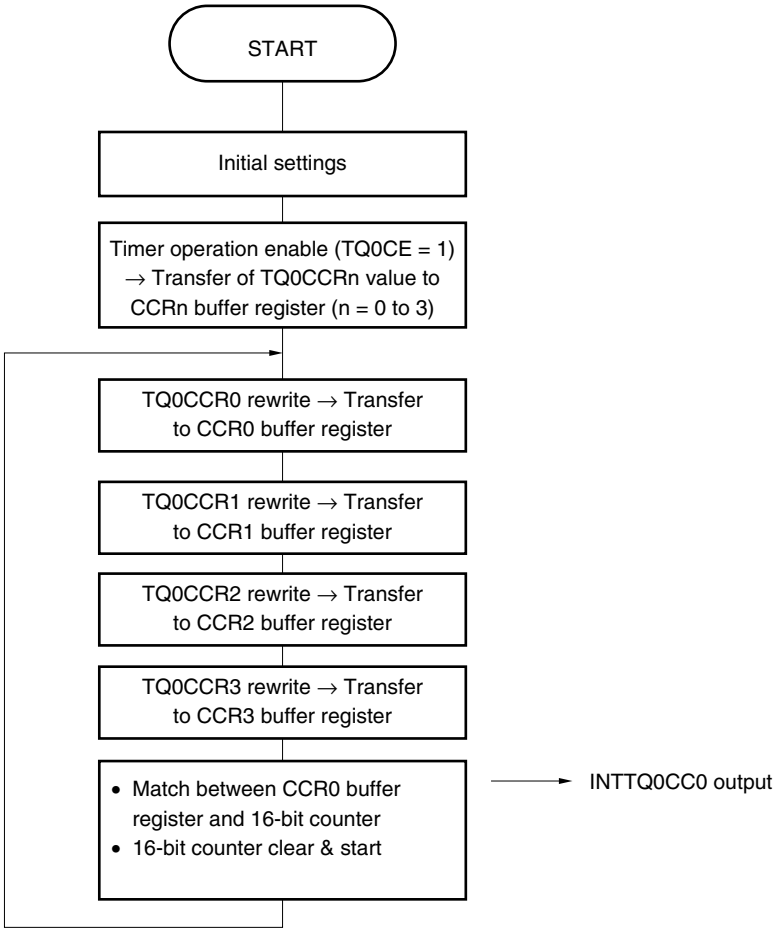
8.5.1 Anytime write and reload

TQ0CCR0 to TQ0CCR3 register rewrite is possible for timer Q during timer operation (TQ0CE = 1), but the write method (anytime write, reload) differs depending on the mode.

(1) Anytime write

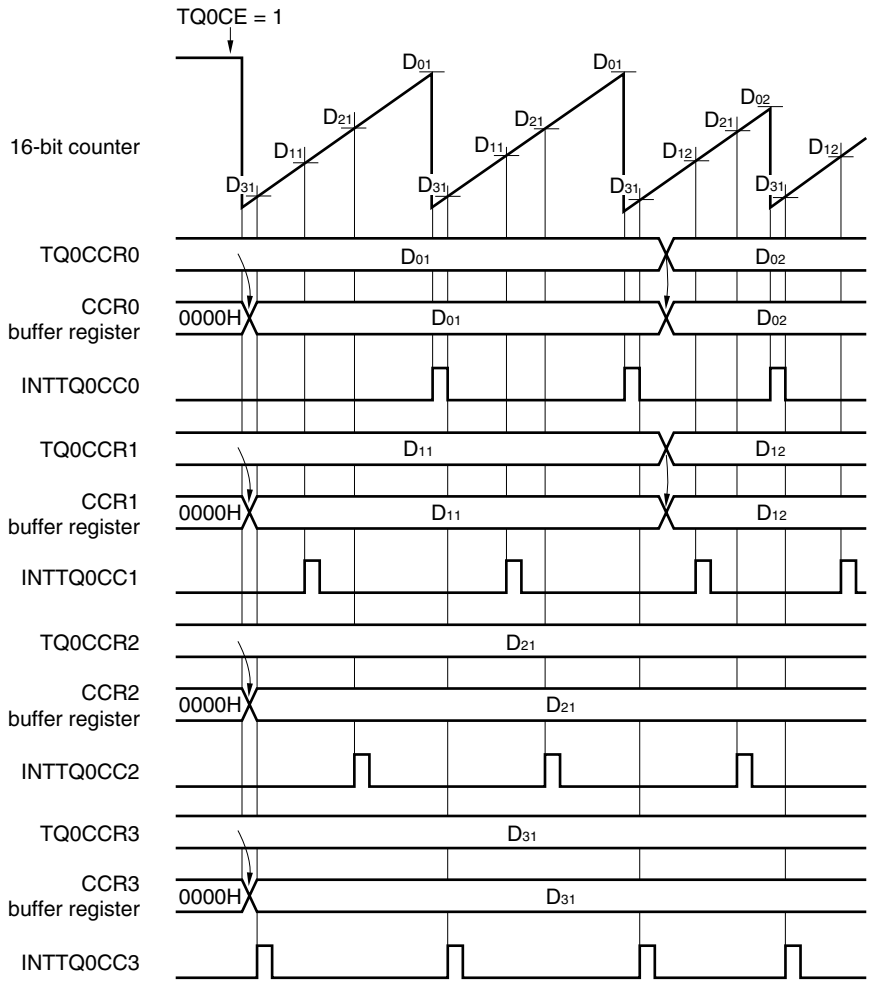
When the TQ0CCR0 to TQ0CCR3 registers are written during timer operation, the write data is transferred at that time to the CCR0 buffer register and used as the 16-bit counter comparison value.

Figure 8-2. Flowchart of Basic Operation for Anytime Write



Remark The above flowchart illustrates an example of the operation in the interval timer mode.

Figure 8-3. Timing Diagram for Anytime Write



Remarks 1. D_{01}, D_{02}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. The above timing chart illustrates an example of the operation in the interval timer mode.

(2) Reload

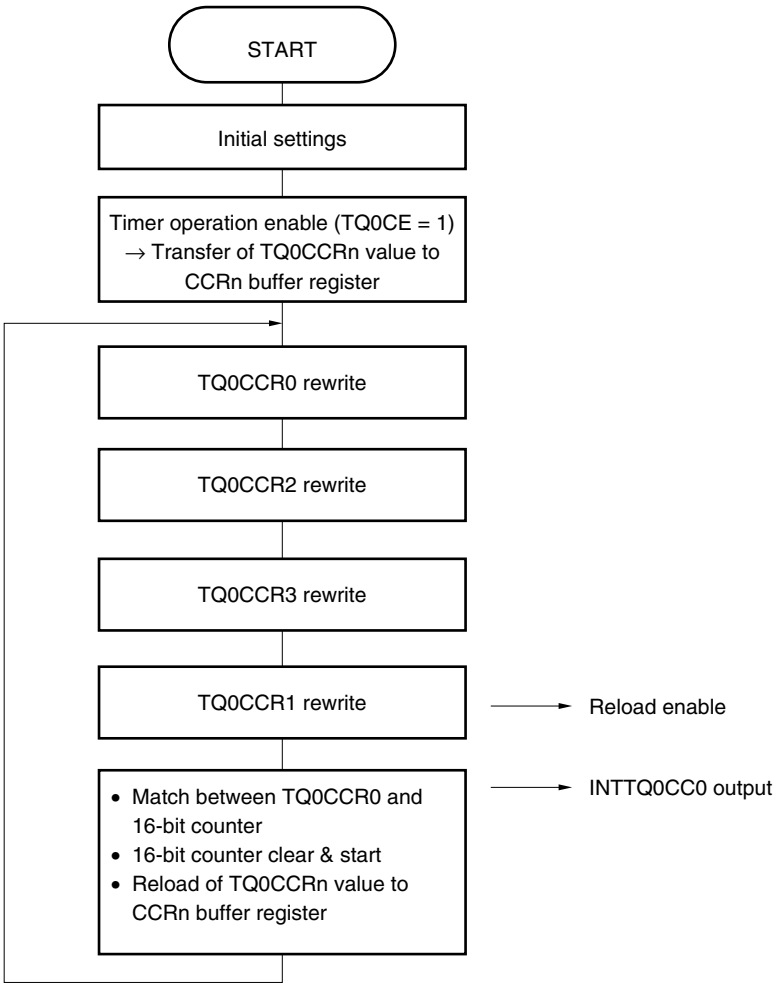
When the TQ0CCRn register is written during timer operation via the CCRn buffer register, the write data is used as the 16-bit counter comparison value. The TQ0CCRn register can be rewritten when TQ0CE = 1.

In order for the setting value when the TQ0CCRn register is rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to the CCRn buffer register), it is necessary to rewrite TQ0CCR0 and finally write to the TQ0CCR1 register before the 16-bit counter value and the CCRn buffer register value match.

When the CCRn buffer register value later matches the 16-bit counter value, the TQ0CCRn register value is reloaded to the CCRn buffer register.

Whether to enable or disable the next reload timing is controlled by writing to the TQ0CCR1 register.

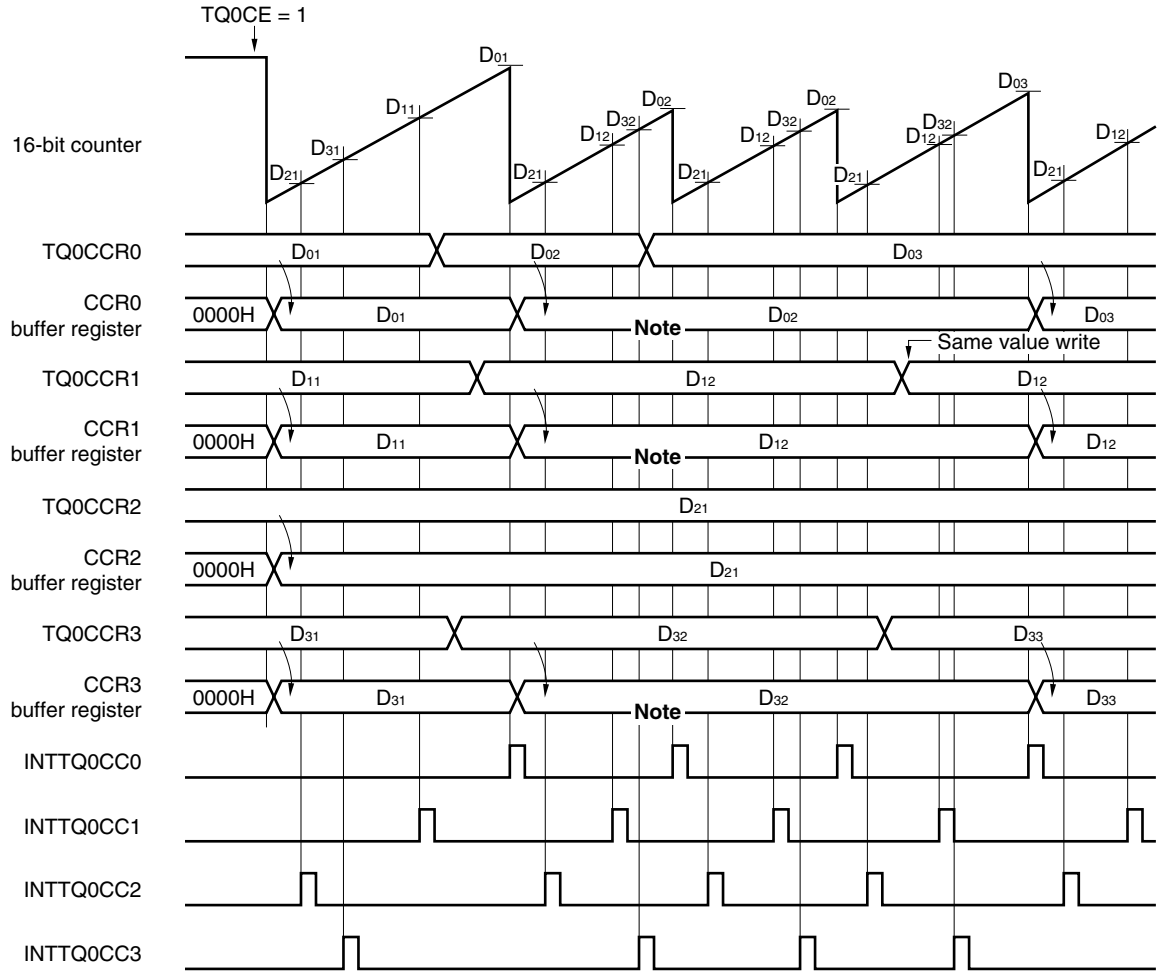
Figure 8-4. Flowchart of Basic Operation for Reload



Caution Writing to the TQ0CCR1 register includes enabling of reload. Therefore, rewrite the TQ0CCR1 register after rewriting the other TQ0CCR registers.

Remark The above flowchart illustrates an example of the operation in the PWM mode.

Figure 8-5. Timing Chart for Reload



Note Reload is not performed because the TQ0CCR1 register was not rewritten.

Remarks 1. D_{01}, D_{02}, D_{03}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}, D_{32}, D_{33}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. The above flowchart illustrates the operation in the PWM mode as an example.

8.5.2 Interval timer mode (TQ0MD2 to TQ0MD0 = 000)

In the interval timer mode, an interrupt request signal (INTTQ0CC0) is output upon a match between the setting value of the TQ0CCR0 register and the value of the 16-bit counter, and the 16-bit counter is cleared. The TQ0CCRn register can be rewritten when TQ0CE = 1, and when a value is set to TQ0CCRn with a write instruction from the CPU, it is transferred to the CCRn buffer register through anytime write, and is used as the value for comparison with the 16-bit counter value.

In the interval timer mode, the 16-bit counter is cleared only upon a match between the value of the 16-bit counter and the value of the CCR0 buffer register.

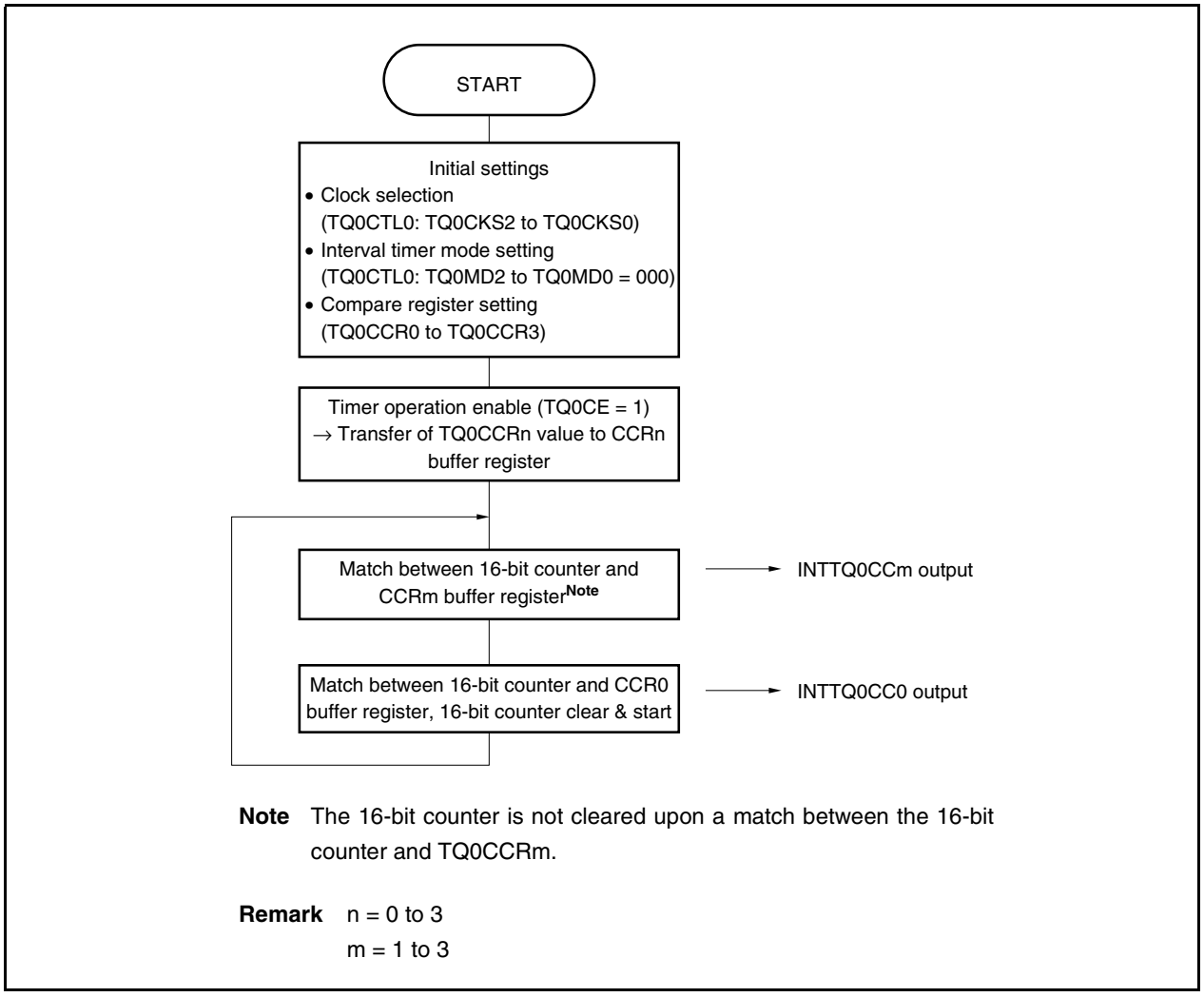
16-bit counter clearing using the TQ0CCRm register is not performed. However, the setting value of the TQ0CCRm register is transferred to the CCRm buffer register and compared with the value of the 16-bit counter, and an interrupt request (INTTQ0CCm) is output if these values match.

Moreover, TOQ0n pin output is also possible by setting the TQ0OEn bit to 1.

When the TQ0CCRm register is not used, it is recommended to set FFFFH as the setting value for the TQ0CCRm register.

Remark n = 0 to 3
m = 1 to 3

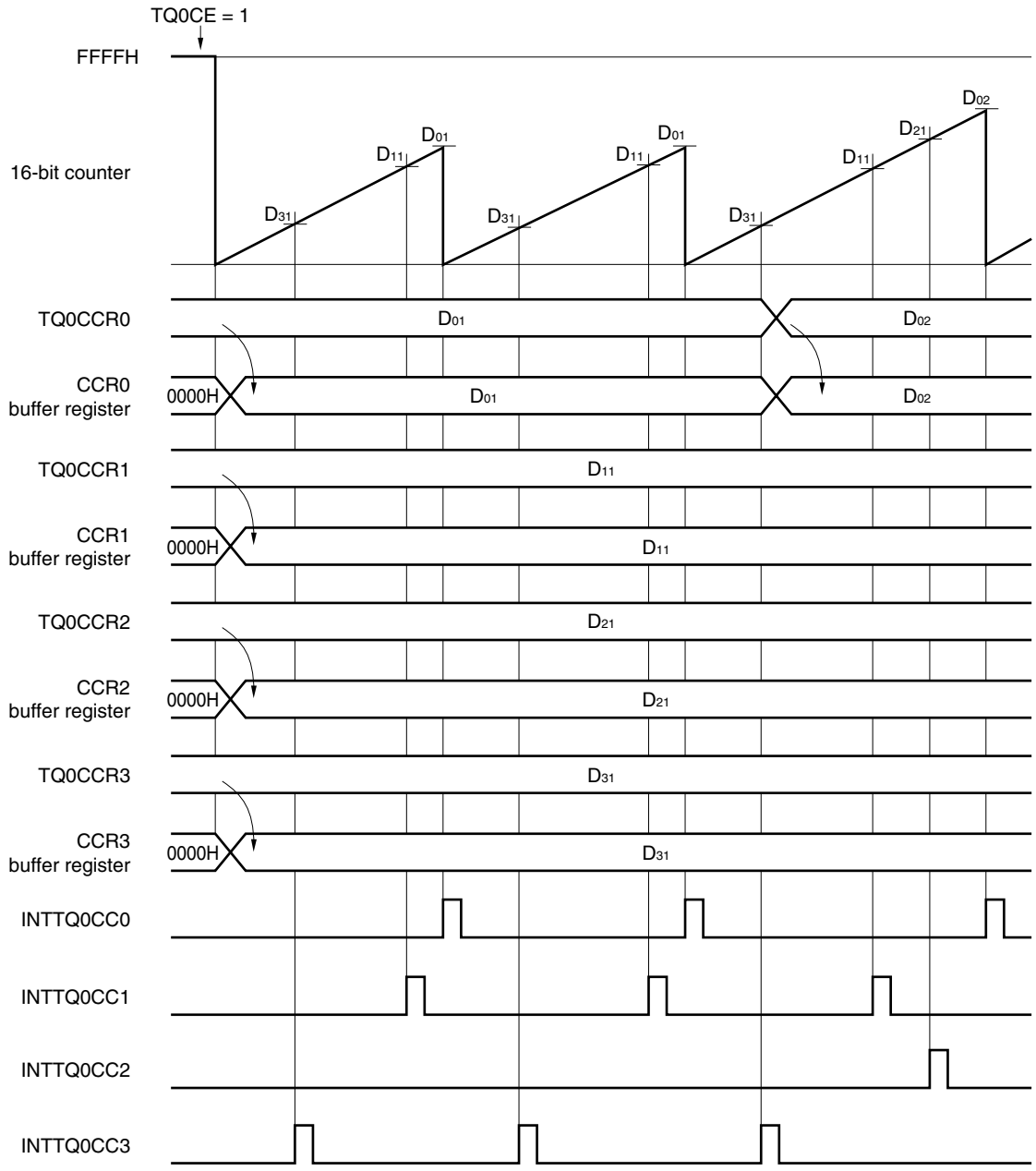
Figure 8-6. Flowchart of Basic Operation in Interval Timer Mode



Note The 16-bit counter is not cleared upon a match between the 16-bit counter and TQ0CCRm.

Remark n = 0 to 3
m = 1 to 3

Figure 8-7. Basic Operation Timing in Interval Timer Mode (1/2)

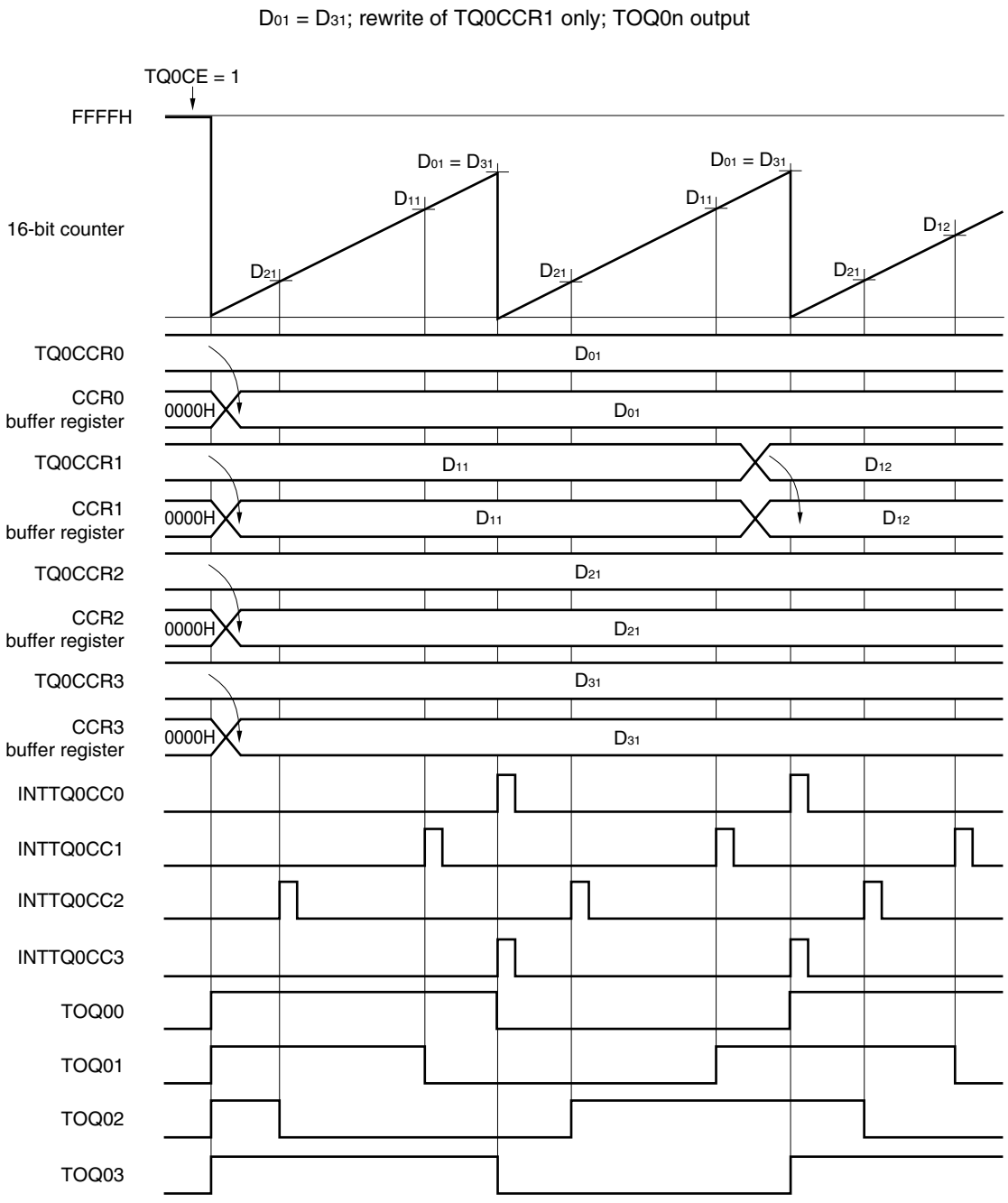


Remarks 1. D_{01}, D_{02}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{11}: Setting value of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 3

Figure 8-7. Basic Operation Timing in Interval Timer Mode (2/2)



Remarks 1. D_{01}: Setting value of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 3

8.5.3 External event count mode (TQ0MD2 to TQ0MD0 = 001)

In the external event count mode, external event count input (TIQ00 pin input) is used as a count-up signal. When the external event count mode is set, count-up is performed using external event count input (TIQ00 pin input), regardless of the setting of the TQ0EEE bit of the TQ0CTL0 register.

In the external event count mode, a match interrupt request (INTTQ0CC0) is output upon a match between the setting value of the TQ0CCR0 register (value of CCR0 buffer register) and the value of the 16-bit counter, and the 16-bit counter is cleared.

When a value is set to the TQ0CCRn register with a write instruction from the CPU, it is transferred to the CCRn buffer register through anytime write, and is used as the value for comparison with the 16-bit counter value.

In the external event count mode, the 16-bit counter is cleared only upon a match between the value of the 16-bit counter and the value of the CCR0 buffer register.

16-bit counter clearing using the TQ0CCRm register is not performed.

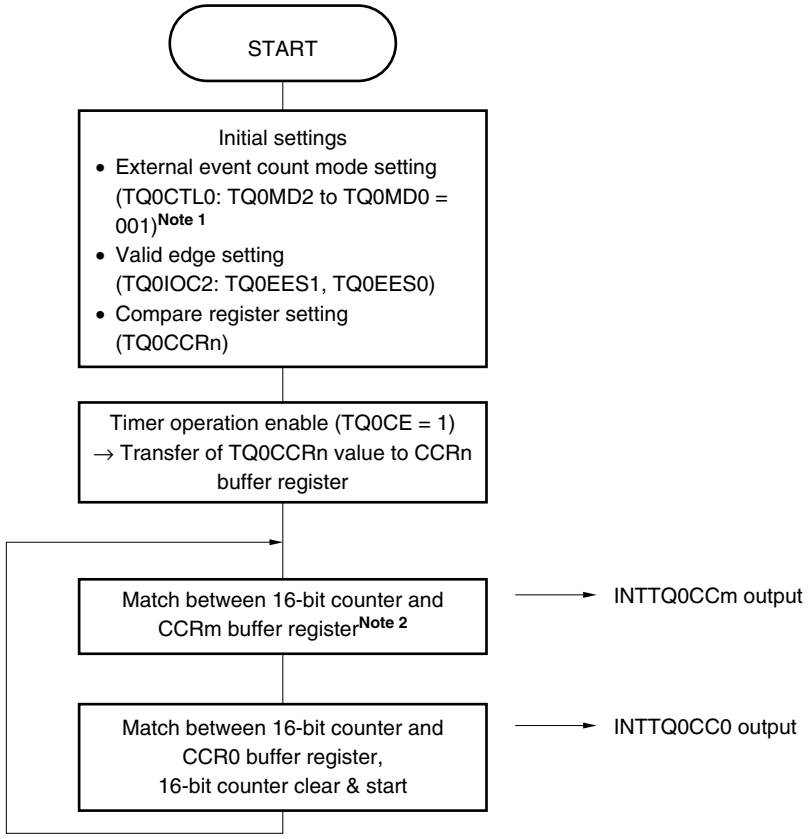
However, the setting value of the TQ0CCRm register is transferred to the CCRm buffer register and compared with the value of the 16-bit counter, and an interrupt request (INTTQ0CCm) is output if these values match.

Moreover, TOQ0n pin output is also possible by setting the TQ0OEn bit to 1.

The TQ0CCR0 register can be rewritten when TQ0CE = 1. When the TQ0CCRm register is not used, it is recommended to set FFFFH as the setting value for the TQ0CCRm register.

Remark n = 0 to 3
m = 1 to 3

Figure 8-8. Flowchart of Basic Operation in External Event Count Mode



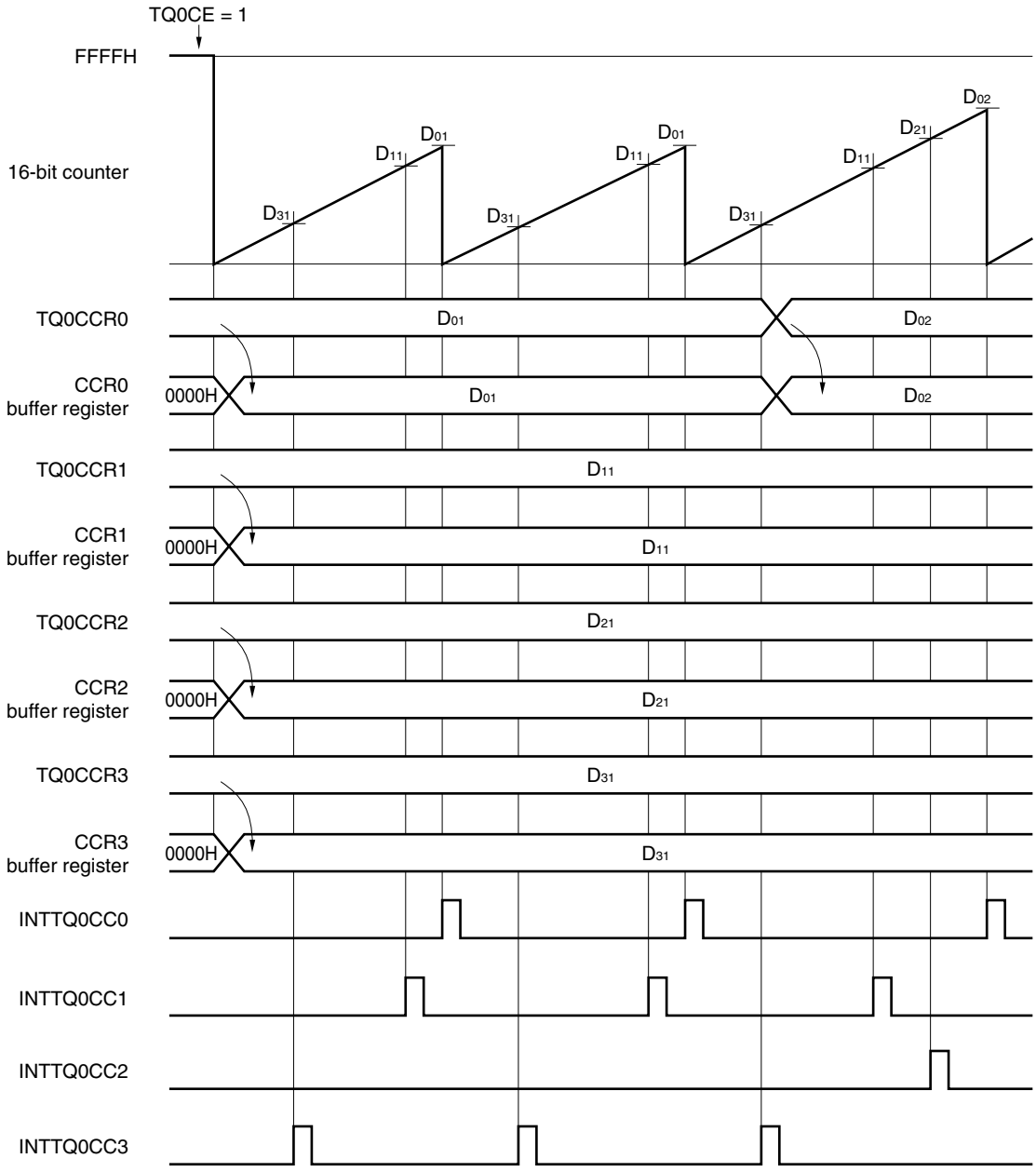
Notes 1. Selection of the TQ0EEE bit has no influence.

2. The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCRm buffer register.

Remark n = 0 to 3
m = 1 to 3

Figure 8-9. Basic Operation Timing in External Event Count Mode (1/2)

(a) Rewrite of TQ0CCR0 only; no TOQ0n output



Remarks 1. D_{01}, D_{02}: Setting values of TQ0CCR0 register (0000H to FFFFH)

D_{11}: Setting value of TQ0CCR1 register (0000H to FFFFH)

D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)

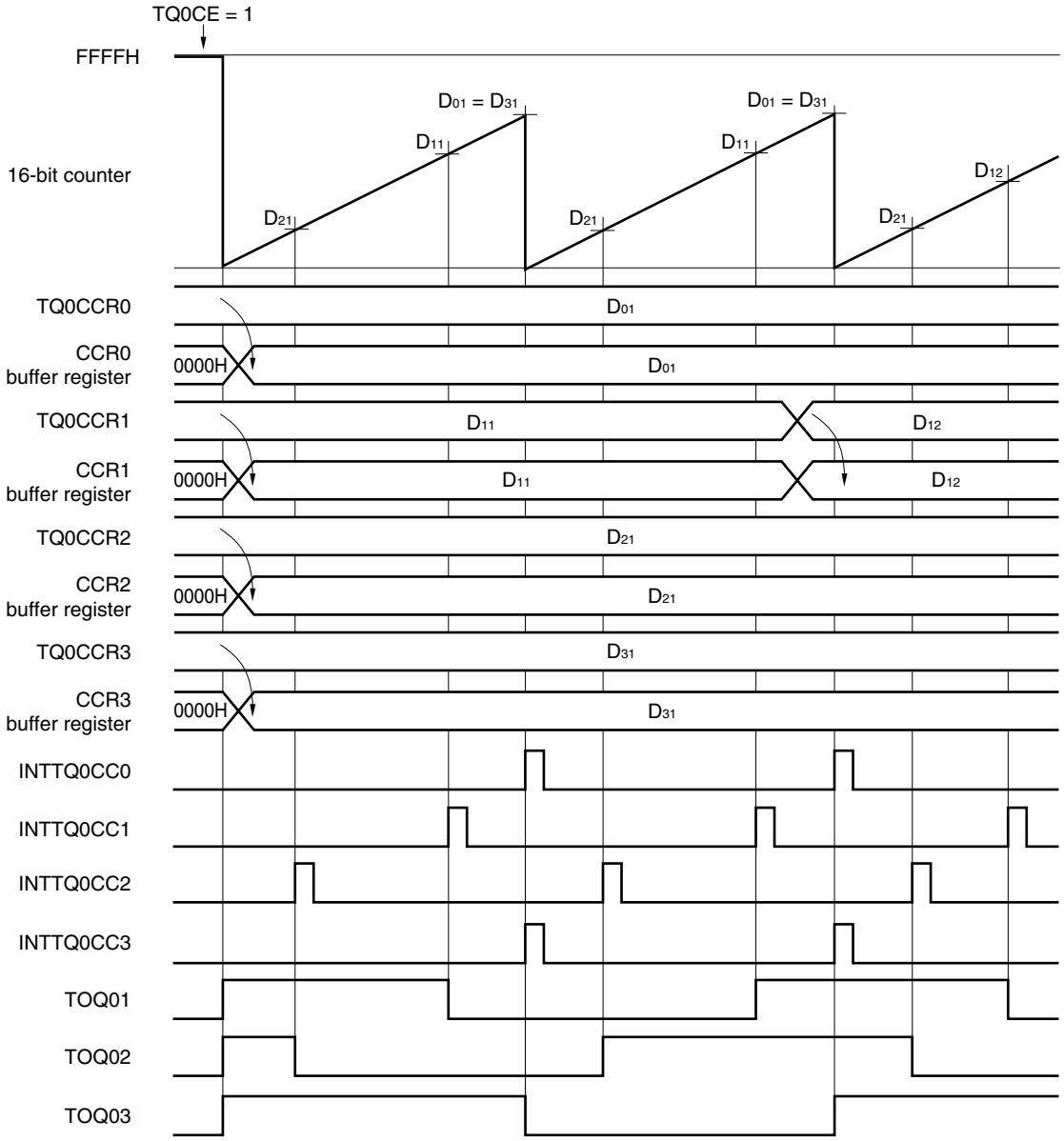
D_{31}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 3

Figure 8-9. Basic Operation Timing in External Event Count Mode (2/2)

(b) $D_{01} = D_{31}$; rewrite of TQ0CCR1 only; TOQ0n output



Remarks 1. D_{01}: Setting value of TQ0CCR0 register (0000H to FFFFH)

D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)

D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)

D_{31}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. Interval time = (Dn + 1) × (count clock cycle)

3. n = 0 to 3

8.5.4 External trigger pulse mode (TQ0MD2 to TQ0MD0 = 010)

In the external trigger pulse mode, setting TQ0CE = 1 causes external trigger input (TIQ00 pin input) wait with the 16-bit counter stopped at FFFFH. The count-up operation starts upon detection of the external trigger input (TIQ00 pin input) edge.

Regarding TOQ0m output control, the reload register (TQ0CCRm) is used as the duty setting register and the compare register (TQ0CCR0) is used as the cycle setting register.

The TQ0CCRn register can be rewritten when TQ0CE = 1.

In order for the setting value when the TQ0CCRn register is rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to the CCRn buffer register), it is necessary to rewrite TQ0CCR0 and finally write to the TQ0CCR1 register before the 16-bit counter value and the TQ0CCR0 register value match. Thereafter, the value of the TQ0CCR0 is reloaded upon a TQ0CCR0 register match.

Whether to enable or disable the next reload timing is controlled by writing to the TQ0CCR1 register. Thus even when wishing only to rewrite the value of the TQ0CCR0 register, also write the same value to the TQ0CCR1 register.

Reload is disabled even when only the TQ0CCR0 register is rewritten. To stop timer Q, set TQ0CE = 0. If the external trigger (TIQ00 pin input) edge is detected several times in the external trigger pulse mode, the 16-bit counter is cleared at the edge detection timing and count-up starts. To realize the same function (software trigger pulse mode) as external trigger pulse mode using a software trigger instead of external trigger input (TIQ00 pin input), set the TQ0EST bit of the TQ0CTL1 register to 1 so that the software trigger is output. The external trigger pulse waveform is output from TOQ0m.

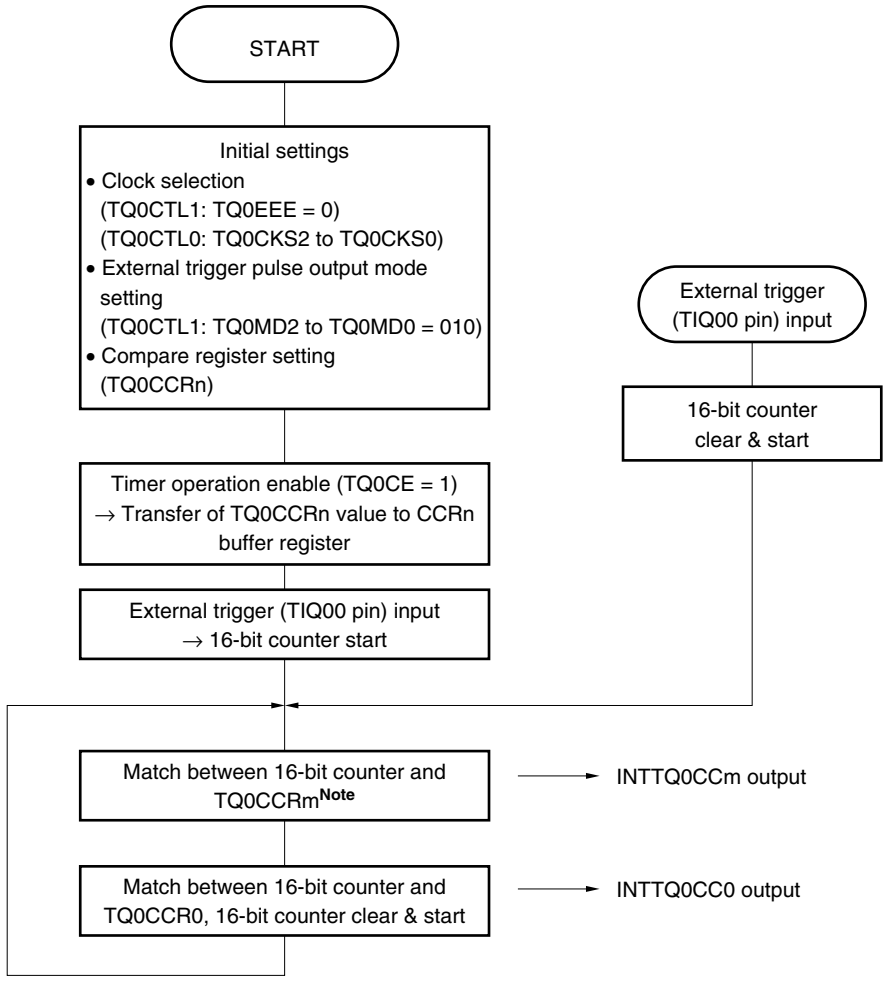
Since the TQ0CCRn register has its function fixed to that of a compare register in the external trigger pulse mode, they cannot be used for capture operation in this mode.

Caution In the external trigger pulse mode, select the internal clock (TQ0EEE bit of TQ0CTL1 register = 0) for the count clock.

Remarks 1. For the reload operation when TQ0CCRn is rewritten during timer operation, refer to **8.5.6 PWM mode**.

2. n = 0 to 3
m = 1 to 3

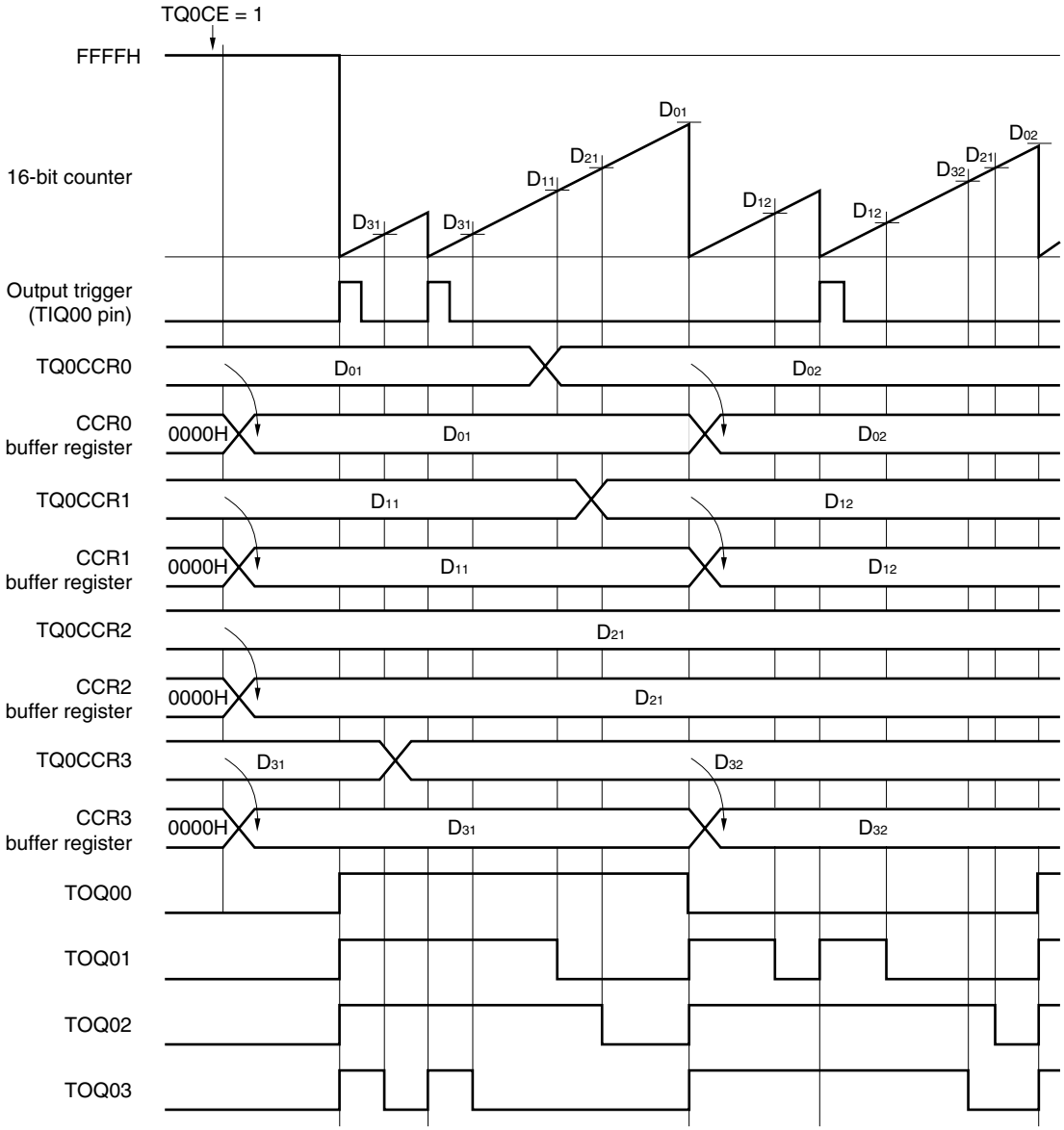
Figure 8-10. Flowchart of Basic Operation in External Trigger Pulse Output Mode



Note The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCRm buffer register.

Remark n = 0 to 3
m = 1 to 3

Figure 8-11. Basic Operation Timing in External Trigger Pulse Output Mode



Remarks 1. D_{01}, D_{02}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}, D_{32}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. TOQm output duty = (Setting value of TQ0CCRm register)/(Setting value of TQ0CCR0 register)
TOQm output cycle = (Setting value of TQ0CCR0 register) × (Count clock cycle)

3. m = 1 to 3

8.5.5 One-shot pulse mode (TQ0MD2 to TQ0MD0 = 011)

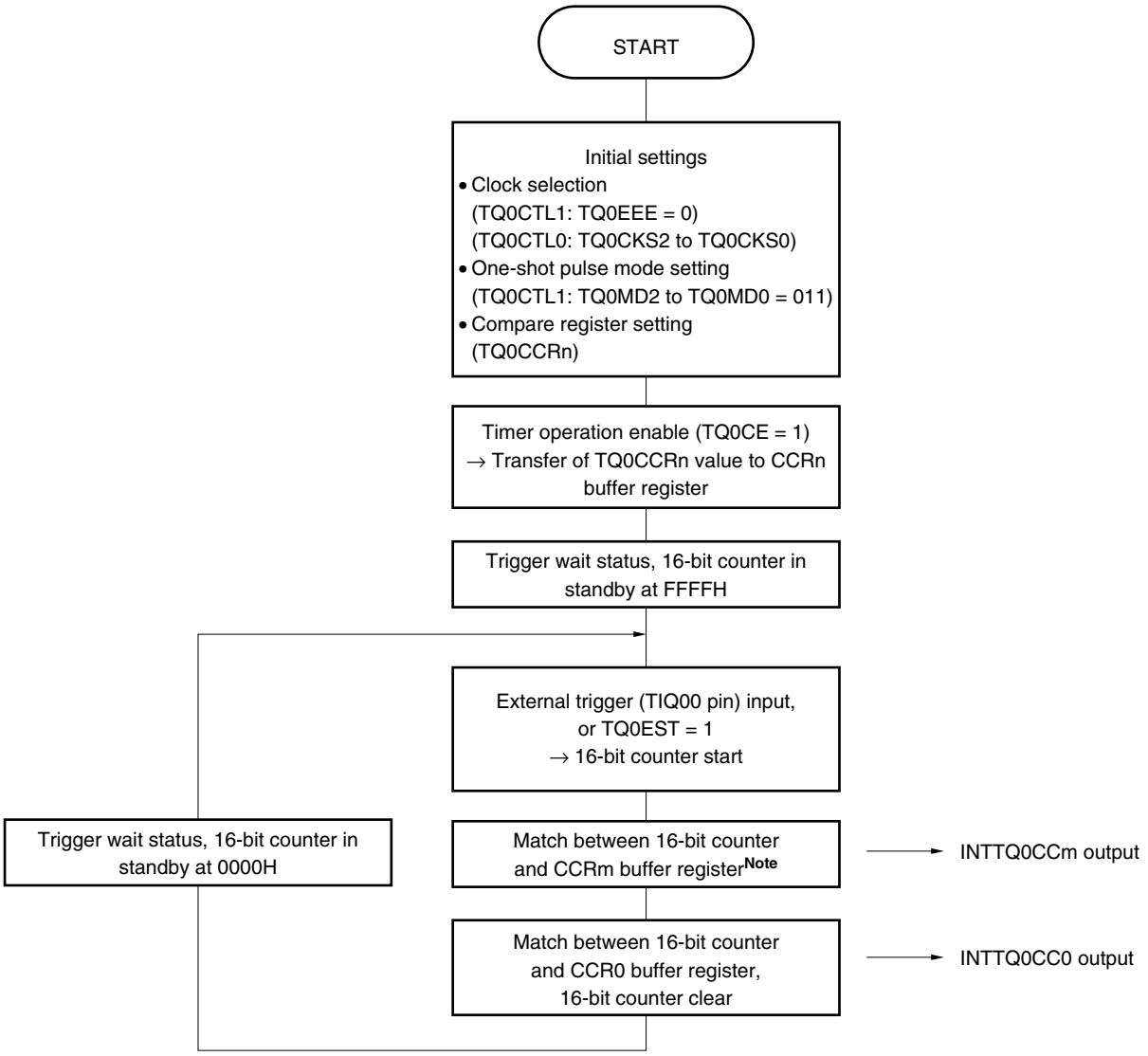
In the one-shot pulse mode, setting TQ0CE = 1 causes TQ0EST bit setting (1) or TIQ00 pin edge detection trigger wait with the 16-bit counter held at FFFFH. The 16-bit counter starts counting up upon trigger input, and upon a match between the value of the 16-bit counter and the value of the CCRm buffer register transferred from the TQ0CCRm register, TOQ0m becomes high level; Upon a match between the value of the 16-bit counter and the value of the CCR0 register transferred from the TQ0CCR0 register, TOQ0m becomes low level and the 16-bit counter is cleared to 0000H and stops. Any trigger input past the first one during 16-bit counter operation is ignored. Be sure to input the second and subsequent triggers when the 16-bit counter has stopped at 0000H. In the one-shot pulse mode, the TQ0CCRn register can be rewritten when TQ0CE = 1. The setting values rewritten to the TQ0CCRn register becomes valid following execution of a write instruction from the CPU, at which time they are transferred to the CCRn buffer register through anytime write, and become the values for comparison with the 16-bit counter value. The one-shot pulse waveform is output from the TOQ0m pin. The TOQn0 pin performs toggle output upon a match between the 16-bit counter and the TQ0CCR0 register.

Since the TQ0CCRn register has their function fixed to that of a compare register in the one-shot pulse mode, they cannot be used for capture operation in this mode.

Caution In the one-shot pulse mode, select the internal clock (TQ0EEE bit of TQ0CTL1 register = 0) for the count clock.

Remark n = 0 to 3
m = 1 to 3

Figure 8-12. Flowchart of Basic Operation in One-Shot Pulse Mode

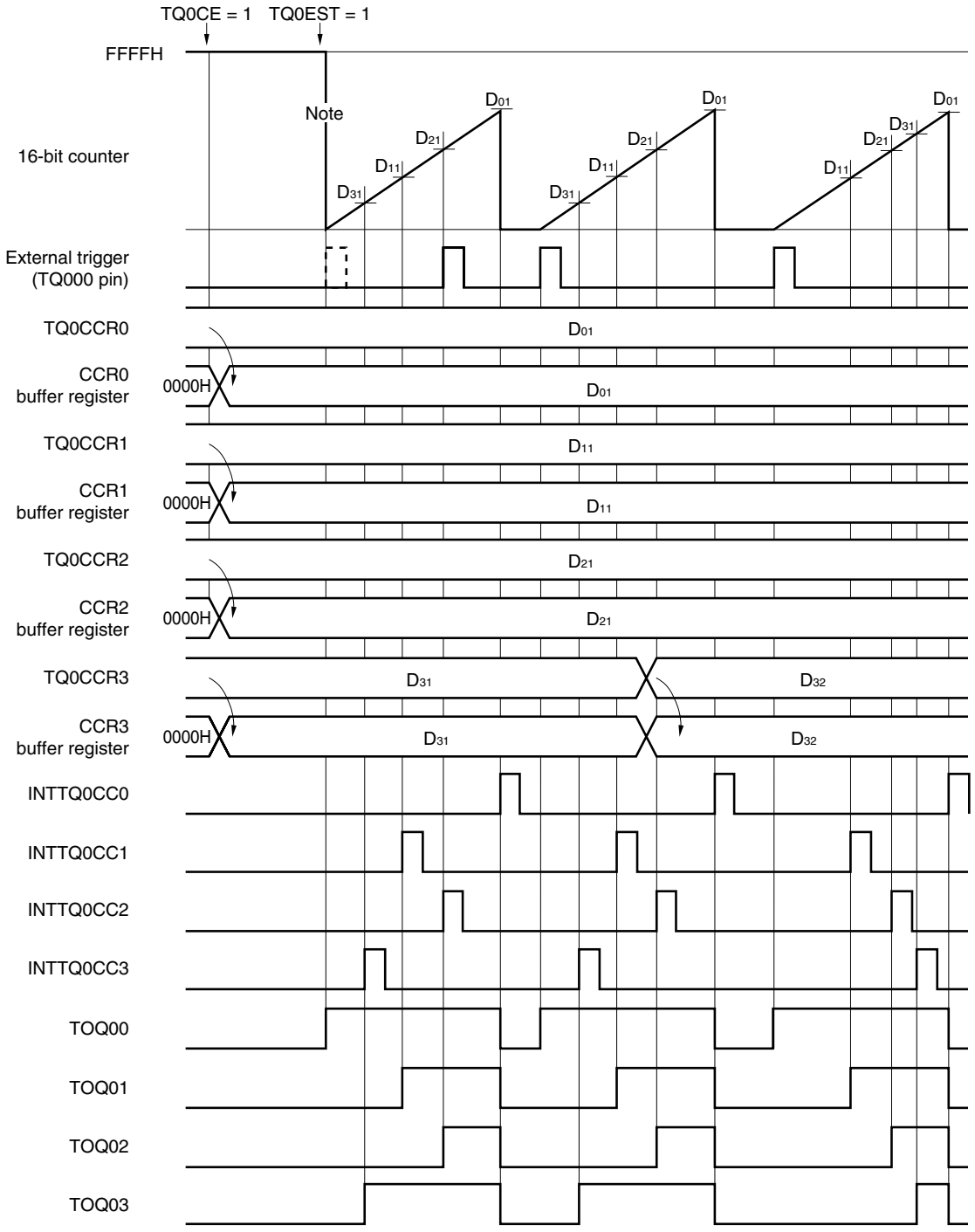


Note The 16-bit counter is not cleared upon a match between the 16-bit counter and the CCRm buffer register.

Caution The 16-bit counter is not cleared and trigger input is ignored even if trigger input is performed during the count-up operation of the 16-bit counter.

Remark n = 0 to 3
m = 1 to 3

Figure 8-13. Timing of Basic Operation in One-Shot Pulse Mode



Note The 16-bit counter starts counting up when either TQ0EST = 1 is set or external trigger (TIQ00 pin) is input.

Remark D_{01}: Setting value of TQ0CCR0 register (0000H to FFFFH)
D_{11}: Setting value of TQ0CCR1 register (0000H to FFFFH)
D_{21}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{31}, D_{32}: Setting value of TQ0CCR3 register (0000H to FFFFH)

8.5.6 PWM mode (TQ0MD2 to TQ0MD0 = 110)

In the PWM mode, TMQ0 capture/compare register m (TQ0CCRm) is used as the duty setting register and TMQ0 capture/compare register 0 (TQ0CCR0) is used as the cycle setting register.

Variable duty PWM is output by setting these two registers and operating the timer.

The TQ0CCRn register can be rewritten when TQ0CE = 1.

In order for the setting value when the TQ0CCRn register is rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be reloaded to CCRn buffer register), it is necessary to rewrite TQ0CCR0 and finally write to the TQ0CCR1 register before the 16-bit counter value and the TQ0CCR0 register value match. Thereafter, the values of the TQ0CCRn register is reloaded upon a TQ0CCR0 register match.

Whether to enable or disable the next reload timing is controlled by writing to the TQ0CCR1 register. Thus even when wishing only to rewrite the value of the TQ0CCR0 register, also write the same value to the TQ0CCR1 register.

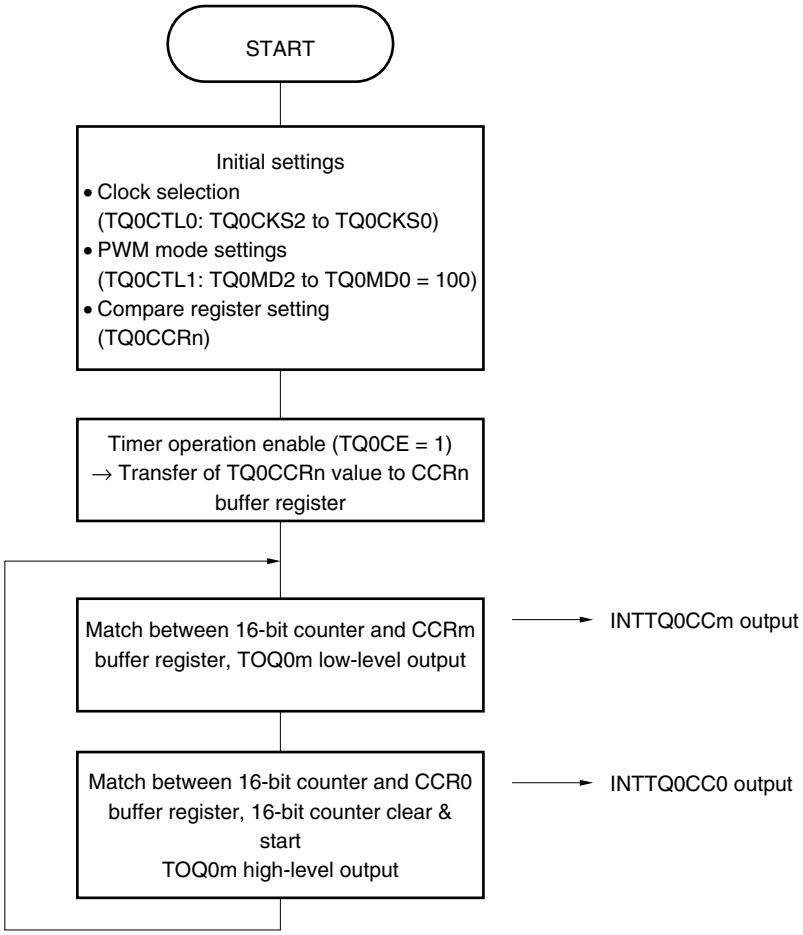
Reload is disabled even when only the TQ0CCR0 register is rewritten. To stop timer Q, set TQ0CE = 0. PWM waveform output is performed from the TOQ0m pin. The TOQ0m pin performs toggle output upon a match between the 16-bit counter and the TQ0CCR0 register.

Since the TQ0CCRn register has its function fixed that of a compare register in the PWM mode, they cannot be used for capture operation in this mode.

Remark n = 0 to 3
m = 1 to 3

Figure 8-14. Flowchart of Basic Operation in PWM Mode (1/2)

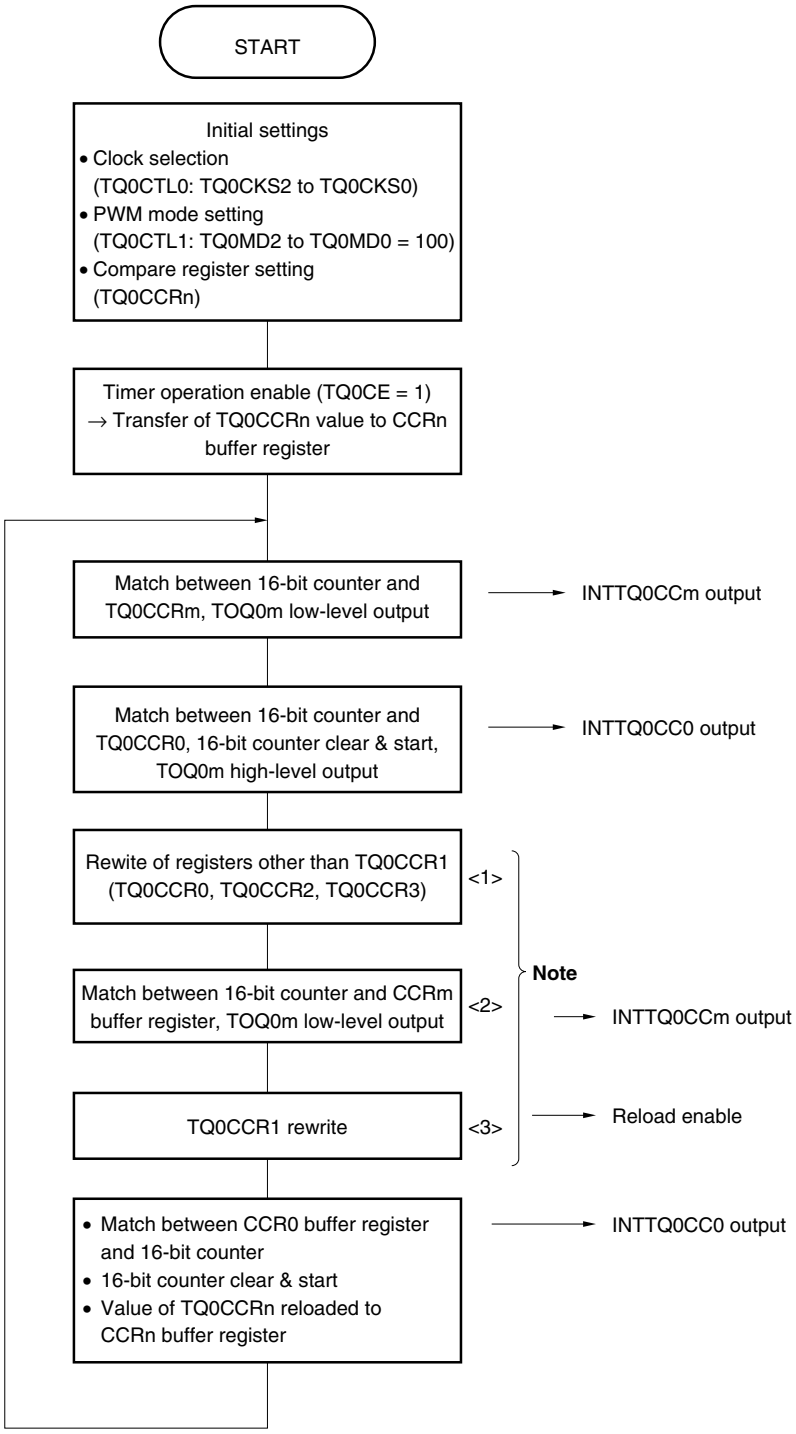
(i) Value of TQ0CCRn register not rewritten during timer operation



Remark n = 0 to 3
m = 1 to 3

Figure 8-14. Flowchart of Basic Operation in PWM Mode (2/2)

(ii) Value of TQ0CCRn register rewritten during timer operation

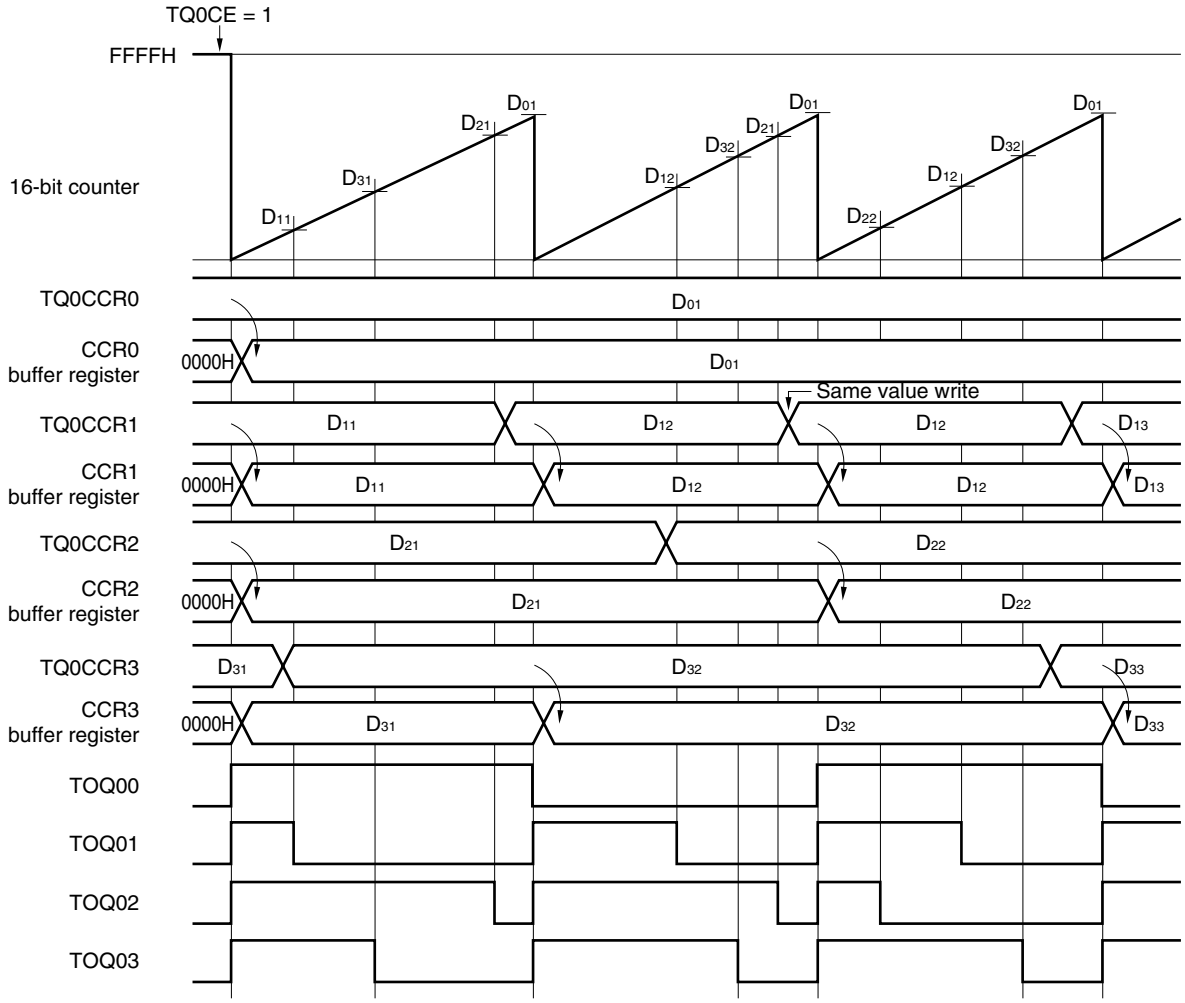


Note The timing of <2> in the above flowchart may differ depending on the rewrite timing of steps <1> and <3> and the value of TQ0CCRm, but make sure that step <3> comes after step <1>.

Remark n = 0 to 3
m = 1 to 3

Figure 8-15. Basic Operation Timing in PWM Mode (1/2)

(i) TQ0CCR1 to TQ0CCR3 values rewritten



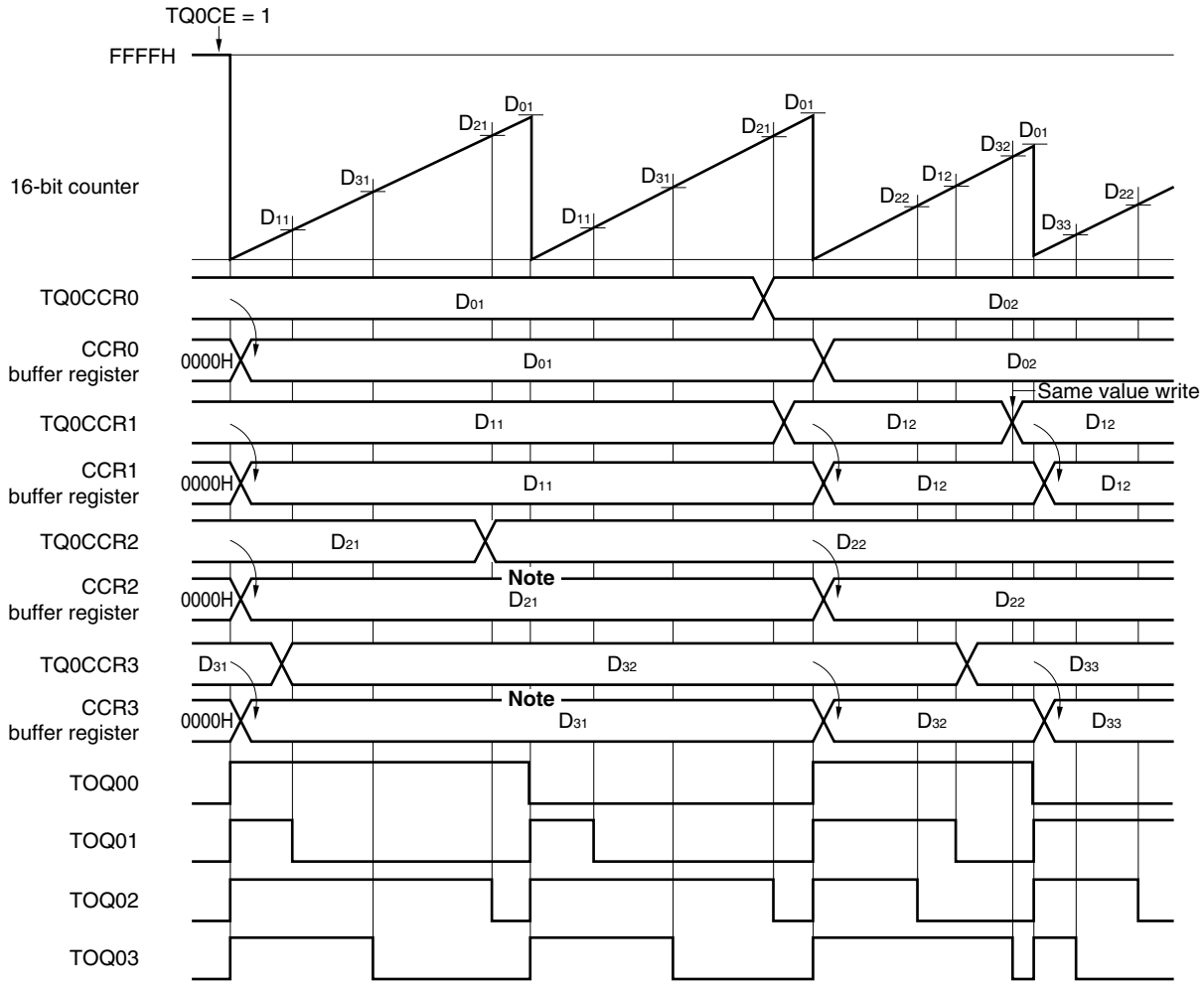
Remarks 1. D_{10}: Setting value of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}, D_{13}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}, D_{22}: Setting values of TQ0CCR2 register (0000H to FFFFH)
D_{31}, D_{32}, D_{33}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. TOQ0m output duty = (Setting value of TQ0CCRm register)/(Setting value of TQ0CCR0 register)
TOQ0m output cycle = (Setting value of TQ0CCR0 register + 1) × (Count clock cycle)
TOQ00 output toggle width = (Setting value of TQ0CCR0 register + 1) × (Count clock cycle)

3. m = 1 to 3

Figure 8-15. Basic Operation Timing in PWM Mode (2/2)

(ii) TQ0CCR0 to TQ0CCR3 values rewritten



Note Reload is not performed because the TQ0CCR1 register was not rewritten.

Remarks 1. D_{01}, D_{02}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{21}, D_{22}: Setting values of TQ0CCR2 register (0000H to FFFFH)
D_{31}, D_{32}, D_{33}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. TOQ0m output duty = (Setting value of TQ0CCRm register)/(Setting value of TQ0CCR0 register)
TOQ0m output cycle = (Setting value of TQ0CCR0 register + 1) × (Count clock cycle)
TOQ00 output toggle width = (Setting value of TQ0CCR0 register + 1) × (Count clock cycle)

3. m = 1 to 3

8.5.7 Free-running mode (TQ0MD2 to TQ0MD0 = 101)

In the free-running mode, both the interval function and the compare function can be realized by operating the 16-bit counter as a free-running counter and selecting capture/compare operation with the TQ0CCS3 to TQ0CCS0 bits of the TQ0OPT0 register.

The settings of the TQ0CCS7 to TQ0CCS0 bits of the TQ0OPT0 register are valid only in the free-running mode.

TQ0CCSn	Operation
0	Use TQ0CCRn register as compare register
1	Use TQ0CCRn register as capture register

- Using TQ0CCRn register as compare register

  An interrupt is output upon a match between the 16-bit counter and the CCRn buffer register in the free-running mode (interval function).

  Rewrite during compare timer operation is enabled and performed with anytime write. (Once the compare value has been written, synchronization with the internal clock is done and this value is used as the 16-bit counter comparison value.)

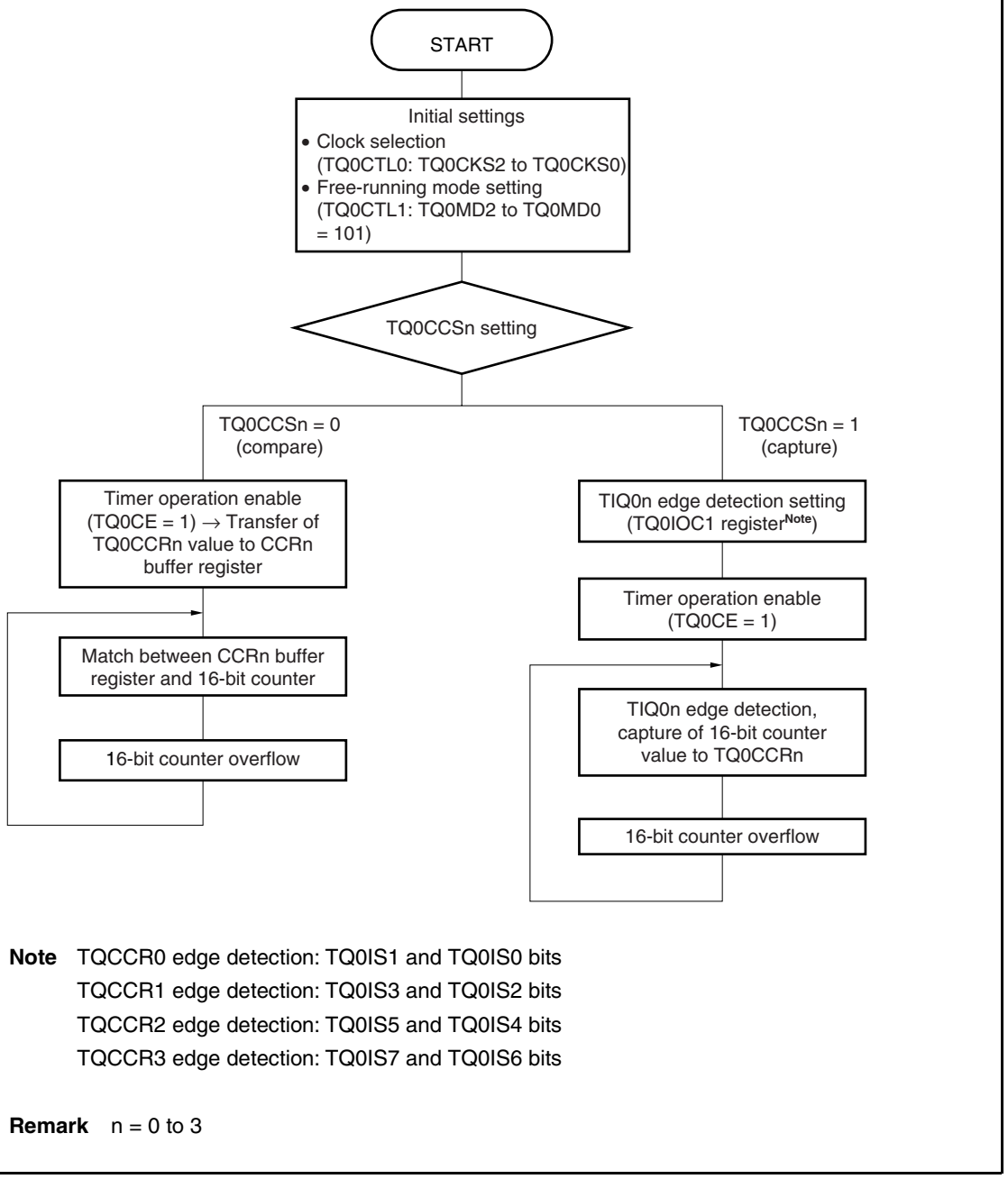
  When timer output (TOQ0n) has been enabled, TOQ0n performs toggle output upon a match between the 16-bit counter and the CCRn buffer register.

- Using TQ0CCRn register as capture register

  The value of the 16-bit counter is saved to the TQ0CCRn register upon TIQ0n pin edge detection.

Remark n = 0 to 3

Figure 8-16. Flowchart of Basic Operation in Free-Running Mode



Note TQCCR0 edge detection: TQ0IS1 and TQ0IS0 bits
TQCCR1 edge detection: TQ0IS3 and TQ0IS2 bits
TQCCR2 edge detection: TQ0IS5 and TQ0IS4 bits
TQCCR3 edge detection: TQ0IS7 and TQ0IS6 bits

Remark n = 0 to 3

(1) TQ0CCSn = 0 setting (interval function description)

When TQ0CE = 1 is set, the 16-bit counter counts from 0000H to FFFFH and the free-running count-up operation continues until TQ0CE = 0 is set. In this mode, when a value is written to the TQ0CCRn register, it is transferred to the CCRn buffer register (anytime write). In this mode, no one-shot pulse is output even when a one-shot pulse trigger is input. Moreover, when TQ0OEn = 1 is set, TOQ0n performs toggle output upon a match between the 16-bit counter and the CCRn buffer register.

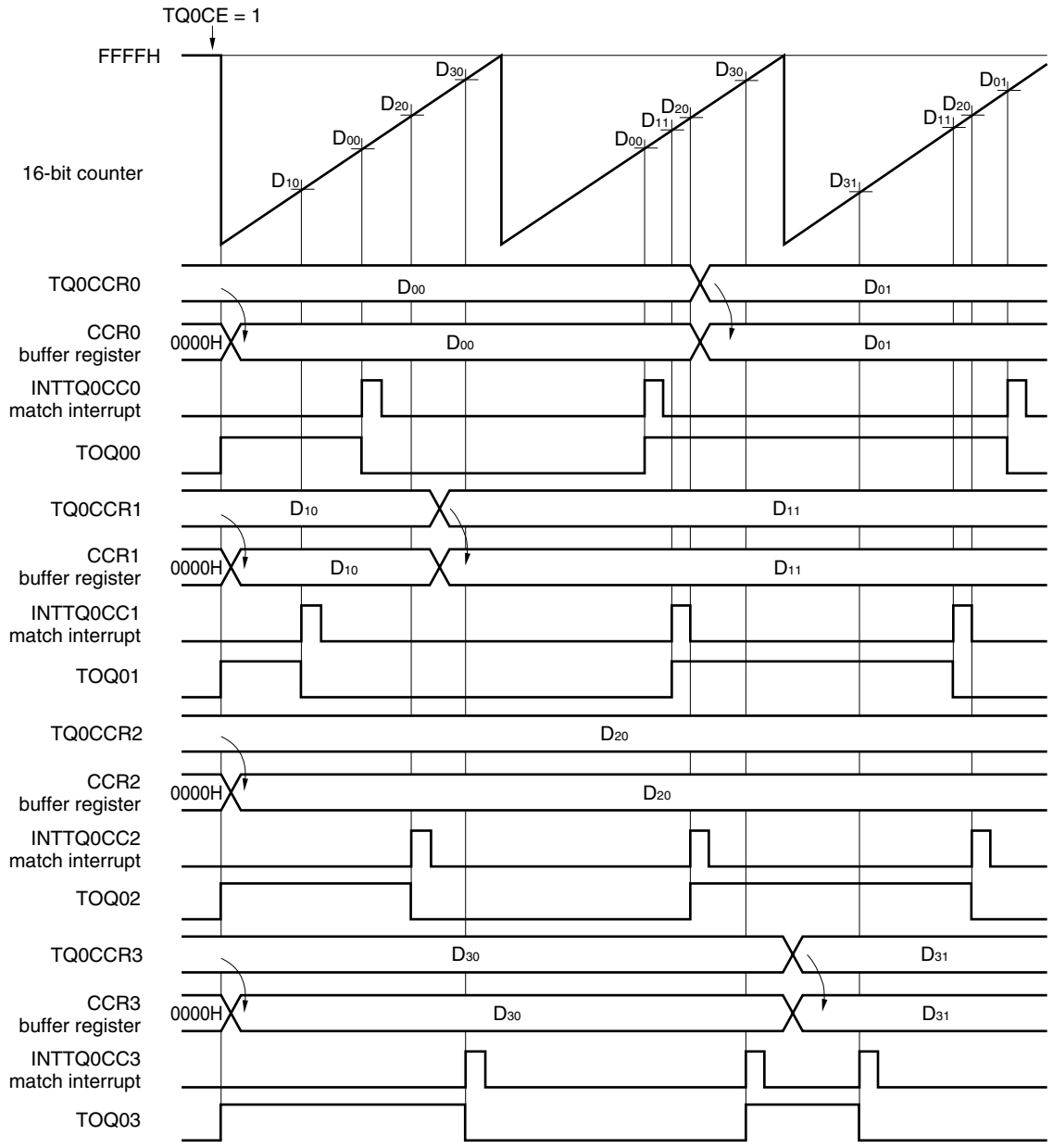
(2) TQ0CCSn = 1 setting (capture function description)

When TQ0CE = 1, the 16-bit counter counts from 0000H to FFFFH and free-running count-up operation continues until TQ0CE = 0 is set. During this time, values are captured by capture trigger operation and are written to the TQ0CCRn register.

Regarding capture in the vicinity of overflow (FFFFH), judgment is made using the overflow flag (TQ0OVF). However, if overflow occurs twice (2 or more free-running cycles), the capture trigger interval cannot be judged with the TQ0OVF flag.

Remark n = 0 to 3

Figure 8-17. Basic Operation Timing in Free-Running Mode
(TQ0CCS3 = 0, TQ0CCS2 = 0, TQ0CCS1 = 0, TQ0CCS0 = 0)



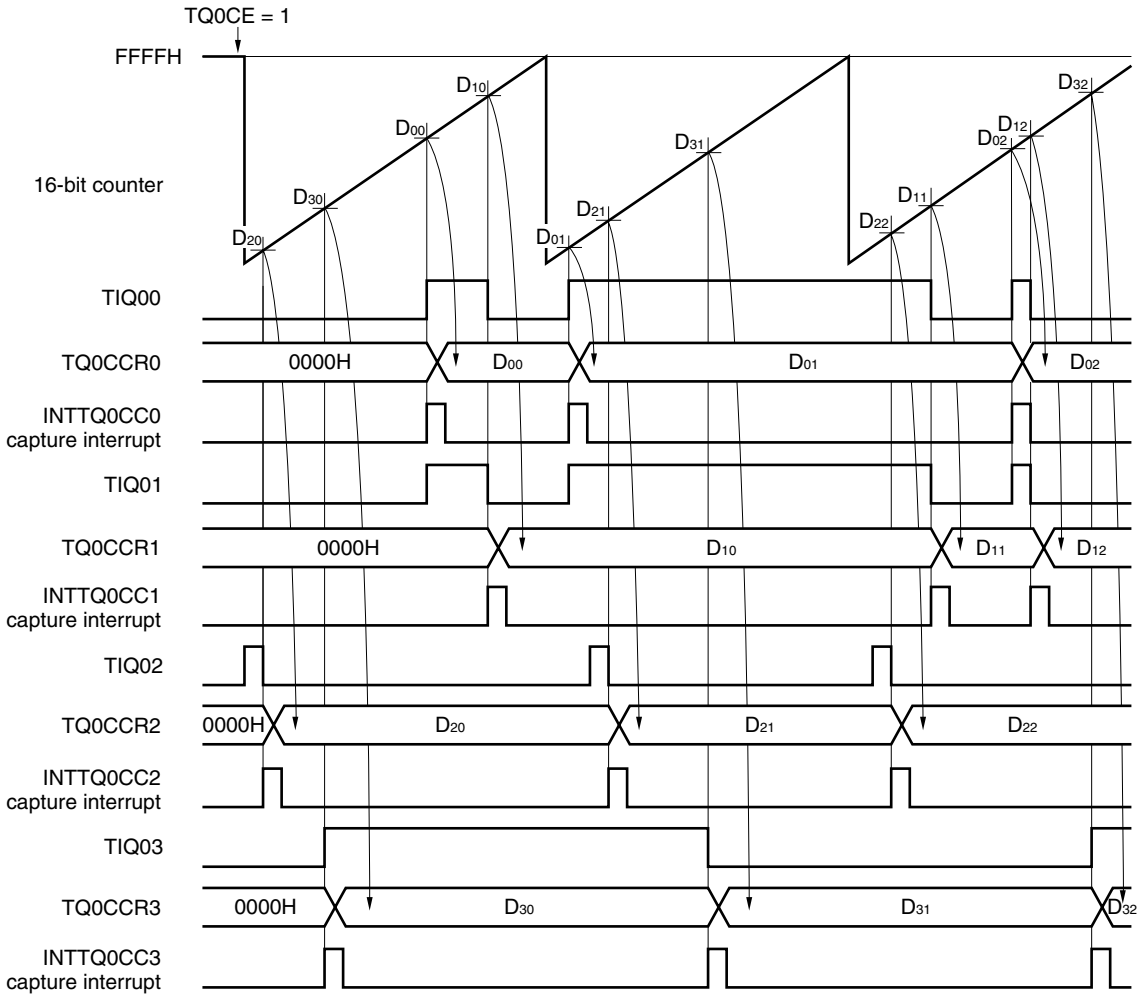
Remarks 1. D_{00}, D_{01}: Setting values of TQ0CCR0 register (0000H to FFFFH)
D_{10}, D_{11}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{20}: Setting value of TQ0CCR2 register (0000H to FFFFH)
D_{30}, D_{31}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. TOQ0n output toggle width = (Setting value of TQ0CCRn register) × (Count clock cycle)

3. TOQ0n output rises to high level when counting is started.

4. n = 0 to 3

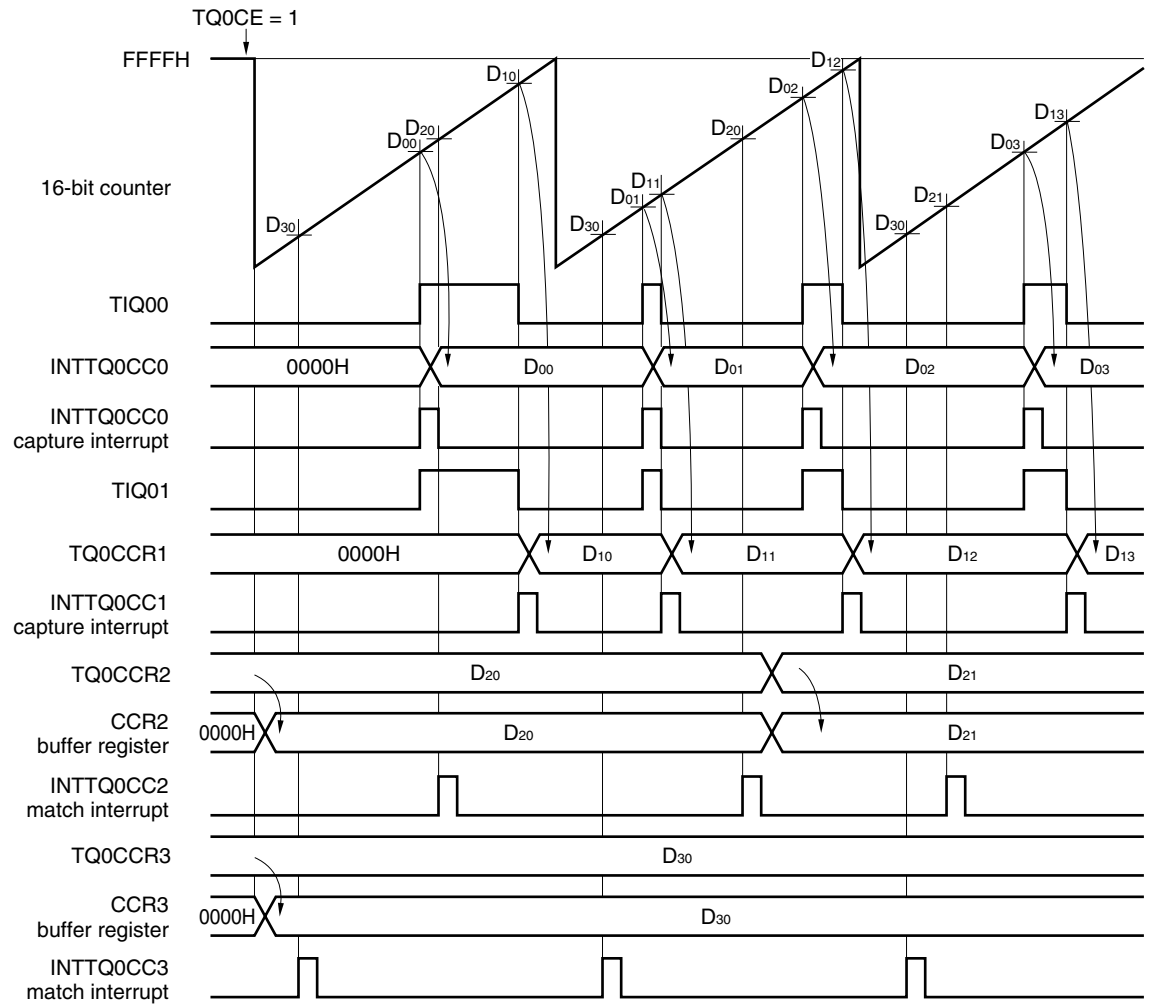
Figure 8-18. Basic Operation Timing in Free-Running Mode
(TQ0CCS3 = 1, TQ0CCS2 = 1, TQ0CCS1 = 1, TQ0CCS0 = 1)



Remarks 1. D_{00}, D_{01}, D_{02}: Values captured to TQ0CCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Values captured to TQ0CCR1 register (0000H to FFFFH)
D_{20}, D_{21}, D_{22}: Values captured to TQ0CCR2 register (0000H to FFFFH)
D_{30}, D_{31}, D_{32}: Values captured to TQ0CCR3 register (0000H to FFFFH)

2. TIQ00: Set to rising edge detection (TQ0IS1, TQ0IS0 = 01)
TIQ01: Set to falling edge detection (TQ0IS3, TQ0IS2 = 10)
TIQ02: Set to falling edge detection (TQ0IS5, TQ0IS4 = 10)
TIQ03: Set to detection of both rising and falling edges (TQ0IS7, TQ0IS6 = 11)

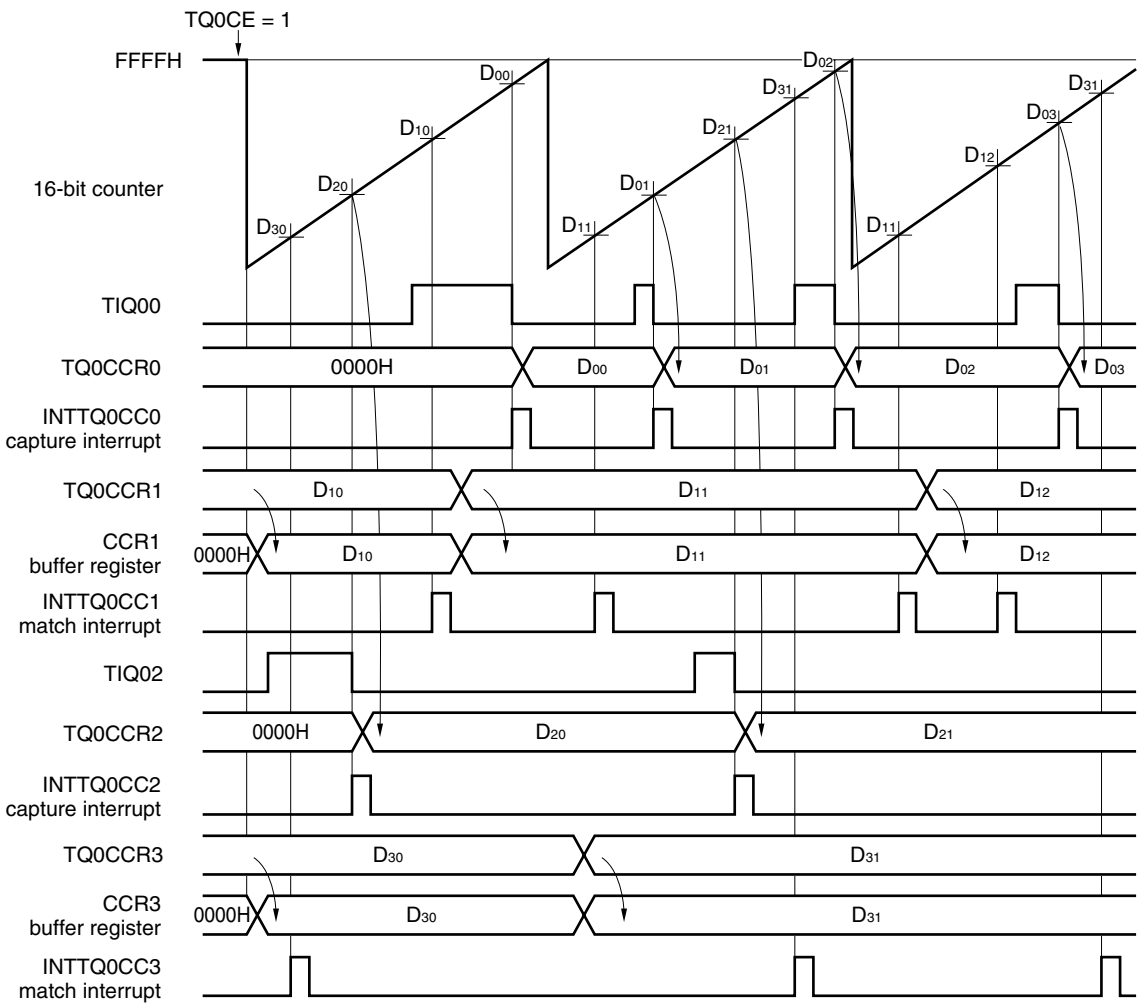
Figure 8-19. Basic Operation Timing in Free-Running Mode
(TQ0CCS3 = 1, TQ0CCS2 = 1, TQ0CCS1 = 1, TQ0CCS0 = 0)



Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Values captured to TQ0CCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Values captured to TQ0CCR1 register (0000H to FFFFH)
D_{20}, D_{21}: Setting values of TQ0CCR2 register (0000H to FFFFH)
D_{30}: Setting value of TQ0CCR3 register (0000H to FFFFH)

2. TIQ00: Set to rising edge detection (TQ0IS1, TQ0IS0 = 01)
TIQ01: Set to falling edge detection (TQ0IS3, TQ0IS2 = 10)

Figure 8-20. Basic Operation Timing in Free-Running Mode
(TQ0CCS3 = 1, TQ0CCS2 = 0, TQ0CCS1 = 0, TQ0CCS0 = 1)



Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Values captured to TQ0CCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Setting values of TQ0CCR1 register (0000H to FFFFH)
D_{20}, D_{21}: Values captured to TQ0CCR2 register (0000H to FFFFH)
D_{30}, D_{31}: Setting values of TQ0CCR3 register (0000H to FFFFH)

2. TIQ00: Set to falling edge detection (TQ0IS1, TQ0IS0 = 10)
TIQ02: Set to falling edge detection (TQ0IS5, TQ0IS4 = 10)

(c) Overflow flag

When the counter overflows from FFFFH to 0000H in the free-running mode, the overflow flag (TQ0OVF) is set to 1 and an overflow interrupt (INTTQ0OV) is output.

The overflow flag is cleared by writing 0 from the CPU.

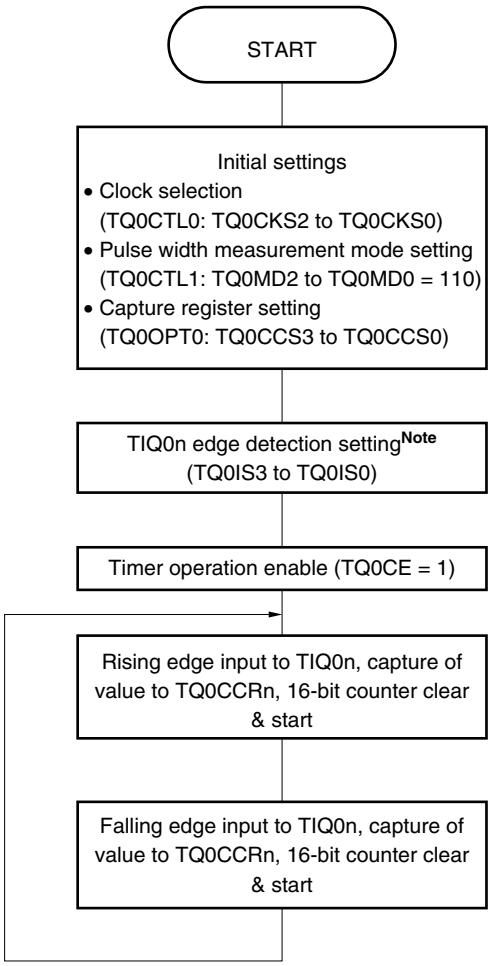
8.5.8 Pulse width measurement mode (TQ0MD2 to TQ0MD0 = 110)

In the pulse width measurement mode, free-running count is performed, and upon detection of both the rising and falling edges of TIQ00, the 16-bit counter value is saved to capture register n (TQ0CCRn) and the 16-bit counter is cleared to 0000H. The external input pulse width can be measured as a result.

However, when measuring a large pulse width that exceeds 16-bit counter overflow, perform judgment with the overflow flag. Since measurement of pulses for which overflow occurs twice or more is not possible, adjust the operating frequency of the 16-bit counter.

Caution In the pulse width measurement mode, select the internal clock (TQ0EEE of TQ0CTL1 register = 0).

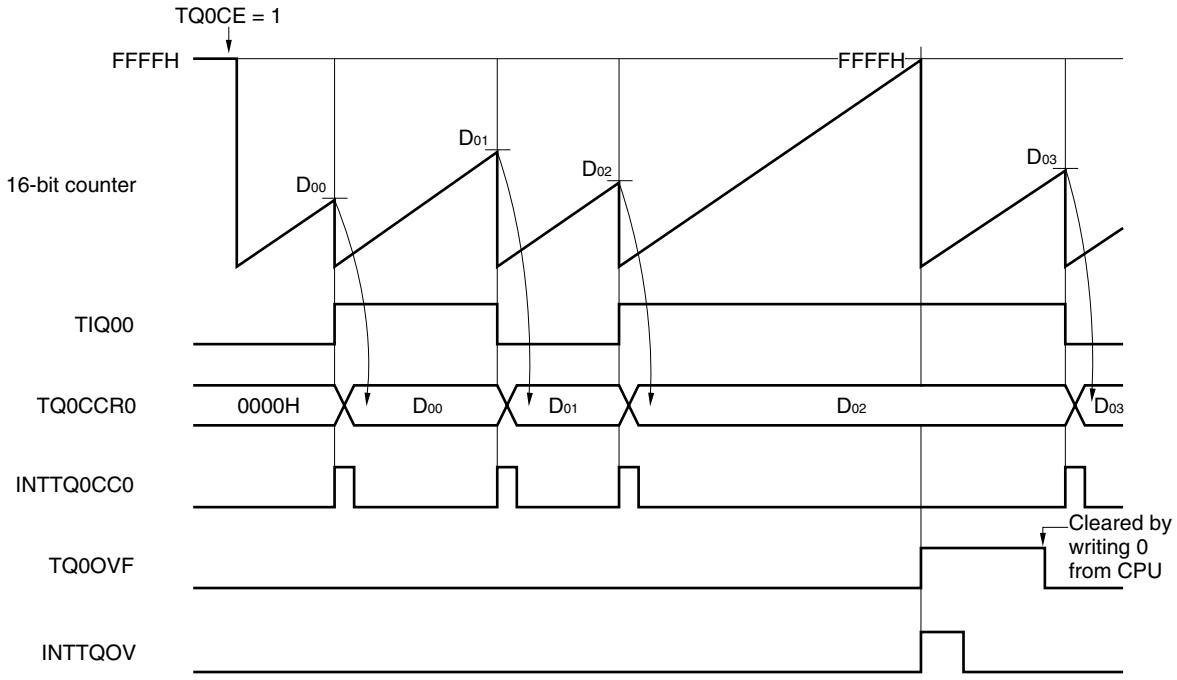
Figure 8-21. Flowchart of Basic Operation in Pulse Width Measurement Mode



Note External pulse input is possible any for TIQ00 to TIQ03, but only one can be selected.
Specify “both the rising and the falling edges” for edge detection. Specify the edge of the external input pulse that is not used as “No edge to be detected”.

Remark n = 0 to 3

Figure 8-22. Basic Operation Timing in Pulse Width Measurement Mode



Remarks 1. D_{00}, D_{01}, D_{02}, D_{03}: Values captured to TQ0CCR0 register (0000H to FFFFH)

2. TIQ00: Set to detection of both rising and falling edges

CHAPTER 9 16-BIT INTERVAL TIMER M

9.1 Outline

- Interval function
- 8 clocks selectable
- Simple counter × 1
  (The simple counter is a counter that does not use a counter read buffer and cannot be read during timer count operation.)
- Simple compare × 1
  (Simple compare is a type of compare that does not use a compare write buffer and the compare register cannot be written during timer counter operation.)
- Compare match interrupt × 1

Timer M supports only the clear & start mode. The free-running mode is not supported. For operation equivalent to that in the free-running mode, set the compare register to FFFFH to start the 16-bit counter in order to realize a function that uses the match interrupt as the overflow timing.

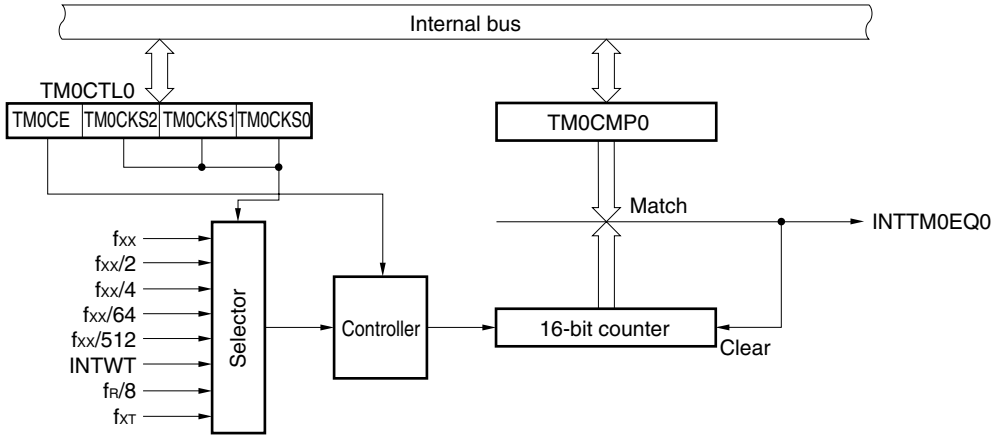
9.2 Configuration

TMM includes the following hardware.

Table 9-1. Configuration of TMM

Item	Configuration
Timer register	16-bit counter
Register	TMM compare register 0 (TM0CMP0)
Control register	TMM0 control register (TM0CTL0)

Figure 9-1. Block Diagram of TMM



Remark f_{XX}: Internal system clock frequency
f_R: Ring-OSC clock frequency
f_{XT}: Subclock frequency

(1) TMM0 compare register 0 (TM0CMP0)

The TM0CMP0 register is a 16-bit compare register.

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

Reset input clears this register to 0000H.

The same value can always be written to the TM0CMP0 register by software.

TM0CMP0 register rewrite is prohibited when the TM0CE bit = 1.

After reset: 0000H R/W Address: FFFFF694H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TM0CMP0																

9.3 Control Register

(1) TMM0 control register (TM0CTL0)

The TM0CTL0 register is an 8-bit register that controls the timer operation.

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

Reset input clears this register to 00H.

The same value can always be written to the TM0CTL0 register by software.

After reset: 00H R/W Address: FFFFF690H

	<7>	6	5	4	3	2	1	0
TM0CTL0	TM0CE	0	0	0	0	TM0CKS2	TM0CKS1	TM0CKS0

TM0CE	Internal clock operation enable/disable specification
0	Internal clock operation disabled (TMM reset asynchronously)
1	Internal clock operation enabled

The internal clock control and internal circuit reset for timer M are performed asynchronously with the TM0CE bit. When the TM0CE bit is cleared to 0, the internal clock of timer M is disabled (fixed to low level) and is reset asynchronously to the TMM latch.
When the TM0CE bit is set to 1, the internal clock is enabled after 2 input clocks have been input after 1 is written to the TM0CE bit, and the count-up operation starts.

TM0CKS2	TM0CKS1	TM0CKS0	Internal count clock selection
0	0	0	f_{XX}
0	0	1	$f_{XX}/2$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/64$
1	0	0	$f_{XX}/512$
1	0	1	INTWT
1	1	0	$f_R/8$
1	1	1	f_{XT}

Caution The TM0CKS2 to TM0CKS0 bits can be rewritten when TM0CE = 0. However, the TM0CKS2 to TM0CKS0 and TM0CE bits are mapped to the same register. Therefore, when changing the value of TM0CE from 0 to 1, it is also possible to change the value of bits TM0CKS2 to TM0CKS0.

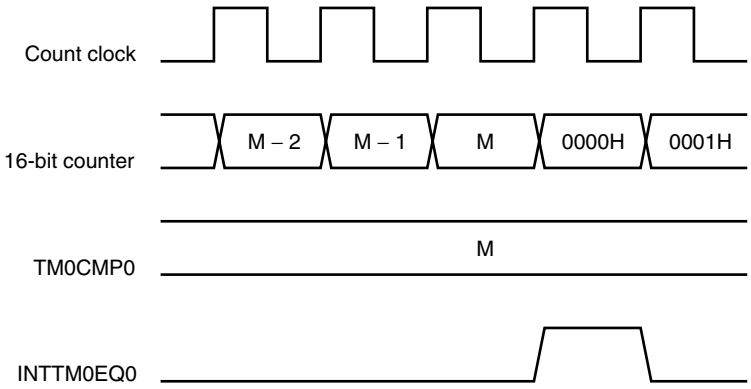
Remark f_{XX}: Internal system clock frequency
f_R: Ring-OSC frequency
f_{XT}: Subclock frequency

9.4 Operation

9.4.1 Interval timer mode

In the interval timer mode, a match interrupt signal (INTTM0EQ0) is output when the value of TMM0 compare register 0 (TM0CMP0) and the 16-bit counter value match, and at the same timing the counter is cleared to 0000H and the count-up operation is started again.

The free-running mode can also be used when the TM0CMP0 register is set to FFFFH.



Caution To set the interval time to M clocks, set M – 1 to the TM0CMP0 register.

9.4.2 Clock generator and clock enable timing

Since the value of the TM0CE bit is changed from 0 to 1 and the second clock becomes the initial pulse of the timer count-up signal, a miscount of 1 clock occurs.

CHAPTER 10 REAL-TIME OUTPUT FUNCTION (RTO)

10.1 Function

The real-time output function transfers preset data to real-time output buffer registers 0L and 0H (RTBL0 and RTBH0), and then transfers this data by hardware to an external device via the output latches, upon occurrence of an external interrupt or external trigger. The pins through which the data is output to an external device constitute a port called a real-time output function (RTO).

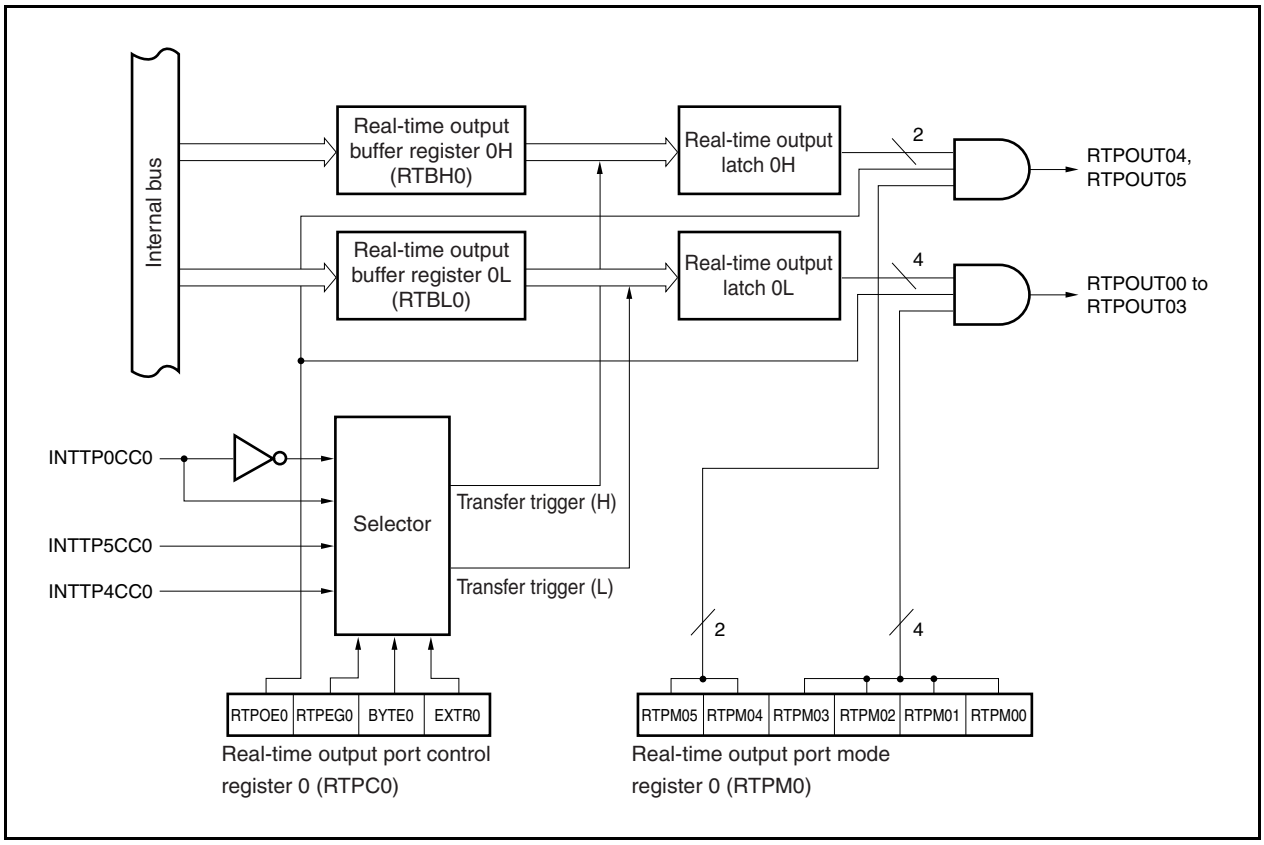
Because RTO can output signals without jitter, it is suitable for controlling a stepper motor.

In the V850ES/SG2, one 6-bit real-time output port channel is provided.

The real-time output port can be set to the port mode or real-time output port mode in 1-bit units.

The block diagram of RTO is shown below.

Figure 10-1. Block Diagram of RTO



10.2 Configuration

RTO consists of the following hardware.

Table 10-1. Configuration of RTO

Item	Configuration
Registers	Real-time output buffer registers 0L, 0H (RTBL0, RTBH0)
Control registers	Real-time output port mode register 0 (RTPM0) Real-time output port control register 0 (RTPC0)

(1) Real-time output buffer registers 0L, 0H (RTBL0, RTBH0)

The RTBL0 and RTBH0 registers are 4-bit registers that hold output data in advance.

These registers are mapped to independent addresses in the peripheral I/O register area.

These registers can be read or written in 8-bit or 1-bit units.

Reset input clears these registers to 00H.

If an operation mode of 4 bits × 1 channel or 2 bits × 1 channel is specified (BYTE0 bit of RTPC0 register = 0), data can be individually set to the RTBL0 and RTBH0 registers. The data of both these registers can be read at once by specifying the address of either of these registers.

If an operation mode of 6 bits × 1 channel is specified (BYTE0 bit = 1), 8-bit data can be set to both the RTBL0 and RTBH0 registers by writing the data to either of these registers. Moreover, the data of both these registers can be read at once by specifying the address of either of these registers.

Table 10-2 shows the operation when the RTBL0 and RTBH0 registers are manipulated.

After reset: 00H R/W Address: RTBL0 FFFFF6E0H, RTBH0 FFFFF6E2H

	7	6	5	4	3	2	1	0
RTBL0					RTBL03	RTBL02	RTBL01	RTBL00
RTBH0	0	0	RTBH05	RTBH04				

Caution When writing to bits 6 and 7 of the RTBH0 register, always write 0.

Table 10-2. Operation During Manipulation of Real-Time Output Buffer Register 0

Operation Mode	Register to Be Manipulated	Read		Write[Note]	
		Higher 4 Bits	Lower 4 Bits	Higher 4 Bits	Lower 4 Bits
4 bits × 1 channel, 2 bits × 1 channel	RTBL0	RTBH0	RTBL0	Invalid	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	Invalid
6 bits × 1 channel	RTBL0	RTBH0	RTBL0	RTBH0	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	RTBL0

Note After setting the real-time output port, set output data to the RTBL0 and RTBH0 registers by the time a real-time output trigger is generated.

10.3 Control Registers

RTO is controlled using the following two registers.

- Real-time output port mode register 0 (RTPM0)
- Real-time output port control register 0 (RTPC0)

(1) Real-time output port mode register 0 (RTPM0)

The RTPM0 register selects the real-time output port mode or port mode in 1-bit units.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF6E4H

	7	6	5	4	3	2	1	0
RTPM0	0	0	RTPM05	RTPM04	RTPM03	RTPM02	RTPM01	RTPM00

RTPM0	Selection of real-time output port
0	Port mode
1	Real-time output port mode

Cautions 1. By enabling the real-time output operation (RTPOE0 bit of RTPC0 register = 1), the bits specified for the real-time output port mode in the RTPOUT00 to RTPOUT05 signals perform real-time output, and the bits specified for the port mode output 0.

2. If real-time output is disabled (RTPOE0 bit = 0), the real-time output signal bits (RTPOUT00 to RTPOUT05) all output 0, regardless of the RTPM0 register setting.

(2) Real-time output port control register 0 (RTPC0)

The RTPC0 register is a register that sets the operation mode and output trigger of the real-time output port.

The relationship between the operation mode and output trigger of the real-time output port is as shown in Table 10-3.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF6E5H

	<7>	6	5	4	3	2	1	0
RTPC0	RTPOE0	RTPEG0	BYTE0	EXTR0	0	0	0	0

RTPOE0	Control of real-time output operation
0	Disables operation[Note 1]
1	Enables operation

RTPEG0	Valid edge of INTTP0CC0 signal
0	Falling edge[Note 2]
1	Rising edge

BYTE0	Specification of channel configuration for real-time output
0	4 bits × 2 channels
1	8 bits × 1 channel

Notes 1. When the real-time output operation is disabled (RTPOE0 bit = 0), all the bits of the real-time output signals (RTPOUT00 to RTPOUT05) output “0”.

2. The INTTP0CC0 signal is output for 1 clock of the count clock selected by TMP0.

Caution Set the RTPEG0, BYTE0, and EXTR0 bits only when RTPOE0 = 0.

Table 10-3. Operation Modes and Output Triggers of Real-Time Output Port

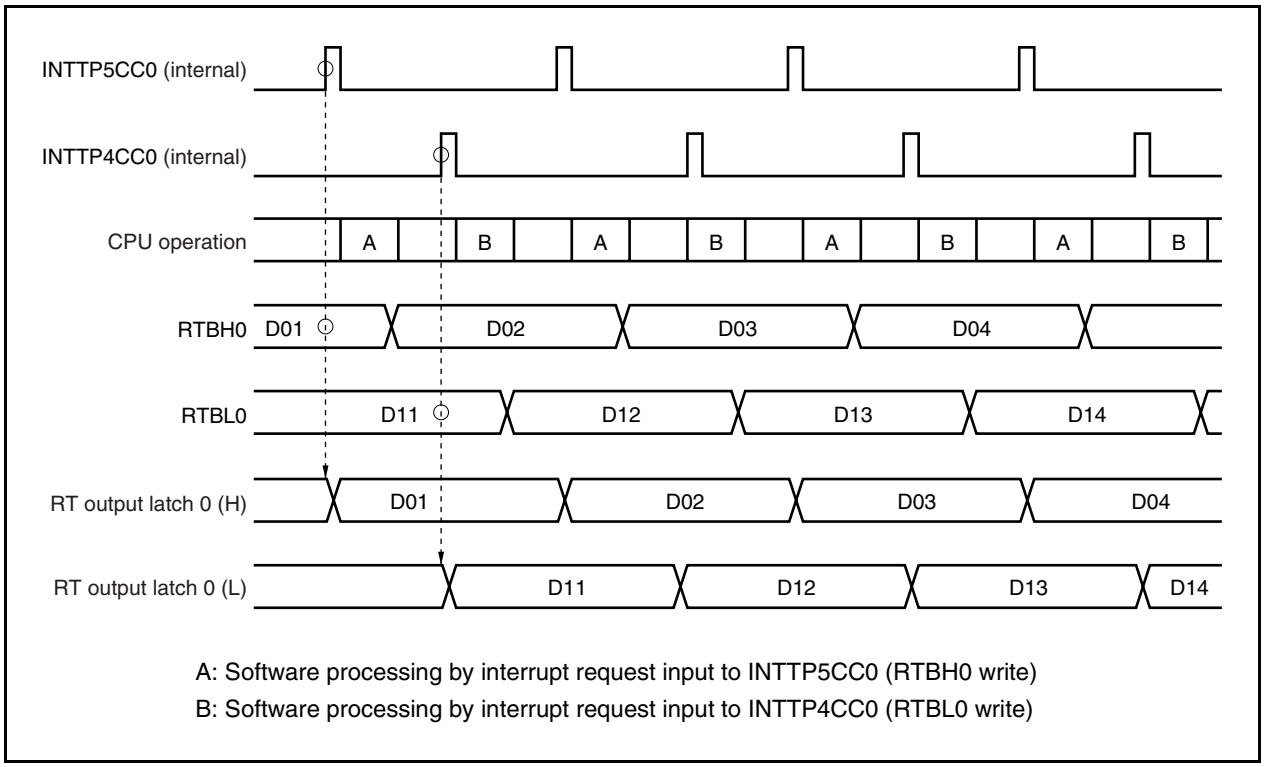
BYTE0	EXTR0	Operation Mode	RTBH0 (RTP04, RTP05)	RTBL0 (RTP00 to RTP03)
0	0	4 bits × 1 channel, 2 bits × 1 channel	INTTP5CC0	INTTP4CC0
	1		INTTP4CC0	INTTP0CC0
1	0	6 bits × 1 channel	INTTP4CC0	
	1		INTTP0CC0	

10.4 Operation

If the real-time output operation is enabled by setting the RTPOE0 bit of the RTPC0 register to 1, the data of real-time output buffer registers 0L and 0H (RTBH0 and RTBL0) is transferred to the real-time output latch in synchronization with the generation of the selected transfer trigger (set by the EXTR0 and BYTE0 bits of the RTPC0 register). Of the transferred data, only the data of the bits specified in the real-time output port mode by the RTPM0 register is output from the RTPOUT00 to RTPOUT05 bits. The bits specified in the port mode by the RTPM0 register output 0.

If the real-time output operation is disabled by clearing the RTPOE0 bit to 0, the RTPOUT00 to RTPOUT05 signals output 0 regardless of the setting of the RTPM0 register.

Figure 10-2. Example of Operation Timing of RTO0 (When EXTR0 Bit = 0, BYTE0 Bit = 0)



Remark For the operation during standby, see **CHAPTER 24 STANDBY FUNCTION**.

10.5 Usage

(1) Disable real-time output.
Clear the RTPOE0 bit of the RTPC0 register to 0.

(2) Perform initialization as follows.
- Set the alternate-function pins of port 5
  Set the PFC5.PFC5n bit and PFCE5.PFCE5n bit to 1, and then set the PMC5.PMC5n bit to 1 (n = 0 to 5).
- Specify the real-time output port mode or port mode in 1-bit units.
  Set the RTPM0 register.
- Channel configuration: Select the trigger and valid edge.
  Set the EXTR0, BYTE0, and RTPEG0 bits of the RTPC0 register.
- Set the initial values to the RTBH0 and RTBL0 registers[Note 1].

(3) Enable real-time output.
Set the RTPOE0 bit = 1.

(4) Set the next output value to the RTBH0 and RTBL0 registers by the time the selected transfer trigger is generated[Note 2].

(5) Set the next real-time output value to the RTBH0 and RTBL0 registers through interrupt servicing corresponding to the selected trigger.

Notes 1. If write to the RTBH0 and RTBL0 registers is performed when the RTPOE0 bit = 0, that value is transferred to real-time output latches 0H and 0L, respectively.

2. Even if write is performed to the RTBH0 and RTBL0 registers when the RTPOE0 bit = 1, data transfer to real-time output latches 0H and 0L is not performed.

10.6 Cautions

(1) Prevent the following conflicts by software.

- Conflict between real-time output disable/enable switching (RTPOE0 bit) and selected real-time output trigger.
- Conflict between write to the RTBH0 and RTBL0 registers in the real-time output enabled status and the selected real-time output trigger.

(2) Before performing initialization, disable real-time output (RTPOE0 bit = 0).

(3) Once real-time output has been disabled (RTPOE0 bit = 0), be sure to initialize the RTBH0 and RTBL0 registers before enabling real-time output again (RTPOE0 bit = 0 → 1).

CHAPTER 11 WATCH TIMER FUNCTIONS

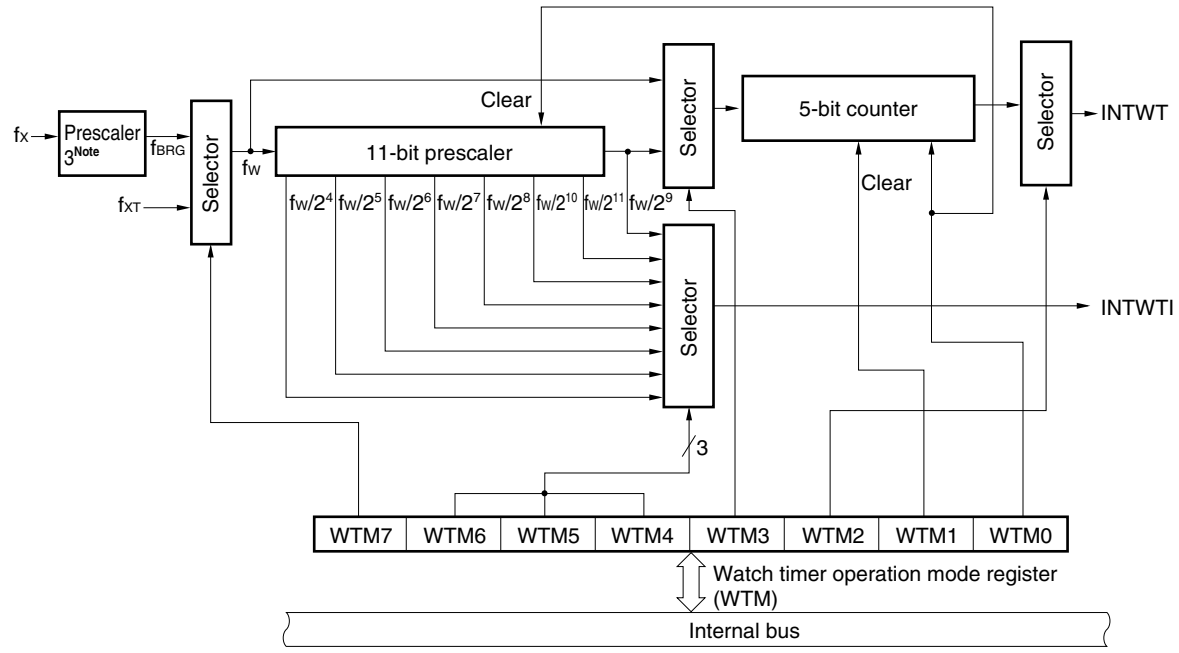
11.1 Functions

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.

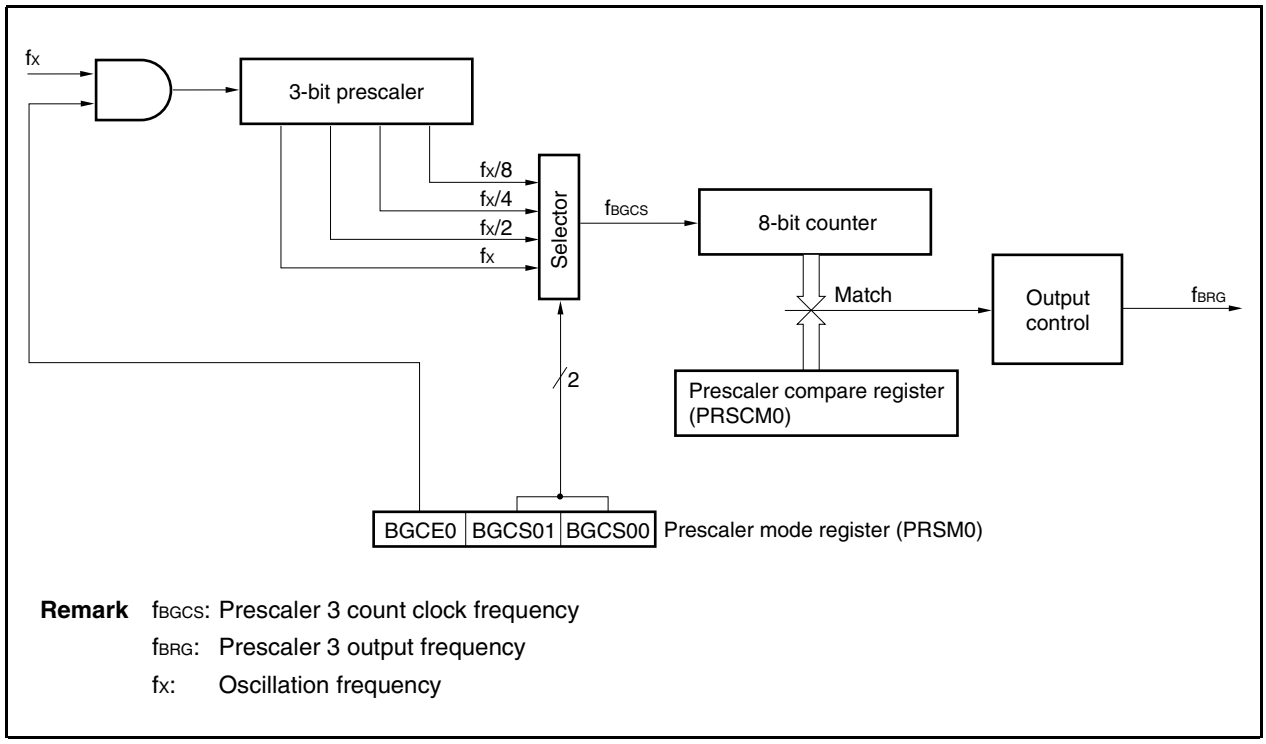
Figure 11-1. Block Diagram of Watch Timer



Note For details about prescaler 3, see **Figure 11-2 Block Diagram of Prescaler 3**.

Remark f_{BRG}: Prescaler 3 output frequency
f_X: Oscillation frequency
f_{XT}: Subclock frequency
f_W: Watch timer clock frequency
INTWT: Watch timer interrupt
INWTI: Interval timer interrupt

Figure 11-2. Block Diagram of Prescaler 3



Remark f_{BGCS}: Prescaler 3 count clock frequency
f_{BRG}: Prescaler 3 output frequency
f_X: Oscillation frequency

(1) Watch timer

The watch timer generates an interrupt request (INTWT) at time intervals of 0.5 or 0.25 seconds by using the main clock or subclock.

Caution When using a clock obtained by dividing the main clock as the watch timer count clock, set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain a divided clock frequency of 32.768 kHz.

(2) Interval timer

The watch timer generates an interrupt request (INTWTI) at time intervals specified in advance.

Table 11-1. Interval Time of Interval Timer

Interval Time	Operating at f_W = 32.768 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
$2^{11} \times 1/f_W$	62.5 ms

Remark f_W: Watch timer clock frequency

11.2 Configuration

The watch timer consists of the following hardware.

Table 11-2. Configuration of Watch Timer

Item	Configuration
Counter	5 bits × 1
Prescaler	11 bits × 1
Control register	Watch timer operation mode register (WTM)

11.3 Control Registers

the watch timer operation mode register (WTM) controls the watch timer. Before operating the watch timer, set the count clock and the interval time.

(1) Watch timer operation mode register (WTM)

The WTM 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.

This register is set by an 8-bit or 1-bit memory manipulation instruction.

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

(1/2)

After reset: 00H R/W Address: FFFFF680H

	7	6	5	4	3	2	<1>	<0>
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0

WTM7	WTM6	WTM5	WTM4	Selection of interval time of prescaler
0	0	0	0	$2^4/f_W$ (488 μs: $f_W = f_{XT}$)
0	0	0	1	$2^5/f_W$ (977 μs: $f_W = f_{XT}$)
0	0	1	0	$2^6/f_W$ (1.95 ms: $f_W = f_{XT}$)
0	0	1	1	$2^7/f_W$ (3.91 ms: $f_W = f_{XT}$)
0	1	0	0	$2^8/f_W$ (7.81 ms: $f_W = f_{XT}$)
0	1	0	1	$2^9/f_W$ (15.6 ms: $f_W = f_{XT}$)
0	1	1	0	$2^{10}/f_W$ (31.3 ms: $f_W = f_{XT}$)
0	1	1	1	$2^{11}/f_W$ (62.5 ms: $f_W = f_{XT}$)
1	0	0	0	$2^4/f_W$ (488 μs: $f_W = f_{BRG}$)
1	0	0	1	$2^5/f_W$ (977 μs: $f_W = f_{BRG}$)
1	0	1	0	$2^6/f_W$ (1.95 ms: $f_W = f_{BRG}$)
1	0	1	1	$2^7/f_W$ (3.90 ms: $f_W = f_{BRG}$)
1	1	0	0	$2^8/f_W$ (7.81 ms: $f_W = f_{BRG}$)
1	1	0	1	$2^9/f_W$ (15.6 ms: $f_W = f_{BRG}$)
1	1	1	0	$2^{10}/f_W$ (31.2 ms: $f_W = f_{BRG}$)
1	1	1	1	$2^{11}/f_W$ (62.5 ms: $f_W = f_{BRG}$)

(2/2)

WTM7	WTM3	WTM2	Selection of set time of watch flag
0	0	0	$2^{14}/f_W$ (0.5 s: $f_W = f_{XT}$)
0	0	1	$2^{13}/f_W$ (0.25 s: $f_W = f_{XT}$)
0	1	0	$2^5/f_W$ (977 μs: $f_W = f_{XT}$)
0	1	1	$2^4/f_W$ (488 μs: $f_W = f_{XT}$)
1	0	0	$2^{14}/f_W$ (0.5 s: $f_W = f_{BRG}$)
1	0	1	$2^{13}/f_W$ (0.25 s: $f_W = f_{BRG}$)
1	1	0	$2^5/f_W$ (977 μs: $f_W = f_{BRG}$)
1	1	1	$2^4/f_W$ (488 μs: $f_W = f_{BRG}$)

WTM1	Control of 5-bit counter operation
0	Clears after operation stops
1	Starts

WTM0	Watch timer operation enable
0	Stops operation (clears both prescaler and 5-bit counter)
1	Enables operation

Caution Rewrite the WTM2 to WTM7 bits while both the WTM0 and WTM1 bits are 0.

Remarks 1. f_W: Watch timer clock frequency

2. Values in parentheses apply when f_W = 32.768 kHz

11.4 Operation

11.4.1 Operation as watch timer

The watch timer generates an interrupt request at fixed time intervals. The watch timer operates using time intervals of 0.5 seconds with the subclock (32.768 kHz) or prescaler 3 (at 32.768 kHz).

The count operation starts when the WTM1 and WTM0 bits of the WTM register are set to 11. When the WTM0 bit is cleared to 0, the 11-bit prescaler and 5-bit counter are cleared and the count operation stops.

The time of the watch timer can be adjusted by clearing the WTM1 bit to 0 and then the 5-bit counter. At this time, an error of up to 15.6 ms may occur.

The interval timer may be cleared by clearing the WTM0 bit to 0. However, because the 5-bit counter is cleared at the same time, an error of up to 0.5 seconds may occur when the watch timer overflows (INTWT).

11.4.2 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 count value set in advance.

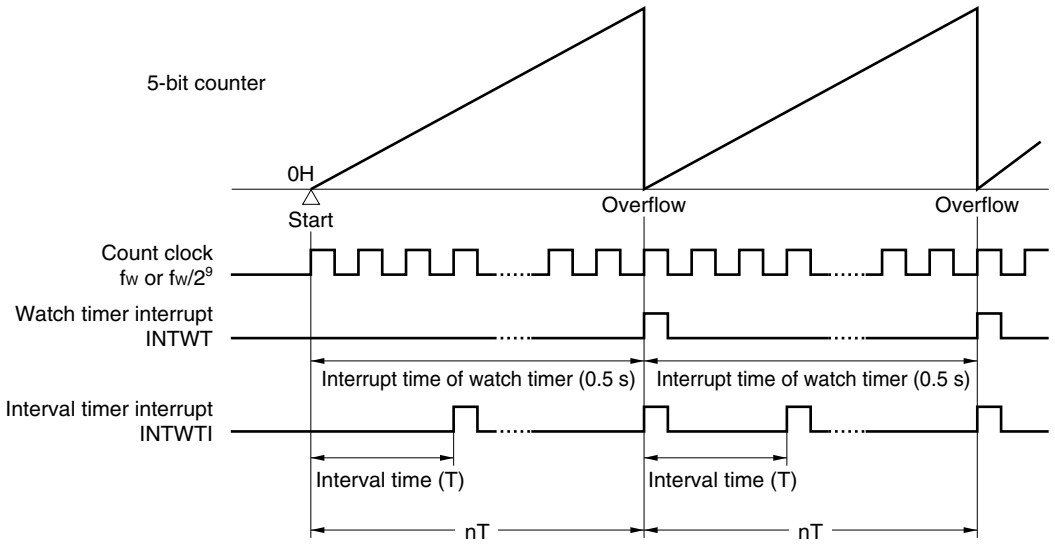
The interval time can be selected by the WTM4 to WTM7 bits of the WTM register.

Table 11-3. Interval Time of Interval Timer

WTM7	WTM6	WTM5	WTM4	Interval Time	
0	0	0	0	$2^4 \times 1/fw$	488 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	0	1	$2^5 \times 1/fw$	977 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	0	$2^6 \times 1/fw$	1.95 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	1	$2^7 \times 1/fw$	3.91 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	0	$2^8 \times 1/fw$	7.81 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	1	$2^9 \times 1/fw$	15.6 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	0	$2^{10} \times 1/fw$	31.3 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	1	$2^{11} \times 1/fw$	62.5 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
1	0	0	0	$2^4 \times 1/fw$	488 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	0	1	$2^5 \times 1/fw$	977 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	0	$2^6 \times 1/fw$	1.95 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	1	$2^7 \times 1/fw$	3.91 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	0	$2^8 \times 1/fw$	7.81 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	1	$2^9 \times 1/fw$	15.6 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	0	$2^{10} \times 1/fw$	31.3 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	1	$2^{11} \times 1/fw$	62.5 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)

Remark f_W: Watch timer clock frequency

Figure 11-3. Operation Timing of Watch Timer/Interval Timer



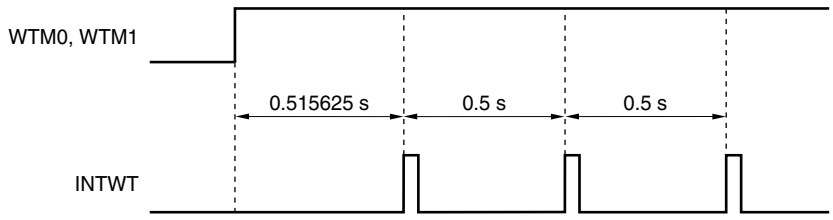
Remark fw: Watch timer clock frequency
Values in parentheses apply when count clock fw = 32.768 kHz.
n: Number of interval timer operations

11.4.3 Cautions

Some time is required before the first watch timer interrupt request signal (INTWT) is generated after operation is enabled (WTM1 and WTM0 bits of WTM register = 1).

Figure 11-4. Example of Generation of Watch Timer Interrupt Request Signal (INTWT)
(When Interrupt Period = 0.5 s)

It takes 0.515625 seconds for the first INTWT signal to be generated ($2^9 \times 1/32768 = 0.015625$ s longer). The INTWT signal is then generated every 0.5 seconds.



11.5 Prescaler 3

The prescaler 3 has the following function.

- Generation of watch timer count clock (source clock: main oscillation clock)

11.5.1 Control register

(1) Prescaler mode register 0 (PRSM0)

The PRSM0 register controls the generation of the watch timer count clock.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF8B0H

	7	6	5	<4>	3	2	1	0
PRSM0	0	0	0	BGCE0	0	0	BGCS01	BGCS00

BGCE0	Prescaler output
0	Disabled (fixed to 0)
1	Enabled

BGCS01	BGCS00	Selection of prescaler 3 clock (f_{BGCS})		
			5 MHz	4 MHz
0	0	f_X	50 ns	250 ns
0	1	$f_X/2$	100 ns	500 ns
1	0	$f_X/4$	200 ns	1 μs
1	1	$f_X/8$	400 ns	2 μs

Cautions 1. Do not change the values of the BGCS00 and BGCS01 bits during watch timer operation.

2. Set the PRSM0 register before setting the BGCE0 bit to 1.

3. Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.

(2) Prescaler compare register 0 (PRSCM0)

The PRSCM0 register is an 8-bit compare register.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF8B1H

	7	6	5	4	3	2	1	0
PRSCM0	PRSCM07	PRSCM06	PRSCM05	PRSCM04	PRSCM03	PRSCM02	PRSCM01	PRSCM00

Cautions 1. Do not rewrite the PRSCM0 register during watch timer operation.

2. Set the PRSCM0 register before setting the BGCE0 bit of the PRSM0 register to 1.

3. Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.

11.5.2 Generation of watch timer count clock

The clock (f_{BRG}) input to the watch timer can be corrected to approximate 32.768 kHz.

The relationship between the main clock (f_X), prescaler 3 clock selection bit BGCSn setting value (m), PRSCM0 register setting value (N) and output clock (f_{BRG}) is as follows.

$$f_{BRG} = \frac{f_X}{2^m \times N \times 2}$$

Example: When f_X = 4.00 MHz, m = 0 (BGCS01 bit = BGCS00 bit = 0), and N = 3DH

f_{BRG} = 32.787 kHz

Remark f_{BRG}: Watch timer count clock

N: PRSCM0 register setting value (1 to FFH)

In the case of PRSCM0 register setting value 00H, N = 256

m: BGCS01, BGCS00 bit setting value (0 to 3)

n = 00, 01

CHAPTER 12 FUNCTIONS OF WATCHDOG TIMER 2

12.1 Functions

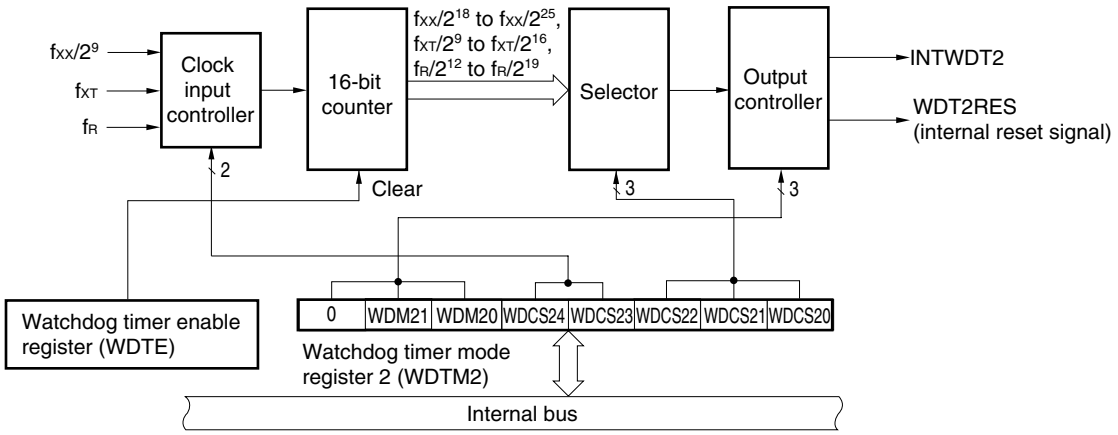
Watchdog timer 2 has the following functions.

- Default start watchdog timer[Note 1]
  → Reset mode: Reset operation upon overflow of watchdog timer 2 (generation of WDT2RES signal)
  → Non-maskable interrupt request mode: NMI operation upon overflow of watchdog timer 2 (generation of INTWDT2 signal)[Note 2]
- Input selectable from main clock, Ring-OSC, and subclock as the source clock

Notes 1. Watchdog timer 2 automatically starts in the reset mode following reset release.
When watchdog timer 2 is not used, either stop its operation before reset is executed through this function, or clear watchdog timer 2 once and stop it within the next interval time.
Also, perform write to the WDTM2 register for verification purposes only once, even if the default settings (reset mode, interval time: $f_R/2^{19}$) need not be changed.

2. Restoring using the RETI instruction following non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT2) is not possible. Therefore, following completion of interrupt servicing, perform a system reset.

Figure 12-1. Block Diagram of Watchdog Timer 2



Remark f_{XX}: Internal system clock frequency
f_{XT}: Subclock frequency
f_R: Ring-OSC clock frequency
INTWDT2: Non-maskable interrupt request signal from watchdog timer 2
WDTRES2: Watchdog timer 2 reset signal

12.2 Configuration

Watchdog timer 2 consists of the following hardware.

Table 12-1. Configuration of Watchdog Timer 2

Item	Configuration
Control registers	Oscillation stabilization time select register (OSTS) Watchdog timer mode register 2 (WDTM2) Watchdog timer enable register (WDTE)

12.3 Control Registers

(1) Oscillation stabilization time select register (OSTS)

The OSTS register selects the oscillation stabilization time following reset or release of the stop mode.

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

Reset input sets this register to 06H.

After reset: 06H R/W Address: FFFFF6C0H

	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/setup time[Note]	f_X	
				4 MHz	5 MHz
0	0	0	$2^{10}/f_X$	0.256 ms	0.205 ms
0	0	1	$2^{11}/f_X$	0.512 ms	0.410 ms
0	1	0	$2^{12}/f_X$	1.024 ms	0.819 ms
0	1	1	$2^{13}/f_X$	2.048 ms	1.638 ms
1	0	0	$2^{14}/f_X$	4.096 ms	3.277 ms
1	0	1	$2^{15}/f_X$	8.192 ms	6.554 ms
1	1	0	$2^{16}/f_X$	16.38 ms	13.107 ms
1	1	1	Setting prohibited		

Note The oscillation stabilization time and setup time are required when the software stop mode and idle mode are released, respectively.

(2) Watchdog timer mode register 2 (WDTM2)

The WDTM2 register sets the overflow time and operation clock of watchdog timer 2.

This register can be read or written in 8-bit units. This register can be read any number of times, but it is write-only once following reset release.

Reset input sets this register to 67H.

After reset: 67H R/W Address: FFFFF6D0H

	7	6	5	4	3	2	1	0
WDTM2	0	WDM21	WDM20	WDCS24	WDCS23	WDCS22	WDCS21	WDCS20

WDM21	WDM20	Selection of operation mode of watchdog timer 2
0	0	Stops operation
0	1	Non-maskable interrupt request mode (generation of INTWDT2 signal)
1	–	Reset mode (generation of WDT2RES signal)

Cautions 1. For details of bits WDCS20 to WDCS24, see Table 12-2 Watchdog Timer 2 Clock Selection.

2. Although watchdog timer 2 can be stopped just by stopping the Ring-OSC operation, set the WDTM2 register to 00H to secure the operation (to avoid selection of the main clock or subclock due to an erroneous write operation).

3. If the WDTM2 register is rewritten twice after reset, an overflow signal is forcibly generated.

4. To stop the operation of watchdog timer 2, set the RSTP bit of the RCM register to 1 (to stop Ring-OSC) and write 00H to the WDTM2 register.

Table 12-2. Watchdog Timer 2 Clock Selection

WDCS24	WDCS23	WDCS22	WDCS21	WDCS20	Selected clock	100 kHz (MIN.)	200 kHz (TYP.)	400 kHz (MAX.)
0	0	0	0	0	$2^{12}/f_R$	41.0 ms	20.5 ms	10.2 ms
0	0	0	0	1	$2^{13}/f_R$	81.9 ms	41.0 ms	20.5 ms
0	0	0	1	0	$2^{14}/f_R$	163.8 ms	81.9 ms	41.0 ms
0	0	0	1	1	$2^{15}/f_R$	327.7 ms	163.8 ms	81.9 ms
0	0	1	0	0	$2^{16}/f_R$	655.4 ms	327.7 ms	163.8 ms
0	0	1	0	1	$2^{17}/f_R$	1310.7 ms	655.4 ms	327.7 ms
0	0	1	1	0	$2^{18}/f_R$	2621.4 ms	1310.7 ms	655.4 ms
0	0	1	1	1	$2^{19}/f_R$	5242.9 ms	2621.47 ms	1310.7 ms
						f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 10 MHz
0	1	0	0	0	$2^{18}/f_{XX}$	13.1 ms	16.4 ms	26.2 ms
0	1	0	0	1	$2^{19}/f_{XX}$	26.2 ms	32.8 ms	52.4 ms
0	1	0	1	0	$2^{20}/f_{XX}$	52.4 ms	65.5 ms	104.9 ms
0	1	0	1	1	$2^{21}/f_{XX}$	104.9 ms	131.1 ms	209.7 ms
0	1	1	0	0	$2^{22}/f_{XX}$	209.7 ms	262.1 ms	419.4 ms
0	1	1	0	1	$2^{23}/f_{XX}$	419.4 ms	524.3 ms	838.9 ms
0	1	1	1	0	$2^{24}/f_{XX}$	838.9 ms	1048.6 ms	1677.7 ms
0	1	1	1	1	$2^{25}/f_{XX}$	1677.7 ms	2097.2 ms	3355.4 ms
						f_{XT} = 32.768 kHz		
1	×	0	0	0	$2^9/f_{XT}$	15.625 ms		
1	×	0	0	1	$2^{10}/f_{XT}$	31.25 ms		
1	×	0	1	0	$2^{11}/f_{XT}$	62.5 ms		
1	×	0	1	1	$2^{12}/f_{XT}$	125 ms		
1	×	1	0	0	$2^{13}/f_{XT}$	250 ms		
1	×	1	0	1	$2^{14}/f_{XT}$	500 ms		
1	×	1	1	0	$2^{15}/f_{XT}$	1000 ms		
1	×	1	1	1	$2^{16}/f_{XT}$	2000 ms		

(3) Watchdog timer enable register (WDTE)

The counter of the watchdog timer 2 is cleared and counting restarted by writing “ACH” to the WDTE register.

The WDTE register can be read or written in 8-bit units.

Reset input sets this register to 9AH.

After reset: 9AH R/W Address: FFFFF6D1H

	7	6	5	4	3	2	1	0
WDTE								

Cautions 1. When a value other than “ACH” is written to the WDTE register, an overflow signal is forcibly output.

2. When a 1-bit memory manipulation instruction is executed for the WDTE register, an overflow signal is forcibly output (an error results in the assembler).

3. The read value of the WDTE register is “9AH” (value that differs from written value “ACH”).

12.4 Operation

(1) Oscillation stabilization time selection function

The wait time until the oscillation stabilizes after the software STOP mode is released is controlled by the OSTS register.

The OSTS register can be read or written 8-bit units.

Reset input sets this register to 06H.

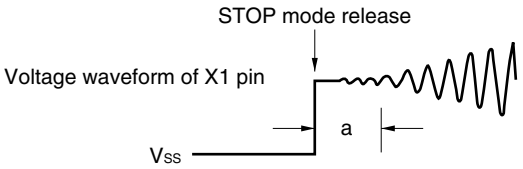
After reset: 06H R/W Address: FFFFF6C0H

	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/setup time[Note]	fx	
				4 MHz	5 MHz
0	0	0	$2^{10}/f_x$	0.256 ms	0.205 ms
0	0	1	$2^{11}/f_x$	0.512 ms	0.410 ms
0	1	0	$2^{12}/f_x$	1.024 ms	0.819 ms
0	1	1	$2^{13}/f_x$	2.048 ms	1.638 ms
1	0	0	$2^{14}/f_x$	4.096 ms	3.277 ms
1	0	1	$2^{15}/f_x$	8.192 ms	6.554 ms
1	1	0	$2^{16}/f_x$	16.38 ms	13.107 ms
1	1	1	Setting prohibited		

Note The oscillation stabilization time and setup time are required when the software STOP mode and idle mode are released, respectively.

Cautions 1. The wait time following release of the software STOP mode does not include the time until the clock oscillation starts (“a” in the figure below) following release of the software STOP mode, regardless of whether the software STOP mode is released through $\overline{\text{RESET}}$ input or the occurrence of an interrupt request signal.



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

3. The oscillation stabilization time following reset release is $2^{16}/f_x$ (because the initial value of the OSTS register = 06H).

Remark f_x = Oscillation frequency

(2) Operation as watchdog timer 2

Watchdog timer 2 automatically starts in the reset mode following reset release.

The WDTM2 register can be written to only once following reset using byte access. To use watchdog timer 2, write the operation mode and the interval time to the WDTM2 register using an 8-bit memory manipulation instruction. After this, the operation of watchdog timer 2 cannot be stopped.

The WDCS24 to WDCS20 bits of the WDTM2 register are used to select the watchdog timer 2 loop detection time interval. Writing ACH to the WDTE register clears the counter of watchdog timer 2 and starts the counting operation again. After the counting operation has started, write ACH to WDTE within the time interval of detecting a loop.

If the time interval expires without ACH being written to the WDTE register, a reset signal (WDT2RES) or a non-maskable interrupt request signal (INTWDT2) is generated, depending on the set values of the WDM21 and WDM20 bits of the WDTM2 register.

To not use watchdog timer 2, write 1FH to the WDTM2 register.

If the non-maskable interrupt request mode has been set, restoring using the RETI instruction following non-maskable interrupt servicing is not possible. Therefore, following completion of interrupt servicing, perform a system reset.

CHAPTER 13 A/D CONVERTER

13.1 Functions

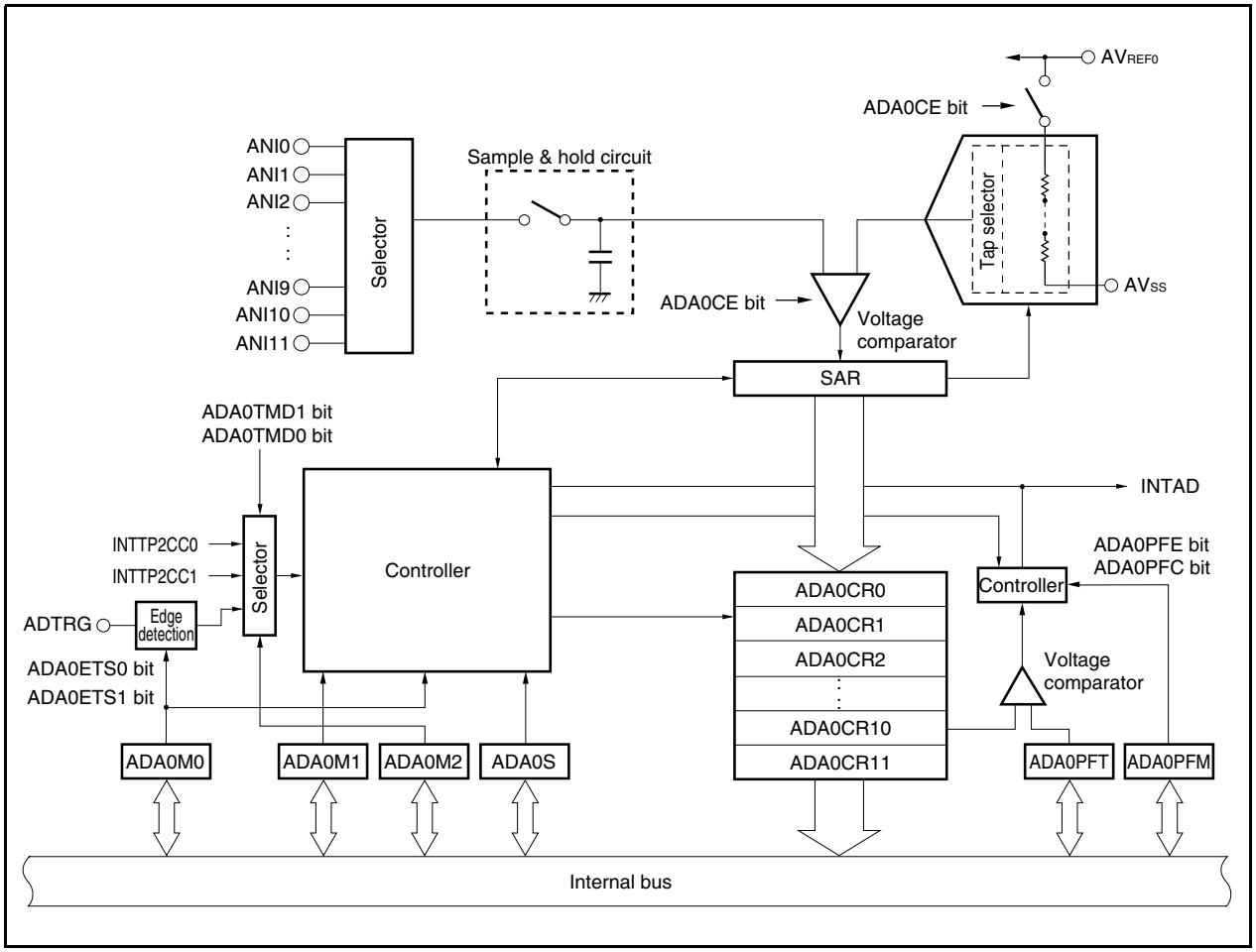
The A/D converter converts analog input signals into digital values, has a resolution of 10 bits, and can handle 12 channels of analog input signals (ANI0 to ANI11).

The A/D converter has the following features.

- 10-bit resolution
- 12 channels
- Successive approximation method
- Operating voltage: AV_{REF0} = 3.0 to 3.6 V
- Analog input voltage: 0 V to AV_{REF0}
- The following functions are provided as operation modes.
  - Continuous select mode
  - Continuous scan mode
  - One-shot select mode
  - One-shot scan mode
- The following functions are provided as trigger modes.
  - Software trigger mode
  - External trigger mode (external, 1)
  - Timer trigger mode
- Power-fail monitor function (conversion result compare function)

The block diagram of the A/D converter is shown below.

Figure 13-1. Block Diagram of A/D Converter



13.2 Configuration

The A/D converter includes the following hardware.

Table 13-1. Configuration of A/D Converter

Item	Configuration
Analog inputs	12 channels (ANI0 to ANI11 pins)
Registers	Successive approximation register (SAR) A/D conversion result registers 0 to 11 (ADA0CR0 to ADA0CR11) A/D conversion result registers 0H to 11H (ADCR0H to ADCR11H): Only higher 8 bits can be read
Control registers	A/D converter mode registers 0, 1 (ADA0M0, ADA0M1) A/D converter channel specification register 0 (ADA0S)

(1) Successive approximation register (SAR)

The SAR 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 comparison result starting from the most significant bit (MSB).

When the comparison result has been held down to the least significant bit (LSB) (i.e. when A/D conversion has been completed), the contents of the SAR register are transferred to the ADA0CRn register.

Remark n = 0 to 11

(2) A/D conversion result register n (ADA0CRn), A/D conversion result register nH (ADA0CRnH)

The ADA0CRn register is a 16-bit register that stores the A/D conversion result. ADA0ARn consist of 12 registers and the A/D conversion result is stored in the 10 higher bits of the AD0CRn register corresponding to analog input. (The lower 6 bits are fixed to 0.)

The ADA0CRn register is read-only, in 16-bit units.

Moreover, when using only the higher 8 bits of the A/D conversion result, the ADA0CRnH register is read-only, in 8-bit units.

Caution A write operation to the ADA0M0 and ADA0S registers may cause the contents of the ADA0CRn register to become undefined. After the conversion, read the conversion result before performing write to the ADA0M0 and ADA0S registers. Correct conversion results may not be read if a sequence other than the above is used.

Remark n = 0 to 11

(3) Power-fail compare threshold value register (ADA0PFT)

The ADA0PFT register sets a threshold value that is compared with the value of A/D conversion result register nH (ADA0CRnH). The 8-bit data set to the ADA0PFT register is compared with the higher 8 bits (ADA0CRnH) of the A/D conversion result register.

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

Reset input clears this register to 00H.

(4) Sample & hold circuit

The sample & hold circuit samples each of the analog input signals selected by 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.

(5) Voltage comparator

The voltage comparator compares a voltage value that has been sampled and held with the voltage value of the series resistor string.

(6) Series resistor string

This series resistor string is connected between AV_{REF0} and AV_{SS} and generates a voltage for comparison with the analog input signal.

(7) ANI0 to ANI11 pins

These are analog input pins for the 12 channels of the A/D converter and are used to input analog signals to be converted into digital signals. Pins other than ones selected as analog input with the ADA0S register can be used as input ports.

Caution Make sure that the voltages input to ANI0 to ANI11 pins do not exceed the rated values. In particular if a voltage higher than AV_{REF0} is input to a channel, the conversion value of that channel becomes undefined, and the conversion values of the other channels may also be affected.

(8) AV_{REF0} pin

This is the pin used to input the reference voltage of the A/D converter. The signals input to the ANI0 to ANI11 pins are converted to digital signals based on the voltage applied between the AV_{REF0} and AV_{SS} pins.

(9) AV_{SS} pin

This is the ground pin of the A/D converter. Always make the potential at this pin the same as that at the V_{SS} pin even when the A/D converter is not used.

13.3 Control Registers

The A/D converter is controlled by the following registers.

- A/D converter mode registers 0, 1, 2 (ADA0M0, ADA0M1, ADA0M2)
- A/D converter channel specification register 0 (ADA0S)
- Power-fail compare mode register (ADA0PFM)

The following registers are also used.

- A/D conversion result register n (ADA0CRn)
- A/D conversion result register nH (ADA0CRnH)
- Power-fail compare threshold value register (ADA0PFT)

(1) A/D converter mode register 0 (ADA0M0)

The ADA0M0 register is an 8-bit register that specifies the operation mode and controls conversion operation.

This register can be read or written in 8-bit or 1-bit units. However, bit 0 is read-only.

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF200H

	<7>	6	5	4	3	2	1	<0>
ADA0M0	ADA0CE	0	ADA0MD1	ADA0MD0	ADA0ETS1	ADA0ETS0	ADA0TMD	ADA0EF

ADA0CE	A/D conversion control
0	Stops conversion
1	Enables conversion

ADA0MD1	ADA0MD0	Specification of A/D converter operation mode
0	0	Continuous select mode
0	1	Continuous scan mode
1	0	One-shot select mode
1	1	One-shot scan mode

ADA0ETS1	ADA0ETS0	Specification of external trigger (ADTRG pin) input valid edge
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Detection of both rising and falling edges

ADA0TMD	Trigger mode specification
0	Software trigger mode
1	External trigger mode/timer trigger mode

ADA0EF	A/D converter status display
0	A/D conversion stopped
1	A/D conversion in progress

Cautions 1. If bit 0 is written to, this is ignored.

2. Changing the ADA0FR2 to ADA0FR0 bits of the ADA0M1 register during conversion (ADA0CE0 bit = 1) is prohibited.

3. When not using the A/D converter, stop the operation by setting ADA0CE bit = 0 to reduce the consumption current.

(2) A/D converter mode register 1 (ADA0M1)

The ADA0M1 register is an 8-bit register that specifies the conversion time.

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

Reset input clears this bit to 00H.

After reset: 00H R/W Address: FFFFF201H

	7	6	5	4	3	2	1	0
ADA0M1	ADA0HS1	0	0	0	0	ADA0FR2	ADA0FR1	ADA0FR0

ADA0HS1	Specification of normal conversion mode/high-speed mode (A/D conversion time)
0	Normal conversion mode
1	High-speed conversion mode

Caution Be sure to clear bits 6 to 3 to 0.

Remark For A/D conversion time setting examples, see **Tables 13-2** and **13-3**.

Table 13-2. Normal Conversion Mode Setting Examples

ADA0FR2	ADA0FR1	ADA0FR0	A/D Conversion Time[Note]	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 4 MHz
0	0	0	65/f_{XX}	Setting prohibited	Setting prohibited	16.25 μs
0	0	1	130/f_{XX}	6.50 μs	8.13 μs	Setting prohibited
0	1	0	195/f_{XX}	9.75 μs	12.19 μs	Setting prohibited
0	1	1	258/f_{XX}	12.90 μs	16.13 μs	Setting prohibited
1	0	0	310/f_{XX}	15.50 μs	19.38 μs	Setting prohibited
1	0	1	362/f_{XX}	18.10 μs	22.63 μs	Setting prohibited
1	1	0	414/f_{XX}	20.70 μs	Setting prohibited	Setting prohibited
1	1	1	466/f_{XX}	23.30 μs	Setting prohibited	Setting prohibited

Note Set the A/D conversion time so that 5.16 μs ≤ conversion time ≤ 26.0 μs.

Table 13-3. High-Speed Conversion Mode Setting Examples

ADA0FR2	ADA0FR1	ADA0FR0	A/D conversion Time[Note 1]	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 4 MHz	A/D Stabilization Time[Note 2]
0	0	0	26/f_{XX}	Setting prohibited	Setting prohibited	6.5 μs	13/f_{XX}
0	0	1	52/f_{XX}	2.60 μs	3.25 μs	Setting prohibited	26/f_{XX}
0	1	0	78/f_{XX}	3.90 μs	4.88 μs	Setting prohibited	39/f_{XX}
0	1	1	104/f_{XX}	5.20 μs	6.50 μs	Setting prohibited	50/f_{XX}
1	0	0	130/f_{XX}	6.50 μs	8.13 μs	Setting prohibited	50/f_{XX}
1	0	1	156/f_{XX}	7.80 μs	9.75 μs	Setting prohibited	50/f_{XX}
1	1	0	182/f_{XX}	9.10 μs	Setting prohibited	Setting prohibited	50/f_{XX}
1	1	1	208/f_{XX}	10.40 μs	Setting prohibited	Setting prohibited	50/f_{XX}

Notes 1. Set the conversion time so that 2.08 μs ≤ conversion time ≤ 10.40 μs.

2. When the ADA0CE bit of the ADA0M0 register is changed from 0 to 1 to secure the A/D converter stabilization time (1 μs), prior to the first conversion only, A/D conversion starts after one of the above clock values is input.

(3) A/D converter mode register (ADA0M2)

The ADA0M2 register specifies the hardware trigger mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF203H

	7	6	5	4	3	2	1	0
ADA0M2	0	0	0	0	0	0	ADA0TMD1	ADA0TMD0

ADA0TMD1	ADA0TMD0	Specification of hardware trigger mode
0	0	External trigger mode (when ADTRG pin valid edge detected)
0	1	Timer trigger mode 0 (when INTTP2CC0 interrupt request generated)
1	0	Timer trigger mode 1 (when INTTP2CC1 interrupt request generated)
1	1	Setting prohibited

Caution Be sure to clear bits 7 to 2 to 0.

(4) Analog input channel specification register 0 (ADA0S)

The ADA0S register is a register that specifies the port for inputting the analog voltage to be converted into a digital signal.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF202H

	7	6	5	4	3	2	1	0
ADA0S	0	0	0	0	ADA0S3	ADA0S2	ADA0S1	ADA0S0

ADA0S3	ADA0S2	ADA0S1	ADA0S0	Select mode	Scan mode
0	0	0	0	ANI0	ANI0
0	0	0	1	ANI1	ANI0, ANI1
0	0	1	0	ANI2	ANI0 to ANI2
0	0	1	1	ANI3	ANI0 to ANI3
0	1	0	0	ANI4	ANI0 to ANI4
0	1	0	1	ANI5	ANI0 to ANI5
0	1	1	0	ANI6	ANI0 to ANI6
0	1	1	1	ANI7	ANI0 to ANI7
1	0	0	0	ANI8	ANI0 to ANI8
1	0	0	1	ANI9	ANI0 to ANI9
1	0	1	0	ANI10	ANI0 to ANI10
1	0	1	1	ANI11	ANI0 to ANI11
1	1	0	0	Setting prohibited	Setting prohibited
1	1	0	1	Setting prohibited	Setting prohibited
1	1	1	0	Setting prohibited	Setting prohibited
1	1	1	1	Setting prohibited	Setting prohibited

Caution Be sure to clear bits 7 to 4 to 0.

(5) A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH)

The ADA0CRn and ADA0CRnH registers store A/D conversion results.

These registers are read-only, in 16-bit or 8-bit units. However, specify the ADA0CRn register for 16-bit access and the ADA0CRnH register for 8-bit access. The 10 bits of the conversion result can be read from the higher 10 bits of the ADA0CRn register, and 0 can be read from the lower 6 bits. The higher 8 bits of the conversion result can be read from the ADA0CRn register.

After reset: Undefined R Address: ADA0CR0 FFFFF210H, ADA0CR1 FFFFF212H, ADA0CR2 FFFFF214H, ADA0CR3 FFFFF216H, ADA0CR4 FFFFF218H, ADA0CR5 FFFFF21AH, ADA0CR6 FFFFF21CH, ADA0CR7 FFFFF21EH, ADA0CR8 FFFFF220H, ADA0CR9 FFFFF222H, ADA0CR10 FFFFF224H, ADA0CR11 FFFFF226H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADA0CRn (n = 0 to 11)	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0	0	0	0	0	0

After reset: Undefined R Address: ADA0CR0H FFFFF211H, ADA0CR1H FFFFF213H, ADA0CR2H FFFFF215H, ADA0CR3H FFFFF217H, ADA0CR4H FFFFF219H, ADA0CR5H FFFFF21BH, ADA0CR6H FFFFF21DH, ADA0CR7H FFFFF21FH, ADA0CR8H FFFFF221H, ADA0CR9H FFFFF223H, ADA0CR10H FFFFF225H, ADA0CR11H FFFFF227H

	7	6	5	4	3	2	1	0
ADA0CRnH (n = 0 to 11)	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI11) and the A/D conversion result (of A/D conversion result register n (ADA0CRn)) is as follows:

$$ADA0CR = INT\left(\frac{V_{IN}}{AV_{REF0}} \times 1{,}024 + 0.5\right)$$

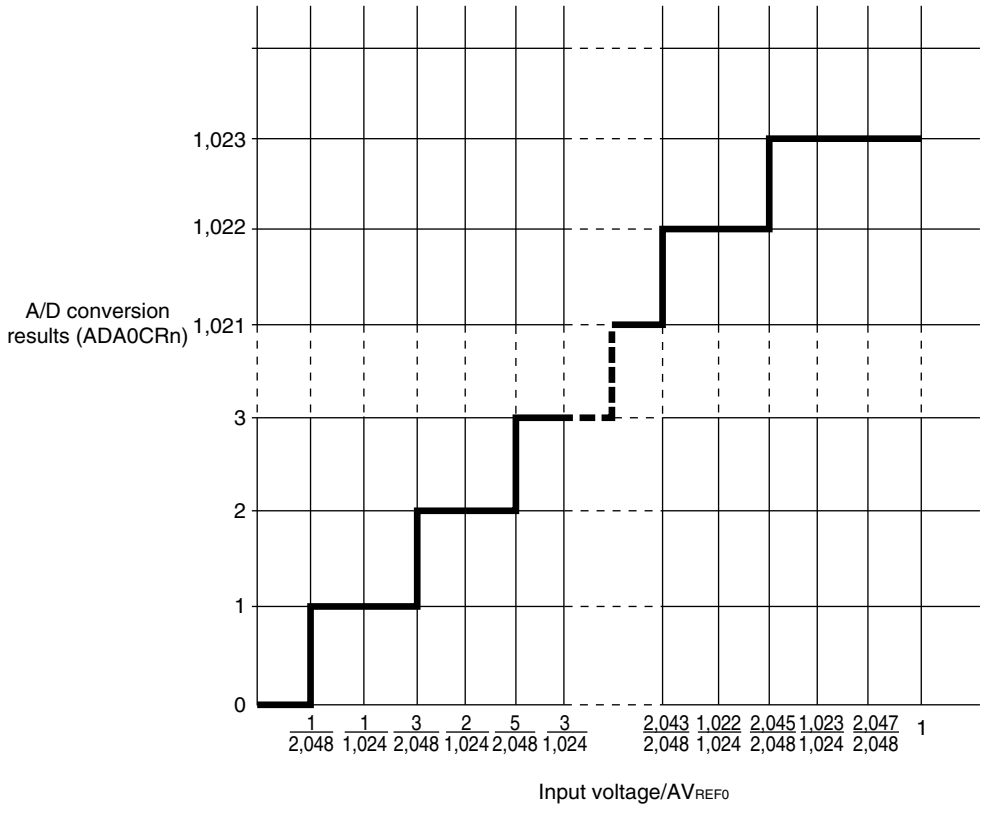
Or,

$$(ADA0CR - 0.5) \times \frac{AV_{REF0}}{1{,}024} \leq V_{IN} < (ADA0CR + 0.5) \times \frac{AV_{REF0}}{1{,}024}$$

INT( ): Function that returns the integer of the value in ( )
V_{IN}: Analog input voltage
AV_{REF0}: AV_{REF0} pin voltage
ADA0CR: Value of A/D conversion result register n (ADA0CRn)

Figure 13-2 shows the relationship between the analog input voltage and the A/D conversion results.

Figure 13-2. Relationship Between Analog Input Voltage and A/D Conversion Results



Remark n = 0 to 11

(6) Power-fail compare mode register (ADA0PFM)

The ADA0PFM register is an 8-bit register that sets the power-fail compare mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF204H

	<7>	<6>	5	4	3	2	1	0
ADA0PFM	ADA0PFE	ADA0PFC	0	0	0	0	0	0

ADA0PFE	Selection of power-fail compare enable/disable
0	Power-fail compare enabled
1	Power-fail compare disabled

ADA0PFC	Selection of power-fail compare mode
0	Generates an interrupt request signal (INTAD) when ADA0CRnH ≥ ADA0PFT
1	Generates an interrupt request signal (INTAD) when ADA0CRnH < ADA0PFT

Cautions 1. In the select mode, the 8-bit data set to the ADA0PFT register is compared with the value of the ADA0CRnH register specified by the ADA0S register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register and the INTAD signal is generated. If it does not match, however, the interrupt signal is not generated.

2. In the scan mode, the 8-bit data set to the ADA0PFT register is compared with the contents of the ADA0CR0H register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, however, the INTAD signal is not generated. Regardless of the comparison result, the scan operation is continued and the conversion result is stored in the ADA0CRn register until the scan operation is completed. However, the INTAD signal is not generated after the scan operation has been completed.

(7) Power-fail compare threshold value register (ADA0PFT)

The ADA0PFT register sets the compare value in the power-fail compare mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF205H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
ADA0PFT								

13.4 Operation

13.4.1 Basic operation

<1> Set the operation mode, trigger mode, and conversion time for executing A/D conversion by using the ADA0M0, ADA0M1, ADA0M2, and ADA0S registers. When the ADA0CE bit of the ADA0M0 register is set, conversion is started in the software trigger mode and the A/D converter waits for a trigger in the external timer trigger mode.

<2> When A/D conversion is started, the voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<3> When the sample & hold circuit samples the input channel for a specific time, it enters the hold status, and holds the input analog voltage until the A/D conversion is completed.

<4> Set bit 9 of the successive approximation register (SAR). The tap selector selects (1/2) AV_{REF0} as the voltage tap of the series resistor string.

<5> The voltage difference between the voltage of the series resistor string and the analog input voltage is compared by the voltage comparator. If the analog input voltage is higher than (1/2) AV_{REF0}, the MSB of the SAR register remains set. If it is lower than (1/2) AV_{REF0}, the MSB is reset.

<6> Next, bit 8 of the SAR register is automatically set and the next comparison is started. Depending on the value of bit 9 to which a result has been already set, the voltage tap of the series resistor string is selected as follows.

- Bit 9 = 1: (3/4) AV_{REF0}
- Bit 9 = 0: (1/4) AV_{REF0}

This voltage tap and the analog input voltage are compared and, depending on the result, bit 8 is manipulated as follows.

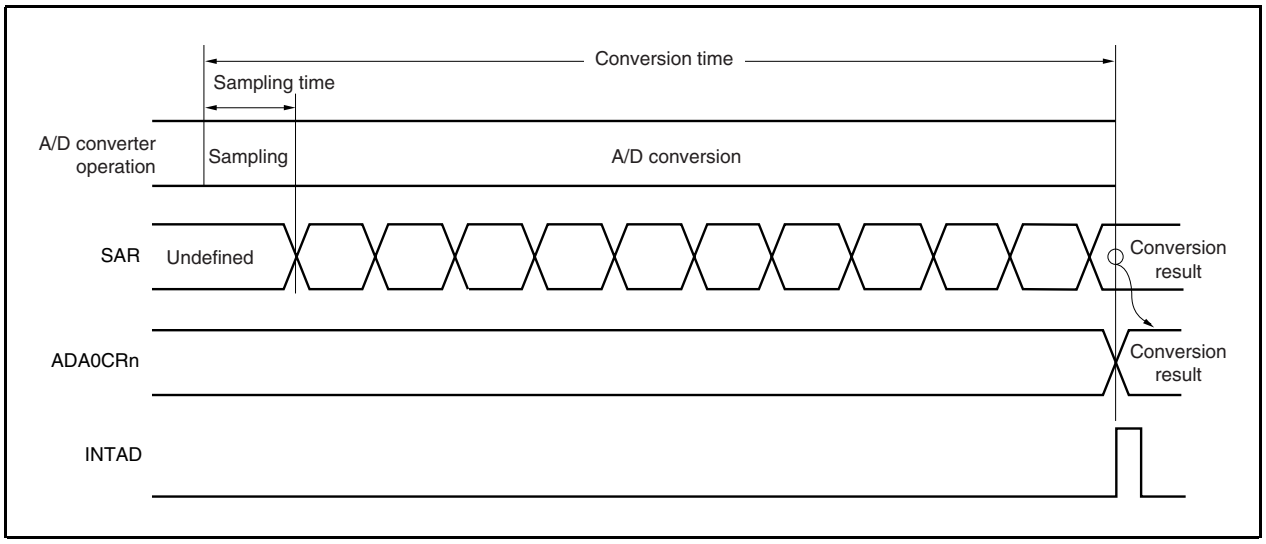
Analog input voltage ≥ Voltage tap: Bit 8 = 1

Analog input voltage ≤ Voltage tap: Bit 8 = 0

<7> This comparison is continued to bit 0 of the SAR register.

<8> When comparison of the 10 bits has been completed, the valid digital result is stored in the SAR register, which is then transferred to and stored in the ADA0CRn register. At the same time, an A/D conversion end interrupt request signal (INTAD) is generated.

Figure 13-3. A/D Converter Basic Operation



13.4.2 Trigger mode

The timing of starting the conversion operation is specified by setting a trigger mode. The trigger mode includes a software trigger mode and hardware trigger modes. The hardware trigger modes include timer trigger modes 0 and 1, and external trigger mode. The ADA0TMD bit of the ADA0M0 register is used to set the trigger mode. The hardware trigger modes are set by the ADA0TMD1 and ADA0TMD0 bits of the ADA0M2 register.

(1) Software trigger mode

When the ADA0CE bit of the ADA0M0 register is set to 1, the signal of the analog input pin (ANI0 to ANI11 pin) specified by the ADA0S register is converted. When conversion is complete, the result is stored in the ADA0CRn register. At the same time, the A/D conversion end interrupt request signal (INTAD) is generated.

If the operation mode specified by the ADA0MD1 and ADA0MD0 bits of the ADA0M0 register is the continuous select/scan mode, the next conversion is started, unless the ADA0CE bit is cleared to 0 after completion of the first conversion. Conversion is performed once and completed if the operation mode is the one-shot select/scan mode.

When conversion is started, the ADA0EF bit is set to 1 (indicating that the operation is in progress).

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is aborted and started again from the beginning.

(2) External trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI11 pin) specified by the ADA0S register is started when an external trigger is input (to the ADTRG pin). How the edge of the external trigger is to be detected (i.e., whether the rising edge, falling edge, or both rising and falling edges are to be detected) can be specified by using the ADA0ETS1 and ATA0ETS0 bits of the ADA0M0 register. When the ADA0CE bit of the ADA0M0 register set to 1, the A/D converter waits for the trigger, and starts conversion after the external trigger has been input.

When conversion is completed, the result of conversion is stored in the ADA0CRn register. At the same time, the A/D conversion end interrupt request signal (INTAD) is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that the operation is in progress). While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that the operation is stopped). If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during the conversion operation, the conversion is not aborted, and the A/D converter waits for the trigger again.

(3) Timer trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI11 pin) specified by the ADA0S register is started by the compare match interrupt request signal (INTTP2CC0 or INTTP2CC1) of the capture/compare register connected to the timer. Either of the compare match interrupt request signals (INTTP2CC0 and INTTP2CC1) of the timer is selected by the ADA0TMD1 and ADA0TMD0 bits of the ADA0M2 register, and conversion is started at the rising edge of the specified compare match interrupt request signal. When the ADA0CE bit of the ADA0M0 register is set to 1, the A/D converter waits for a trigger, and starts conversion when the compare match interrupt signal of the timer is input.

When conversion is completed, the result of the conversion is stored in the ADA0CRn register. At the same time, the A/D conversion end interrupt request signal (INTAD) is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that the operation is in progress). While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that the operation is stopped). If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is stopped and the A/D converter waits for the trigger again.

13.4.3 Operation mode

Four operation modes are available as modes to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

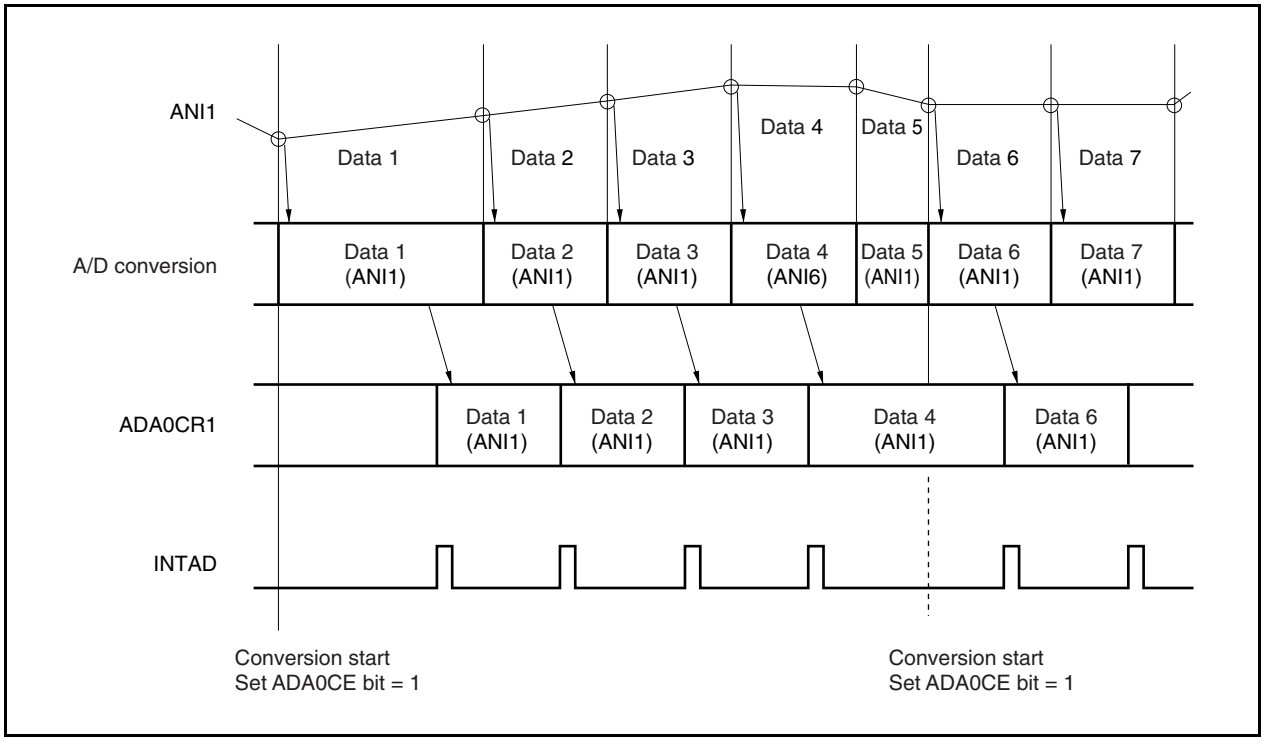
The operation mode is selected by the ADA0MD1 and ADA0MD0 bits of the ADA0M0 register.

(1) Continuous select mode

In this mode, the voltage of one analog input pin selected by the ADA0S register is continuously converted into a digital value.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin. In this mode, an analog input pin corresponds to an ADA0CRn register on a one-to-one basis. Each time A/D conversion has been completed, the A/D conversion end interrupt request signal (INTAD) is generated. After completion of conversion, the next conversion is started, unless the ADA0CE bit of the ADA0M0 register is cleared to “0” (n = 0 to 11).

Figure 13-4. Timing Example of Continuous Select Mode Operation (ADA0S = 01H)



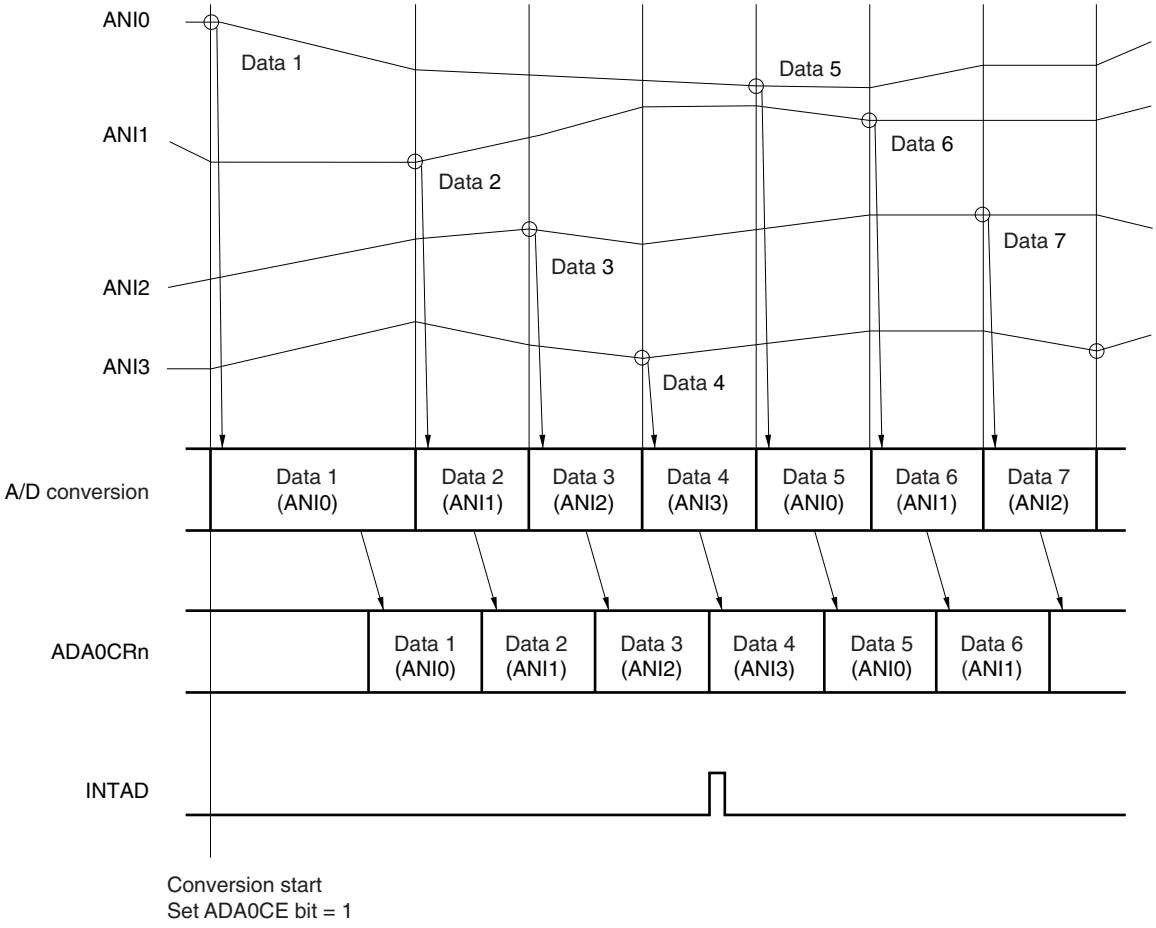
(2) Continuous scan mode

In this mode, the voltages on the analog input pins specified by the ADA0S register are sequentially selected, starting from the ANI0 pin, and converted into digital values.

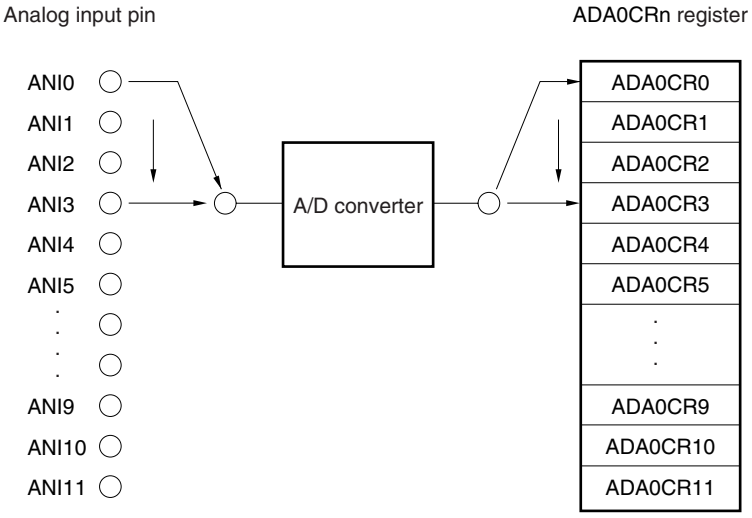
The result of conversion is stored in the ADA0CRn register corresponding to the analog input pin. When conversion of the signals on the analog input pins selected by the ADA0S register is complete, the A/D conversion end interrupt request signal (INTAD) is generated, and A/D conversion is started again from the ANI0 pin, unless the ADA0CE bit of the ADA0M0 register is cleared to 0 (n = 0 to 11).

Figure 13-5. Timing Example of Continuous Scan Mode Operation (ADA0S Register = 03H)

(a) Timing example



(b) Block diagram

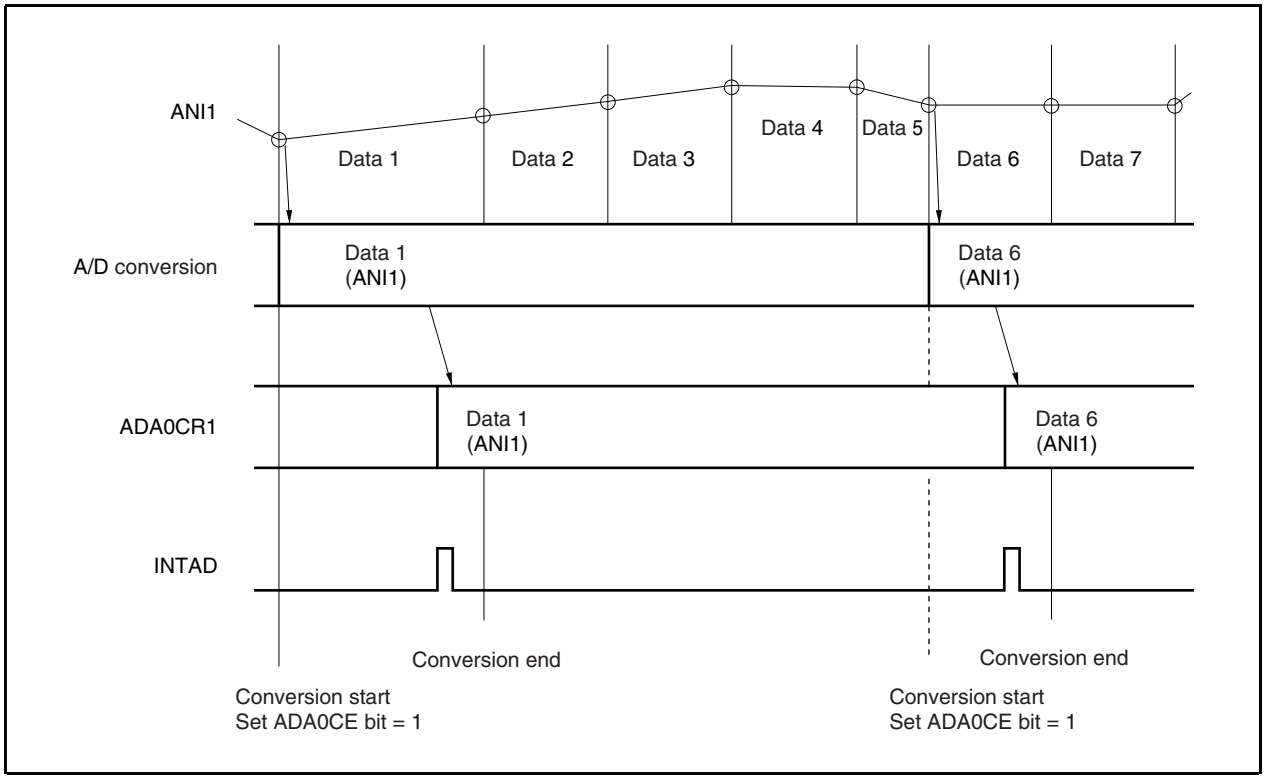


(3) One-shot select mode

In this mode, the voltage on one analog input pin specified by the ADA0S register is converted into a digital value only once.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin. In this mode, an analog input pin and an ADA0CRn register correspond on a one-to-one basis. When A/D conversion has been completed once, the A/D conversion end interrupt request signal (INTAD) is generated. The A/D conversion operation is stopped after it has been completed (n = 0 to 11).

Figure 13-6. Timing Example of One-Shot Select Mode Operation (ADA0S Register = 01H)



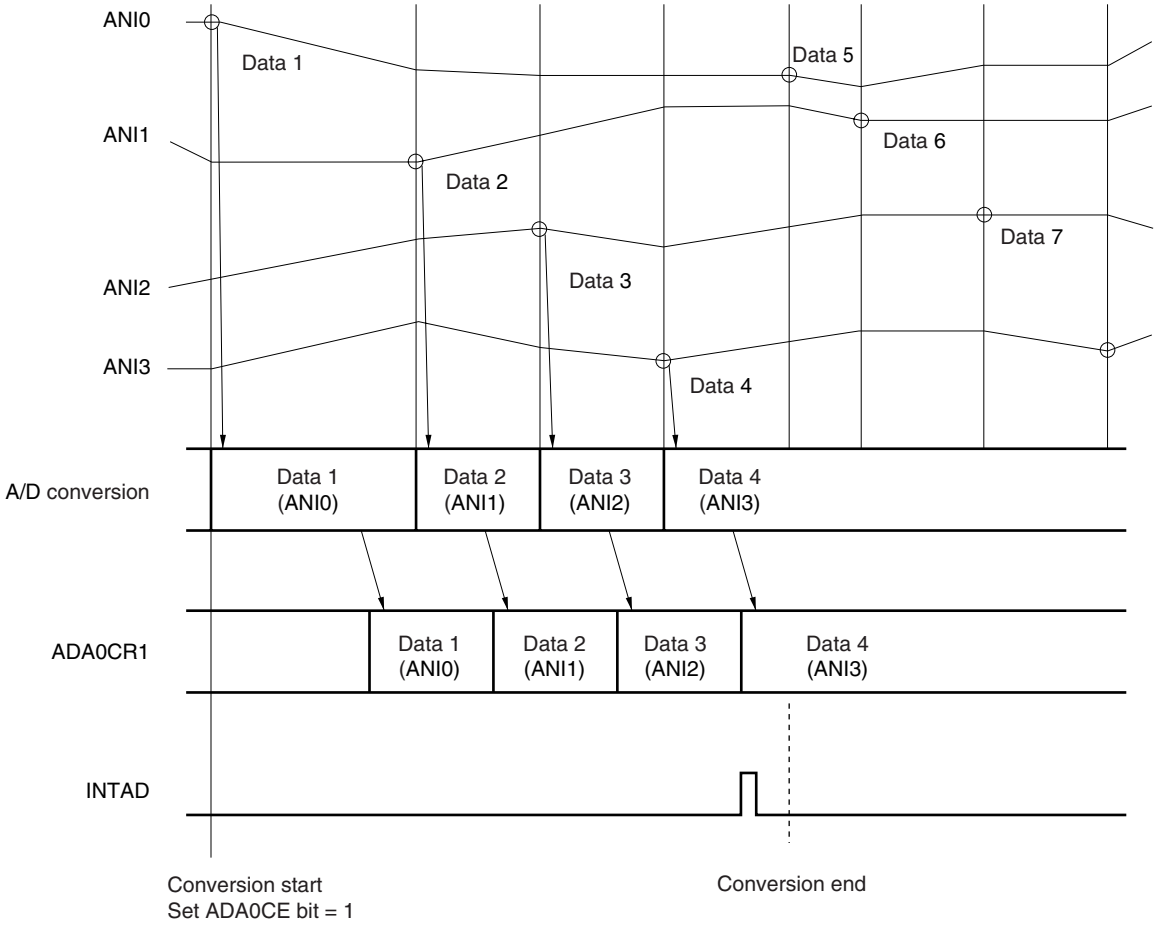
(4) One-shot scan mode

In this mode, the voltages on the analog input pins specified by the ADA0S register are sequentially selected and converted into digital values, starting from the ANI0 pin.

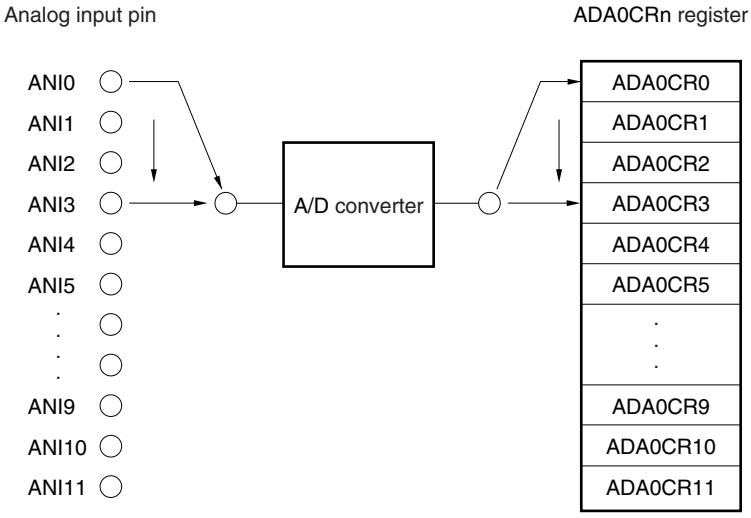
The A/D conversion result is stored in the ADA0CRn register corresponding to the analog input pin. When the voltages on the analog input pins specified by the ADA0S register have been converted, the A/D conversion end interrupt request signal (INTAD) is generated. The A/D conversion is stopped after it has been completed (n = 0 to 11).

Figure 13-7. Timing Example of One-Shot Scan Mode Operation (ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



13.4.4 Power-fail compare mode

The A/D conversion end interrupt request signal (INTAD) can be controlled as follows by the ADA0PFM and ADA0PFT registers.

- When the ADA0PFE bit = 0, the INTAD signal is generated each time conversion has been completed (normal use of the A/D converter).
- When the ADA0PFE bit = 1 and when the ADA0PFC bit = 0, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CR0H ≥ ADA0PFT.
- When the ADA0PFE bit = 1 and when the ADA0PFC bit = 1, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CR0H < ADA0PFT.

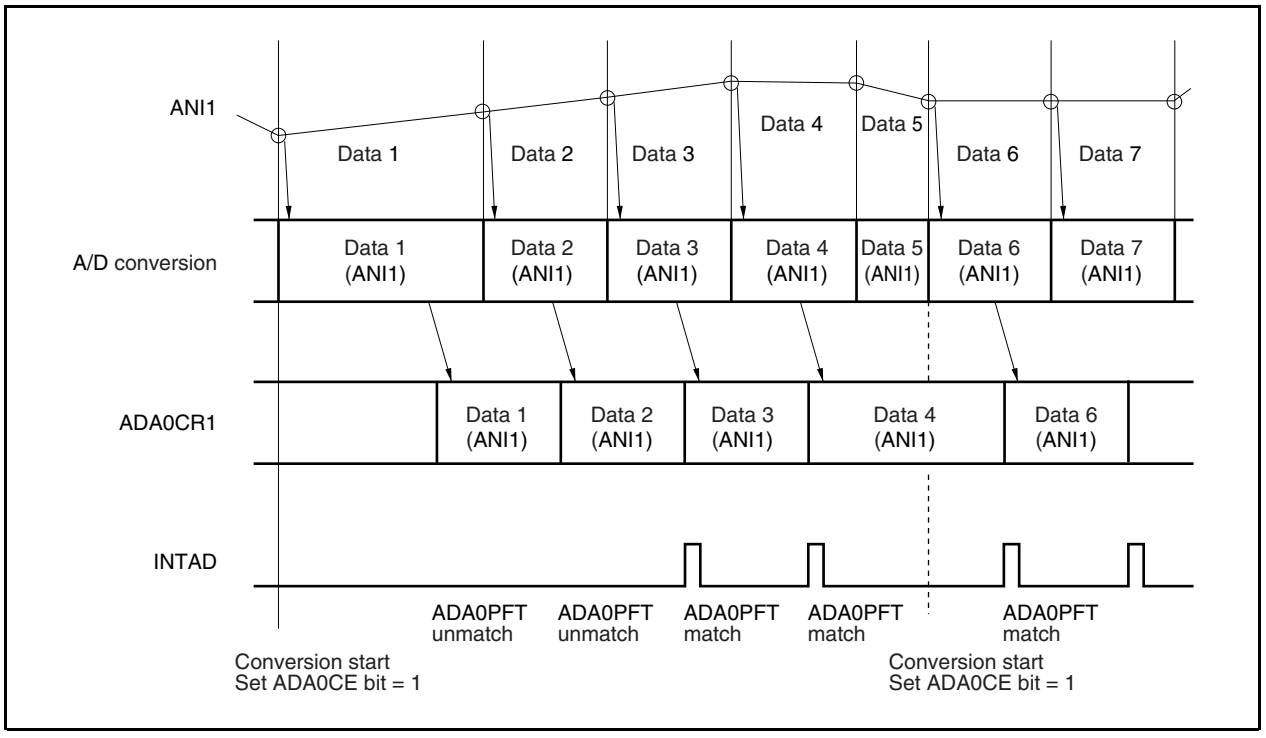
Remark n = 0 to 11

In the power-fail compare mode, four modes are available as modes to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

(1) Continuous select mode

In this mode, the voltage of one analog input pin selected by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail compare matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. After completion of the first conversion, the next conversion is started, unless the ADA0CE bit of the ADA0M0 register is cleared to 0 (n = 0 to 11).

Figure 13-8. Timing Example of Continuous Select Mode Operation (When Power-Fail Compare Is Made: ADA0S Register = 01H)



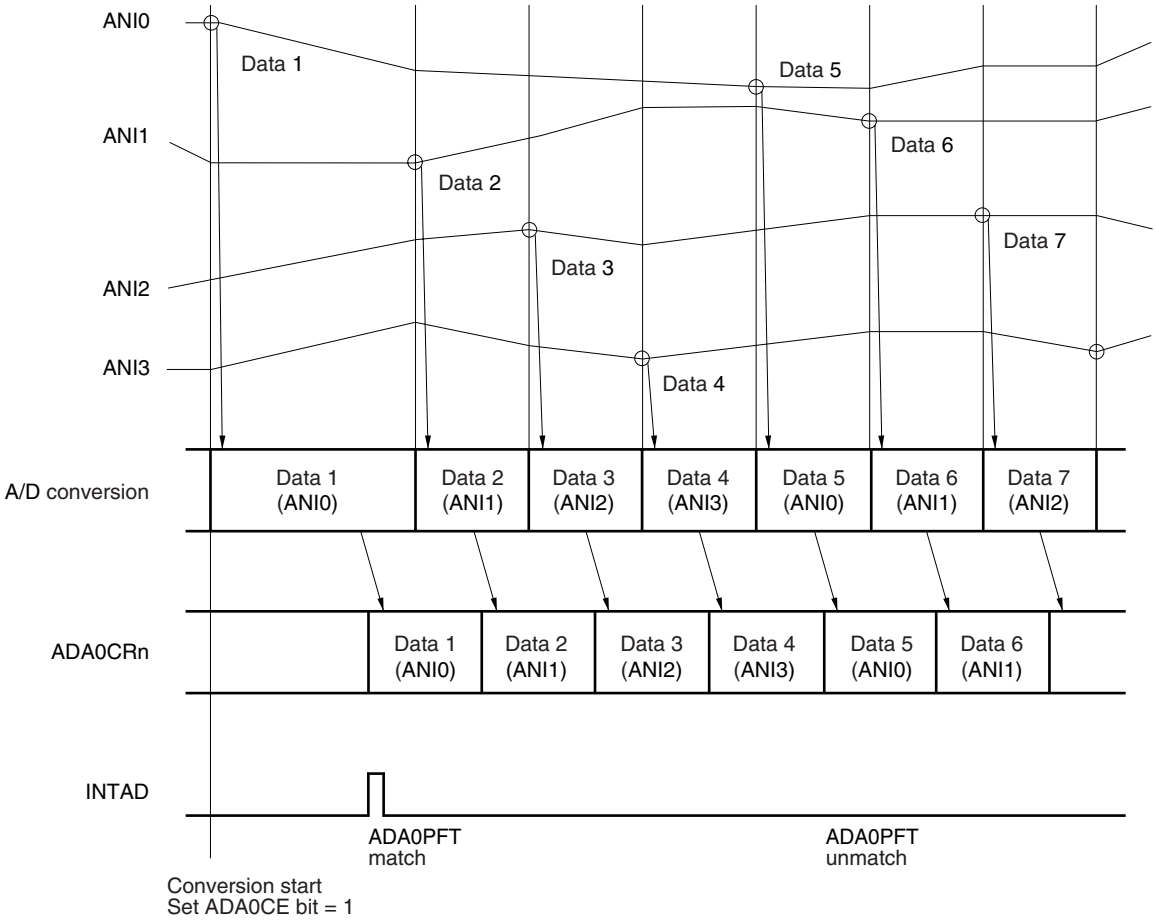
(2) Continuous scan mode

In this mode, the voltages on the analog input pins specified by the ADA0S register are sequentially selected, starting from the ANI0 pin, and converted into digital values, and the set value of the ADA0CR0H register of channel 0 is compared with the value of the ADA0PFT register. If the result of power-fail compare matches the condition set by the ADA0PFC bit of the ADA0PFM register, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is not generated.

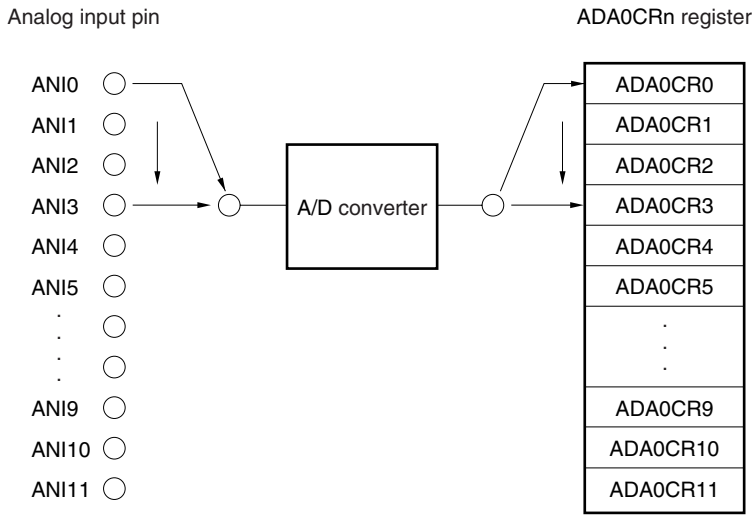
After the result of the first conversion has been stored in the ADA0CR0 register, the results of sequentially converting the voltages on the analog input pins specified by the ADA0S register are continuously stored. After completion of conversion, the next conversion is started from the ANI0 pin again, unless the ADA0CE bit of the ADA0M0 register is cleared to 0.

Figure 13-9. Timing Example of Continuous Scan Mode Operation (When Power-Fail Compare Is Made: ADA0S Register = 03H)

(b) Timing example



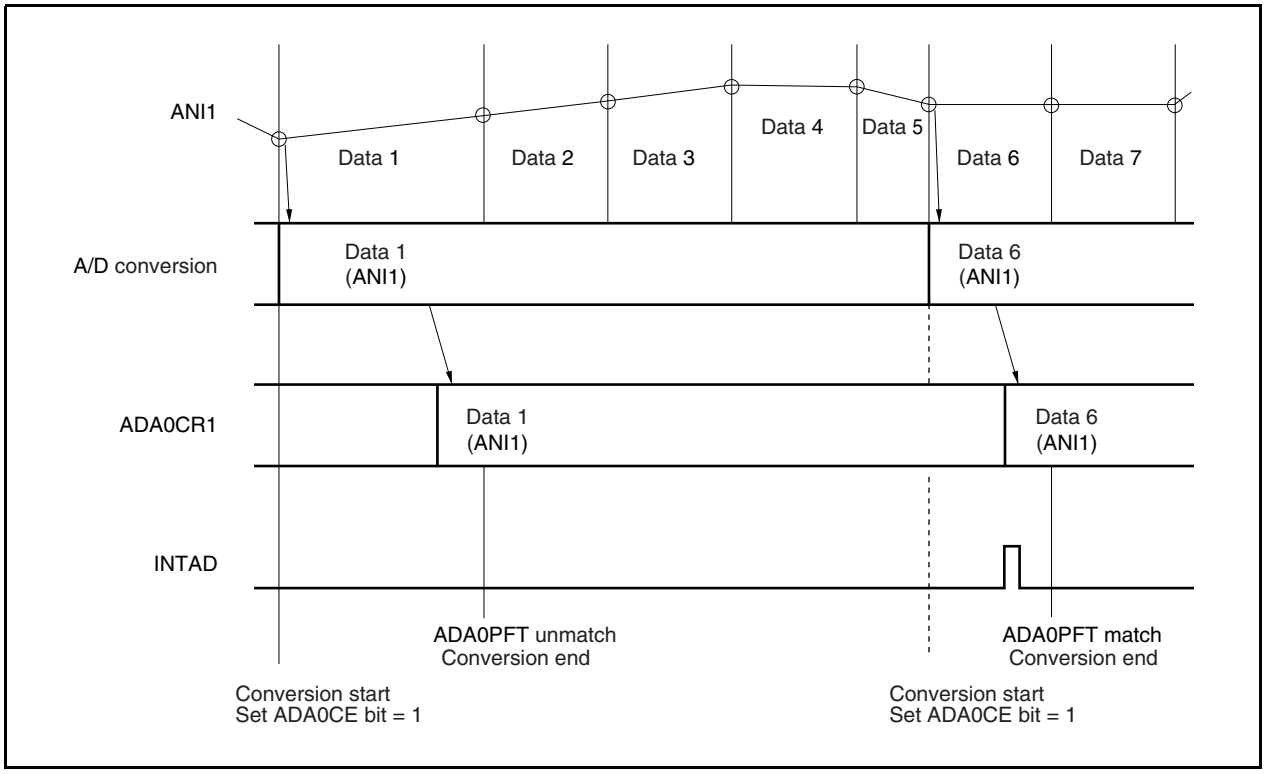
(b) Block diagram



(3) One-shot select mode

In this mode, the result of converting the voltage of one analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail compare matches the condition set by the ADA0PFC bit of the ADA0PFM register, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. Conversion is stopped after it has been completed.

Figure 13-10. Timing Example of One-Shot Select Mode Operation (When Power-Fail Compare Is Made: ADA0S Register = 01H)

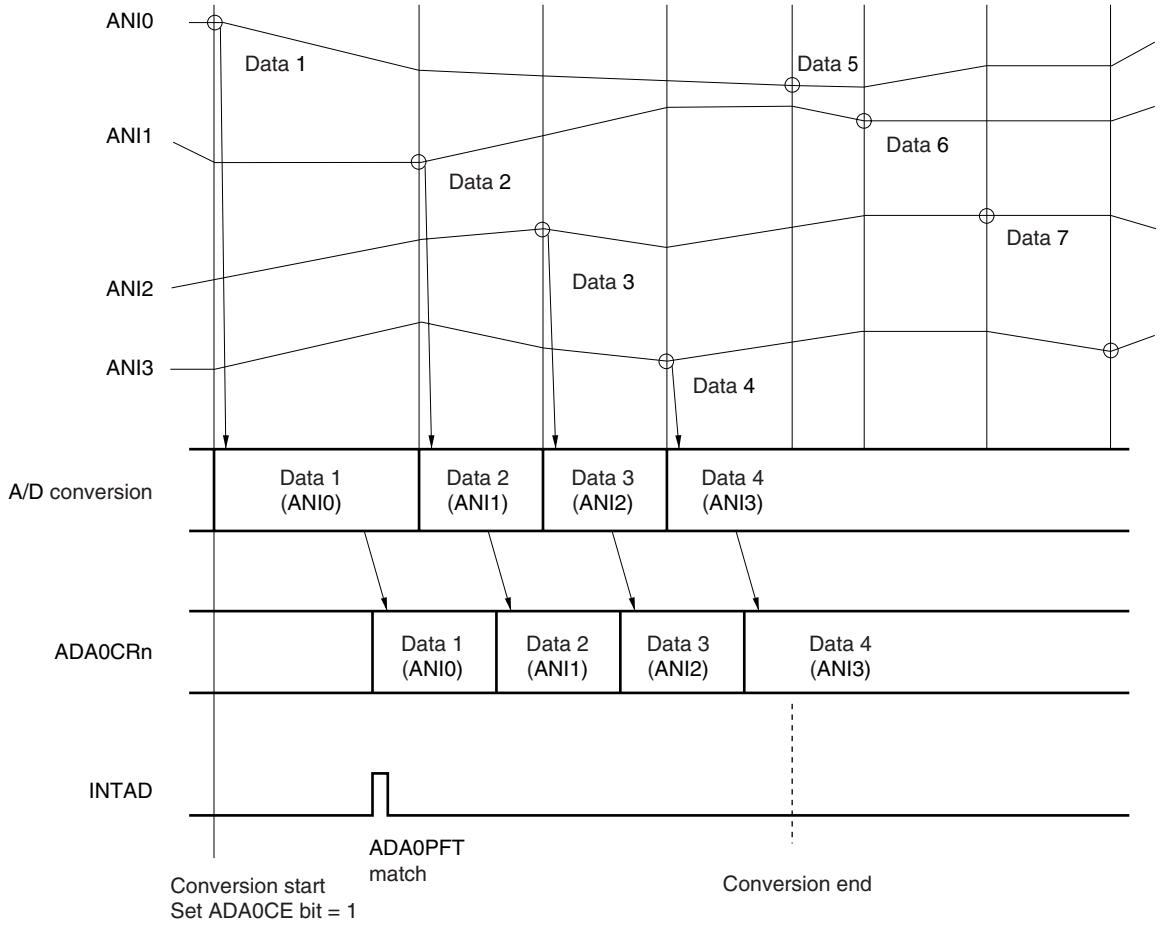


(4) One-shot scan mode

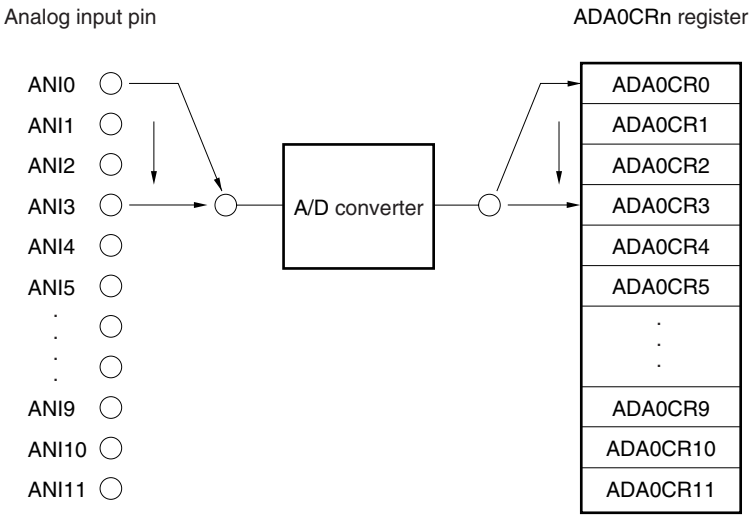
The voltages on the analog input pins selected by the ADA0S register are sequentially converted and the results of conversion are sequentially stored, starting from the ANI0 pin, and the set value of the ADA0CR0H register of channel 0 is compared with the set value of the ADA0PFT register. If the result of power-fail compare matches the condition set by the ADA0PFC bit of the ADA0PFM register, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD0 signal is not generated. After the result of the first conversion has been stored in the ADA0CR0 register, the results of converting the signals on the analog input pins specified by the ADA0S register are sequentially stored. The conversion is stopped after it has been completed (n = 0 to 11).

Figure 13-11. Timing Example of One-Shot Scan Mode Operation (When Power-Fail Compare Is Made: ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



13.5 Cautions

(1) When A/D converter is not used

When the A/D converter is not used, the power consumption can be reduced by clearing the ADA0CE bit of the ADA0M0 register to 0.

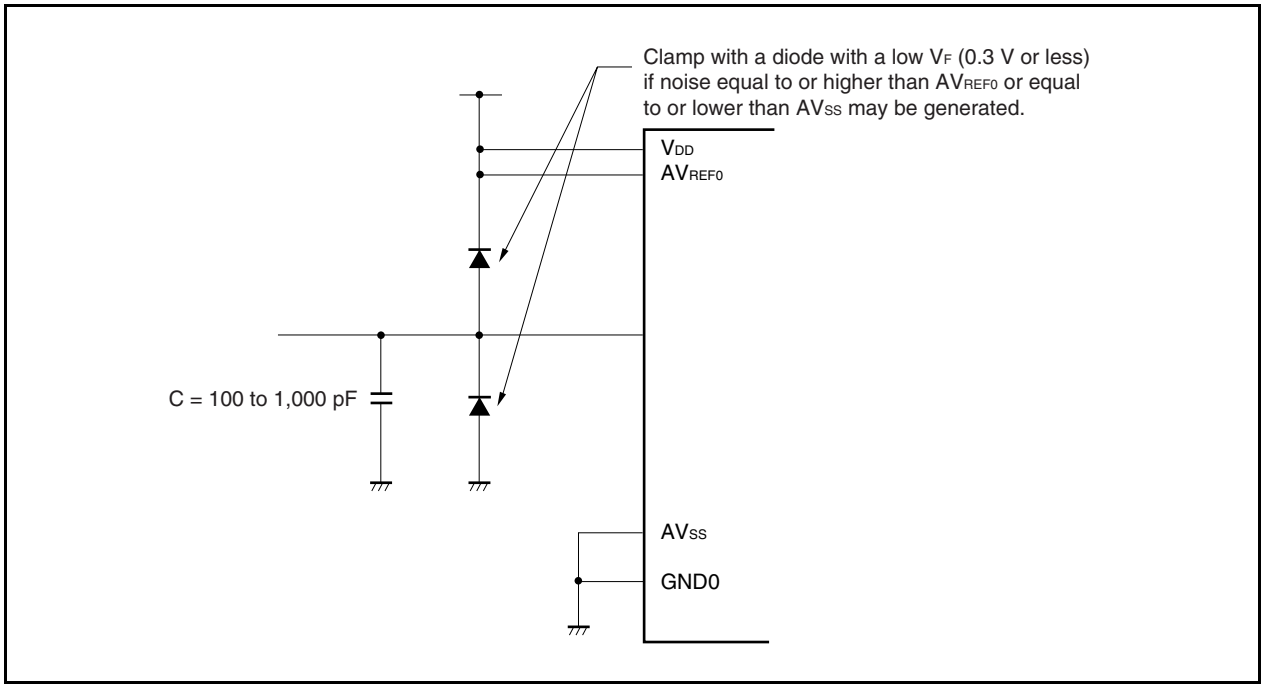
(2) Input range of ANI0 to ANI11 pins

Input the voltage within the specified range to the ANI0 to ANI11 pins. If a voltage equal to or higher than AV_{REF0} or equal to or lower than AV_{SS} (even within the range of the absolute maximum ratings) is input to any of these pins, the conversion value of that channel is undefined, and the conversion value of the other channels may also be affected.

(3) Countermeasures against noise

To maintain the 10-bit resolution, the ANI0 to ANI11 pins must be effectively protected from noise. The influence of noise increases as the output impedance of the analog input source becomes higher. To lower the noise, connecting an external capacitor as shown in Figure 13-12 is recommended.

Figure 13-12. Processing of Analog Input Pin



(4) Alternate I/O

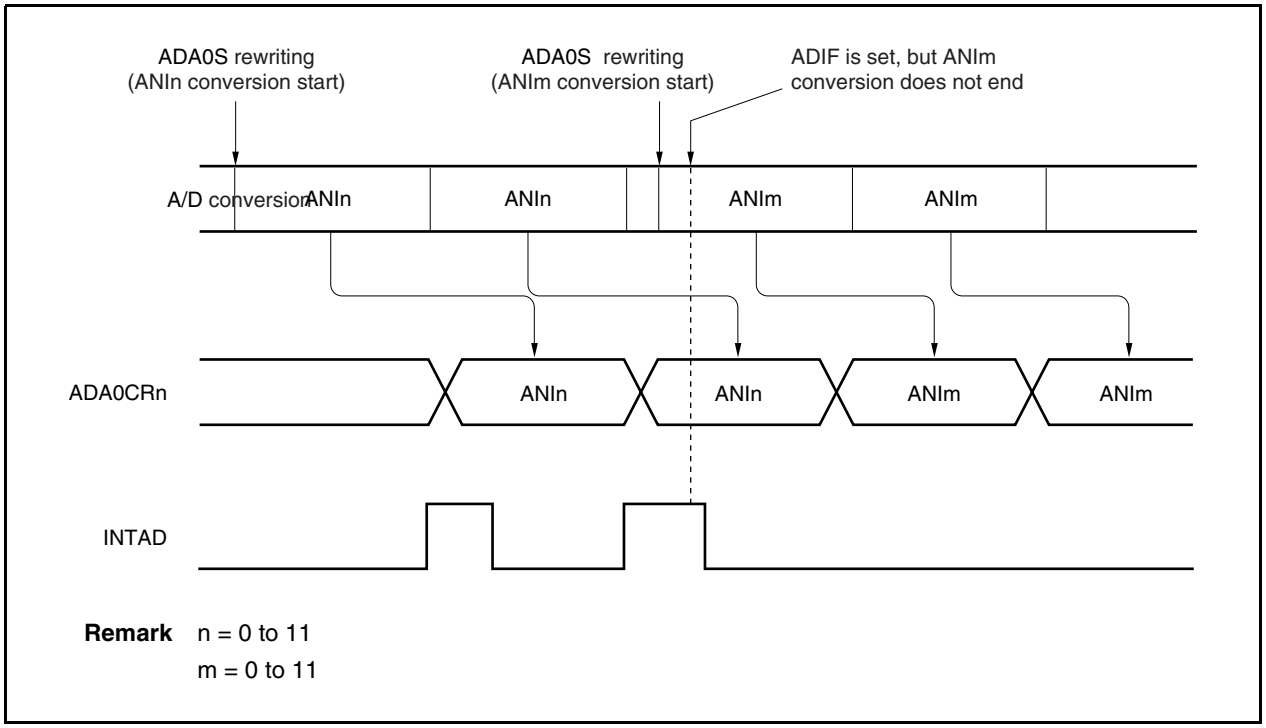
The analog input pins (ANI0 to ANI11) function alternately as port pins. When selecting one of the ANI0 to ANI11 pins to execute A/D conversion, do not execute an instruction to read an input port or write to an output port during conversion as the conversion resolution may drop.

If a digital pulse is applied to a pin adjacent to the pin whose input signal is being converted, the A/D conversion value may not be as expected due to the influence of coupling noise. Therefore, do not apply a pulse to the pin adjacent to the pin under A/D conversion.

(5) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the contents of the ADA0S register are changed. If the analog input pin is changed during A/D conversion, therefore, the result of converting the previously selected analog input signal may be stored and the conversion end interrupt request flag may be set immediately before the ADA0S register is rewritten. If the ADIF flag is read immediately after the ADA0S register is rewritten, the ADIF flag may be set even though the A/D conversion of the newly selected analog input pin has not been completed. When A/D conversion has been stopped, clear the ADIF flag before resuming conversion.

Figure 13-13. Generation Timing of A/D Conversion End Interrupt Request



Remark n = 0 to 11
m = 0 to 11

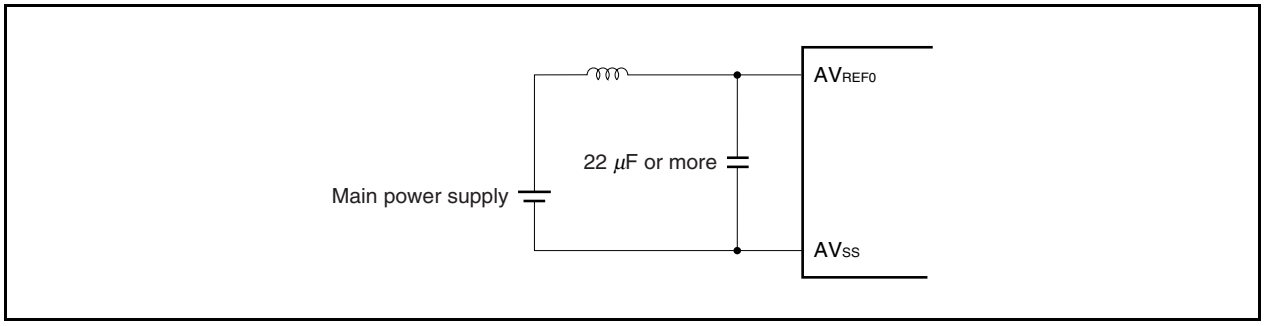
(6) AV_{REF0} pin

(a) The AV_{REF0} pin is also used as the power supply pin of the A/D converter and supplies power to the port with which it is multiplexed. In an application where a backup power supply is used, be sure to supply the same voltage as V_{DD} to the AV_{REF0} pin as shown in Figure 13-14.

(b) The AV_{REF0} pin is also used as the reference voltage pin of the A/D converter. If the source supplying power to the AV_{REF0} pin has a high impedance or if the power supply has a low current supply capability, the reference voltage may fluctuate due to the current that flows during conversion (especially, immediately after the conversion operation enable bit ADA0CE has been set to 1). As a result, the conversion accuracy may drop. To avoid this, it is recommended to connect a capacitor across the AV_{REF0} and AV_{SS} pins to suppress the reference voltage fluctuation.

(c) If the source supplying power to the AV_{REF0} pin has a high DC resistance (for example, because of insertion of a diode), the voltage when conversion is enabled may be lower than the voltage when conversion is stopped, because of a voltage drop caused by the A/D conversion current.

Figure 13-14. AV_{REF0} Pin Processing Example



(7) Reading ADA0CRn register

When the ADA0M0 to ADA0M2 or ADA0S register is written, the contents of the ADA0CRn register may be undefined. Read the conversion result after completion of conversion and before writing the ADA0M0 to ADA0M2 and ADA0S registers. The correct conversion result may not be read at a timing different from the above.

13.6 How to Read A/D Converter Characteristics Table

This section describes the terms related to the A/D converter.

(1) Resolution

The minimum analog input voltage that can be recognized, i.e., the ratio of an analog input voltage to 1 bit of digital output is called 1 LSB (least significant bit). The ratio of 1 LSB to the full scale is expressed as %FSR (full-scale range). %FSR is the ratio of a range of convertible analog input voltages expressed as a percentage, and can be expressed as follows, independently of the resolution.

1%FSR = (Maximum value of convertible analog input voltage – Minimum value of convertible analog input voltage)/100
= (AV_{REF0} – 0)/100
= AV_{REF0}/100

Where the resolution is 10 bits, 1 LSB is as follows:

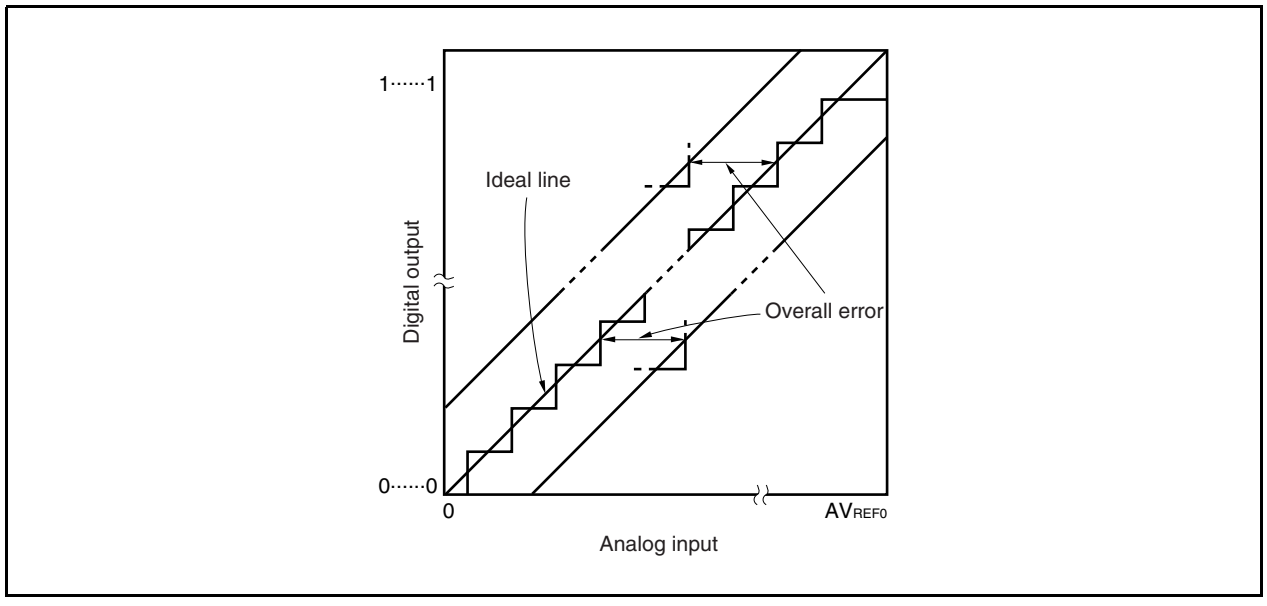
1 LSB = $1/2^{10}$ = 1/1,024
= 0.098%FSR

The accuracy is determined by the overall error, independently of the resolution.

(2) Overall error

This is the maximum value of the difference between an actually measured value and a theoretical value.
It is a total of zero-scale error, full-scale error, linearity error, and a combination of these errors.
The overall error in the characteristics table does not include the quantization error.

Figure 13-15. Overall Error

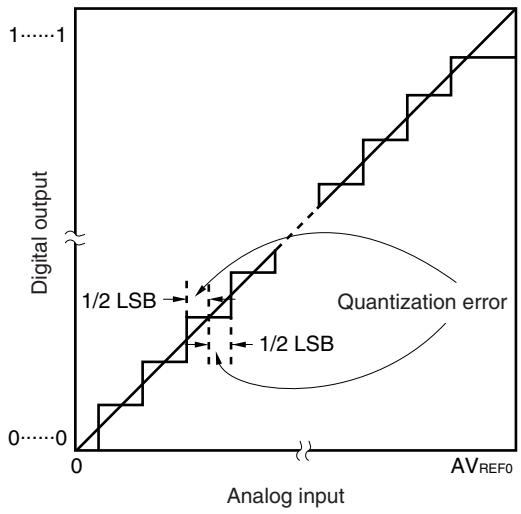


(3) Quantization error

This is an error of ±1/2 LSB that inevitably occurs when an analog value is converted into a digital value. Because the A/D converter converts analog input voltages in a range of ±1/2 LSB into the same digital codes, a quantization error is unavoidable.

This error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, or differential linearity error in the characteristics table.

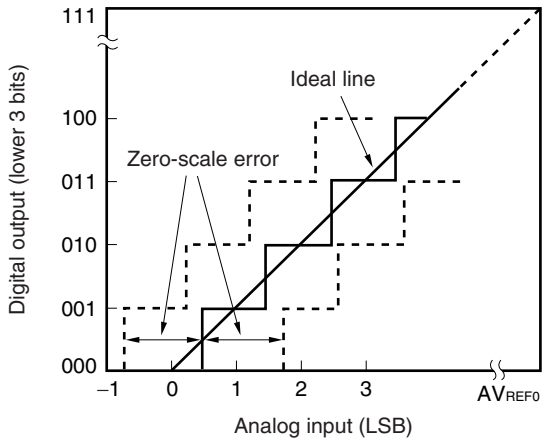
Figure 13-16. Quantization Error



(4) Zero-scale error

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 0…000 to 0…001 (1/2 LSB).

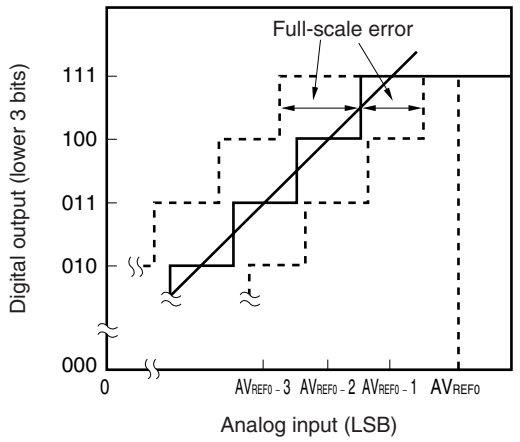
Figure 13-17. Zero-Scale Error



(5) Full-scale error

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 1…110 to 0…111 (full scale – 3/2 LSB).

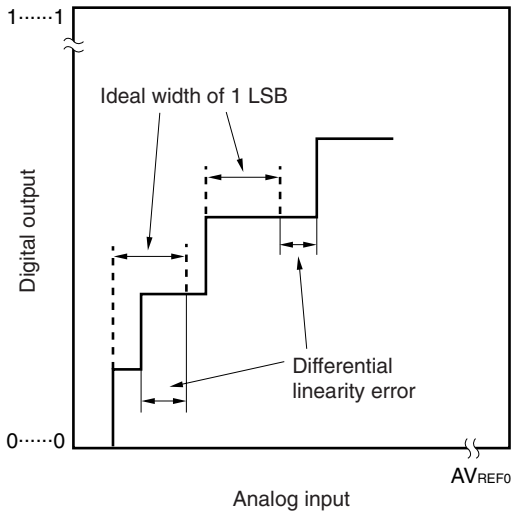
Figure 13-18. Full-Scale Error



(6) Differential linearity error

Ideally, the width to output a specific code is 1 LSB. This error indicates the difference between the actually measured value and its theoretical value when a specific code is output.

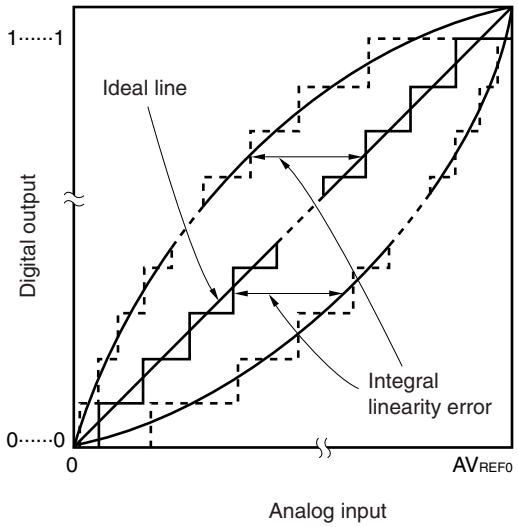
Figure 13-19. Differential Linearity Error



(7) Integral linearity error

This error indicates the extent to which the conversion characteristics differ from the ideal linear relations. It indicates the maximum value of the difference between the actually measured value and its theoretical value where the zero-scale error and full-scale error are 0.

Figure 13-20. Integral Linearity Error



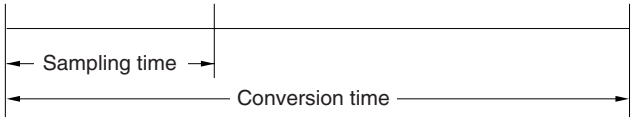
(8) Conversion time

This is the time required to obtain a digital output after an analog input voltage has been assigned.
The conversion time in the characteristics table includes the sampling time.

(9) Sampling time

This is the time for which the analog switch is ON to load an analog voltage to the sample & hold circuit.

Figure 13-21. Sampling Time



CHAPTER 14 D/A CONVERTER

14.1 Functions

The D/A converter has the following functions.

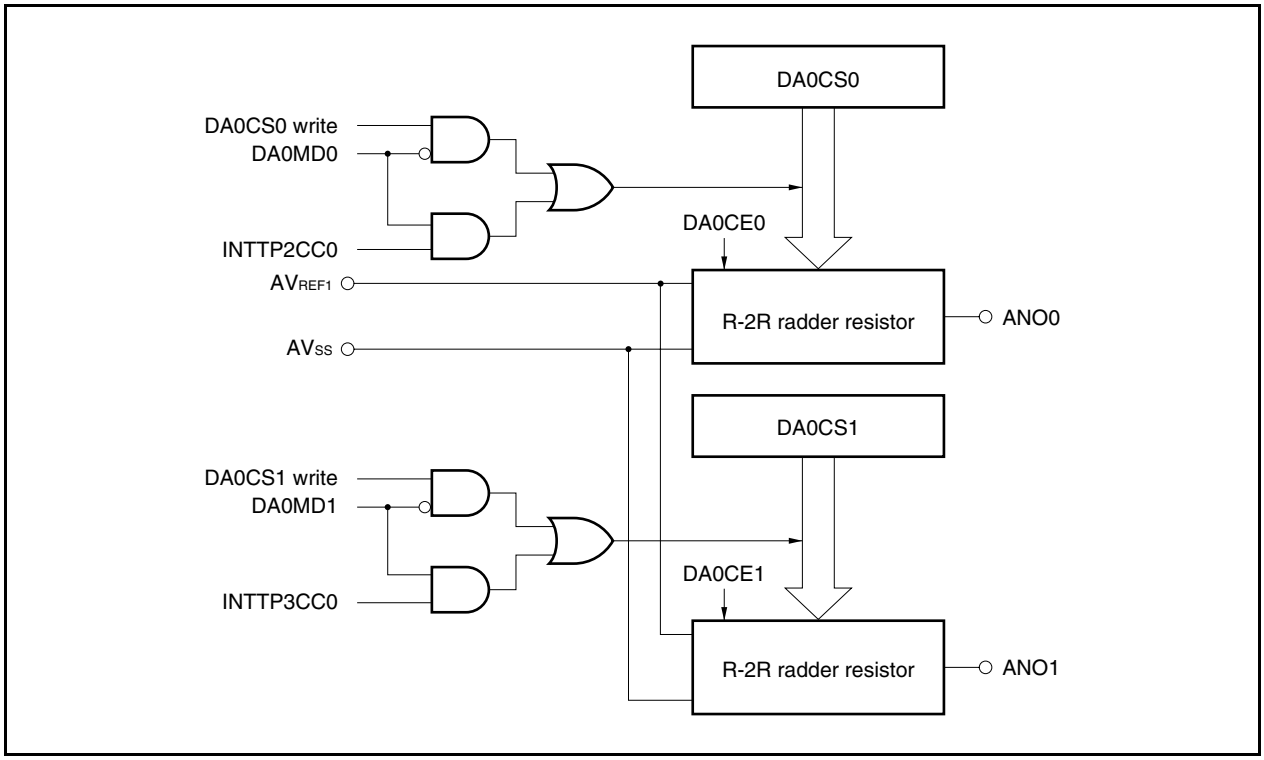
- 8-bit resolution × 2 channels (DA0CS0, DA0CS1)
- R-2R radder method
- Conversion time: 20 μs max. (AV_{REF1} = 3.0 to 3.6 V)
- Analog output voltage: AV_{REF1} × m/256 (m = 0 to 255; value set to DA0CSn register)
- Operation modes: Normal mode, real-time output mode

Remark n = 0, 1

14.2 Configuration

The D/A converter configuration is shown below.

Figure 14-1. Block Diagram of D/A Converter



The D/A converter consists of the following hardware.

Table 14-1. Configuration of D/A Converter

Item	Configuration
Control registers	D/A converter mode register (DA0M) D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

14.3 Control Registers

The registers that control the D/A converter are as follows.

- D/A converter mode register (DA0M)
- D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

(1) D/A converter mode register (DA0M)

The DA0M register controls the operation of the D/A converter.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF282H

	7	6	<5>	<4>	3	2	1	0
DA0M	0	0	DA0CE1	DA0CE0	0	0	DA0MD1	DA0MD0

DA0CEn	Control of D/A converter operation enable/disable (n = 0, 1)
0	Disables operation
1	Enables operation

DA0MDn	Selection of D/A converter operation mode (n = 0, 1)
0	Normal mode
1	Real-time output mode[Note]

Note The output trigger in the real-time output mode (DA0MDn bit = 1) is as follows.

- When n = 0: INTTP2CC0 signal (see **CHAPTER 7 16-BIT TIMER/EVENT COUNTER P**)
- When n = 1: INTTP3CC0 signal (see **CHAPTER 7 16-BIT TIMER/EVENT COUNTER P**)

(2) D/A conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

The DA0CS0 and DA0CS1 registers set the analog voltage value output to the ANO0 and ANO1 pins.

These registers can be read or written in 8-bit units.

Reset input clears these registers to 00H.

After reset: 00H R/W Address: DA0CS0 FFFFF280H, DA0CS1 FFFFF281H

	7	6	5	4	3	2	1	0
DA0CSn	DA0CSn7	DA0CSn6	DA0CSn5	DA0CSn4	DA0CSn3	DA0CSn2	DA0CSn1	DA0CSn0

Caution In the real-time output mode (DA0MDn bit = 1), set the DA0CSn register before the INTTP2CC0/INTTP3CC0 signals are generated. D/A conversion starts when the INTTP2CC0/INTTP3CC0 signals are generated.

Remark n = 0, 1

14.4 Operation

14.4.1 Operation in normal mode

D/A conversion is performed using a write operation to the DA0CSn register as the trigger.

The setting method is described below.

<1> Set the DA0MDn bit of the DA0M register to 0 (normal mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

Steps <1> and <2> above constitute the initial settings.

<3> Set the DA0CEn bit of the DA0M register to 1 (D/A conversion enable).

D/A conversion starts when this setting is performed.

<4> To perform subsequent D/A conversions, write to the DA0CSn register.

The previous D/A conversion result is held until the next D/A conversion is performed.

Remark n = 0, 1

14.4.2 Operation in real-time output mode

D/A conversion is performed using the interrupt request signals (INTTP2CC0 and INTTP3CC0) of 16-bit timer/event counters P2 and P3 (TMP2 and TMP3) as triggers.

The setting method is described below.

<1> Set the DA0MDn bit of the DA0M register to 1 (real-time output mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

<3> Set the DA0CEn bit of the DA0M register to 1 (D/A conversion enable).

Steps <1> to <3> above constitute the initial settings.

<4> Operate TMP2 and TMP3.

<5> D/A conversion starts when the INTTP2CC0 and INTTP3CC0 signals are generated.

<6> The INTTP2CC0 and INTTP3CC0 signals are generated when subsequent D/A conversions are performed.

Before performing the next D/A conversion (generation of INTTP2CC0 and INTTP3CC0 signals), set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

14.4.3 Cautions

Observe the following cautions when using the D/A converter of the V850ES/SG2.

(1) Do not change the set value of the DA0CSn register while the trigger signal is being issued in the real-time output mode.

(2) Before changing the operation mode, be sure to clear the DA0CEn bit of the DA0M register to 0.

(3) When using the P10/ANO0 and P11/ANO1 pins as port pins, make sure that their input level does not change much.

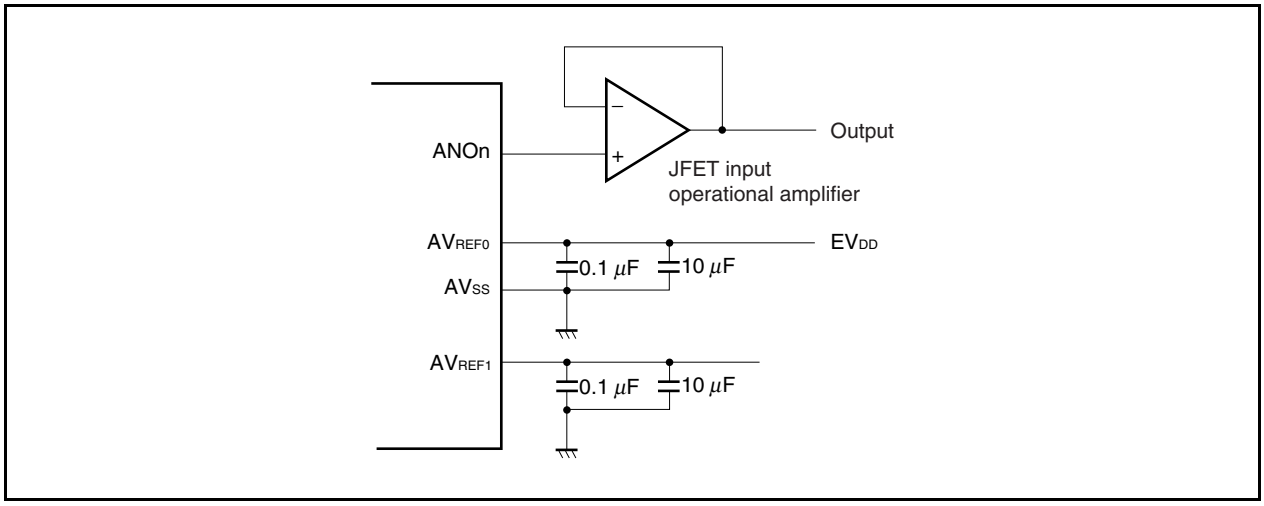
(4) Make sure that $AV_{REF0} = EV_{DD} = AV_{REF1}$ = 3.0 to 3.6 V. If this range is exceeded, the operation is not guaranteed.

(5) Apply power to AV_{REF1} at the same timing as AV_{REF0}.

(6) No current can be output from the ANOn pin (n = 0, 1) because the output impedance of the D/A converter is high. When connecting a resistor of 5 MΩ or less, insert a JFET input operational amplifier between the resistor and the ANOn pin.

(7) Do not perform D/A conversion of two channels at the same time. Otherwise, the conversion accuracy may drop.

Figure 14-2. External Pin Connection Example



15.1 Mode Switching of UARTA and Other Serial Interfaces

15.1.1 CSIB4 and UARTA0 mode switching

In the V850ES/SG2, CSIB4 and UARTA0 are alternate functions of the same pin and therefore cannot be used simultaneously. CSIB4 and UARTA0 switching must be set in advance using the PMC3 and PFC3 registers.

Caution The transmit/receive operation of CSIB4 and UARTA0 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 15-1. CSIB4 and UARTA0 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF446H, FFFFF447H

	15	14	13	12	11	10	9	8
PMC3	0	0	0	0	0	0	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

After reset: 0000H R/W Address: FFFFF466H, FFFF467H

	15	14	13	12	11	10	9	8
PFC3	0	0	0	0	0	0	PFC39	PFC38

7	6	5	4	3	2	1	0
PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

After reset: 00H R/W Address: FFFFF706H

	7	6	5	4	3	2	1	0
PFCE3L	0	PFCE36	PFCE35	PFCE34	PFCE33	PFCE32	0	0

PMC32	PFCE32	PFC32	Operation mode
0	×	×	Port I/O mode
1	0	0	ASCKA0 mode
1	0	1	$\overline{\text{SCKB4}}$ mode

PMC3n	PFC3n	Operation mode
0	×	Port I/O mode
1	0	UARTA0 mode
1	1	CSIB4 mode

Remarks 1. n = 0, 1

2. × = don't care

15.1.2 UARTA2 and I²C00 mode switching

In the V850ES/SG2, UARTA2 and I²C00 are alternate functions of the same pin and therefore cannot be used simultaneously. UARTA2 and I²C00 switching must be set in advance using the PMC3 and PFC3 registers.

Caution The transmit/receive operation of UARTA2 and I²C00 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 15-2. UARTA2 and I²C00 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF446H, FFFFF447H

	15	14	13	12	11	10	9	8
PMC3	0	0	0	0	0	0	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

After reset: 0000H R/W Address: FFFFF466H, FFFF467H

	15	14	13	12	11	10	9	8
PFC3	0	0	0	0	0	0	PFC39	PFC38

7	6	5	4	3	2	1	0
PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

PMC3n	PFC3n	Operation mode
0	×	Port I/O mode
1	0	UARTA2 mode
1	1	I²C00 mode

Remarks 1. n = 8, 9

2. × = don't care

15.1.3 UARTA1 and I²C02 mode switching

In the V850ES/SG2, UARTA1 and I²C02 are alternate functions of the same pin and therefore cannot be used simultaneously. UARTA1 and I²C02 switching must be set in advance using the PMC9, PFC9, and PMCE9 registers.

Caution The transmit/receive operation of UARTA1 and I²C02 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 15-3. UARTA1 and I²C02 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF452H, FFFFF453H

PMC9

15	14	13	12	11	10	9	8
PMC915	PMC914	PMC913	PMC912	PMC911	PMC910	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC97	PMC96	PMC95	PMC94	PMC93	PMC92	PMC91	PMC90

After reset: 0000H R/W Address: FFFFF472H, FFFF473H

PFC9

15	14	13	12	11	10	9	8
PFC915	PFC914	PFC913	PFC912	PFC911	PFC910	PFC99	PFC98

7	6	5	4	3	2	1	0
PFC97	PFC96	PFC95	PFC94	PFC93	PFC92	PFC91	PFC90

After reset: 0000H R/W Address: FFFFF712H, FFFFF713

PFCE9

15	14	13	12	11	10	9	8
PFCE915	PFCE914	0	0	0	0	0	0

7	6	5	4	3	2	1	0
PFCE97	PFCE96	PFCE95	PFCE94	PFCE93	PFCE92	PFCE91	PFCE90

PMC9n	PFCE9n	PFC9n	Operation mode
1	1	0	UARTA1 mode
1	1	1	I²C02 mode

Remark n = 0, 1

15.2 Features

- Transfer speed: 300 bps to 312.5 kbps (using internal system clock of 20 MHz and dedicated baud rate generator)
- Full-duplex communication: Internal UARTA receive data register n (UAnRX)
  Internal UARTA transmit data register n (UAnTX)
- 2-pin configuration: TXDAn: Transmit data output pin
  RXDAn: Receive data input pin
- Receive error output function
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources: 2
  - Reception complete end interrupt (INTUAnR): This interrupt is generated upon transfer of receive data from the shift register to receive buffer register n when an interrupt is generated by ORing three types of reception errors or after serial transfer completion, in the reception enabled status.
  - Transmission enable interrupt (INTUAnT): This interrupt is generated upon transfer of transmit data from the transmit buffer register to the shift register in the transmission enabled status.
- Character length of transmit/receive data specified by the UAnCTL0 register
- Character length: 7, 8 bits
- Parity function: Odd, even, 0, none
- Transmission stop bit: 1, 2 bits
- On-chip dedicated baud rate generator
- MSB/LSB-first transfer selectable
- Transmit/receive data inversion possible
- 13 to 20 bits selectable for the SBF (Sync Break Field) in the LIN (Local Interconnect Network) communication format
- Recognition of 11 bits or more possible for SBF reception in LIN communication format
- SBF reception flag provided

Remark n = 0 to 2

15.3 Configuration

(1) UARTAn control register 0 (UAnCTL0)

The UAnCTL0 register is an 8-bit register used to specify the asynchronous serial interface operation.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is an 8-bit register used to select the input clock for the asynchronous serial interface.

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is an 8-bit register used to control the baud rate for the asynchronous serial interface.

(4) UARTAn option control register 0 (UAnOPT0)

The UAnOPT0 register is an 8-bit register used to control serial transfer for the asynchronous serial interface.

(5) UARTAn status register (UAnSTR)

The UAnSTRn register consists of flags indicating the error contents when a reception error occurs. Each one of the reception error flags is set (to 1) upon occurrence of a reception error and is reset (to 0) by reading the UAnSTR register.

(6) UARTAn receive shift register

This is a shift register used to convert the serial data input to the RXDAn pin into parallel data. Upon reception of 1 byte of data and detection of the stop bit, the receive data is transferred to the UAnRX register.

This register cannot be manipulated directly.

(7) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit register that holds receive data. When 7 characters are received, 0 is stored in the highest bit (when LSB first received).

In the reception enabled status, receive data is transferred from the UARTAn receive shift register to the UAnRX register in synchronization with the completion of shift-in processing of 1 frame.

Transfer to the UAnRX register also causes reception complete interrupt request signal (INTUAnR) to be output.

(8) UARTAn transmit shift register

The transmit shift register is a shift register used to convert the parallel data transferred from the UAnTX register into serial data.

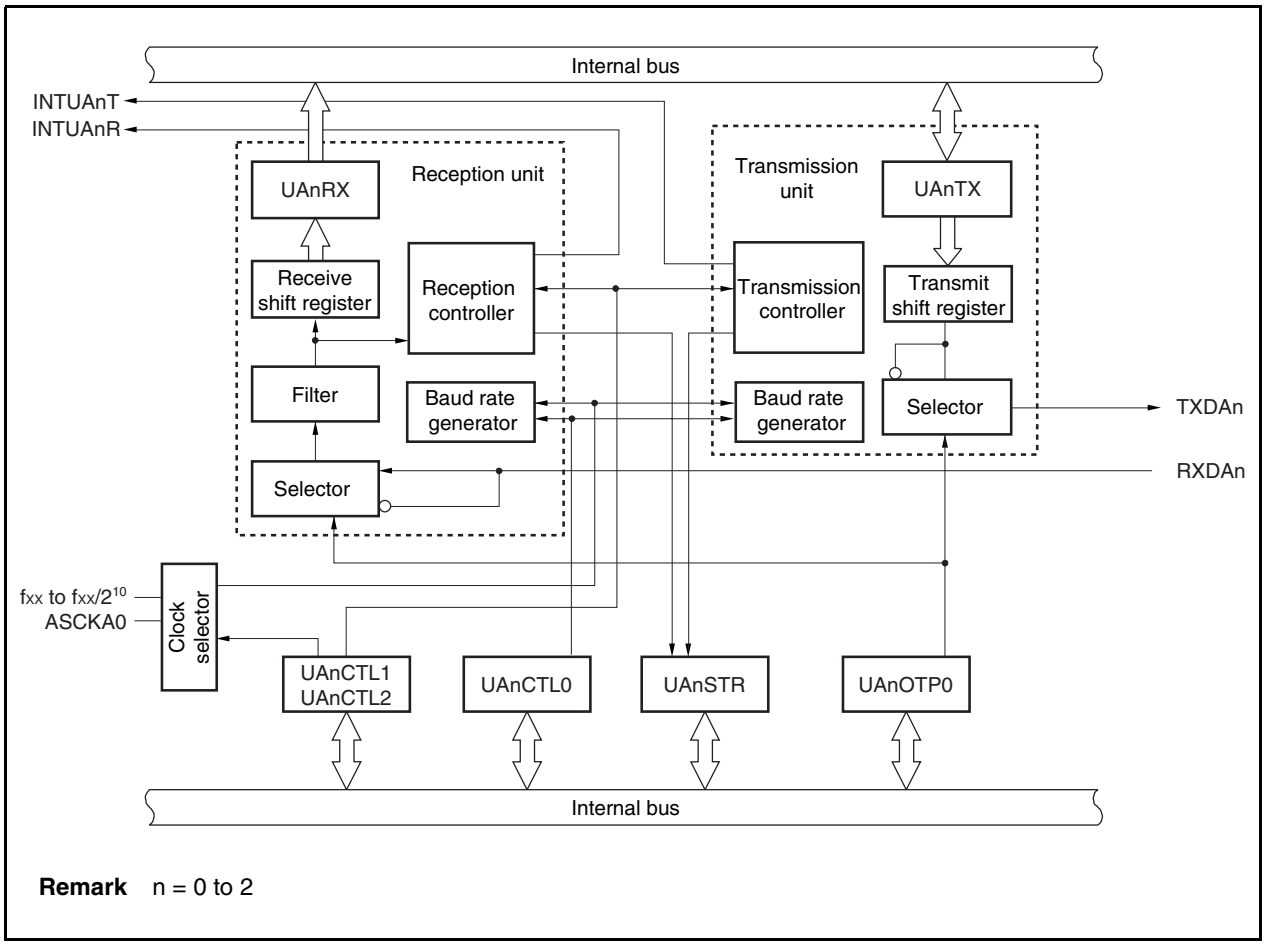
When 1 byte of data is transferred from the UAnTX register, the shift register data is output from the TXDAn pin.

This register cannot be manipulated directly.

(9) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit transmit data buffer. Transmission starts when transmit data is written to the UAnTX register. When data can be written to the UAnTX register (when data of one frame is transferred from the UAnTX register to the UARTn transmit shift register), the transmission enable interrupt request signal (INUAnT) is generated.

Figure 15-4. Block Diagram of Asynchronous Serial Interface n



Remark n = 0 to 2

15.4 Control Registers

(1) UARTAn control register 0 (UAnCTL0)

The UAnCTL0 register is an 8-bit register that controls the UARTAn serial transfer operation.

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

Reset input sets this register to 10H.

Caution Be sure to set the UAnPWR bit = 1 and the UAnRXE bit = 1 while the RXDAn pin is high level (when UAnRDL bit of UAnOP0 register = 0).

If the UAnPWR bit = 1 and the UAnRXE bit = 1 are set while the RXDAn pin is low level, reception will inadvertently start.

After reset: 10H R/W Address: UA0CTL0 FFFFFA00H, UA1CTL0 FFFFFA10H, UA2CTL0 FFFFFA20H

UAnCTL0 (n = 0 to 2)

<7>	<6>	<5>	<4>	3	2	1	0
UAnPWR	UAnTXE	UAnRXE	UAnDIR	UAnPS1	UAnPS0	UAnCL	UAnSL

UAnPWR	UARTAn operation control
0	Disable operating clock operation (UARTAn reset asynchronously)
1	Enable operating clock operation
Operating clock control and UARTAn asynchronous reset are performed with the UAnPWR bit. The TXDAn pin output is fixed to high level by setting the UAnPWR bit to 0.	

UAnTXE	Transmission operation enable
0	Stop transmission operation
1	Enable transmission operation
The TXDAn pin output is fixed to high level by setting the UAnTXE bit to 0. Since the UAnTXE bit is initialized by the operating clock, to initialize the transmission unit, set the UAnTXE bit from 0 to 1, and 2 clocks later, the transmission enabled status is entered. When the UAnPWR bit = 0, the value written to the UAnTXE bit is ignored.	

UAnRXE	Reception operation enable
0	Stop reception operation
1	Enable reception operation
The receive operation is stopped by setting the UAnRXE bit to 0. Therefore, even if the prescribed data is transferred, no reception completion interrupt is output and the UAnRX register is not updated. Since the UAnRXE bit is synchronized using the operating clock, to initialize the reception unit, set the UAnRXE bit from 0 to 1, and 2 clocks later, the reception enabled status is entered. When the UAnPWR bit = 0, the value written to the UAnRXE bit is ignored.	

(2/2)

UAnDIR	Transfer direction selection
0	MSB-first transfer
1	LSB-first transfer
This register can be rewritten only when the UAnPWR bit = 0 or the UAnTXE bit = the UAnRXE bit = 0.	

UAnPS1	UAnPS0	Parity selection during transmission	Parity selection during reception
0	0	No parity output	Reception with no parity
0	1	0 parity output	Reception with 0 parity
1	0	Odd parity output	Odd parity check
1	1	Even parity output	Even parity check

- This register is rewritten only when the UAnPWR bit = 0 or the UAnTXE bit = the UAnRXE bit = 0.
- If “Reception with 0 parity” is selected during reception, a parity check is not performed. Therefore, since the UAnPE bit of the UAnSTR register is not set, no error interrupt is output.
- When transmission and reception are performed in the LIN format, set the UAnPS1 and UAnPS0 bits to 00.

UAnCL	Specification of data character length of 1 frame of transmit/receive data
0	7 bits
1	8 bits
This register can be rewritten only when the UAnPWR bit = 0 or the UAnTXE bit = the UAnRXE bit = 0.	

UAnSL	Specification of length of stop bit for transmit data
0	1 bit
1	2 bits
This register can be rewritten only when the UAnPWR bit = 0 or the UAnTXE bit = the UAnRXE bit = 0.	

Remark For details of parity, see **15.6.9 Parity types and operations**.

(2) UARTAn control register 1 (UAnCTL1)

For details, see **15.7 (2) UARTAn control register 1 (UAnCTL1)**.

(3) UARTAn control register 2 (UAnCTL2)

For details, see **15.7 (3) UARTAn control register 2 (UAnCTL2)**.

(4) UARTAn option control register 0 (UAnOPT0)

The UAnOPT0 register is an 8-bit register that controls the serial transfer operation of the UARTAn register.

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

Reset input sets this register to 14H.

(1/2)

After reset: 14H R/W Address: UA0OPT0 FFFFFA03H, UA1OPT0 FFFFFA13H, UA2OPT0 FFFFFA23H

	<7>	6	5	4	3	2	1	0
UAnOPT0 (n = 0 to 2)	UAnSRF	UAnSRT	UAnSTT	UAnSLS2	UAnSLS1	UAnSLS0	UAnTDL	UAnRDL

UAnSRF	SBF reception flag
0	When the UAnPWR bit = the UAnRXE bit of the UAnCTL0 register = 0 are set. Also upon normal end of SBF reception.
1	During SBF reception

- SBF (Sync Brake Field) reception is judged during LIN communication.
- The UAnSRF bit is held high when an SBF reception error occurs, and then SBF reception is started again.

UAnSRT	SBF reception trigger
0	–
1	SBF reception trigger

- This is the SBF reception trigger bit during LIN communication, and when read, “0” is always read. For SBF reception, set the UAnSRT bit (to 1) to enable SBF reception.
- Set the UAnSRT bit after setting the UAnPWR bit = the UAnRXE bit of the UAnCTL0 register = 1.

UAnSTT	SBF transmission trigger
0	–
1	SBF transmission trigger

- This is the SBF transmission trigger bit during LIN communication, and when read, “0” is always read.
- Set the UAnSTT bit after setting the UAnPWR bit = the UAnRXE bit of the UAnCTL0 register = 1.

(2/2)

UAnSLS2	UAnSLS1	UAnSLS0	SBF length selection
1	0	1	13-bit output (reset value)
1	0	0	14-bit output
1	1	1	15-bit output
0	1	0	16-bit output
0	0	1	17-bit output
0	0	0	18-bit output
0	1	1	19-bit output
1	1	0	20-bit output
This register can be set when the UAnPWR bit of the UAnCTL0 register = 0 or when the UAnRXE bit of the UAnCTL0 register = 0.			

UAnTDL	Transmit data level bit
0	Normal output of transfer data
1	Inverted output of transfer data
• The value of the TXDAn pin can be inverted using the UAnTDL bit. • This register can be set when the UAnPWR bit of the UAnCTL0 register = 0 or when the UAnTXE bit of the UAnCTL0 register = 0.	

UAnRDL	Receive data level bit
0	Normal input of transfer data
1	Inverted input of transfer data
• The value of the RXDAn pin can be inverted using the UAnRDL bit. • This register can be set when the UAnPWR bit of the UAnCTL0 register = 0 or the UAnRXE bit of the UAnCTL0 register = 0.	

(5) UARTAn status register (UAnSTR)

The UAnSTR register is an 8-bit register that displays the UARTAn transfer status and reception error contents. This register can be read or written in 8-bit or 1-bit units, but the UAnTSF bit is a read-only bit, while the UAnPE, UAnFE, and UAnOVE bits can both be read and written. However, these bits can only be cleared by writing 0 and they cannot be set by writing 1. (If 1 is written to them, the hold status is entered.)

The initialization conditions are shown below.

Register/Bit	Initialization Conditions
UAnSTR register	• Reset input • UAnPWR bit of UAnCTL0 register = 0
UAnTSF bit	• UAnTXE bit of UAnCTL0 register = 0
UAnPE, UAnFE, UAnOVE bits	• 0 write • UAnRXE bit of UAnCTL0 register = 0

After reset: 00H R/W Address: UA0STR FFFFFA04H, UA1STR FFFFFA14H, UA2STR FFFFFA24H

UAnSTR (n = 0 to 2)

<7>	6	5	4	3	<2>	<1>	<0>
UAnTSF	0	0	0	0	UAnPE	UAnFE	UAnOVE

UAnTSF	Transfer status flag
0	• When the UAnPWR bit of the UAnCTL0 register = 0 or the UAnTXE bit of the UAnCTL0 register = 0 has been set. • When, following transfer completion, there was no next data transfer from UAnTX
1	Write to UAnTXB bit
The UAnTSF bit is always 1 when performing continuous transmission. When initializing the transmission unit, check that the UAnTSF bit = 0 before performing initialization. The transmit data is not guaranteed when initialization is performed while the UAnTSF bit = 1.	

UAnPE	Parity error flag
0	• When the UAnPWR bit of the UAnCTL0 register = 0 or the UAnRXE bit of the UAnCTL0 register = 0 has been set. • When 0 has been written
1	• When parity of data and parity bit do not match during reception.
• The operation of the UAnPE bit is controlled by the settings of the UAnPS1 and UAnPS0 bits of the UAnCTL0 register. • The UAnPE bit can be read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the hold status is entered.	

UAnFE	Framing error flag
0	• When the UAnPWR bit of the UAnCTL0 register = 0 or the UAnRXE bit of the UAnCTL0 register = 0 has been set • When 0 has been written
1	When no stop bit is detected during reception
• Only the first bit of the receive data stop bits is checked, regardless of the value of the UAnSL bit of the UAnCTL0 register. • The UAnFE bit can be both read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the hold status is entered.	

UAnOVE	Overrun error flag
0	• When the UAnPWR bit of the UAnCTL0 register = 0 or the UAnRXE bit of the UAnCTL0 register = 0 has been set. • When 0 has been written
1	When receive data has been set to the UAnRXB register and the next receive operation is completed before that receive data has been read
• When an overrun error occurs, the data is discarded without the next receive data being written to the receive buffer. • The UAnOVE bit can be both read and written, but it can only be cleared by writing 0 to it. When 1 is written to this bit, the hold status is entered.	

(6) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit buffer register that stores parallel data converted by receive shift register.

The data stored in the receive shift register is transferred to the UAnRX register upon completion of reception of 1 byte of data.

During LSB-first reception when the data length has been specified as 7 bits, the receive data is transferred to bits 6 to 0 of the UAnRX register and the MSB always becomes 0. During MSB-first reception, the receive data is transferred to bits 7 to 1 of the UAnRX register and the LSB always becomes 0.

When an overrun error (UAnOVE) occurs, the receive data at this time is not transferred to the UAnRX register.

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

In addition to reset input, the UAnRX register can be set to FFH by clearing the UAnPWR bit of the UAnCTL0 register to 0.

After reset: FFH R Address: UA0RX FFFFFA06H, UA1RX FFFFFA16H, UA2RX FFFFFA26H

	7	6	5	4	3	2	1	0
UAnRX								

(n = 0 to 2)

(7) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit register used to set transmit data.

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

Reset input sets this register to FFH.

After reset: FFH R/W Address: UA0TX FFFFFA07H, UA1TX FFFFFA17H, UA2TX FFFFFA27H

	7	6	5	4	3	2	1	0
UAnTX								

(n = 0 to 2)

15.5 Interrupt Request Signals

The following two interrupt request signals are generated from UARTAn.

- Reception complete interrupt request signal (INTUAnR)
- Transmission enable interrupt request signal (INTUAnT)

The default priority for these two interrupt request signals is highest for the reception complete interrupt request signal.

Table 15-1. Interrupts and Their Default Priorities

Interrupt	Priority
Reception complete	High
Transmission enable	Low

(1) Reception complete interrupt request signal (INTUAnR)

A reception complete interrupt request signal is output when data is shifted into the receive shift register and transferred to the UAnRX register in the reception enabled status.

A reception complete interrupt request signal can be generated instead of a reception error interrupt even when a reception error has occurred.

When a reception complete interrupt request signal is received and the data is read, read the UAnSTR register and check that the reception result is not an error.

No reception complete interrupt request signal is generated in the reception disabled status.

(2) Transmission enable interrupt request signal (INTUAnT)

If transmit data is transferred from the UAnTX register to the UARTAn transmit shift register with transmission enabled, the transmission enable interrupt request signal is generated.

15.6 Operation

15.6.1 Data format

Full-duplex serial data reception and transmission is performed.

As shown in Figure 15-5, one data frame of transmit/receive data consists of a start bit, character bits, parity bit, and stop bit(s).

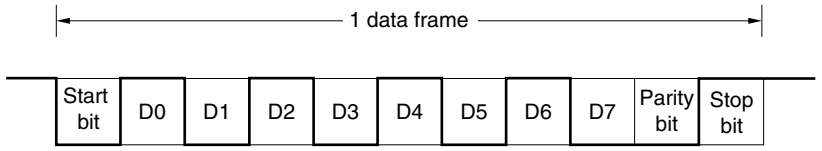
Specification of the character bit length within 1 data frame, parity selection, specification of the stop bit length, and specification of MSB/LSB-first transfer are performed using the UAnCTL0 register.

Moreover, control of UART output/inverted output for the TXDAn bit is performed using the UAnTDL bit of the UAnOPT0 register.

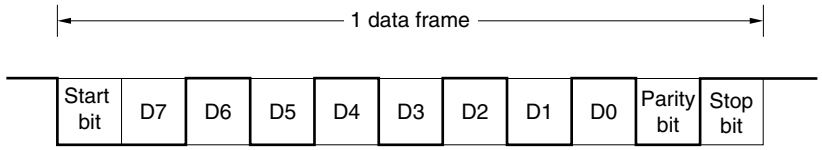
- Start bit.....1 bit
- Character bits.....7 bits/8 bits
- Parity bit .....Even parity/odd parity/0 parity/no parity
- Stop bit .....1 bit/2 bits

Figure 15-5. UARTA Transmit/Receive Data Format

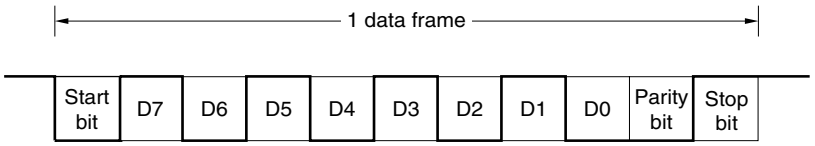
(a) 8-bit data length, LSB first, even parity, 1 stop bit, transfer data: 55H



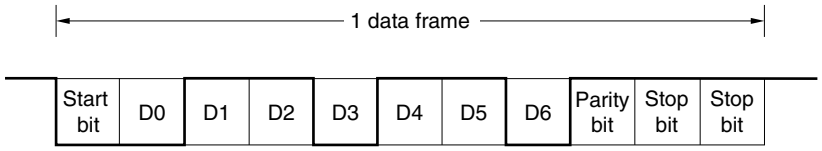
(b) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H



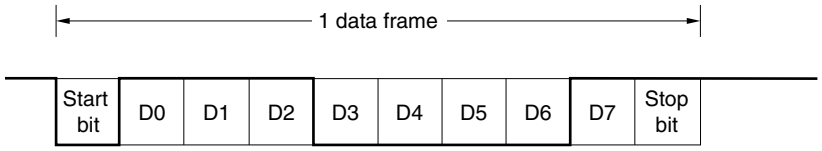
(c) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H, TXDAn inversion



(d) 7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36H



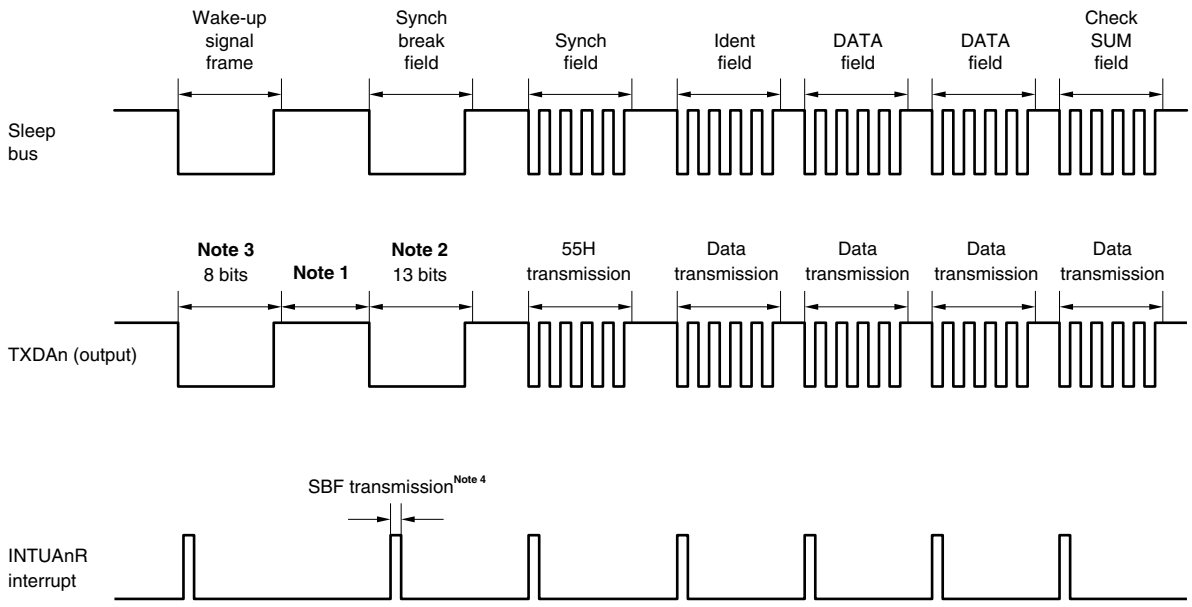
(e) 8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87H



15.6.2 SBF transmission/reception format

The V850ES/SG2 has an SBF (Sync Break Field) transmission/reception control function to enable use of the LIN (Local Interconnect Network) function.

Figure 15-6. LIN Transmission Manipulation Outline



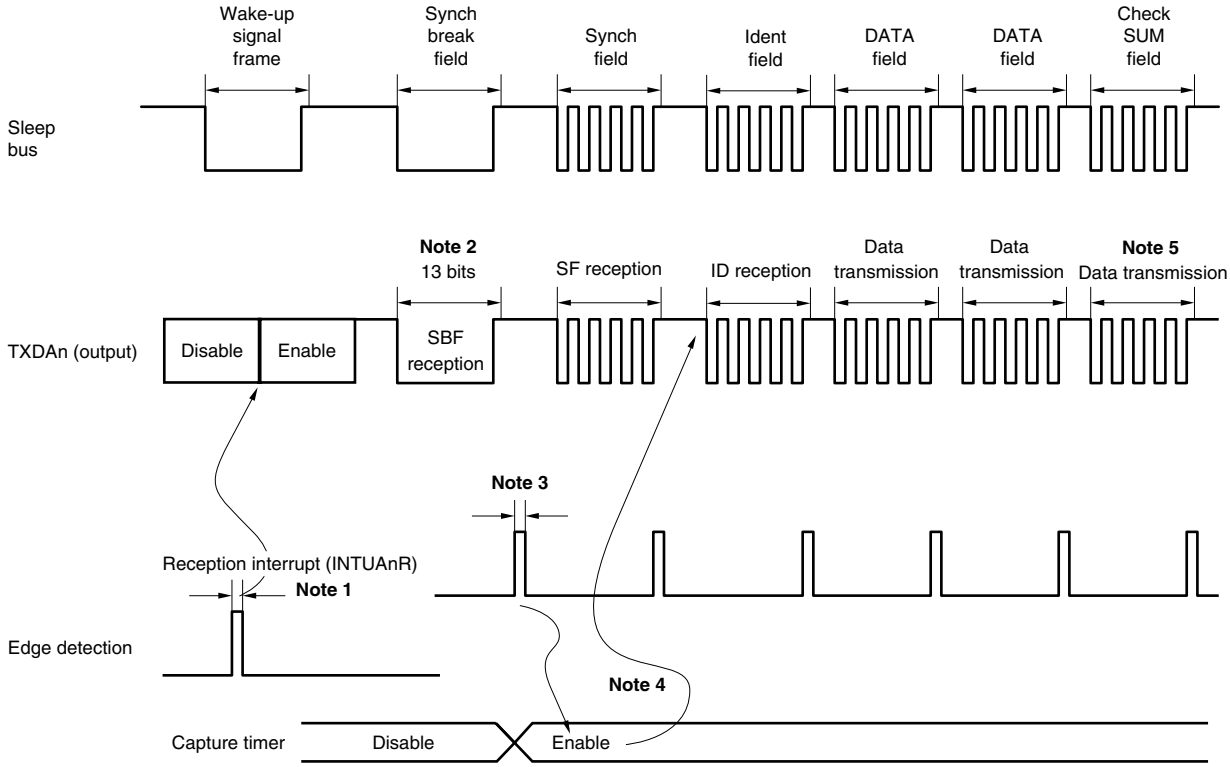
Notes 1. The interval between each field is controlled by software.

2. SBF output is performed by hardware. The output width is the bit length set by bits UAnSBL2 to UAnSBL0 of the UAnOPT0 register. If even finer output width adjustments are required, such adjustments can be performed using bits UAnBRS7 to UAnBRS0 of the UAnCTLn register.

3. 80H transfer in the 8-bit mode is substituted for the wake-up signal frame.

4. A transmission enable interrupt request signal (INTUAnT) is output at the start of each transmission. The INTUAnT signal is also output at the start of each SBF transmission.

Figure 15-7. LIN Reception Manipulation Outline



Notes 1. The wakeup signal is sent by the pin edge detector, UARTA is enabled, and the SBF reception mode is set.

2. The receive operation is performed until detection of the stop bit. Upon detection of SBF reception of 11 or more bits, normal SBF reception end is judged, and an interrupt signal is output. Upon detection of SBF reception of less than 11 bits, an SBF reception error is judged, no interrupt signal is output, and the mode returns to the SBF reception mode.

3. If SBF reception ends normally, an interrupt signal is output. The timer is enabled by an SBF reception complete interrupt. Moreover, error detection for the UAnOVE, UanPE, and UAnFE bits of the UAnSTR register is suppressed and UART communication error detection processing and UARTAn receive shift register and data transfer of the UAnRX register are not performed. The UARTAn receive shift register holds the initial value, FFH.

4. The RXDAn pin is connected to TI (capture input) of the timer, the transfer rate is calculated, and the baud rate error is calculated. The value of the UAnCTL2 register obtained by compensating the baud rate error after dropping UARTA enable is set again, causing the status to become the reception status.

5. Check-sum field distinctions are made by software. The UARTA is initialized following CSF reception, and the processing for setting the SBF reception mode again is performed by software.

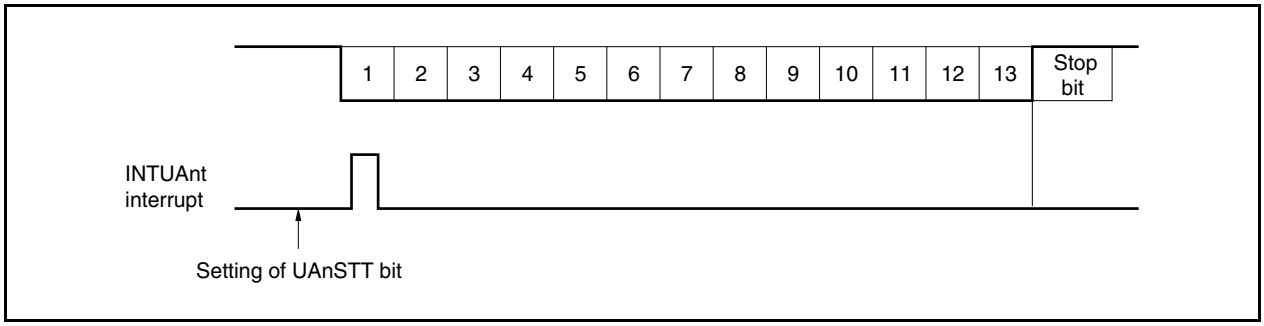
15.6.3 SBF transmission

When the UAnPWR bit = the UAnTXE bit of the UAnCTL0 register = 1, the transmission enabled status is entered, and SBF transmission is started by setting (to 1) the SBF transmission trigger (UAnSTT bit of UAnOPT0 register).

Thereafter, a low-level width of bits 13 to 20 specified by the UAnSLS2 to UAnSLS0 bits of the UAnOPT0 register is output. A transmission enable interrupt request signal (INTUAnT) is generated upon SBF transmission start. Following the end of SBF transmission, the UAnSTT bit is automatically cleared. Thereafter, the UART transmission mode is restored.

Transmission is suspended until the data to be transmitted next is written to the UAnTX register, or until the SBF transmission trigger (UAnSTT bit) is set.

Figure 15-8. SBF Transmission



15.6.4 SBF reception

The reception enabled status is achieved by setting the UAnPWR bit of the UAnCTL0 register to 1 and then setting the UAnRX bit of the UAnCTL0 register to 1.

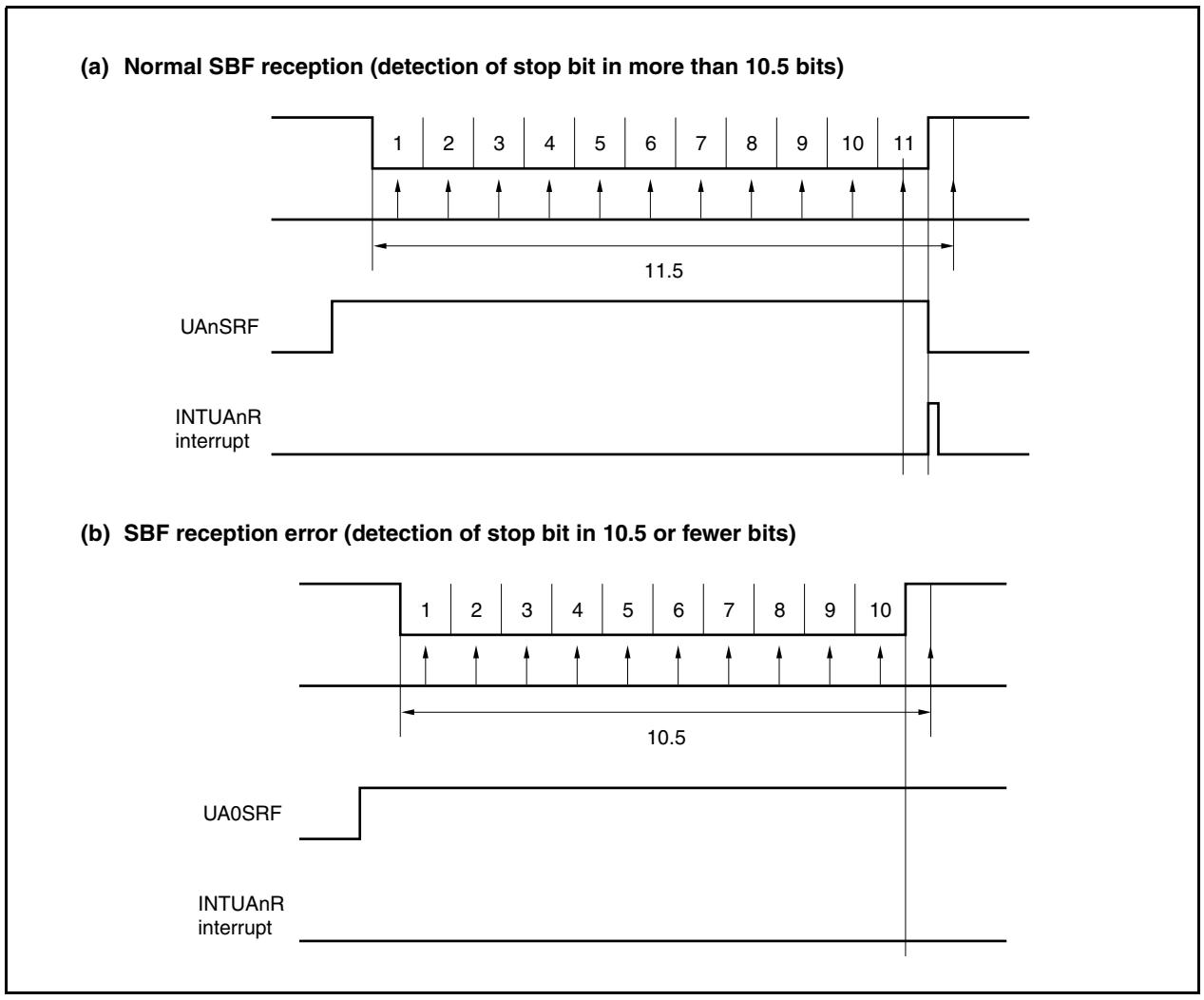
The SBF reception wait status is set by setting the SBF reception trigger (UAnSTR bit of the UAnOPT0 register) to 1.

In the SBF reception wait status, similarly to the UART reception wait status, the RXDAn pin is monitored and start bit detection is performed.

Following detection of the start bit, reception is started and the internal counter counts up according to the set baud rate.

When a stop bit is received, if the SBF width is 11 or more bits, normal processing is judged and a reception complete interrupt request signal (INTUAnR) is output. Error detection for the UAnOVE, UAnPE, and UAnFE bits of the UAnSTR register is suppressed and UART communication error detection processing is not performed. Moreover, UARTAn reception shift register and data transfer of the UAnRX register are not performed and FFH, the initial value, is held. If the SBF width is 10 or fewer bits, reception is terminated as error processing without outputting an interrupt, and the SBF reception mode is returned to. The UAnSRF bit is not cleared at this time.

Figure 15-9. SBF Reception



15.6.5 UART transmission

A high level is output to the TXDAn pin by setting the UAnPWR bit of the UAnCTL0 register to 1.

Next, the transmission enabled status is set by setting the UAnTXE bit of the UAnCTL0 register to 1, and transmission is started by writing transmit data to the UAnTX register. The start bit, parity bit, and stop bit are automatically added.

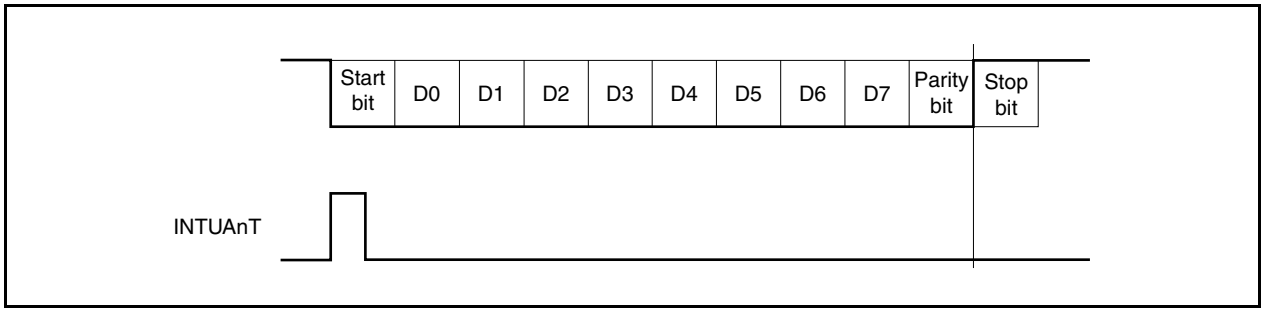
The data in the UAnTX register is transferred to the UARTAn transmit shift register upon the start of the transmit operation.

A transmission enable interrupt request signal (INTUAnT) is generated upon completion of transmission of the data of the UAnTX register to the UARTAn transmit shift register, and thereafter the contents of the UARTAn transmit shift register are output to the TXDAn pin LSB first.

Write of the next transmit data to the UAnTX register is enabled by generating the INTUAnT signal.

Continuous transmission is enabled by writing the data to be transmitted next to the UAnTX register during transfer.

Figure 15-10. UART Transmission



15.6.6 Continuous transmission procedure

UARTA can write the next transmit data to the UAnTX register when the UARTAn transmit shift register starts the shift operation. The transfer timing of the UARTAn transmit shift register can be judged from the transmission enable interrupt request signal (INTUAnT). Transmission can be performed without interruption even during interrupt processing following the transmission of 1 data frame via the INTUAnT signal, and an efficient communication rate can thus be achieved.

During continuous transmission, overrun (the completion of the next transmission before the first transmission completion processing has been executed) may occur.

An overrun can be detected by incorporating a program that can count the number of transmit data and by referencing transfer status flag (UAnTSF bit of UAnSRT register).

Caution During continuous transmission execution, perform initialization after checking that the UAnTSF bit is 0. The transmit data cannot be guaranteed when initialization is performed when the UAnTSF bit is 1.

Figure 15-11. Continuous Transmission Processing Flow

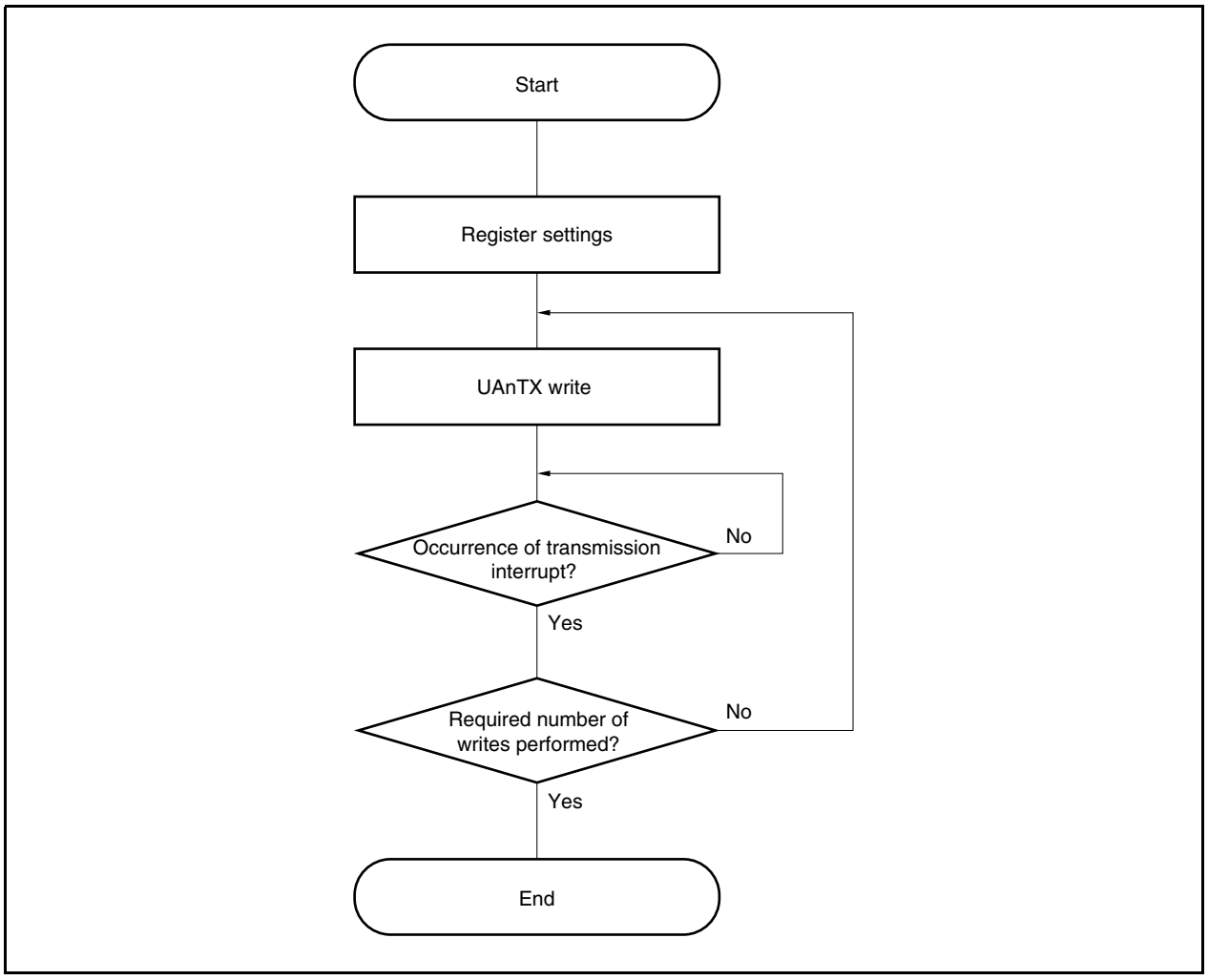
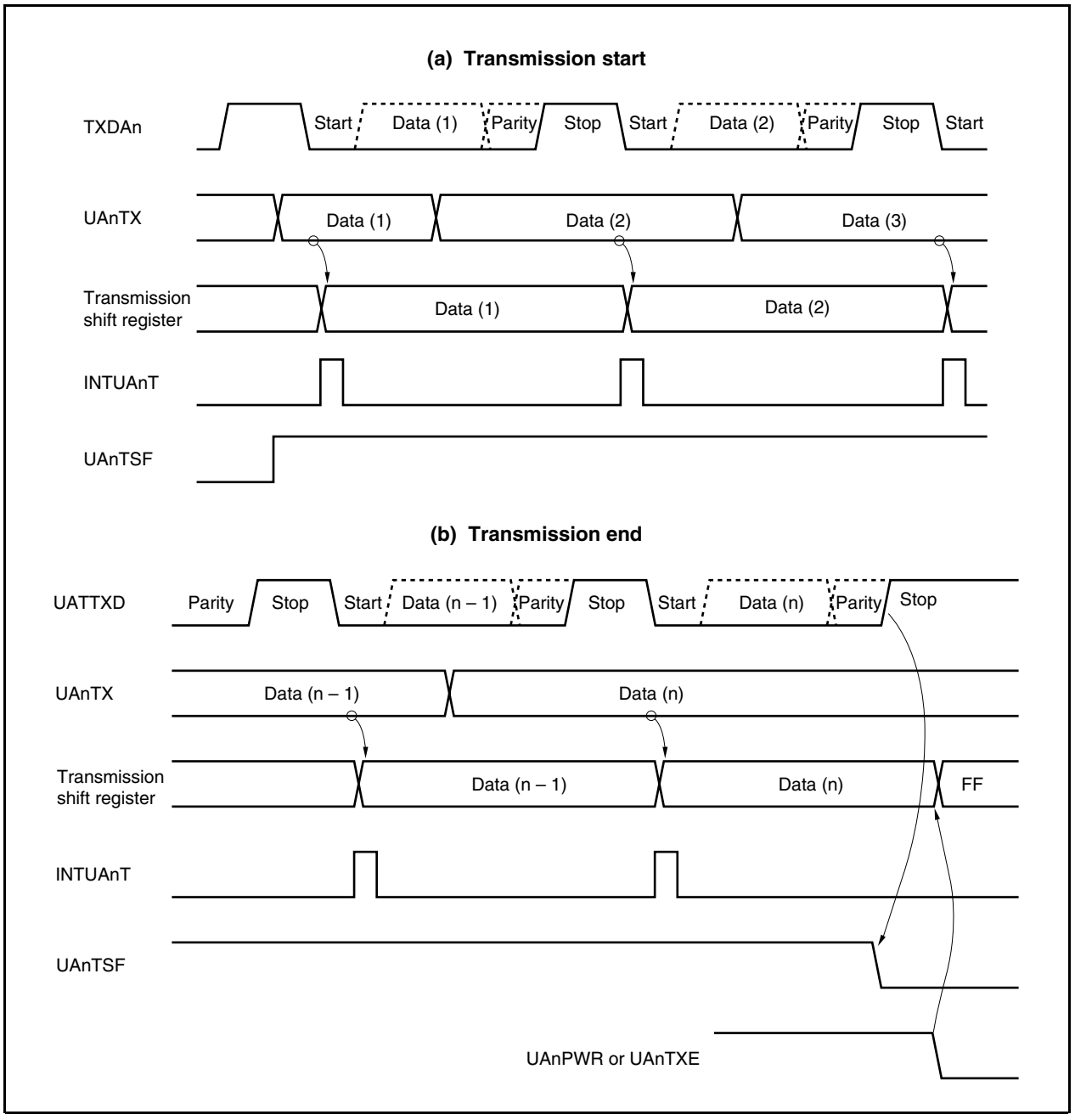


Figure 15-12. Continuous Transfer Operation Timing



15.6.7 UART reception

The reception wait status is set by setting the UAnPWR bit of the UAnCTL0 register to 1 and then setting the UAnRX bit of the UAnCTL0 register to 1. In the reception wait status, the RXDAn pin is monitored and start bit detection is performed.

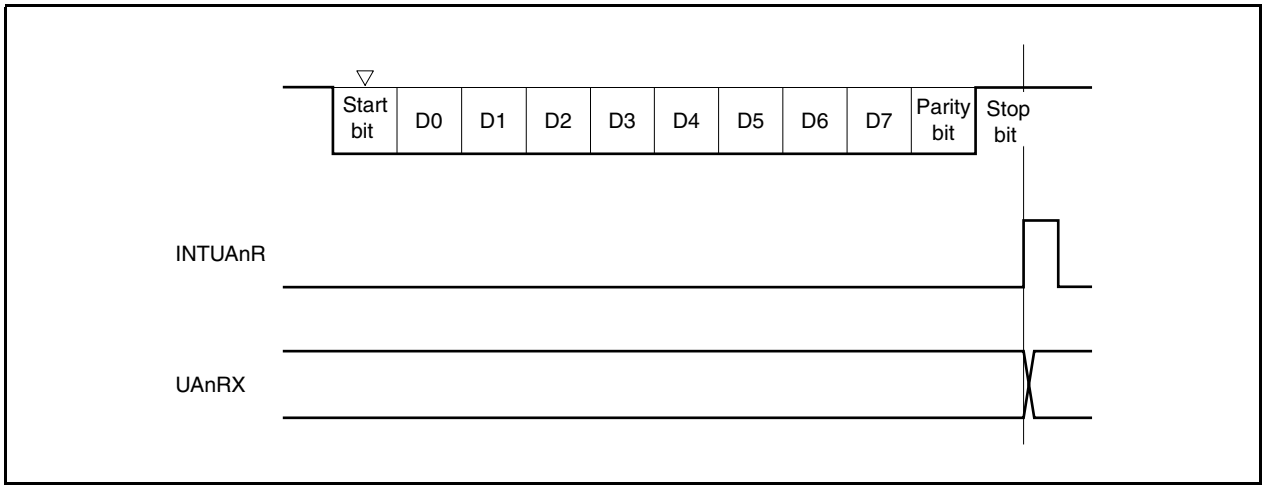
Start bit detection is performed using a two-step detection routine.

First, an 8-bit counter starts upon detection of the falling edge of the RXDAn pin. When the 8-bit counter has counted the UAnCTL2 register setting value, the level of the RXDAn pin is monitored again (corresponds to the ▽ mark in Figure 15-3). If the RXDAn pin is low level at this time too, a start bit is recognized. After a start bit has been recognized, the receive operation starts, and serial data is saved to the UARTAn receive shift register according to the set baud rate.

When the reception complete interrupt request signal (INTUAnR) is output upon reception of the stop bit, the data of the UARTAn receive shift register is written to the UAnRX register. However, if an overrun error (UAnOVE bit of UAnSTR register) occurs, the receive data at this time is not written to the UAnRX register.

Even if a parity error (UAnPE bit of UAnSTR register) or a framing error (UAnFE bit of UAnSRT register) occurs during reception, reception continues until the stop bit reception position, and INTUAnR is output following reception completion.

Figure 15-13. UART Reception



Cautions 1. Be sure to read the UAnRX register even when a reception error occurs. If the UAnRX register is not read, an overrun error occurs during reception of the next data, and reception errors continue occurring indefinitely.

2. The operation during reception is performed assuming that there is only one stop bit. A second stop bit is ignored.

15.6.8 Reception error

Errors during a receive operation are of three types: parity errors, framing errors, and overrun errors. A data reception result error flag is set to the UAnSTR register and a reception complete interrupt request signal (INTUAnR) is output.

During reception error interrupt processing, it is possible to ascertain which error occurred during reception by reading the contents of the UAnSTR register.

The reception error flag is cleared by writing 0 to it.

- Reception error causes

Error Flag	Reception Error	Cause
UAnPE	Parity error	Received parity bit does not match the setting
UAnFE	Framing error	Stop bit not detected
UAnOVE	Overrun error	Reception of next data completed before data was read from receive buffer

15.6.9 Parity types and operations

Caution When using the LIN function, fix the UAnPS1 and UAnPS0 bits of the UAnCTL0 register to 00.

The parity bit is used to detect bit errors in the communication data. Normally the same parity is used on the transmission side and the reception side.

In the case of even parity and odd parity, it is possible to detect "1" bit errors (odd count). In the case of 0 parity and no parity, errors cannot be detected.

(a) Even parity

(i) During transmission

The number of bits whose value is "1" among the transmit data, including the parity bit, is controlled so as to be an even number. The parity bit values are as follows.

- Odd number of bits whose value is "1" among transmit data: 1
- Even number of bits whose value is "1" among transmit data: 0

(ii) During reception

The number of bits whose value is "1" among the reception data, including the parity bit, is counted, and if it is an odd number, a parity error is output.

(b) Odd parity

(i) During transmission

Opposite to even parity, the number of bits whose value is "1" among the transmit data, including the parity bit, is controlled so that it is an odd number. The parity bit values are as follows.

- Odd number of bits whose value is "1" among transmit data: 0
- Even number of bits whose value is "1" among transmit data: 1

(ii) During reception

The number of bits whose value is "1" among the receive data, including the parity bit, is counted, and if it is an even number, a parity error is output.

(c) 0 parity

During transmission, the parity bit is always made 0, regardless of the transmit data.

During reception, parity bit check is not performed. Therefore, no parity error is generated, regardless of whether the parity bit is 0 or 1.

(d) No parity

No parity bit is added to the transmit data.

Reception is performed assuming that there is no parity bit. No parity error occurs since there is no parity bit.

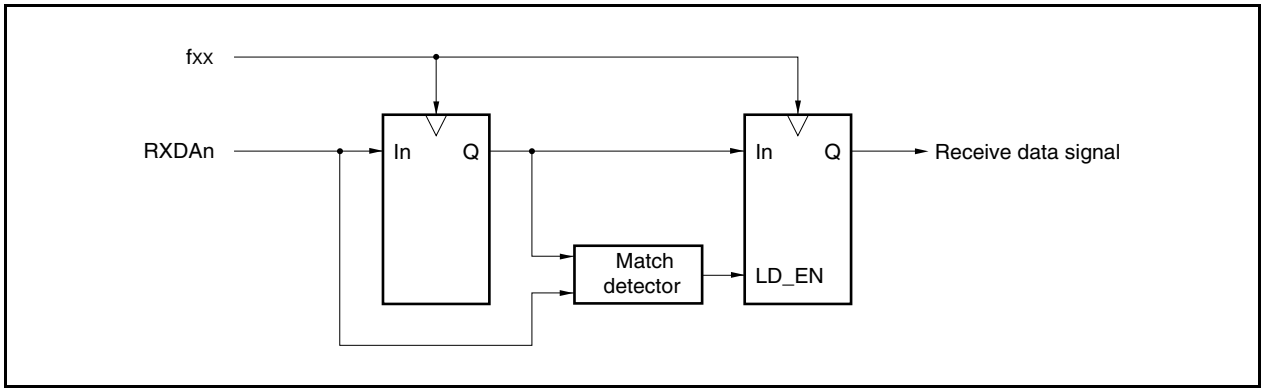
15.6.10 Receive data noise filter

This filter performs the RXDAn pin sampling using the internal system clock (fxx).

When the same sampling value is read twice, the match detector output changes and sampling as the input data is performed.

Moreover, since the circuit is as shown in Figure 15-14, the processing that goes on within the receive operation is delayed by 2 clocks in relation to the external signal status.

Figure 15-14. Noise Filter Circuit



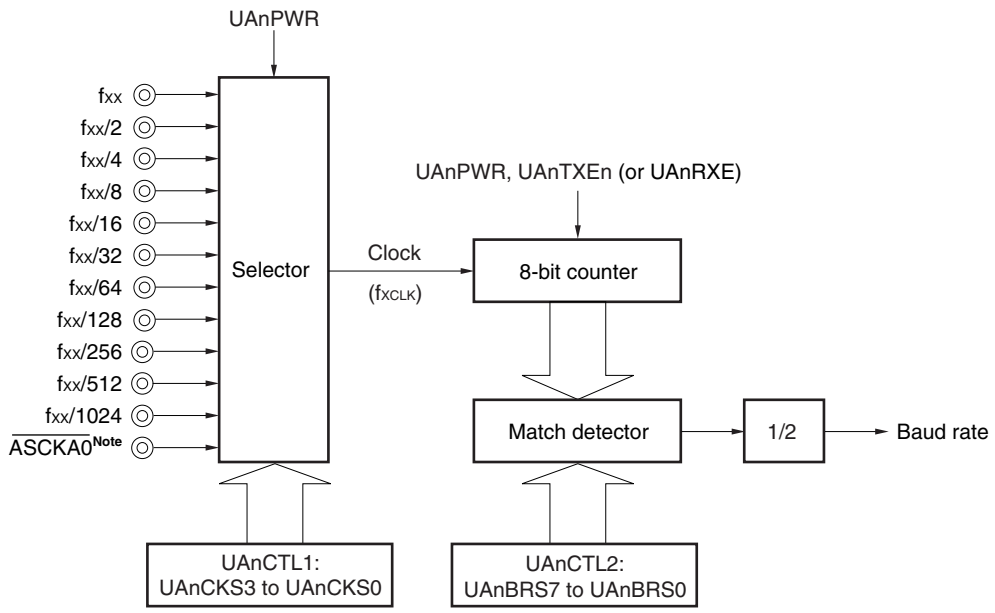
15.7 Dedicated Baud Rate Generator

The dedicated baud rate generator is configured of a source clock selector block and an 8-bit programmable counter, and generates a serial clock during transmission and reception with UARTAn. Regarding the serial clock, a dedicated baud rate generator output can be selected for each channel.

There is an 8-bit counter for transmission and another one for reception.

(1) Baud rate generator configuration

Figure 15-15. Configuration of Baud Rate Generator



Note Only UARTA0 is valid, setting UARTA1 and UARTA2 is prohibited.

Remarks 1. n = 0 to 2

2. fxx: Internal system clock

(a) Base clock (Clock)

When the UAnPWR bit of the UAnCTL0 register is 1, the clock selected by bits UAnCKS3 to UAnCKS0 of the UAnCTL1 register is supplied to the 8-bit counter. This clock is called the base clock (Clock) and its frequency is called f_{XCLK}. When the UAnPWR bit = 0, the clock is fixed to the low level.

(b) Serial clock generation

A serial clock can be generated by setting the UAnCTL1 register and the UAnCTL2 register (n = 0 to 2).

The base clock is selected by UAnCKS3 to UAnCKS0 bits of the UAnCTL1 register.

The frequency division value for the 8-bit counter can be set using bits UAnBRS7 to UAnBRS0 of the UAnCTL2 register.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is an 8-bit register that selects the UARTAn clock.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: UA0CTL1 FFFFFA01H, UA1CTL1 FFFFFA11H, UA2CTL1 FFFFFA21H

	7	6	5	4	3	2	1	0
UAnCTL1 (n = 0 to 2)	0	0	0	0	UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0

UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0	Base clock (f_{XCLK}) selection
0	0	0	0	fxx
0	0	0	1	fxx/2
0	0	1	0	fxx/4
0	0	1	1	fxx/8
0	1	0	0	fxx/16
0	1	0	1	fxx/32
0	1	1	0	fxx/64
0	1	1	1	fxx/128
1	0	0	0	fxx/256
1	0	0	1	fxx/512
1	0	1	0	fxx/1,024
1	0	1	1	External clock[Note] (ASCKA0 pin)
Other than above				Setting prohibited

Note Only UARTA0 is valid; setting UARTA1 and UARTA2 is prohibited.

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is an 8-bit register that selects the baud rate (serial transfer speed) clock of UARTAn.

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

Reset input sets this register to FFH.

After reset FFH R/W Address: UA0CTL2 FFFFFA02H, UA1CTL2 FFFFFA12H, UA2CTL2 FFFFFA22H

	7	6	5	4	3	2	1	0
UAnCTL2	UAnBRS7	UAnBRS6	UAnBRS5	UAnBRS4	UAnBRS3	UAnBRS2	UAnBRS1	UAnBRS0

(n = 0 to 2)

UAn BRS7	UAn BRS6	UAn BRS5	UAn BRS4	UAn BRS3	UAn BRS2	UAn BRS1	UAn BRS0	Default (k)	Serial clock
0	0	0	0	0	0	×	×	×	Setting prohibited
0	0	0	0	0	1	0	0	4	$f_{XCLK}/4$
0	0	0	0	0	1	0	1	5	$f_{XCLK}/5$
0	0	0	0	0	1	1	0	6	$f_{XCLK}/6$
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	0	0	252	$f_{XCLK}/252$
1	1	1	1	1	1	0	1	253	$f_{XCLK}/253$
1	1	1	1	1	1	1	0	254	$f_{XCLK}/254$
1	1	1	1	1	1	1	1	255	$f_{XCLK}/255$

Remark f_{XCLK}: Clock frequency selected by the UAnCKS3 to UAnCKS0 bits of the UAnCTL1 register

(4) Baud rate

The baud rate is obtained by the following equation.

$$\text{Baud rate} = \frac{f_{XCLK}}{2 \times k} \text{ [bps]}$$

f_{XCLK} = Frequency of base clock (Clock) selected by bits UAnCKS3 to UAnCKS0 of UAnCTL1 register
k = Value set using bits UAnBRS7 to UAnBRS0 of UAnCTL2 register (k = 4, 5, 6, ..., 255)

(5) Baud rate error

The baud rate error is obtained by the following equation.

$$\text{Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (correct baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

Cautions 1. The baud rate error during transmission must be within the error tolerance on the receiving side.

2. The baud rate error during reception must satisfy the range indicated in section (7) Allowable baud rate range during reception.

Example Base clock (Clock) frequency = 20 MHz = 20,000,000 Hz
Setting value of bits UAnBRS7 to UAnBRS0 of UAnCTL2 register = 01000001B (k = 65)
Target baud rate = 153,600

$$\text{Baud rate} = 20{,}000{,}000 / (2 \times 65) = 153{,}846 \text{ [bps]}$$

$$\text{Error} = (153{,}846/153{,}600 - 1) \times 100 = 0.160 \text{ [\%]}$$

(6) Baud rate setting example

Table 15-2. Baud Rate Generator Setting Data

Baud Rate	f_{XX} = 20 MHz			f_{XX} = 16 MHz			f_{XX} = 10 MHz		
(bps)	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)
300	09H	41H	0.16	0AH	1AH	0.16	08H	41H	0.16
600	08H	41H	0.16	0AH	0DH	0.16	07H	41H	0.16
1200	07H	41H	0.16	09H	0DH	0.16	06H	41H	0.16
2400	06H	41H	0.16	08H	0DH	0.16	05H	41H	0.16
4800	05H	41H	0.16	07H	0DH	0.16	04H	41H	0.16
9600	04H	41H	0.16	06H	0DH	0.16	03H	41H	0.16
19200	03H	41H	0.16	05H	0DH	0.16	02H	41H	0.16
31250	06H	05H	0.00	06H	04H	0.00	05H	05H	0.00
38400	02H	41H	0.16	04H	0DH	0.16	01H	41H	0.16
76800	01H	41H	0.16	03H	0DH	0.16	00H	41H	0.16
153600	00H	41H	0.16	02H	0DH	0.16	00H	21H	−1.36
312500	03H	04H	0.00	01H	0DH	−1.54	02H	04H	0.00

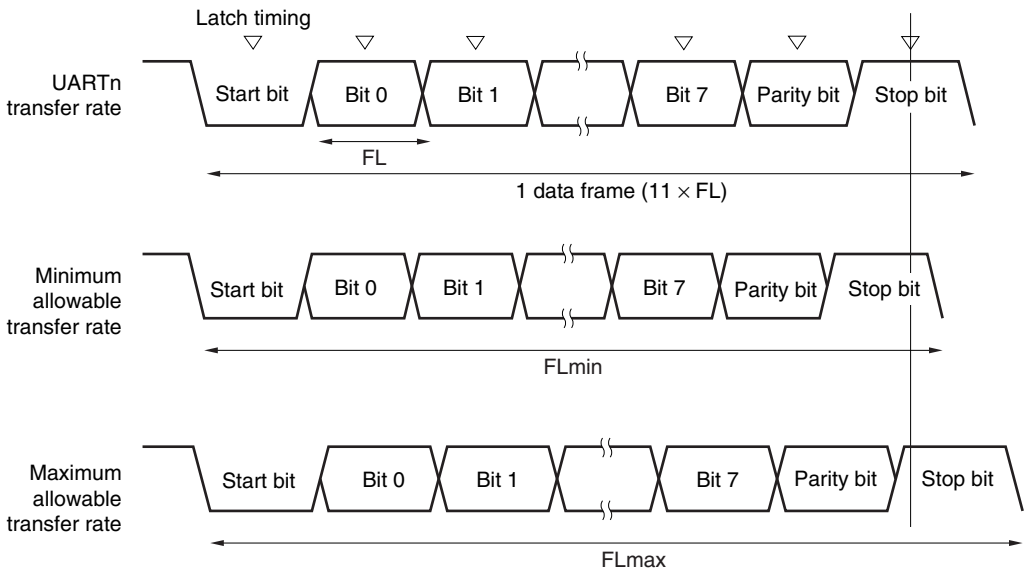
Remark f_{XX}: Internal system clock
ERR: Baud rate error (%)

(7) Allowable baud rate range during reception

The baud rate error range at the destination that is allowable during reception is shown below.

Caution The baud rate error during reception must be set within the allowable error range using the following equation.

Figure 15-16. Allowable Baud Rate Range During Reception



Remark n = 0 to 2

As shown in Figure 15-16, the receive data latch timing is determined by the counter set using the UAnCTL2 register following start bit detection. The transmit data can be normally received if up to the last data (stop bit) can be received in time for this latch timing.

When this is applied to 11-bit reception, the following results in terms of logic.

$FL = (Brate)^{-1}$

Brate: UARTAn baud rate (n = 0 to 2)
k: UAnCTL2 setting value (n = 0 to 2)
FL: 1-bit data length
Latch timing margin: 2 clocks

Minimum allowable transfer rate: $FLmin = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$

Therefore, the maximum baud rate that can be received by the destination is as follows.

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, obtaining the following maximum allowable transfer rate yields the following.

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FLmax = \frac{21k - 2}{20\ k} FL \times 11$$

Therefore, the minimum baud rate that can be received by the destination is as follows.

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

Obtaining the allowable baud rate error for UARTn and the destination from the above-described equations for obtaining the minimum and maximum baud rate values yields the following.

Table 15-3. Maximum/Minimum Allowable Baud Rate Error

Divide Ratio (k)	Maximum Allowable Baud Rate Error	Minimum Allowable Baud Rate Error
8	+3.53%	−3.61%
20	+4.26%	−4.31%
50	+4.56%	−4.58%
100	+4.66%	−4.67%
255	+4.72%	−4.73%

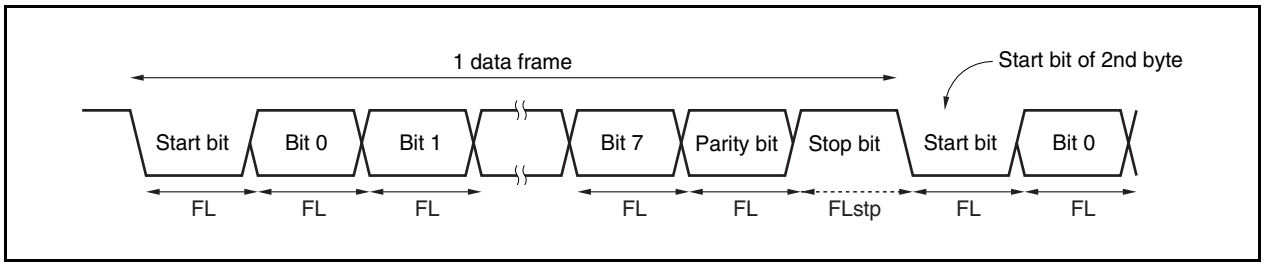
Remarks 1. The reception accuracy depends on the bit count in 1 frame, the input clock frequency, and the division ratio (k). The higher the input clock frequency and the larger the division ratio (k), the higher the accuracy is.

2. k: UAnCTL2 setting value (n = 0 to 2)

(8) Baud rate during continuous transmission

During continuous transmission, the transfer rate from the stop bit to the next start bit is usually 2 clocks longer. However, timing initialization is performed through start bit detection by the receiving side, so this has no influence on the transfer result.

Figure 15-17. Transfer Rate During Continuous Transfer



Assuming 1 bit data length: FL; stop bit length: FLstp; and base clock frequency: f_{XCLK}, we obtain the following equation.

$$FLstop = FL + 2/f_{XCLK}$$

Therefore, the transfer rate during continuous transmission is as follows.

$$\text{Transfer rate} = 11 \times FL + 2/f_{XCLK}$$

CHAPTER 16 3-WIRE VARIABLE-LENGTH SERIAL I/O (CSIB)

16.1 Mode Switching of CSIB and Other Serial Interfaces

16.1.1 CSIB4 and UARTA0 mode switching

In the V850ES/SG2, CSIB4 and UARTA0 are alternate functions of the same pin and therefore cannot be used simultaneously. CSIB4 and UARTA0 switching must be set in advance using the PMC3 and PFC3 registers.

Caution The transmit/receive operation of CSIB4 and UARTA0 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 16-1. CSIB4 and UARTA0 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF446H, FFFFF447H

	15	14	13	12	11	10	9	8
PMC3	0	0	0	0	0	0	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

After reset: 0000H R/W Address: FFFFF466H, FFFF467H

	15	14	13	12	11	10	9	8
PFC3	0	0	0	0	0	0	PFC39	PFC38

7	6	5	4	3	2	1	0
PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

After reset: 00H R/W Address: FFFFF706H

	7	6	5	4	3	2	1	0
PFCE3L	0	PFCE36	PFCE35	PFCE34	PFCE33	PFCE32	0	0

PMC32	PFCE32	PFC32	Operation mode
0	×	×	Port I/O mode
1	0	0	ASCKA0 mode
1	0	1	$\overline{\text{SCKB4}}$ mode

PMC3n	PFC3n	Operation mode
0	×	Port I/O mode
1	0	UARTA0 mode
1	1	CSIB4 mode

Remarks 1. n = 0, 1

2. × = don't care

16.1.2 CSIB0 and I²C01 mode switching

In the V850ES/SG2, CSIB0 and I²C01 are alternate functions of the same pin and therefore cannot be used simultaneously. CSIB0 and I²C01 switching must be set in advance using the PMC4 and PFC4 registers.

Caution The transmit/receive operation of CSIB0 and I²C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 16-2. CSIB0 and I²C01 Mode Switch Settings

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PMC4n	PFC4n	Operation mode
0	×	Port I/O mode
1	0	CSIB0 mode
1	1	I²C01 mode

Remarks 1. n = 0, 1

2. × = don't care

16.2 Features

- Master mode and slave mode selectable
- 8-bit to 16-bit transfer, 3-wire serial interface
- Interrupt request signals (INTCBnT, INTCBnR) × 2
- Serial clock and data phase switchable
- Transfer data length selectable in 1-bit units between 8 and 16 bits
- Transfer data MSB-first/LSB-first switchable
- 3-wire transfer SOBn: Serial data output
  SIBn: Serial data input
  $\overline{\text{SCKBn}}$: Serial clock output
- Transmission mode, reception mode, and transmission/reception mode specifiable

Remark n = 0 to 4

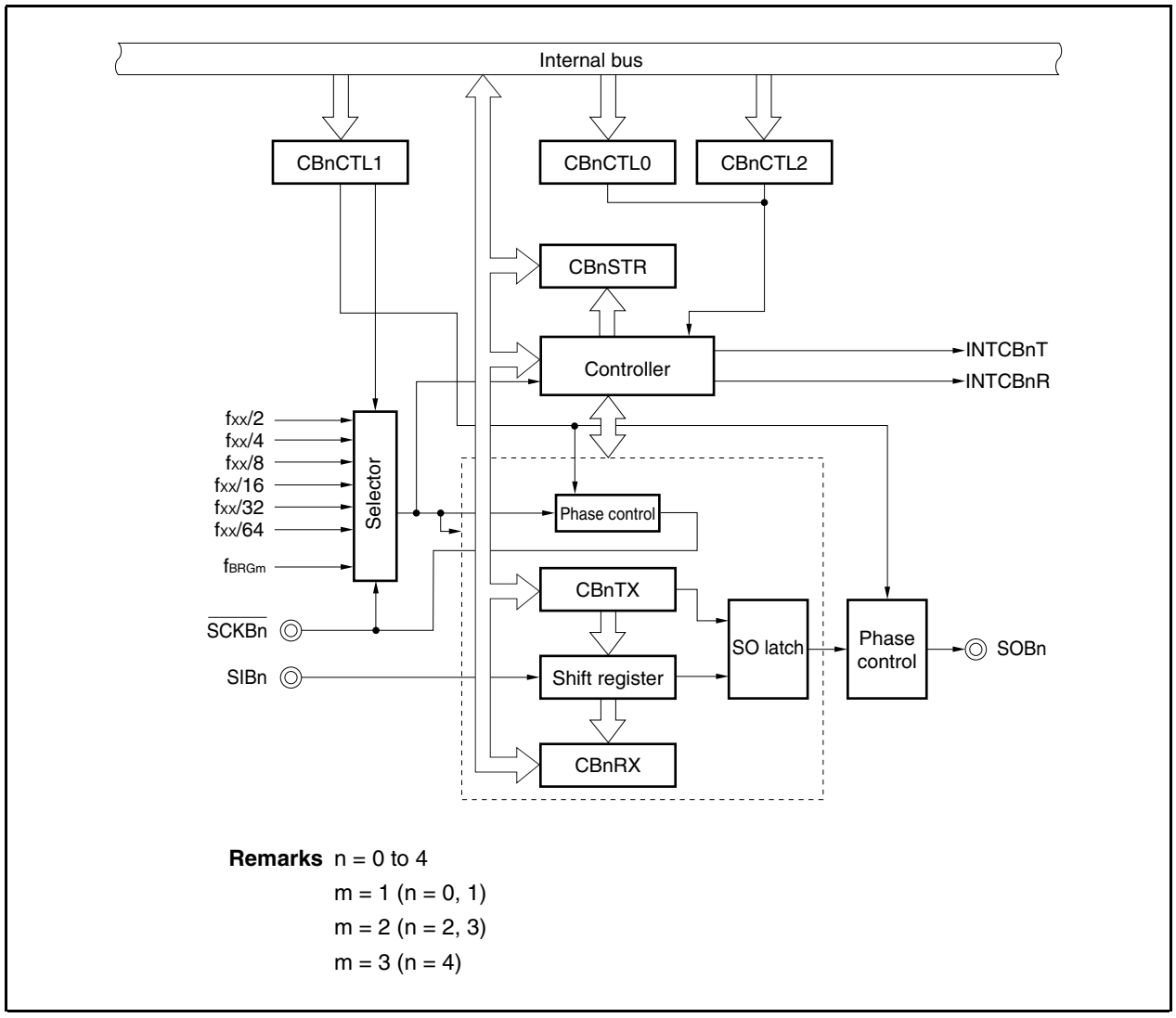
16.3 Configuration

CSIB includes the following hardware.

Table 16-1. Configuration of CSIB

Item	Configuration
Registers	CSIBn receive data register (CBnRX)
	CSIBn transmit data register (CBnTX)
Control registers	CSIBn control register 0 (CBnCTL0)
	CSIBn control register 1 (CBnCTL1)
	CSIBn control register 2 (CBnCTL2)
	CSIBn status register (CBnSTR)

Figure 16-3. Block Diagram of CSIB



Remarks n = 0 to 4
m = 1 (n = 0, 1)
m = 2 (n = 2, 3)
m = 3 (n = 4)

(1) CSIBn receive data register (CBnRX)

The CBnRX register is a 16-bit buffer register that holds receive data.

This register is read-only, in 16-bit units.

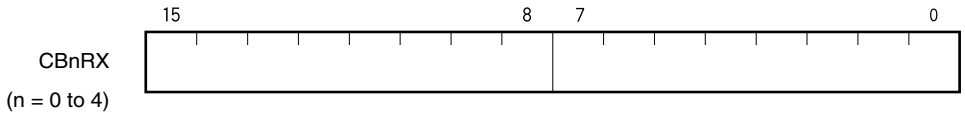
The receive operation is started by reading the CBnRX register in the reception enabled status.

If the transfer data length is 8 bits, this register is read-only as the CBnRXL register for lower 8 bits of the CBnRX register in 8-bit units.

Reset input clears this register to 0000H.

In addition to reset input, the CBnRX register can be initialized by clearing (to 0) the CBnPWR bit of the CBnCTL0 register.

After reset: 0000H R Address: CB0RX FFFFFD04H, CB1RX FFFFFD14H, CB2RX FFFFFD24H, CB3RX FFFFFD34H, CB4RX FFFFFD44H



(2) CSIB transmit data register (CBnTX)

The CBnTX register is a 16-bit buffer register used to write the CSIB transfer data.

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

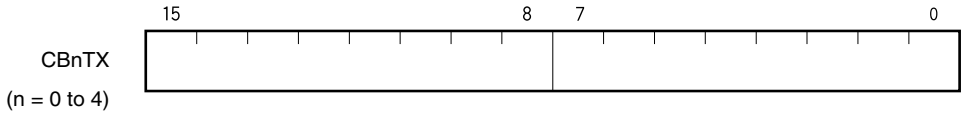
The transmit operation is started by writing data to the CBnTX register in the transmission enabled status.

If the transfer data length is 8 bits, this register is read-only as the CBnTXL register for lower 8 bits of the CBnTX register in 8-bit units.

Reset input clears this register to 0000H.

In addition to reset input, the CBnTX register can be initialized by clearing (to 0) the CBnPWR bit of the CBnCTL0 register.

After reset 0000H R/W Address: CB0TX FFFFFD06H, CB1TX FFFFFD16H, CB2TX FFFFFD26H, CB3TX FFFFFD36H, CB4TX FFFFFD46H



16.4 Control Registers

The following registers are used to control CSIB.

- CSIBn control register 0 (CBnCTL0)
- CSIBn control register 1 (CBnCTL1)
- CSIBn control register 2 (CBnCTL2)
- CSIBn status register (CBnSTR)

(1) CSIBn control register 0 (CBnCTL0)

The CBnCTL0 register is a register that controls the CSIB serial transfer operation.

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

Reset input sets this register to 01H.

(1/2)

After reset: 01H R/W Address: CB0CTL0 FFFFFD00H, CB1CTL0 FFFFFD10H, CB2CTL0 FFFFFD20H, CB3CTL0 FFFFFD30H, CB4CTL0 FFFFFD40H

CBnCTL0 (n = 0 to 4)

<7>	<6>	<5>	<4>	3	2	1	<0>
CBnPWR	CBnTXE[Note]	CBnRXE[Note]	CBnDIR[Note]	0	0	CBnTMS[Note]	CBnSCE

CBnPWR	Specification of CSIB operation stop/enable
0	Stops clock operation and reset the internal circuit
1	Enables clock operation
• The CBnPWR bit controls the CSIB operating clock and resets the internal circuit.	

CBnTXE[Note]	Specification of transmit operation stop/enable
0	Stops transmit operation
1[Note]	Enables trasmit operation
• The SOBn serial output pin is fixed to low level and communication is stopped by clearing the CBnTXE bit to 0.	

CBnRXE[Note]	Specification of receive operation stop/enable
0	Stops receive operation
1	Enables receive operation
• When the CBnRXE bit is cleared to 0, no reception complete interrupt is output even when the prescribed data is transferred in order to stop the receive operation, and the receive data register CBnRX0 is not updated.	

Note Rewrite is possible only when the CBnPWR bit = 0. However, the CBnPWR bit = 1 can also be set at the same time.

(2/2)

CBnDIR[Note]	Specification of transfer direction mode (MSB/LSB)
0	MSB first
1	LSB first

CBnTMS[Note]	Transfer mode specification
0	Single transfer mode
1	Continuous transfer mode

- When the CBnTMS bit = 0, single transfer results, so continuous transmission/ continuous reception are not supported. Even in the case of transmission only, an interrupt is output upon completion of reception transfer.

CBnSCE	Specification of start transfer disable/enable
0	Clock output stopped
1	Clock output enabled

- The CBnSCE bit controls start of the transfer operation in the master mode.
  If only reception is enabled (CBnRXE bit = 1, CBnTXE bit = 0) in the single transfer mode, the reception operation is started when the CBnRX0 register is read. To read the last receive data, clear the CBnSCE bit to 0 to disable the start of the next reception operation.
  Similarly, if only reception is enabled in the continuous transfer mode (CBnTMS bit = 1), the reception operation after reception of the last data has been completed can be disabled by clearing the CBnSCE bit one clock before completion of reception of the last data. After the last data has been read, reception is enabled again by setting the CBnSCE bit to 1 again and reading dummy data from the CBnRX0 register.
  In the slave reception mode, the CBnSCE bit is also used to enable the internal operation clock. Therefore, use the CBnSCE bit when it is set to 1.

Note Rewrite is possible only when the CBnPWR bit = 0. However, the CBnPWR bit = 1 can also be set at the same time.

(2) CSIBn control register 1 (CB0CTL1)

The CB0CTL1 is an 8-bit register that controls the CSIB serial transfer operation.

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

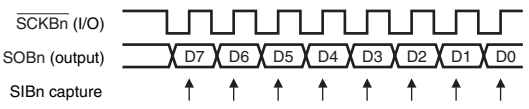
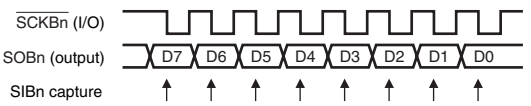
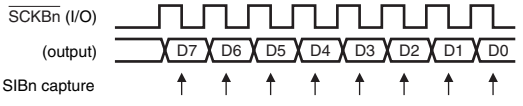
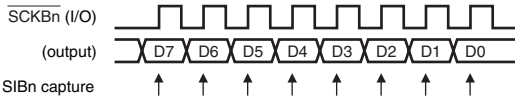
Reset input clears this register to 00H.

Caution The CBnCTL1 register can be rewritten when the CBnPWR bit of the CBnCTL0 register = 0 or when both the CBnTXE and CBnRXE bits = 0.

After reset 00H R/W Address: CB0CTL1 FFFFFD01H, CB1CTL1 FFFFFD11H, CB2CTL1 FFFFFD21H, CB3CTL1 FFFFFD31H, CB4CTL1 FFFFFD41H

CBnCTL1 (n = 0 to 4)

7	6	5	4	3	2	1	0
0	0	0	CBnCKP	CBnDAP	CBnCKS2	CBnCKS1	CBnCKS0

CBnCKP	CBnDAP	Specification of data transmission/reception timing in relation to SCKBn
0	0	
0	1	
1	0	
1	1	

CBnCKS2	CBnCKS1	CBnCKS0	Input clock	Mode
0	0	0	$f_{XX}/2$	Master mode
0	0	1	$f_{XX}/4$	Master mode
0	1	0	$f_{XX}/8$	Master mode
0	1	1	$f_{XX}/16$	Master mode
1	0	0	$f_{XX}/32$	Master mode
1	0	1	$f_{XX}/64$	Master mode
1	1	0	$f_{BRG}m$	Master mode
1	1	1	External clock ($\overline{SCKBn}$)	Slave mode

Note When n = 0, 1, m = 1
When n = 2, 3, m = 2
When n = 4, m = 3
For details on the baud rate generator, see **16.8 Baud Rate Generator**.

(3) CSIBn control register 2 (CBnCTL2)

The CBnCTL2 register is an 8-bit register that controls the number of CSIB serial transfer bits.

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

Reset input clears this register to 00H.

Caution The CBnCTL2 register can be rewritten only when the CBnPWR bit of the CBnCTL0 register = 0 or when both the CB0TXE and CB0RXE bits = 0.

After reset: 00H R/W Address: CB0CTL2 FFFFFD02H, CB1CTL2 FFFFFD12H, CB2CTL2 FFFFFD22H, CB3CTL2 FFFFFD32H, CB4CTL2 FFFFFD42H

	7	6	5	4	3	2	1	0
CBnCTL2	0	0	0	0	CBnCL3	CBnCL2	CBnCL1	CBnCL0

CBnCL3	CBnCL2	CBnCL1	CBnCL0	Serial register bit length
0	0	0	0	8 bits
0	0	0	1	9 bits
0	0	1	0	10 bits
0	0	1	1	11 bits
0	1	0	0	12 bits
0	1	0	1	13 bits
0	1	1	0	14 bits
0	1	1	1	15 bits
1	×	×	×	16 bits

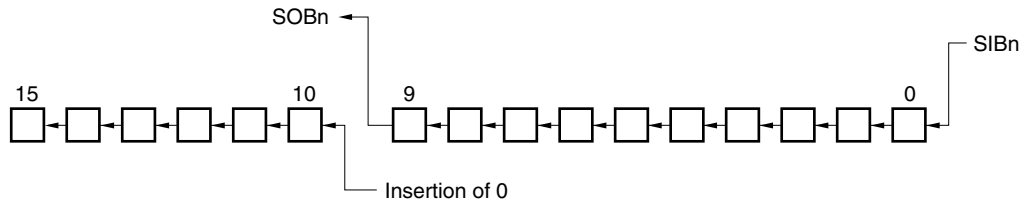
Caution If the number of transfer bits is other than 8 or 16, prepare and use data stuffed from the LSB of the CBnTX and CBnRX registers.

(a) Transfer data length change function

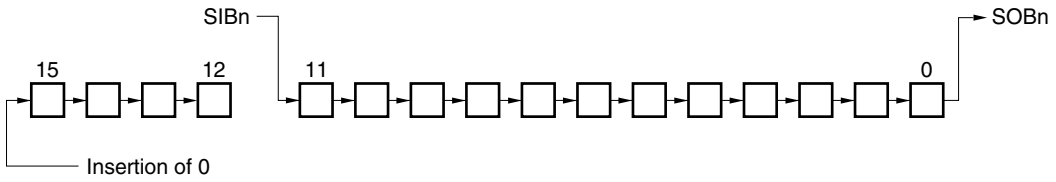
The CSIB transfer data length can be set in 1-bit units between 8 and 16 bits using bits CBnCL3 to CBnCL0 of the CBnCTL2 register.

When the transfer bit length is set to a value other than 16 bits, set the data to the CBnTX or CBnRX register starting from the LSB, regardless of whether the transfer start bit is the MSB or LSB. Any data can be set for the higher bits that are not used, but the receive data becomes 0 following serial transfer.

(i) Transfer bit length = 10 bits, MSB first



(ii) Transfer bit length = 12 bits, LSB first



(4) CSIBn status register (CBnSTR)

The CBnSTR register is an 8-bit register that displays the CSIB status.

This register can be read or written in 8-bit or 1-bit units, but the CBnSTF flag is a read-only.

Reset input clears this register to 00H.

In addition to reset input, the CBnSTR register can be initialized by clearing (0) the CBnPWR bit of the CBnCTL0 register.

After reset 00H R/W Address: CB0STR FFFFFD03H, CB1STR FFFFFD13H, CB2STR FFFFFD23H, CB3STR FFFFFD33H, CB4STR FFFFFD43H

CBnSTR (n = 0 to 4)

<7>	6	5	4	3	2	1	<0>
CBnTSF	0	0	0	0	0	0	CBnOVE

CBnTSF	Transfer operation status flag
0	Idle status
1	Operating status

- During transmission, this register is set when data is prepared in the CBnTX register, and during reception, it is set when a dummy read of the CBnRX register is performed.
  The clear timing is after the end of the edge of the last clock.

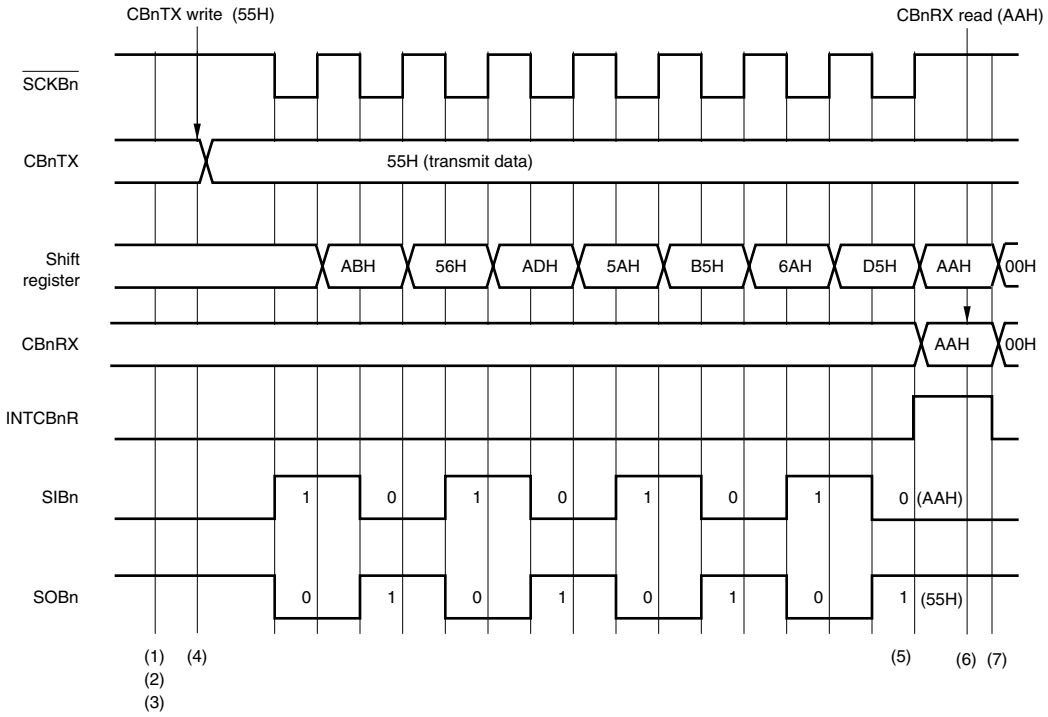
CBnOVE	Overrun error flag
0	No overrun
1	Overrun

- An overrun error occurs when the next reception starts without performing a CPU read of the value of the receive buffer, upon completion of the receive operation. The CBnOVE flag displays the overrun error occurrence status in this case.
- The CBnOVE flag is cleared by writing 0 to it. It cannot be set even by writing 1 to it.

16.5 Operation

16.5.1 Single transfer (master mode, transmission/reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnDAP bit of the CBnCTL1 register = 0, transfer data length = 8 bits (CBnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the transmission/reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Write transfer data to the CBnTX register (transmission start).

(5) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX (CBnRXL) register is possible.

(6) Read the CBnRX register before clearing the CBnPWR bit to 0.

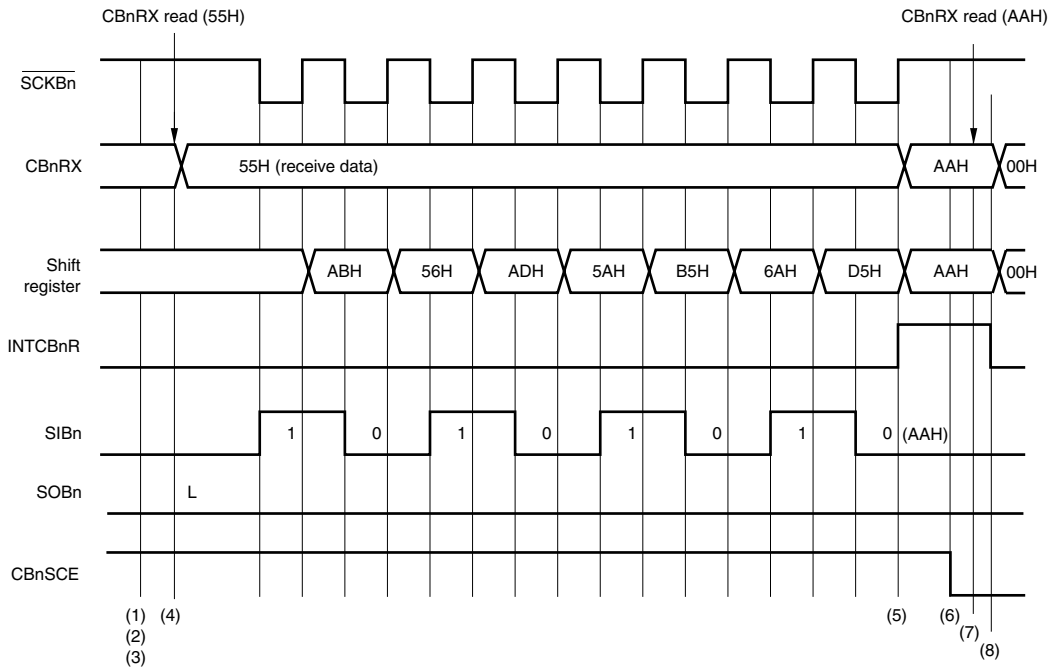
(7) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of transmission/reception).

To continue transfer, repeat steps (4) to (6) before (7).

Remark The processing of steps (2) and (3) can be set simultaneously.

16.5.2 Single transfer mode (master mode, reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnDAP bit of the CBnCTL1 register = 0, transfer data length = 8 bits (CBnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



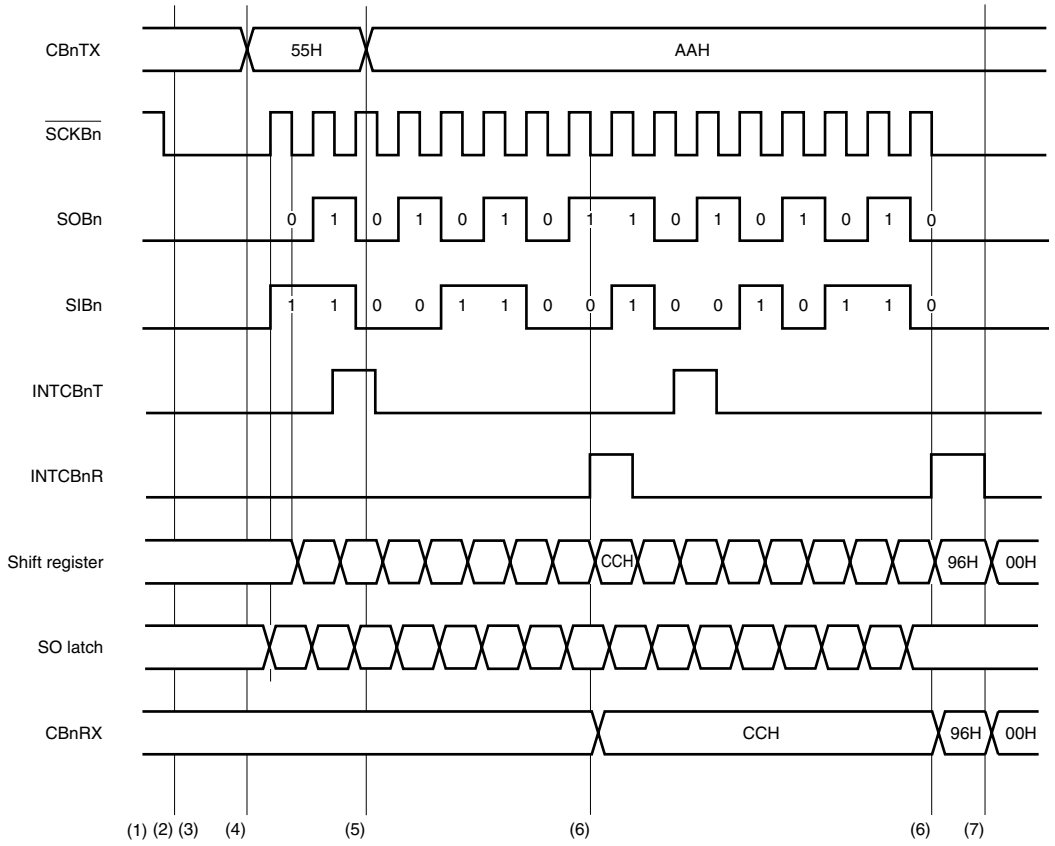
(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.
(2) Set the CBnRXE bit of the CBnCTL0 register to 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the reception enabled status.
(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.
(4) Perform a dummy read of the CBnRX register (reception start trigger).
(5) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX (CBnRXL) register is possible.
(6) Set the CBnSCE bit of the CBnCTL0 register to 0 to set the reception end data status.
(7) Read the CBnRX register before clearing the CBnPWR bit to 0.
(8) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of reception).

To continue transfer, repeat steps (4) and (5) before (6). (At this time, (4) is not a dummy read, but a receive data read combined with the reception trigger.)

Remark The processing of steps (2) and (3) can be set simultaneously.

16.5.3 Continuous mode (master mode, transmission/reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 1, the CBnDAP bit of the CBnCTL1 register = 0, transfer data length = 8 bits (CBnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the transmission/reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Write transfer data to the CBnTX register (transmission start).

(5) The transmission complete interrupt request signal (INTCBnR) is received and transfer data is written to the CBnTX register.

(6) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX (CBnRXL) register is possible.

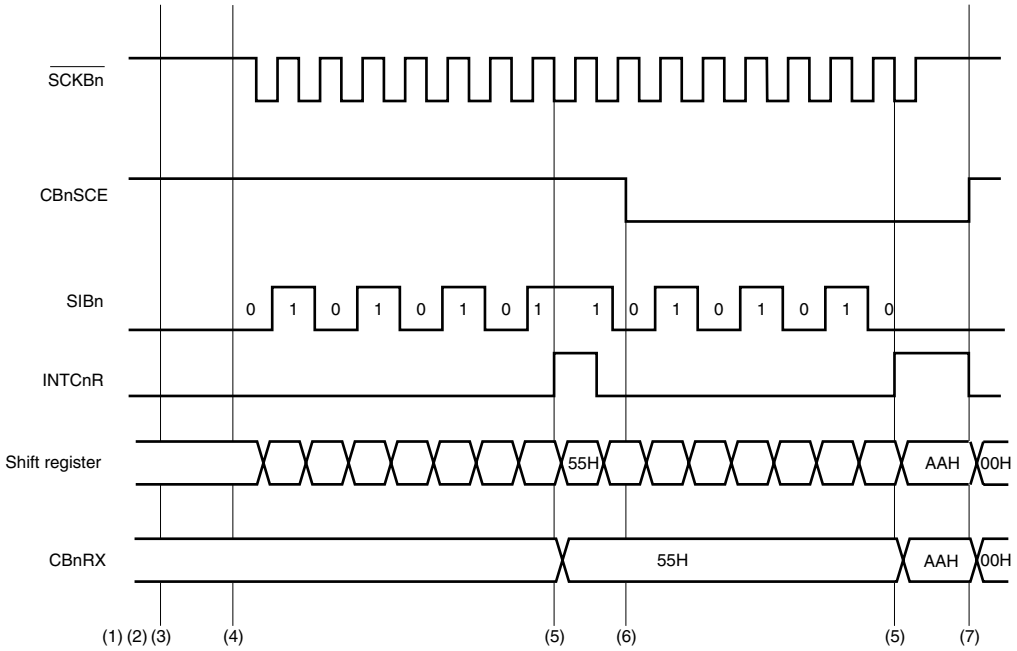
Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.

(7) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of transmission/reception).

To continue transfer, repeat steps (4) to (6) before (7).

16.5.4 Continuous mode (master mode, reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, the CBnDAP bit of the CBnCTL1 register = 1, transfer data length = 8 bits (CBnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnRXE bit of the CBnCTL0 register to 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Perform a dummy read of the CBnRX register (reception start trigger).

(5) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX (CBnRXL) register is possible.

Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.

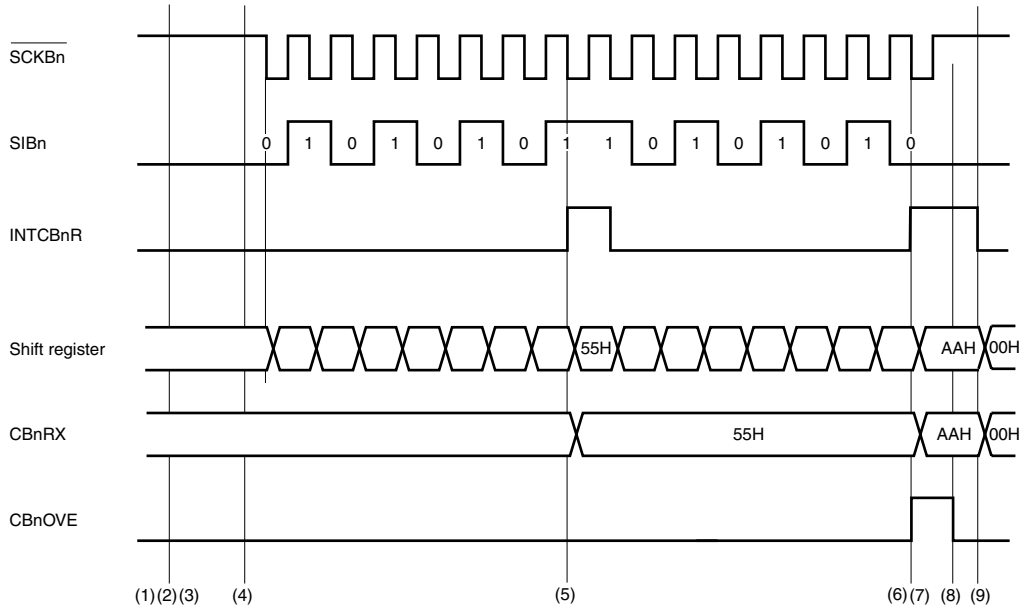
(6) Set the CBnSCE bit of the CBnCTL0 register = 0 to set the reception end data status.

(7) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of reception).

To continue transfer, repeat steps (4) and (5) before (6).

16.5.5 Continuous reception mode (error)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, the CBnDAP bit of the CBnCTL1 register = 1, transfer data length = 8 bits (CBnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnRXE bit of the CBnCTL0 register = 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Perform a dummy read of the CBnRX register (reception start trigger).

(5) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX (CBnRXL) register is possible.

(6) If the data could not be read before the end of the next transfer, the CBnOVE flag of the CBnSTR register is set upon the end of reception and a reception complete interrupt request signal (INTCBnR) is output.

Read the CBnRX register before the next receive data arrives.

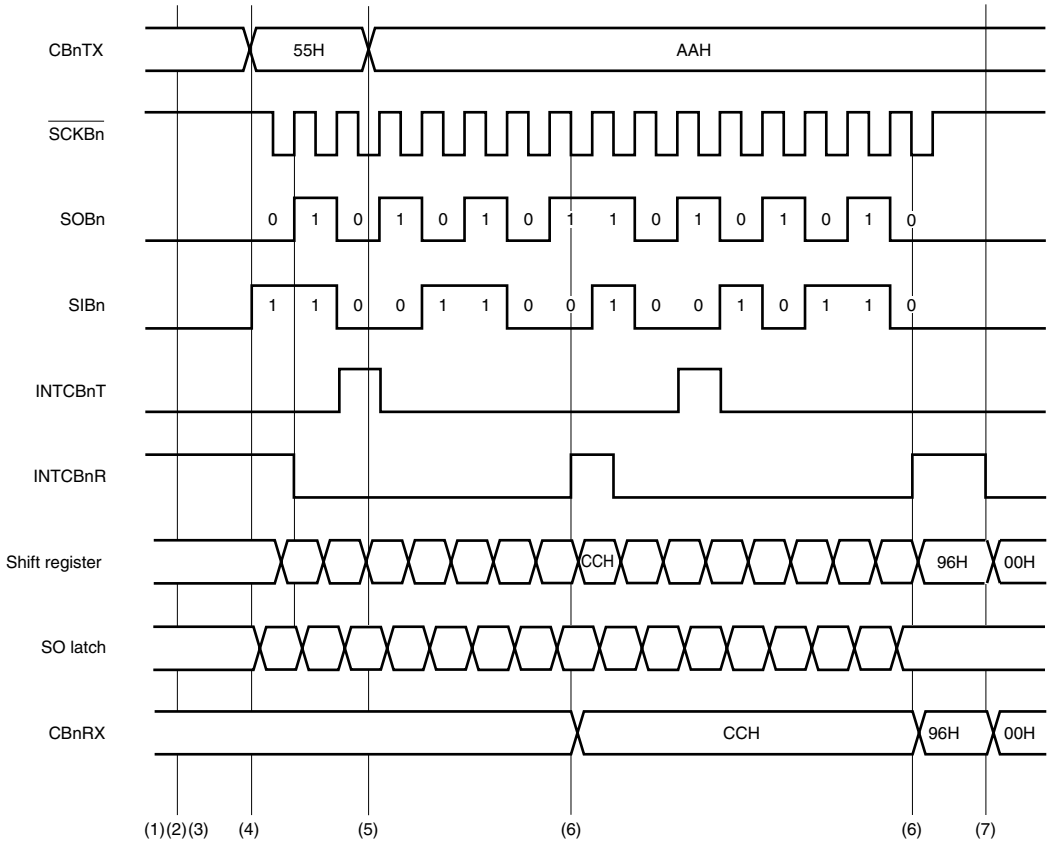
(7) Overrun error processing is performed after checking that the CBnOVE bit = 1 in the INTCBnR interrupt servicing.

(8) Clear (0) CBnOVE.

(9) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of reception).

16.5.6 Continuous mode (slave mode, transmission/reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, the CBnDAP bit of the CBnCTL1 register = 1, transfer data length = 8 bits (CSnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the transmission/reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Write the transfer data to the CBnTX register.

(5) The transmission enable interrupt request signal (INTCBnT) is received and the transfer data is written to the CBnTX register.

(6) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX register is possible.

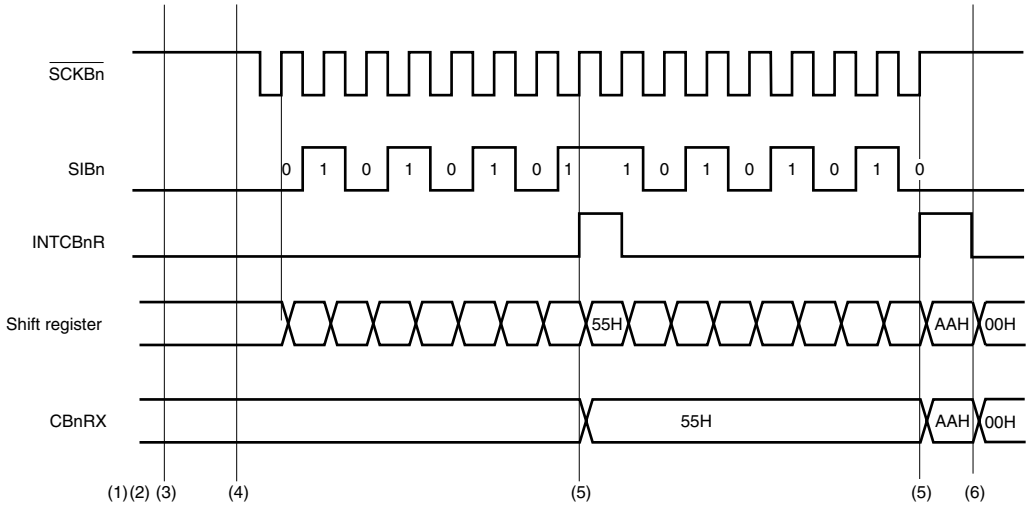
Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.

(7) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of transmission/reception).

To continue transfer, repeat steps (4) to (6) before (7).

16.5.7 Continuous mode (slave mode, reception mode)

MSB first (CBnDIR bit of CBnCTL0 register = 0), the CBnCKP bit of the CBnCTL1 register = 0, the CBnDAP bit of the CBnCTL1 register = 0, transfer data length = 8 bits (CSnCL3 to CBnCL0 bits of CBnCTL2 register = 0, 0, 0, 0)



(1) Set the CBnCTL1 and CBnCTL2 registers to specify the transfer mode.

(2) Set the CBnDIR bit of the CBnCTL0 register = 1 at the same time as specifying the transfer mode using the CBnDIR bit of the CBnCTL0 register, to set the reception enabled status.

(3) Set the CBnPWR bit of the CBnCTL0 register = 1 to enable CSIB operating clock supply.

(4) Perform a dummy read of the CBnRX register (reception start trigger).

(5) The reception complete interrupt request signal (INTCBnR) is output, notifying the CPU that reading the CBnRX register is possible.
Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.

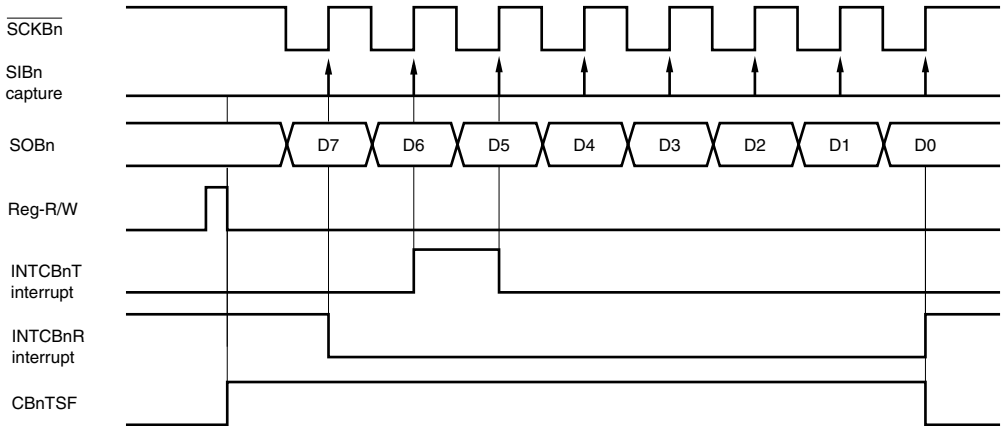
(6) Check that the CBnTSF bit of the CBnSTR register = 0 and set the CBnPWR bit to 0 to stop clock supply to CSIB (end of reception).

To continue transfer, repeat steps (4) and (5) before (6).

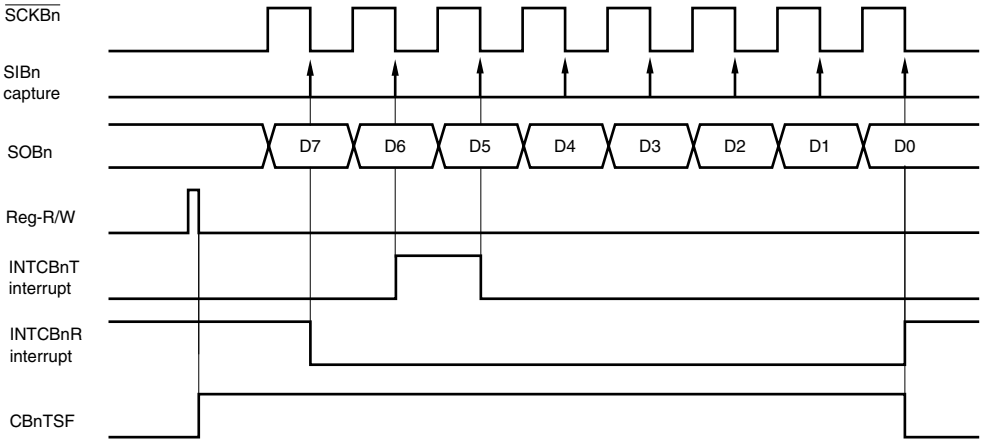
16.5.8 Clock timing

(1/2)

(i) CBnCKP = 0, CBnDAP = 0

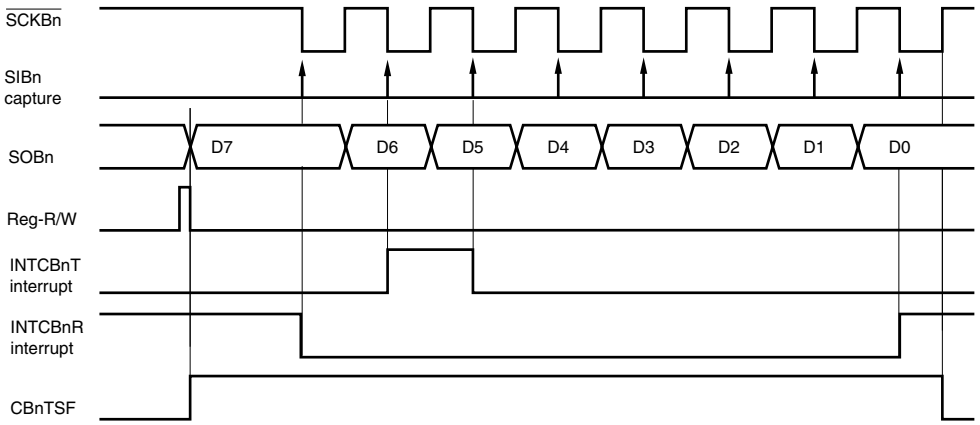


(ii) CBnCKP = 1, CBnDAP = 0

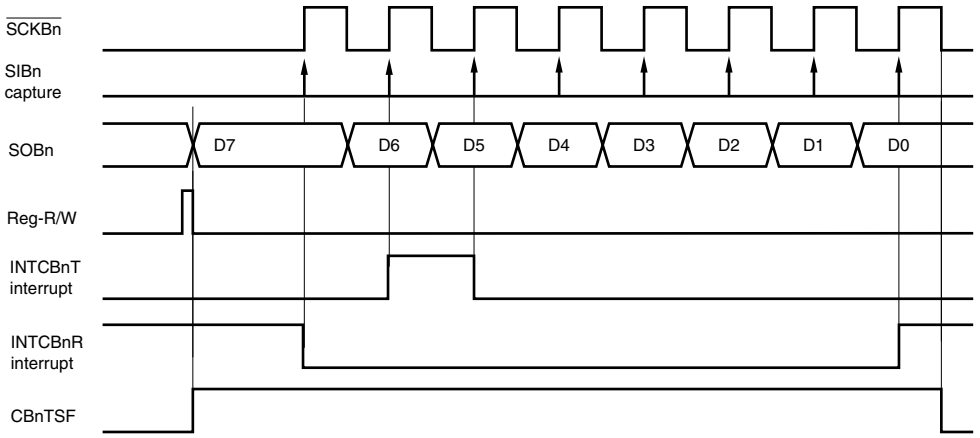


(2/2)

(iii) CBnCKP = 0, CBnDAP = 1



(iv) CBnCKP = 1, CBnDAP = 1



16.6 Output Pins

(1) $\overline{\text{SCKBn}}$ pin

When CSIBn operation is disabled (CBnPWR bit of CBnCTL0 register = 0), the $\overline{\text{SCKBn}}$ pin output status is as follows.

CBnCKP	$\overline{\text{SCKBn}}$ Pin Output
0	Fixed to high level
1	Fixed to low level

Remarks 1. The $\overline{\text{SCKBn}}$ pin output changes when the CBnCKP bit of the CBnCTL1 register is rewritten.

2. n = 0 to 4

(2) SOBn pin

When CSIBn operation is disabled (CBnPWR bit = 0), the SOBn pin output status is as follows.

CBnTXE	CBnDAP	CBnDIR	SOBn Pin Output
0	×	×	Fixed to low level
1	0	×	SOBn latch value (low level)
	1	0	CBnTXn value (MSB)
		1	CBnTXn value (LSB)

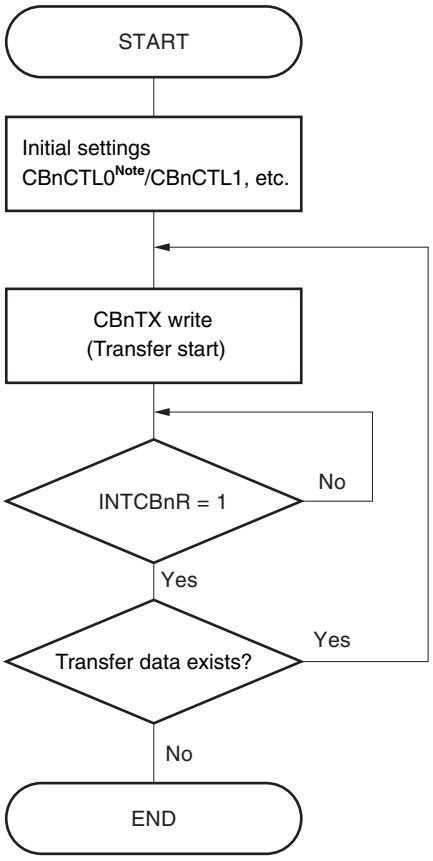
Remarks 1. The SOBn pin output changes when any one of the CBnTXE, CBnDAP, and CBnDIR bits of the CBnCTL1 register is rewritten.

2. n = 0 to 4

3. ×: don't care

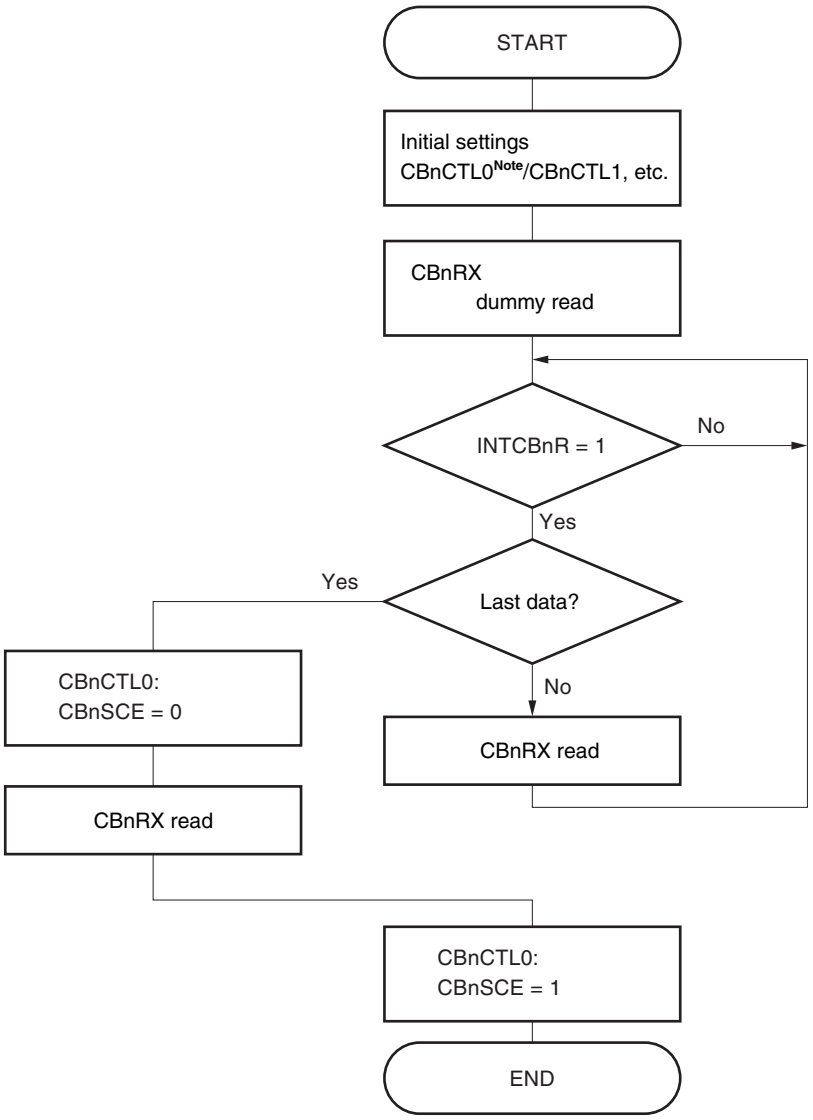
16.7 Operation Flow

(1) Single transmission



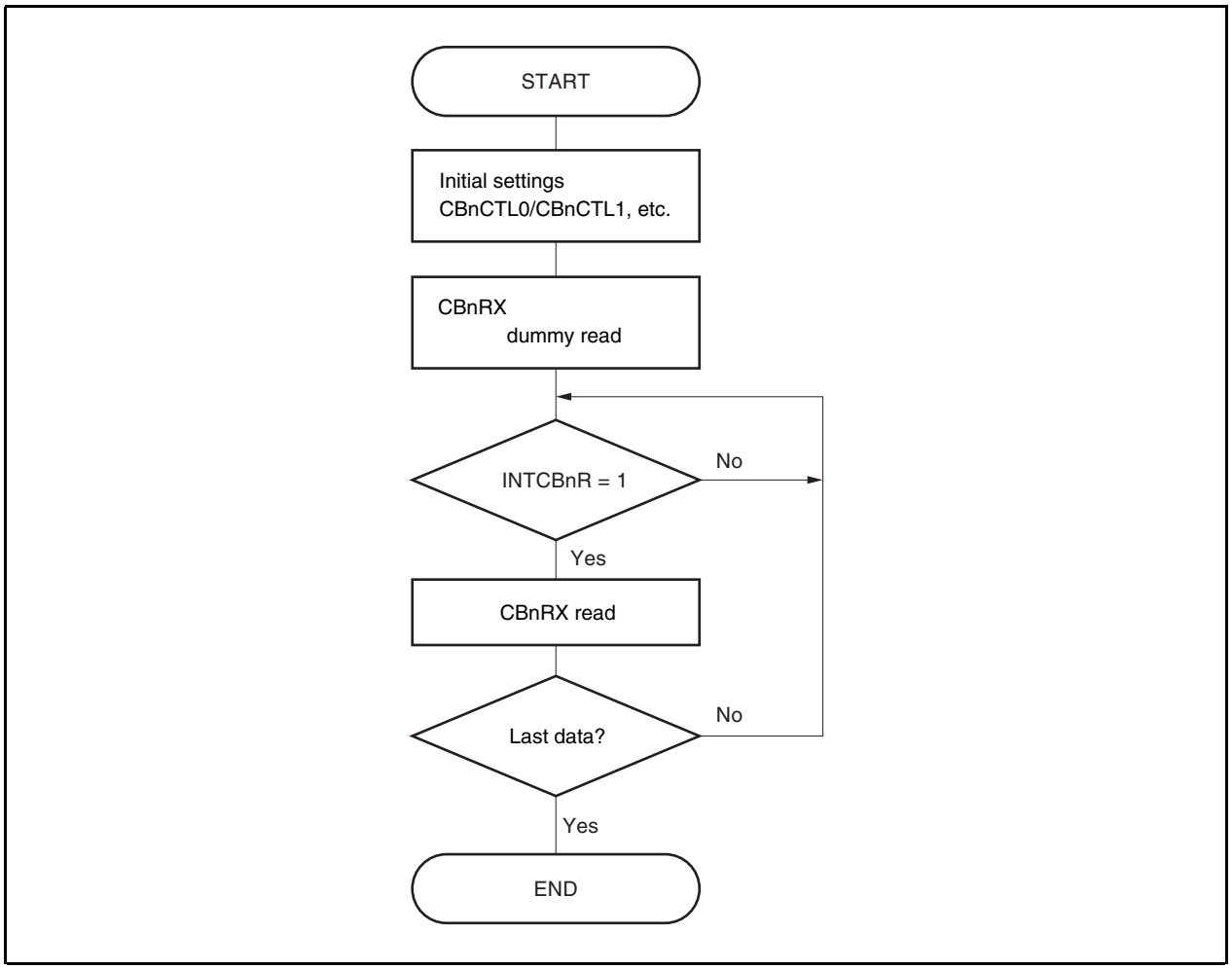
Note Set the CBnSCE bit to 1 as part of the initial settings.

(2) Single reception (master)

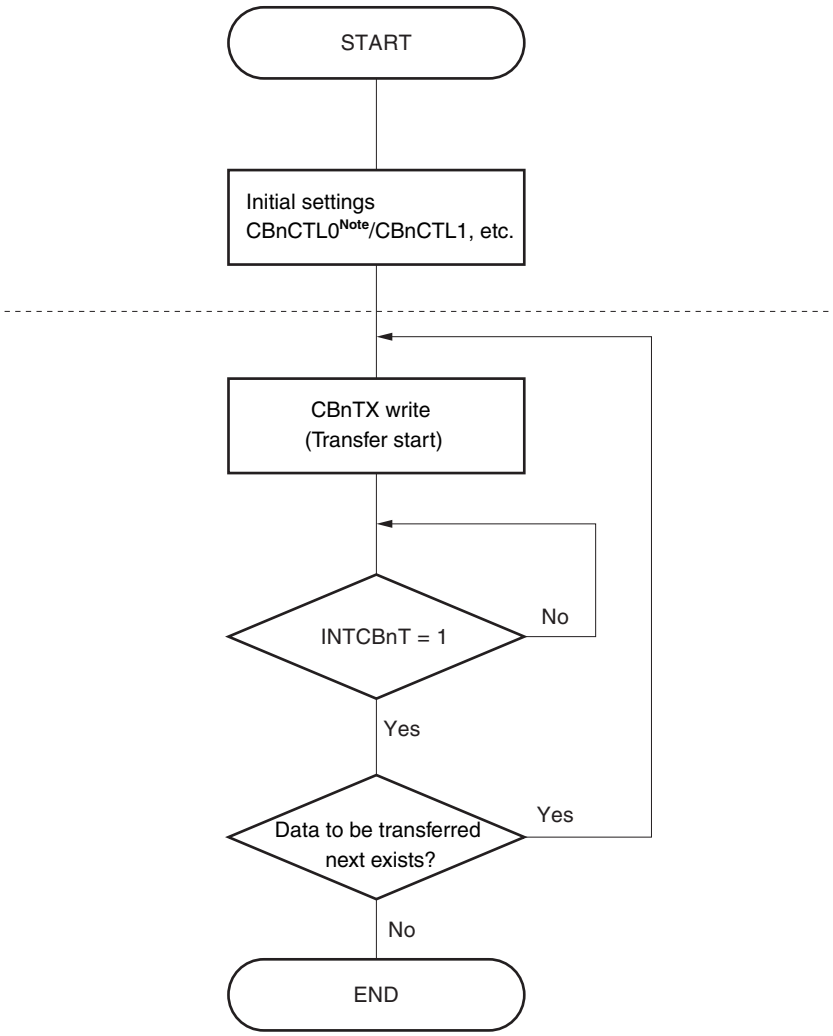


Note Set the CBnSCE bit to 1 as part of the initial settings.

(3) Single reception (slave)



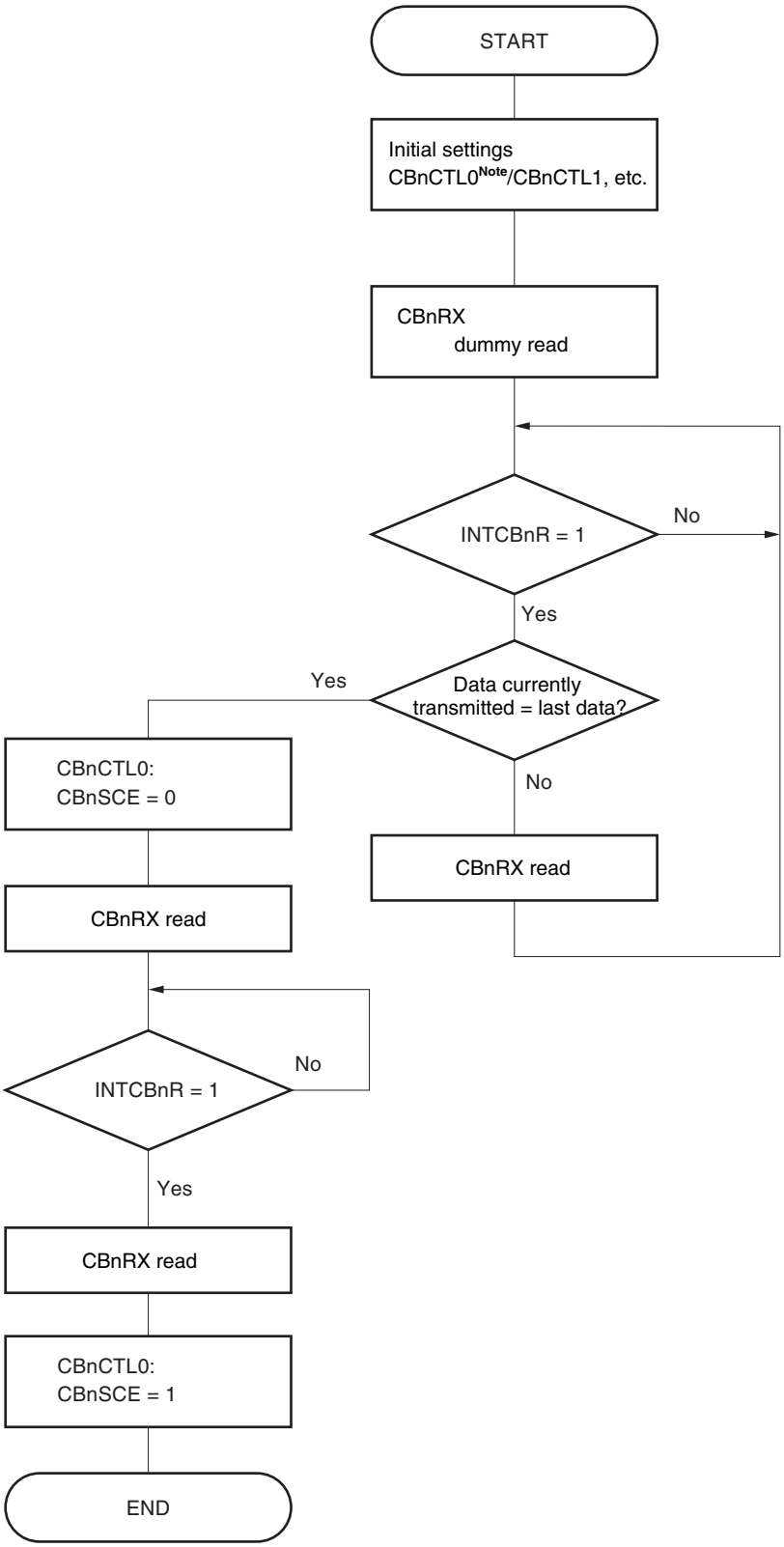
(4) Continuous transmission



Note Set the CBnSCE bit to 1 as part of the initial settings.

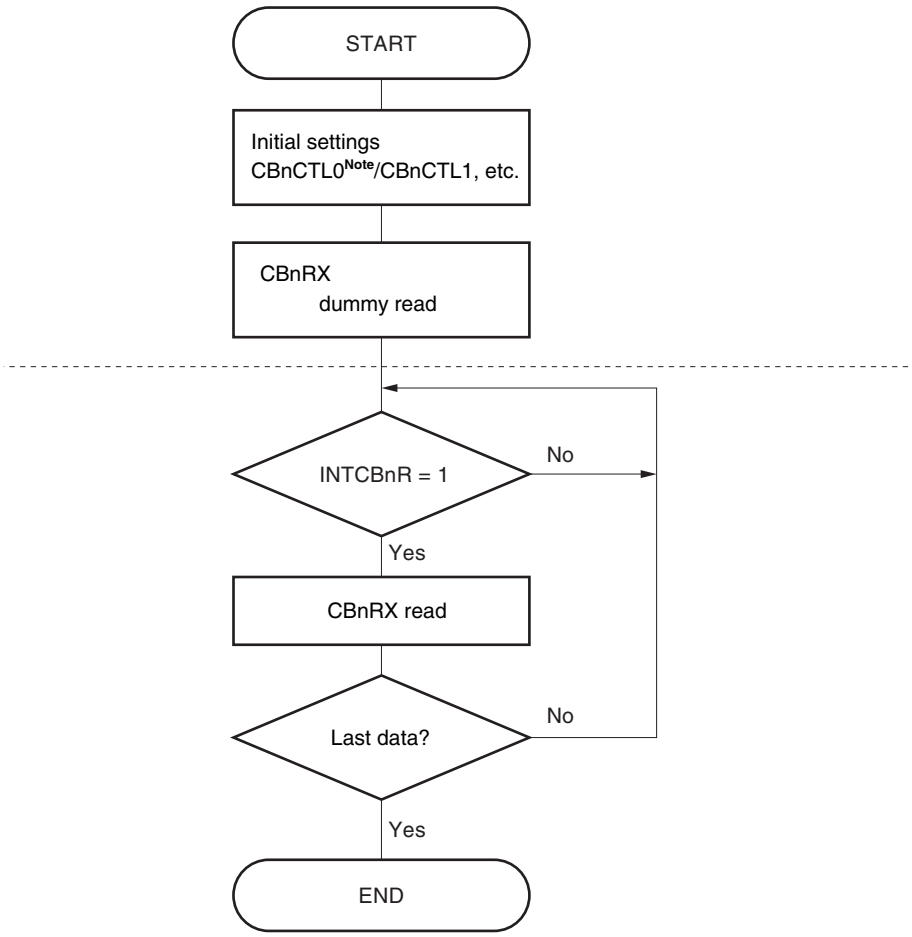
Remark The steps below the broken line constitute the transmission flow. Execute only the steps below the broken line when starting the second and subsequent transmissions.

(5) Continuous reception (master)



Note Set the CBnSCE bit to 1 as part of the initial settings.

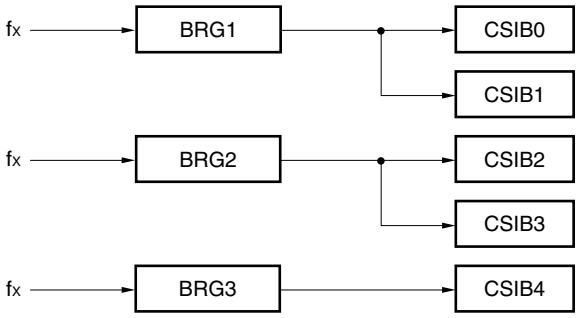
(6) Continuous reception (slave)



Note Set the CBnSCE bit to 1 as part of the initial settings.

16.8 Baud Rate Generator

The BRG1 to BRG3 and CSIB0 to CSIB4 baud rate generators are connected as shown in the following block diagram.



(1) Prescaler mode registers 1 to 3 (PRSM1 to PRSM3)

The PRSM1 to PRSM3 registers control generation of the baud rate signal for CSIB.

These registers can be read or written in 8-bit or 1-bit units.

Reset input clears these registers to 00H.

After reset: 00H R/W Address: PRSM1 FFFFF320H, PRSM2 FFFFF324H, PRSM3 FFFFF328H

PRSMm (m = 1 to 3)

7	6	5	<4>	3	2	1	0
0	0	0	BGCEm	0	0	BGCS1m	BGCS0m

BGCEm	Baud rate output
0	Disabled
1	Enabled

BGCS1m	BGCS0m	Input clock selection (f_{BGCSm})	Setting value (k)
0	0	f_{XX}	0
0	1	$f_{XX}/2$	1
1	0	$f_{XX}/4$	2
1	1	$f_{XX}/8$	3

Cautions 1. Do not rewrite the PRSMm register during operation.

2. Set the PRSMm register before setting the BGCEm bit to 1.

(2) Prescaler compare registers 1 to 3 (PRSCM1 to PRSCM3)

The PRSCM1 to PRSCM3 registers are 8-bit compare registers.

These registers can be read or written in 8-bit units.

Reset input clears these registers to 00H.

After reset: 00H R/W Address: PRSCM1 FFFFF321H, PRSCM2 FFFFF325H, PRSCM3 FFFFF329H

	7	6	5	4	3	2	1	0
PRSCMm	PRSCMm7	PRSCMm6	PRSCMm5	PRSCMm4	PRSCMm3	PRSCMm2	PRSCMm1	PRSCMm0

Cautions 1. Do not rewrite the PRSCMm register during operation.

2. Set the PRSCMm register before setting the BGCEm bit of the PRSMm register to 1.

16.8.1 Baud rate generation

The transmission/reception clock is generated by dividing the main clock. The baud rate generated from the main clock is obtained by the following equation.

$$f_{BRGm} = \frac{f_{XX}}{2^k \times N \times 2}$$

Remark f_{BRGm}: BRGm count clock

f_{XX}: Main clock oscillation frequency

k: PRSM register setting value ($0 \leq k \leq 3$)

N: PRSCM register setting value (1 to FFH)

In the case of PRSCM register setting values 01H to FFH and 00H, N = 256

m: 1 to 3

To use the I²C bus function, set the P38/SDA00, P39/SCL00, P40/SDA01, P41/SCL01, P90/SDA02, and P91/SCL02 pins to N-ch open-drain output.

17.1 Mode Switching of I²C Bus and Other Serial Interfaces

17.1.1 UARTA2 and I²C00 mode switching

In the V850ES/SG2, UARTA2 and I²C00 are alternate functions of the same pin and therefore cannot be used simultaneously. UARTA2 and I²C00 switching must be set in advance using the PMC3 and PFC3 registers.

Caution The transmit/receive operation of UARTA2 and I²C00 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 17-1. UARTA2 and I²C00 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF446H, FFFFF447H

	15	14	13	12	11	10	9	8
PMC3	0	0	0	0	0	0	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

After reset: 0000H R/W Address: FFFFF466H, FFFF467H

	15	14	13	12	11	10	9	8
PFC3	0	0	0	0	0	0	PFC39	PFC38

7	6	5	4	3	2	1	0
PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

PMC3n	PFC3n	Operation mode
0	×	Port I/O mode
1	0	UARTA2 mode
1	1	I²C00 mode

Remarks 1. n = 8, 9

2. × = don't care

17.1.2 CSIB0 and I²C01 mode switching

In the V850ES/SG2, CSIB0 and I²C01 are alternate functions of the same pin and therefore cannot be used simultaneously. CSIB0 and I²C01 switching must be set in advance using the PMC4 and PFC4 registers.

Caution The transmit/receive operation of CSIB0 and I²C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 17-2. CSIB0 and I²C01 Mode Switch Settings

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PMC4n	PFC4n	Operation mode
0	×	Port I/O mode
1	0	CSIB0 mode
1	1	I²C01 mode

Remarks 1. n = 0, 1

2. × = don't care

17.1.3 UARTA1 and I²C02 mode switching

In the V850ES/SG2, UARTA1 and I²C02 are alternate functions of the same pin and therefore cannot be used simultaneously. UARTA1 and I²C02 switching must be set in advance using the PMC9, PFC9, and PMCE9 registers.

Caution The transmit/receive operation of UARTA1 and I²C02 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 17-3. UARTA1 and I²C02 Mode Switch Settings

After reset: 0000H R/W Address: FFFFF452H, FFFFF453H

PMC9

15	14	13	12	11	10	9	8
PMC915	PMC914	PMC913	PMC912	PMC911	PMC910	PMC39	PMC38

7	6	5	4	3	2	1	0
PMC97	PMC96	PMC95	PMC94	PMC93	PMC92	PMC91	PMC90

After reset: 0000H R/W Address: FFFFF472H, FFFF473H

PFC9

15	14	13	12	11	10	9	8
PFC915	PFC914	PFC913	PFC912	PFC911	PFC910	PFC99	PFC98

7	6	5	4	3	2	1	0
PFC97	PFC96	PFC95	PFC94	PFC93	PFC92	PFC91	PFC90

After reset: 0000H R/W Address: FFFFF712H, FFFFF713

PFCE9

15	14	13	12	11	10	9	8
PFCE915	PFCE914	0	0	0	0	0	0

7	6	5	4	3	2	1	0
PFCE97	PFCE96	PFCE95	PFCE94	PFCE93	PFCE92	PFCE91	PFCE90

PMC9n	PFCE9n	PFC9n	Operation mode
1	1	0	UARTA1 mode
1	1	1	I²C02 mode

Remark n = 0, 1

17.2 Features

I²C00 to I²C02 have the following two modes.

- Operation stopped mode
- I²C (Inter IC) bus mode (multimasters supported)

(1) Operation stopped mode

In this mode, serial transfers are not performed, thus enabling a reduction in power consumption.

(2) I²C bus mode (multimaster support)

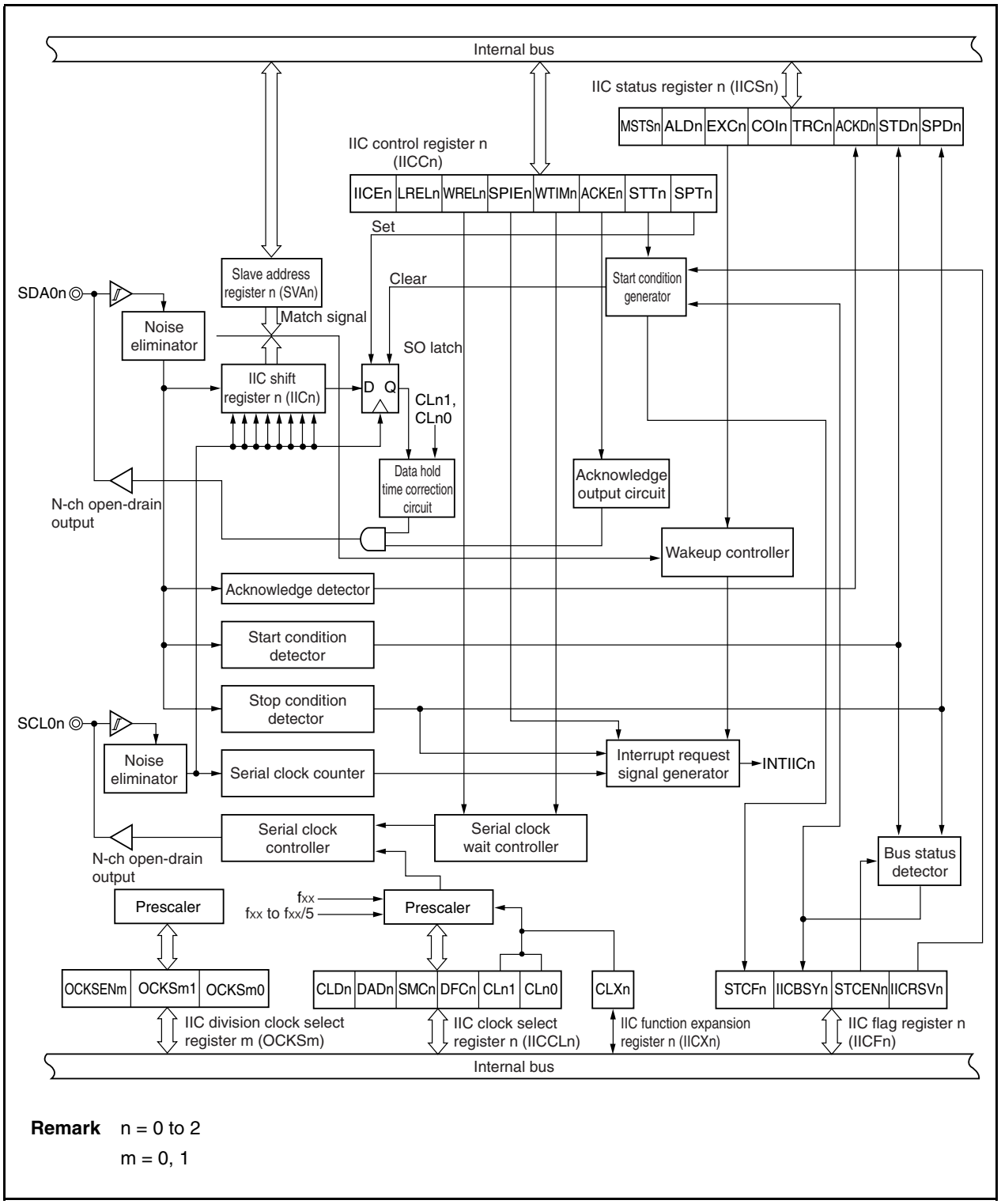
This mode is used for 8-bit data transfers with several devices via two lines: a serial clock pin (SCL0n) and a serial data bus pin (SDA0n).

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 an application program that controls the I²C bus.

Since SCL0n and SDA0n pins are N-ch open-drain outputs, I²C0n requires pull-up resistors for the serial clock line and the serial data bus line.

Remark n = 0 to 2

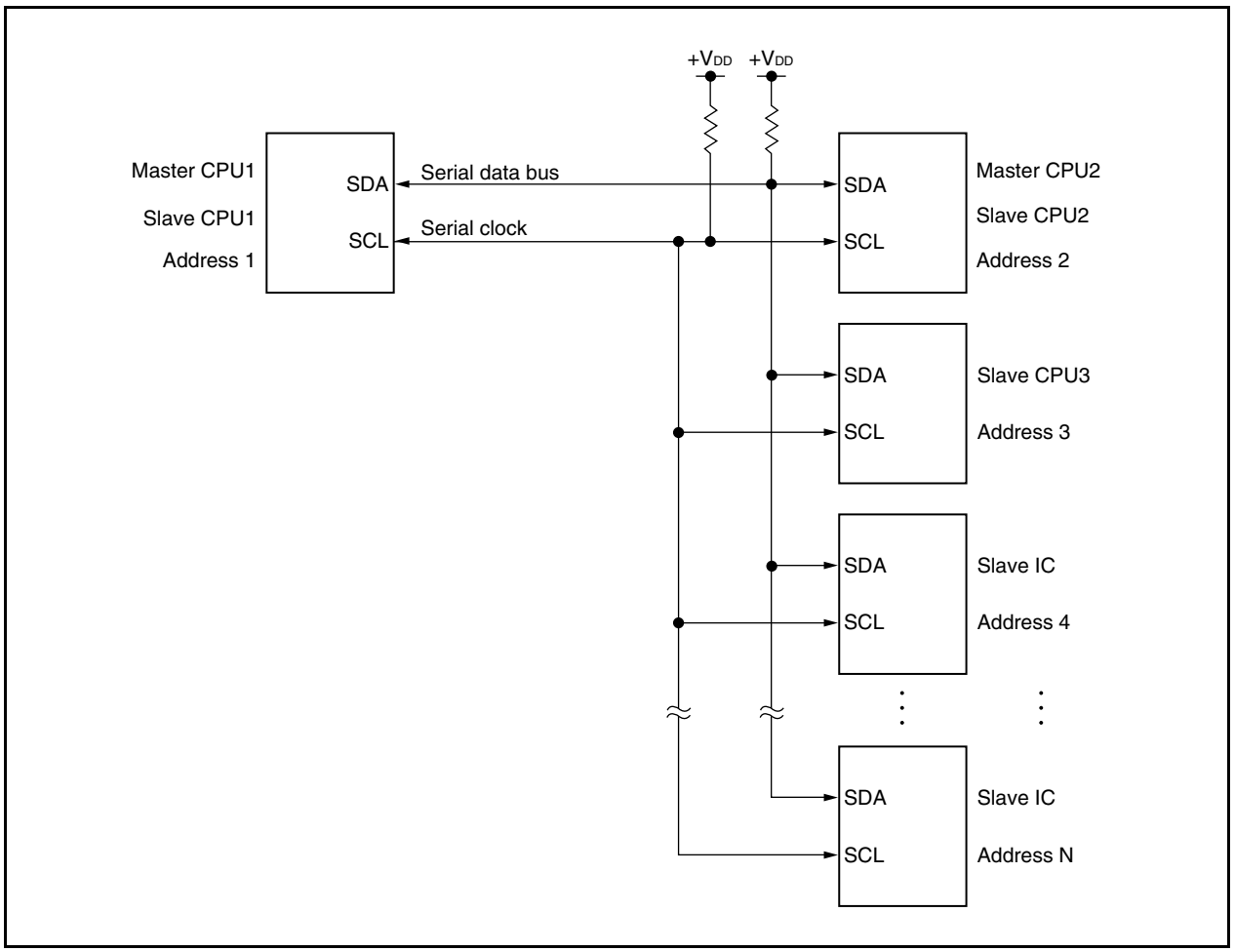
Figure 17-4. Block Diagram of I²C0n



Remark n = 0 to 2
m = 0, 1

A serial bus configuration example is shown below.

Figure 17-5. Serial Bus Configuration Example Using I²C Bus



17.3 Configuration

I²C0n includes the following hardware (n = 0 to 2).

Table 17-1. Configuration of I²C0n

Item	Configuration
Registers	IIC shift register n (IICn) Slave address register n (SVAn)
Control registers	IIC control register n (IICCn) IIC status register n (IICSn) IIC flag register n (IICF0n) IIC clock select register n (IICCLn) IIC function expansion register n (IICXn) IIC division clock select registers 0, 1 (OCKS0, OCKS1)

(1) IIC shift register n (IICn)

The IICn register converts 8-bit serial data to 8-bit parallel data and converts 8-bit parallel data to 8-bit serial data, and can be used for both transmission and reception (n = 0 to 2).

Write and read operations to the IICn register are used to control the actual transmit and receive operations.

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

Reset input clears this register to 00H.

(2) Slave address register n (SVAn)

The SVAn register sets local addresses when in slave mode (n = 0 to 2).

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

Reset input clears this register to 00H.

(3) SO latch

The SO latch is used to retain the output level of the SDA0n pin (n = 0 to 2).

(4) Wake-up controller

This circuit generates an interrupt request when the address received by this register matches the address value set to the SVAn register or when an extension code is received (n = 0 to 2).

(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 transmitted or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIICn).

An I^2C interrupt is generated following either of two triggers.

- Eighth or ninth clock of the serial clock (set by WTIMn bit of IICCn register)
- Interrupt request generated when a stop condition is detected (set by SPIEn bit of IICCn register)

Remark n = 0 to 2

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0n pin from a sampling clock (n = 0 to 2).

(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 SCL0n pin.

(12) Start condition generator

A start condition is issued when the STTn bit of the IICCn register is set.

However, in the communication reservation disabled status (IICRSVn bit of IICFn register = 1), this request is ignored and the STCFn bit of the IICFn register is set if the bus is not released (IICBSYn bit of IICFn register = 1).

(13) Bus status detector

Whether the bus is released or not is ascertained by detecting a start condition and stop condition.

However, the bus status cannot be detected immediately after operation, so set the initial status by using the STCENn bit of the IICCFn register.

17.4 Control Registers

I²C0 to I²C2 are controlled by the following registers.

- IIC control registers 0 to 2 (IICC0 to IICC2)
- IIC status registers 0 to 2 (IICS0 to IICS2)
- IIC flag registers 0 to 2 (IICF0 to IICF2)
- IIC clock select registers 0 to 2 (IICCL0 to IICCL2)
- IIC function expansion registers 0 to 2 (IICX0 to IICX2)
- IIC division clock select registers 0, 1 (OCKS0, OCKS2)

The following registers are also used.

- IIC shift registers 0 to 2 (IIC0 to IIC2)
- Slave address registers 0 to 2 (SVA0 to SVA2)

(1) IIC control registers 0 to 2 (IICC0 to IICC2)

The IICC0 to IICC2 registers enable/disable I²C0n operations, set the wait timing, and set other I²C operations (n = 0 to 2).

These registers can be read or written in 8-bit or 1-bit units.

Reset input clears these registers to 00H.

Caution In I²C00 to I²C02 bus mode, set the PMC3, PMC4, PMC9, PFC3, PFC4, PFC9, and PFCE9 registers as follows.

Pin	PMCn Register	PFCn Register	PFCEn Register
P38/TXDA2/SDA00	PMC3.PMC38 bit = 0	PFC3.PFC38 bit = 1	–
P39/RXDA2/SCL00	PMC3.PMC39 bit = 0	PFC3.PFC39 bit = 1	–
P40/SIB0/SDA01	PMC4.PMC40 bit = 0	PFC4.PFC40 bit = 1	–
P41/SOB0/SCL01	PMC4.PMC41 bit = 0	PFC4.PFC41 bit = 1	–
P90/A0/KR6/TXDA1/SDA02	PMC9.PMC90 bit = 0	PFC9.PFC90 bit = 1	PFCE9.PFCE90 bit = 1
P91/A1/KR7/RXDA1/SCL02	PMC9.PMC91 bit = 0	PFC9.PFC91 bit = 1	PFCE9.PFCE91 bit = 1

(1/4)

After reset: 00H R/W Address: IICC0 FFFFFD82H, IICC1 FFFFFD92H, IICC2 FFFFFDA2H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICCn	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

(n = 0 to 2)

IICEn	Specification of I²Cn operation enable/disable
0	Operation stopped. The IICSn register preset. Internal operation stopped.
1	Operation enabled.

Condition for clearing (IICEn bit = 0)	Condition for setting (IICEn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

LRELn	Exit from communications
0	Normal operation
1	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 SCL0n and SDA0n lines are set for high impedance. The STTn and SPTn bits and the MSTSn, EXCn, COIn, TRCn, ACKDn, and STDn bits of IICSn register are cleared.

The standby mode following exit from communications remains in effect until the following communication entry conditions are met.
- 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 (LRELn bit = 0)[Note]	Condition for setting (LRELn bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

WRELn	Wait cancellation control
0	Wait not canceled
1	Wait canceled. This setting is automatically cleared after wait is canceled.

Condition for clearing (WRELn bit = 0)[Note]	Condition for setting (WRELn bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

SPIEn	Enable/disable generation of interrupt request when stop condition is detected
0	Disabled
1	Enabled

Condition for clearing (SPIEn bit = 0)[Note]	Condition for setting (SPIEn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

Note This flag's signal is invalid when the IICEn bit = 0.

(2/4)

WTIMn	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. 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 $\overline{ACK}$ 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 (WTIMn bit = 0)[Note]	Condition for setting (WTIMn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

ACKEn	Acknowledge control
0	Acknowledgment disabled.
1	Acknowledgment enabled. During the ninth clock period, the SDA0n line is set to low level. However, the $\overline{ACK}$ is invalid during address transfers and is valid when the EXCn bit = 1.
Condition for clearing (ACKEn bit = 0)[Note]	Condition for setting (ACKEn bit = 1)[Note]
• Cleared by instruction • After reset	• Set by instruction

Note This flag's signal is invalid when the IICEn bit = 0.

Remark n = 0 to 2

(3/4)

STTn	Start condition trigger
0	Start condition is not generated.
1	When bus is released (in STOP mode): Generates a start condition (for starting as master). The SDA0n 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, the SCL0n line is changed to low level. When bus is not used: If the communication reservation function is enabled (IICRSVn bit of IICFn register = 0) • This trigger functions as a start condition reserve flag. When set, it releases the bus and then automatically generates a start condition. If the communication reservation function is disabled (IICRSVn = 1) • The STCFn bit of the IICFn register is set. This trigger does not generate a start condition. In the wait state (when master device) Generates a restart condition after releasing the wait.

Cautions concerning set timing

For master reception: Cannot be set during transfer. Can be set only when the ACKEn bit 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 ACK period. Set during the wait period.

For slave: Even when the communication reservation function is disabled (IICRSVn bit = 1), the communication reservation status is entered.

• Cannot be set at the same time as the SPTn bit

Condition for clearing (STTn bit = 0)	Condition for setting (STTn bit = 1)
• Cleared by loss in arbitration • Cleared after start condition is generated by master device • When the LRELn bit = 1 • When the IICEn bit = 0 • After reset	• Set by instruction

Remarks 1. The STTn bit is 0 if it is read immediately after data setting.

2. n = 0 to 2

(4/4)

SPTn	Stop condition trigger
0	Stop condition is not generated.
1	Stop condition is generated (termination of master device's transfer). After the SDA0n line goes to low level, either set the SCL0n line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDA0n line is changed from low level to high level and a stop condition is generated.

Cautions concerning set timing

- For master reception: Cannot be set during transfer.
  Can be set only when the ACKEn bit 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 ACK period. Set during the wait period.
- Cannot be set at the same time as the STTn bit.
- The SPTn bit can be set only when in master mode.[Note]
- When the WTIMn bit has been set to 0, if the SPTn bit 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, the WTIMn bit should be changed from 0 to 1 during the wait period following output of eight clocks, and the SPTn bit should be set during the wait period that follows output of the ninth clock.

Condition for clearing (SPTn bit = 0)	Condition for setting (SPTn bit = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When the LRELn bit = 1 • When the IICEn bit = 0 • After reset	• Set by instruction

Note Set the SPTn bit only in master mode. However, when the IICRSVn bit is 0, the SPTn bit 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 **17.5 Cautions**.

Caution When the TRCn bit = 1, the WRELn bit is set during the ninth clock and wait is canceled, after which the TRCn bit is cleared and the SDA0n line is set to high impedance.

Remarks 1. The SPTn bit is 0 if it is read immediately after data setting.

2. n = 0 to 2

(2) IIC status registers 0 to 2 (IICS0 to IICS2)

The IICS0 to IICS2 registers indicate the status of the I²C0n bus (n = 0 to 2).

These registers are read-only, 8-bit or 1-bit units.

Reset input clears these registers to 00H.

(1/3)

After reset: 00H R Address: IICCS0 FFFFFD86H, IICCS1 FFFFFD96H, IICCS2 FFFFFDA6H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

(n = 0 to 2)

MSTSn	Master device status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTSn bit = 0)	Condition for setting (MSTSn bit = 1)
• When a stop condition is detected • When the ALDn bit = 1 • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • After reset	• When a start condition is generated

ALDn	Arbitration loss detection
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”. The MSTSn bit is cleared.

Condition for clearing (ALDn bit = 0)	Condition for setting (ALDn bit = 1)
• Automatically cleared after the IICSn register is read[Note] • When the IICEn bit changes from 1 to 0 • After reset	• When the arbitration result is a “loss”.

EXCn	Detection of extension code reception
0	Extension code was not received.
1	Extension code was received.

Condition for clearing (EXCn bit = 0)	Condition for setting (EXCn bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • After reset	• When the high-order 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 the IICSn register.

(2/3)

COIn	Matching addresses detection
0	Addresses do not match.
1	Addresses match.

Condition for clearing (COIn bit = 0)	Condition for setting (COIn bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • After reset	• When the received address matches the local address (SVAn register) (set at the rising edge of the eighth clock).

TRCn	Transmit/receive status detection
0	Receive status (other than transmit status). The SDA0n line is set for high impedance.
1	Transmit status. The value in the SO latch is enabled for output to the SDA0n line (valid starting at the falling edge of the first byte's ninth clock).

Condition for clearing (TRCn bit = 0)	Condition for setting (TRCn bit = 1)
• When a stop condition is detected • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • Cleared by the WRELn bit = 1[Note] • When the ALDn bit changes from 0 to 1 • After reset Master • When "1" is output to the first byte's LSB (transfer direction specification bit) Slave • When a start condition is detected When not used for communication	Master • When a start condition is generated Slave • When "1" is input by the first byte's LSB (transfer direction specification bit)

ACKDn	ACK detection
0	ACK was not detected.
1	ACK was detected.

Condition for clearing (ACKDn bit = 0)	Condition for setting (ACKD bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • After reset	• After the SDA0n bit is set to low level at the rising edge of the SCL0n pin's ninth clock

Note The TRCn bit is cleared and SDA0n line becomes high impedance when the WRELn bit is set and the wait state is canceled at the ninth clock by the TRCn bit = 1.

Remark n = 0 to 2

(3/3)

STDn	Start condition detection
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect

Condition for clearing (STDn bit = 0)	Condition for setting (STDn bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by the LRELn bit = 1 • When the IICEn bit changes from 1 to 0 • After reset	When a start condition is detected

SPDn	Stop condition detection
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 (SPDn bit = 0)	Condition for setting (SPDn bit = 1)
• At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When the IICEn bit changes from 1 to 0 • After reset	When a stop condition is detected

Remark n = 0 to 2

(3) IIC flag registers 0 to 2 (IICF0 to IICF2)

The IICF0 to IICF2 registers set the I²C0n operation mode and indicate the I²C bus status.

These registers can be read or written in 8-bit or 1-bit units. However, the STCFn and IICBSYn bits are read-only.

IICRSVn enables/disables the communication reservation function (see **17.4 Communication Reservation**).

The initial value of the IICBSYn bit is set by using the STCENn bit (see **17.5 Cautions**).

The IICRSVn and STCENn bits can be written only when operation of I²C0n is disabled (IICEn bit of IICCn register = 0). After operation is enabled, IICFn can be read (n = 0 to 2).

Reset input clears these registers to 00H.

After reset: 00H R/W[Note] Address: IICF0 FFFFFD8AH, IICF1 FFFFFD9AH, IICF2 FFFFFDAAH

	<7>	<6>	5	4	3	2	<1>	<0>
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

(n = 0 to 2)

STCFn	STTn bit clear
0	Start condition issued
1	The STTn bit cleared

Condition for clearing (STCFn bit = 0)	Condition for setting (STCFn bit = 1)
• Cleared by the STTn bit of the IICCn register = 1 • After reset	• Clearance of the STTn bit when communication reservation disabled (IICRSVn bit = 1)

IICBSYn	I²C0n bus status
0	Bus released status
1	Bus communication status

Condition for clearing (IICBSYn bit = 0)	Condition for setting (IICBSYn bit = 1)
• When stop condition is detected • After reset	• When start condition is detected • By setting the IICEn bit of the IICCn register when the STCENn bit = 0

STCENn	Initial start enable trigger
0	Start conditions cannot be generated until a stop condition is detected following operation enable (IICEn bit = 1).
1	Start conditions can be generated even if a stop condition is not detected following operation enable (IICEn bit = 1).

Condition for clearing (STCENn bit = 0)	Condition for setting (STCENn bit = 1)
• When start condition is detected • After reset	• Setting by instruction

IICRSVn	Communication reservation function disable bit
0	Communication reservation enabled
1	Communication reservation disabled

Condition for clearing (IICRSVn bit = 0)	Condition for setting (IICRSVn bit = 1)
• Clearing by instruction • After reset	• Setting by instruction

Note Bits 6 and 7 are read-only bits.

Cautions 1. Write the STCENn bit only when operation is stopped (IICEn bit = 0).

2. When the STCENn bit = 1, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status immediately after the I²Cn bus operation is enabled. Therefore, to issue the first start condition (STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(4) IIC clock select registers 0 to 2 (IICCL0 to IICCL2)

The IICCL0 to IICCL2 registers set the transfer clock for the I²C0n bus.

These registers can be read or written in 8-bit or 1-bit units. However, the CLDn and DADn bits are read-only.

The SMCn, CLn1, and CLn0 bits are set by the combination of the CLXn bit of the IICXn register and the OCKSm1 and OCKSm0 bits of the OCKSm register (see **17.4 (6) I²C0n transfer clock setting method**) (n = 0 to 2, m = 0, 1).

Reset input clears these registers to 00H.

After reset: 00H R/W[Note] Address: IICCL0 FFFFFD84H, IICCL1 FFFFFD94H, IICCL2 FFFFFDA4H

	7	6	<5>	<4>	3	2	1	0
IICCLn	0	0	CLDn	DADn	SMCn	DFCn	CLn1	CLn0

(n = 0 to 2)

CLDn	Detection of SCL0n pin level (valid only when IICEn bit of IICCn register = 1)
0	The SCL0n pin was detected at low level.
1	The SCL0n pin was detected at high level.

Condition for clearing (CLDn bit = 0)	Condition for setting (CLDn bit = 1)
• When the SCL0n pin is at low level • When the IICEn bit = 0 • After reset	• When the SCL0n pin is at high level

DADn	Detection of SDA0n pin level (valid only when IICEn bit = 1)
0	The SDA0n pin was detected at low level.
1	The SDA0n pin was detected at high level.

Condition for clearing (DADn bit = 0)	Condition for setting (DAD0n bit = 1)
• When the SDA0n pin is at low level • When the IICEn bit = 0 • After reset	• When the SDA0n pin is at high level

SMCn	Operation mode switching
0	Operates in standard mode.
1	Operates in high-speed mode.

DFCn	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
Digital filter can be used only in high-speed mode. In high-speed mode, the transfer clock does not vary regardless of the DFCn bit setting (on/off).	

Note Bits 4 and 5 of IICCLn are read-only bits.

Caution Be sure to clear bits 7 and 6 of IICCLn to 0.

(5) IIC function expansion registers 0 to 2 (IICX0 to IICX2)

The IICX0 to IICS2 registers set I²C0n function expansion (valid only in the high-speed mode).

These registers can be read or written in 8-bit or 1-bit units. However, the CLDn and DADn bits are read-only.

Setting of the CLXn bit is performed in combination with the SMCn, CLn1, and CLn0 bits of the IICCLn register and the OCKSm1 and OCKSm0 bits of the OCKSm register (see **17.4 (6) I²C0n transfer clock setting method**) (m = 0, 1).

Reset input clears these registers to 00H.

After reset: 00H R/W Address: IICX0 FFFFFD85H, IICX1 FFFFFD95H, IICX2 FFFFFDA5H

	7	6	5	4	3	2	1	<0>
IICXn	0	0	0	0	0	0	0	CLXn

(n = 0 to 2)

(6) I²C0n transfer clock setting method

The I²C0n transfer clock frequency (f_{SCL}) is calculated using the following expression (n = 0 to 2).

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

m = 12, 18, 24, 36, 44, 48, 54, 60, 66, 72, 86, 88, 96, 132, 172, 176, 198, 220, 258, 344 (see **Table 17-2 Clock Settings**).

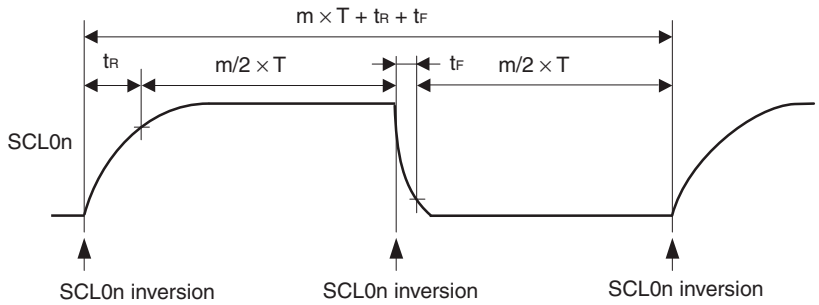
T: $1/f_{XX}$

t_R: SCL0n pin rise time

t_F: SCL0n pin fall time

For example, the I²C0n transfer clock frequency (f_{SCL}) when f_{XX} = 19.2 MHz, m = 198, t_R = 200 ns, and t_F = 50 ns is calculated using following expression.

$$f_{SCL} = 1/(198 \times 52 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 94.7 \text{ kHz}$$



The clock to be selected can be set by the combination of the SMCn, CLn1, and CLn0 bits of the IICCLn register, the CLXn bit of the IICXn register, and the OCKSm1 and OCKSm0 bits of the OCKSm register (n = 0 to 2, m = 0, 1).

Table 17-2. Clock Settings (1/2)

IICX0	IICCL0			Selection Clock	Transfer Clock	Settable Main Clock Frequency (fxx) Range	Operating Mode
Bit 0	Bit 3	Bit 1	Bit 0				
CLX0	SMC0	CL01	CL00				
0	0	0	0	fxx (when OCKS0 = 14H set)	fxx/44	2.00 MHz ≤ fxx ≤ 4.19 MHz	Standard mode (SMC0 = 0)
				fxx/2 (when OCKS0 = 10H set)	fxx/88	4.00 MHz ≤ fxx ≤ 8.38 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/132	6.00 MHz ≤ fxx ≤ 12.57 MHz	
				fxx/4 (when OCKS0 = 12H set)	fxx/176	8.00 MHz ≤ fxx ≤ 16.76 MHz	
				fxx/5 (when OCKS0 = 13H set)	fxx/220	10.00 MHz ≤ fxx ≤ 20.00 MHz	
0	0	0	1	fxx (when OCKS0 = 14H set)	fxx/86	4.19 MHz ≤ fxx ≤ 8.38 MHz	
				fxx/2 (when OCKS0 = 10H set)	fxx/172	8.38 MHz ≤ fxx ≤ 16.76 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/258	12.57 MHz ≤ fxx ≤ 20.00 MHz	
				fxx/4 (when OCKS0 = 12H set)	fxx/344	16.76 MHz ≤ fxx ≤ 20.00 MHz	
0	0	1	0	fxx[Note]	fxx/86	4.19 MHz ≤ fxx ≤ 8.38 MHz	
0	0	1	1	fxx (when OCKS0 = 14H set)	fxx/66	6.4 MHz	
				fxx/2 (when OCKS0 = 10H set)	fxx/132	12.8 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/198	19.2 MHz	
0	1	0	×	fxx (when OCKS0 = 14H set)	fxx/24	4.19 MHz ≤ fxx ≤ 8.38 MHz	High-speed mode (SMC0 = 1)
				fxx/2 (when OCKS0 = 10H set)	fxx/48	8.00 MHz ≤ fxx ≤ 16.76 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/72	12.00 MHz ≤ fxx ≤ 20.00 MHz	
				fxx/4 (when OCKS0 = 12H set)	fxx/96	16.00 MHz ≤ fxx ≤ 20.00 MHz	
0	1	1	0	fxx[Note]	fxx/24	4.00 MHz ≤ fxx ≤ 8.38 MHz	
0	1	1	1	fxx (when OCKS0 = 14H set)	fxx/18	6.4 MHz	
				fxx/2 (when OCKS0 = 10H set)	fxx/36	12.8 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/54	19.2 MHz	
1	1	0	×	fxx (when OCKS0 = 14H set)	fxx/24	4.19 MHz ≤ fxx ≤ 8.38 MHz	
				fxx/2 (when OCKS0 = 10H set)	fxx/24	8.00 MHz ≤ fxx ≤ 8.38 MHz	
				fxx/3 (when OCKS0 = 11H set)	fxx/36	12.00 MHz ≤ fxx ≤ 12.57 MHz	
				fxx/4 (when OCKS0 = 12H set)	fxx/48	16.00 MHz ≤ fxx ≤ 16.67 MHz	
				fxx/5 (when OCKS0 = 13H set)	fxx/60	20.00 MHz	
1	1	1	0	fxx[Note]	fxx/12	4.00 MHz ≤ fxx ≤ 4.19 MHz	
Other than above				Setting prohibited	–	–	–

Note Since the selection clock is fxx regardless of the value set to the OCKS0 register, set the OCKS0 register = 00H (I²C division clock stopped status).

Remark ×: don't care

Table 17-2. Clock Settings (2/2)

IICXm	IICCLm			Selection Clock	Transfer Clock	Settable Main Clock Frequency (f_{XX}) Range	Operating Mode
Bit 0	Bit 3	Bit 1	Bit 0				
CLXm	SMCm	CLm1	CLm0				
0	0	0	0	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/44$	2.00 MHz ≤ f_{XX} ≤ 4.19 MHz	Standard mode (SMCm = 0)
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/88$	4.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/132$	6.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKS1 = 12H set)	$f_{XX}/176$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/5$ (when OCKS1 = 13H set)	$f_{XX}/220$	10.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	0	0	1	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/86$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/172$	8.38 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/258$	12.57 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKS1 = 12H set)	$f_{XX}/344$	16.76 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	0	1	0	f_{XX} Note	$f_{XX}/86$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
0	0	1	1	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/66$	6.4 MHz	
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/132$	12.8 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/198$	19.2 MHz	
0	1	0	×	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/24$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	High-speed mode (SMCm = 1)
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/48$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/72$	12.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKS1 = 12H set)	$f_{XX}/96$	16.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	1	1	0	f_{XX} Note	$f_{XX}/24$	4.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
0	1	1	1	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/18$	6.4 MHz	
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/36$	12.8 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/54$	19.2 MHz	
1	1	0	×	f_{XX} (when OCKS1 = 14H set)	$f_{XX}/24$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/2$ (when OCKS1 = 10H set)	$f_{XX}/24$	8.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKS1 = 11H set)	$f_{XX}/36$	12.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKS1 = 12H set)	$f_{XX}/48$	16.00 MHz ≤ f_{XX} ≤ 16.67 MHz	
				$f_{XX}/5$ (when OCKS1 = 13H set)	$f_{XX}/60$	20.00 MHz	
1	1	1	0	f_{XX} Note	$f_{XX}/12$	4.00 MHz ≤ f_{XX} ≤ 4.19 MHz	
Other than above				Setting prohibited	–	–	–

Note Since the selection clock is f_{XX} regardless of the value set to the OCKS1 register, set the OCKS1 register = 00H (I^2C division clock stopped status).

Remarks 1. m = 1, 2

2. ×: don't care

(7) IIC division clock select registers 0, 1 (OCKS0, OCKS1)

The OCKS0 and OCKS1 registers control the I²C0n division clock.

These registers can be read or written in 8-bit or 1-bit units.

Reset input clears these registers to 00H.

After reset: 01H R/W Address: OCKS0 FFFFF340H, OCKS1 FFFFF344H

	7	6	5	4	3	2	1	0
OCKSm	0	0	0	OCKSENm	0	OCKSm2	OCKSm1	OCKSm0

(m = 0, 1)

OCKSENn	Operation settings of I²C division clock
0	Disable I²C division clock operation
1	Enable I²C division clock operation

OCKSm2	OCKSm1	OCKSm0	Selection of I²C division clock
0	0	0	$f_{XX}/2$
0	0	1	$f_{XX}/3$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/5$
1	0	0	f_{XX}

(8) IIC shift registers 0 to 2 (IIC0 to IIC2)

The IIC0 to IIC2 registers are used for serial transmission/reception (shift operations) synchronized with the serial clock. These registers can be read or written in 8-bit units, but data should not be written to the IICn register during a data transfer (n = 0 to 2).

Reset input clears these registers to 00H.

After reset: 00H R/W Address: IIC0 FFFFFD80H, IIC1 FFFFFD90H, IIC2 FFFFFDA0H

	7	6	5	4	3	2	1	0
IICn								

(n = 0 to 2)

(9) Slave address registers 0 to 2 (SVA0 to SVA2)

The SVA0 to SVA2 registers hold the I²C bus's slave addresses.

These registers can be read or written in 8-bit units, but bit 0 should be fixed to 0.

Reset input clears these registers to 00H.

After reset: 00H R/W Address: SVA0 FFFFFD83H, SVA1 FFFFFD93H, SVA2 FFFFFDA3H

	7	6	5	4	3	2	1	0
SVAn								0

(n = 0 to 2)

17.5 I²C Bus Mode Functions

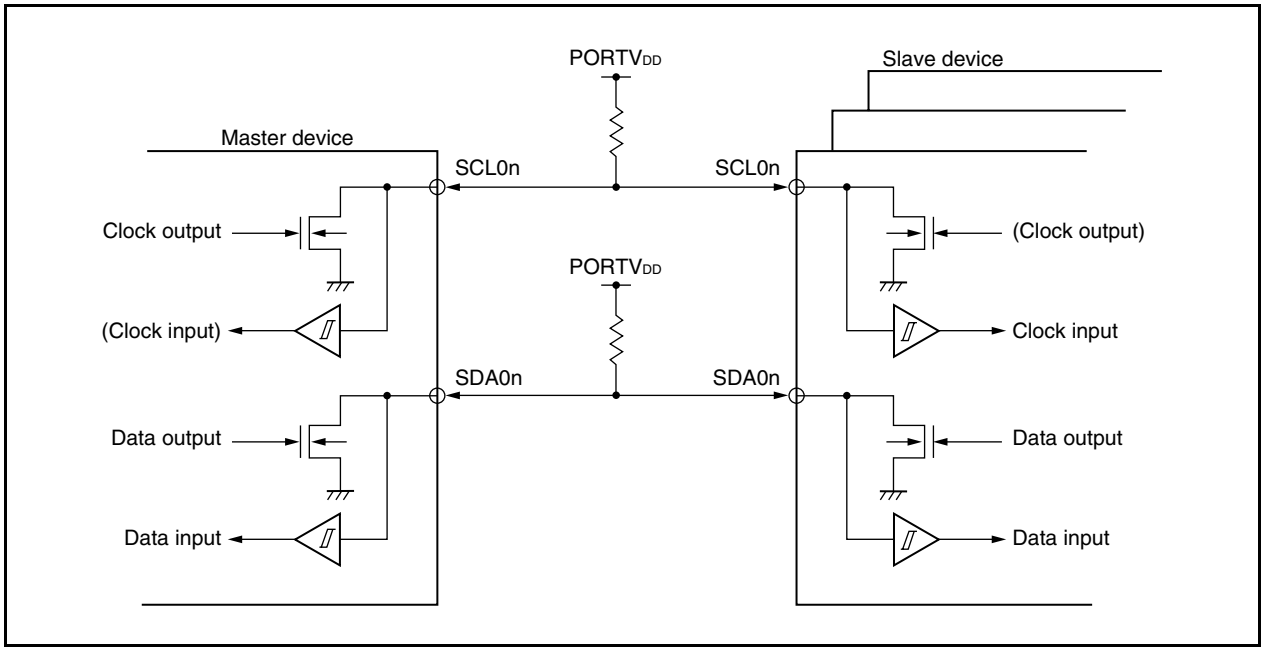
17.5.1 Pin configuration

The serial clock pin (SCL0n) and serial data bus pin (SDA0n) are configured as follows (n = 0 to 2).

SCL0n .....This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.
SDA0n .....This pin is used for serial data input and output.
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.

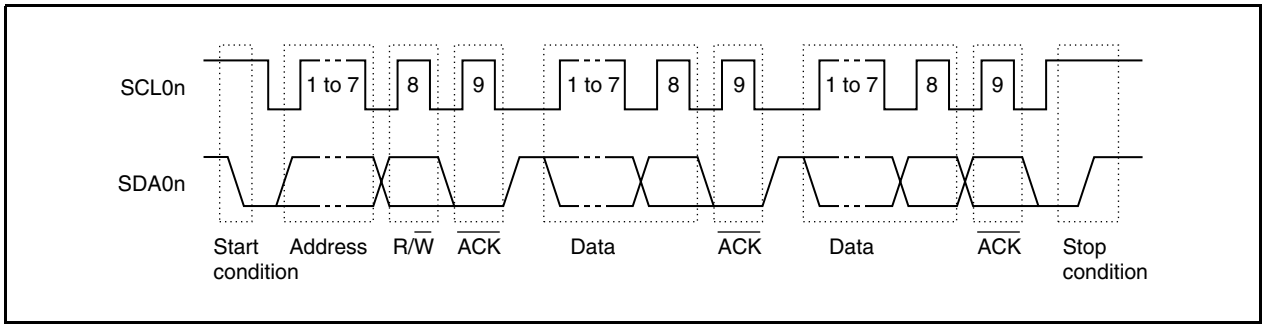
Figure 17-6. Pin Configuration Diagram



17.6 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 17-7. I²C Bus Serial Data Transfer Timing



The master device outputs the start condition, slave address, and stop condition.

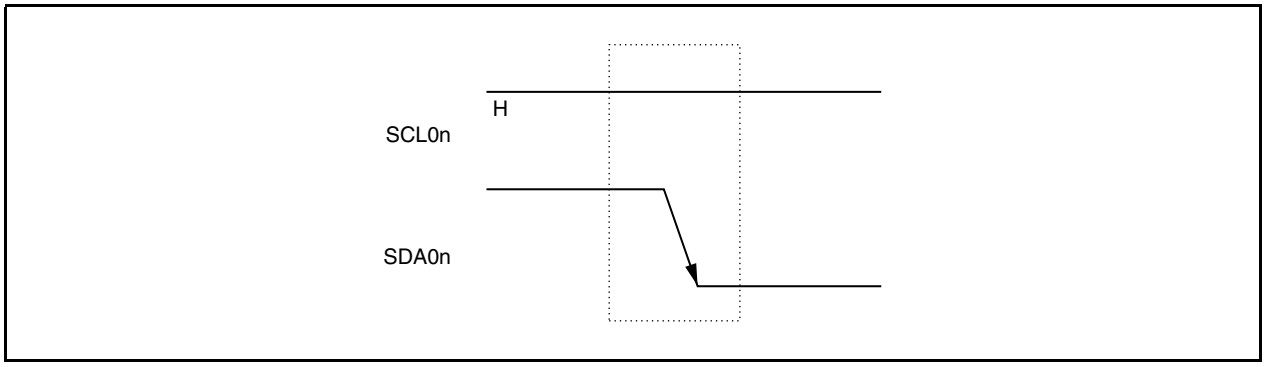
The acknowledge signal ($\overline{ACK}$) can be output by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCL0n) is continuously output by the master device. However, in the slave device, the SCL0n pin's low-level period can be extended and a wait can be inserted (n = 0 to 2).

17.6.1 Start condition

A start condition is met when the SCL0n pin is high level and the SDA0n pin changes from high level to low level. The start conditions for the SCL0n and SDA0n pins 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 (n = 0 to 2).

Figure 17-8. Start Conditions



A start condition is output when the STTn bit of the IICCn register is set (1) after a stop condition has been detected (SPDn bit of IICSn register = 1). When a start condition is detected, the STDn bit of the IICSn register is set (1) (n = 0 to 2).

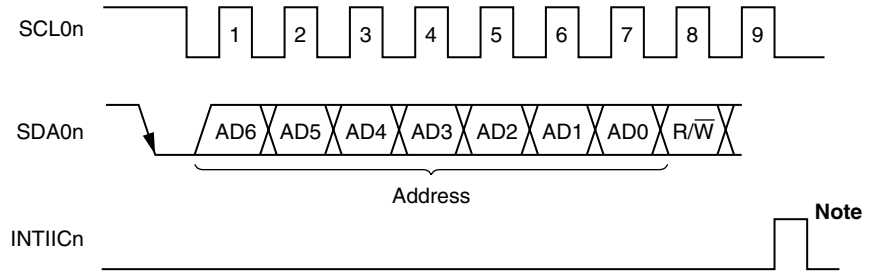
17.6.2 Addresses

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 address data matches the data values stored in the SVAn register. If the address data matches the values of the SVAn register, the slave device is selected and communicates with the master device until the master device transmits a start condition or stop condition (n = 0 to 2).

Figure 17-9. Address



Note The interrupt request signal (INTIICn) is generated if a local address or extension code is received during slave device operation.

Remark n = 0 to 2

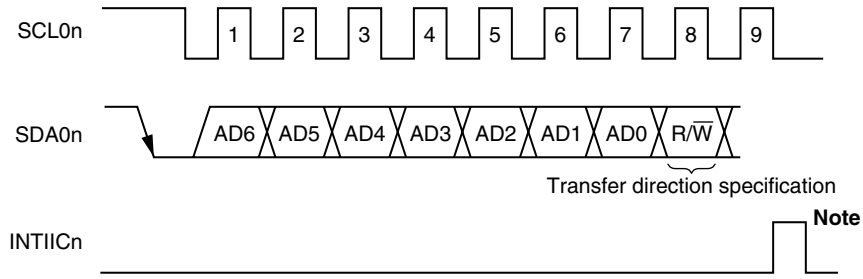
The slave address and the eighth bit, which specifies the transfer direction as described in **17.6.3 Transfer direction specification** below, are written together to IIC shift register n (IICn) and then output. Received addresses are written to the IICn register (n = 0 to 2).

The slave address is assigned to the higher 7 bits of the IICn register.

17.6.3 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 17-10. Transfer Direction Specification



Note The INTIICn signal is generated if a local address or extension code is received during slave device operation.

Remark n = 0 to 2

17.6.4 Acknowledge signal ($\overline{\text{ACK}}$)

The acknowledge signal ($\overline{\text{ACK}}$) is used by the transmitting and receiving devices to confirm serial data reception.

The receiving device returns one $\overline{\text{ACK}}$ signal for each 8 bits of data it receives. The transmitting device normally receives an $\overline{\text{ACK}}$ signal after transmitting 8 bits of data. However, when the master device is the receiving device, it does not output an $\overline{\text{ACK}}$ signal after receiving the final data to be transmitted. The transmitting device detects whether or not an $\overline{\text{ACK}}$ signal is returned after it transmits 8 bits of data. When an $\overline{\text{ACK}}$ signal is returned, the reception is judged as normal and processing continues. If the slave device does not return an $\overline{\text{ACK}}$ signal, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return an $\overline{\text{ACK}}$ signal may be caused by the following two factors.

(a)Reception was not performed normally.

(b)The final data was received.

When the receiving device sets the SDA0n line to low level during the ninth clock, the $\overline{\text{ACK}}$ signal becomes active (normal receive response).

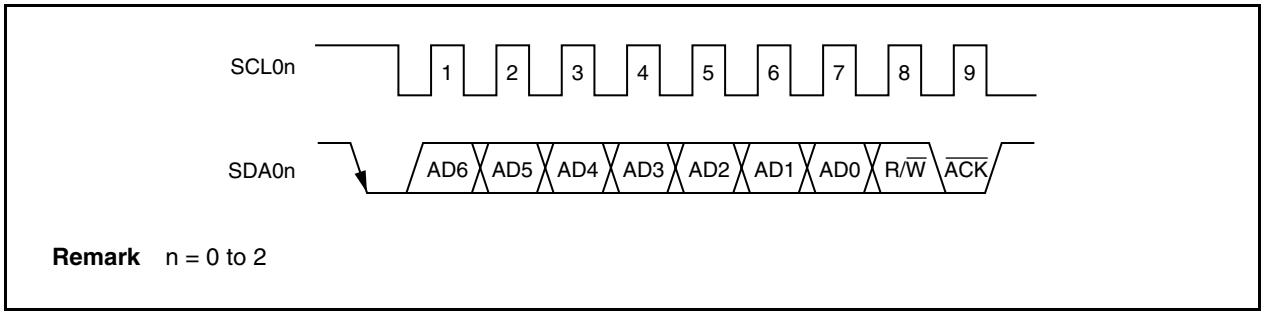
When the ACKEn bit of the IICCn register is set to 1, automatic $\overline{\text{ACK}}$ signal generation is enabled (n = 0 to 2).

Transmission of the eighth bit following the 7 address data bits causes the TRCn bit of the IICSn register to be set. When this TRCn bit's value is 0, it indicates receive mode. Therefore, the ACKEn bit should be set to 1.

When the slave device is receiving (when TRCn bit = 0), if the slave device does not need to receive any more data after receiving several bytes, clearing the ACKEn bit to 0 will prevent the master device from starting transmission of the subsequent data.

Similarly, when the master device is receiving (when TRCn bit = 0) and the subsequent data is not needed and when either a restart condition or a stop condition should therefore be output, clearing the ACKEn bit to 0 will prevent the $\overline{\text{ACK}}$ signal from being returned. This prevents the MSB from being output via the SDA0n line (i.e., stops transmission) during transmission from the slave device.

Figure 17-11. $\overline{\text{ACK}}$ Signal



Remark n = 0 to 2

When the local address is received, an $\overline{\text{ACK}}$ signal is automatically output in synchronization with the falling edge of the SCL0n pin's eighth clock regardless of the value of the ACKEn bit. 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 SCL0n pin's eighth clock if the ACKEn bit is set to 1 before wait cancellation.

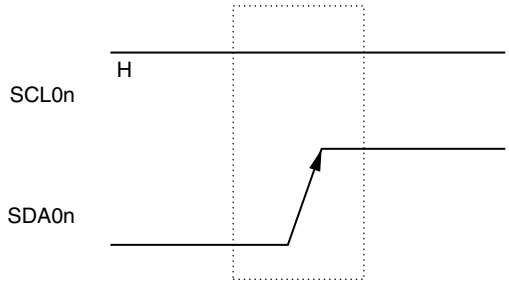
When 9-clock wait is selected: The $\overline{\text{ACK}}$ signal is automatically output at the falling edge of the SCL0n pin's eighth clock if the ACKEn bit has already been set to 1.

17.6.5 Stop condition

When the SCL0n pin is high level, changing the SDA0n pin from low level to high level generates a stop condition (n = 0 to 2).

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 17-12. Stop Condition



Remark n = 0 to 2

A stop condition is generated when the SPTn bit of the IICCn register is set to 1. When the stop condition is detected, the SPDn bit of the IICSn register is set to 1 and the INTIICn register is generated when the SPIEn of the IICCn signal is set to 1 (n = 0 to 2).

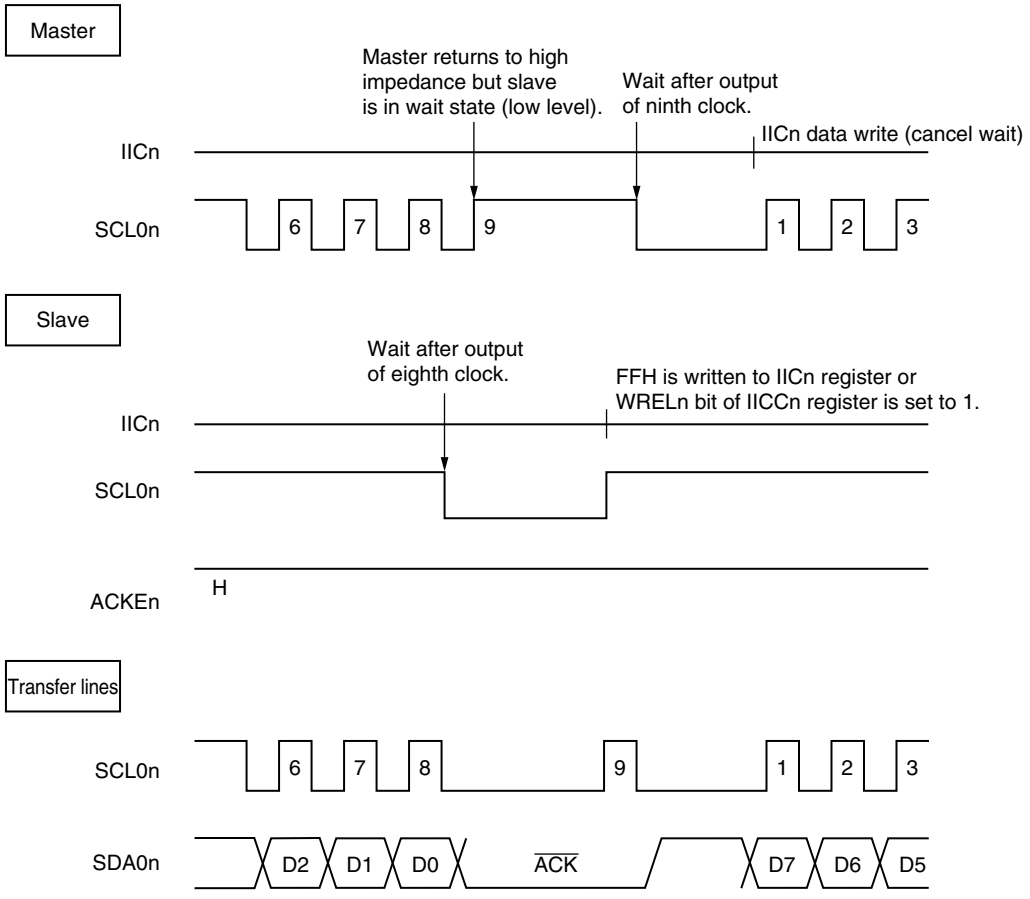
17.6.6 Wait signal ($\overline{WAIT}$)

The wait signal ($\overline{WAIT}$) 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 SCL0n 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 (n = 0 to 2).

Figure 17-13. Wait Signal (1/2)

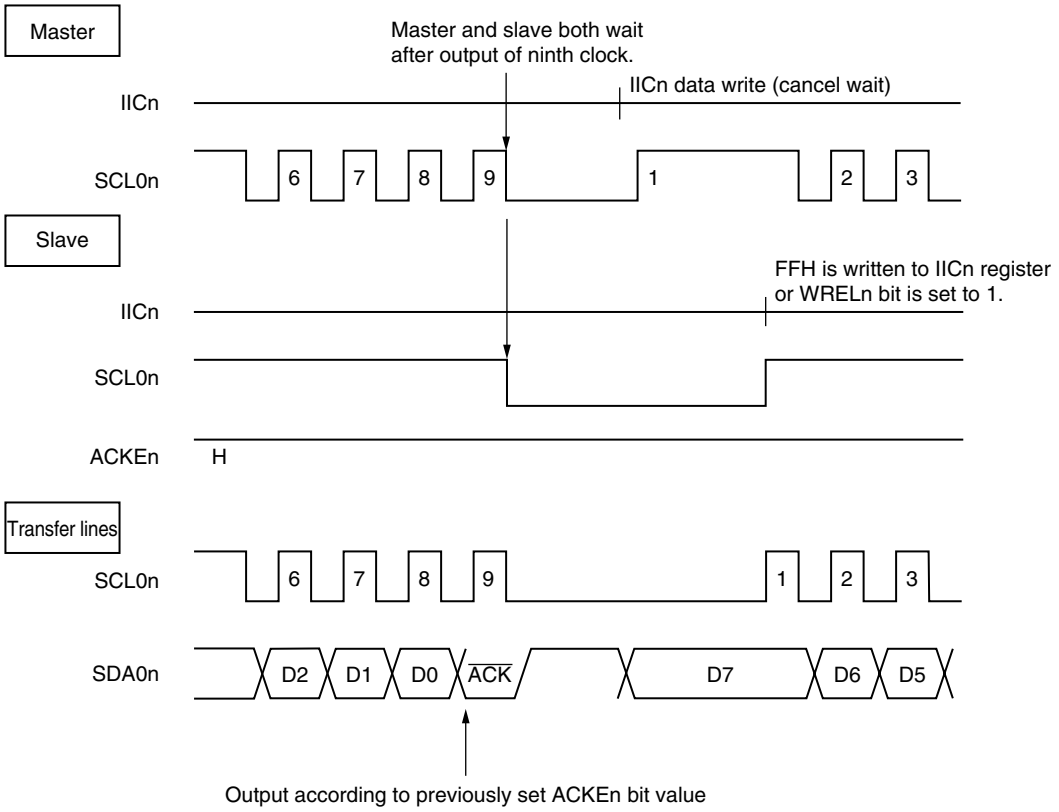
(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master: transmission, slave: reception, and ACKEn bit of IICCn register = 1)



Remark n = 0 to 2

Figure 17-13. Wait Signal (2/2)

(2) When master and slave devices both have a nine-clock wait (master: transmission, slave: reception, and ACKEn bit = 1)



Remark n = 0 to 2

A wait may be automatically generated depending on the setting of the WTIMn bit of the IICCn register (n = 0 to 2).

Normally, when the WRELn bit of the IICCn register is set to 1 or when FFH is written to the IICn register on the receiving side, the wait status is canceled and the transmitting side writes data to the IICn register to cancel the wait status.

The master device can also cancel the wait status via either of the following methods.

- By setting the STTn bit of the IICCn register to 1
- By setting the SPTn bit of the IICCn register to 1

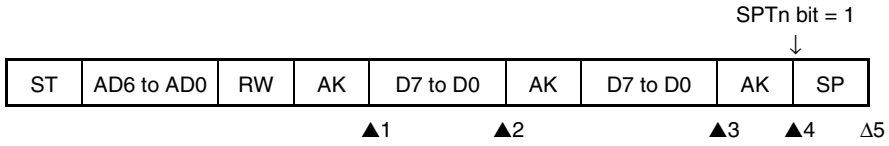
17.7 I²C Interrupt Request Signals (INTIICn)

The following shows the value of the IICSn register at the INTIICn interrupt request signal generation timing and at the INTIICn signal timing (n = 0 to 2).

17.7.1 Master device operation

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

<1> When WTIMn bit = 0



▲1: IICSn register = 10XXX110B

▲2: IICSn register = 10XXX000B

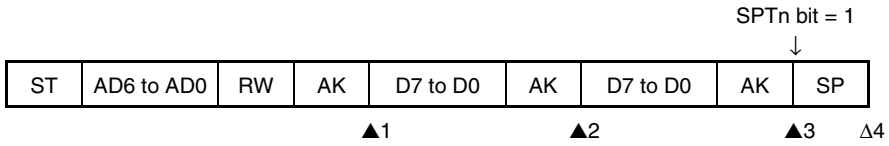
▲3: IICSn register = 10XXX000B (WTIMn bit = 1)

▲4: IICSn register = 10XXXX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

<2> When WTIMn bit = 1



▲1: IICSn register = 10XXX110B

▲2: IICSn register = 10XXX100B

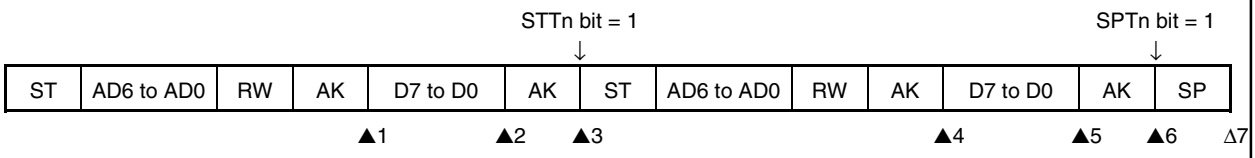
▲3: IICSn register = 10XXXX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

<1> When WTIMn bit = 0



▲1: IICSn register = 10XXX110B

▲2: IICSn register = 10XXX000B (WTIMn bit = 1)

▲3: IICSn register = 10XXXX00B (WTIMn bit = 0)

▲4: IICSn register = 10XXX110B (WTIMn bit = 0)

▲5: IICSn register = 10XXX000B (WTIMn bit = 1)

▲6: IICSn register = 10XXXX00B

Δ 7: IICSn register = 00000001B

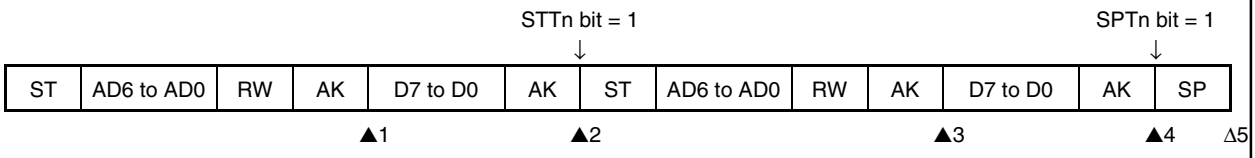
Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1



▲1: IICSn register = 10XXX110B

▲2: IICSn register = 10XXXX00B

▲3: IICSn register = 10XXX110B

▲4: IICSn register = 10XXXX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

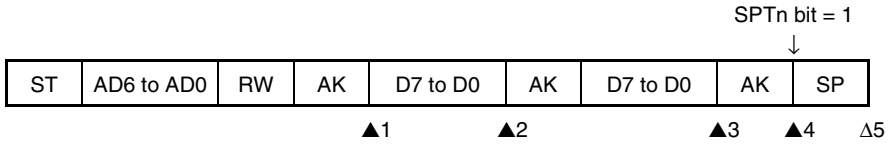
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

<1> When WTIMn bit = 0



▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X000B

▲3: IICSn register = 1010X000B (WTIMn bit = 1)

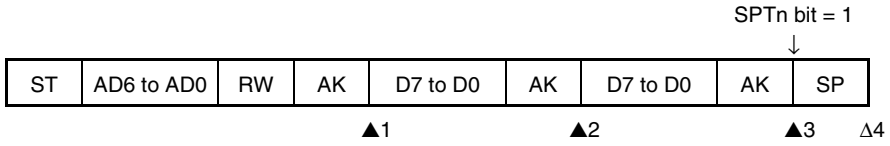
▲4: IICSn register = 1010XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1



▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X100B

▲3: IICSn register = 1010XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0 to 2

17.7.2 Slave device operation (when receiving slave address data (matches with address))

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When WTIMn bit = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

<2> When WTIMn bit = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, match with address)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1	▲2					▲3	▲4	Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, match with address)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1		▲2				▲3	▲4	Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, extension code reception)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1		▲2				▲3		▲4		Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, extension code reception)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1			▲2			▲3	▲4		▲5	Δ6

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X110B

▲5: IICSn register = 0010XX00B

Δ 6: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, mismatch with address (= not extension code))

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1	▲2					▲3		Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 00000X10B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, mismatch with address (= not extension code))

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1		▲2				▲3		Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 00000X10B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

17.7.3 Slave device operation (when receiving extension code)

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When WTIMn bit = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

<2> When WTIMn bit = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 ▲4 Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0 to 2

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, match with address)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1		▲2					▲3	▲4	Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, match with address)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1	▲2		▲3				▲4		▲5 Δ6

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0001X110B

▲5: IICSn register = 0001XX00B

Δ 6: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, extension code reception)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1		▲2				▲3		▲4	Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, extension code reception)

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1	▲2		▲3			▲4	▲5		▲6 Δ7

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0010X010B

▲5: IICSn register = 0010X110B

▲6: IICSn register = 0010XX00B

Δ 7: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, mismatch with address (= not extension code))

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1		▲2					▲3		Δ4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 00000X10B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1 (after restart, mismatch with address (= not extension code))

ST	AD6 to AD0	RW	AK	D7 to D0	AK	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1	▲2		▲3				▲4		Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 00000X10B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

17.7.4 Operation without communication

(1) Start ~ Code ~ Data ~ Data ~ Stop

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

Δ1

Δ 1: IICSn register = 00000001B

Remarks 1. Δ: Generated only when SPIEn bit = 1

2. n = 0 to 2

17.7.5 Arbitration loss operation (operation as slave after arbitration loss)

(1) When arbitration loss occurs during transmission of slave address data

<1> When WTIMn bit = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0101X110B (Example: when ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0101X110B (Example: when ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(2) When arbitration loss occurs during transmission of extension code

<1> When WTIMn bit = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP
			▲1		▲2		▲3	Δ4

▲1: IICSn register = 0110X010B (Example: when ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

<2> When WTIMn bit = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP
			▲1	▲2		▲3	▲4	Δ5

▲1: IICSn register = 0110X010B (Example: when ALDn bit is read during interrupt servicing)

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

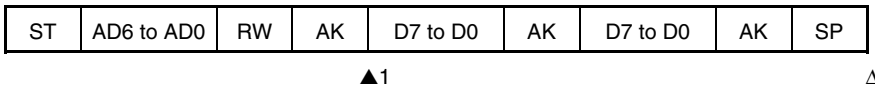
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

17.7.6 Operation when arbitration loss occurs (no communication after arbitration loss)

(1) When arbitration loss occurs during transmission of slave address data



▲1: IICSn register = 01000110B (Example: when ALDn bit is read during interrupt servicing)

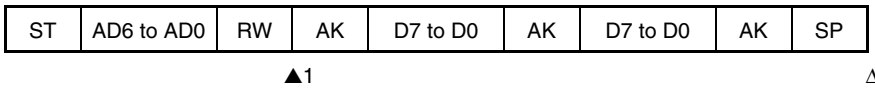
Δ 2: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

2. n = 0 to 2

(2) When arbitration loss occurs during transmission of extension code



▲1: IICSn register = 0110X010B (Example: when ALDn bit is read during interrupt servicing)

IICCn's LRELn bit is set to 1 by software

Δ 2: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(3) When arbitration loss occurs during data transfer

<1> When WTIMn bit = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 Δ3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000000B (Example: when ALDn bit is read during interrupt servicing)

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

2. n = 0 to 2

<2> When WTIMn bit = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP

▲1 ▲2 Δ3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000100B (Example: when ALDn bit is read during interrupt servicing)

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

2. n = 0 to 2

(4) When arbitration loss occurs due to restart condition during data transfer

<1> Not extension code (Example: mismatches with address)

ST	AD6 to AD0	RW	AK	D7 to Dn	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP

▲1 ▲2 Δ3

▲1: IICSn register = 1000X110B

▲2: IICSn register = 01000110B (Example: when ALDn bit is read during interrupt servicing)

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. Dn = D6 to D0
n = 0 to 2

<2> Extension code

ST	AD6 to AD0	RW	AK	D7 to Dn	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP

▲1 ▲2 Δ3

▲1: IICSn register = 1000X110B

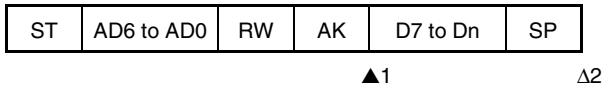
▲2: IICSn register = 0110X010B (Example: when ALDn bit is read during interrupt servicing)

IICCn's LRELn bit is set to 1 by software

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. Dn = D6 to D0
n = 0 to 2

(5) When arbitration loss occurs due to stop condition during data transfer



▲1: IICSn register = 1000X110B

Δ 2: IICSn register = 01000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

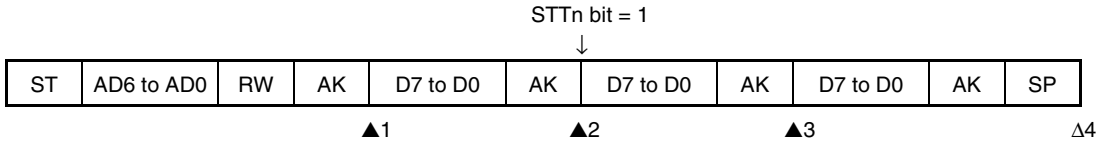
X: don't care

2. Dn = D6 to D0

n = 0 to 2

(6) When arbitration loss occurs due to low-level of SDA0n pin when attempting to generate a restart condition

When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 01000100B (Example: when ALDn bit is read during interrupt servicing)

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

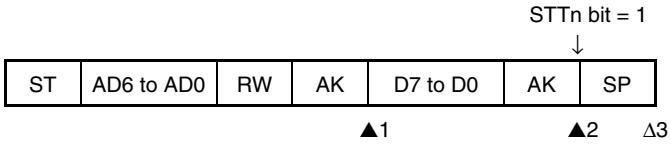
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(7) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

Δ 3: IICSn register = 01000001B

Remarks 1. ▲: Always generated

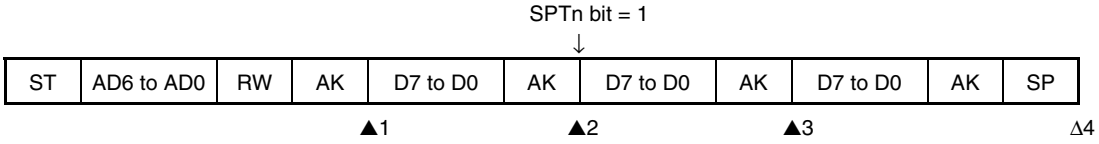
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

(8) When arbitration loss occurs due to low-level of SDA0n pin when attempting to generate a stop condition

When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 01000000B (Example: when ALDn bit is read during interrupt servicing)

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0 to 2

17.8 Interrupt Request Signal (INTIICn) Generation Timing and Wait Control

The setting of the WTIMn bit of the IICCn register determines the timing by which the INTIICn register is generated and the corresponding wait control, as shown below (n = 0 to 2).

Table 17-3. INTIICn Generation Timing and Wait Control

WTIMn Bit	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 INTIICn signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to the SVAn register.
At this point, the $\overline{ACK}$ signal is output regardless of the value set to the ACKEn bit of the IICCn register. For a slave device that has received an extension code, the INTIICn signal occurs at the falling edge of the eighth clock.

2. If the received address does not match the contents of the SVAn register, neither the INTIICn signal nor a wait occurs.

Remarks 1. 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.

2. n = 0 to 2

(1) During address transmission/reception

- Slave device operation: Interrupt and wait timing are determined regardless of the WTIMn bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIMn bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By setting the WRELn bit of the IICCn register to 1
- By writing to the IICn register
- By start condition setting (STTn bit of IICCn register = 1)
- By step condition setting (SPTn bit of IICCn register = 1)

When an 8-clock wait has been selected (WTIMn bit = 0), the output level of the $\overline{ACK}$ signal must be determined prior to wait cancellation.

Remark n = 0 to 2

(5) Stop condition detection

The INTIICn signal is generated when a stop condition is detected.

Remark n = 0 to 2

17.9 Address Match Detection Method

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. The INTIICn signal occurs when a local address has been set to the SVAn register and when the address set to the SVAn register matches the slave address sent by the master device, or when an extension code has been received (n = 0 to 2).

17.10 Error Detection

In I²C bus mode, the status of the serial data bus pin (SDA0n) during data transmission is captured by the IICn register of the transmitting device, so the data of the IICn register prior to transmission can be compared with the transmitted IICn data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match (n = 0 to 2).

17.11 Extension Code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (EXCn bit of IICSn register) is set for extension code reception and an interrupt request signal (INTIICn) is issued at the falling edge of the eighth clock (n = 0 to 2).
The local address stored in the SVAn register is not affected.

(2) If 11110xx0 is set to the SVAn register by a 10-bit address transfer and 11110xx0 is transferred from the master device, the results are as follows. Note that the INTIICn signal occurs at the falling edge of the eighth clock (n = 0 to 2)

- Higher four bits of data match: EXCn bit = 1
- Seven bits of data match: COIn bit of IICSn register = 1

(3) Since the processing after the interrupt request signal 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 the LRELn bit of the IICCn register to 1 and the CPU will enter the next communication wait state.

Table 17-4. 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

17.12 Arbitration

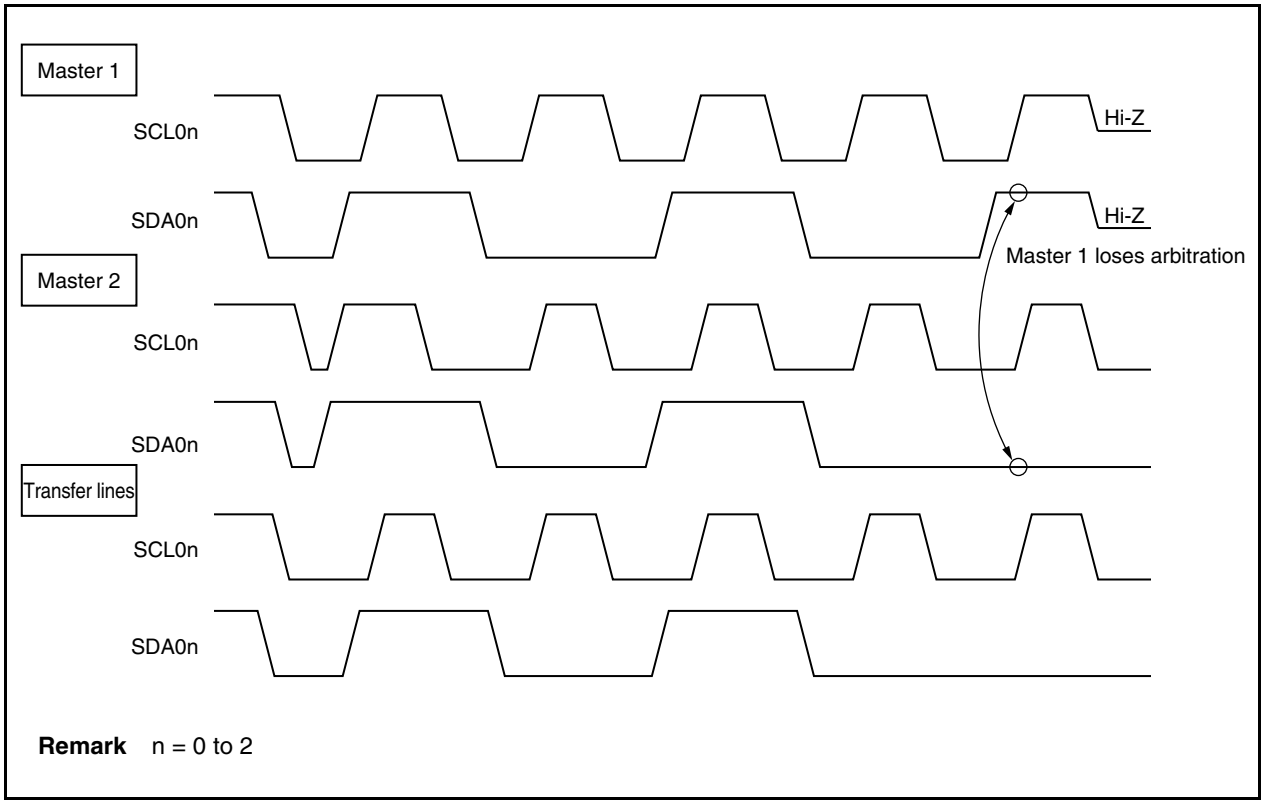
When several master devices simultaneously output a start condition (when IICCn.STTn bit is set to 1 before IICSn.STDn bit is set to 1), communication among the master devices is performed while the number of clocks are being adjusted until the data differs. This kind of operation is called arbitration (n = 0 to 2).

When one of the master devices loses in arbitration, an arbitration loss flag (ALDn bit of IICSn register) is set via the timing by which the arbitration loss occurred, and the SCL0n and SDA0n lines are both set to high impedance, which releases the bus (n = 0 to 2).

The arbitration loss is detected based on the timing of the next interrupt request signal (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALDn bit = 1 setting that has been made by software (n = 0 to 2).

For details of interrupt request timing, see **17.7 I²C Interrupt Request Signals (INTIICn)**.

Figure 17-14. Arbitration Timing Example



Remark n = 0 to 2

Table 17-5. Status During Arbitration and Interrupt Request Signal 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 $\overline{ACK}$ signal transfer period after data reception	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is output (when IICCn.SPIEn bit = 1)[Note 2]
When the SDA0n pin 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 IICCn.SPIEn bit = 1)[Note 2]
When the DSA0n pin 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 the SCL0n pin is at low level while attempting to output a restart condition	

Notes 1. When the WTIMn bit of the IICCn register = 1, an interrupt request signal occurs at the falling edge of the ninth clock. When the WTIMn bit = 0 and the extension code's slave address is received, an interrupt request signal occurs at the falling edge of the eighth clock (n = 0 to 2).

2. When there is a possibility that arbitration will occur, set the SPIEn bit = 1 for master device operation (n = 0 to 2).

17.13 Wakeup Function

The I²C bus slave function is a function that generates an interrupt request signal (INTIICn) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary interrupt request signals from occurring when addresses do not match.

When a start condition is detected, wake-up standby mode is set. This wake-up 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, the SPIEn bit of the IICCn register is set regardless of the wake up function, and this determines whether interrupt request signals are enabled or disabled (n = 0 to 2).

17.14 Communication Reservation

17.14.1 When communication reservation function is enabled (IICRSVn bit of IICFn register = 0)

To start master device communications when not currently using the 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 ($\overline{ACK}$ signal is not returned and the bus was released when the LRELn bit of the IICCn register was set to “1”) (n = 0 to 2).

If the STTn bit of the IICCn register is set while the bus is not used, a start condition is automatically generated and 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 the IICn register causes the master’s address transfer to start. At this point, the SPIEn bit of the IICCn register should be set (n = 0 to 2).

When STTn has been set, the operation mode (as start condition or as communication reservation) is determined according to the bus status (n = 0 to 2).

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 the STTn bit, set the STTn bit, wait for the wait period, then check the MSTSn bit of the IICSn register) (n = 0 to 2).

The wait periods, which should be set via software, are listed in Table 17-6. These wait periods can be set by the SMCn, CLn1, and CLn0 bits of the IICCLn register and the CLXn bit of the IICXn register (n = 0 to 2).

Table 17-6. Wait Periods

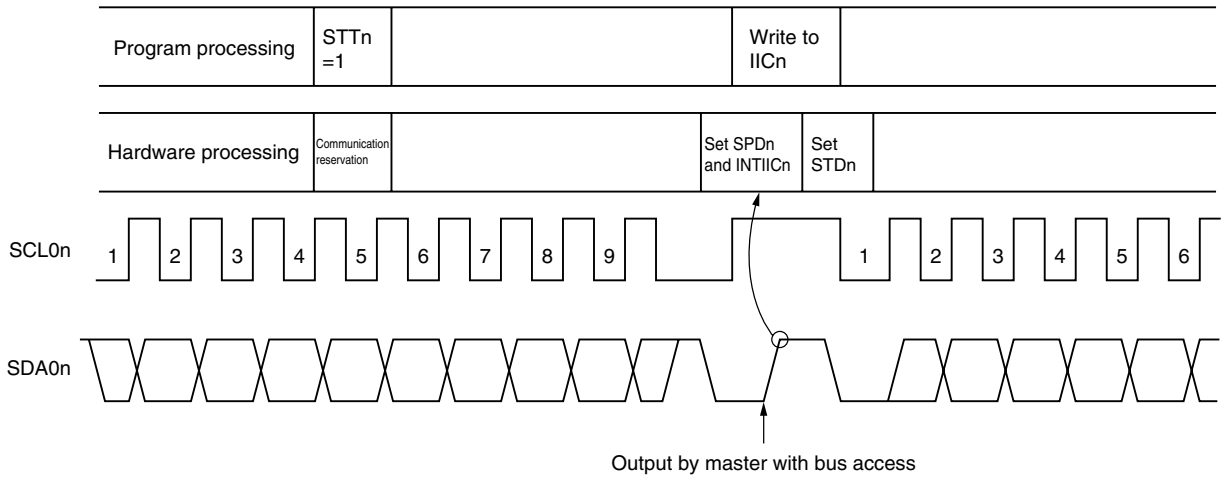
Selection Clock	CLXn	SMCn	CLn1	CLn0	Wait Period
f_{XX} (when OCKSm = 14H set)	0	0	0	0	26 clocks
$f_{XX}/2$ (when OCKSm = 10H set)	0	0	0	0	52 clocks
$f_{XX}/3$ (when OCKSm = 11H set)	0	0	0	0	78 clocks
$f_{XX}/4$ (when OCKSm = 12H set)	0	0	0	0	104 clocks
$f_{XX}/5$ (when OCKSm = 13H set)	0	0	0	0	130 clocks
f_{XX} (when OCKSm = 14H set)	0	0	0	1	47 clocks
$f_{XX}/2$ (when OCKSm = 10H set)	0	0	0	1	94 clocks
$f_{XX}/3$ (when OCKSm = 11H set)	0	0	0	1	141 clocks
$f_{XX}/4$ (when OCKSm = 12H set)	0	0	0	1	188 clocks
f_{XX}	0	0	1	0	47 clocks
f_{XX} (when OCKSm = 14H set)	0	1	0	×	16 clocks
$f_{XX}/2$ (when OCKSm = 10H set)	0	1	0	×	32 clocks
$f_{XX}/3$ (when OCKSm = 11H set)	0	1	0	×	48 clocks
$f_{XX}/4$ (when OCKSm = 12H set)	0	1	0	×	64 clocks
f_{XX}	0	1	1	0	16 clocks
f_{XX} (when OCKSm = 14H set)	1	1	0	×	10 clocks
$f_{XX}/2$ (when OCKSm = 10H set)	1	1	0	×	20 clocks
$f_{XX}/3$ (when OCKSm = 11H set)	1	1	0	×	30 clocks
$f_{XX}/4$ (when OCKSm = 12H set)	1	1	0	×	40 clocks
f_{XX}	1	1	1	0	10 clocks

Remarks 1. n = 0 to 2
m = 0, 1

2. × = don't care

The communication reservation timing is shown below.

Figure 17-15. Communication Reservation Timing

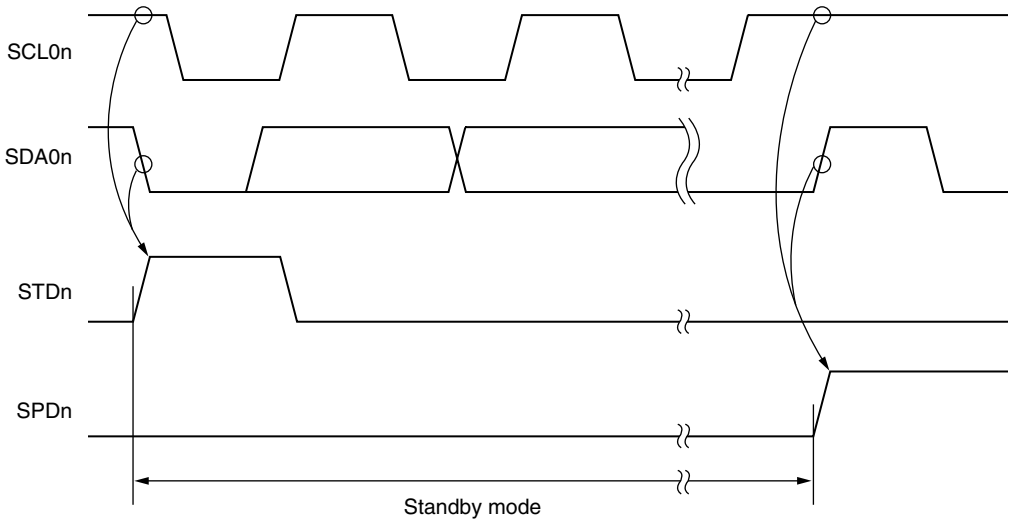


IICn: IIC shift register n
STTn: Bit 1 of IIC control register n (IICCn)
STDn: Bit 1 of IIC status register n (IICSn)
SPDn: Bit 0 of IIC status register n (IICSn)

Remark n = 0 to 2

Communication reservations are accepted via the following timing. After the STDn bit of the IICSn register is set to 1, a communication reservation can be made by setting the STTn bit of the IICCn register to 1 before a stop condition is detected (n = 0 to 2).

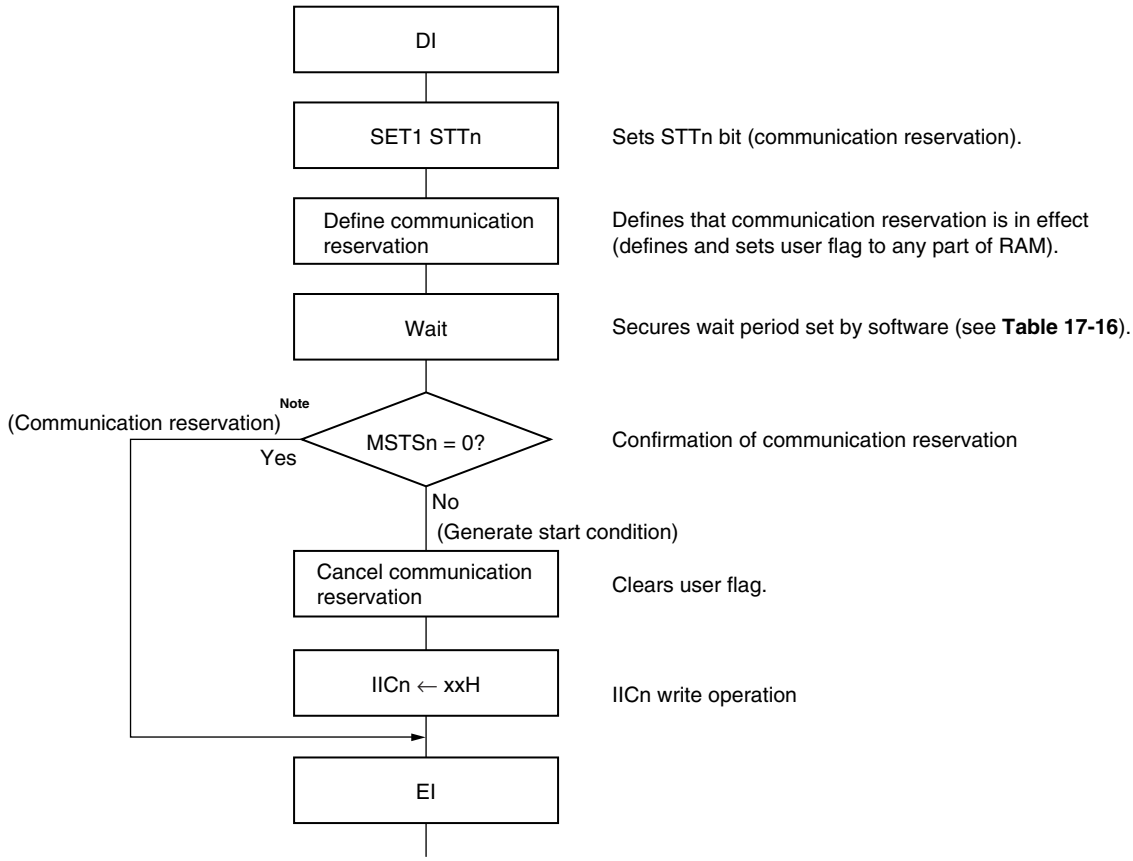
Figure 17-16. Timing for Accepting Communication Reservations



Remark n = 0 to 2

The communication reservation flowchart is illustrated below.

Figure 17-17. Communication Reservation Flowchart



Note The communication reservation operation executes a write to the IICn register when a stop condition interrupt request occurs.

Remark n = 0 to 2

17.14.2 When communication reservation function is disabled (IICRSVn bit of IICFn register = 1)

When the STTn bit of the IICCn register is set when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ signal is not returned and the bus was released when the LRELn bit of the IICCn register was set to 1) (n = 0 to 2).

To confirm whether the start condition was generated or request was rejected, check the STCFn flag of the IICFn register. The time shown in Table 17-7 is required until the STCFn flag is set after setting the STTn bit = 1. Therefore, secure the time by software.

Table 17-7. Wait Periods

OCKSENn	OCKSn1	OCKSn0	CLn1	CLn0	Wait Period
1	0	0	0	×	6 clocks
1	0	1	0	×	9 clocks
1	1	0	0	×	12 clocks
1	1	1	0	×	15 clocks
0	0	0	1	0	3 clocks

Remarks **1.** ×: don't care
2. n = 0 to 2

17.15 Cautions

(1) When STCENn bit of IICFn register = 0

Immediately after the I²C0n operation is enabled, the bus communication status (IICBSYn bit of IICFn register = 1) is recognized regardless of the actual bus status. To execute master communication in the status where a stop condition has not been detected, generate a stop condition and then release the bus before starting the master communication.

Use the following sequence for generating a stop condition.

<1> Set the IICCLn register.
<2> Set the IICEn bit of the IICCn register.
<3> Set the SPTn bit of the IICCn register.

(2) When STCENn bit of IICFn register = 1

Immediately after I²C0n operation is enabled, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status. To issue the first start condition (STTn bit of IICCn register = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

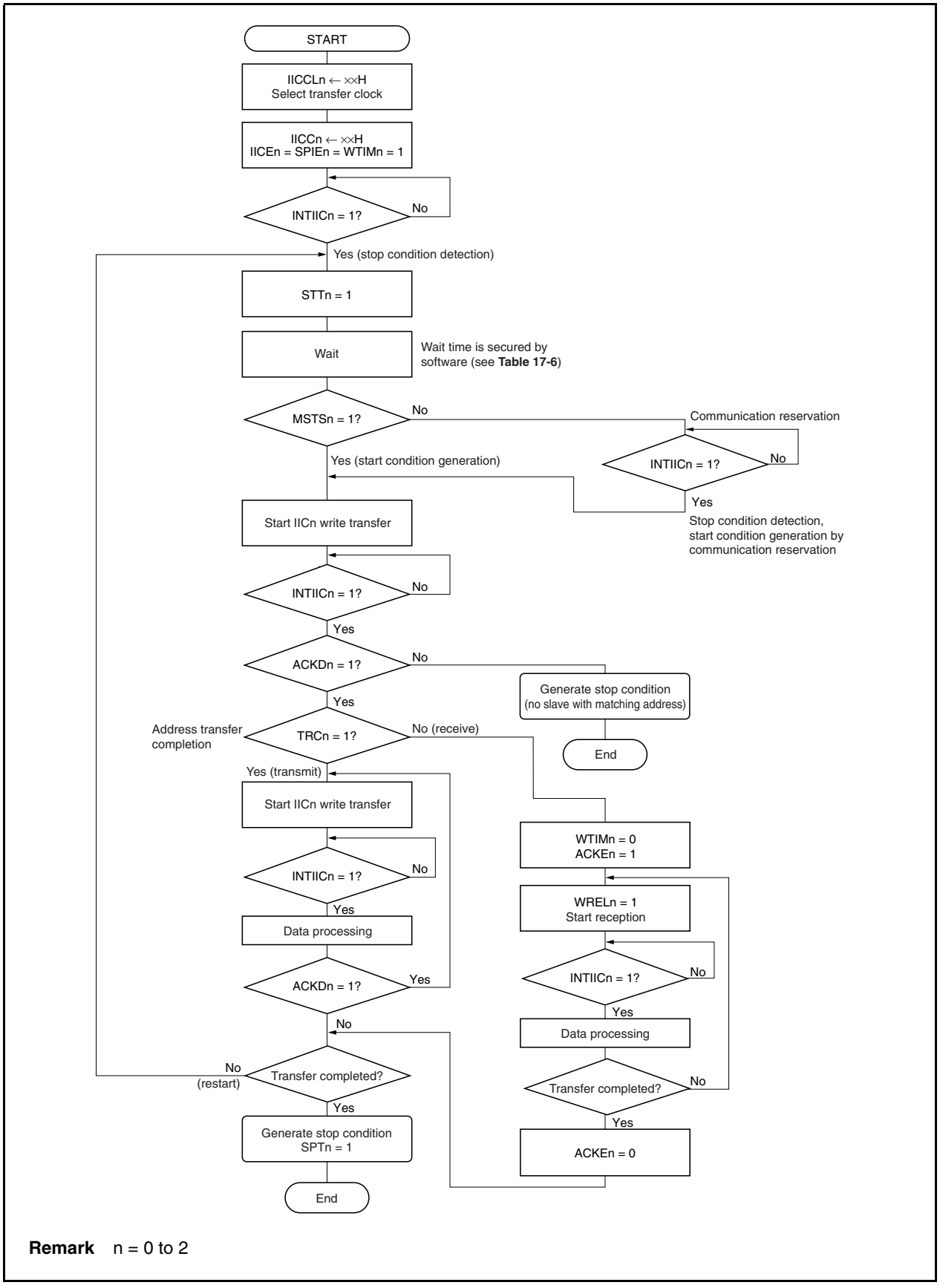
Remark n = 0 to 2

17.16 Communication Operations

17.16.1 Master operation 1

The following shows the flowchart for master communication when the communication reservation function is enabled (IICRSVn bit of IICFn register = 0) and the master operation is started after a stop condition is detected (STCENn bit of IICFn register = 0).

Figure 17-18. Master Operation Flowchart (1)

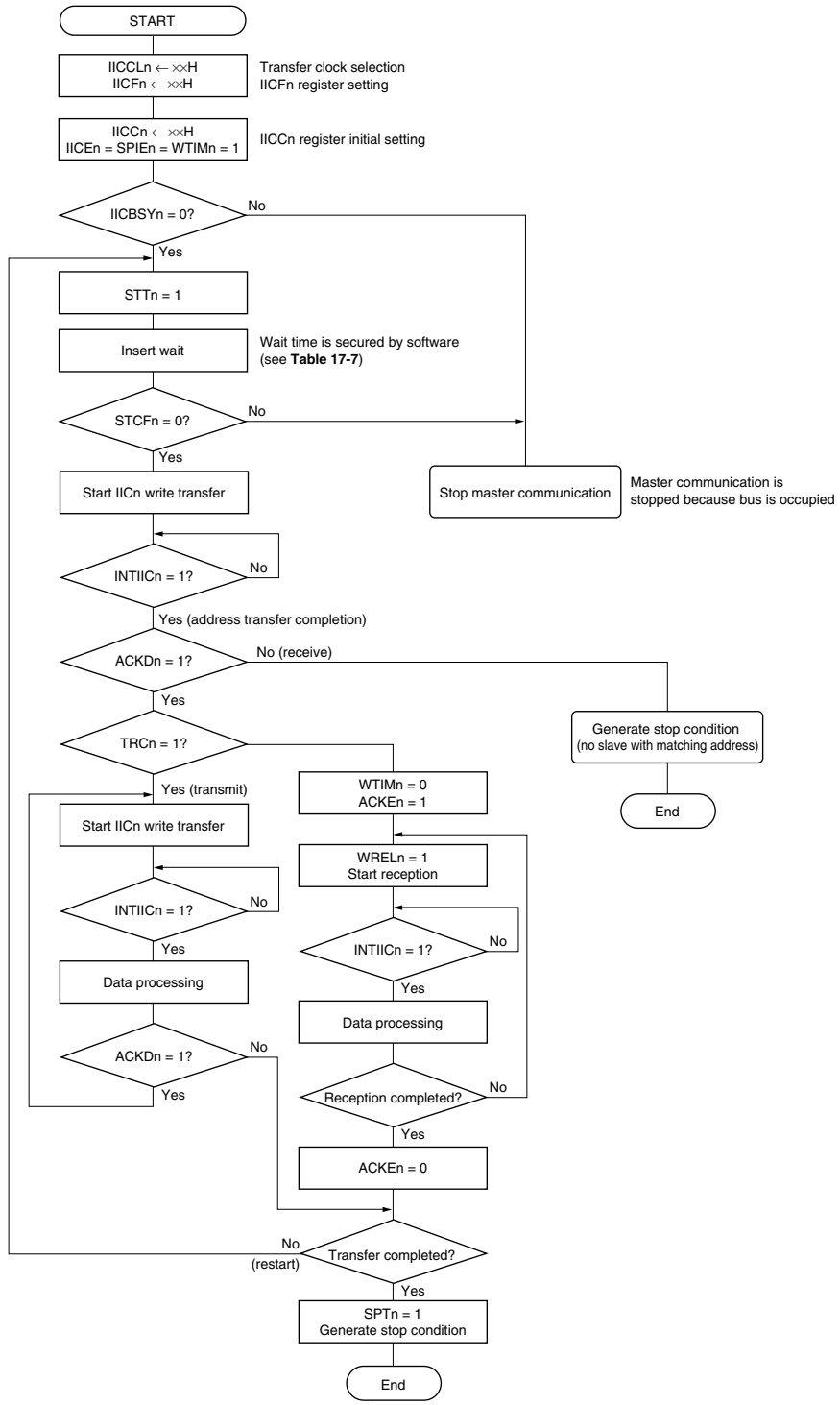


Remark n = 0 to 2

17.16.2 Master operation 2

The following shows the flowchart for master communication when the communication reservation function is disabled (IICRSVn bit = 1) and the master operation is started without detecting a stop condition (STCENn bit = 1).

Figure 17-19. Master Operation Flowchart (2)

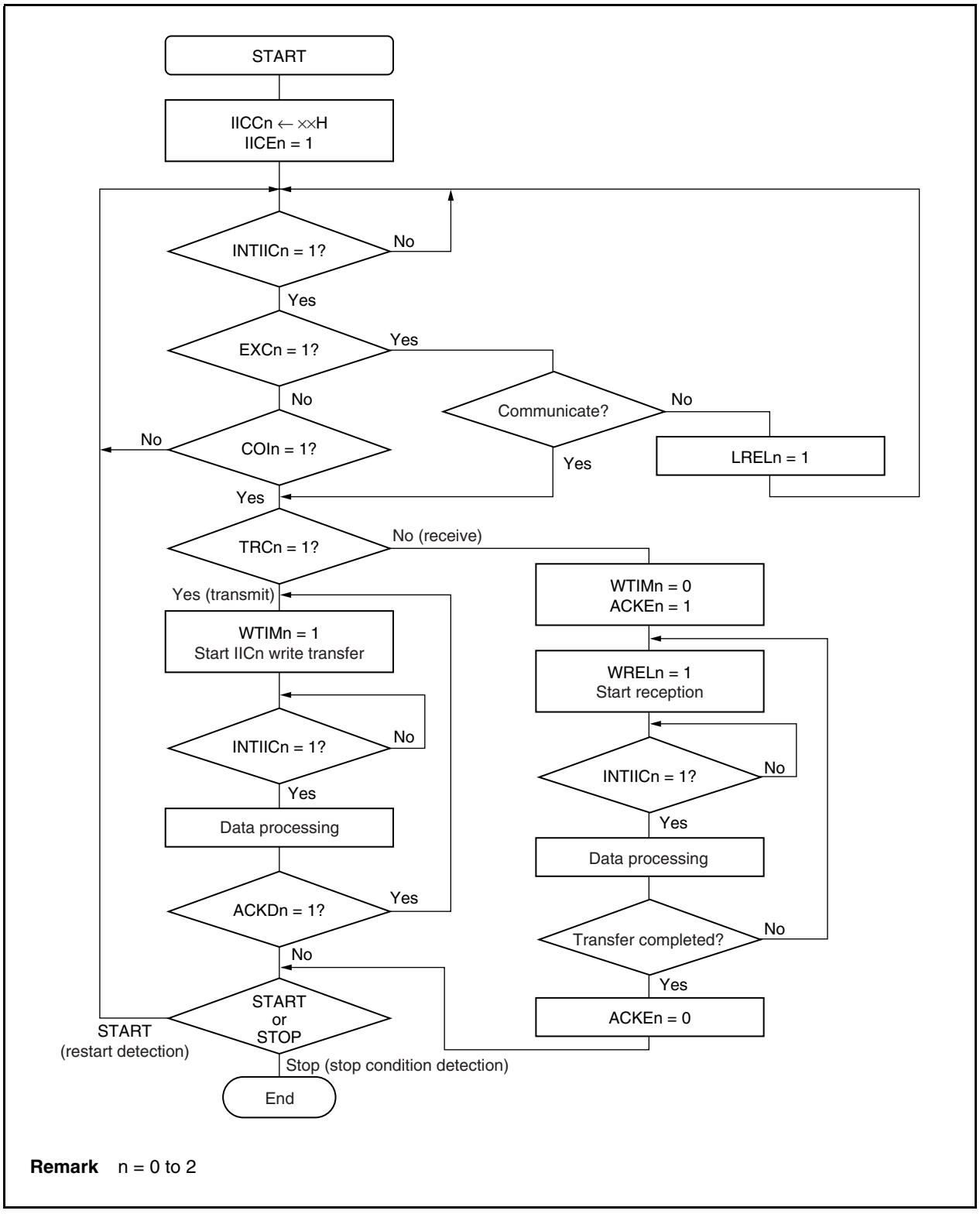


Remark n = 0 to 2

17.16.3 Slave operation

The following shows the flowchart for slave communication.

Figure 17-20. Slave Operation Flowchart



Remark n = 0 to 2

17.17 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 TRCn bit of the IICSn register, which specifies the data transfer direction, and then starts serial communication with the slave device.

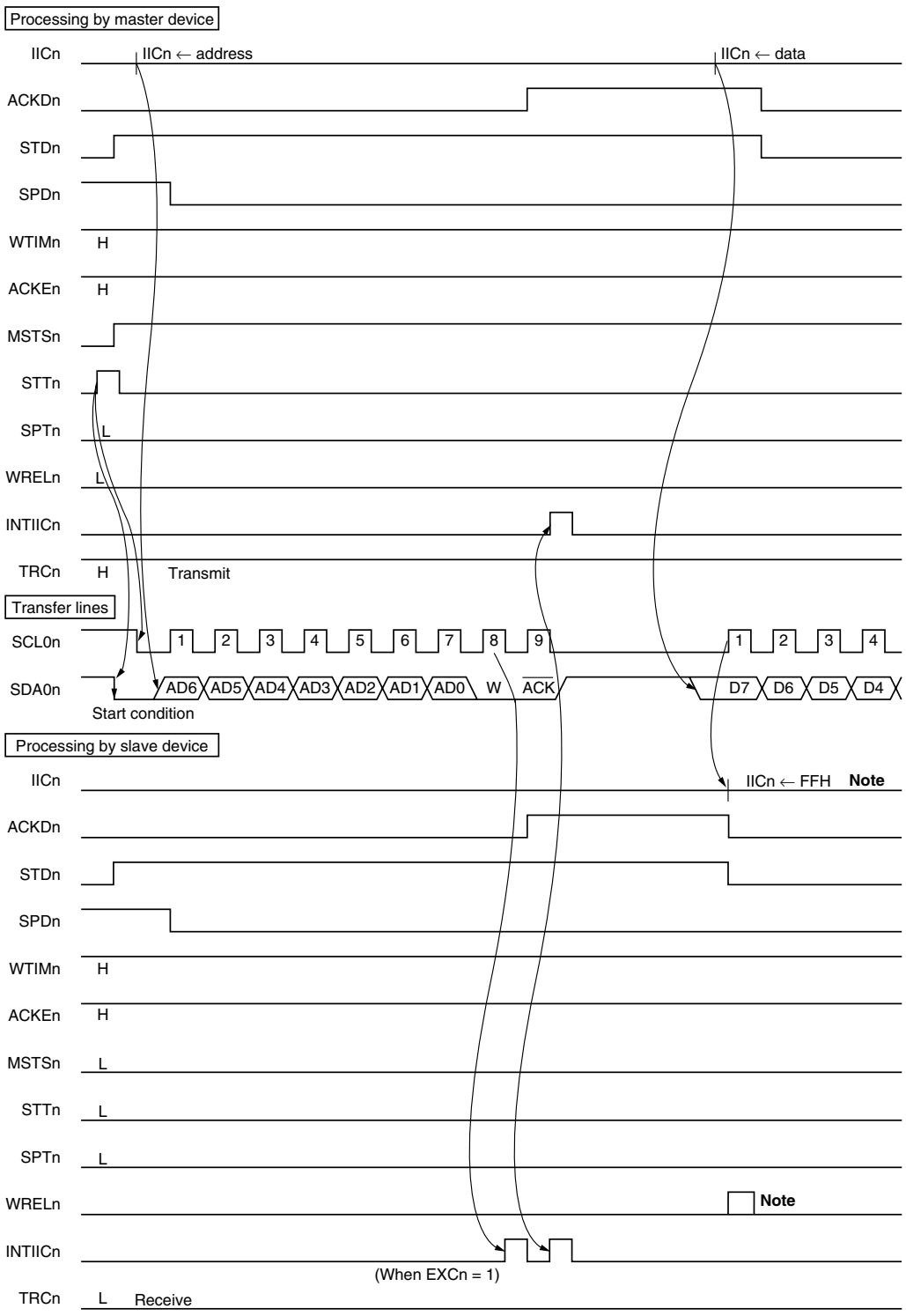
The shift operation of the IICn register is synchronized with the falling edge of the serial clock pin (SCL0n). The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0n pin.

Data input via the SDA0n pin is captured by the IICn register at the rising edge of the SCL0n pin.

The data communication timing is shown below.

Remark n = 0 to 2

Figure 17-21. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)

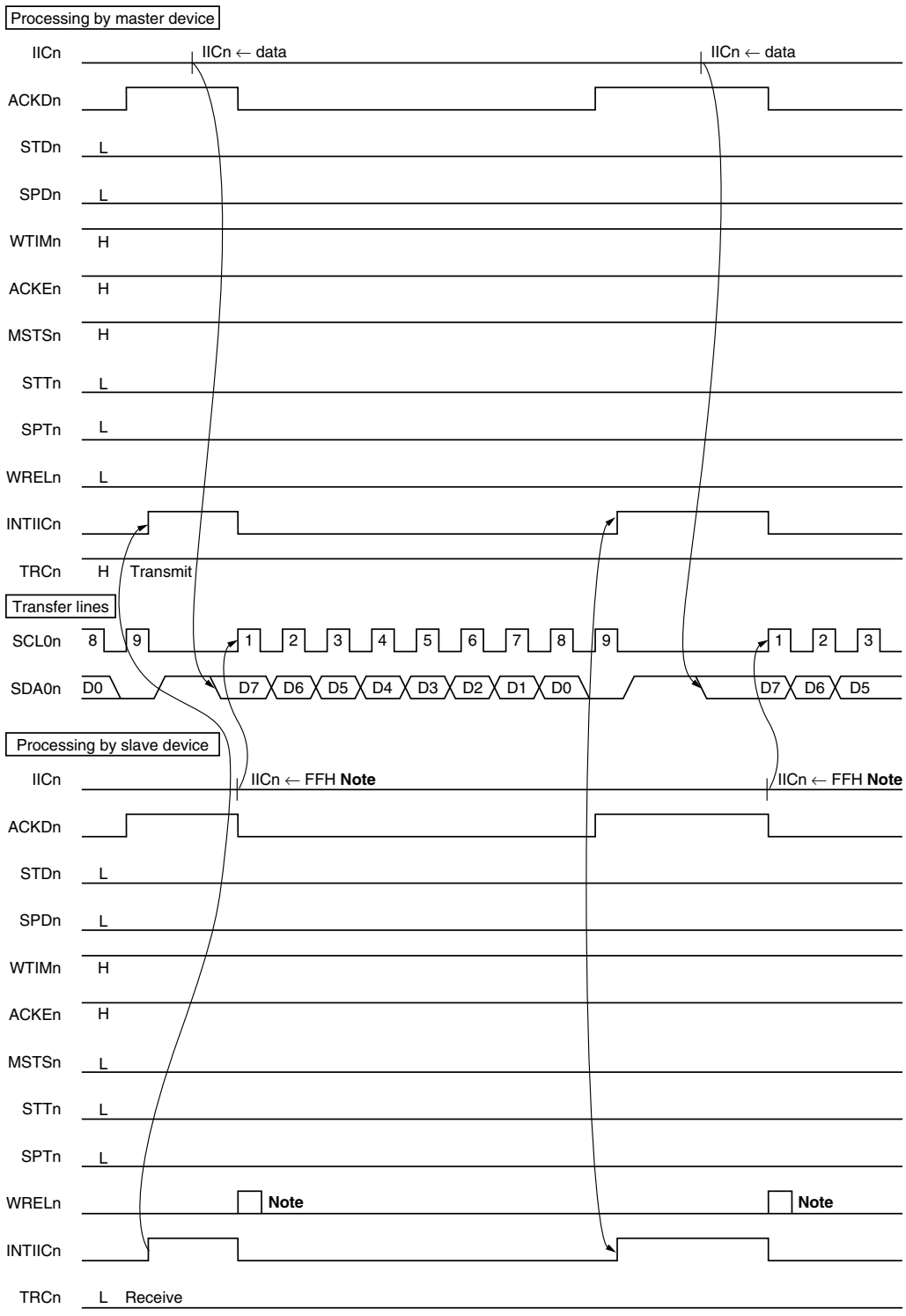


Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

Figure 17-21. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)

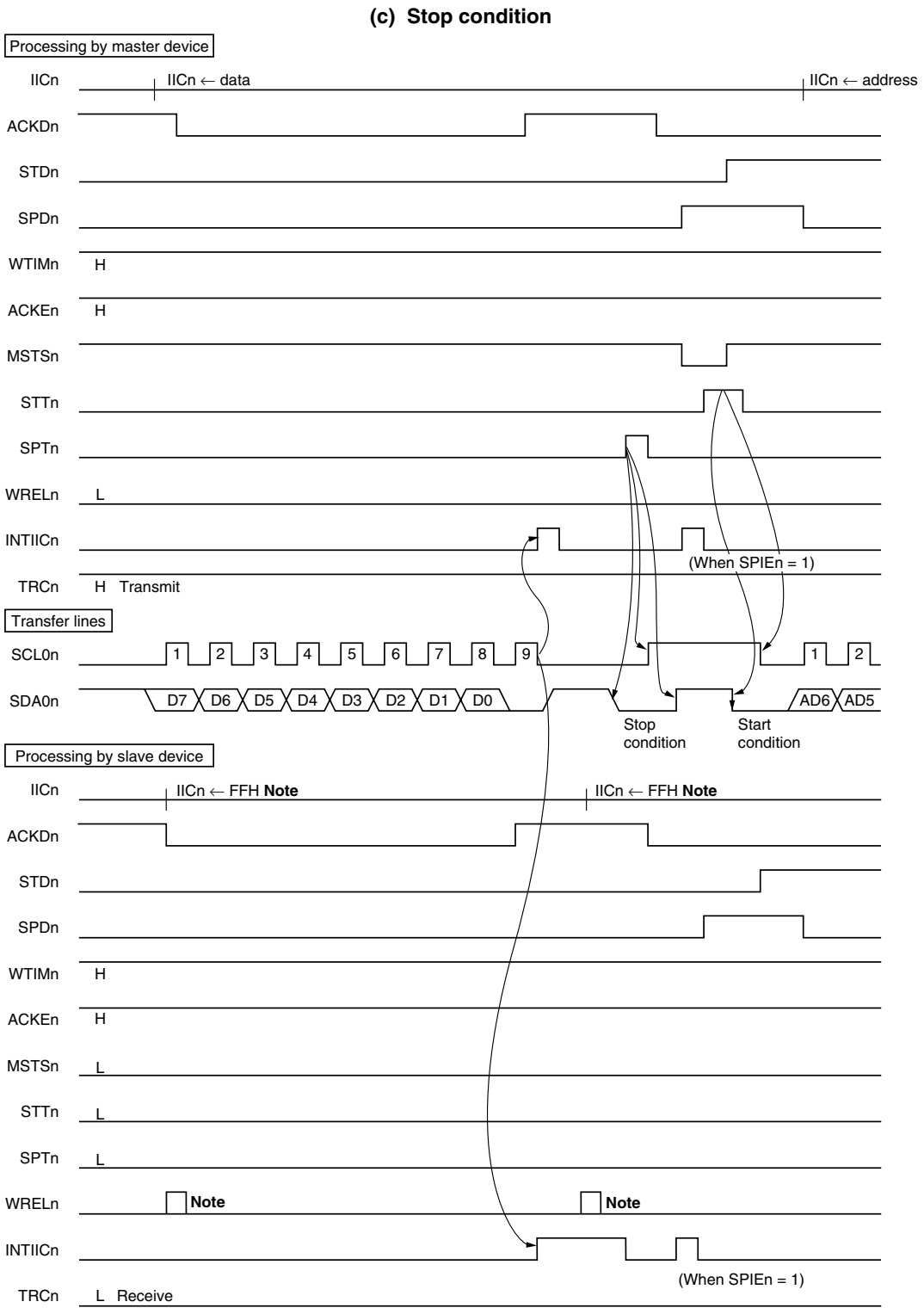
(b) Data



Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

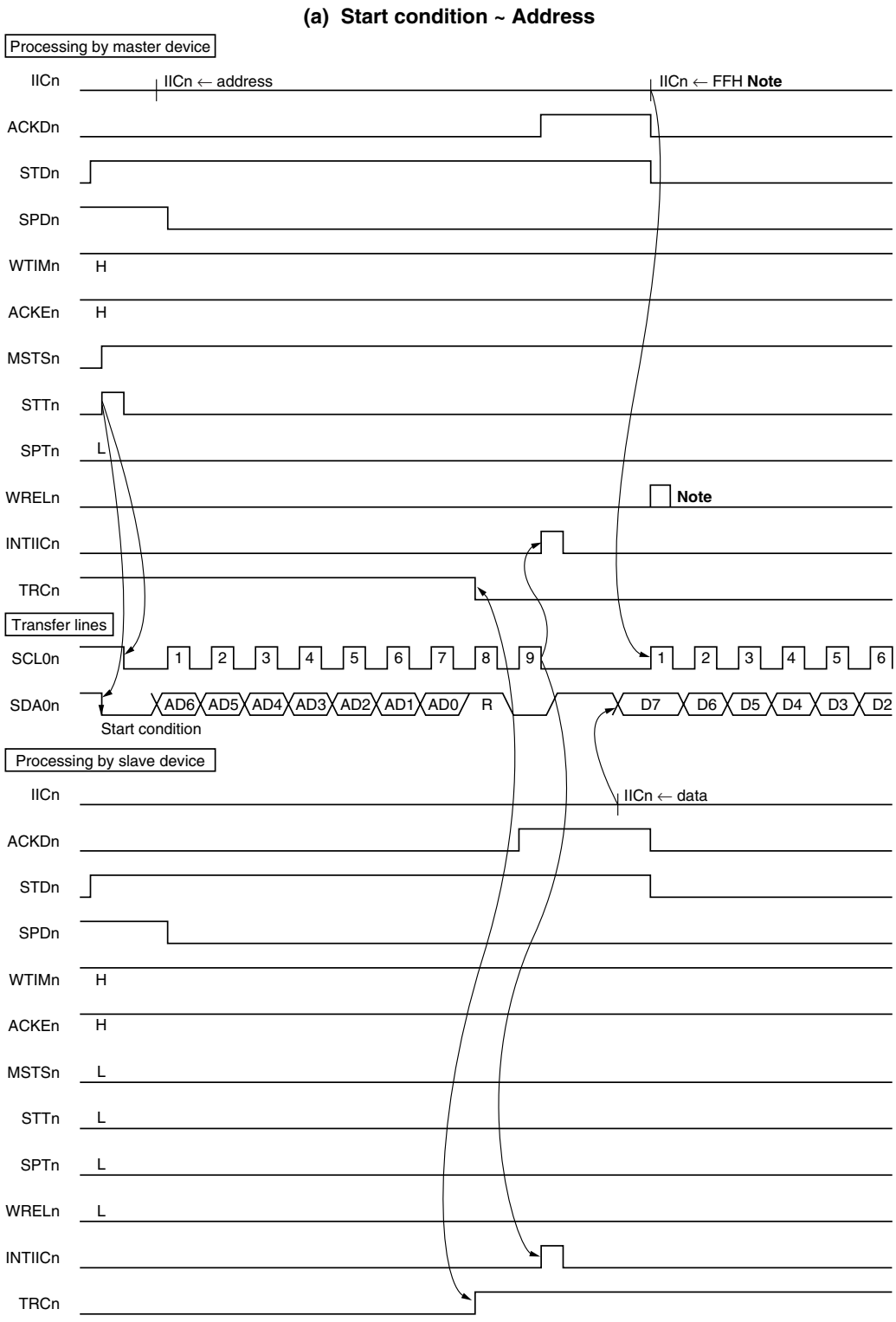
Figure 17-21. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)



Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

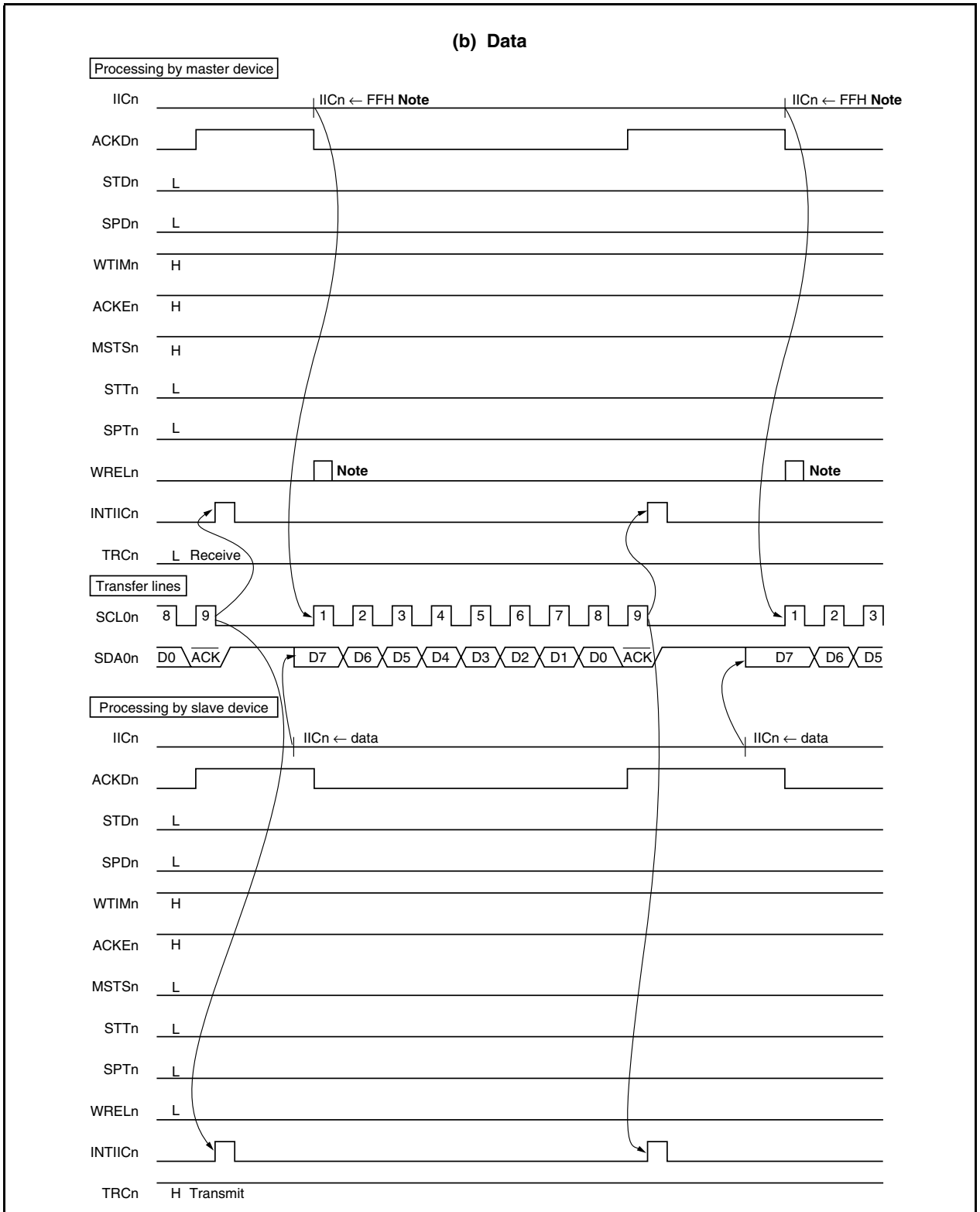
Figure 17-22. Example of Slave to Master Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)



Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

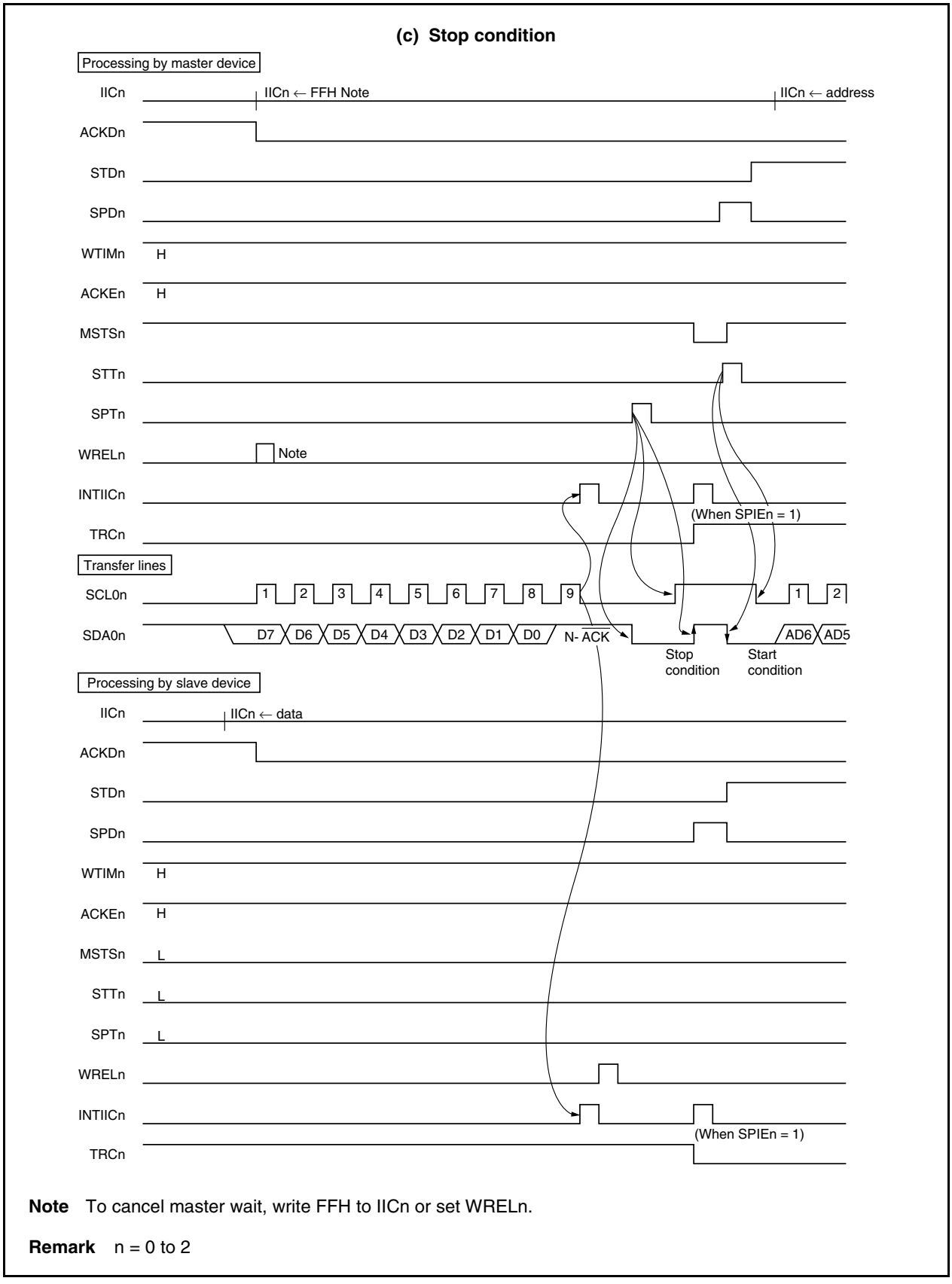
Figure 17-22. Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)



Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

Figure 17-22. Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)



Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0 to 2

CHAPTER 18 IEBus CONTROLLER

IEBus (Inter Equipment Bus) is a small-scale digital data transfer system that transfers data between units. To implement IEBus with the V850ES/SG2, an external IEBus driver and receiver are necessary because they are not provided.

The internal IEBus controller of the V850ES/SG2 is of negative logic.

The following models of the V850ES/SG2 have an on-chip IEBus controller.

- μPD703270, 703270Y, 703271, 703271Y, 703272, 703272Y, 703273, 703273Y, 70F3271, 70F3271Y, 70F3273, 70F3273Y

18.1 Functions

18.1.1 Communication protocol of IEBus

The communication protocol of the IEBus is as follows.

(1) Multi-task mode

All the units connected to the IEBus can transfer data to the other units.

(2) Broadcasting communication function

Communication between one unit and multiple units can be performed as follows.

- Group-unit broadcast communication: Broadcast communication to group units
- All-unit broadcast communication: Broadcast communication to all units.

(3) Effective transfer rate

The effective transfer rate is in mode 1 or mode 2 (the V850ES/SG2 does not support mode 0 for the effective transfer rate).

- Mode 1: Approx. 17 kbps
- Mode 2: Approx. 26 kbps

Caution Different modes must not be mixed on one IEBus.

(4) Communication mode

Data transfer is executed in half-duplex asynchronous communication mode.

(5) Access control: CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

The priority of the IEBus is as follows:

<1> Broadcast communication takes precedence over individual communication (communication from one unit to another).

<2> The lower master address takes precedence.

(6) Communication scale

The communication scale of IEBus is as follows.

- Number of units: 50 MAX.
- Cable length: 150 m MAX. (when twisted pair cable is used)

Caution The communication scale in an actual system differs depending on the characteristics of the cables, etc., constituting the IEBus driver/receiver and IEBus.

18.1.2 Determination of bus mastership (arbitration)

An operation to occupy the bus is performed when a unit connected to the IEBus controls the other units. This operation is called arbitration.

When two or more units simultaneously start transmission, arbitration is used to grant one of the units the permission to occupy the bus.

Because only one unit is granted the bus mastership as a result of arbitration, the priority conditions of the bus are predetermined as follows.

Caution The bus mastership is released if communication is aborted.

(1) Priority by communication type

Broadcast communication (communication from one unit to multiple units) takes precedence over normal communication (communication from one unit to another).

(2) Priority by master address

If the communication type is the same, communication with the lower master address takes precedence.

A master address consists of 12 bits, with unit 000H having the highest priority and unit FFFH having the lowest priority.

18.1.3 Communication mode

The IEBus has three communication modes each having a different transfer rate. The V850ES/SG2 supports communication modes 1 and 2. The transfer rate and the maximum number of transfer bytes in one communication frame in communication modes 1 and 2 are as shown in Table 18-1.

Table 18-1. Transfer Rate and Maximum Number of Transfer Bytes in Each Communication Mode

Communication Mode	Maximum Number of Transfer Bytes (Bytes/Frame)	Effective Transfer Rate (kbps)[Note]
1	32	Approx. 17
2	128	Approx. 26

Note The effective transfer rate when the maximum number of transfer bytes is transmitted.

Select the communication mode for each unit connected to the IEBus before starting communication. If the communication mode of the master unit and that of the partner unit (slave unit) are not the same, communication is not correctly executed.

18.1.4 Communication address

With the IEBus, each unit is assigned a specific 12-bit address. This communication address consists of the following identification numbers.

- Higher 4 bits: Group number (number to identify the group to which each unit belongs)
- Lower 8 bits: Unit number (number to identify each unit in a group)

18.1.5 Broadcast communication

Normally, transmission or reception is performed between the master unit and its partner slave unit on a one-to-one basis. During broadcast communication, however, two or more slave units exist and the master unit executes transmission to these slave units. Because two or more slave units exist, the NACK signal is returned by the communicating slave unit as an acknowledge bit.

Whether broadcast communication or normal communication is to be executed is selected by the broadcast bit (for this bit, refer to **18.1.6 (2) Broadcast bit**).

Broadcast communication is classified into two types: group-unit broadcast communication and all-unit broadcast communication. Group-unit broadcast and all-unit broadcast are identified by the value of the slave address (for the slave address, refer to **18.1.6 (4) Slave address field**).

(1) Group-unit broadcast communication

Broadcast communication is performed to the units in a group identified by the group number indicated by the higher 4 bits of the communication address.

(2) All-unit broadcast communication

Broadcast communication is performed to all the units, regardless of the value of the group number.

18.1.6 Transfer format of IEBus

Figure 18-1 shows the transfer signal format of the IEBus.

Figure 18-1. IEBus Transfer Signal Format

	Header		Master address field		Slave address field			Control field			Telegraph length field			Data field					
Frame format	Start bit	Broad-cast bit	Master address bit	P	Slave address bit	P	A	Control bit	P	A	Tele-graph length bit	P	A	Data bit	P	A	Data bit	P	A

Remarks 1. P: Parity bit
A: Acknowledge ($\overline{\text{ACK}}$/NACK) bit

2. The master unit ignores the acknowledge bit during broadcast communication.

(1) Start bit

The start bit is a signal that informs the other units of the start of data transfer. The unit that is to start data transfer outputs a high-level signal (start bit) from the $\overline{\text{IETX}}$ pin for a specific time, and then starts outputting the broadcast bit.

If another unit has already output its start bit when one unit is to output the start bit, this unit does not output the start bit but waits for completion of output of the start bit by the other unit. When the output of the start bit by the other unit is complete, the unit starts outputting the broadcast bit in synchronization with the completion of the start bit output by the other unit.

The units other than the one that has started communication detect this start bit, and enter the reception status.

(2) Broadcast bit

This bit indicates whether the master selects one slave (individual communication) or multiple slaves (broadcast communication) as the other party of communication.

When the broadcast bit is 0, it indicates broadcast communication; when it is 1, individual communication is indicated. Broadcast communication is classified into two types: group-unit communication and all-unit communication. These communication types are identified by the value of the slave address (for the slave address, refer to **18.1.6 (4) Slave address field**).

Because two or more slave units exist as a partner slave unit of communication in the case of broadcast communication, the NACK signal is returned as an acknowledge bit in each field subsequent to the master address field.

If two or more units start transmitting a communication frame at the same time, broadcast communication takes precedence over individual communication, and wins in arbitration.

If one unit occupies the bus as the master, the value set to the broadcast request flag (BCR.ALLRQ bit) is output.

(3) Master address field

The master address field is output by the master to inform a slave of the master's address.

The configuration of the master address field is as shown in Figure 18-2.

If two or more units start transmitting the broadcast bit at the same time, the master address field makes a judgment of arbitration.

The master address field compares the data it outputs with the data on the bus each time it has output one bit. If the master address output by the master address field is found to be different from the data on the bus as a result of comparison, it is assumed that the master has lost in arbitration. As a result, the master stops transmission and enters the reception status.

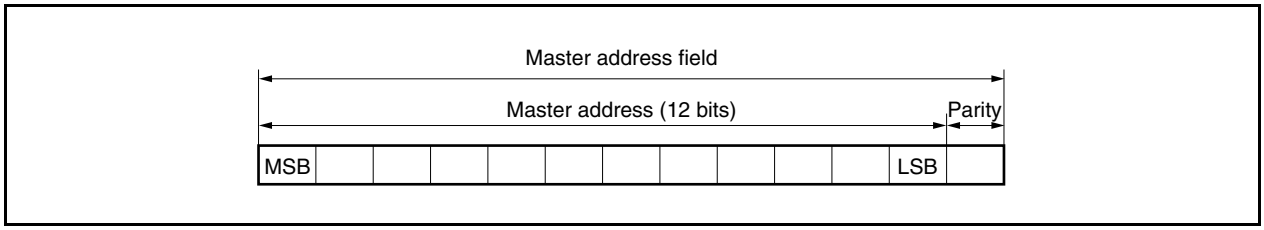
Because the IEBus is configured of wired AND, the unit having the minimum master address of the units participating in arbitration (arbitration masters) wins in arbitration.

After a 12-bit master address has been output, only one unit remains in the transmission status as one master unit.

Next, this master unit outputs a parity bit, determines the master address of other unit, and starts outputting a slave address field.

If one unit occupies the bus as the master, the address set by the UAR register is output.

Figure 18-2. Master Address Field



(4) Slave address field

The master outputs the address of the unit with which it is to communicate.

Figure 18-3 shows the configuration of the slave address field.

A parity bit is output after a 12-bit slave address has been transmitted in order to prevent a wrong slave address from being received by mistake. Next, the master unit detects an $\overline{ACK}$ signal from the slave unit to confirm that the slave unit exists on the bus. When the master has detected the $\overline{ACK}$ signal, it starts outputting the control field. During broadcast communication, however, the master does not confirm the acknowledge bit but starts outputting the control field.

The slave unit outputs the $\overline{ACK}$ signal if its slave address matches and if the slave detects that the parities of both the master address and slave address are even. The slave unit judges that the master address or slave address has not been correctly received and outputs the NACK signal if the parities are odd. At this time, the master unit is in the standby (monitor) status, and communication ends.

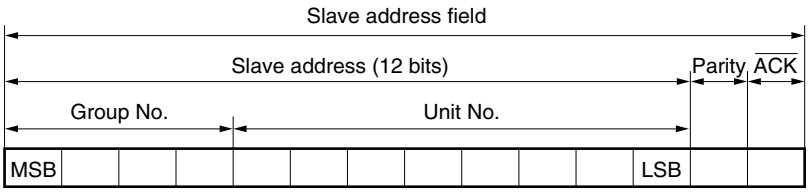
During broadcast communication, the slave address is used to identify group-unit broadcast or all-unit broadcast, as follows:.

If slave address is FFFH: All-unit broadcast communication
If slave address is other than FFFH: Group-unit broadcast communication

Remark The group No. during group-unit broadcasting communication is the value of the higher 4 bits of the slave address.

If one unit occupies the bus as the master, the address set by the SAR register is output.

Figure 18-3. Slave Address Field



(5) Control field

The master outputs the operation it requires the slave to perform, by using this field.

The configuration of the control field is as shown in Figure 18-4.

If the parity following the control bit is even and if the slave unit can execute the function required by the master unit, the slave unit outputs an $\overline{ACK}$ signal and starts outputting the telegraph length field. If the slave unit cannot execute the function required by the master unit even if the parity is even, or if the parity is odd, the slave unit outputs the NACK signal, and returns to the standby (monitor) status.

The master unit starts outputting the telegraph field after detecting the $\overline{ACK}$ signal.

If the master can detect the NACK signal, the master unit enters the standby status, and communication ends. During broadcast communication, however, the master unit does not confirm the acknowledge bit, and starts outputting the telegraph length field.

The contents of the control bits are shown below.

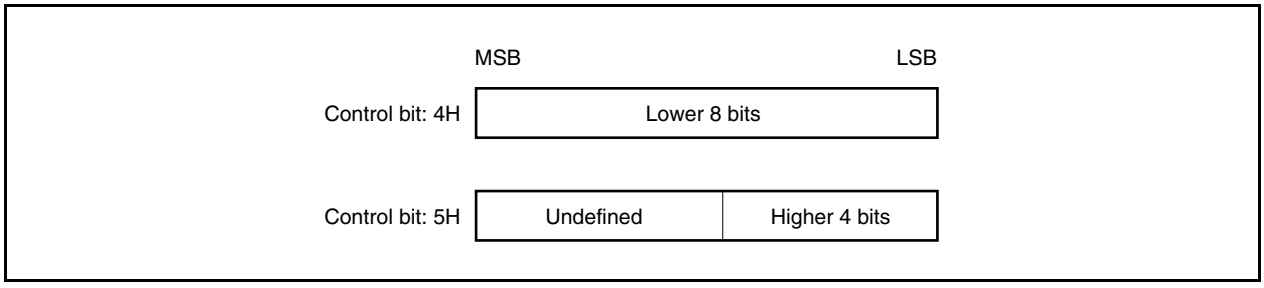
Table 18-2. Contents of Control Bits

Bit 3[Note 1]	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock[Note 2]
0	1	0	0	Read lock address (lower 8 bits)[Note 3]
0	1	0	1	Read lock address (higher 4 bits)[Note 3]
0	1	1	0	Read slave status and unlock[Note 2]
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock[Note 2]
1	0	1	1	Write data and lock[Note 2]
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

Notes 1. The telegraph length bit of the telegraph length field and data transfer direction of the data field change as follows depending on the value of bit 3 (MSB).
If bit 3 is '1': Transfer from master unit to slave unit
If bit 3 is '0': Transfer from slave unit to master unit

2. This is a control bit that specifies locking or unlocking (refer to **18.1.7 (4) Locking and unlocking**).

3. The lock address is transferred in 1-byte (8-bit) units and is configured as follows:



If the control bit received from the master unit is not as shown in Table 18-3, the unit locked by the master unit rejects acknowledging the control bit, and outputs the NACK signal.

Table 18-3. Control Field for Locked Slave Unit

Bit 3[Note 1]	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	1	0	0	Read lock address (lower 8 bits)
0	0	0	1	Read lock address (higher 4 bits)

Moreover, units for which lock is not set by the master unit reject acknowledgment and output a NACK signal when the control data shown in Table 18-4 is acknowledged.

Table 18-4. Control Field for Unlocked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	1	0	0	Lock address read (lower 8 bits)
0	1	0	1	Lock address read (higher 4 bits)

If one unit occupies the bus as the master, the value set to the CDR register is output.

Figure 18-4. Control Field

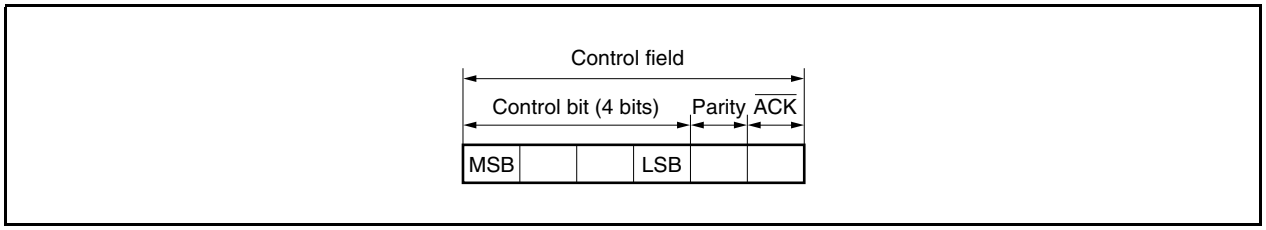


Table 18-5. Acknowledge Signal Output Condition of Control Field

(a) If received control data is AH, BH, EH, or FH

Communication Type (USR.ALLTRANS bit) Individual Communication = 0 Broadcast Communication = 1	Communication Target (USR.SLVRQ bit) Slave Specification = 1 No Specification = 0	Lock Status (USR.LOCK bit) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR register) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (BCR.ENSLVTX bit)	Slave Reception Enable (BCR.ENSLVRX bit)	Received Control Data			
						AH	BH	EH	FH
0	1	0	don't care	don't care	1	○			
		1	1						
Other than above						×			

(b) If received control data is 0H, 3H, 4H, 5H, 6H, or 7H

Communication Type (USR.ALLTRANS bit) Individual Communication = 0 Broadcast Communication = 1	Communication Target (USR.SLVRQ bit) Slave Specification = 1 No Specification = 0	Lock Status (USR.LOCK bit) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR register) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (BCR.ENSLVTX bit)	Slave Reception Enable (BCR.ENSLVRX bit)	Received Control Data					
						0H	3H	4H	5H	6H	7H
0	1	0	don't care	0	don't care	○	×	×	×	○	×
				1		○	○	×	×	○	○
		1	0	don't care		○	×	○	○	×	×
			1	0		○	×	○	○	○	×
				1		○	○	○	○	○	○
Other than above						×					

Caution If the received control data is other than the data shown in Table 18-5, × (NACK signal is returned) is unconditionally assumed.

Remarks 1. ○: $\overline{\text{ACK}}$ signal is returned.

×: NACK signal is returned.

(6) Telegraph length field

This field is output by the transmission side to inform the reception side of the number of bytes of the transmit data.

The configuration of the telegraph length field is as shown in Figure 18-5.

Table 18-6 shows the relationship between the telegraph length bit and the number of transmit data.

Figure 18-5. Telegraph Length Field

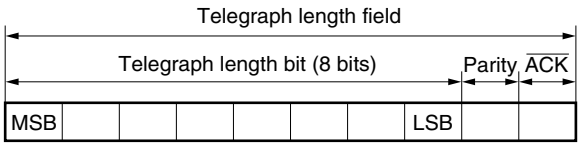


Table 18-6. Contents of Telegraph Length Bit

Telegraph Length Bit (Hex)	Number of Transmit Data Bytes
01H	1 byte
02H	2 bytes
FFH	255 bytes
00H	256 bytes

The operation of the telegraph length field differs depending on whether the master transmits data (when control bit 3 is 1) or receives data (when control bit 3 is 0).

(a) When master transmits data

The telegraph length bit and parity bit are output by the master unit and the synchronization signals of bits are output by the master unit. When the slave unit detects that the parity is even, it outputs the $\overline{\text{ACK}}$ signal, and starts outputting the data field. During broadcast communication, however, the slave unit outputs the NACK signal.

If the parity is odd, the slave unit judges that the telegraph length bit has not been correctly received, outputs the NACK signal, and returns to the standby (monitor) status. At this time, the master unit also returns to the standby status, and communication ends.

(b) When master receives data

The telegraph length bit and parity bit are output by the slave unit and the synchronization signals of bits are output by the master unit. If the master unit detects that the parity bit is even, it outputs the $\overline{\text{ACK}}$ signal.

If the parity bit is odd, the master unit judges that the telegraph length bit has not been correctly received, outputs the NACK signal, and returns to the standby status. At this time, the slave unit also returns to the standby status, and communication ends.

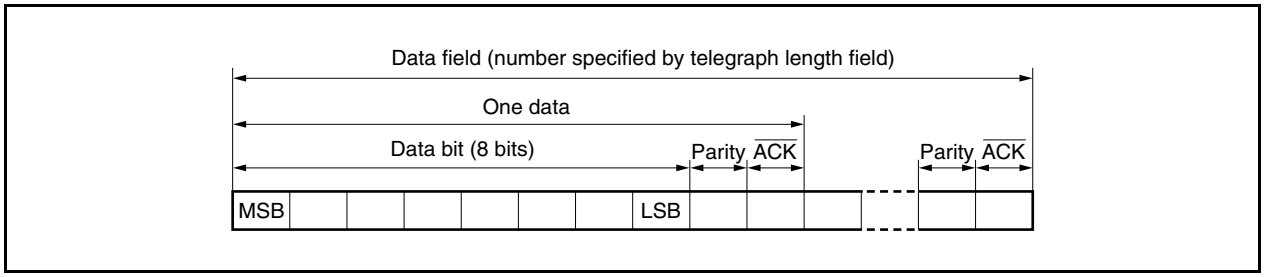
(7) Data field

This is data output by the transmission side.

The master unit transmits or receives data to or from a slave unit by using the data field.

The configuration of the data field is as shown below.

Figure 18-6. Data Field



Following the data bit, the parity bit and acknowledge bit are respectively output by the master unit and slave unit.

Use broadcast communication only for when the master unit transmits data. At this time, the acknowledge bit is ignored.

The operation differs as follows depending on whether the master transmits or receives data.

(a) When master transmits data

When the master units writes data to a slave unit, the master unit transmits the data bit and parity bit to the slave unit. If the parity is even and the receive data is not stored in the DR register when the slave unit has received the data bit and parity bit, the slave unit outputs an $\overline{ACK}$ signal. If the parity is odd or if the receive data is stored in the DR register, the slave unit rejects receiving the data, and outputs the NACK signal.

If the slave unit outputs the NACK signal, the master unit transmits the same data again. This operation continues until the master detects the $\overline{ACK}$ signal from the slave unit, or the data exceeds the maximum number of transmit bytes.

If there is more data and the maximum number of transmit bytes is not exceeded when the parity is even and when the slave unit outputs the $\overline{ACK}$ signal, the master unit transmits the next data.

During broadcast communication, the slave unit outputs the NACK signal, and the master unit transfers 1 byte of data at a time. If the parity is odd or the DR register is storing receive data after the slave unit has received the data bit and parity bit during broadcast communication, the slave unit judges that reception has not been performed correctly, and stops reception.

(b) When master receives data

When the master unit reads data from a slave unit, the master unit outputs a sync signal corresponding to all the read bits.

The slave unit outputs the contents of the data and parity bits to the bus in response to the sync signal from the master unit.

The master unit reads the data and parity bits output by the slave unit, and checks the parity.

If the parity is odd, or if the DR register is storing a receive data, the master unit rejects accepting the data, and outputs the NACK signal. If the maximum number of transmit bytes is within the value that can be transmitted in one communication frame, the master unit repeats reading the same data.

If the parity is even and the DR register is not storing a receive data, the master unit accepts the data and outputs the $\overline{\text{ACK}}$ signal. If the maximum number of transmit bytes is within the value that can be transmitted in one frame, the master unit reads the next data.

Caution Do not operate master reception in broadcast communication, because the slave unit cannot be defined and data transfer cannot be performed correctly.

(8) Parity bit

The parity bit is used to check to see if the transmit data has no error.

The parity bit is appended to each data of the master address, slave address, control, telegraph length, and data bits.

The parity is an even parity. If the number of bits in data that are '1' is odd, the parity bit is '1'. If the number of bits in the data that are '1' is even, the parity bit is '0'.

(9) Acknowledge bit

During normal communication (communication from one unit to another), an acknowledge bit is appended to the following locations to check if the data has been correctly received.

- End of slave address field
- End of control field
- End of telegraph length field
- End of data field

The definition of the acknowledge bit is as follows.

- 0: Indicates that the transmit data is recognized ($\overline{\text{ACK}}$ signal).
- 1: Indicates that the transmit data is not recognized (NACK signal).

During broadcast communication, however, the contents of the acknowledge bit are ignored.

(a) Last acknowledge bit of slave field

The last acknowledge bit of the slave field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the master address bit or slave address bit is incorrect
- If a timing error (error in bit format) occurs
- If a slave unit does not exist

(b) Last acknowledge bit of control field

The last acknowledge bit of the control field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the control bit is incorrect
- If control bit 3 is '1' (write operation) when the slave reception enable flag (BCR.ENSLVRX bit) is not set (1)[Note]
- If the control bit indicates reading of data (3H or 7H) when the slave transmission enable flag (BCR.ENSLVTX bit) is not set (1)[Note]
- If a unit other than that has set locking requests 3H, 6H, 7H, AH, BH, EH, or FH of the control bit when locking is set
- If the control bit indicates reading of lock addresses (4H, 5H) even when locking is not set
- If a timing error occurs
- If the control bit is undefined

Note Refer to **18.3 (1) IEBus control register (BCR)**.

Cautions 1. The $\overline{\text{ACK}}$ signal is always returned when the control data of the slave status request is received, even if the ENSLVTX bit = 0.

2. The NACK signal is returned by the acknowledge bit in the control field when the control data for data/command writing is received, even if the ENSLVRX bit = 0.
Slave reception can be disabled (communication stopped) by ENSLVRX bit only in the case of individual communication. In the case of broadcast communication, communication is maintained and the data request interrupt request signal (INTIE1) or IEBus end interrupt request signal (INTIE2) is generated.

(c) Last acknowledge bit of telegraph length field

The last acknowledge bit of the telegraph length field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the telegraph length bit is incorrect
- If a timing error occurs

(d) Last acknowledge bit of data field

The last acknowledge bit of the data field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the data bit is incorrect[Note]
- If a timing error occurs after the preceding acknowledge bit has been transmitted
- If the receive data is stored in the DR register and no more data can be received[Note]

Note In this case, when the communication executed is individual communication, if the maximum number of transmit bytes is within the value that can be transmitted in one frame, the transmission side executes transmission of that data field again. For broadcast communication, the transmission side does not execute transmission again, a communication error occurs on the reception side and reception stops.

18.1.7 Transfer data

(1) Slave status

The master unit can learn why the slave unit did not return the $\overline{ACK}$ signal by reading the slave status.

The slave status is determined according to the result of the last communication the slave unit has executed.

All the slave units can supply information on the slave status.

The configuration of the slave status is shown below.

Figure 18-7. Bit Configuration of Slave Status

MSB LSB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 0[Note 1]	Meaning
0	Transmit data is not written in DR register
1	Transmit data is written in DR register

Bit 1[Note 2]	Meaning
0	Receive data is not stored in DR register
1	Receive data is stored in DR register

Bit 2	Meaning
0	Unit is not locked
1	Unit is locked

Bit 3	Meaning
0	Fixed to 0

Bit 4[Note 3]	Meaning
0	Slave transmission is stopped
1	Slave transmission is ready

Bit 5	Meaning
0	Fixed to 0

Bit 7	Bit 6	Meaning	
0	0	Mode 0	Indicates the highest mode supported by the unit[Note 4].
0	1	Mode 1	
1	0	Mode 2	
1	1	Not used	

Notes 1. After reset: Bit 0 is set to 1.

2. The receive buffer size is 1 byte.

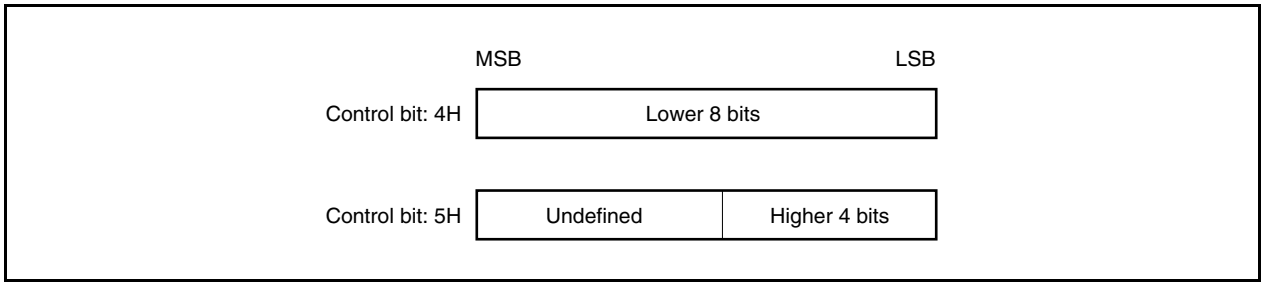
3. When the V850ES/SG2 serves as a slave unit, this bit corresponds to the status indicated by BCR.ENSLVTX bit.

4. Bits 7 and 6 are fixed to “10” because the V850ES/SG2 can support modes 1 and 2.

(2) Lock address

When the lock address is read (control bit: 4H or 5H), the address (12 bits) of the master unit that has issued the lock instruction is configured in 1-byte units as shown below and read.

Figure 18-8. Configuration of Lock Address



(3) Data

If the control bit indicates reading of data (3H or 7H), the data in the data buffer of the slave unit is read by the master unit.

If the control bit indicates writing of data (BH or FH), the data received by the slave unit is processed according to the operation rule of that slave unit.

(4) Locking and unlocking

The lock function is used when a message is transferred in two or more communication frames.

The unit that is locked does not receive data from units other than the one that has locked the unit (does not receive broadcast communication).

A unit is locked or unlocked as follows.

(a) Locking

If the communication frame is completed without succeeding to transmit or receive data of the number of bytes specified by the telegraph length bit after the telegraph length field has been transmitted or received ($\overline{ACK}$ = 0) by the control bit that specifies locking (3H, AH, or BH), the slave unit is locked by the master unit. At this time, the bit (bit 2) in the byte indicating the slave status is set to '1'.

(b) Unlocking

After transmitting or receiving data of the number of data bytes specified by the telegraph length bit in one communication frame by the control bit that has specified locking (3H, AH, or BH), or the control bit that has specified unlocking (6H), the slave unit is unlocked by the master unit. At this time, the bit related to locking (bit 2) in the byte indicating the slave status is reset to '0'.

Locking or unlocking is not performed during broadcast communication.

Locking and unlocking conditions are shown below.

Table 18-7. Lock Setting Conditions

Control Data	Broadcast Communication		Individual Communication	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Cannot be locked	Lock set
AH, BH	Cannot be locked	Cannot be locked	Cannot be locked	Lock set
0H, 4H, 5H, EH, FH	Cannot be locked	Cannot be locked	Cannot be locked	Cannot be locked

Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated up to the maximum number of transfer bytes with being output.

Table 18-8. Unlock Release Conditions (While Locked)

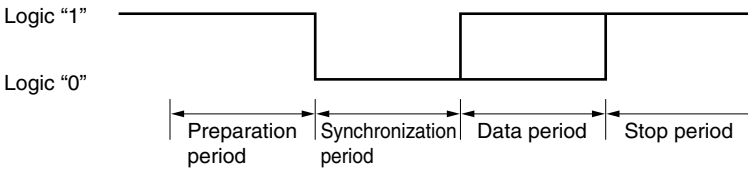
Control Data	Broadcast Communication from Lock Request Unit		Individual Communication from Lock Request Unit	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Unlocked	Remains locked
AH, BH	Unlocked	Unlocked	Unlocked	Remains locked
0H, 4H, 5H, EH, FH	Remains locked	Remains locked	Remains locked	Remains locked

Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated up to the maximum number of transfer bytes with being output.

18.1.8 Bit format

The format of the bits constituting the communication frame of the IEBus is shown below.

Figure 18-9. Bit Format of IEBus



Preparation period: First low-level (logic “1”) period
Synchronization period: Next high-level (logic “0”) period
Data period: Period indicating value of bit
Stop period: Last low-level (logic “1”) period

The synchronization period and data period are almost equal to each other in length.

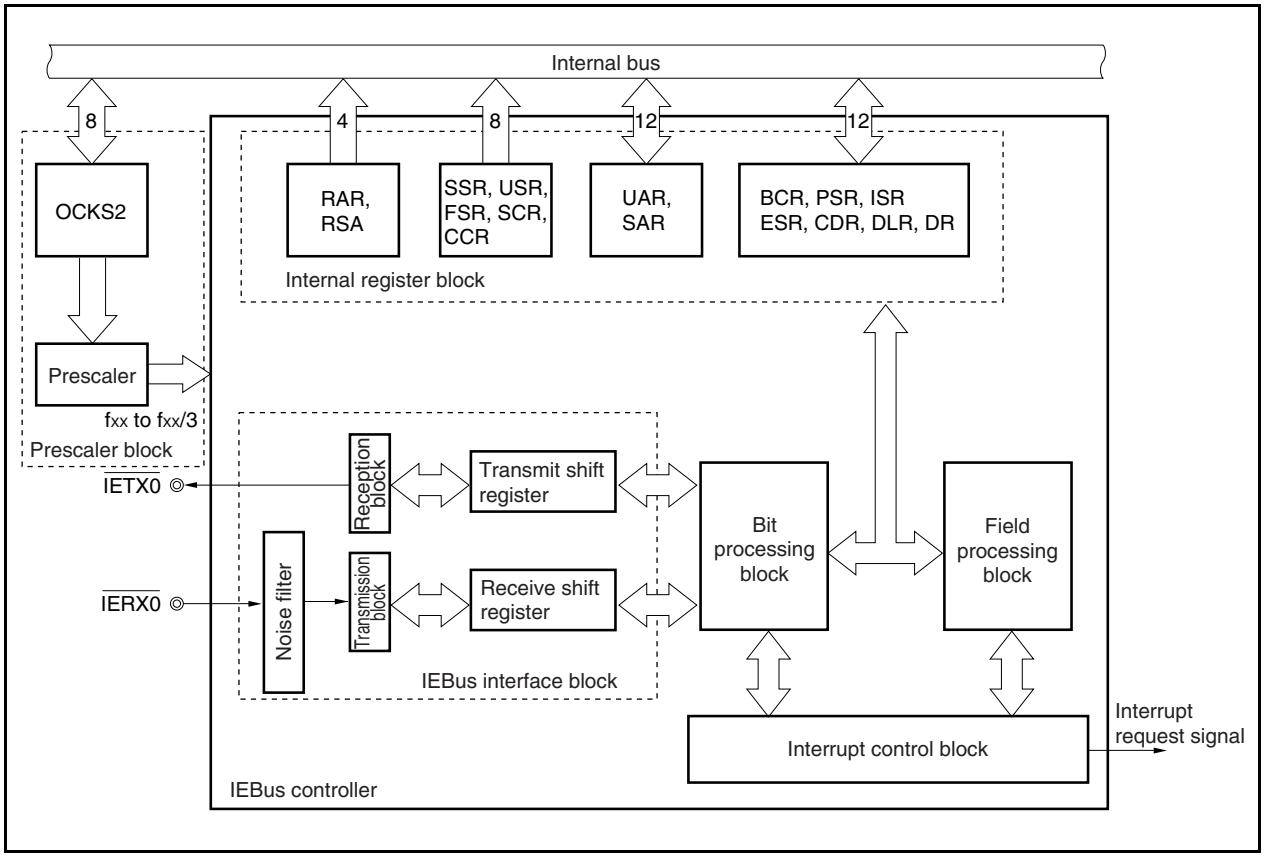
The IEBus synchronizes each bit. The specifications on the time of the entire bit and the time related to the period allocated to that bit differ depending on the type of the transmit bit, or whether the unit is the master unit or a slave unit.

The master and slave units monitor whether each period (preparation period, synchronization period, data period, and stop period) is output for specified time while they are in communication. If a period is not output for the specified time, the master and slave units report a timing error, immediately terminate communication and enter the standby status.

18.2 Configuration

The block diagram of the IEBus controller is shown below.

Figure 18-10. IEBus Controller Block Diagram



(1) Hardware configuration and functions

IEBus mainly consists of the following six internal blocks.

- Interrupt control block
- Internal registers
- Bit processing block
- Field processing block
- IEBus interface block
- Prescaler block

(a) Interrupt control block

This control block transfers interrupt request signals from the IEBus controller to the CPU.

(b) Internal registers

These registers set data to the control registers and fields that control IEBus (for the internal registers, refer to **18.3 Control Registers**).

(c) Bit processing block

This block generates and breaks down bit timing, and mainly consists of a bit sequence ROM, 8-bit preset timer, and comparator.

(d) Field processing block

This block generates each field in the communication frame, and mainly consists of a field sequence ROM, 4-bit down counter, and comparator.

(e) IEBus interface block

This is the interface block for an external driver/receiver, and mainly consists of a noise filter, shift register, and transmission/reception block (collision detector, parity detector, parity generator, and $\overline{\text{ACK}}$/NACK generator).

(f) Prescaler block

This block selects the clock to be supplied to the IEBus controller.

18.3 Control Registers

The registers that control the IEBus controller are shown below.

Table 18-9. Control Registers of IEBus Controller

Address	Function Register Name	Symbol	R/W	Bit Unit for Manipulation			After Reset
				1	8	16	
FFFFF348H	IEBus clock select register	OCKS2	R/W		√		00H
FFFFF360H	IEBus control register	BCR		√	√		
FFFFF361H	IEBus power save register	PSR		√	√		
FFFFF362H	IEBus slave status register	SSR	R		√		81H
FFFFF363H	IEBus unit status register	USR			√		00H
FFFFF364H	IEBus interrupt status register	ISR	R/W	√	√		
FFFFF365H	IEBus error status register	ESR		√	√		
FFFFF366H	IEBus unit address register	UAR				√	0000H
FFFFF368H	IEBus slave address register	SAR				√	
FFFFF36AH	IEBus partner address register	PAR	R			√	
FFFFF36CH	IEBus receive slave address register	RSA				√	
FFFFF36EH	IEBus control data register	CDR	R/W		√		00H
FFFFF36FH	IEBus telegraph length register	DLR			√		01H
FFFFF370H	IEBus data register	DR			√		00H
FFFFF371H	IEBus field status register	FSR	R		√		
FFFFF372H	IEBus success count register	SCR			√		01H
FFFFF373H	IEBus communication count register	CCR			√		20H

(1) IEBus control register (BCR)

The BCR register is an 8-bit register that controls the operations of the IEBus controller.

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

Reset input clears this register to 00H.

After reset: 00H RW Address: FFFFF360H

	<7>	<6>	<5>	<4>	<3>	2	1	0
BCR	ENIEBUS	MSTRQ	ALLRQ	ENSLVTX	ENSLVRX	0	0	0

ENIEBUS	Communication enable flag
0	IEBus unit stopped
1	IEBus unit active

MSTRQ	Master request flag
0	IEBus unit not requested as master
1	IEBus unit requested as master

ALLRQ	Broadcast request flag
0	Individual communication requested
1	Broadcast communication requested

ENSLVTX	Slave transmission enable flag
0	Slave transmission disabled
1	Slave transmission enabled

ENSLVRX	Slave reception enable flag
0	Slave reception disabled
1	Slave reception enabled

Cautions 1. While IEBus is operating as the master, writing to the BCR register (including bit manipulation instructions) is disabled until either the end of that communication or frame, or until communication is stopped by the occurrence of an arbitration-loss communication error. Master requests cannot therefore be multiplexed. However, the case when communication has been forcibly stopped (ENIEBUS flag = 0) is not problem.

2. If a bit manipulation instruction for the BCR register conflicts with a hardware reset of the MSTRQ bit, the BCR register may not operate normally. The following countermeasures are recommended in this case.

- **Because the hardware reset is instigated in the acknowledgement period of the slave address field, be sure to observe Caution 1 of (b) Master request flag (MSTRQ) below.**
- **Be sure to observe the caution above regarding writing to the BCR register.**

(a) Communication enable flag (ENIEBUS)...Bit 7

<Set/clear conditions>

Set: By software

Clear: By software

The IEBus controller participates in communication differently depending on the timing of setting the ENIEBUS bit (1), as follows.

Table 18-10. Timing of Setting ENIEBUS Bit and Participation in Communication

Timing of Setting ENIEBUS Bit	How IEBus Controller Participates in Communication
If communication is not performed on IEBus	Participates in communication from the next frame or starts communication.
If other bus master is communicating start bit while communication is in progress on IEBus	Participates in communication from that frame if the start bit is detected. If the start bit is not detected, participates in communication from the next frame.
If communication is in progress on IEBus after start bit from other bus master is detected	Participates in communication from the next frame.

If the ENIEBUS bit is cleared (0), communication is immediately stopped even while it is in progress, and the internal flags and registers are reset, with some exceptions. The registers that are not reset by the ENIEBUS bit are listed in the table below.

The IEBus controller does not respond even if another unit starts communication when the ENIEBUS bit = 0.

Table 18-11. Registers That Are Not Reset by ENIEBUS Bit

Registers Not Reset by ENIEBUS Bit	Remark
UAR	Not reset
SAR	Not reset
CDR	Data written from CPU is not reset but data received during communication is reset.
DLR	Data written from CPU is not reset but data received during communication is reset.
DR	Data written from CPU is not reset but data received during communication is reset.

Caution Before setting the ENIEBUS bit (1), the following registers must be set depending on the mode of communication to be started.

Table 18-12. Registers That Must Be Set Before Each Communication

Mode of Communication	Registers That Must Be Set in Advance
Master transmission	UAR, SAR, CDR, DLR, DR (first 1-byte data)
Master reception	UAR, SAR, CDR
Slave transmission[Note]	UAR, DLR, DR (first 1-byte data)[Note]
Slave reception	UAR

Note When starting slave transmission, information such as the value to be set to the DLR register and which data is to be returned (value to be set to the DR register) must be assigned in advance.

(b) Master request flag (MSTRQ)...Bit 6

<Set/clear conditions>

Set: By software

Clear: Cleared (0) by hardware when master communication is started and immediately before the start interrupt of the master occurs.

Cleared (0) by hardware before a communication error occurs.

When the ENIEBUS bit is cleared.

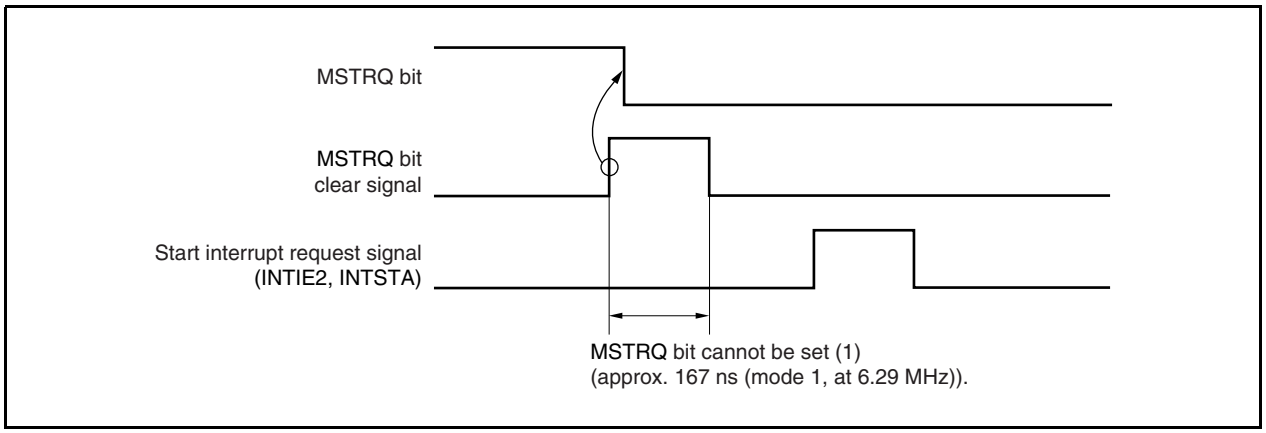
When the MSTRQ bit is set (1), the IEBus controller starts communication on IEBus as the master.

If communication is in progress on IEBus (if the start bit cannot be detected while the start bit is being communicated or if communication is in progress after the start bit has been detected), however, the controller waits until the current frame ends (holds the master request pending), outputs the start bit after the frame has ended, and starts communication as the master.

Cautions 1. If the IEBus controller has lost in arbitration, issue the master request again by software.

In doing so, set (1) the MSTRQ bit at a timing other than that illustrated below.

Figure 18-11. Timing at Which MSTRQ Bit Cannot Be Set



2. When a master request has been sent and bus mastership acquired, do not set the MSTRQ, ENSLVTX, or ENSLVRX bit until the end of communication (i.e. the communication end flag (ISR.ENDTRNS bit) or frame end flag (ISR.ENDFRAM bit) is set (1)) as setting these flags disables interrupt request signal generation. However, these flags can be set if communication has been aborted.

(c) Broadcast request flag (ALLRQ)...Bit 5

<Set/clear conditions>

Set: By software

Clear: By software

Caution When requesting broadcast communication, always set (1) the ALLRQ bit, then the MSTRQ bit.

(d) Slave transmission enable flag (ENSLVTX)...Bit 4

<Set/clear conditions>

Set: By software

Clear: By software

Cautions 1. The ENSLVTX bit must be set before the parity bit in the control field is received.

2. Clear the ENSLVTX bit (0) before setting the MSTRQ bit (1) when making a master request. This is to avoid transmission of the data of the DR register that tries master transmission if the controller loses in arbitration after master operation and if slave transmission is requested by the master.

3. When returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

4. If control data (3H or 7H) for data/command writing is received when the ENSLVTX bit = 0, the NACK signal is returned by the acknowledge bit in the control field.

5. The status interrupt request signals (INTIE2, INTSTA) will be generated and communication continued when the control data of a slave status request is returned, even if the ENSLVTX bit = 0.

(e) Slave reception enable flag (ENSLVRX)...Bit 3

<Set/clear conditions>

Set: By software

Clear: By software

Cautions 1. The ENSLVRX bit must be set before the parity bit in the control field is received.

2. While the CPU is busy with other processing, slave reception can be prevented by clearing the ENSLVRX bit (0). During individual communication, the NACK signal is returned in the control field and communication is completed. During broadcast communication, communication cannot be completed because the acknowledge bit is ignored. However, the IEBus controller does not respond to the broadcast communication and does not generate an interrupt request signal.

3. When returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

(2) IEBus power save register (PSR)

The PSR register is an 8-bit register that controls the internal clock and communication mode of the IEBus controller.

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

Reset input clears this register to 00H.

After reset: 00H RW Address: FFFFF361H

	<7>	<6>	5	4	3	2	1	0
PSR	ENCLK	IEMODE	0	0	0	0	0	0

ENCLK	Internal clock operation enable flag
0	Stop internal clock of IEBus controller
1	Enable internal clock of IEBus controller

IEMODE	IEBus communication mode setting flag
0	Set communication mode 1
1	Set communication mode 2

Cautions 1. Do not set the PSR register while communication is enabled (BCR.ENIEBUS bit = 1).

2. Be sure to clear bits 5 to 0 to 0.

(3) IEBus slave status register (SSR)

The SSR register is an 8-bit register that indicates the communication status of the slave unit. After receiving a slave status transmission request from the master, read this register by software, and write a slave status to the DR register to transmit the slave status. At this time, the telegraph length is automatically set to “01H”, so setting of the DLR register is not required (because it is preset by hardware).

Bits 6 and 7 indicate the highest mode supported by the unit, and are fixed to “10” (mode 2).

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

Reset input sets this register to 81H.

After reset: 81H R Address: FFFFF362H

	7	6	5	<4>	3	<2>	<1>	<0>
SSR	1	0	0	STATSLV	0	STATLOCK	STATRX	STATTX

STATSLV	Slave transmission status flag
0	Slave transmission stops
1	Slave transmission enabled

STATLOCK	Lock status flag
0	Unlock status
1	Lock status

STATRX	DR register receive status
0	Receive data not stored in DR register
1	Receive data stored in DR register

STATTX	DR register transmit status
0	Transmit data not stored in DR register
1	Transmit data stored in DR register

(a) Slave transmission status flag (STATSLV)...Bit 4

Reflects the contents of the slave transmission enable flag (BCR.ENSLVTX bit).

(b) Lock status flag (STATLOCK)...Bit 2

Reflects the contents of the lock flag (USR.LOCK bit).

(c) DR register reception status (STATRX)...Bit 1

This flag indicates the DR register reception state.

(d) DR register transmission status (STATTX)...Bit 0

This flag indicates the DR register transmission state.

(4) IEBus unit status register (USR)

The USR register is an 8-bit register that indicates the IEBus unit status.

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

Reset input clears this register to 00H.

After reset: 00H R Address: FFFFF363H

	7	6	5	4	3	2	1	0
USR	0	SLVRQ	ARBIT	ALLTRNS	ACK	LOCK	0	0

SLVRQ	Slave request flag
0	No request from master to slave
1	Request from master to slave

ARBIT	Arbitration result flag
0	Arbitration win
1	Arbitration loss

ALLTRNS	Broadcast communication flag
0	Individual communication status
1	Broadcast communication status

ACK	Acknowledge transmission flag
0	NACK signal transmitted
1	$\overline{\text{ACK}}$ signal transmitted

LOCK	Lock status flag
0	Unit unlocked
1	Unit locked

(a) Slave request flag (SLVRQ)...Bit 6

A flag indicating whether there has been a slave request from the master.

<Set/clear conditions>

Set: When the unit is requested as a slave (if the condition in **Table 18-13 Slave Request Condition (SLVRQ Bit Setting Condition**) is satisfied), this flag is set (1) by hardware when the acknowledge period of the slave address field starts.

Clear: This flag is cleared (0) by hardware when the unit is not requested as a slave (if the condition in **Table 18-13 Slave Request Condition (SLVRQ Bit Setting Condition**) is not satisfied). The reset timing is the same as the set timing. If the unit is requested as a slave immediately after communication has been correctly received (when the SLVRQ bit = 1), and if a parity error occurs in the slave address field for that communication, the flag is not cleared.

Table 18-13. Slave Request Condition (SLVRQ Bit Setting Condition)

Status of Unit	Received Master Address	Communication Mode	Received Slave Address
Not locked	don't care	Individual	UAR register matching
		Broadcast	Group matching
			FFFH
Locked	Locked master matching	Individual	UAR register matching
		Broadcast	Group matching
			FFFH

Caution If a unit other than the locked master communicates with the unit while the unit is locked, the SLVRQ bit is not set but the $\overline{\text{ACK}}$ signal is returned to the slave address field. This is because communication must be continued, even if a unit other than the locked master returns the signal, if the control data is a slave status request.

(b) Arbitration result flag (ARBIT)...Bit 5

A flag that indicates the result of arbitration.

<Set/clear conditions>

Set: This flag is set (1) when the data output by the IEBus unit during the arbitration period does not match the bus line data.

Clear: This flag is cleared (0) by the start bit timing.

Cautions 1. The timing at which the arbitration result flag (ARBIT bit) is cleared differs depending on whether the unit outputs a start bit.

- **If start bit is output: The flag is cleared at the output start timing.**
- **If start bit is not output: The flag is cleared at the detection timing of the start bit (approx. 160 μs (mode 1, at 6.29 MHz) after output)**

2. The flag is cleared (0) at the detection timing of the start bit if the other unit outputs the start bit earlier and the unit does not output the start bit after the master request.

(c) Broadcast communication flag (ALLTRNS)...Bit 4

Flag indicating whether the unit is performing broadcast communication. The contents of the flag are updated in the broadcast field of each frame.

Except for initialization (reset) by system reset, the set/clear conditions vary depending on the receive data of the broadcast field bit.

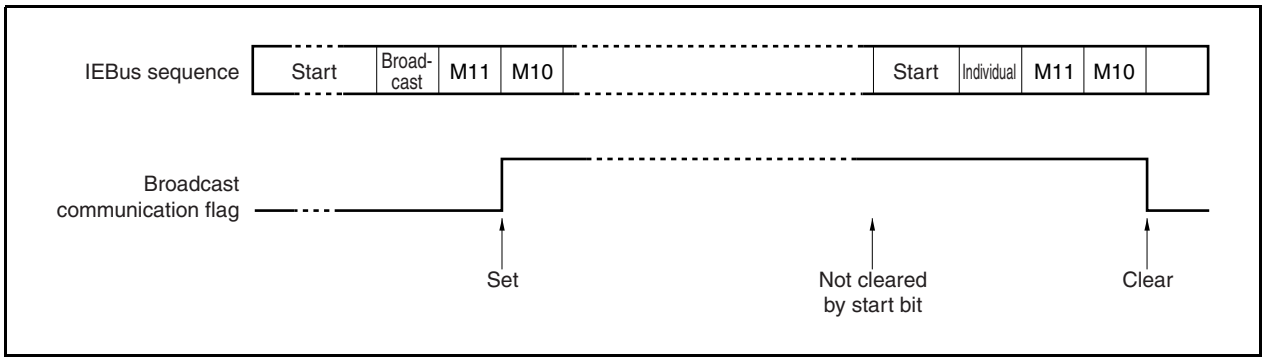
<Set/clear conditions>

Set: When “broadcast” is received by the broadcast field

Clear: When “individual” is received by the broadcast field, or upon the input of a system reset.

Caution The broadcast flag is updated regardless of whether IEBus is the communication target or not.

Figure 18-12. Example of Broadcast Communication Flag Operation



(d) Acknowledge transmission flag (ACK)...Bit 3

A flag that indicates whether the $\overline{\text{ACK}}$ signal has been transmitted in the acknowledge bit period of the acknowledge bit field when IEBus is the receiving unit. The contents of the flag are updated in the acknowledge bit period of each frame. However, if the internal circuit is initialized by the occurrence of a parity error, etc., the contents are not updated in the acknowledge bit period of that field.

(e) Lock status flag (LOCK)...Bit 2

A flag that indicates whether the unit is locked.

<Set/clear conditions>

Set: This flag is set (1) when the communication end flag (ISR.ENDTRNS bit) goes low and the frame end flag (ISR.ENDFRAM bit) goes high after receipt of a lock specification (3H, 6H, AH, BH) in the control field.

Clear: When the communication enable flag (BCR.ENIEBUS bit) is cleared (0).

When the communication end flag (ENDTRNS bit) is set (1) after receipt of a lock release (3H, 6H, AH, BH) in the control field.

Caution Lock specification/release is not possible in broadcast communication. In the lock status, individual communication from a unit other than the one that requests locking is not acknowledged. However, even communication from a unit other than the one that requests locking is acknowledged as long as the communication is a slave status request.

(5) IEBus interrupt status register (ISR)

The ISR register indicates the interrupt source when IEBus issues an interrupt request signal. This register is read to generate an interrupt request signal, after which the specified interrupt processing is carried out.

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

Reset input clears this register to 00H.

After reset: 00H R/W[Note 1] Address: FFFFF364H

	7	<6>	5	4	3	2	1	0
ISR	0	IEERR	STARTF	STATUSF	ENDTRNS	ENDFRAM	0	0

IEERR	Communication error flag (during communication)
0	No communication error
1	Communication error

STARTF	Start interrupt flag
0	Start interrupt request signal did not occur
1	Start interrupt request signal occurred

STATUSF	Status transmission flag (slave)
0	No slave status/lock address (higher 4 bits, lower 8 bits) transmission request
1	Slave status/lock address (higher 4 bits, lower 8 bits) transmission request

ENDTRNS	Communication end flag
0	Communication does not end after the number of bytes set in the telegraph length field have been transferred
1	Communication ends after the number of bytes set in the telegraph length field have been transferred

ENDFRAM	Frame end flag
0	The frame (transfer of the maximum number of bytes) does not end
1	The frame (transfer of the maximum number of bytes) ends

Notes 1. Only the IEERR bit can be written, and only to 0 (i.e., the IEERR bit can only be cleared). The IEERR bit is not set (1) even if 1 is written to it.

2. Mode 1: 32 bytes
Mode 2: 128 bytes

(a) Communication error flag (IEERR)...Bit 6

A flag that indicates a communication error has occurred. When a communication error occurs, the INTIE2 and INTERR interrupt request signals are generated.

<Set/clear conditions>

Set: The flag is set (1) if a timing error, parity error (except in the data field), NACK reception error (except in the data field), underrun error, overrun error (that occurs during broadcast communication reception), or write error occurs.

Clear: By software

(b) Start interrupt flag (STARTF)...Bit 5

A flag that indicates the start interrupt. When a start interrupt occurs, the INTIE2 and INTSTA interrupt request signals are generated.

<Set/clear conditions>

Set: This flag is set (1) in the slave address field, upon a master request.
When IEBus is a slave unit, this flag is set (1) upon a request from the master (only if it was a slave request in the locked state from the unit requesting a lock).

Clear: This flag is cleared (0) if the status transmission interrupt, communication end interrupt, frame end interrupt, or INTIE1 interrupt request signal is generated.

(c) Status transmission flag (STATUSF)...Bit 4

A flag that indicates the master requested transmission of the slave status and lock address (higher 4 bits and lower 8 bits) when the controller was serving as a slave.

<Set/clear conditions>

Set: This flag is set (1) when 0H, 4H, 5H, or 6H is received in the control field from the master when the IEBus is a slave unit.

Clear: This flag is cleared (0) if the start interrupt, communication end interrupt, frame end interrupt, or INTIE1 interrupt request signal is generated.

(d) Communication end flag (ENDTRANS)...Bit 3

A flag that indicates whether communication ends after the number of bytes set in the telegraph length field have been transferred. When a communication error occurs, the INTIE2 and INTSTA interrupt request signals are generated.

<Set/clear conditions>

Set: This flag is set (1) when the count value of the SCR register is 00H.

Clear: This flag is cleared (0) if the start interrupt, status transmission interrupt, frame end interrupt (if the communication end interrupt does not occur), or INTIE1 interrupt request signal is generated.

(e) Frame end flag (ENDFRAM)...Bit 2

A flag that indicates whether communication ends after the maximum number of bytes (mode 1: 32 bytes, mode 2: 128 bytes) have been transferred.

<Set/clear conditions>

Set: This flag is set (1) when the count value of the CCR register is 00H.

Clear: This flag is cleared (0) if the start interrupt, status transmission interrupt, communication end interrupt (if the frame end interrupt does not occur), or INTIE1 interrupt request signal is generated

Cautions 1. If both the CCR and SCR registers are cleared to 00H, the ENDTRNS and ENDFRAM bits are set (1) at the same time.

2. If the last data field is the NACK signal when the maximum number of transmitted bytes is reached as a result of retransmitting the data, the ENDFRAM bit and IEERR (NACK reception error) bit are set at the same time.

(6) IEBus error status register (ESR)

The ESR register indicates the source of the communication error interrupt request signal of IEBus. Each bit of this register is set (1) as soon as the communication error flag (ISR.IEERR bit) is set (1). The source of a communication error, if any, can be identified by checking the contents of this register.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF365H

	<7>	<6>	<5>	<4>	<3>	<2>	1	<0>
ESR	TERR	PERR	NERR	UERR	OERR	WERR	0	DEFLAG

TERR	Timing error occurrence flag
0	Timing error did not occur
1	Timing error occurred

PERR	Parity error occurrence flag
0	Parity error did not occur
1	Parity error occurred

NERR	NACK reception error occurrence flag
0	NACK reception error does not occur
1	NACK reception error occurred

UERR	Underrun error occurrence flag
0	Underrun error did not occur
1	Underrun error occurred

OERR	Overrun error occurrence flag
0	Overrun error did not occur
1	Overrun error occurred

WERR	Write error occurrence flag
0	Write error did not occur
1	Write error occurred

DEFLAG	Third party error occurrence flag
0	Error occurred during communication with unit
1	Error occurred during communication with station other than unit

Cautions 1. Each bit can only be cleared (0). It cannot be set (1) even if 1 is written to it.

2. The value of the ESR register is updated when an error occurs. If the ESR register is read at this time, however, an undefined value is read. It is recommended to read the ESR register in error interrupt servicing.

3. If a communication error occurs, the IEBus controller returns to the default status and makes preparation for communication. If communication is started without the error corrected, the error flag accumulates the error. Correct the error before the next communication is started.

(a) Timing error occurrence flag (TERR)…Bit 7

<Set/clear conditions>

Set: This flag is set (1) if a timing error occurs.

Clear: By software

A timing error occurs if the high-/low-level width of the communication bit is not the defined value. The defined value of the high- and low-level width is set to the bit processing block and monitored by the internal timer. If a timing error occurs, the INTERR and INTIE2 interrupt request signals are generated.

(b) Parity error occurrence flag (PERR)…Bit 6

<Set/clear conditions>

Set: This flag is set (1) if a parity error occurs.

Clear: By software

A parity error occurs if the parity generated in each field does not match the received parity while the controller is serving as a receiver unit. If the parity does not match in the data field during individual communication, however, the NACK signal is returned and retransmission of data is requested. Therefore, the parity error does not occur.

Table 18-14. Operation if Parity Does Not Match

Field	Communication Mode	Operation if Parity Does Not Match
Master address field	Individual/broadcast	Parity error occurs.
Slave address field	Individual/broadcast	Parity error occurs.
Control data field	Individual/broadcast	Parity error occurs.
Telegraph length field	Individual/broadcast	Parity error occurs.
Data field	Individual	Retransmission is requested by returning NACK signal.
	Broadcast	Parity error occurs.

(c) NACK reception error occurrence flag (NERR)…Bit 5

<Set/clear conditions>

Set: This flag is set (1) if a NACK reception error occurs.

Clear: By software

A NACK reception error occurs if the NACK signal is received during the acknowledge bit period of the slave address field, control data field, or telegraph length field during individual communication, regardless of whether the controller is operating as the master or a slave. If the NACK signal is received during the acknowledge bit period of the data field, a NACK reception error does not occur because data is retransmitted. If the NACK signal is received during the acknowledge period of the last data field when the maximum number of transfer bytes is reached, the NACK reception error occurs.

The NACK reception error does not occur during broadcast communication because the ACK/$\overline{\text{NACK}}$ signal is not identified.

The NACK reception error does not occur during third-party communication because only the timing/parity error is detected as an error.

(d) Underrun error occurrence flag (UERR) … Bit 4

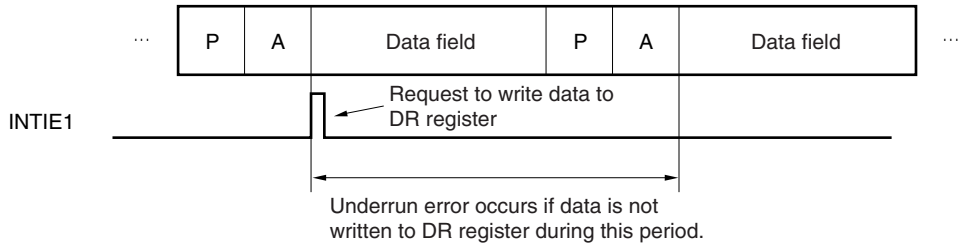
<Set/clear conditions>

Set: This flag is set (1) if an underrun error occurs.

Clear: By software

An underrun error occurs if the next data is not transmitted to the DR register in time before the $\overline{ACK}$ signal is received. If the NACK signal is received during individual communication and during the acknowledge bit period, the underrun error does not occur because the data is retransmitted.

Figure 18-13. Timing of Underrun Error Occurrence



Remark P: Parity bit

A: Acknowledge bit

(e) Overrun error occurrence flag (OERR) … Bit 3

<Set/clear conditions>

Set: This flag is set (1) if an overrun error occurs.

Clear: By software

If 1-byte data is stored in the DR register while the IEBus controller serves as a receiver unit, the data request interrupt request signal (INTIE1) is generated, and the DR register is read by means of DMA or by software. If this reading is delayed and the next data is received, an overrun error occurs.

Cautions 1. If the DR register is not read and the number of retransmitted data reaches the maximum number of transmitted bytes (32 bytes) after the overrun error has occurred, the frame end interrupt request signal (INTSTA or INTIE2) occurs. The overrun status is maintained until the DR register is read, even after the frame has ended.

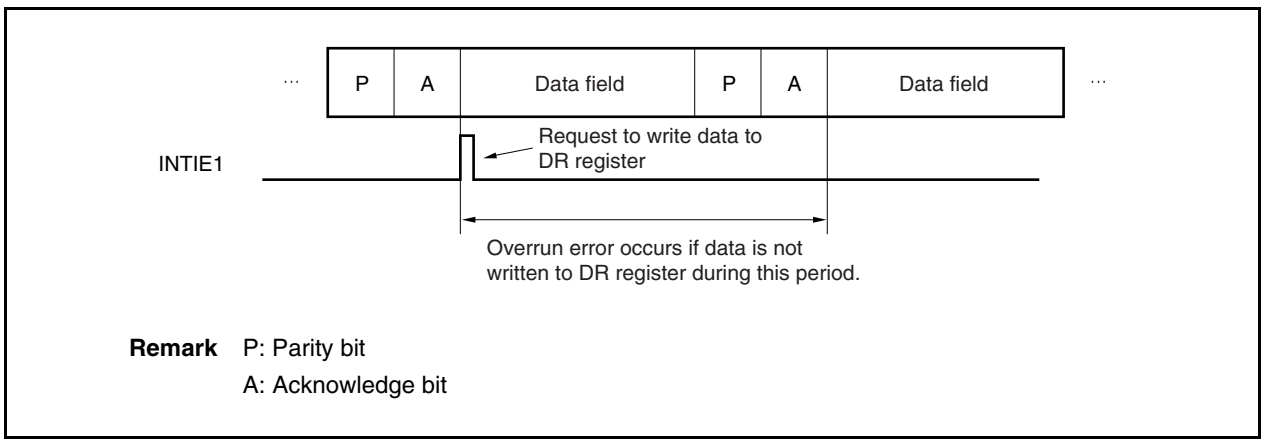
2. The overrun status is cleared only when the DR register is read and when the system is reset. Therefore, be sure to read the DR register in the communication error interrupt processing program.

3. The next data cannot be transmitted in the overrun status if it is 2 bytes or more. Because the data request interrupt request signal (INTIE1) does not occur, the transmit data cannot be set and an underrun error occurs. Therefore, be sure to execute transmission after clearing the overrun status.

Remark During individual communication reception, the NACK signal is returned during the acknowledge bit period of the next data. In response, the transmitter unit retransmits data. Therefore, the CCR register is decremented but the SCR register is not decremented.

During broadcast communication reception, the communication error interrupt request signal (INTIE2) is generated and reception is stopped. At this time, the DR register is not updated. The INTIE1 signal is not generated. The STATRX bit of the SSR register is held set (to 1). The overrun status is cleared when data is received after the DR register has been read.

Figure 18-14. Timing of Overrun Error Occurrence



Remark P: Parity bit

A: Acknowledge bit

(f) Write error occurrence flag (WERR) … Bit 2

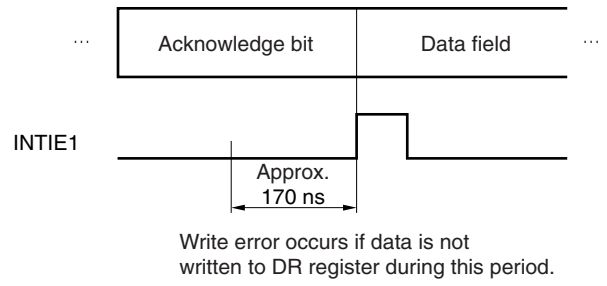
<Set/clear conditions>

Set: This flag is set (1) if a write error occurs.

Clear: By software

A write error occurs if the data written to the DR register is not transmitted in the data field during unit transmission. The timing of occurrence of a write error is illustrated below.

Figure 18-15. Timing of Write Error Occurrence



Cautions 1. Even when the WERR bit is set (1), the INTIE1 interrupt request signal may be generated.

2. If the NACK signal is returned, the WERR bit is not set because data is retransmitted.

(g) Third-party error occurrence flag (DEFLAG)…Bit 0

<Set/clear conditions>

Set: This flag is set (1) if a timing error or parity error occurs during communication regardless of the unit (during communication between third parties).

Clear: By software

Caution If an error occurs before the third-party communication starts even when the slave address field does not match that of the unit (for example, if the NACK signal is received when the received address does not match that of the unit in the slave address field (if the NERR bit is set (1))), the DEFLAG bit is not set (1).

Remark Communication between third parties may take place in the following two cases.

<1> If the received address in the slave address field does not match that of the unit (during individual communication: Matching with UAR register, during broadcast communication: Matching with group or FFFH) and if communication continues after the $\overline{\text{ACK}}$ signal has been received, the unit monitors that communication.

<2> If the unit cannot respond to the received control data in the control field during broadcast communication and if communication continues, the unit monitors that communication. For example, this happens when the unit receives control data FH from master during broadcast communication but the slave reception enable flag of the unit is disabled (BCR.ENSLVRX bit = 0) (the NACK signal is returned and communication ends during individual communication).

(7) IEBus unit address register (UAR)

The UAR register sets the unit address of an IEBus unit. This register must always be set before starting communication.

Sets the unit address (12 bits) to bits 11 to 0.

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

Reset input clears this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
UAR	0	0	0	0													FFFFF366H	0000H	R/W

Caution Do not set the UAR register while communication is enabled (BCR.ENIEBUS bit = 1).

(8) IEBus slave address register (SAR)

During a master request, the value of this register is reflected in the value of the transmit data in the slave address field. The SAR register must always be set before starting communication.

The SAR register sets the slave address (12 bits) to bits 11 to 0.

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

Reset input clears this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
SAR	0	0	0	0													FFFFF368H	0000H	R/W

Caution Do not set the SAR register while communication is enabled (BCR.ENIEBUS bit = 1).

(9) IEBus partner address register (PAR)

The PAR register stores the master address value received in the master address field regardless of whether the unit is operating as the master or a slave.

If a request “4H” to read the lock address (lower 8 bits) is received from the master, read the value of this register by software, and write the data of the lower 8 bits to the DR register.

If a request “5H” to read the lock address (higher 4 bits) is received from the master, read the value of this register by software and write the data of bits 11 to 8 to the higher 4 bits of the DR register.

The PAR register sets the partner address (12 bits) to bits 11 to 0.

This register is read-only, in 16-bit units.

Reset input clears this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
PAR	0	0	0	0													FFFFF36AH	0000H	R

Caution The PAR register stores an address value if the parity is correct and the unit is not locked when the parity period of the master address field expires. If the PAR register is read at this time, an undefined value is read.

(10) IEBus receive slave address register (RSA)

The RSA register stores the slave address value received in the slave address field regardless of whether the unit is operating as the master or a slave.

This register is read-only, in 16-bit units.

Reset input clears this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
RSA	0	0	0	0													FFFFF36CH	0000H	R

Caution The RSA register stores an address value if the parity is correct and the unit is not locked when the parity period of the slave address field expires. If the RSA register is read at this time, an undefined value is read.

(11) IEBus control data register (CDR)

The CDR register can be read or written in 8-bit units.

Reset input clears this register to 00H.

Remark The CDR register consists of a write register and a read register and data written to the CDR register cannot be read as is. The data read from this register is the data received by IEBus communication.

(a) When master unit

The data of the lower 4 bits is reflected in the data transmitted in the control field. During a master request, the CDR register must be set in advance before starting communication.

(b) When slave unit

The data received in the control field is written to the lower 4 bits.

When the status transmission flag (ISR.STATUSF bit) is set (1), an interrupt request signal (INTIE2) is issued, and each processing should be performed by software, according to the value of the lower 4 bits of the CDR register.

After reset: 00H R/W Address: FFFFF36EH

	7	6	5	4	3	2	1	0
CDR	0	0	0	0	MOD	SELCL2	SELCL1	SELCL0

MOD	SELCL2	SELCL1	SELCL0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (lower 4 bits)
0	1	1	0	Read slave status and unlock
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock
1	0	1	1	Write data and lock
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

Cautions 1. Because the slave unit must judge whether the received data is a "command" or "data", read the value of the CDR register after completing communication.

2. If the master unit sets an undefined value, the slave unit returns the NACK signal and communication is aborted. During broadcast communication, the master unit ignores the acknowledge bit and continues communication. Therefore, do not set an undefined value.

(c) Slave status return operation

When IEBus receives a request to transfer from master to slave status or a lock address request (control data: 0H, 6H), whether the $\overline{\text{ACK}}$/NACK signal in the control field is returned or not depends on the status of the IEBus unit.

(1) If 0H or 6H control data was received in the unlocked state → $\overline{\text{ACK}}$ signal returned
(2) If 4H or 5H control data was received in the unlocked state → NACK signal returned
(3) If 0H, 4H, 5H or 6H control data was received in the locked state from the unit that sent the lock request → $\overline{\text{ACK}}$ signal returned
(4) If 0H, 4H, or 5H control data was received in the locked state from other than the unit that sent the lock request → $\overline{\text{ACK}}$ signal returned
(5) If 6H control data was received in the locked state from other than the unit that sent the lock request → NACK signal returned

In all of the above cases, the acknowledgement of a slave status or lock request will cause the STATUSF bit to be set (1) and the status interrupt signal (INTIE2, INTSTA) to be generated. The generation timing is at the end of the control field parity bit (at the start of the acknowledge bit). However, if NACK is returned, a NACK receive error is generated after the acknowledge bit, and communication is terminated.

Figure 18-16. Interrupt Request Signal Generation Timing (for (1), (3), and (4))

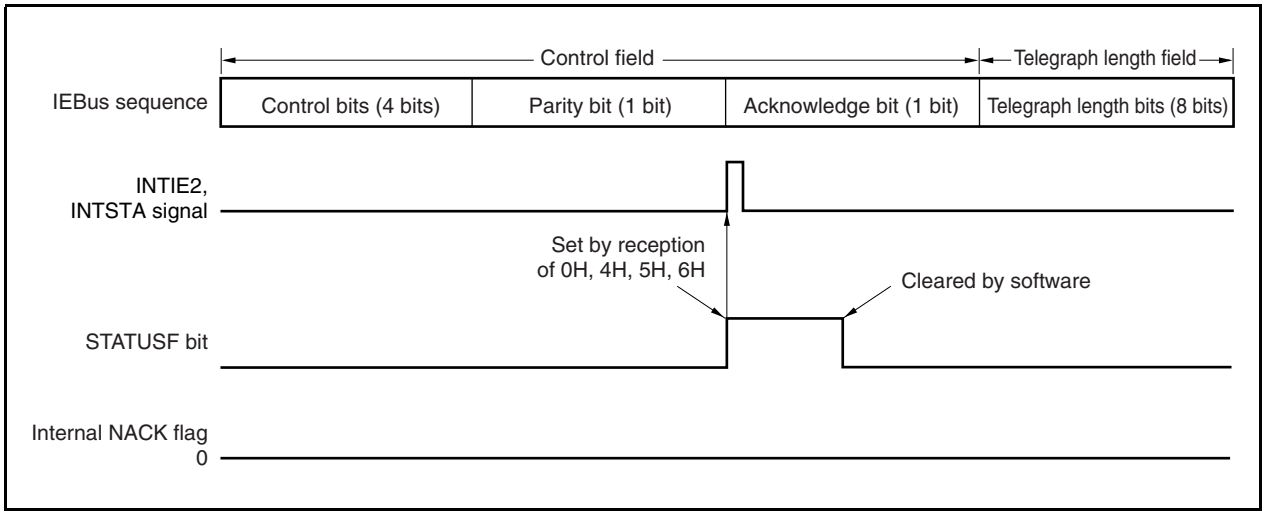
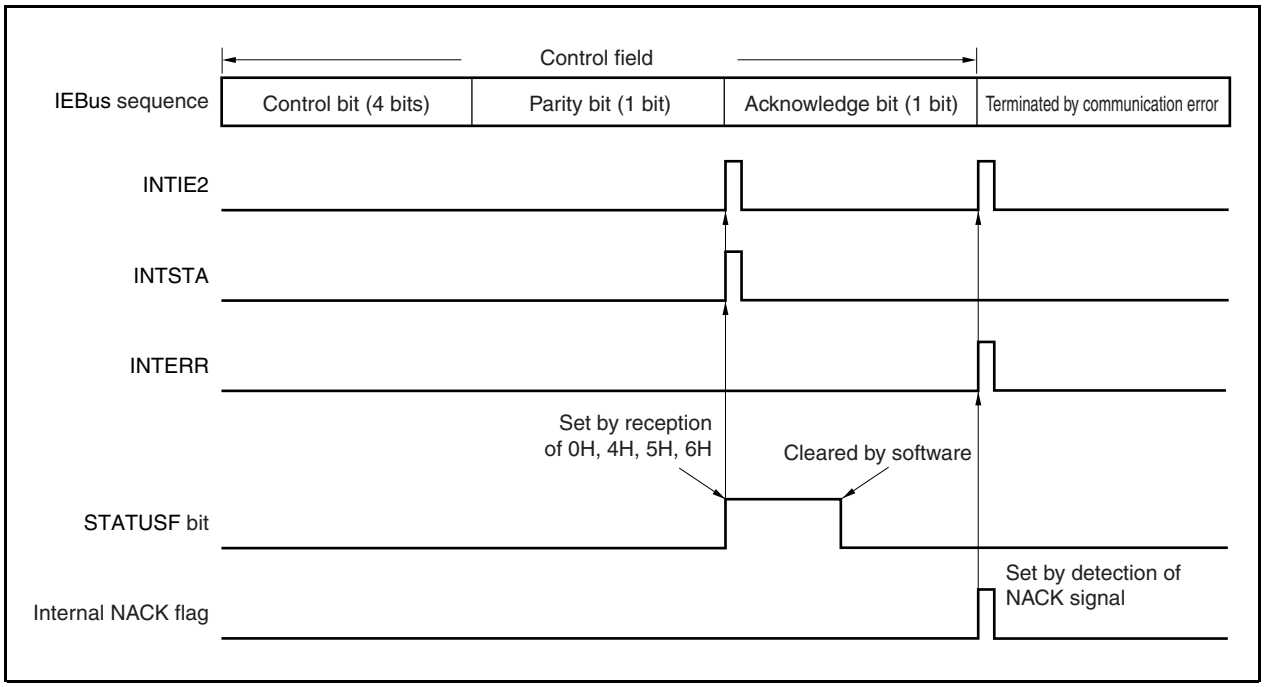
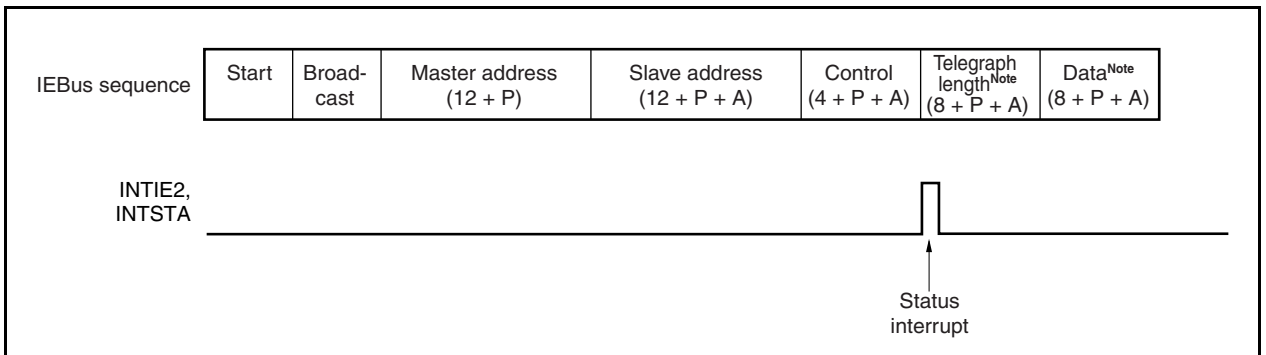


Figure 18-17. Interrupt Request Signal Generation Timing (for (2) and (5))



Because in (4) and (5) the communication was from other than the unit that sent the lock request while IEBus was in the locked state, the start or communication end interrupt request signals (INTIE2, INTSTA) are not generated, even if the IEBus unit is the communication target. The STATUSF bit is set (1) and the status interrupt request signals (INTIE2, INTSTA) are generated, however, if a slave status or lock address request is acknowledged. Note that even if the same control data is received while IEBus is in the locked state, the interrupt generation timing for INTIE2 and INTSTA differs depending on whether the master unit (3) or another unit (4) is requesting the locked state.

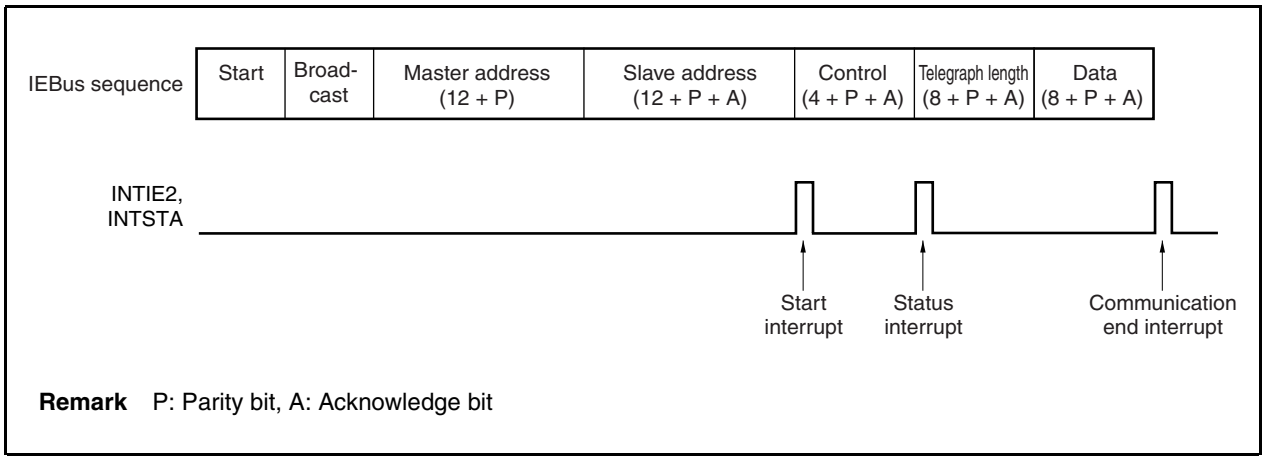
Figure 18-18. Timing of INTIE2 and INTSTA Interrupt Request Signal Generation in Locked State (for (4) and (5))



Note The telegraph length and data modes are not set in the case of (5) because the NACK signal is returned.

Remark P: Parity bit, A: Acknowledge bit

Figure 18-19. Timing of INTIE2 and INTSTA Interrupt Request Signal Generation in Locked State (for (3))



Remark P: Parity bit, A: Acknowledge bit

(12) IEBus telegraph length register (DLR)

The DLR register can be read or written in 8-bit units.

Reset input sets this register to 01H.

(a) When transmission unit ... Master transmission, slave transmission

The data of this register is reflected in the data transmitted in the telegraph length field and indicates the number of bytes of the transmit data. The DLR register must be set in advance before transmission.

(b) When reception unit ... Master reception, slave reception

The receive data in the telegraph length field transmitted from the transmission unit is written to this register.

Remark The DLR register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

After reset: 01H R/W Address: FFFFF36FH

	7	6	5	4	3	2	1	0
DLR								

Bit								Setting value	Remaining number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	256 bytes

Cautions 1. When the master issues a request (0H, 4H, 5H, or 6H) for transmission of a slave status or a lock address (higher 4 bits and lower 8 bits), 01H is transmitted as the telegraph length regardless of the contents of the DLR register. It is therefore not necessary to set the DLR register by software.

2. When the IEBus controller serves as a receiver unit, the DLR register stores a telegraph length if the value of the parity bit of the telegraph length field is correct. If the DLR register is read at this time, an undefined value is read.

(13) IEBus data register (DR)

The DR register sets the communication data (8 bits) to bits 7 to 0.

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

Reset input clears this register to 00H.

Remark The DR register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

(a) When transmission unit

The data (1 byte) written to the DR register is stored in the transmit shift register of the IEBus interface block. It is then output from the most significant bit, and an interrupt request signal (INTIE1) is generated each time 1 byte has been transmitted. If the NACK signal is received after 1-byte data has been transferred during individual transfer, data is not transferred from the DR register to the transmit shift register, and the same data is retransmitted. At this time, INTIE1 signal is not generated.

INTIE1 signal is generated when the transmit shift register stores the DR register value. However, when the last byte and 32nd byte (the last byte of 1 communication frame) is stored in the transmit shift register, the INTIE1 signal is not generated.

(b) When reception unit

One byte of the data received by the receive shift register of the IEBus interface block is stored in this register.

Each time 1 byte has been correctly received, an interrupt request signal (INTIE1) is generated.

When transmit/receive data is transferred to and from the DR register, using DMA can reduce the CPU processing load.

After reset: 00H R/W Address: FFFFF370H

	7	6	5	4	3	2	1	0
DR								

Cautions 1. If the next data is not in time while the transmission unit is set, an underrun occurs, and a communication error interrupt request signal (INTIE2, INTERR) occurs, stopping transmission.

2. If data is not read in time before the next data is read when the IEBus controller functions as a receiver unit during individual communication reception, the NACK signal is returned by the acknowledge bit of the data field, requesting the master to retransmit the data. If the DR register is not read after the data has reached the maximum number of transmit bytes, however, the frame end interrupt request signal (INTIE2, INTSTA) and NACK reception error interrupt request signal (INTIE2, INTERR) are generated at the same time.

3. If data is not read in time before the next data is received when the IEBus controller functions as a receiver unit during broadcast communication reception, an overrun error occurs and the communication error interrupt request signal (INTIE2, INTERR) is generated.

4. When the IEBus controller serves as a receiver unit, the DR register stores receive data if the value of the parity bit of the data field is correct. If the DR register is read at this time, an undefined value is read.

(14) IEBus field status register (FSR)

The FSR register stores the status of the field status of the IEBus controller if an interrupt request signal (INTIE1, INTIE2, INTSTA, or INTERR) is generated.

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

Reset input clears this register to 00H.

Cautions 1. If an interrupt request signal is generated during communication between third parties, the FSR register is cleared to 00H. However, because only an interrupt request signal that is generated if an error occurs is generated during communication between third parties, the error can be identified as that during communication between third parties, by reading third-party error flag (ESR.DEFLAG bit).

2. The FSR register updates the status information when an interrupt request signal is generated. If the FSR register is read at this time, however, an undefined value is read.

3. If another interrupt request signal is generated before the FSR register is read, the status information when the preceding interrupt occurred is updated by the status information when the new interrupt occurs.

4. Use the FSR register only for problem analysis; do not use it with the actual software.

After reset: 00H R Address: FFFFF371H

	7	6	5	4	3	2	1	0
FSR	0	0	0	0	0	0	FSTATE1	FSTATE0

Remark For the explanation of the FSTATE1 and FSTATE0 bits, refer to **Table 18-15 Field Status**.

Table 18-15. Field Status

Field Status	Explanation		
	Master/Slave	Field	Transmission/Reception
Slave transmission status FSR = 00H	Slave operation	Start field	Reception
		Master address field	
		Slave address field	
		Control data field	
		Telegraph length field	
		Data field	
Slave transmission status FSR = 01H	Slave operation	Telegraph length field	Transmission
		Data field	
Master reception status FSR = 02H	Master operation	Telegraph length field	Reception
		Data field	
Master transmission status FSR = 03H	Master operation	Start field	Transmission
		Master address field	
		Slave address field	
		Control data field	
		Telegraph length field	
		Data field	

(15) IEBus success count register (SCR)

The SCR register indicates the number of remaining communication bytes.

The count value of the counter in which the value set by the DLR register is decremented by the $\overline{ACK}$ signal in the data field is read from this register. When the count value has reached “00H”, the communication end flag (ISR.ENDTRNS bit) is set (1).

After reset: 01H R Address: FFFFF372H

	7	6	5	4	3	2	1	0
SCR								

Bit								Setting value	Remaining number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	0 bytes (end of communication) or 256 bytes[Note]

Note The actual counter consists of 9 bits. When “00H” is read, it cannot be judged whether the remaining number of communication data bytes is 0 (end of communication) or 256. Therefore, either the communication end flag (ENDTRNS bit) is used, or if “00H” is read when the first interrupt occurs at the beginning of communication, the remaining number of communication data bytes is judged to be 256.

Caution The SCR register is updated when the parity period of the telegraph field expires and when the $\overline{ACK}$ signal of the data field is received. If the SCR register is read at this time, however, an undefined value is read.

(16) IEBus communication count register (CCR)

The CCR register indicates the number of bytes remaining from the communication byte number specified by the communication mode.

This register indicates the number of transfer bytes.

The maximum number of transmitted bytes per frame defined in each mode (mode 1: 32 bytes, mode 2: 128 bytes) is preset to this register. The count value of the counter that is decremented during the acknowledge bit period of the data field regardless of the $\overline{\text{ACK}}$/NACK signal is read from this register. Whereas the SCR register is decremented during normal communication ($\overline{\text{ACK}}$ signal), the CCR register is decremented when 1 byte has been communicated, regardless of whether the signal is $\overline{\text{ACK}}$ or NACK. When the count value has reached “00H”, the frame end flag (ISR.ENDFRAM bit) is set (1).

The preset value of the maximum number of transmitted bytes per frame is 20H (32 bytes) in mode 1 and 80H (128 bytes) in mode 2.

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

Reset input sets this register to 20H.

After reset: 20H R Address: FFFFF373H

	7	6	5	4	3	2	1	0
CCR								

Caution The maximum number of transmit bytes is preset to the CCR register when the start bit is transmitted or received, and the register is decremented when the parity period of the data field expires. If the CCR register is read at this time, however, an undefined value is read.

(17) IEBus clock select register (OCKS2)

The OCKS2 register selects the clock of IEBus. The main clock frequencies that can be used are shown below. No other main clock frequencies can be used.

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

Reset input clears this register to 00H.

- 6.0 MHz/6.291 MHz
- 12.0 MHz/12.582 MHz
- 18.0 MHz/18.873 MHz

After reset: 00H R/W Address: FFFFF348H

	7	6	5	4	3	2	1	0
OCKS2	0	0	0	OCKSEN2	0	OCKS22	OCKS21	OCKS20

OCKSEN2	IEBus clock operation specification
0	IEBus clock operation stops
1	5 IEBus clock operation enabled

OCKS22	OCKS21	OCKS20	IEBus clock selection
0	0	0	$f_{XX}/2$ (when f_{XX} = 12.0 MHz or f_{XX} = 12.852 MHz)
0	0	1	$f_{XX}/3$ (when f_{XX} = 18.0 MHz or f_{XX} = 18.873 MHz)
1	0	0	f_{XX} (when f_{XX} = 6.0 MHz or f_{XX} = 6.291 MHz)
Other than above			Setting prohibited

18.4 Interrupt Operations of IEBus Controller

18.4.1 Interrupt control block

Interrupt request signal

<1>	Communication error	IEERR
	(i) Timing error:	TERR
	(ii) Parity error:	PERR
	(iii) NACK receive error:	NERR
	(iv) Underrun error:	UERR
	(v) Overrun error:	OERR
	(vi) Write error:	WERR
<2>	Start interrupt	STARTF
<3>	Status communication	STATUSF
<4>	End of communication	ENDTRNS
<5>	End of frame	ENDFRAM
<6>	Transmit data write request	$\overline{\text{STATTX}}$
<7>	Receive data read request	STATRX

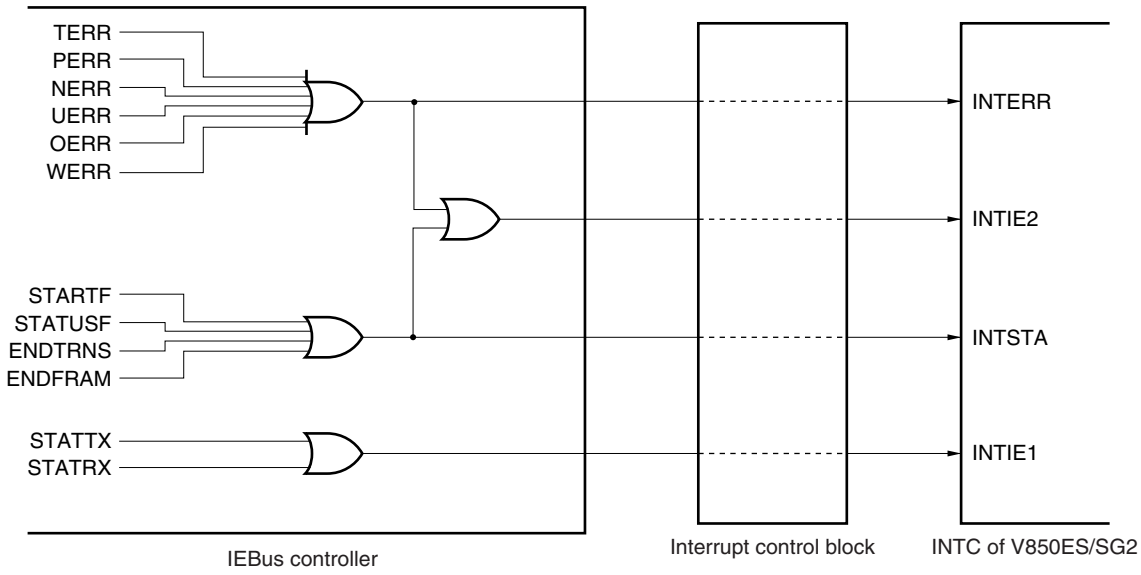
A communication error <1> occurs if any of the above error sources (i) to (vi) is generated.

These error sources are assigned to the error status register (ESR) (refer to **Table 18-18 Communication Error Source Processing List**).

The above interrupt signals <1> to <5> are assigned to the ISR register (refer to **Table 18-17 Interrupt Source List**).

The configuration of the interrupt control block is illustrated below.

Figure 18-20. Configuration of Interrupt Control Block



Cautions 1. The logical sum (OR) output of the STATRX and $\overline{\text{STATTX}}$ signals is treated as an interrupt request signal (INTIE1).

2. The logical sum (OR) output of the TERR, PERR, NERR, UERR, OERR, and WERR signals is treated as a communication error (IEERR) or an interrupt request signal (INTERR).

3. The logical sum (OR) output of the STARTF, STATUSF, ENDTRNS, and ENDFRAM signals is treated as an interrupt request signal (INTSTA).

4. The logical sum (OR) output of the IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM signals (logical sum (OR) output of INTSTA and INTERR signals) is treated as an interrupt request signal (INTIE2).

Table 18-16. Interrupt Request Signal Generation Source List

Interrupt Source		Symbol	Interrupt Request Signal			
			INTIE1	INTIE2	INTERR	INTSTA
Communication error interrupt		IEERR		√	√	
	Timing error	TERR				
	Parity error	PERR				
	NACK reception error	NERR				
	Underrun error	UERR				
	Overrun error	OERR				
	Write error	WERR				
Start interrupt		STARTF		√		√
Status transmission		STATUSF		√		√
End of Communication		ENDTRNS		√		√
End of frame		ENDFRAM		√		√
Transmit data write request		$\overline{\text{STATTX}}$	√			
Receive data write request		STATRX	√			

18.4.2 Example of identifying interrupt

The IEBus controller processes interrupts in the following two ways.

- Using three interrupt request signals: INTIE1, INTERR, and INTSTA
- Using two interrupt request signals: INTIE1 and INTIE2

Caution Mask the interrupt sources that are not used so that the interrupts do not occur.

How an interrupt is identified in each of the above cases is explained below.

(1) When INTIE1, INTERR, and INTSTA signals are used

Figure 18-21. Example of Identifying INTIE1 Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)

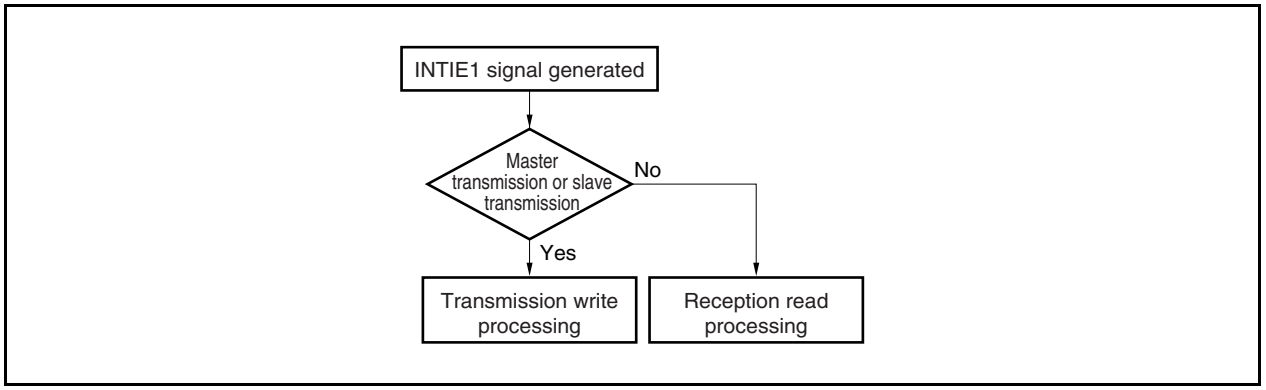


Figure 18-22. Example of Identifying INTERR Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)

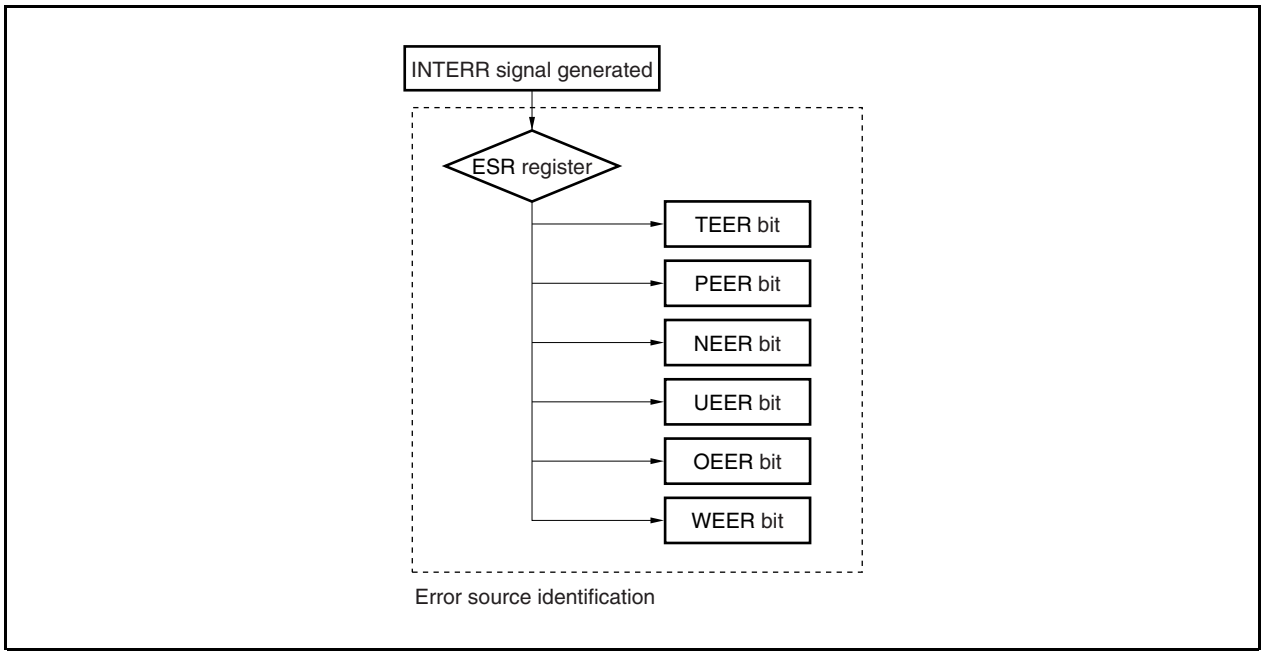
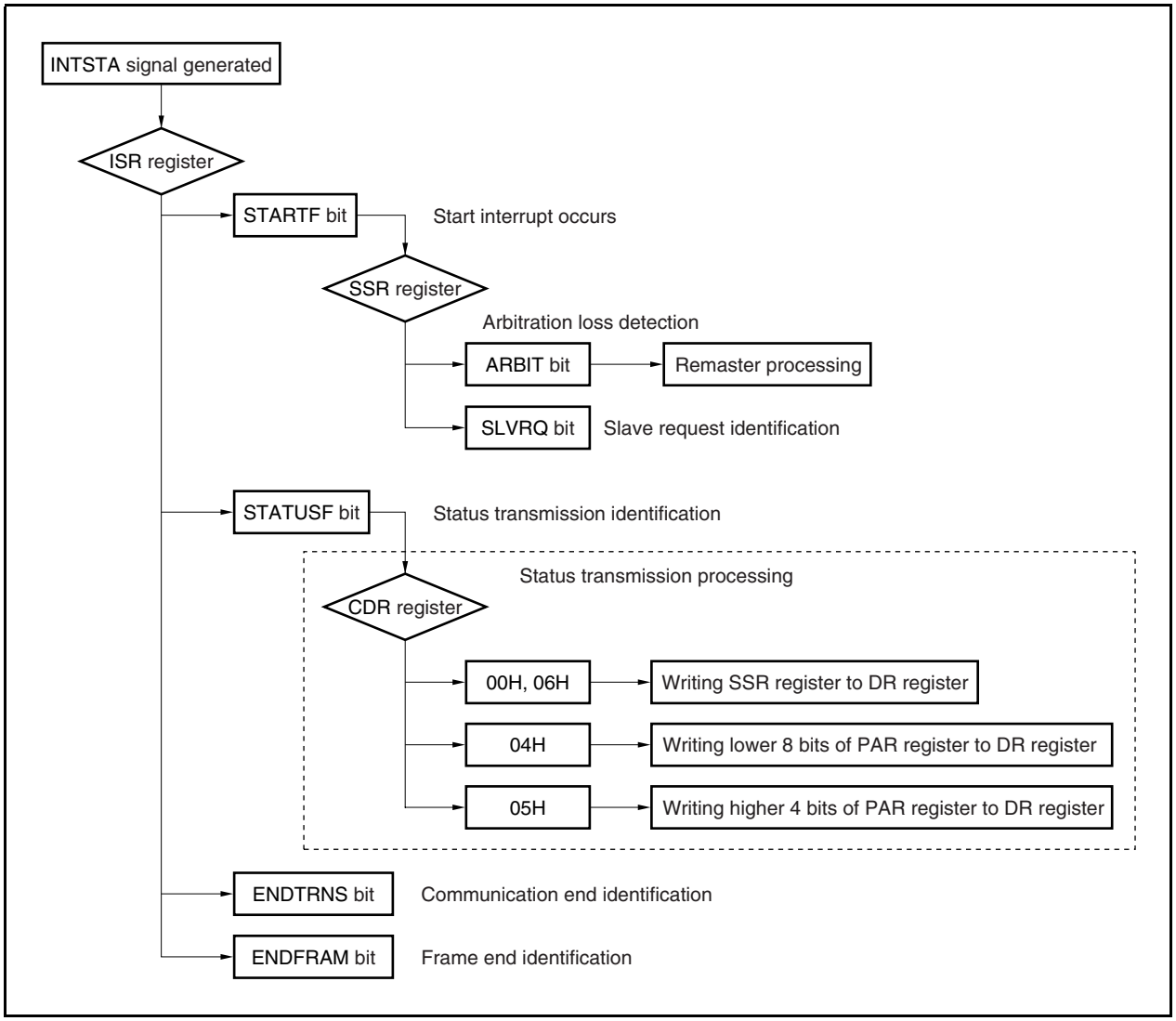


Figure 18-23. Example of Identifying INTSTA Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)



(2) When INTIE1 and INTIE2 signals are used

Figure 18-24. Example of Identifying INTIE1 Signal Interrupt (When INTIE1 and INTIE2 Signals Are Used)

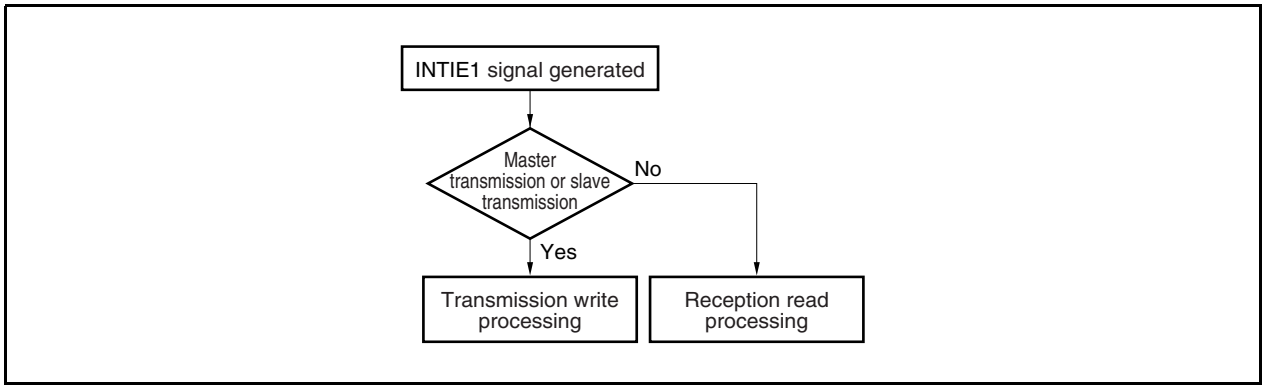
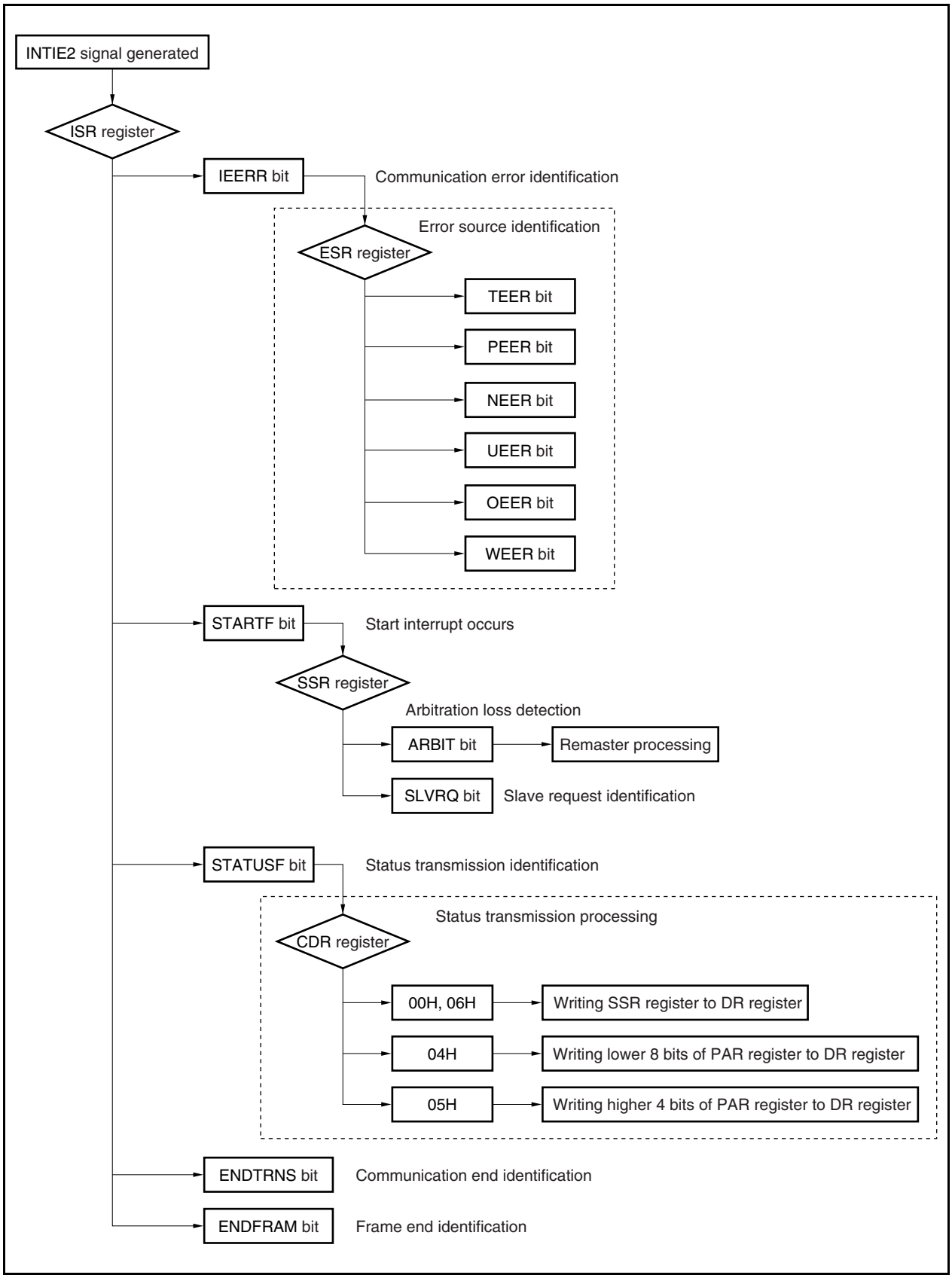


Figure 18-25. Example of Identifying INTIE2 Signal Interrupt (When INTIE1 and INTIE2 Signals Are Used)



18.4.3 Interrupt source list

The interrupt request signals of the internal IEBus controller in the V850ES/SG2 can be classified into vector interrupts and DMA transfer interrupts. These interrupt request signals can be specified via software manipulation.

The interrupt sources are listed below.

Table 18-17. Interrupt Source List

Interrupt Source		Condition of Generation		Software Processing After Generation of Interrupt Request Signal	Remark
		Unit	Field		
Communication error	Timing error	Master/slave	All fields	Undo communication processing	Communication error is logical sum (OR) output of timing error, parity error, NACK reception error, underrun error, overrun error, and write error.
	Parity error	Reception	Other than data (individual)		
			All fields (broadcast)		
	NACK reception	Reception (Transmission)	Other than data (individual)		
	Underrun error	Transmission	Data		
	Overrun error	Reception	Data (broadcast)		
	Write error	Transmission	Data		
Start interrupt		Master	Slave/address	Slave request judgment Arbitration judgment (If lost, remaster processing) Communication preparation processing	Interrupt always occurs if lost in arbitration during master request
		Slave	Slave/address	Slave request judgment Communication preparation processing	Generated only during slave request
Status transmission		Slave	Control	Refer to transmission processing example such as slave status.	Interrupt occurs regardless of slave transmission enable flag Interrupt occurs if NACK is returned in the control field.
End of communication		Transmission	Data	DMA transfer end processing	Set if SCR register is cleared to 00H
		Reception	Data	DMA transfer end processing Receive data processing	
End of frame		Transmission	Data	Retransmission preparation processing	Set if CCR register is cleared to 00H
		Reception	Data	Re-reception preparation processing	
Transmit data write		Transmission	Data	Reading of transmit data[Note]	Set after transfer transmission data to internal shift register This does not occur when the last data is transferred.
Receive data read		Reception	Data	Reading of received data[Note]	Set after normal data reception

Note If DMA transfer or software manipulation is not executed.

18.4.4 Communication error source processing list

The following table shows the occurrence conditions of the communication errors (timing error, NACK reception error, overrun error, underrun error, parity error, and write error), error processing by the IEBus controller, and examples of processing by software.

Table 18-18. Communication Error Source Processing List (1/2)

		Timing Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	If bit specification timing is not correct			
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 signal occurs • To start bit waiting status **Remark** Communication between other units does not end.		• Transmission stops. • INTIE2 signal occurs • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs • NACK signal is returned. • To start bit waiting status		• Transmission stops. • INTIE2 signal occurs • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	

		NACK Reception Error				
Occurrence condition	Unit status	Reception		Transmission		
	Occurrence condition	Unit NACK signal transmission		Unit NACK signal transmission		
	Location of occurrence	Other than data field	Data field	Other than data field	Data field	NACK signal reception of data of 32nd byte
Broadcast communication	Hardware processing	–	–	–	–	–
	Software processing	–	–	–	–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• INTIE2 signal does not occur. • Data retransmitted by other unit is received.	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• INTIE2 signal does not occur. • Retrans-mission processing	• INTIE2 signal occurs. • To start bit waiting status
	Software processing	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

Table 18-18. Communication Error Source Processing List (2/2)

		Overrun Error		Underrun Error/Write Error	
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	DR register cannot be read in time before the next data is received.		DR register cannot be written in time before the next data is transmitted.	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	–	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status **Remarks 1.** Communication between other units does not end. **2.** Data cannot be received until the overrun status is cleared.	–	• Transmission stops. • INTIE2 signal occurs. • To start bit waiting status
	Software processing	–	• DR register is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)
Individual communication	Hardware processing	–	• INTIE2 signal does not occur. • NACK signal is returned. • Data is retransmitted from other unit. **Remark** Data cannot be received until overrun status is cleared.	–	• Transmission stops. • INTIE2 signal occurs. • To start bit waiting status
	Software processing	–	• DR register is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

		Parity Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	Received data and received parity do not match.		–	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status **Remark** Communication between other units does not end.		–	–
	Software processing	• Error processing (such as retransmission request)		–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• Reception does not stop. • INTIE2 signal does not occur. • NACK signal is returned. • Data retransmitted by other unit is received.	–	–
	Software processing	• Error processing (such as retransmission request)	–	–	–

18.5 Interrupt Request Signal Generation Timing and Main CPU Processing

18.5.1 Master transmission

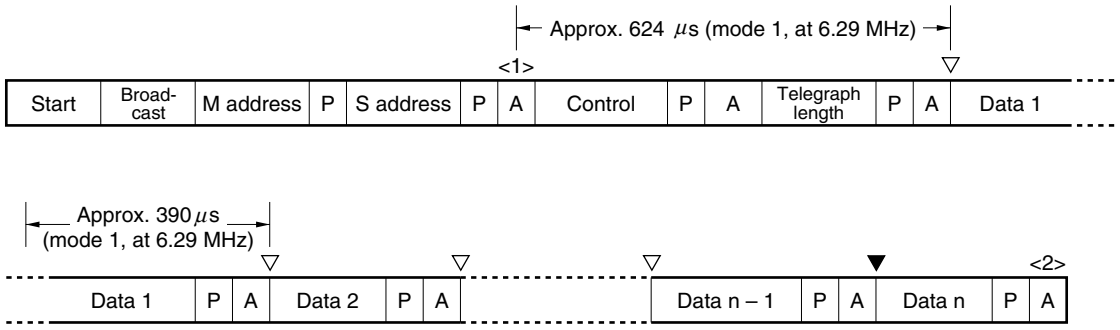
Initial preparation processing:

Sets a unit address, slave address, control data, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the BCR register (enable communication, master request, and slave reception).

Figure 18-26. Master Transmission



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of slave request → Slave reception processing
(See **18.5.1 (1) Slave reception processing**)

↓

Judgment of arbitration result → Remaster request processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Recommunication processing
(See **18.5.1 (3) Recommunication processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

Remarks 1. ▽: Interrupt request signal (INTIE1) occurrence (See **18.5.1 (2) Interrupt request signal (INTIE1) occurrence**)
The transmit data of the second and subsequent bytes is written to the DR register by DMA transfer.
At this time, the data transfer direction is RAM → on-chip peripheral I/O

2. ▼: An interrupt request signal (INTIE1) does not occur.

3. n = Final number of data bytes

(1) Slave reception processing

If a slave reception request is confirmed during vector interrupt servicing, the data transfer direction of the macro service must change from RAM → on-chip peripheral I/O to on-chip peripheral I/O → RAM until the first data is received. The maximum pending period of this data transfer direction changing processing is about 1,040 μs in communication mode 1 (at 6.29 MHz).

(2) Interrupt request signal (INTIE1) occurrence

If the NACK signal is received from the slave in the data field, an interrupt request signal (INTIE1) is not issued to the interrupt controller (INTC), and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt request signal (INTERR) occurs due to occurrence of underrun, and communication ends midway.

(3) Recommunication processing

In the vector interrupt servicing in <2>, it is judged whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remainder of the data must be transmitted.

18.5.2 Master reception

Before performing master reception, it is necessary to notify the unit that will be the slave of slave transmission. Therefore, more than two communication frames are necessary for master reception.

The slave unit prepares the transmit data, sets (1) the slave transmission enable flag (BCR.ENSLVTX bit), and waits.

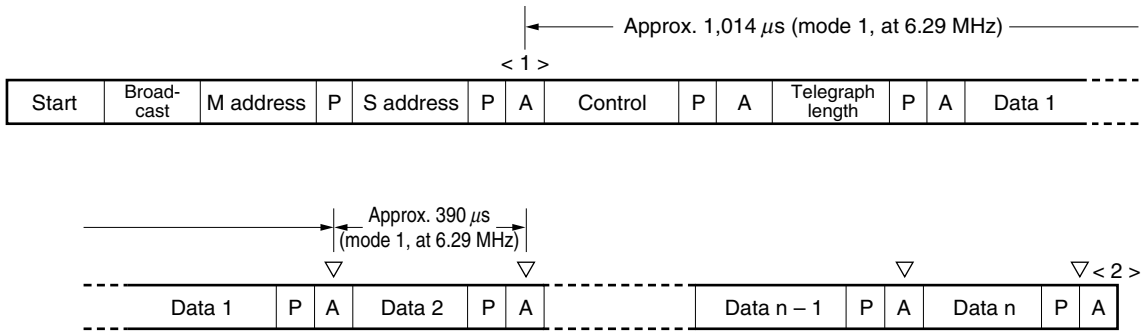
Initial preparation processing:

Set a unit address, slave address, and control data.

Communication start processing:

Set the BCR register (enable communication and master request).

Figure 18-27. Master Reception



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of slave request → Slave processing
↓
Judgment of arbitration result → Remaster request processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of end of communication → End of communication processing
↓
Judgment of end of frame → Frame end processing (See **18.5.2 (2) Frame end processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

Remarks 1. ▽: Interrupt request signal (INTIE1) occurrence (See **18.5.2 (1) Interrupt request signal (INTIE1) occurrence**)

The receive data stored in the DR register is read by DMA transfer.

At this time, the data transfer direction is on-chip peripheral I/O → RAM.

2. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is transmitted (hardware processing) in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted from the slave.

If the receive data is not read by the time the next data is received, the hardware automatically transmits the NACK signal.

(2) Frame end processing

In the vector interrupt servicing in <2>, it is judged whether the data has been correctly received within one frame. If the data has not been correctly received (if the number of data to be received in one frame could not be received), a request to retransmit the data must be made to the slave in the next communication frame.

18.5.3 Slave transmission

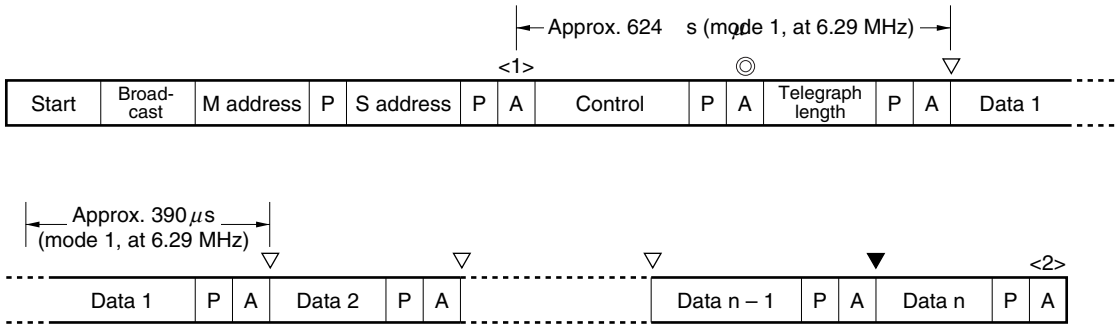
Initial preparation processing:

Set a unit address, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the BCR register (enable communication, slave transmission, and slave reception).

Figure 18-28. Slave Transmission



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of slave request

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **18.5.3 (2) Frame end processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

Remarks 1. ▽: Interrupt request signal (INTIE1) occurrence (See **18.5.3 (1) Interrupt request signal (INTIE1) occurrence**).

The transmit data of the second and subsequent bytes is written to the DR register by DMA transfer.

At this time, the data transfer direction is RAM → on-chip peripheral I/O.

2. ▼: An interrupt request signal (INTIE1) does not occur.

3. ◎: Interrupt request signal (INTIE2) occurrence

An interrupt request signal occurs only when 0H, 4H, 5H, or 6H is received in the control field in the slave status (for the slave status response operation during the locked status, refer to **18.3 (11) IEBus control data register (CDR)**).

4. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is received from the master in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt request signal (INTERR) occurs due to occurrence of underrun, and communication is abnormally ended.

(2) Frame end processing

In the vector interrupt servicing in <2>, it is judged whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remaining data must be transmitted.

18.5.4 Slave reception

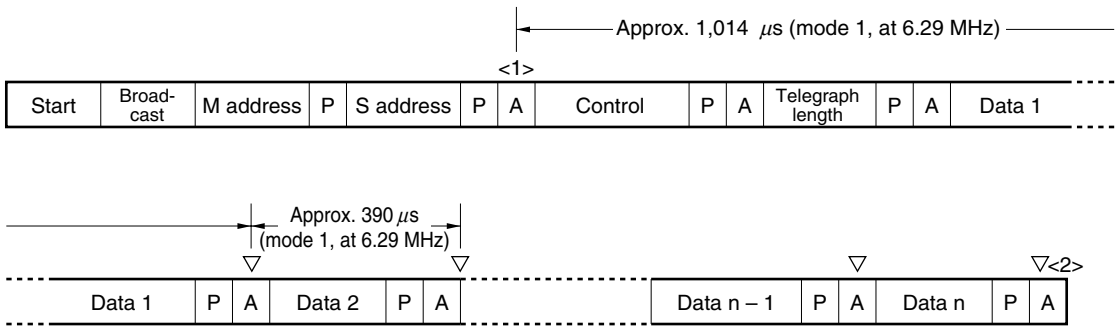
Initial preparation processing:

Set a unit address.

Communication start processing:

Set the BCR register (enable communication, disables slave transmission, and enables slave reception).

Figure 18-29. Slave Reception



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of slave request → Slave processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **18.5.4 (2) Frame end processing**).

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

Remarks 1. ▽: Interrupt request signal (INTIE1) occurrence (See **18.5.4 (1) Interrupt request signal (INTIE1) occurrence**).

The receive data stored in the DR register is read by DMA transfer.

At this time, the data transfer direction is on-chip peripheral I/O → RAM.

2. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is transmitted in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted from the master.

If the receive data is not read by the time the next data is received, the NACK signal is automatically transmitted.

(2) Frame end processing

In the vector interrupt servicing in <2>, it is judged whether the data has been correctly received within one frame.

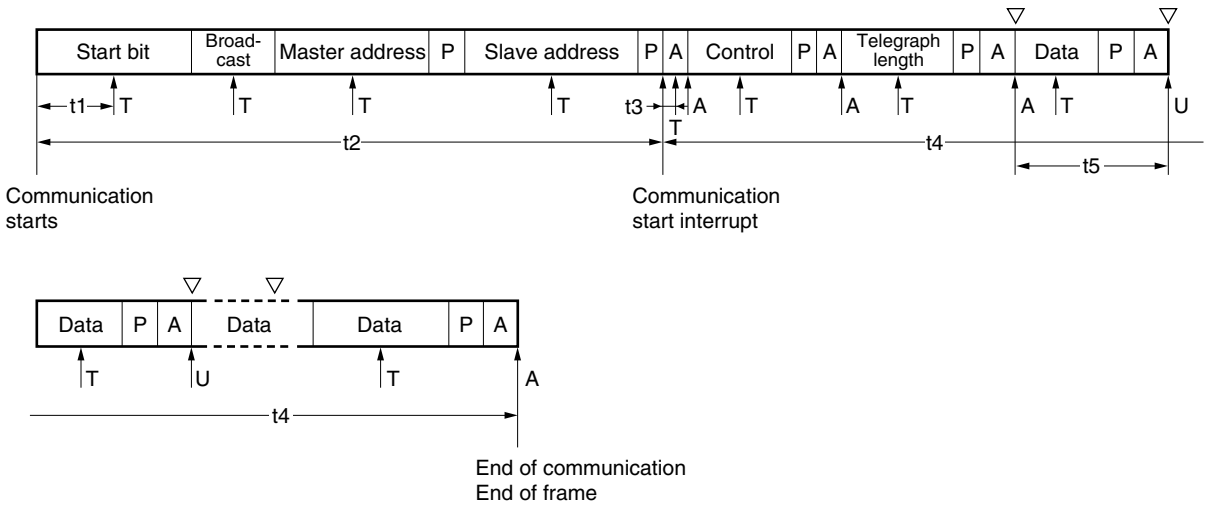
18.5.5 Interval of occurrence of interrupt request signal for IEBus control

Each control interrupt request signal must occur at each point of communication and perform the necessary processing until the next interrupt request signal occurs. Therefore, the IEBus control block is controlled by software, taking the shortest time of this interrupt request signal occurrence interval into consideration.

The locations at which the following interrupt request signals may occur are indicated by ↑ in the field where it may occur. ↑ does not mean that the interrupt request signal occurs at each of the points indicated by ↑. If an error interrupt request signal (timing error, parity error, or NACK receive error) occurs, the IEBus internal circuit is initialized. As a result, the following interrupt request signal does not occur in that communication frame.

(1) Master transmission

Figure 18-30. Master Transmission (Interval of Interrupt Request Signal Occurrence)

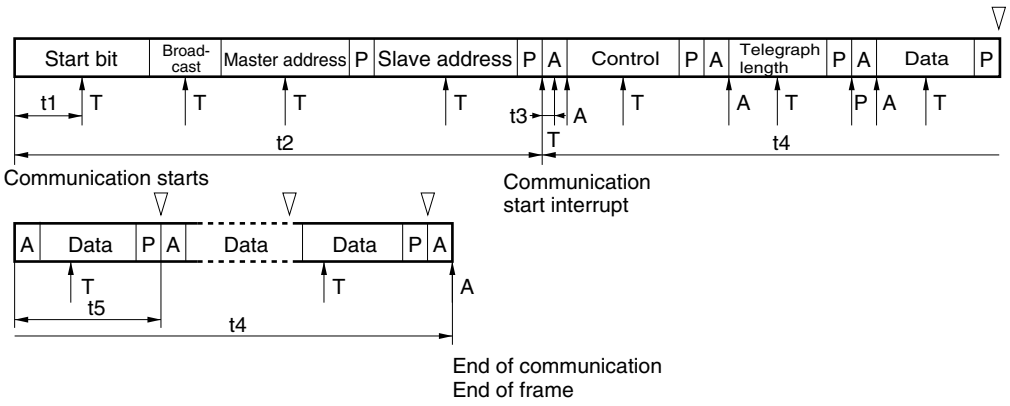


Remarks 1. T: Timing error
A: NACK receive error
U: Underrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.
Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).
t1: Communication starts → timing error (approx. 93 μs)
t2: Communication starts → communication start interrupt (approx. 1,282 μs)
t3: Communication start interrupt → timing error (approx. 15 μs
t4: Communication start interrupt → end of communication (approx. 1,012 μs)
t5: Transmission data request interrupt interval (approx. 375 μs)

(2) Master reception

Figure 18-31. Master Reception (Interval of Interrupt Request Signal Occurrence)



Remarks 1. T: Timing error

P: Parity error

A: NACK receive error

▽: Data set interrupt request signal (INTIE1)

2. End of frame occurs at the end of 32-byte data.

Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).

t1: Communication starts → timing error (approx. 93 μs)

t2: Communication starts → communication start interrupt (approx. 1,282 μs)

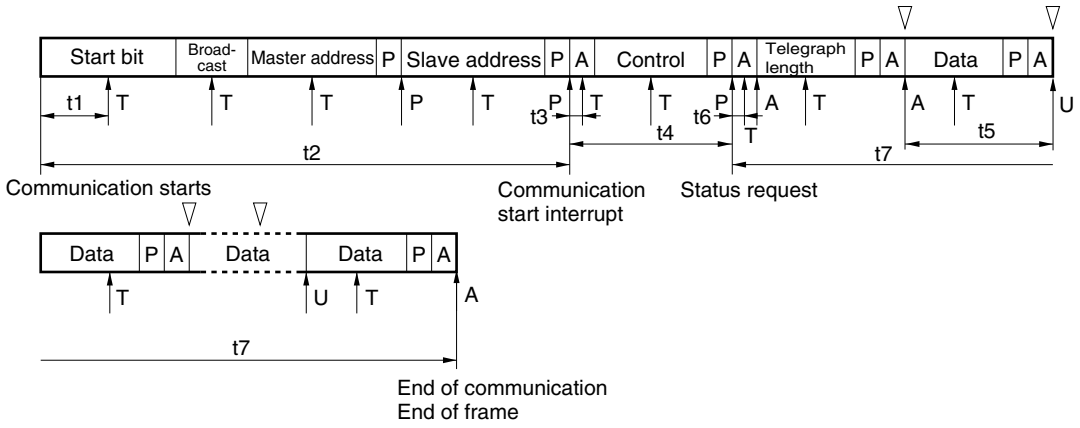
t3: Communication start interrupt → timing error (approx. 15 μs)

t4: Communication start interrupt → end of communication (approx. 1,012 μs)

t5: Receive data read interval (approx. 375 μs)

(3) Slave transmission

Figure 18-32. Slave Transmission (Interval of Interrupt Request Signal Occurrence)

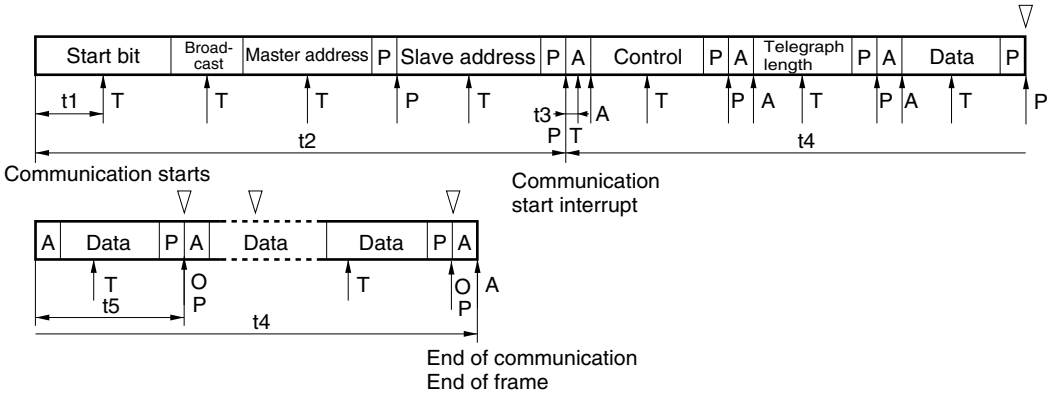


Remarks 1. T: Timing error
P: Parity error
A: NACK receive error
U: Underrun error
▽: Data set interrupt request signal (INTIE1)

2. End of frame occurs at the end of 32-byte data.
Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).
t1: Communication starts → timing error (approx. 96 μs)
t2: Communication starts → communication start interrupt (approx. 1,192 μs)
t3: Communication start interrupt → timing error (approx. 15 μs)
t4: Communication start interrupt → status request (approx. 225 μs)
t5: Transmission data request interrupt interval (approx. 375 μs)
t6: Status request → timing error (approx. 15 μs)
t7: Status request → end of communication (approx. 787 μs)

(4) Slave reception

Figure 18-33. Slave Reception (Interval of Interrupt Request Signal Occurrence)



Remarks 1. T: Timing error

P: Parity error

A: NACK receive error

O: Overrun error

▽: Data set interrupt request signal (INTIE1)

2. End of frame occurs at the end of 32-byte data.

Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).

t1: Communication starts → timing error (approx. 96 μs)

t2: Communication starts → communication start interrupt (approx. 1,192 μs)

t3: Communication start interrupt → timing error (approx. 15 μs)

t4: Communication start interrupt → end of communication (approx. 1,012 μs)

t5: Receive data read interval (approx. 375 μs)

CHAPTER 19 CAN CONTROLLER

19.1 Outline

The V850ES/SG2 features an on-chip 1-channel CAN (Controller Area Network) controller that complies with the CAN protocol as standardized in ISO 11898.

The V850ES/SG2 products with an on-chip CAN controller are as follows.

- μPD703280, 703280Y, 703281, 703281Y, 703282, 703282Y, 703283, 703283Y, 70F3281, 70F3281Y, 70F3283, 70F3283Y

19.1.1 Features

- Compliant with ISO 11898 and tested according to ISO DIS 16845
- Standard frame and expanded frame transmission/reception enabled
- Transfer rate: 1 Mbps max.
- 32 message buffers/channel
- Receive/transmit history list function
- Automatic block transmission function
- Multi-buffer receive block function
- Mask setting of four patterns is possible for each channel.

19.1.2 Overview of functions

Table 19-1 presents an overview of CAN controller functions.

Table 19-1. Overview of Functions

Function	Description
Protocol	CAN Protocol ISO 11898 (standard and extended frame transmission/reception)
Baud rate	Maximum 1 Mbps (@CAN clock input ≥ 8 MHz)
Data storage	32 message buffers/channel Each message buffer can be set to be either a transmit message buffer or a receive message buffer.
Message reception	• Unique ID can be set in each message buffer. • Mask setting of four patterns is possible for each channel. • CAN receive interrupt after each message storage • Buffers for reception can be used to form a multi-buffer receive FIFO (multi-buffer receive block function). • Receive history list function
Message transmission	• Unique ID can be set in each message buffer. • CAN transmit completion interrupt for each message • Message buffers #0-7 appointed as the transmission message buffers for transmission in order of the buffer number including programmable delay between messages (automatic block transmission function). • Transmit history list function
Remote frame processing	• Remote frame handling by message buffer defined for transmit
Time stamp functions	• A time stamp function can be set for receive messages. • Trigger for time stamp capture selectable (SOF or EOF detection in a CAN message frame) • A time stamp function can be set for transmit messages. • Specific bytes in the data field are replaced by the captured time stamp.
Diagnostics	• Readable error counters • “Valid protocol operation flag” for verification of bus connections • Receive-only mode • Single-shot mode • CAN protocol error type decoding • Self-test mode including self-reception
Power save modes	• CAN sleep mode (wakeup function using CAN bus enabled) • CAN stop mode (wakeup function using CAN bus disabled)

19.1.3 Configuration

The CAN controller is composed of the following four blocks.

(1) NPB interface

This functional block provides an NPB (NEC peripheral I/O bus) interface and a means of transmitting and receiving signals between the CAN and the host CPU.

(2) MAC (Memory Access Controller)

This functional block controls access to the CAN protocol layer and to the CAN RAM within the CAN module.

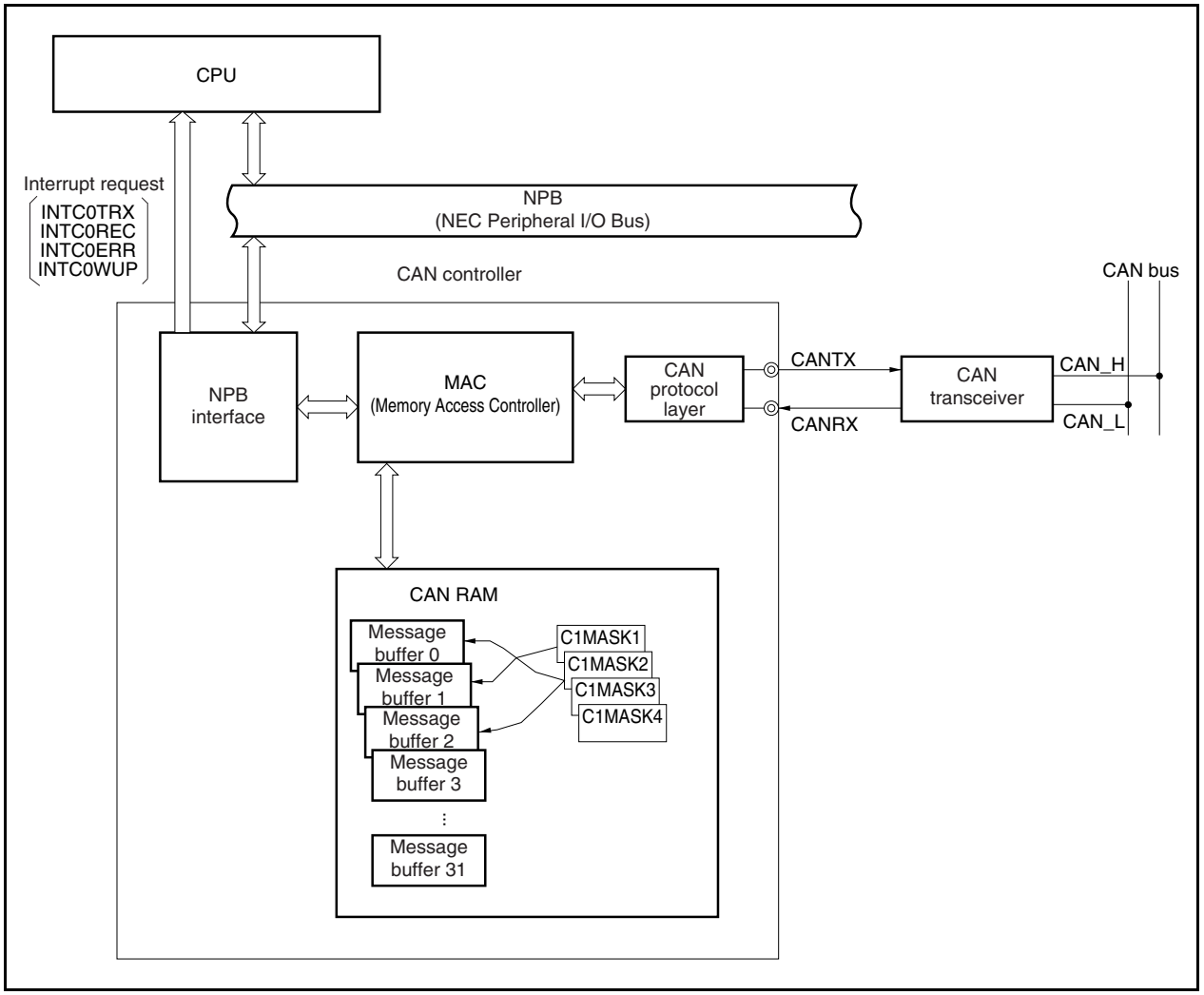
(3) CAN protocol layer

This functional block is involved in the operation of the CAN protocol 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 19-1. Block Diagram of CAN Module

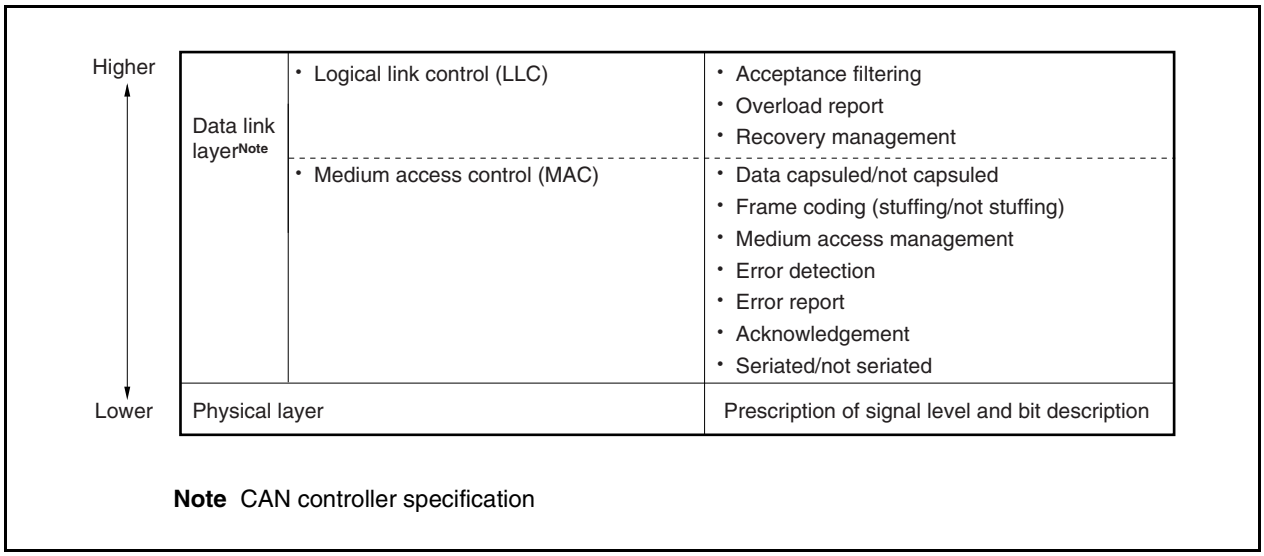


19.2 CAN Protocol

CAN (Controller Area Network) is a high-speed multiplex communication protocol for real-time communication in automotive applications (class C). CAN is prescribed by ISO 11898. For details, refer to the **ISO 11898** specifications.

The CAN specification is generally divided into two layers: a physical layer and a data link layer. In turn, the data link layer includes logical link control and medium access control. The composition of these layers is illustrated below.

Figure 19-2. Composition of Layers



Note CAN controller specification

19.2.1 Frame format

(1) Standard format frame

In this format, 2048 different identifiers can be set.

- The standard format frame uses 11-bit identifiers, which means that it can handle up to 2048 messages.

(2) Extended format frame

This format is used to set identifiers of approx. 5.3 million types.

- The extended format frame uses 29-bit (11 bits + 18 bits) identifiers which increase the number of messages that can be handled to 2048×2^{18} messages.
- Extended format frame is set when “recessive level” (CMOS level equals “1”) is set for both the SRR and IDE bits in the arbitration field.

19.2.2 Frame types

The following four types of frames are used in the CAN protocol.

Table 19-2. Frame Types

Frame Type	Description
Data frame	Frame used to transmit data
Remote frame	Frame used to request a data frame
Error frame	Frame used to report error detection
Overload frame	Frame used to delay the next data frame or remote frame

(1) Bus value

The bus values are divided into dominant and recessive.

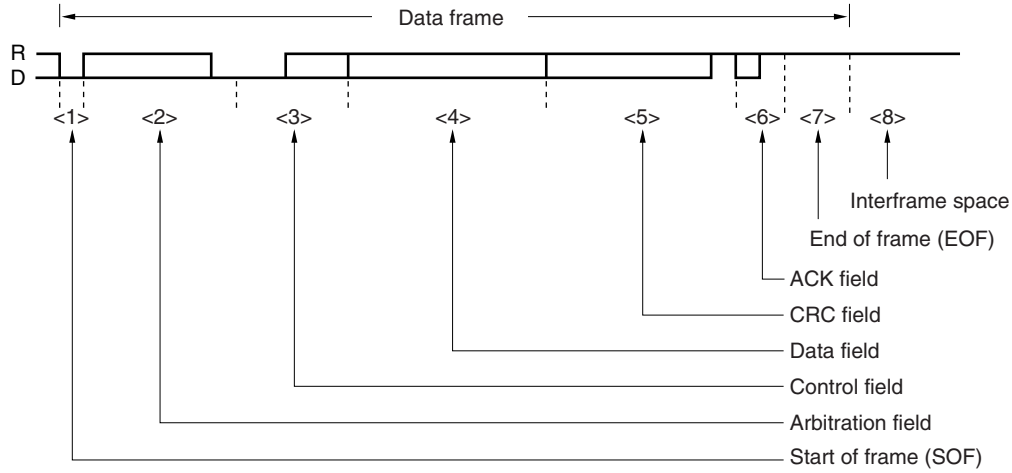
- Dominant level is indicated by logical 0.
- Recessive level is indicated by logical 1.
- When a dominant level and a recessive level are transmitted simultaneously, the bus value becomes dominant level.

19.2.3 Data frame and remote frame

(1) Data frame

A data frame is composed of seven fields.

Figure 19-3. Data Frame

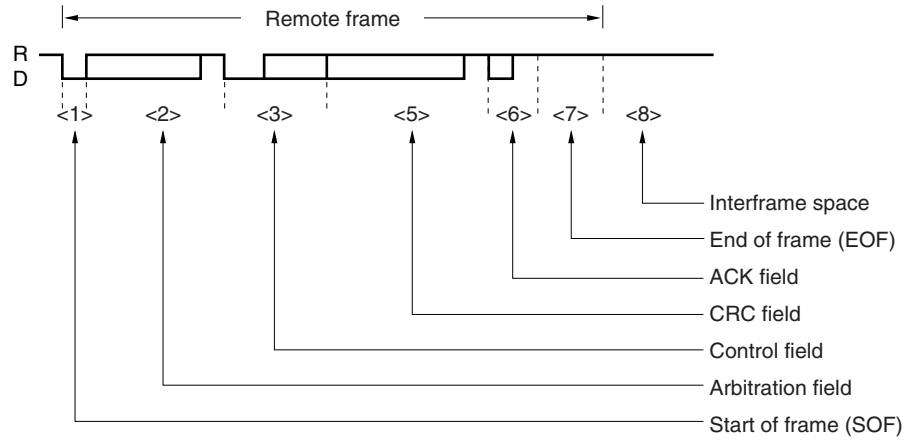


Remark D: Dominant = 0
R: Recessive = 1

(2) Remote frame

A remote frame is composed of six fields.

Figure 19-4. Remote Frame



Remarks 1. The data field is not transferred even if the control field's data length code is not "0000B".

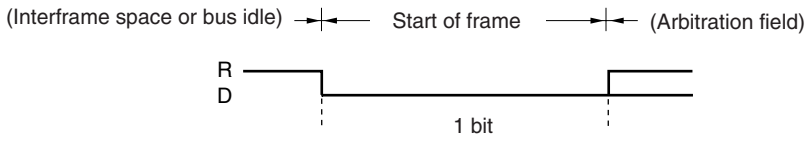
2. D: Dominant = 0
R: Recessive = 1

(3) Description of fields

<1> Start of frame (SOF)

The start of frame field is located at the start of a data frame or remote frame.

Figure 19-5. Start of Frame (SOF)



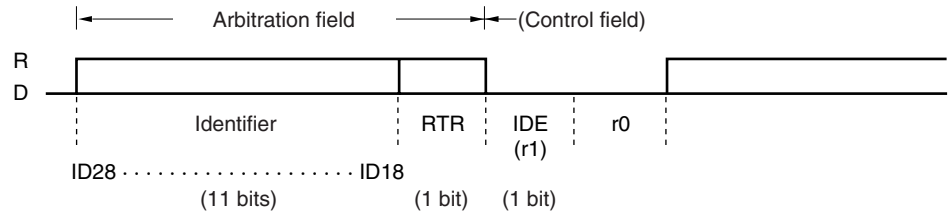
Remark D: Dominant = 0
R: Recessive = 1

- If dominant level is detected in the bus idle state, the start of frame is recognized.
- If recessive level is detected at the sample point of the start of frame, the preceding dominant level is judged as noise and the bus idle state is entered again.

<2> Arbitration field

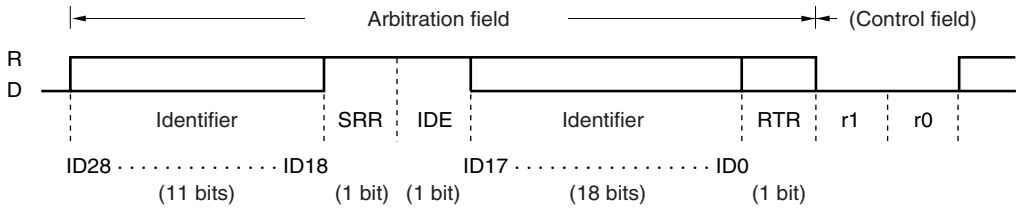
The arbitration field is used to evaluate the priority between data frames, remote frames, and frame formats (standard or extended identifier).

Figure 19-6. Arbitration Field (in Standard Format Mode)



Remark D: Dominant = 0
R: Recessive = 1

Figure 19-7. Arbitration Field (in Extended Format Mode)



Cautions 1. ID28 to ID0 are identifier bits.
2. Identifier bits are transferred in MSB-first order.

Remark D: Dominant = 0
R: Recessive = 1

Table 19-3. RTR Frame Settings

Frame Type	RTR Bit
Data frame	0 (D)
Remote frame	1 (R)

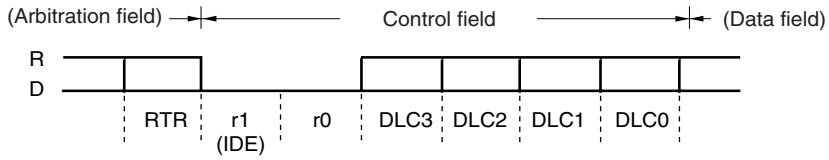
Table 19-4. Frame Format Setting (IDE Bit) and Number of Identifier (ID) Bits

Frame Format	SRR Bit	IDE Bit	No. of Bits
Standard format mode	None	0 (D)	11 bits
Extended format mode	1 (R)	1 (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).

Figure 19-8. Control Field



Remark D: Dominant = 0
R: Recessive = 1

In a standard format frame, the control field’s IDE bit is the same as the r1 bit.

Table 19-5. Data Length Code Settings

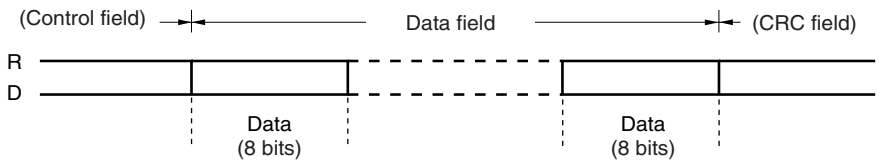
Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other than above				8 bytes regardless of the value of DLC3 to DLC0

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 (byte units) set by the control field. Up to 8 units of data can be set.

Figure 19-9. Data Field

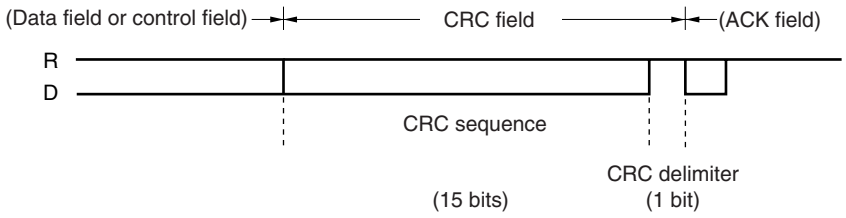


Remark D: Dominant = 0
R: Recessive = 1

<5> CRC field

The CRC field is a 16-bit field that is used to check for errors in transmit data.

Figure 19-10. CRC Field



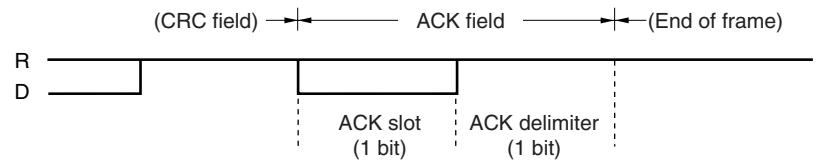
Remark D: Dominant = 0
R: Recessive = 1

- The polynomial P(X) used to generate the 15-bit CRC sequence is expressed as follows.
  $P(X) = X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$
- Transmitting node: The CRC sequence calculated from the data (before bit stuffing) in the start of frame, arbitration field, control field, and data field is transferred.
- 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 issues an error frame.

<6> ACK field

The ACK field is used to confirm normal reception.

Figure 19-11. ACK Field



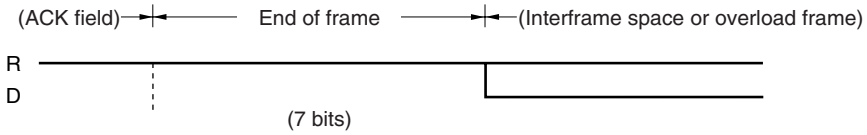
Remark D: Dominant = 0
R: Recessive = 1

- If no CRC error is detected, the receiving node sets the ACK slot to the dominant level.
- The transmitting node outputs two recessive-level bits.

<7> End of frame (EOF)

The end of frame field indicates the end of data frame/remote frame.

Figure 19-12. End of Frame (EOF)



Remark D: Dominant = 0
R: Recessive = 1

<8> Interframe space

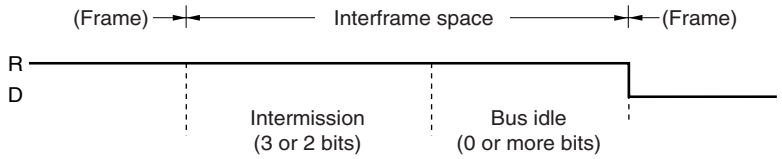
The interframe space is inserted after a data frame, remote frame, error frame, or overload frame to separate one frame from the next.

- The length of this field differs depending on the error status.

(a) Error active node

The error active node inserts a 3-bit intermission field before bus idle is encountered or another frame is transmitted.

Figure 19-13. Interframe Space (Error Active Node)



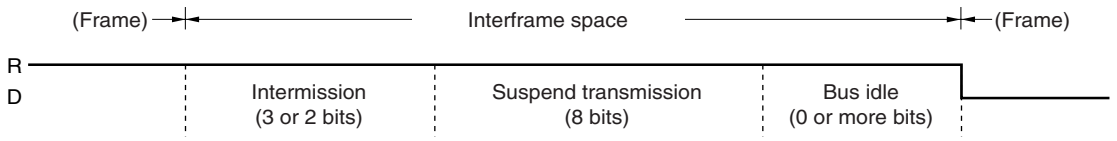
Remarks 1. Bus idle: State in which the bus is not used by any node.

2. D: Dominant = 0
R: Recessive = 1

(b) Error passive node

The error passive node inserts a 3-bit intermission field, followed by a suspend transmission field before bus idle is encountered or another frame is transmitted.

Figure 19-14. Interframe Space (Error Passive Node)



Remarks 1. Bus idle: State in which the bus is not used by any node.
Suspend transmission: Sequence of 8 recessive bits transmitted from the node in the error passive status after the intermission field.

2. D: Dominant = 0
R: Recessive = 1

- Operation in error status

Table 19-6. Operation in Error Status

Error Status	Operation
Error active	Any node in this state is able to start a transmission whenever the bus is idle.
Error passive	Any node in this state has to wait for 11 consecutive recessive bits before initiating a transmission.

- Operation when the third bit of the intermission field is dominant level

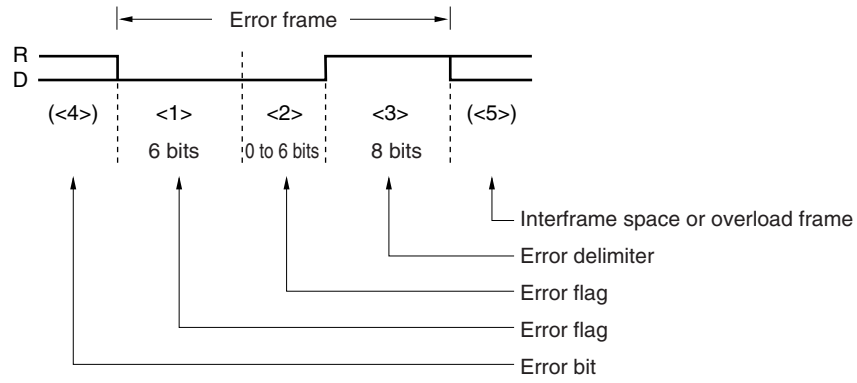
Table 19-7. Operation When Third Bit of Intermission Is Dominant Level

Error Status	Operation
No pending transmissions	A receive operation is performed when a start of frame output by another node is detected.
Pending transmission exists	The identifier is transmitted when a start of frame output by the local node is detected.

19.2.4 Error frame

- This frame is sent from a node if an error is detected.
- The type of error frame is defined by its error flag: an active error flag or passive error flag. Which kind of flag a node transmits after detecting an error condition depends on the internal count of the error counters of each node.

Figure 19-15. Error Frame



Remark D: Dominant = 0
R: Recessive = 1

Table 19-8. Definition of Each Field

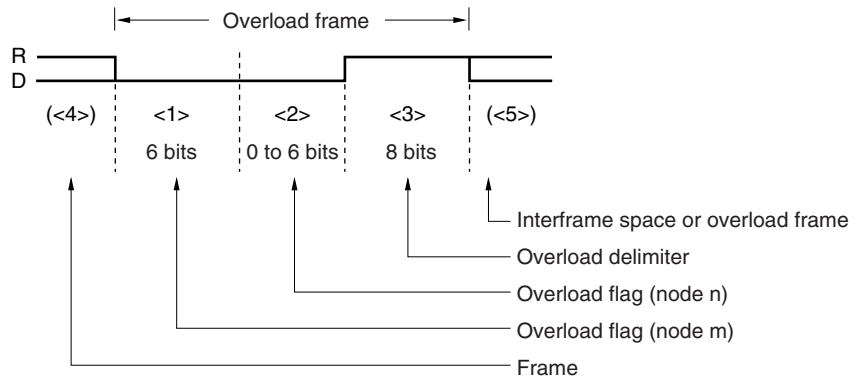
No.	Name	Bit Count	Definition
<1>	Error flag	6	Error active node: Sends 6 bits of dominant level continuously. Error passive node: Sends 6 bits of recessive level continuously.
<2>	Error flag	0 to 6	Nodes receiving an “error flag” detect bit stuff errors and issue error flags themselves.
<3>	Error delimiter	8	Sends 8 bits of recessive level continuously. In the case of monitoring the dominant level at the 8th bit, an overload frame is transmitted after the next bit.
<4>	Erroneous bit	–	An error frame is transmitted continuously after the bit where the error has occurred (in the case of a CRC error, transmission continues after the ACK delimiter).
<5>	Interframe space/overload frame	3/14 20 MAX.	Interframe space or overload frame continues.

19.2.5 Overload frame

An overload frame is transmitted under the following conditions.

- When the receiving node is not yet ready to receive.
- If a dominant level is detected at the first two bits in intermission mode.
- If a dominant level is detected at the last bit (8th bit) of the end of frame or at the last bit (8th bit) of the error delimiter/overload delimiter.

Figure 19-16. Overload Frame



Remarks 1. Node n/node m: Each node (n ≠ m)

2. D: Dominant = 0
R: Recessive = 1

Table 19-9. Field Definition of Overload Frame

No	Name	Bit Count	Definition
<1>	Overload flag starting from node m	6	Consecutive output of 6 dominant-level 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	Consecutive output of 8 recessive-level bits. If a dominant level is detected at 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	–	An interframe space or overload frame starts from here.

Remark Node n/node m: Each node (n ≠ m)

19.3 Functions

19.3.1 Arbitration

If two or more nodes happen to start transmission at the same time, the access conflict is solved by a bit-wise arbitration mechanism during transmission of the arbitration field.

(1) When a node starts transmission

During bus idle, the node having the output data can transmit.

(2) When more than one node starts transmission

- The node with the lower identifier wins the arbitration.
- The transmitting node compares its output arbitration field and the data level on the bus.
- It loses arbitration when it sends a recessive level and reads a dominant level from the bus.

Table 19-10. Arbitration

Level Detection	Status of Node in Arbitration
Conformity of level	Continuous transmission
Non-conformity of level	The data output is stopped from the next bit and reception operation starts

(3) Priority of data frame and remote frame

- When a data frame and remote frame with the same message identifier are on the bus, the data frame has priority because its RTR bit carries a ‘dominant level’. The data frame wins the arbitration.

19.3.2 Bit stuffing

When the same level continues for more than 5 bits, bit stuffing (inserting 1 bit with the inverse level) takes place.

- Due to this, resynchronization of the bit timing can be done at least every 10 bits.
- Nodes detecting an error condition send an error frame without implementing the bit stuffing rule, indicating this message to be erroneous for all nodes.

Table 19-11. Bit Stuffing

Transmission	During the transmission of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, 1 bit level with inverse-level data is inserted before the following bit.
Reception	During the reception of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, the reception is continued by deleting the next bit.

19.3.3 Multi masters

As the bus priority is determined by the identifier, any node can be the bus master.

19.3.4 Multi cast

Any message can be received by any node (broadcast).

19.3.5 Sleep mode/stop function

This is a function to put the CAN controller in waiting mode to achieve low power consumption. The sleep mode of the CAN complies with the method described in ISO 11898.

The CAN sleep mode can be woken up by bus operation, but the CAN stop mode is fully controlled by the CPU device.

19.3.6 Error control function

(1) Error types

Table 19-12. Error Types

Type	Description of Error		Detection State	
	Detection Method	Detection Condition	Transmission/ Reception	Field/Frame
Bit error	Comparison of output level and level on the bus (except stuff bit)	Mismatch of levels	Transmission/ reception node	Bit that output data on the bus at the start of frame to the end of frame, error frame and overload frame.
Stuff error	Check of the reception data at the stuff bit	6 consecutive bits of the same output level	Transmission/ reception node	Start of frame to CRC sequence
CRC error	Comparison of the CRC generated from the reception data and the received CRC sequence	Mismatch of CRC	Reception node	Start of frame to data field
Form error	Field/frame check of the fixed format	Detection of the fixed format error	Reception node	CRC delimiter ACK field End of frame Error frame Overload frame
ACK error	Check of the ACK slot by the transmission node	Detection of recessive level in ACK slot	Transmission node	ACK slot

(2) Output timing of error frame

Table 19-13. Output Timing of Error Frame

Type	Output Timing
Bit error, stuff error, form error, ACK error	The error frame is started at the next bit timing following the detected error
Error passive	The CRC error frame is started at the next bit timing following the ACK delimiter

(3) Measures when error occurs

- The transmission node re-transmits the data frame or the remote frame after the error frame.
- The CAN standard (ISO-11898) allows programmable suppression of this re-transmission. It is called single-shot mode.

(4) Error state

(a) Types of error state

- There are three types of error state: error active, error passive and bus off.
- The transmission error counter (TEC) and the reception error counter (REC) control the error state.
- The error counters are incremented on each error occurrence (see **Table 19-15**).
- If the value of an error counter exceeds 96, the warning level for the error passive state is reached.
- When only one node is active at startup, it may not receive an acknowledgment of a transmitted message.

  This will increment TEC until the error passive state is reached. The bus off state will not be reached because, for this specific condition, TEC will not increment any more if a value greater than 127 is reached.
- A node in the bus off state will not issue any dominant level at the CAN transmit pin. The reception of messages is not affected by the bus off state.

Table 19-14. Types of Error

Type	Operation	Value of Error Counter	Output Error Flag Type
Error active	Transmission/ reception	0 to 127	Active error flag (6 bits of dominant level continue)
Error passive	Transmission	128 to 255	Passive error flag (6 bits of recessive level continue)
	Reception	128 or more	
Bus off	Transmission	More than 255	Communication cannot be made
	Reception	–	Does not exist

(b) Error counter

- The error counter counts up when an error has occurred, and counts down upon successful transmission and reception. The error counters are updated at the first bit of an error flag.

Table 19-15. Error Counter

State	Transmission Error Counter (TEC)	Reception Error Counter (REC)
Reception node detects an error (except bit error in the active error flag or overload flag).	No change	+1
Reception node detects a dominant level following the error flag of its own error frame.	No change	+8
Transmission node transmits an error flag. Exception: 1. ACK error is detected in the error passive state and dominant level is not detected in the passive error flag sent. 2. Stuff error generation in arbitration field.	+8	No change
Bit error detection in active error flag and overload flag when transmitting node is in error active state.	+8	No change
Bit error detection in active error flag and overload flag when receiving node is in error active state.	No change	+8
When the node detects fourteen continuous dominant bits counted from the beginning of the active error flag or the overload flag, and every time, eight subsequent dominant bits after that are detected. Every time when the node detects eight continuous dominant bits after the passive error flag.	+8	+8
When the transmitting node has completed transmission without error.	–1 (0 when error counter = 0)	No change
When the reception node has completed reception without error.	No change	–1 ($1 \leq REC \leq 127$) 0 (REC = 0) 119 to 127 (REC > 127)

(c) Overload frame

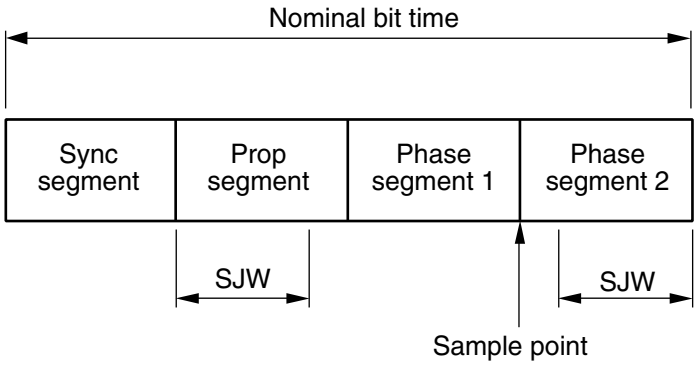
- If the recessive level of the first intermission bit is driven to the dominant level, an overload frame occurs on the bus. Upon detection of an overload frame, transmit requests will be postponed until the bus becomes idle.

19.3.7 Baud rate control function

(1) Nominal bit time (8 to 25 time quanta)

- The definition of 1 data bit time is as follows.

Figure 19-17. Nominal Bit Time (8 to 25 Time Quanta)



Remark Minimum time for one time/quantum (TQ) = 1/fx

- Sync segment: In this segment the bit synchronization is performed.
- Prop segment: This segment absorbs delays of the output buffer, the CAN bus and the input buffer. Prop segment time = (output buffer delay) + (CAN bus delay) + (input buffer delay).
- Phase segment 1/2: These segments compensate the data bit time error. The larger the size measured in TQ, the larger the tolerable error.
- The synchronization jump width (SJW) specifies the synchronization range. The SJW is programmable. SJW can have less or the same number of TQ as phase segment 2.

Table 19-16. Segment Name and Segment Length

Segment Name	Segment Length (Allowable Number of TQ)
Sync segment (Synchronization segment)	1
Prop segment (Propagation segment)	Programmable 1 to 8
Phase segment 1 (Phase buffer segment 1)	Programmable 1 to 8
Phase segment 2 (Phase buffer segment 2)	Maximum of phase segment 1 and the IPT[Note]
SJW	Programmable 1 to 4

Note IPT = Information Processing Time. It needs to be less than or equal to 2 TQ.

(2) Adjusting synchronization of the data bit

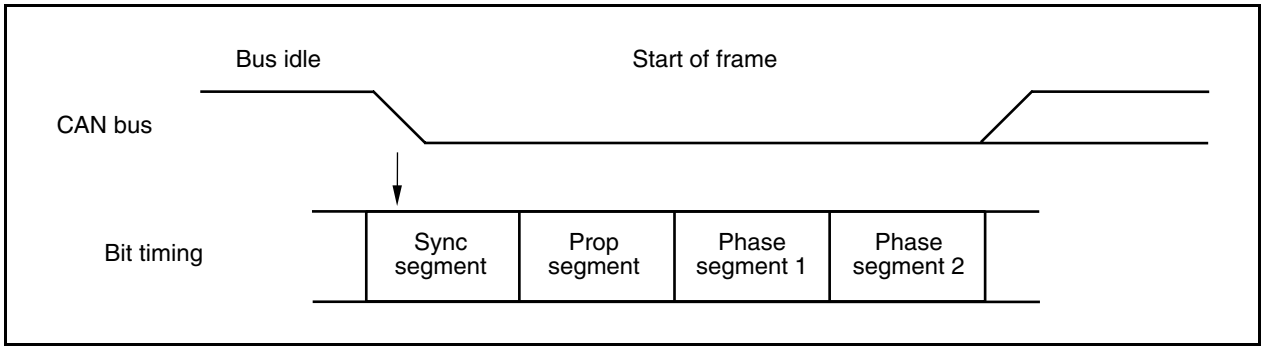
- The transmission node transmits data synchronized with the transmission node bit timing.
- The reception node adjusts synchronization at recessive to dominant edges on the bus.
  Depending on the protocol this synchronization can be a hard or soft synchronization.

(a) Hard synchronization

This type of synchronization is performed when the reception node detects a start of frame in the bus idle state.

- When the node detects the falling edge of an SOF, the current time quantum becomes the synchronization segment. The length of the following segments are defined by the values programmed into the SYNC0 and SYNC1 registers.

Figure 19-18. Adjusting Synchronization of Data Bit

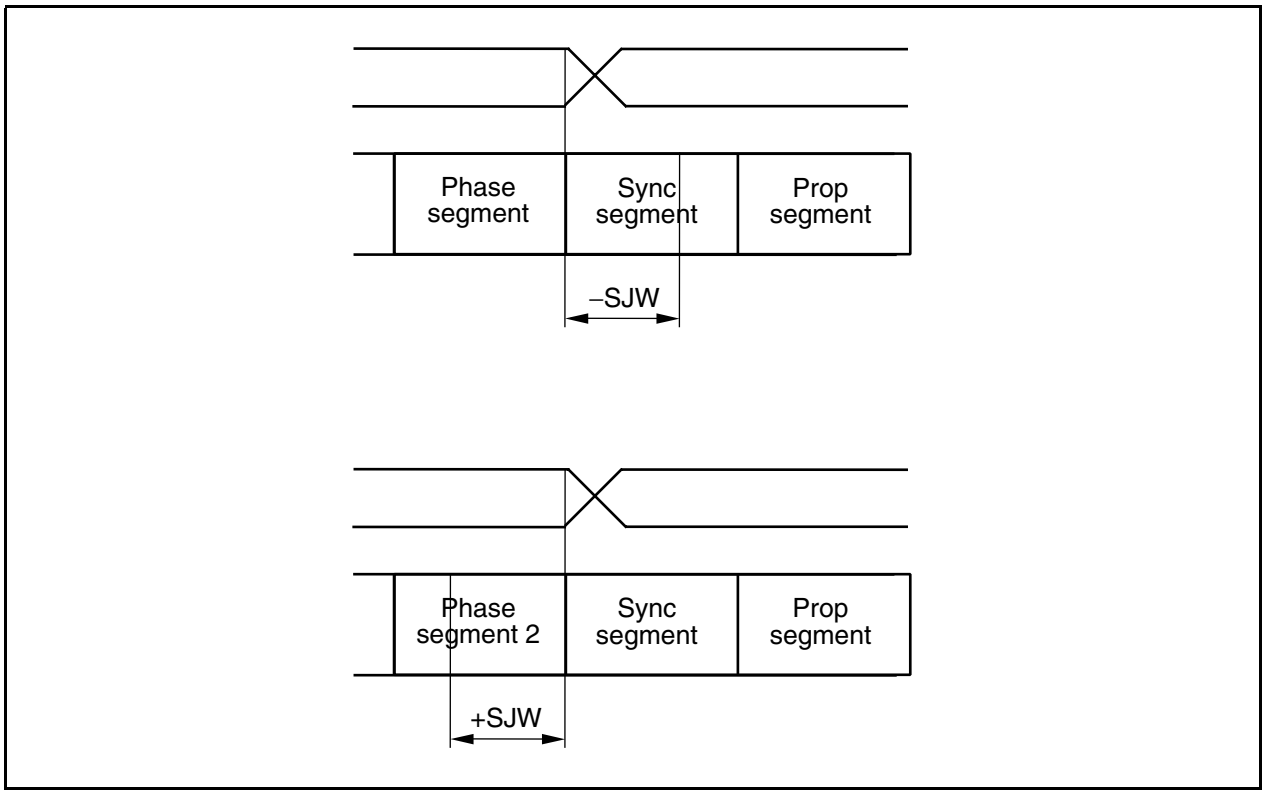


(b) Soft synchronization

When a recessive to dominant level change on the bus is detected, a soft synchronization is performed.

- If the phase error is larger than the programmed SJW value, the node will adjust the timing by applying this SJW value. Full synchronization is achieved by subsequent adjustments on the next recessive to dominant edge(s).
- Errors that are equal to or less than the programmed SJW are corrected instantly and full synchronization is achieved already for the next bit.
- The TQ at which the edge occurs becomes the sync segment forcibly if the phase error is less than or equal to SJW.

Figure 19-19. Bit Synchronization



19.3.8 State transition chart

Figure 19-20. Transmission State Transition Chart

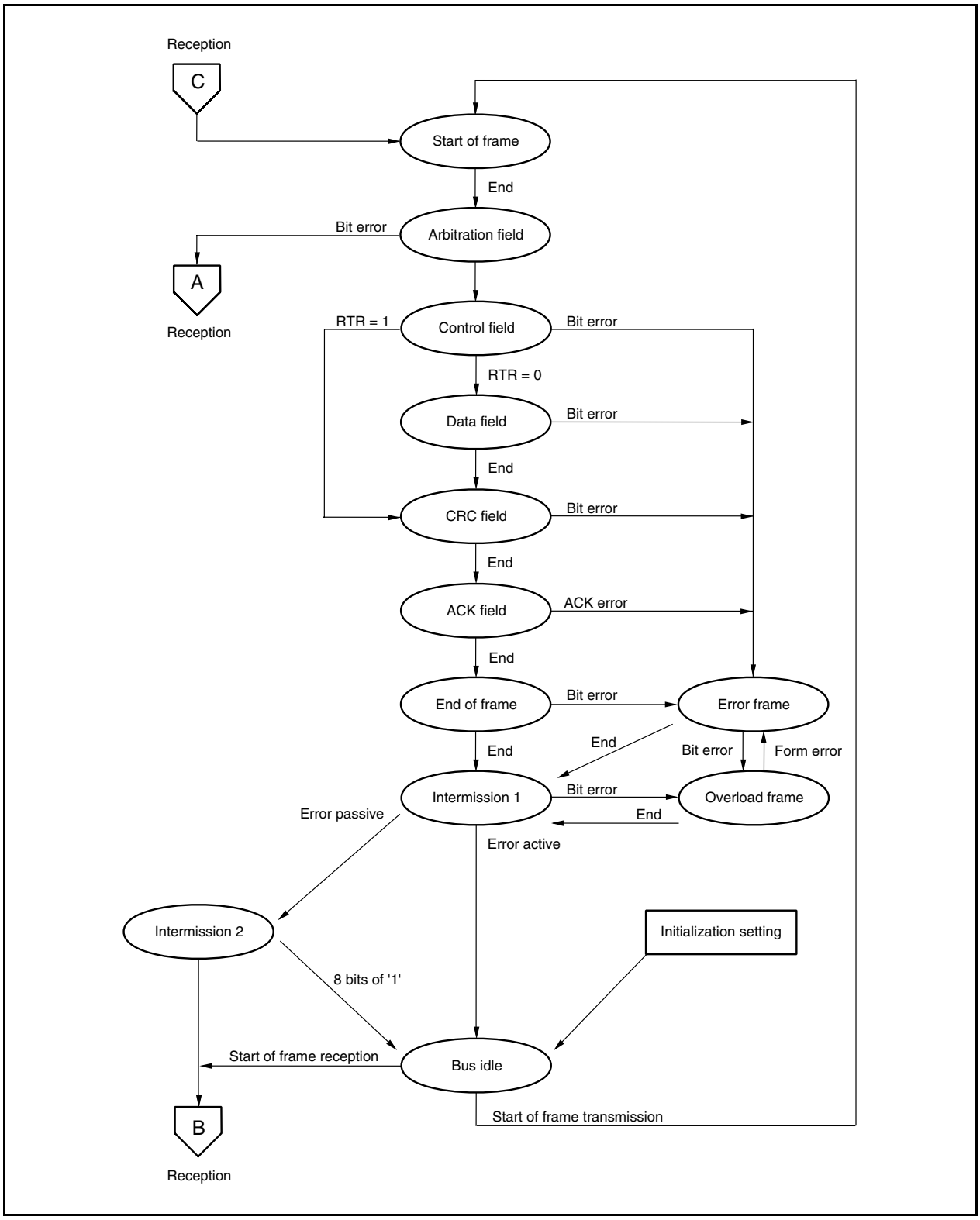


Figure 19-21. Reception State Transition Chart

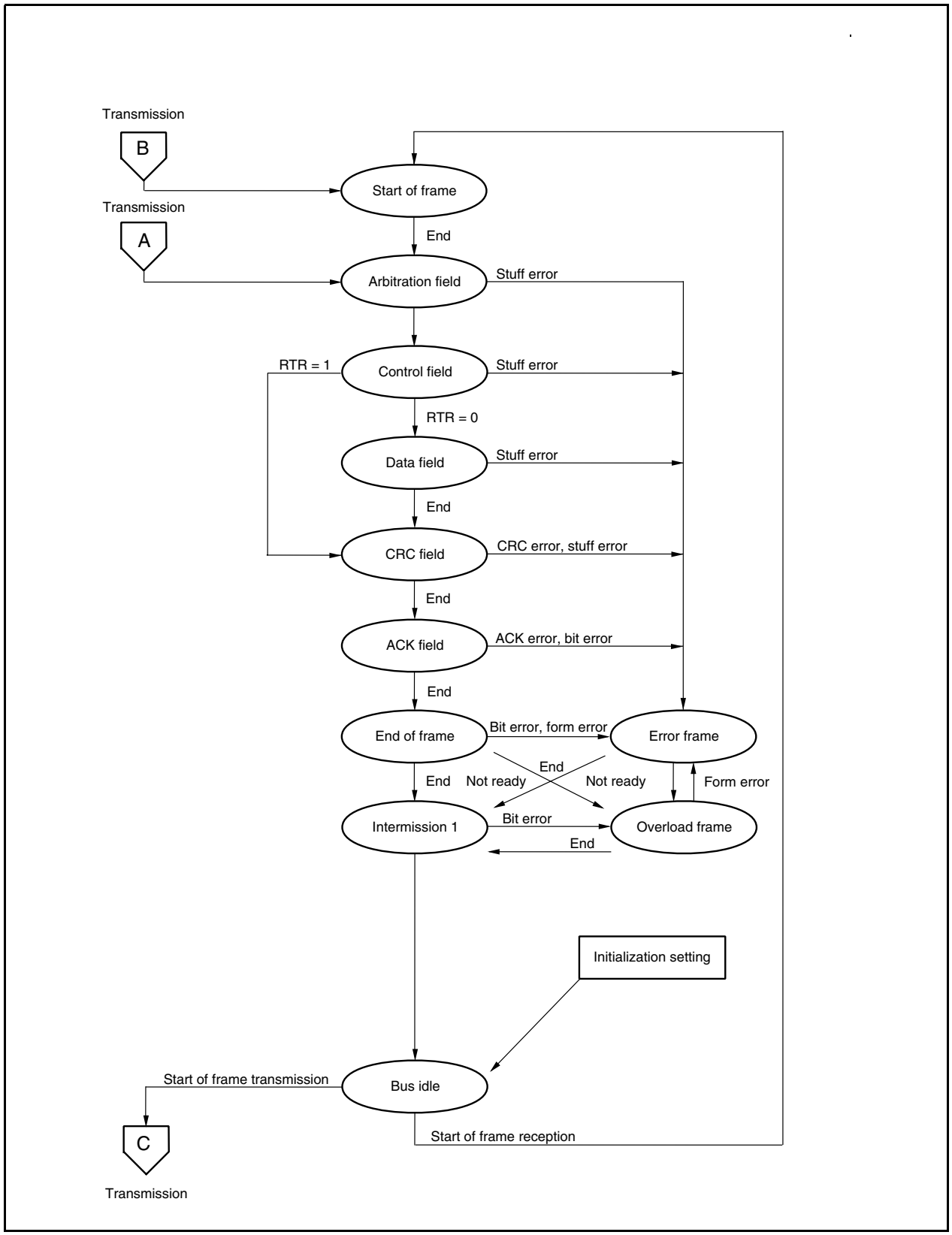
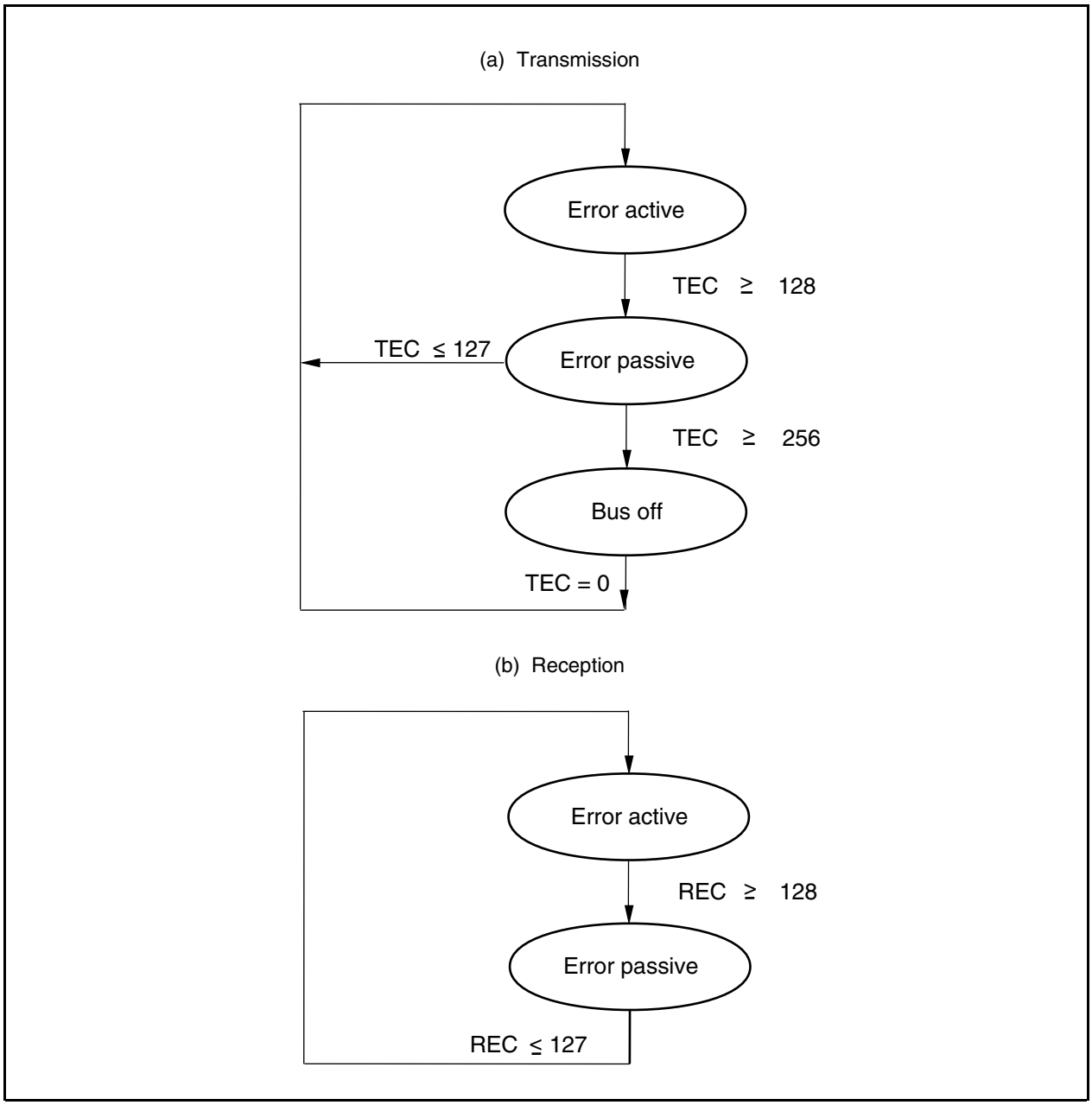


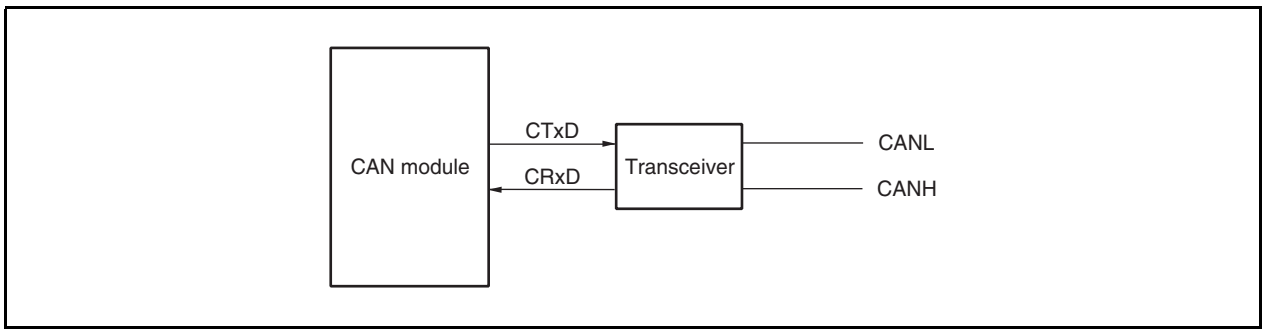
Figure 19-22. Error State Transition Chart



19.4 Connection with Target System

The CAN module has to be connected to the CAN bus using an external transceiver.

Figure 19-23. Connection to CAN Bus



19.5 Internal Registers of CAN Controller

19.5.1 CAN controller configuration

Table 19-17. List of CAN Controller Registers

Item	Configuration
Control registers	CAN0 module control register (C0GMCTRL)
	CAN0 module clock selection register (C0GMCS)
	CAN0 automatic block transmission register (C0GMABT)
	CAN0 automatic block transmission delay register (C0GMABTD)
	CAN0 module mask 1 registers (C0MASK1L, C0MASK1H)
	CAN0 module mask 2 registers (C0MASK2L, C0MASK2H)
	CAN0 module mask 3 registers (C0MASK3L, C0MASK3H)
	CAN0 module mask 4 registers (C0MASK4L, C0MASK4H)
	CAN0 module control register (C0CTRL)
	CAN0 module last error code register (C0LEC)
	CAN0 module information register (C0INFO)
	CAN0 module error counter register (C0ERC)
	CAN0 module interrupt enable register (C0IE)
	CAN0 module interrupt status register (C0INTS)
	CAN0 module bit-rate prescaler register (C0BRP)
	CAN0 module bit rate register (C0BTR)
	CAN0 module last In-pointer register (C0LIPT)
	CAN0 module receive history list register (C0RGPT)
	CAN0 module last out-pointer register (C0LOPT)
	CAN0 module transmit history list register (C0TGPT)
	CAN0 module time stamp register (C0TS)
Message buffer registers	CAN0 Message data byte 01 register m (C0MDATA01m)
	CAN0 Message data byte 0 register m (C0MDATA0m)
	CAN0 Message data byte 1 register m (C0MDATA1m)
	CAN0 Message data byte 23 register m (C0MDATA23m)
	CAN0 Message data byte 2 register m (C0MDATA2m)
	CAN0 Message data byte 3 register m (C0MDATA3m)
	CAN0 Message data byte 45 register m (C0MDATA45m)
	CAN0 Message data byte 4 register m (C0MDATA4m)
	CAN0 Message data byte 5 register m (C0MDATA5m)
	CAN0 Message data byte 67 register m (C0MDATA67m)
	CAN0 Message data byte 6 register m (C0MDATA6m)
	CAN0 Message data byte 7 register m (C0MDATA7m)
	CAN0 Message data length code register m (C0MDLCm)
	CAN0 Message register m (C0MCONFm)
	CAN0 Message Identifier register m (C0MIDLm, C0MIDHm)
	CAN0 Message control register m (C0MCTRLm)

Remark m: Number of buffer (0 to 31)

19.5.2 Register access type

Table 19-18. Control Register Access Types

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC000H	CAN0 module control register	C0GMCTRL	R/W			√	0000H
FFFEC002H	CAN0 module clock selection register	C0GMCS			√		0FH
FFFEC006H	CAN0 automatic block transmission register	C0GMABT				√	0000H
FFFEC008H	CAN0 automatic block transmission delay register	C0GMABTD			√		00H
FFFEC040H	CAN0 module mask 1 register	C0MASK1L				√	Undefined
FFFEC042H		C0MASK1H				√	Undefined
FFFEC044H	CAN0 module mask 2 register	C0MASK2L				√	Undefined
FFFEC046H		C0MASK2H				√	Undefined
FFFEC048H	CAN0 module mask 3 register	C0MASK3L				√	Undefined
FFFEC04AH		C0MASK3H				√	Undefined
FFFEC04CH	CAN0 module mask 4 register	C0MASK4L				√	Undefined
FFFEC04EH		C0MASK4H				√	Undefined
FFFEC050H	CAN0 module control register	C0CTRL				√	0000H
FFFEC052H	CAN0 module last error code register	C0LEC			√		00H
FFFEC053H	CAN0 module information register	C0INFO	R		√		00H
FFFEC054H	CAN0 module error counter register	C0ERC				√	0000H
FFFEC056H	CAN0 module interrupt enable register	C0IE	R/W			√	0000H
FFFEC058H	CAN0 module interrupt status register	C0INTS				√	0000H
FFFEC05AH	CAN0 module bit-rate prescaler register	C0BRP			√		FFH
FFFEC05CH	CAN0 module bit-rate register	C0BTR				√	370FH
FFFEC05EH	CAN0 module last in-pointer register	C0LIPT	R		√		Undefined
FFFEC060H	CAN0 module receive history list register	C0RGPT	R/W			√	xxxH, xx10B
FFFEC062H	CAN0 module last out-pointer register	C0LOPT	R		√		Undefined
FFFEC064H	CAN0 module transmit history list register	C0TGPT	R/W			√	xxxH, xx10B
FFFEC066H	CAN0 module time stamp register	C0TS				√	0000H

Table 19-19. Message Buffer Register Access Types

(1/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC100H	CAN0 message data byte 01 register 00	C0MDATA0100	R/W			√	Undefined
FFFEC100H	CAN0 message data byte 0 register 00	C0MDATA000			√		Undefined
FFFEC101H	CAN0 message data byte 1 register 00	C0MDATA100			√		Undefined
FFFEC102H	CAN0 message data byte 23 register 00	C0MDATA2300				√	Undefined
FFFEC102H	CAN0 message data byte 2 register 00	C0MDATA200			√		Undefined
FFFEC103H	CAN0 message data byte 3 register 00	C0MDATA300			√		Undefined
FFFEC104H	CAN0 message data byte 45 register 00	C0MDATA4500				√	Undefined
FFFEC104H	CAN0 message data byte 4 register 00	C0MDATA400			√		Undefined
FFFEC105H	CAN0 message data byte 5 register 00	C0MDATA500			√		Undefined
FFFEC106H	CAN0 message data byte 67 register 00	C0MDATA6700				√	Undefined
FFFEC106H	CAN0 message data byte 6 register 00	C0MDATA600			√		Undefined
FFFEC107H	CAN0 message data byte 7 register 00	C0MDATA700			√		Undefined
FFFEC108H	CAN0 message data length code register 00	C0MDLC00			√		0000xxxx
FFFEC109H	CAN0 message configuration register 00	C0MCONF00			√		Undefined
FFFEC10AH	CAN0 message identifier register 00	C0MIDL00				√	Undefined
FFFEC10CH		C0MIDH00				√	Undefined
FFFEC10EH	CAN0 message control register 00	C0MCTRL00				√	00x00000 000xx000B
FFFEC120H	CAN0 message data byte 01 register 01	C0MDATA0101				√	Undefined
FFFEC120H	CAN0 message data byte 0 register 01	C0MDATA001			√		Undefined
FFFEC121H	CAN0 message data byte 1 register 01	C0MDATA101			√		Undefined
FFFEC122H	CAN0 message data byte 23 register 01	C0MDATA2301				√	Undefined
FFFEC122H	CAN0 message data byte 2 register 01	C0MDATA201			√		Undefined
FFFEC123H	CAN0 message data byte 3 register 01	C0MDATA301			√		Undefined
FFFEC124H	CAN0 message data byte 45 register 01	C0MDATA4501				√	Undefined
FFFEC124H	CAN0 message data byte 4 register 01	C0MDATA401			√		Undefined
FFFEC125H	CAN0 message data byte 5 register 01	C0MDATA501			√		Undefined
FFFEC126H	CAN0 message data byte 67 register 01	C0MDATA6701				√	Undefined
FFFEC126H	CAN0 message data byte 6 register 01	C0MDATA601			√		Undefined
FFFEC127H	CAN0 message data byte 7 register 01	C0MDATA701			√		Undefined
FFFEC128H	CAN0 message data length code register 01	C0MDLC01			√		0000xxxx
FFFEC129H	CAN0 message configuration register 01	C0MCONF01			√		Undefined
FFFEC12AH	CAN0 message identifier register 01	C0MIDL01				√	Undefined
FFFEC12CH		C0MIDH01				√	Undefined
FFFEC12EH	CAN0 message control register 01	C0MCTRL01				√	00x00000 000xx000B

(2/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC140H	CAN0 message data byte 01 register 02	C0MDATA0102	R/W			√	Undefined
FFFEC140H	CAN0 message data byte 0 register 02	C0MDATA002			√		Undefined
FFFEC141H	CAN0 message data byte 1 register 02	C0MDATA102			√		Undefined
FFFEC142H	CAN0 message data byte 23 register 02	C0MDATA2302				√	Undefined
FFFEC142H	CAN0 message data byte 2 register 02	C0MDATA202			√		Undefined
FFFEC143H	CAN0 message data byte 3 register 02	C0MDATA302			√		Undefined
FFFEC144H	CAN0 message data byte 45 register 02	C0MDATA4502				√	Undefined
FFFEC144H	CAN0 message data byte 4 register 02	C0MDATA402			√		Undefined
FFFEC145H	CAN0 message data byte 5 register 02	C0MDATA502			√		Undefined
FFFEC146H	CAN0 message data byte 67 register 02	C0MDATA6702				√	Undefined
FFFEC146H	CAN0 message data byte 6 register 02	C0MDATA602			√		Undefined
FFFEC147H	CAN0 message data byte 7 register 02	C0MDATA702			√		Undefined
FFFEC148H	CAN0 message data length code register 02	C0MDLC02			√		0000xxxx
FFFEC149H	CAN0 message configuration register 02	C0MCONF02			√		Undefined
FFFEC14AH	CAN0 message identifier register 02	C0MIDL02				√	Undefined
FFFEC14CH		C0MIDH02				√	Undefined
FFFEC14EH	CAN0 message control register 02	C0MCTRL02				√	00x00000 000xx000B
FFFEC160H	CAN0 message data byte 01 register 03	C0MDATA0103				√	Undefined
FFFEC160H	CAN0 message data byte 0 register 03	C0MDATA003			√		Undefined
FFFEC161H	CAN0 message data byte 1 register 03	C0MDATA103			√		Undefined
FFFEC162H	CAN0 message data byte 23 register 03	C0MDATA2303				√	Undefined
FFFEC162H	CAN0 message data byte 2 register 03	C0MDATA203			√		Undefined
FFFEC163H	CAN0 message data byte 3 register 03	C0MDATA303			√		Undefined
FFFEC164H	CAN0 message data byte 45 register 03	C0MDATA4503				√	Undefined
FFFEC164H	CAN0 message data byte 4 register 03	C0MDATA403			√		Undefined
FFFEC165H	CAN0 message data byte 5 register 03	C0MDATA503			√		Undefined
FFFEC166H	CAN0 message data byte 67 register 03	C0MDATA6703				√	Undefined
FFFEC166H	CAN0 message data byte 6 register 03	C0MDATA603			√		Undefined
FFFEC167H	CAN0 message data byte 7 register 03	C0MDATA703			√		Undefined
FFFEC168H	CAN0 message data length code register 03	C0MDLC03			√		0000xxxx
FFFEC169H	CAN0 message configuration register 03	C0MCONF03			√		Undefined
FFFEC16AH	CAN0 message identifier register 03	C0MIDL03				√	Undefined
FFFEC16CH		C0MIDH03				√	Undefined
FFFEC16EH	CAN0 message control register 03	C0MCTRL03				√	00x00000 000xx000B

(3/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC180H	CAN0 message data byte 01 register 04	C0MDATA0104	R/W			√	Undefined
FFFEC180H	CAN0 message data byte 0 register 04	C0MDATA004			√		Undefined
FFFEC181H	CAN0 message data byte 1 register 04	C0MDATA104			√		Undefined
FFFEC182H	CAN0 message data byte 23 register 04	C0MDATA2304				√	Undefined
FFFEC182H	CAN0 message data byte 2 register 04	C0MDATA204			√		Undefined
FFFEC183H	CAN0 message data byte 3 register 04	C0MDATA304			√		Undefined
FFFEC184H	CAN0 message data byte 45 register 04	C0MDATA4504				√	Undefined
FFFEC184H	CAN0 message data byte 4 register 04	C0MDATA404			√		Undefined
FFFEC185H	CAN0 message data byte 5 register 04	C0MDATA504			√		Undefined
FFFEC186H	CAN0 message data byte 67 register 04	C0MDATA6704				√	Undefined
FFFEC186H	CAN0 message data byte 6 register 04	C0MDATA604			√		Undefined
FFFEC187H	CAN0 message data byte 7 register 04	C0MDATA704			√		Undefined
FFFEC188H	CAN0 message data length code register 04	C0MDLC04			√		0000xxxx
FFFEC189H	CAN0 message configuration register 04	C0MCONF04			√		Undefined
FFFEC18AH	CAN0 message identifier register 04	C0MIDL04				√	Undefined
FFFEC18CH		C0MIDH04				√	Undefined
FFFEC18EH	CAN0 message control register 04	C0MCTRL04				√	00x00000 000xx000B
FFFEC1A0H	CAN0 message data byte 01 register 05	C0MDATA0105				√	Undefined
FFFEC1A0H	CAN0 message data byte 0 register 05	C0MDATA005			√		Undefined
FFFEC1A1H	CAN0 message data byte 1 register 05	C0MDATA105			√		Undefined
FFFEC1A2H	CAN0 message data byte 23 register 05	C0MDATA2305				√	Undefined
FFFEC1A2H	CAN0 message data byte 2 register 05	C0MDATA205			√		Undefined
FFFEC1A3H	CAN0 message data byte 3 register 05	C0MDATA305			√		Undefined
FFFEC1A4H	CAN0 message data byte 45 register 05	C0MDATA4505				√	Undefined
FFFEC1A4H	CAN0 message data byte 4 register 05	C0MDATA405			√		Undefined
FFFEC1A5H	CAN0 message data byte 5 register 05	C0MDATA505			√		Undefined
FFFEC1A6H	CAN0 message data byte 67 register 05	C0MDATA6705				√	Undefined
FFFEC1A6H	CAN0 message data byte 6 register 05	C0MDATA605			√		Undefined
FFFEC1A7H	CAN0 message data byte 7 register 05	C0MDATA705			√		Undefined
FFFEC1A8H	CAN0 message data length code register 05	C0MDLC05			√		0000xxxx
FFFEC1A9H	CAN0 message configuration register 05	C0MCONF05			√		Undefined
FFFEC1AAH	CAN0 message identifier register 05	C0MIDL05				√	Undefined
FFFEC1ACH		C0MIDH05				√	Undefined
FFFEC1AEH	CAN0 message control register 05	C0MCTRL05				√	00x00000 000xx000B

(4/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC1C0H	CAN0 message data byte 01 register 06	C0MDATA0106	R/W			√	Undefined
FFFEC1C0H	CAN0 message data byte 0 register 06	C0MDATA006			√		Undefined
FFFEC1C1H	CAN0 message data byte 1 register 06	C0MDATA106			√		Undefined
FFFEC1C2H	CAN0 message data byte 23 register 06	C0MDATA2306				√	Undefined
FFFEC1C2H	CAN0 message data byte 2 register 06	C0MDATA206			√		Undefined
FFFEC1C3H	CAN0 message data byte 3 register 06	C0MDATA306			√		Undefined
FFFEC1C4H	CAN0 message data byte 45 register 06	C0MDATA4506				√	Undefined
FFFEC1C4H	CAN0 message data byte 4 register 06	C0MDATA406			√		Undefined
FFFEC1C5H	CAN0 message data byte 5 register 06	C0MDATA506			√		Undefined
FFFEC1C6H	CAN0 message data byte 67 register 06	C0MDATA6706				√	Undefined
FFFEC1C6H	CAN0 message data byte 6 register 06	C0MDATA606			√		Undefined
FFFEC1C7H	CAN0 message data byte 7 register 06	C0MDATA706			√		Undefined
FFFEC1C8H	CAN0 message data length code register 06	C0MDLC06			√		0000xxxx
FFFEC1C9H	CAN0 message configuration register 06	C0MCONF06			√		Undefined
FFFEC1CAH	CAN0 message identifier register 06	C0MIDL06				√	Undefined
FFFEC1CCH		C0MIDH06				√	Undefined
FFFEC1CEH	CAN0 message control register 06	C0MCTRL06				√	00x00000 000xx000B
FFFEC1E0H	CAN0 message data byte 01 register 07	C0MDATA0107				√	Undefined
FFFEC1E0H	CAN0 message data byte 0 register 07	C0MDATA007			√		Undefined
FFFEC1E1H	CAN0 message data byte 1 register 07	C0MDATA107			√		Undefined
FFFEC1E2H	CAN0 message data byte 23 register 07	C0MDATA2307				√	Undefined
FFFEC1E2H	CAN0 message data byte 2 register 07	C0MDATA207			√		Undefined
FFFEC1E3H	CAN0 message data byte 3 register 07	C0MDATA307			√		Undefined
FFFEC1E4H	CAN0 message data byte 45 register 07	C0MDATA4507				√	Undefined
FFFEC1E4H	CAN0 message data byte 4 register 07	C0MDATA407			√		Undefined
FFFEC1E5H	CAN0 message data byte 5 register 07	C0MDATA507			√		Undefined
FFFEC1E6H	CAN0 message data byte 67 register 07	C0MDATA6707				√	Undefined
FFFEC1E6H	CAN0 message data byte 6 register 07	C0MDATA607			√		Undefined
FFFEC1E7H	CAN0 message data byte 7 register 07	C0MDATA707			√		Undefined
FFFEC1E8H	CAN0 message data length code register 07	C0MDLC07			√		0000xxxx
FFFEC1E9H	CAN0 message configuration register 07	C0MCONF07			√		Undefined
FFFEC1EAH	CAN0 message identifier register 07	C0MIDL07				√	Undefined
FFFEC1ECH		C0MIDH07				√	Undefined
FFFEC1EEH	CAN0 message control register 07	C0MCTRL07				√	00x00000 000xx000B

(5/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC200H	CAN0 message data byte 01 register 08	C0MDATA0108	R/W			√	Undefined
FFFEC200H	CAN0 message data byte 0 register 08	C0MDATA008			√		Undefined
FFFEC201H	CAN0 message data byte 1 register 08	C0MDATA108			√		Undefined
FFFEC202H	CAN0 message data byte 23 register 08	C0MDATA2308				√	Undefined
FFFEC202H	CAN0 message data byte 2 register 08	C0MDATA208			√		Undefined
FFFEC203H	CAN0 message data byte 3 register 08	C0MDATA308			√		Undefined
FFFEC204H	CAN0 message data byte 45 register 08	C0MDATA4508				√	Undefined
FFFEC204H	CAN0 message data byte 4 register 08	C0MDATA408			√		Undefined
FFFEC205H	CAN0 message data byte 5 register 08	C0MDATA508			√		Undefined
FFFEC206H	CAN0 message data byte 67 register 08	C0MDATA6708				√	Undefined
FFFEC206H	CAN0 message data byte 6 register 08	C0MDATA608			√		Undefined
FFFEC207H	CAN0 message data byte 7 register 08	C0MDATA708			√		Undefined
FFFEC208H	CAN0 message data length code register 08	C0MDLC08			√		0000xxxx
FFFEC209H	CAN0 message configuration register 08	C0MCONF08			√		Undefined
FFFEC20AH	CAN0 message identifier register 08	C0MIDL08				√	Undefined
FFFEC20CH		C0MIDH08				√	Undefined
FFFEC20EH	CAN0 message control register 08	C0MCTRL08				√	00x00000 000xx000B
FFFEC220H	CAN0 message data byte 01 register 09	C0MDATA0109				√	Undefined
FFFEC220H	CAN0 message data byte 0 register 09	C0MDATA009			√		Undefined
FFFEC221H	CAN0 message data byte 1 register 09	C0MDATA109			√		Undefined
FFFEC222H	CAN0 message data byte 23 register 09	C0MDATA2309				√	Undefined
FFFEC222H	CAN0 message data byte 2 register 09	C0MDATA209			√		Undefined
FFFEC223H	CAN0 message data byte 3 register 09	C0MDATA309			√		Undefined
FFFEC224H	CAN0 message data byte 45 register 09	C0MDATA4509				√	Undefined
FFFEC224H	CAN0 message data byte 4 register 09	C0MDATA409			√		Undefined
FFFEC225H	CAN0 message data byte 5 register 09	C0MDATA509			√		Undefined
FFFEC226H	CAN0 message data byte 67 register 09	C0MDATA6709				√	Undefined
FFFEC226H	CAN0 message data byte 6 register 09	C0MDATA609			√		Undefined
FFFEC227H	CAN0 message data byte 7 register 09	C0MDATA709			√		Undefined
FFFEC228H	CAN0 message data length code register 09	C0MDLC09			√		0000xxxx
FFFEC229H	CAN0 message configuration register 09	C0MCONF09			√		Undefined
FFFEC22AH	CAN0 message identifier register 09	C0MIDL09				√	Undefined
FFFEC22CH		C0MIDH09				√	Undefined
FFFEC22EH	CAN0 message control register 09	C0MCTRL09				√	00x00000 000xx000B

(6/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC240H	CAN0 message data byte 01 register 10	C0MDATA0110	R/W			√	Undefined
FFFEC240H	CAN0 message data byte 0 register 10	C0MDATA010			√		Undefined
FFFEC241H	CAN0 message data byte 1 register 10	C0MDATA110			√		Undefined
FFFEC242H	CAN0 message data byte 23 register 10	C0MDATA2310				√	Undefined
FFFEC242H	CAN0 message data byte 2 register 10	C0MDATA210			√		Undefined
FFFEC243H	CAN0 message data byte 3 register 10	C0MDATA310			√		Undefined
FFFEC244H	CAN0 message data byte 45 register 10	C0MDATA4510				√	Undefined
FFFEC244H	CAN0 message data byte 4 register 10	C0MDATA410			√		Undefined
FFFEC245H	CAN0 message data byte 5 register 10	C0MDATA510			√		Undefined
FFFEC246H	CAN0 message data byte 67 register 10	C0MDATA6710				√	Undefined
FFFEC246H	CAN0 message data byte 6 register 10	C0MDATA610			√		Undefined
FFFEC247H	CAN0 message data byte 7 register 10	C0MDATA710			√		Undefined
FFFEC248H	CAN0 message data length code register 10	C0MDLC10			√		0000xxxx
FFFEC249H	CAN0 message configuration register 10	C0MCONF10			√		Undefined
FFFEC24AH	CAN0 message identifier register 10	C0MIDL10				√	Undefined
FFFEC24CH		C0MIDH10				√	Undefined
FFFEC24EH	CAN0 message control register 10	C0MCTRL10				√	00x00000 000xx000B
FFFEC260H	CAN0 message data byte 01 register 11	C0MDATA0111				√	Undefined
FFFEC260H	CAN0 message data byte 0 register 11	C0MDATA011			√		Undefined
FFFEC261H	CAN0 message data byte 1 register 11	C0MDATA111			√		Undefined
FFFEC262H	CAN0 message data byte 23 register 11	C0MDATA2311				√	Undefined
FFFEC262H	CAN0 message data byte 2 register 11	C0MDATA211			√		Undefined
FFFEC263H	CAN0 message data byte 3 register 11	C0MDATA311			√		Undefined
FFFEC264H	CAN0 message data byte 45 register 11	C0MDATA4511				√	Undefined
FFFEC264H	CAN0 message data byte 4 register 11	C0MDATA411			√		Undefined
FFFEC265H	CAN0 message data byte 5 register 11	C0MDATA511			√		Undefined
FFFEC266H	CAN0 message data byte 67 register 11	C0MDATA6711				√	Undefined
FFFEC266H	CAN0 message data byte 6 register 11	C0MDATA611			√		Undefined
FFFEC267H	CAN0 message data byte 7 register 11	C0MDATA711			√		Undefined
FFFEC268H	CAN0 message data length code register 11	C0MDLC11			√		0000xxxx
FFFEC269H	CAN0 message configuration register 11	C0MCONF11			√		Undefined
FFFEC26AH	CAN0 message identifier register 11	C0MIDL11				√	Undefined
FFFEC26CH		C0MIDH11				√	Undefined
FFFEC26EH	CAN0 message control register 11	C0MCTRL11				√	00x00000 000xx000B

(7/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC280H	CAN0 message data byte 01 register 12	C0MDATA0112	R/W			√	Undefined
FFFEC280H	CAN0 message data byte 0 register 12	C0MDATA012			√		Undefined
FFFEC281H	CAN0 message data byte 1 register 12	C0MDATA112			√		Undefined
FFFEC282H	CAN0 message data byte 23 register 12	C0MDATA2312				√	Undefined
FFFEC282H	CAN0 message data byte 2 register 12	C0MDATA212			√		Undefined
FFFEC283H	CAN0 message data byte 3 register 12	C0MDATA312			√		Undefined
FFFEC284H	CAN0 message data byte 45 register 12	C0MDATA4512				√	Undefined
FFFEC284H	CAN0 message data byte 4 register 12	C0MDATA412			√		Undefined
FFFEC285H	CAN0 message data byte 5 register 12	C0MDATA512			√		Undefined
FFFEC286H	CAN0 message data byte 67 register 12	C0MDATA6712				√	Undefined
FFFEC286H	CAN0 message data byte 6 register 12	C0MDATA612			√		Undefined
FFFEC287H	CAN0 message data byte 7 register 12	C0MDATA712			√		Undefined
FFFEC288H	CAN0 message data length code register 12	C0MDLC12			√		0000xxxx
FFFEC289H	CAN0 message configuration register 12	C0MCONF12			√		Undefined
FFFEC28AH	CAN0 message identifier register 12	C0MIDL12				√	Undefined
FFFEC28CH		C0MIDH12				√	Undefined
FFFEC28EH	CAN0 message control register 12	C0MCTRL12				√	00x00000 000xx000B
FFFEC2A0H	CAN0 message data byte 01 register 13	C0MDATA0113				√	Undefined
FFFEC2A0H	CAN0 message data byte 0 register 13	C0MDATA013			√		Undefined
FFFEC2A1H	CAN0 message data byte 1 register 13	C0MDATA113			√		Undefined
FFFEC2A2H	CAN0 message data byte 23 register 13	C0MDATA2313				√	Undefined
FFFEC2A2H	CAN0 message data byte 2 register 13	C0MDATA213			√		Undefined
FFFEC2A3H	CAN0 message data byte 3 register 13	C0MDATA313			√		Undefined
FFFEC2A4H	CAN0 message data byte 45 register 13	C0MDATA4513				√	Undefined
FFFEC2A4H	CAN0 message data byte 4 register 13	C0MDATA413			√		Undefined
FFFEC2A5H	CAN0 message data byte 5 register 13	C0MDATA513			√		Undefined
FFFEC2A6H	CAN0 message data byte 67 register 13	C0MDATA6713				√	Undefined
FFFEC2A6H	CAN0 message data byte 6 register 13	C0MDATA613			√		Undefined
FFFEC2A7H	CAN0 message data byte 7 register 13	C0MDATA713			√		Undefined
FFFEC2A8H	CAN0 message data length code register 13	C0MDLC13			√		0000xxxx
FFFEC2A9H	CAN0 message configuration register 13	C0MCONF13			√		Undefined
FFFEC2AAH	CAN0 message identifier register 13	C0MIDL13				√	Undefined
FFFEC2ACH		C0MIDH13				√	Undefined
FFFEC2AEH	CAN0 message control register 13	C0MCTRL13				√	00x00000 000xx000B

(8/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC2C0H	CAN0 message data byte 01 register 14	C0MDATA0114	R/W			√	Undefined
FFFEC2C0H	CAN0 message data byte 0 register 14	C0MDATA014			√		Undefined
FFFEC2C1H	CAN0 message data byte 1 register 14	C0MDATA114			√		Undefined
FFFEC2C2H	CAN0 message data byte 23 register 14	C0MDATA2314				√	Undefined
FFFEC2C2H	CAN0 message data byte 2 register 14	C0MDATA214			√		Undefined
FFFEC2C3H	CAN0 message data byte 3 register 14	C0MDATA314			√		Undefined
FFFEC2C4H	CAN0 message data byte 45 register 14	C0MDATA4514				√	Undefined
FFFEC2C4H	CAN0 message data byte 4 register 14	C0MDATA414			√		Undefined
FFFEC2C5H	CAN0 message data byte 5 register 14	C0MDATA514			√		Undefined
FFFEC2C6H	CAN0 message data byte 67 register 14	C0MDATA6714				√	Undefined
FFFEC2C6H	CAN0 message data byte 6 register 14	C0MDATA614			√		Undefined
FFFEC2C7H	CAN0 message data byte 7 register 14	C0MDATA714			√		Undefined
FFFEC2C8H	CAN0 message data length code register 14	C0MDLC14			√		0000xxxx
FFFEC2C9H	CAN0 message configuration register 14	C0MCONF14			√		Undefined
FFFEC2CAH	CAN0 message identifier register 14	C0MIDL14				√	Undefined
FFFEC2CCH		C0MIDH14				√	Undefined
FFFEC2CEH	CAN0 message control register 14	C0MCTRL14				√	00x00000 000xx000B
FFFEC2E0H	CAN0 message data byte 01 register 15	C0MDATA0115				√	Undefined
FFFEC2E0H	CAN0 message data byte 0 register 15	C0MDATA015			√		Undefined
FFFEC2E1H	CAN0 message data byte 1 register 15	C0MDATA115			√		Undefined
FFFEC2E2H	CAN0 message data byte 23 register 15	C0MDATA2315				√	Undefined
FFFEC2E2H	CAN0 message data byte 2 register 15	C0MDATA215			√		Undefined
FFFEC2E3H	CAN0 message data byte 3 register 15	C0MDATA315			√		Undefined
FFFEC2E4H	CAN0 message data byte 45 register 15	C0MDATA4515				√	Undefined
FFFEC2E4H	CAN0 message data byte 4 register 15	C0MDATA415			√		Undefined
FFFEC2E5H	CAN0 message data byte 5 register 15	C0MDATA515			√		Undefined
FFFEC2E6H	CAN0 message data byte 67 register 15	C0MDATA6715				√	Undefined
FFFEC2E6H	CAN0 message data byte 6 register 15	C0MDATA615			√		Undefined
FFFEC2E7H	CAN0 message data byte 7 register 15	C0MDATA715			√		Undefined
FFFEC2E8H	CAN0 message data length code register 15	C0MDLC15			√		0000xxxx
FFFEC2E9H	CAN0 message configuration register 15	C0MCONF15			√		Undefined
FFFEC2EAH	CAN0 message identifier register 15	C0MIDL15				√	Undefined
FFFEC2ECH		C0MIDH15				√	Undefined
FFFEC2EEH	CAN0 message control register 15	C0MCTRL15				√	00x00000 000xx000B

(9/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC300H	CAN0 message data byte 01 register 16	C0MDATA0116	R/W			√	Undefined
FFFEC300H	CAN0 message data byte 0 register 16	C0MDATA016			√		Undefined
FFFEC301H	CAN0 message data byte 1 register 16	C0MDATA116			√		Undefined
FFFEC302H	CAN0 message data byte 23 register 16	C0MDATA2316				√	Undefined
FFFEC302H	CAN0 message data byte 2 register 16	C0MDATA216			√		Undefined
FFFEC303H	CAN0 message data byte 3 register 16	C0MDATA316			√		Undefined
FFFEC304H	CAN0 message data byte 45 register 16	C0MDATA4516				√	Undefined
FFFEC304H	CAN0 message data byte 4 register 16	C0MDATA416			√		Undefined
FFFEC305H	CAN0 message data byte 5 register 16	C0MDATA516			√		Undefined
FFFEC306H	CAN0 message data byte 67 register 16	C0MDATA6716				√	Undefined
FFFEC306H	CAN0 message data byte 6 register 16	C0MDATA616			√		Undefined
FFFEC307H	CAN0 message data byte 7 register 16	C0MDATA716			√		Undefined
FFFEC308H	CAN0 message data length code register 16	C0MDLC16			√		0000xxxx
FFFEC309H	CAN0 message configuration register 16	C0MCONF16			√		Undefined
FFFEC30AH	CAN0 message identifier register 16	C0MIDL16				√	Undefined
FFFEC30CH		C0MIDH16				√	Undefined
FFFEC30EH	CAN0 message control register 16	C0MCTRL16				√	00x00000 000xx000B
FFFEC320H	CAN0 message data byte 01 register 17	C0MDATA0117				√	Undefined
FFFEC320H	CAN0 message data byte 0 register 17	C0MDATA017			√		Undefined
FFFEC321H	CAN0 message data byte 1 register 17	C0MDATA117			√		Undefined
FFFEC322H	CAN0 message data byte 23 register 17	C0MDATA2317				√	Undefined
FFFEC322H	CAN0 message data byte 2 register 17	C0MDATA217			√		Undefined
FFFEC323H	CAN0 message data byte 3 register 17	C0MDATA317			√		Undefined
FFFEC324H	CAN0 message data byte 45 register 17	C0MDATA4517				√	Undefined
FFFEC324H	CAN0 message data byte 4 register 17	C0MDATA417			√		Undefined
FFFEC325H	CAN0 message data byte 5 register 17	C0MDATA517			√		Undefined
FFFEC326H	CAN0 message data byte 67 register 17	C0MDATA6717				√	Undefined
FFFEC326H	CAN0 message data byte 6 register 17	C0MDATA617			√		Undefined
FFFEC327H	CAN0 message data byte 7 register 17	C0MDATA717			√		Undefined
FFFEC328H	CAN0 message data length code register 17	C0MDLC17			√		0000xxxx
FFFEC329H	CAN0 message configuration register 17	C0MCONF17			√		Undefined
FFFEC32AH	CAN0 message identifier register 17	C0MIDL17				√	Undefined
FFFEC32CH		C0MIDH17				√	Undefined
FFFEC32EH	CAN0 message control register 17	C0MCTRL17				√	00x00000 000xx000B

(10/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC340H	CAN0 message data byte 01 register 18	C0MDATA0118	R/W			√	Undefined
FFFEC340H	CAN0 message data byte 0 register 18	C0MDATA018			√		Undefined
FFFEC341H	CAN0 message data byte 1 register 18	C0MDATA118			√		Undefined
FFFEC342H	CAN0 message data byte 23 register 18	C0MDATA2318				√	Undefined
FFFEC342H	CAN0 message data byte 2 register 18	C0MDATA218			√		Undefined
FFFEC343H	CAN0 message data byte 3 register 18	C0MDATA318			√		Undefined
FFFEC344H	CAN0 message data byte 45 register 18	C0MDATA4518				√	Undefined
FFFEC344H	CAN0 message data byte 4 register 18	C0MDATA418			√		Undefined
FFFEC345H	CAN0 message data byte 5 register 18	C0MDATA518			√		Undefined
FFFEC346H	CAN0 message data byte 67 register 18	C0MDATA6718				√	Undefined
FFFEC346H	CAN0 message data byte 6 register 18	C0MDATA618			√		Undefined
FFFEC347H	CAN0 message data byte 7 register 18	C0MDATA718			√		Undefined
FFFEC348H	CAN0 message data length code register 18	C0MDLC18			√		0000xxxx
FFFEC349H	CAN0 message configuration register 18	C0MCONF18			√		Undefined
FFFEC34AH	CAN0 message identifier register 18	C0MIDL18				√	Undefined
FFFEC34CH		C0MIDH18				√	Undefined
FFFEC34EH	CAN0 message control register 18	C0MCTRL18				√	00x00000 000xx000B
FFFEC360H	CAN0 message data byte 01 register 19	C0MDATA0119				√	Undefined
FFFEC360H	CAN0 message data byte 0 register 19	C0MDATA019			√		Undefined
FFFEC361H	CAN0 message data byte 1 register 19	C0MDATA119			√		Undefined
FFFEC362H	CAN0 message data byte 23 register 19	C0MDATA2319				√	Undefined
FFFEC362H	CAN0 message data byte 2 register 19	C0MDATA219			√		Undefined
FFFEC363H	CAN0 message data byte 3 register 19	C0MDATA319			√		Undefined
FFFEC364H	CAN0 message data byte 45 register 19	C0MDATA4519				√	Undefined
FFFEC364H	CAN0 message data byte 4 register 19	C0MDATA419			√		Undefined
FFFEC365H	CAN0 message data byte 5 register 19	C0MDATA519			√		Undefined
FFFEC366H	CAN0 message data byte 67 register 19	C0MDATA6719				√	Undefined
FFFEC366H	CAN0 message data byte 6 register 19	C0MDATA619			√		Undefined
FFFEC367H	CAN0 message data byte 7 register 19	C0MDATA719			√		Undefined
FFFEC368H	CAN0 message data length code register 19	C0MDLC19			√		0000xxxx
FFFEC369H	CAN0 message configuration register 19	C0MCONF19			√		Undefined
FFFEC36AH	CAN0 message identifier register 19	C0MIDL19				√	Undefined
FFFEC36CH		C0MIDH19				√	Undefined
FFFEC36EH	CAN0 message control register 19	C0MCTRL19				√	00x00000 000xx000B

(11/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC380H	CAN0 message data byte 01 register 20	C0MDATA0120	R/W			√	Undefined
FFFEC380H	CAN0 message data byte 0 register 20	C0MDATA020			√		Undefined
FFFEC381H	CAN0 message data byte 1 register 20	C0MDATA120			√		Undefined
FFFEC382H	CAN0 message data byte 23 register 20	C0MDATA2320				√	Undefined
FFFEC382H	CAN0 message data byte 2 register 20	C0MDATA220			√		Undefined
FFFEC383H	CAN0 message data byte 3 register 20	C0MDATA320			√		Undefined
FFFEC384H	CAN0 message data byte 45 register 20	C0MDATA4520				√	Undefined
FFFEC384H	CAN0 message data byte 4 register 20	C0MDATA420			√		Undefined
FFFEC385H	CAN0 message data byte 5 register 20	C0MDATA520			√		Undefined
FFFEC386H	CAN0 message data byte 67 register 20	C0MDATA6720				√	Undefined
FFFEC386H	CAN0 message data byte 6 register 20	C0MDATA620			√		Undefined
FFFEC387H	CAN0 message data byte 7 register 20	C0MDATA720			√		Undefined
FFFEC388H	CAN0 message data length code register 20	C0MDLC20			√		0000xxxx
FFFEC389H	CAN0 message configuration register 20	C0MCONF20			√		Undefined
FFFEC38AH	CAN0 message identifier register 20	C0MIDL20				√	Undefined
FFFEC38CH		C0MIDH20				√	Undefined
FFFEC38EH	CAN0 message control register 20	C0MCTRL20				√	00x00000 000xx000B
FFFEC3A0H	CAN0 message data byte 01 register 21	C0MDATA0121				√	Undefined
FFFEC3A0H	CAN0 message data byte 0 register 21	C0MDATA021			√		Undefined
FFFEC3A1H	CAN0 message data byte 1 register 21	C0MDATA121			√		Undefined
FFFEC3A2H	CAN0 message data byte 23 register 21	C0MDATA2321				√	Undefined
FFFEC3A2H	CAN0 message data byte 2 register 21	C0MDATA221			√		Undefined
FFFEC3A3H	CAN0 message data byte 3 register 21	C0MDATA321			√		Undefined
FFFEC3A4H	CAN0 message data byte 45 register 21	C0MDATA4521				√	Undefined
FFFEC3A4H	CAN0 message data byte 4 register 21	C0MDATA421			√		Undefined
FFFEC3A5H	CAN0 message data byte 5 register 21	C0MDATA521			√		Undefined
FFFEC3A6H	CAN0 message data byte 67 register 21	C0MDATA6721				√	Undefined
FFFEC3A6H	CAN0 message data byte 6 register 21	C0MDATA621			√		Undefined
FFFEC3A7H	CAN0 message data byte 7 register 21	C0MDATA721			√		Undefined
FFFEC3A8H	CAN0 message data length code register 21	C0MDLC21			√		0000xxxx
FFFEC3A9H	CAN0 message configuration register 21	C0MCONF21			√		Undefined
FFFEC3AAH	CAN0 message identifier register 21	C0MIDL21				√	Undefined
FFFEC3ACH		C0MIDH21				√	Undefined
FFFEC3AEH	CAN0 message control register 21	C0MCTRL21				√	00x00000 000xx000B

(12/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC3C0H	CAN0 message data byte 01 register 22	C0MDATA0122	R/W			√	Undefined
FFFEC3C0H	CAN0 message data byte 0 register 22	C0MDATA022			√		Undefined
FFFEC3C1H	CAN0 message data byte 1 register 22	C0MDATA122			√		Undefined
FFFEC3C2H	CAN0 message data byte 23 register 22	C0MDATA2322				√	Undefined
FFFEC3C2H	CAN0 message data byte 2 register 22	C0MDATA222			√		Undefined
FFFEC3C3H	CAN0 message data byte 3 register 22	C0MDATA322			√		Undefined
FFFEC3C4H	CAN0 message data byte 45 register 22	C0MDATA4522				√	Undefined
FFFEC3C4H	CAN0 message data byte 4 register 22	C0MDATA422			√		Undefined
FFFEC3C5H	CAN0 message data byte 5 register 22	C0MDATA522			√		Undefined
FFFEC3C6H	CAN0 message data byte 67 register 22	C0MDATA6722				√	Undefined
FFFEC3C6H	CAN0 message data byte 6 register 22	C0MDATA622			√		Undefined
FFFEC3C7H	CAN0 message data byte 7 register 22	C0MDATA722			√		Undefined
FFFEC3C8H	CAN0 message data length code register 22	C0MDLC22			√		0000xxxx
FFFEC3C9H	CAN0 message configuration register 22	C0MCONF22			√		Undefined
FFFEC3CAH	CAN0 message identifier register 22	C0MIDL22				√	Undefined
FFFEC3CCH		C0MIDH22				√	Undefined
FFFEC3CEH	CAN0 message control register 22	C0MCTRL22				√	00x00000 000xx000B
FFFEC3E0H	CAN0 message data byte 01 register 23	C0MDATA0123				√	Undefined
FFFEC3E0H	CAN0 message data byte 0 register 23	C0MDATA023			√		Undefined
FFFEC3E1H	CAN0 message data byte 1 register 23	C0MDATA123			√		Undefined
FFFEC3E2H	CAN0 message data byte 23 register 23	C0MDATA2323				√	Undefined
FFFEC3E2H	CAN0 message data byte 2 register 23	C0MDATA223			√		Undefined
FFFEC3E3H	CAN0 message data byte 3 register 23	C0MDATA323			√		Undefined
FFFEC3E4H	CAN0 message data byte 45 register 23	C0MDATA4523				√	Undefined
FFFEC3E4H	CAN0 message data byte 4 register 23	C0MDATA423			√		Undefined
FFFEC3E5H	CAN0 message data byte 5 register 23	C0MDATA523			√		Undefined
FFFEC3E6H	CAN0 message data byte 67 register 23	C0MDATA6723				√	Undefined
FFFEC3E6H	CAN0 message data byte 6 register 23	C0MDATA623			√		Undefined
FFFEC3E7H	CAN0 message data byte 7 register 23	C0MDATA723			√		Undefined
FFFEC3E8H	CAN0 message data length code register 23	C0MDLC23			√		0000xxxx
FFFEC3E9H	CAN0 message configuration register 23	C0MCONF23			√		Undefined
FFFEC3EAH	CAN0 message identifier register 23	C0MIDL23				√	Undefined
FFFEC3ECH		C0MIDH23				√	Undefined
FFFEC3EEH	CAN0 message control register 23	C0MCTRL23				√	00x00000 000xx000B

(13/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC400H	CAN0 message data byte 01 register 24	C0MDATA0124	R/W			√	Undefined
FFFEC400H	CAN0 message data byte 0 register 24	C0MDATA024			√		Undefined
FFFEC401H	CAN0 message data byte 1 register 24	C0MDATA124			√		Undefined
FFFEC402H	CAN0 message data byte 23 register 24	C0MDATA2324				√	Undefined
FFFEC402H	CAN0 message data byte 2 register 24	C0MDATA224			√		Undefined
FFFEC403H	CAN0 message data byte 3 register 24	C0MDATA324			√		Undefined
FFFEC404H	CAN0 message data byte 45 register 24	C0MDATA4524				√	Undefined
FFFEC404H	CAN0 message data byte 4 register 24	C0MDATA424			√		Undefined
FFFEC405H	CAN0 message data byte 5 register 24	C0MDATA524			√		Undefined
FFFEC406H	CAN0 message data byte 67 register 24	C0MDATA6724				√	Undefined
FFFEC406H	CAN0 message data byte 6 register 24	C0MDATA624			√		Undefined
FFFEC407H	CAN0 message data byte 7 register 24	C0MDATA724			√		Undefined
FFFEC408H	CAN0 message data length code register 24	C0MDLC24			√		0000xxxx
FFFEC409H	CAN0 message configuration register 24	C0MCONF24			√		Undefined
FFFEC40AH	CAN0 message identifier register 24	C0MIDL24				√	Undefined
FFFEC40CH		C0MIDH24				√	Undefined
FFFEC40EH	CAN0 message control register 24	C0MCTRL24				√	00x00000 000xx000B
FFFEC420H	CAN0 message data byte 01 register 25	C0MDATA0125				√	Undefined
FFFEC420H	CAN0 message data byte 0 register 25	C0MDATA025			√		Undefined
FFFEC421H	CAN0 message data byte 1 register 25	C0MDATA125			√		Undefined
FFFEC422H	CAN0 message data byte 23 register 25	C0MDATA2325				√	Undefined
FFFEC422H	CAN0 message data byte 2 register 25	C0MDATA225			√		Undefined
FFFEC423H	CAN0 message data byte 3 register 25	C0MDATA325			√		Undefined
FFFEC424H	CAN0 message data byte 45 register 25	C0MDATA4525				√	Undefined
FFFEC424H	CAN0 message data byte 4 register 25	C0MDATA425			√		Undefined
FFFEC425H	CAN0 message data byte 5 register 25	C0MDATA525			√		Undefined
FFFEC426H	CAN0 message data byte 67 register 25	C0MDATA6725				√	Undefined
FFFEC426H	CAN0 message data byte 6 register 25	C0MDATA625			√		Undefined
FFFEC427H	CAN0 message data byte 7 register 25	C0MDATA725			√		Undefined
FFFEC428H	CAN0 message data length code register 25	C0MDLC25			√		0000xxxx
FFFEC429H	CAN0 message configuration register 25	C0MCONF25			√		Undefined
FFFEC42AH	CAN0 message identifier register 25	C0MIDL25				√	Undefined
FFFEC42CH		C0MIDH25				√	Undefined
FFFEC42EH	CAN0 message control register 25	C0MCTRL25				√	00x00000 000xx000B

(14/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC440H	CAN0 message data byte 01 register 26	C0MDATA0126	R/W			√	Undefined
FFFEC440H	CAN0 message data byte 0 register 26	C0MDATA026			√		Undefined
FFFEC441H	CAN0 message data byte 1 register 26	C0MDATA126			√		Undefined
FFFEC442H	CAN0 message data byte 23 register 26	C0MDATA2326				√	Undefined
FFFEC442H	CAN0 message data byte 2 register 26	C0MDATA226			√		Undefined
FFFEC443H	CAN0 message data byte 3 register 26	C0MDATA326			√		Undefined
FFFEC444H	CAN0 message data byte 45 register 26	C0MDATA4526				√	Undefined
FFFEC444H	CAN0 message data byte 4 register 26	C0MDATA426			√		Undefined
FFFEC445H	CAN0 message data byte 5 register 26	C0MDATA526			√		Undefined
FFFEC446H	CAN0 message data byte 67 register 26	C0MDATA6726				√	Undefined
FFFEC446H	CAN0 message data byte 6 register 26	C0MDATA626			√		Undefined
FFFEC447H	CAN0 message data byte 7 register 26	C0MDATA726			√		Undefined
FFFEC448H	CAN0 message data length code register 26	C0MDLC26			√		0000xxxx
FFFEC449H	CAN0 message configuration register 26	C0MCONF26			√		Undefined
FFFEC44AH	CAN0 message identifier register 26	C0MIDL26				√	Undefined
FFFEC44CH		C0MIDH26				√	Undefined
FFFEC44EH	CAN0 message control register 26	C0MCTRL26				√	00x00000 000xx000B
FFFEC460H	CAN0 message data byte 01 register 27	C0MDATA0127				√	Undefined
FFFEC460H	CAN0 message data byte 0 register 27	C0MDATA027			√		Undefined
FFFEC461H	CAN0 message data byte 1 register 27	C0MDATA127			√		Undefined
FFFEC462H	CAN0 message data byte 23 register 27	C0MDATA2327				√	Undefined
FFFEC462H	CAN0 message data byte 2 register 27	C0MDATA227			√		Undefined
FFFEC463H	CAN0 message data byte 3 register 27	C0MDATA327			√		Undefined
FFFEC464H	CAN0 message data byte 45 register 27	C0MDATA4527				√	Undefined
FFFEC464H	CAN0 message data byte 4 register 27	C0MDATA427			√		Undefined
FFFEC465H	CAN0 message data byte 5 register 27	C0MDATA527			√		Undefined
FFFEC466H	CAN0 message data byte 67 register 27	C0MDATA6727				√	Undefined
FFFEC466H	CAN0 message data byte 6 register 27	C0MDATA627			√		Undefined
FFFEC467H	CAN0 message data byte 7 register 27	C0MDATA727			√		Undefined
FFFEC468H	CAN0 message data length code register 27	C0MDLC27			√		0000xxxx
FFFEC469H	CAN0 message configuration register 27	C0MCONF27			√		Undefined
FFFEC46AH	CAN0 message identifier register 27	C0MIDL27				√	Undefined
FFFEC46CH		C0MIDH27				√	Undefined
FFFEC46EH	CAN0 message control register 27	C0MCTRL27				√	00x00000 000xx000B

(15/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC480H	CAN0 message data byte 01 register 28	C0MDATA0128	R/W			√	Undefined
FFFEC480H	CAN0 message data byte 0 register 28	C0MDATA028			√		Undefined
FFFEC481H	CAN0 message data byte 1 register 28	C0MDATA128			√		Undefined
FFFEC482H	CAN0 message data byte 23 register 28	C0MDATA2328				√	Undefined
FFFEC482H	CAN0 message data byte 2 register 28	C0MDATA228			√		Undefined
FFFEC483H	CAN0 message data byte 3 register 28	C0MDATA328			√		Undefined
FFFEC484H	CAN0 message data byte 45 register 28	C0MDATA4528				√	Undefined
FFFEC484H	CAN0 message data byte 4 register 28	C0MDATA428			√		Undefined
FFFEC485H	CAN0 message data byte 5 register 28	C0MDATA528			√		Undefined
FFFEC486H	CAN0 message data byte 67 register 28	C0MDATA6728				√	Undefined
FFFEC486H	CAN0 message data byte 6 register 28	C0MDATA628			√		Undefined
FFFEC487H	CAN0 message data byte 7 register 28	C0MDATA728			√		Undefined
FFFEC488H	CAN0 message data length code register 28	C0MDLC28			√		0000xxxx
FFFEC489H	CAN0 message configuration register 28	C0MCONF28			√		Undefined
FFFEC48AH	CAN0 message identifier register 28	C0MIDL28				√	Undefined
FFFEC48CH		C0MIDH28				√	Undefined
FFFEC48EH	CAN0 message control register 28	C0MCTRL28				√	00x00000 000xx000B
FFFEC4A0H	CAN0 message data byte 01 register 29	C0MDATA0129				√	Undefined
FFFEC4A0H	CAN0 message data byte 0 register 29	C0MDATA029			√		Undefined
FFFEC4A1H	CAN0 message data byte 1 register 29	C0MDATA129			√		Undefined
FFFEC4A2H	CAN0 message data byte 23 register 29	C0MDATA2329				√	Undefined
FFFEC4A2H	CAN0 message data byte 2 register 29	C0MDATA229			√		Undefined
FFFEC4A3H	CAN0 message data byte 3 register 29	C0MDATA329			√		Undefined
FFFEC4A4H	CAN0 message data byte 45 register 29	C0MDATA4529				√	Undefined
FFFEC4A4H	CAN0 message data byte 4 register 29	C0MDATA429			√		Undefined
FFFEC4A5H	CAN0 message data byte 5 register 29	C0MDATA529			√		Undefined
FFFEC4A6H	CAN0 message data byte 67 register 29	C0MDATA6729				√	Undefined
FFFEC4A6H	CAN0 message data byte 6 register 29	C0MDATA629			√		Undefined
FFFEC4A7H	CAN0 message data byte 7 register 29	C0MDATA729			√		Undefined
FFFEC4A8H	CAN0 message data length code register 29	C0MDLC29			√		0000xxxx
FFFEC4A9H	CAN0 message configuration register 29	C0MCONF29			√		Undefined
FFFEC4AAH	CAN0 message identifier register 29	C0MIDL29				√	Undefined
FFFEC4ACH		C0MIDH29				√	Undefined
FFFEC4AEH	CAN0 message control register 29	C0MCTRL29				√	00x00000 000xx000B

(16/16)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
FFFEC4C0H	CAN0 message data byte 01 register 30	C0MDATA0130	R/W			√	Undefined
FFFEC4C0H	CAN0 message data byte 0 register 30	C0MDATA030			√		Undefined
FFFEC4C1H	CAN0 message data byte 1 register 30	C0MDATA130			√		Undefined
FFFEC4C2H	CAN0 message data byte 23 register 30	C0MDATA2330				√	Undefined
FFFEC4C2H	CAN0 message data byte 2 register 30	C0MDATA230			√		Undefined
FFFEC4C3H	CAN0 message data byte 3 register 30	C0MDATA330			√		Undefined
FFFEC4C4H	CAN0 message data byte 45 register 30	C0MDATA4530				√	Undefined
FFFEC4C4H	CAN0 message data byte 4 register 30	C0MDATA430			√		Undefined
FFFEC4C5H	CAN0 message data byte 5 register 30	C0MDATA530			√		Undefined
FFFEC4C6H	CAN0 message data byte 67 register 30	C0MDATA6730				√	Undefined
FFFEC4C6H	CAN0 message data byte 6 register 30	C0MDATA630			√		Undefined
FFFEC4C7H	CAN0 message data byte 7 register 30	C0MDATA730			√		Undefined
FFFEC4C8H	CAN0 message data length code register 30	C0MDLC30			√		0000xxxx
FFFEC4C9H	CAN0 message configuration register 30	C0MCONF30			√		Undefined
FFFEC4CAH	CAN0 message identifier register 30	C0MIDL30				√	Undefined
FFFEC4CCH		C0MIDH30				√	Undefined
FFFEC4CEH	CAN0 message control register 30	C0MCTRL30				√	00x00000 000xx000B
FFFEC4E0H	CAN0 message data byte 01 register 31	C0MDATA0131				√	Undefined
FFFEC4E0H	CAN0 message data byte 0 register 31	C0MDATA031			√		Undefined
FFFEC4E1H	CAN0 message data byte 1 register 31	C0MDATA131			√		Undefined
FFFEC4E2H	CAN0 message data byte 23 register 31	C0MDATA2331				√	Undefined
FFFEC4E2H	CAN0 message data byte 2 register 31	C0MDATA231			√		Undefined
FFFEC4E3H	CAN0 message data byte 3 register 31	C0MDATA331			√		Undefined
FFFEC4E4H	CAN0 message data byte 45 register 31	C0MDATA4531				√	Undefined
FFFEC4E4H	CAN0 message data byte 4 register 31	C0MDATA431			√		Undefined
FFFEC4E5H	CAN0 message data byte 5 register 31	C0MDATA531			√		Undefined
FFFEC4E6H	CAN0 message data byte 67 register 31	C0MDATA6731				√	Undefined
FFFEC4E6H	CAN0 message data byte 6 register 31	C0MDATA631			√		Undefined
FFFEC4E7H	CAN0 message data byte 7 register 31	C0MDATA731			√		Undefined
FFFEC4E8H	CAN0 message data length code register 31	C0MDLC31			√		0000xxxx
FFFEC4E9H	CAN0 message configuration register 31	C0MCONF31			√		Undefined
FFFEC4EAH	CAN0 message identifier register 31	C0MIDL31				√	Undefined
FFFEC4ECH		C0MIDH31				√	Undefined
FFFEC4EEH	CAN0 message control register 31	C0MCTRL31				√	00x00000 000xx000B

19.5.3 Control bits of message buffers

Table 19-20. Control Bits of Control Registers (1/2)

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
FFFEC000H	C0GMCTRL(W)	0	0	0	0	0	0	0	cGOM
FFFEC001H		0	0	0	0	0	0	sEFSD	sGOM
FFFEC000H	C0GMCTRL(R)	0	0	0	0	0	0	EFSD	GOM
FFFEC001H		MBON	0	0	0	0	0	0	0
FFFEC002H	C0GMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0
FFFEC006H	C0GMABT(W)	0	0	0	0	0	0	0	cABTTRGG
FFFEC007H		0	0	0	0	0	0	sABTCLR	sABTTRG
FFFEC006H	C0GMABT(R)	0	0	0	0	0	0	ABTCLR	ABTTRG
FFFEC007H		0	0	0	0	0	0	0	0
FFFEC008H	C0GMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0
FFFEC040H	C0MASK1L	CM1ID [7:0]							
FFFEC041H		CM1ID [15:8]							
FFFEC042H	C0MASK1H	CM1ID [23:16]							
FFFEC043H		0	0	0	CM1ID [28:24]				
FFFEC044H	C0MASK2L	CM2ID [7:0]							
FFFEC045H		CM2ID [15:8]							
FFFEC046H	C0MASK2H	CM2ID [23:16]							
FFFEC047H		0	0	0	CM2ID [28:24]				
FFFEC048H	C0MASK3L	CM3ID [7:0]							
FFFEC049H		CM3ID [15:8]							
FFFEC04AH	C0MASK3H	CM3ID [23:16]							
FFFEC04BH		0	0	0	CM3ID [28:24]				
FFFEC04CH	C0MASK4L	CM4ID [7:0]							
FFFEC04DH		CM4ID [15:8]							
FFFEC04EH	C0MASK4H	CM4ID [23:16]							
FFFEC04FH		0	0	0	CM4ID [28:24]				
FFFEC050H	C0CTRL(W)	cCCERC	cAL	cVALID	cPS MODE1	cPS MODE0	cOP MODE2	cOP MODE1	cOP MODE0
FFFEC051H		sCCERC	sAL	0	sPS MODE1	sPS MODE0	sOP MODE2	sOP MODE1	sOP MODE0
FFFEC050H	C0CTRL(R)	CCERC	AL	VALID	PS MODE1	PS MODE0	OP MODE2	OP MODE1	OP MODE0
FFFEC051H		0	0	0	0	0	0	0	0
FFFEC052H	C0LEC(W)	0	0	0	0	0	0	0	0
FFFEC052H	C0LEC(R)	0	0	0	0	0	LEC2	LEC1	LEC0
FFFEC053H	C0INFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0

Table 19-20. Control Bits of Control Registers (2/2)

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
FFFEC054H	C0ERC	TEC[7:0]							
FFFEC055H		REC[7:0]							
FFFEC056H	C0IE (W)	0	0	cCIE5	cCIE4	cCIE3	cCIE2	cCIE1	cCIE0
FFFEC057H		0	0	sCIE5	sCIE4	sCIE3	sCIE2	sCIE1	sCIE0
FFFEC056H	C0IE (R)	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0
FFFEC057H		0	0	0	0	0	0	0	0
FFFEC058H	C0INTS (W)	0	0	cCINTS5	cCINTS4	cCINTS3	cCINTS2	cCINTS1	cCINTS0
FFFEC059H		0	0	0	0	0	0	0	0
FFFEC058H	C0INTS (R)	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0
FFFEC059H		0	0	0	0	0	0	0	0
FFFEC05AH	C0BRP	TQPRS[7:0]							
FFFEC05BH	–	Access prohibited (reserved for future use)							
FFFEC05CH	C0BTR	0	0	0	0	TSEG1[3:0]			
FFFEC05DH				SJW[1:0]		0	TSEG2[2:0]		
FFFEC05EH	C0LIPT	LIPT[7:0]							
FFFEC05FH	–	Access prohibited (reserved for future use)							
FFFEC060H	C0RGPT(W)	0	0	0	0	0	0	0	cROVF
FFFEC061H		0	0	0	0	0	0	0	0
FFFEC060H	C0RGPT(R)	0	0	0	0	0	0	RHPM	ROVF
FFFEC061H		RGPT[7:0]							
FFFEC062H	C0LOPT	LOPT[7:0]							
FFFEC063H	–	Access prohibited (reserved for future use)							
FFFEC064H	C0TGPT(W)	0	0	0	0	0	0	0	cTOVF
FFFEC065H		0	0	0	0	0	0	0	0
FFFEC064H	C0TGPT(R)	0	0	0	0	0	0	THPM	TOVF
FFFEC065H		TGPT[7:0]							
FFFEC066H	C0TS(W)	0	0	0	0	0	cTSLOCK	cTSSEL	cTSEN
FFFEC067H		0	0	0	0	0	sTSLOCK	sTSSEL	sTSEN
FFFEC066H	C0TS(R)	0	0	0	0	0	TSLOCK	TSSEL	TSEN
FFFEC067H		0	0	0	0	0	0	0	0
FFFEC068H-	–	Access prohibited (reserved for future use)							

Table 19-21. Control Bits of Message Buffers

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
FFFECxx0H	C0MDATA01m	Message data (byte 0)							
FFFECxx1H		Message data (byte 1)							
FFFECxx0H	C0MDATA0m	Message data (byte 0)							
FFFECxx1H	C0MDATA1m	Message data (byte 1)							
FFFECxx2H	C0MDATA23m	Message data (byte 2)							
FFFECxx3H		Message data (byte 3)							
FFFECxx2H	C0MDATA2m	Message data (byte 2)							
FFFECxx3H	C0MDATA3m	Message data (byte 3)							
FFFECxx4H	C0MDATA45m	Message data (byte 4)							
FFFECxx5H		Message data (byte 5)							
FFFECxx4H	C0MDATA4m	Message data (byte 4)							
FFFECxx5H	C0MDATA5m	Message data (byte 5)							
FFFECxx6H	C0MDATA67m	Message data (byte 6)							
FFFECxx7H		Message data (byte 7)							
FFFECxx6H	C0MDATA6m	Message data (byte 6)							
FFFECxx7H	C0MDATA7m	Message data (byte 7)							
FFFECxx8H	C0MDLCm	0				MDLC3	MDLC2	MDLC1	MDLC0
FFFECxx9H	C0MCONFm	OWS	RTR	MT2	MT1	MT0	MA2	MA1	MA0
FFFECxxAH	C0MIDLm	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
FFFECxxBH		ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
FFFECxxCH	C0MIDHm	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
FFFECxxDH		IDE	0	0	ID28	ID27	ID26	ID25	ID24
FFFECxxEH	C0MCTRLm	0	0	0	cMOW	cIE	cDN	cTRQ	cRDY
FFFECxxFH		0	0	0	0	sIE	sDN	sTRQ	sRDY
FFFECxxEH	C0MCTRLm	0	0	0	MOW	IE	DN	TRQ	RDY
FFFECxxFH		0	0	MUC	0	0	0	0	0

19.6 Control Registers

(1) CAN0 module control register (C0GMCTRL)

(1/2)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0GMCTRL	MBON	0	0	0	0	0	0	0	FFFEC001H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	EFSD	GOM	FFFEC000H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0GMCTRL	0	0	0	0	0	0	Set EFSD	Set GOM	FFFEC001H
	7	6	5	4	3	2	1	0	
	0	0	0	0	0	0	0	Clear GOM	FFFEC000H

(a) Read

MBON	Access Enable Bit for Message Buffers
0	Write access and read access to the message buffers is impossible because CAN module is in CAN sleep and/or CAN stop mode.
1	Write access and read access to the message buffers is possible.

Caution Be sure not to access the message buffer registers (C0MDATA0m, C0MDATA1m, C0MDATA2m, C0MDATA3m, C0MDATA4m, C0MDATA5m, C0MDATA6m, C0MDATA7m, C0MDLCm, C0MCONFm, C0MIDLm, C0MIDHm or C0MCTRLm) while the MBON bit is cleared (0).

Remark MBON: Message buffer access on

EFSD	Enabled Forced Shutdown Request Bit
0	Forced shutdown request is disabled.
1	Forced shutdown request is enabled.

GOM	Global Operation Mode Bit
0	CAN module is in reset state.
1	CAN module is enabled.

(2/2)

(b) Write

Set EFSD	EFSD Bit Setting
0	No change in EFSD bit's value
1	EFSD bit set (1)

Caution When the EFSD bit is set (1), the subsequent CPU access to the CAN module has to clear the GOM bit (0). If the GOM bit is not cleared (0) in the subsequent access, the EFSD bit is cleared (0) automatically (forced shutdown request is invalid).

Set GOM	Clear GOM	GOM Bit Setting
0	1	GOM bit cleared (0)
1	0	GOM bit set (1)
Other than above		No change in GOM bit's value

Remark The GOM bit is cleared (0) only in INIT mode.

(2) CAN0 module clock selection register (C0GMCS)

	7	6	5	4	3	2	1	0	Address	Initial value
C0GMCS	MBON	0	0	0	CCP3	CCP2	CCP1	CCP0	FFFEC002H	0FH

(Read/Write)

CCP3	CCP2	CCP1	CCP1	CAN Module System Clock (f_{CANMOD})
0	0	0	0	f_{CAN}/1
0	0	0	1	f_{CAN}/2
0	0	1	0	f_{CAN}/3
0	0	1	1	f_{CAN}/4
0	1	0	0	f_{CAN}/5
0	1	0	1	f_{CAN}/6
0	1	1	0	f_{CAN}/7
0	1	1	1	f_{CAN}/8
1	0	0	0	f_{CAN}/9
1	0	0	1	f_{CAN}/10
1	0	1	0	f_{CAN}/11
1	0	1	1	f_{CAN}/12
1	1	0	0	f_{CAN}/13
1	1	0	1	f_{CAN}/14
1	1	1	0	f_{CAN}/15
1	1	1	1	f_{CAN}/16

Remark f_{CAN} = Clock supply to CAN

(3) CAN0 automatic block transmission register (C0GMABT)

(1/2)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0GMABT	0	0	0	0	0	0	0	0	FFFEC007H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	ABTCLR	ABTTRG	FFFEC006H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0GMABT	0	0	0	0	0	0	Set ABTCLR	Set ABTTRG	FFFEC007H
	7	6	5	4	3	2	1	0	
	0	0	0	0	0	0	0	Clear ABTTRG	FFFEC006H

Caution Before switching from "normal operating mode with automatic block transmission" to the INIT mode, be sure to clear the bits in the C0GMABT register to their initial values.

(a) Read

ABTCLR	Automatic Block Transmission Engine Clear Request Bit
0	The automatic block transmission engine is in the idle state or operating.
1	Clear request to the automatic block transmission engine. Upon re-start of the automatic block transmission engine by setting ABTTRG (1), the automatic block transmission engine starts transmission from the first ABT message buffer (i.e. message buffer 0).

ABTTRG	Automatic Block Transmission Start Bit
0	Automatic block transmission stop
1	Automatic block transmission start

(2/2)

(b) Write

Set ABTCLR	ABTCLR Bit Setting
0	No change in ABTCLR bit's value
1	ABTCLR bit set (1)

Remarks 1. The ABTCLR bit must not be set (1) when the ABTTRG bit is set (1).

2. The ABTCLR bit is automatically cleared (0) by the internal ABT engine, when a clear request has been accepted by setting the AVTCLR bit (1).

Set ABTTRG	Clear ABTTRG	ABTTRG Bit Setting
0	1	ABTTRG bit cleared (0)
1	0	ABTTRG bit set (1)

Caution Be sure not to set the ABTTRG bit in INIT mode.
The correct operation after INIT mode cannot be guaranteed when the ABTTRG bit is set in INIT mode.

(4) CAN0 automatic block transmission delay register (C0GMABTD)

	7	6	5	4	3	2	1	0	Address	Initial value
C0GMABT	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0	FFFEC008H	00H

(Read/Write)

ABTD3	ABTD2	ABTD1	ABTD0	Data Frame Interval During Automatic Block Transmission (Unit = Bit Time; DBT)
0	0	0	0	0 DBT
0	0	0	1	2^5 DBT
0	0	1	0	2^6 DBT
0	0	1	1	2^7 DBT
0	1	0	0	2^8 DBT
0	1	0	1	2^9 DBT
0	1	1	0	2^{10} DBT
0	1	1	1	2^{11} DBT
1	0	0	0	2^{12} DBT
Other than above				Setting prohibited

Caution Be sure not to change the contents of the C0GMABTD register while the ABTTRG bit is set (1).

(5) CAN0 module mask register (C0MASKaL, C0MASKaH) (a = 1, 2, 3, 4)

Figure 19-24. CAN0 Module Mask 1 Registers (C0MASK1L, C0MASK1H)

	15	14	13	12	11	10	9	8	Address	Initial value
C0MASK1L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	FFFEC041H	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	FFFEC040H	Undefined
	15	14	13	12	11	10	9	8		
C0MASK1H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	FFFEC043H	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	FFFEC042H	Undefined

Figure 19-25. CAN0 Module Mask 2 Registers (C0MASK2L, C0MASK2H)

	15	14	13	12	11	10	9	8	Address	Initial value
C0MASK2L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	FFFEC045H	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	FFFEC044H	Undefined
	15	14	13	12	11	10	9	8		
C0MASK2H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	FFFEC047H	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	FFFEC046H	Undefined

Figure 19-26. CAN0 Module Mask 3 Registers (C0MASK3L, C0MASK3H)

	15	14	13	12	11	10	9	8	Address	Initial value
C0MASK3L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	FFFEC049H	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	FFFEC048H	Undefined
	15	14	13	12	11	10	9	8		
C0MASK3H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	FFFEC04BH	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	FFFEC04AH	Undefined

Figure 19-27. CAN0 Module Mask 4 Registers (C0MASK4L, C0MASK4H)

	15	14	13	12	11	10	9	8	Address	Initial value
C0MASK4L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	FFFEC04DH	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	FFFEC04CH	Undefined
	15	14	13	12	11	10	9	8		
C0MASK4H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	FFFEC04FH	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	FFFEC04EH	Undefined

CMID28 to CMID0	Mask Identifier Pattern
0	The ID bit of a received message frame is compared with the ID bit of the message buffer set by the CMID28 to CMID0 bits.
1	The ID bit of a received message frame is not compared (= is masked) with the ID bit of the message buffer set by the CMID28 to CMID0 bits.

(6) CAN0 module control register (C0CTRL)

(1/4)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0CTRL	0	0	0	0	0	0	RSTAT	TSTAT	FFFEC051H	00H
	7	6	5	4	3	2	1	0		
	CCERC	AL	VALID	PSMODE 1	PSMODE 0	OPMODE 2	OPMODE 1	OPMODE 0	FFFEC050H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0CTRL	Set CCERC	Set AL	0	Set PSMODE 1	Set PSMODE 0	Set OPMODE 2	Set OPMODE 1	Set OPMODE 0	FFFEC051H
	7	6	5	4	3	2	1	0	
	Clear CCERC	Clear AL	Clear VALID	Clear PSMODE 1	Clear PSMODE 0	Clear OPMODE 2	Clear OPMODE 1	Clear OPMODE 0	FFFEC050

(a) Read

RSTAT	CAN Reception Status Bit
0	No reception activity on the CAN bus
1	Reception activity on the CAN bus

Remark The RSTAT bit is set (1) under the following conditions.

- The SOF bit of a receive frame is detected
- Arbitration was lost during a transmission frame

The RSTAT bit is cleared (0) under the following conditions.

- “Recessive” is detected at the 2nd bit of an interframe space.
- Initialization mode was shifted to at the first bit of an interframe space

(2/4)

TSTAT	CAN Transmission Status Bit
0	No transmission activity on the CAN bus
1	Transmission activity on the CAN bus

Remark The TSTAT bit is set (1) under the following conditions.

- The SOF bit of a transmission frame is detected
- The first bit of an error flag is detected during a transmission frame

The TSTAT bit is cleared (0) under the following conditions.

- CAN shifted to the bus-off status
- Arbitration was lost during a transmission frame
- “Recessive” is detected at the 2nd bit of an interframe space.
- Initialization mode was shifted to at the first bit of an interframe space

CCERC	CAN Clear Error Counter Bit
0	While in INIT mode, the CAN module error counter C0ERC and the CAN module information register C0INFO will not be cleared.
1	While in INIT mode, the CAN module error counter C0ERC and the CAN module information register C0INFO will be cleared.

AL	Arbitration Loss Bit
0	In the “single-shot mode”, no re-transmission when an error occurs. Transmit message will not be queued for a re-transmission request when arbitration is lost.
1	In the “single-shot mode”, no re-transmission when an error occurs. Transmit message will be queued for a re-transmission request when arbitration is lost.

Remarks 1. The AL bit is effective only in the “single-shot mode”.

2. If a CAN module operates in the “single-shot mode” and the AL bit is set (1), the interrupt CINTS4 is not generated upon arbitration loss.

VALID	Valid Receive Message Frame Detection Bit
0	No valid message frame reception in the CAN protocol transfer layer since the VALID bit was cleared (0) last time.
1	Valid message frame reception in the CAN protocol transfer layer since the VALID bit was cleared (0) last time.

Remarks 1. A valid reception does not require acceptance of the message frame in a receive message buffer (data frame) or transmit message buffer (remote frame).

2. Before switching from INIT mode to any operational mode, the user has to clear the VALID bit (0).

3. If only two CAN nodes are connected to the CAN bus and one of the CAN nodes is in “normal operating mode” and transmitting message frames while the other CAN node is in “receive-only mode”, the VALID bit will not be set (1) before the transmitting node becomes error passive.

(3/4)

PSMODE1	PSMODE0	Power Save Mode
0	0	No power save mode selected (CAN module is in INIT mode or in one of the operational modes)
0	1	CAN sleep mode
1	0	Setting prohibited
1	1	CAN stop mode

OPMODE2	OPMODE1	OPMODE0	Operation Mode
0	0	0	No operational mode selected (CAN module is in INIT mode)
0	0	1	Normal operating mode
0	1	0	Normal operating mode with automatic block transmission
0	1	1	Receive-only mode
1	0	0	Single-shot mode
1	0	1	Self-test mode
Other than above			Setting prohibited.

(b) Write

Set CCERC	Clear CCERC	CCERC Bit Setting
0	1	CCERC bit cleared (0)
1	0	CCERC bit set (1)
Other than above		No change in CCERC bit's value

Set AL	Clear AL	AL Bit Setting
0	1	AL bit cleared (0)
1	0	AL bit set (1)
Other than above		No change in AL bit's value

Clear VALID	VALID Bit Setting
0	VALID bit not changed
1	VALID bit cleared (0)

Set PSMODE1	Clear PSMODE1	PSMODE1 Bit Setting
0	1	PSMODE1 bit cleared (0)
1	0	PSMODE1 bit set (1)
Other than above		No change in PSMODE1 bit's value

(4/4)

Set PSMODE2	Clear PSMODE2	PSMODE2 Bit Setting
0	1	PSMODE2 bit cleared (0)
1	0	PSMODE2 bit set (1)
Other than above		No change in PSMODE2 bit's value

Set OPMODE0	Clear OPMODE0	OPMODE0 Bit Setting
0	1	OPMODE0 bit cleared (0)
1	0	OPMODE0 bit set (1)
Other than above		No change in OPMODE0 bit's value

Set OPMODE1	Clear OPMODE1	OPMODE1 Bit Setting
0	1	OPMODE1 bit cleared (0)
1	0	OPMODE1 bit set (1)
Other than above		No change in OPMODE1 bit's value

Set OPMODE2	Clear OPMODE2	OPMODE2 Bit Setting
0	1	OPMODE2 bit cleared (0)
1	0	OPMODE2 bit set (1)
Other than above		No change in OPMODE2 bit's value

(7) CAN0 module last error code register (C0LEC)

	7	6	5	4	3	2	1	0	Address	Initial value
C0LEC	0	0	0	0	0	LEC2	LEC1	LEC0	FFFEC052H	00H

(Read/Write)

Remarks 1. Switching the CAN module from an operational mode to the INIT mode does not clear the actual content of C0LEC.

2. When CPU attempts to write C0LEC with data other than 00h, it is simply ignored.

LEC2	LEC1	LEC0	Last Error Code of Protocol Error Type
0	0	0	No error
0	0	1	Stuff error
0	1	0	Form error
0	1	1	ACK error
1	0	0	Bit error (The CAN module tried to send a »recessive« ‘1’ bit as part of the transmitted message (with the exception of the arbitration field), but the monitored CAN bus value was »dominant« ‘0’)
1	0	1	Bit error (The CAN module tried to send a »dominant« ‘0’ bit as part of the transmitted message or as an ACK bit, error or overload frame, but the monitored CAN bus value was »recessive« ‘1’)
1	1	0	CRC error
1	1	1	Unused

(8) CAN0 module information register (C0INFO)

	7	6	5	4	3	2	1	0	Address	Initial value
C0INFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0	FFFEC053H	00H

(Read Only)

BOFF	Bus-off Status Bit
0	CAN is not in the bus-off state (transmission error counter < 255)
1	CAN is in the bus-off state (transmission error counter ≥ 255)

TECS1	TECS0	Transmit Error Counter Status Bit
0	0	Transmission error counter below warning level (< 96)
0	1	Transmission error counter in the warning level range (96 … 127)
1	0	Not used
1	1	Transmission error counter in the error passive or bus-off range (≥ 128)

RECS1	RECS0	Receive Error Counter Status Bit
0	0	Reception error counter below warning level (< 96)
0	1	Reception error counter in the warning level range (96 … 127)
1	0	Not used
1	1	Reception error counter in the error passive range (≥ 128)

(9) CAN0 module error counter register (C0ERC)

	15	14	13	12	11	10	9	8	Address	Initial value
C0ERC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	FFFEC055H	00H
(Read Only)	7	6	5	4	3	2	1	0		
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	FFFEC054H	00H

REC7 to REC0	Receive Error Counter Bit
0 to 255	Number of reception error counter. This reflects the current status of the reception error counter. The count value is defined by the CAN protocol.

TEC7 to TEC0	Transmit Error Counter Bit
0 to 255	Number of transmission error counter. This reflects the current status of the transmission error counter. The count value is defined by the CAN protocol.

(10) CAN0 module interrupt enable register (C0IE)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0IE	0	0	0	0	0	0	0	0	FFFEC057H	00H
	7	6	5	4	3	2	1	0		
	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0	FFFEC056H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0IE	0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0	FFFEC057H
	7	6	5	4	3	2	1	0	
	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0	FFFEC056H

(a) Read

CIE5 to CIE0	CAN Interrupt Enable Bus
0	The corresponding interrupt pending bit in the interrupt control register C0INTx is disabled.
1	The corresponding interrupt pending bit in the interrupt control register C0INTx is enabled.

(2/2)

(b) Write

Set CIE5	Clear CIE5	CIE5 Bit Setting
0	1	CIE5 bit cleared (0)
1	0	CIE5 bit set (1)
Other than above		No change in CIE5 bit's value

Set CIE4	Clear CIE4	CIE4 Bit Setting
0	1	CIE4 bit cleared (0)
1	0	CIE4 bit set (1)
Other than above		No change in CIE4 bit's value

Set CIE3	Clear CIE3	CIE3 Bit Setting
0	1	CIE3 bit cleared (0)
1	0	CIE3 bit set (1)
Other than above		No change in CIE3 bit's value

Set CIE2	Clear CIE2	CIE2 Bit Setting
0	1	CIE2 bit cleared (0)
1	0	CIE2 bit set (1)
Other than above		No change in CIE2 bit's value

Set CIE1	Clear CIE1	CIE1 Bit Setting
0	1	CIE1 bit cleared (0)
1	0	CIE1 bit set (1)
Other than above		No change in CIE1 bit's value

Set CIE0	Clear CIE0	CIE0 Bit Setting
0	1	CIE0 bit cleared (0)
1	0	CIE0 bit set (1)
Other than above		No change in CIE0 bit's value

(11) CAN0 module interrupt status register (C0INTS)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0INTS	0	0	0	0	0	0	0	0	FFFEC059H	00H
	7	6	5	4	3	2	1	0		
	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0	FFFEC058H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0INTS	0	0	0	0	0	0	0	0	FFFEC059H
	7	6	5	4	3	2	1	0	
	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0	FFFEC058H

(a) Read

CINTS5 to CINTS0	CAN interrupt Status Bit
0	The related interrupt source event is not pending.
1	The related interrupt source event is pending.

Interrupt Status Bit	Related Interrupt Source Event
CINTS5	CAN module wakeup from CAN sleep mode interrupt status[Note].
CINTS4	CAN module arbitration loss interrupt status.
CINTS3	CAN module protocol error interrupt status.
CINTS2	CAN module error state interrupt status.
CINTS1	CAN module interrupt status bit for interrupt event ‘Valid message frame reception in message buffer m’.
CINTS0	CAN module interrupt status bit for interrupt event ‘Message frame successfully transmitted from message buffer m’.

Note Only a wakeup from CAN sleep mode by CAN bus operation generates the CINTS5 signal. CAN sleep mode release by the CPU will not generate the CINTS5 signal.

(2/2)

(b) Write

Clear CINT5	CINT5 Bit Setting
0	CINT5 bit not changed
1	CINT5 bit cleared (0)

Clear CINT4	CINT4 Bit Setting
0	CINT4 bit not changed
1	CINT4 bit cleared (0)

Clear CINT3	CINT3 Bit Setting
0	CINT3 bit not changed
1	CINT3 bit cleared (0)

Clear CINT2	CINT2 Bit Setting
0	CINT2 bit not changed
1	CINT2 bit cleared (0)

Clear CINT1	CINT1 Bit Setting
0	CINT1 bit not changed
1	CINT1 bit cleared (0)

Clear CINT0	CINT0 Bit Setting
0	CINT0 bit not changed
1	CINT0 bit cleared (0)

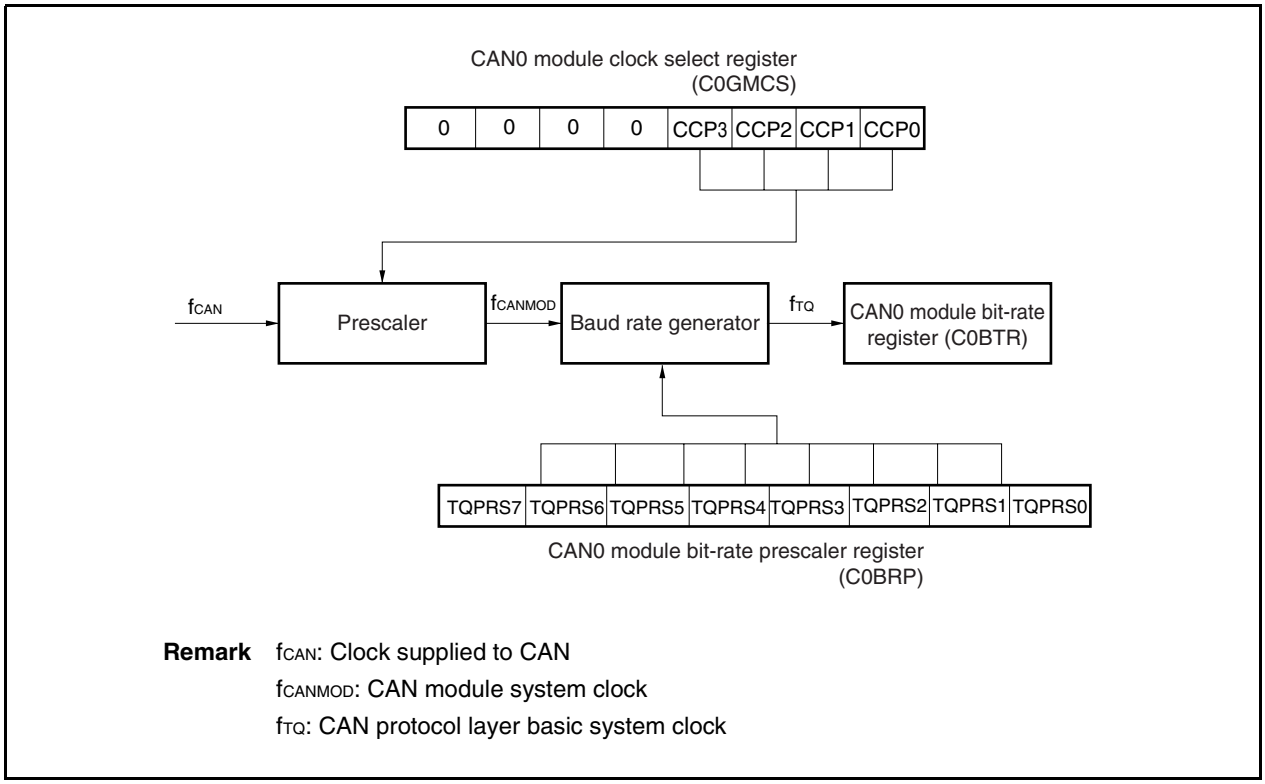
(12) CAN0 module bit-rate prescaler register (C0BRP)

	7	6	5	4	3	2	1	0	Address	Initial value
C0BRP	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0	FFFEC05AH	FFH

(Read/Write)

TQPRS7 to TQPRS0	CAN Protocol Layer Basic System Clock (f_{TQ})
0	f_{CANMOD}/1
1	f_{CANMOD}/2
n	f_{CANMOD}/(n+1)
.....	.....
255	f_{CANMOD}/256

Figure 19-28. CAN Controller Clocks



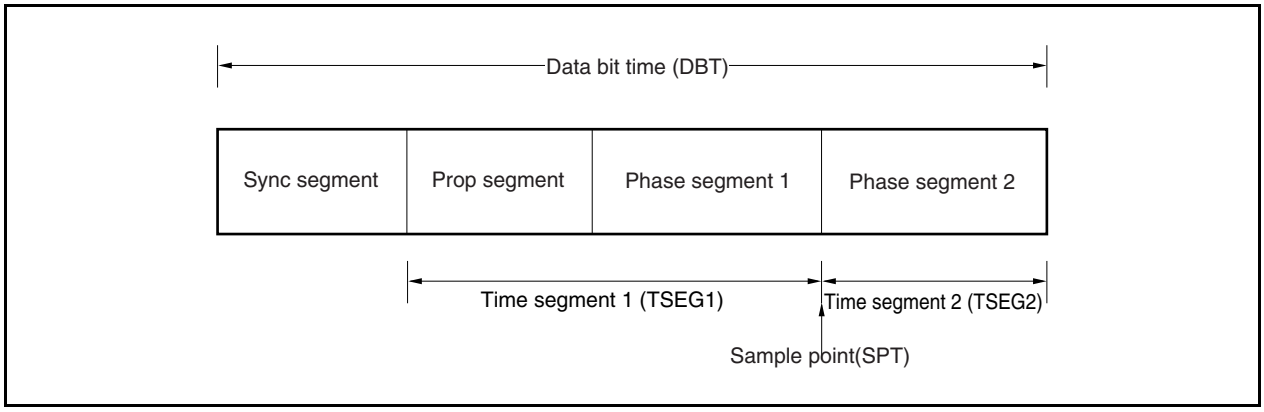
Remark f_{CAN}: Clock supplied to CAN
f_{CANMOD}: CAN module system clock
f_{TQ}: CAN protocol layer basic system clock

(13) CAN0 module bit rate register (C0BTR)

(1/2)

	15	14	13	12	11	10	9	8	Address	Initial value
C0BTR	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20	FFFEC05DH	37H
(Read/Write)	7	6	5	4	3	2	1	0		
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10	FFFEC05CH	0FH

Figure 19-29. Data Bit Time



(2/2)

SJW1	SJW0	Length of Synchronization Jump Width
0	0	1 TQ
0	1	2 TQ
1	0	3 TQ
1	1	4 TQ

TSEG22	TSEG21	TSEG20	Length of Time Segment 2
0	0	0	1 TQ
0	0	1	2 TQ
0	1	0	3 TQ
0	1	1	4 TQ
1	0	0	5 TQ
1	0	1	6 TQ
1	1	0	7 TQ
1	1	1	8 TQ

TSEG13	TSEG12	TSEG11	TSEG10	Length of Time Segment 1
0	0	0	0	Setting prohibited.
0	0	0	1	2 TQ[Note]
0	0	1	0	3 TQ[Note]
0	0	1	1	4 TQ
0	1	0	0	5 TQ
0	1	0	1	6 TQ
0	1	1	1	7 TQ
1	0	0	0	8 TQ
1	0	0	1	9 TQ
1	0	1	0	10 TQ
1	0	1	1	11 TQ
1	1	0	0	12 TQ
1	1	0	1	13 TQ
1	1	1	0	14 TQ
1	1	1	1	15 TQ

Note Setting prohibited when C0BRP = 00H.

Remark TQ = 1/f_{TQ} (f_{TQ}: CAN protocol layer basic system clock)

(14) CAN0 module last in-pointer register (C0LIPT)

	7	6	5	4	3	2	1	0	Address	Initial value
C0LIPT	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0	FFFEC05EH	Undefined

(Read-only)

LIPT7 to LIPT0	Last In-Pointer of Receive History List
0.....$m_{max.}$ Note	Reading the C0LIPT register delivers the message buffer number in which the last data frame was saved or the last remote frame was stored.

Note The maximum number of message buffers implemented in a CAN channel varies ($m_{max.}$ = 16, 32 or 48).

Remark As long as no data frame has been stored in a message buffer or no received remote frame has been stored, an undefined value is read from the C0LIPT register. The user cannot read the C0LIPT register after switching the CAN module to one of the operational modes as long as the RHPM bit is set (1).

(15) CAN0 module receive history list register (C0RGPT)

(1/2)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0RGPT	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0	FFFEC06H1H	Undefined
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	RHPM	ROVF	FFFEC06H0H	xxxxxx10B

(b) Write

	15	14	13	12	11	10	9	8	
C0RGPT	0	0	0	0	0	0	0	0	FFFEC06H1H
	7	6	5	4	3	2	1	0	
	0	0	0	0	0	0	0	Clear ROVF	FFFEC06H0H

(a) Read

RGPT7 to RGPT0	Receive History List Pointer Match
0.....$m_{max.}$ Note 1	Reading the C0RGPT register delivers the message buffer number from which the user has to read received data (receive message buffer) or to which a remote frame was received (transmit message buffer).

RHPM	Receive History List Pointer Match
0	There is no unread message left in the RHL.
1	There is at least one unread message in the RHL.

ROVF	Receive History List Overview Bit
0	No overflow of the RHL occurred. Upon data frame storage or remote frame assignment, the corresponding message buffer number is logged into the RHL.
1	The RHL is fully loaded with the unread message buffer number and all RHL elements besides the last one are preserved. The message buffer number of the subsequent data frame storage or remote frame assignment is always logged in the RHL element LIPT pointer -1 is pointing to Note 2.

Notes 1. The maximum number of message buffers implemented in a CAN channel varies ($m_{max.}$ = 16, 32 or 48).

2. The RHL will be updated, but the LIPT pointer will not be incremented. The position that the LIPT pointer −1 is pointing to is always overwritten.

(b) Write

Clear ROVF	ROVF Bit Setting
0	ROVF bit not changed
1	ROVF bit cleared (0)

(16) CAN0 module last out-pointer register (C0LOPT)

	7	6	5	4	3	2	1	0	Address	Initial value
C0LOPT	LOPT7	LOPT6	LOPT5	LOPT4	LOPT3	LOPT2	LOPT1	LOPT0	FFFEC062H	Undefined

(Read-only)

LOPT7 to LOPT0	Last Out-Pointer of Transmit History List
0.....$m_{max.}$ Note	Reading the C0LOPT register delivers the message buffer number from which the last message frame was sent.

Note The maximum number of message buffers implemented in a CAN channel varies ($m_{max.}$ = 16, 32, or 48)

Remark If no message frame has been sent, an undefined value is read from the C0LOPT register as long as the THPM bit is set (1).

(17) CAN0 module transmit history list register (C0TGPT)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0TGPT	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0	FFFEC065H	Undefined
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	THPM	TOVF	FFFEC064H	xxxxxx10B

(b) Write

	15	14	13	12	11	10	9	8	
C0TGPT	0	0	0	0	0	0	0	0	FFFEC065H
	7	6	5	4	3	2	1	0	
	0	0	0	0	0	0	0	Clear TOVF	FFFEC064H

(a) Read

TGPT7 to TGPT0	Transmit History List Pointer Match
0.....$m_{max.}$ Note 1	Reading the C0TGPT register delivers the message buffer number which can be loaded with new data for the next transmission.

THPM	Transmit History Pointer Match
0	There is at least one unread message in the THL.
1	There is no unread message in the THL.

TOVF	Transmit History List Overflow
0	Upon successful message frame transmission, the corresponding message buffer number is logged into the THL.
1	The THL is fully loaded with the unread message buffer number and all THL elements besides the last one are preserved. The message buffer number of the successfully transmitted message frame is always logged in the THL element LOPT pointer -1 is pointing to Note 2.

Notes 1. The maximum number of message buffers implemented in a CAN channel varies ($m_{max.}$ = 16, 32, or 48).

2. The THL will be updated, but the LOPT pointer will not be incremented. The position that the LOPT pointer -1 is pointing to is always overwritten.

(b) Write

Clear TOVF	ROVF Bit Setting
0	TOVF bit not changed
1	TOVF bit cleared (0)

(18) CAN0 module time stamp register (C0TS)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0TS	0	0	0	0	0	0	0	0	FFFEC067H	00H

7	6	5	4	3	2	1	0		
0	0	0	0	0	TSLOCK	TSSEL	TSEN	FFFEC066H	00H

(b) Write

	15	14	13	12	11	10	9	8	
C0TS	0	0	0	0	0	Set TSLOCK	Set TSSEL	Set TSEN	FFFEC067H

7	6	5	4	3	2	1	0	
0	0	0	0	0	Clear TSLOCK	Clear TSSEL	Clear TSEN	FFFEC066H

Remark The basic time stamp function cannot be used when the CAN module operates in 'normal operating mode with automatic block transmission'

(a) Read

TSLOCK	Time Stamp Lock Function Enable Bit
0	The time stamp lock function is disabled. The TSOUT signal is toggled upon every selected time stamp capture event.
1	The time stamp lock function is enabled. The TSOUT signal generation is locked after a data frame was successfully received in message buffer 0.

TSSEL	Time Stamp Capture Event Selection Bit
0	The time stamp capture event is the SOF event (Start-of-Frame on the CAN bus)
1	The time stamp capture event is the EOF event (the last bit of the End-of-Frame field. The TSOUT signal is generated at the sample point of the last bit in the EOF field.)

TSEN	TSOUT Operation Setting Bit
0	Time stamp signal TSOUT generation is disabled.
1	Time stamp signal TSOUT generation is enabled.

(2/2)

(b) Write

Set TSLOCK	Clear TSLOCK	TSLOCK Bit Setting
0	1	TSLOCK bit cleared (0)
1	0	TSLOCK bit set (1)
Other than above		No change in TSLOCK bit's value

Set TSSEL	Clear TSSEL	TSSEL Bit Setting
0	1	TSSEL bit cleared (0)
1	0	TSSEL bit set (1)
Other than above		No change in TSSEL bit's value

Set TSEN	Clear TSEN	TSEN Bit Setting
0	1	TSEN bit cleared (0)
1	0	TSEN bit set (1)
Other than above		No change in TSEN bit's value

(19) CAN0 message data byte register (C0MDATA x m) (x = 0 to 7, m = 0 to 31)

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA01m	MDATA01 7	MDATA01 6	MDATA01 5	MDATA01 4	MDATA01 3	MDATA01 2	MDATA01 1	MDATA01 0	FFFECxx0H FFFECxx1H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA0m	MDATA0 7	MDATA0 6	MDATA0 5	MDATA0 4	MDATA0 3	MDATA0 2	MDATA0 1	MDATA0 0	FFFECxx0H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA1m	MDATA1 7	MDATA1 6	MDATA1 5	MDATA1 4	MDATA1 3	MDATA1 2	MDATA1 1	MDATA1 0	FFFECxx1H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA23m	MDATA23 7	MDATA23 6	MDATA23 5	MDATA23 4	MDATA23 3	MDATA23 2	MDATA23 1	MDATA23 0	FFFECxx2H FFFECxx3H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA2m	MDATA2 7	MDATA2 6	MDATA2 5	MDATA2 4	MDATA2 3	MDATA2 2	MDATA2 1	MDATA2 0	FFFECxx2H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA3m	MDATA3 7	MDATA3 6	MDATA3 5	MDATA3 4	MDATA3 3	MDATA3 2	MDATA3 1	MDATA3 0	FFFECxx3H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA45m	MDATA45 7	MDATA45 6	MDATA45 5	MDATA45 4	MDATA45 3	MDATA45 2	MDATA45 1	MDATA45 0	FFFECxx4H FFFECxx5H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA4m	MDATA4 7	MDATA4 6	MDATA4 5	MDATA4 4	MDATA4 3	MDATA4 2	MDATA4 1	MDATA4 0	FFFECxx4H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA5m	MDATA5 7	MDATA5 6	MDATA5 5	MDATA5 4	MDATA5 3	MDATA5 2	MDATA5 1	MDATA5 0	FFFECxx5H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA67m	MDATA67 7	MDATA67 6	MDATA67 5	MDATA67 4	MDATA67 3	MDATA67 2	MDATA67 1	MDATA67 0	FFFECxx6H FFFECxx7H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA6m	MDATA6 7	MDATA6 6	MDATA6 5	MDATA6 4	MDATA6 3	MDATA6 2	MDATA6 1	MDATA6 0	FFFECxx6H	Undefined

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDATA7m	MDATA7 7	MDATA7 6	MDATA7 5	MDATA7 4	MDATA7 3	MDATA7 2	MDATA7 1	MDATA7 0	FFFECxx7H	Undefined

(20) CAN0 message data length code register m (C0MDLCm)

	7	6	5	4	3	2	1	0	Address	Initial value
C0MDLCm	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0	FFFECxx8H	0000xxxxB

MDLC3	MDLC2	MDLC1	MDLC0	Message Data Length Code
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
1	0	0	1	Data frame contains 8 bytes in the data field.
1	0	1	0	Data frame contains 8 bytes in the data field.
1	0	1	1	Data frame contains 8 bytes in the data field.
1	1	0	0	Data frame contains 8 bytes in the data field.
1	1	0	1	Data frame contains 8 bytes in the data field.
1	1	1	0	Data frame contains 8 bytes in the data field.
1	1	1	1	Data frame contains 8 bytes in the data field.

Caution Be sure to write 0000B to bits 7 to 4 of the C0MDLCm register.

(21) CAN0 message configuration register (C0MCONFm)

	7	6	5	4	3	2	1	0	Address	Initial value
C0MCONFm	OWS	RTR	MT2	MT1	MT0	0	0	MA0	FFFECxx9H	xxxxxxxxB

OWS	Overwrite Select Bit
0	A newly received data frame does not overwrite the occupied message buffer[Note]. The newly received data frame is discarded.
1	A newly received data frame overwrites an occupied message buffer

Note An occupied message buffer means a receive message buffer into which a data frame has already been accepted (i.e. DN is set(1)), but the CPU has not yet read that message buffer.

RTR	Remote Frame Request Bit[Note]
0	Data frame transmission
1	Remote frame transmission

Note The RTR bit only determines the message frame for the transmission process from a message buffer defined as transmit message buffer (i.e. MT2 to MT0 = 00H).

Upon valid reception of a remote frame, RTR remains cleared (0) in the corresponding transmit message buffer.

If a transmit message buffer is prepared to send a remote frame by setting the RTR bit (1) and a remote frame is received in parallel from the CAN bus, that received remote frame is not accepted by the message buffer. That is, no interrupt generation, no update of the DN flag, no update of the MDLC3 to MDLC0 bit string and no update of the receive history list will be issued for that buffer.

MT2	MT1	MT0	Message Buffer Type
0	0	0	Message buffer is a transmit message buffer.
0	0	1	Message buffer is a receive message buffer not linked to a mask in the assigned CAN I/F channel.
0	1	0	Message buffer is a receive message buffer linked to mask 1 in the assigned CAN I/F channel.
0	1	1	Message buffer is a receive message buffer linked to mask 2 in the assigned CAN I/F channel.
1	0	0	Message buffer is a receive message buffer linked to mask 3 in the assigned CAN I/F channel.
1	0	1	Message buffer is a receive message buffer linked to mask 4 in the assigned CAN I/F channel.
Other than above			Setting prohibited.

MA0	Message Buffer Assignment
0	A message buffer is not assigned to any CAN I/F channel.
1	A message buffer is assigned to CAN I/F channel 1.

Caution Be sure to set bits 2 and 1 to 0.

(22) CAN0 message identifier registers (C0MIDLm, C0MIDHm) (m = 0 to 31)

	15	14	13	12	11	10	9	8	Address	Initial value
C0MIDLm	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	FFFECxxBH	Undefined
(Read/Write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	FFFECxxAH	
	15	14	13	12	11	10	9	8		
C0GMCTRL	IDE	0	0	ID28	ID27	ID26	ID25	ID24	FFFECxxDH	x00xxxxx xxxxxxxxB
(Read/Write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	FFFECxxCH	

IDE	Identifier Extension Bit
0	11-bit standard identifier[Note].
1	29-bit extended identifier

Note The ID17 to ID0 bits are not used and may contain undefined values.

ID28 to ID0	Message Identifier
ID28 to ID18 = 0...(2^{11}–1)	Range for 11-bit standard identifier values
ID28 to ID0 = 0...(2^{29}–1)	Range for 29-bit extended identifier values

(23) CAN0 message control register m (C0MCTRLm)

(a) Read

	15	14	13	12	11	10	9	8	Address	Initial value
C0MCTRLm	0	0	MUC	0	0	0	0	0	FFFECxxFH	00x00000H 000xx000B
	7	6	5	4	3	2	1	0		
	0	0	0	MOW	IE	DN	TRQ	RDY	FFFECxxEH	

(b) Write

	15	14	13	12	11	10	9	8	
C0MCTRLm	0	0	0	0	Set IE	Set DN	Set TRQ	Set RDY	FFFECxxFH
	7	6	5	4	3	2	1	0	
	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY	FFFECxxEH

(a) Read

MUC	Message Buffer Under Change Flag
0	The assigned CAN module is not writing to the message buffer.
1	The assigned CAN module is writing to the message buffer.

MOW	Message Buffer Overwritten Flag
0	A newly received data frame has not overwritten the message buffer.
1	A newly received data frame has overwritten the message buffer.

(2/3)

IE	Interrupt Enable for Message Buffer Interrupt Event
0	Interrupt generation is disabled for the following events: • Interrupt events linked to the CINTS0 interrupt status bit in the C0INTS register (i.e. when MT2 to MT0 = 0, 'Data frame successfully transmitted from message buffer m', 'Remote frame successfully transmitted from message buffer m') • Interrupt events linked to the CINTS1 interrupt status bit in the C0INTS register (i.e. when MT2 to MT0 = 1, 2, 3, 4 or 5, 'Valid data frame reception in message buffer m' and when MT2 to MT0 = 0 'Valid remote frame reception in message buffer m')
1	Interrupt generation is enabled for the following events: • Interrupt events linked to the CINTS0 interrupt status bit in the C0INTS register (i.e. when MT2 to MT0 = 0, 'Data frame successfully transmitted from message buffer m', 'Remote frame successfully transmitted from message buffer m') • Interrupt events linked to the CINTS1 interrupt status bit in the C0INTS register (when MT2 to MT0 = 1, 2, 3, 4 or 5, 'Valid data frame reception in message buffer m' and when MT2 to MT0 = 0 'Valid remote frame reception in message buffer m.')

DN	Message Buffer Data New Bit
0	No data frame has been stored in the message buffer (message buffer is defined as receive message buffer (MT2 to ME0 > 00H)). No remote frame has been stored in the message buffer (message buffer is defined as receive message buffer (MT2 to ME0> 00H)).
1	A data frame has been stored in the message buffer (message buffer is defined as receive message buffer (MT2 to ME0 > 00H)). A remote frame has been stored in the message buffer (message buffer is defined as receive message buffer (MT2 to ME0> 00H)).

Caution Be sure not to set the DN flag (1) by software.

TRQ	Message Buffer Transmit Request Bit
0	No message frame transmission request is pending or ongoing from the message buffer.
1	A message frame transmission request is pending or a message frame transmission is ongoing from the message buffer.

RDY	Message Buffer Ready Bit
0	The CPU can write to the message buffer. The assigned CAN module does not access the message buffer.
1	The assigned CAN module accesses the message buffer. CPU write access to the message buffer is ignored (except write access to the RDY bit, TRQ bit, DN bit and MOW bit).

(3/3)

(b) Write

Clear MOW	MOW Bit Setting
0	MOW bit not changed
1	MOW bit cleared (0)

Set IE	Clear IE	IE Bit Setting
0	1	IE bit cleared (0)
1	0	IE bit set (1)
Other than above		No change in IE bit's value

Set DN	Clear DN	DN Bit Setting
0	1	DN bit cleared (0)
1	0	DN bit set (1)
Other than above		No change in DN bit's value

Set TRQ	Clear TRQ	TRQ Bit Setting
0	1	TRQ bit cleared (0)
1	0	TRQ bit set (1)
Other than above		No change in TRQ bit's value

Set RDY	Clear RDY	RDY Bit Setting
0	1	RDY bit cleared (0)
1	0	RDY bit set (1)
Other than above		No change in RDY bit's value

19.7 Bit Set/Clear Function

The CAN control registers include registers whose bits can be set or cleared via the CPU and via the CAN interface. An operation error occurs if the following registers are written directly. Do not write any values directly via bit manipulation, read/modify/write, or direct writing of target values.

- CAN0 control register (C0GMCTRL)
- CAN0 automatic block transmission register (C0GMABT)
- CAN0 module control register (C0CTRL)
- CAN0 module interrupt enable register (C0IE)
- CAN0 module interrupt status register (C0INTS)
- CAN0 module receive history list register (C0RGPT)
- CAN0 module interrupt history list register (C0TGPT)
- CAN0 module time stamp register (C0TS)
- CAN0 message control register (C0MCTRLm)

Remark m = Message buffer number (0 to 31)

All 16 bits in the above registers can be read via the usual method. Use the procedure described in Figure 19-30 below to set or clear the lower 8 bits in these registers.

Setting or clearing of the lower 8 bits in the above registers is performed in combination with the higher 8 bits (see **Figure 19-31**). Figure 19-30 shows how the values of set bits or clear bits relate to set/clear/no change operations in the corresponding register.

Figure 19-30. Example of Bit Setting/Clearing Operations

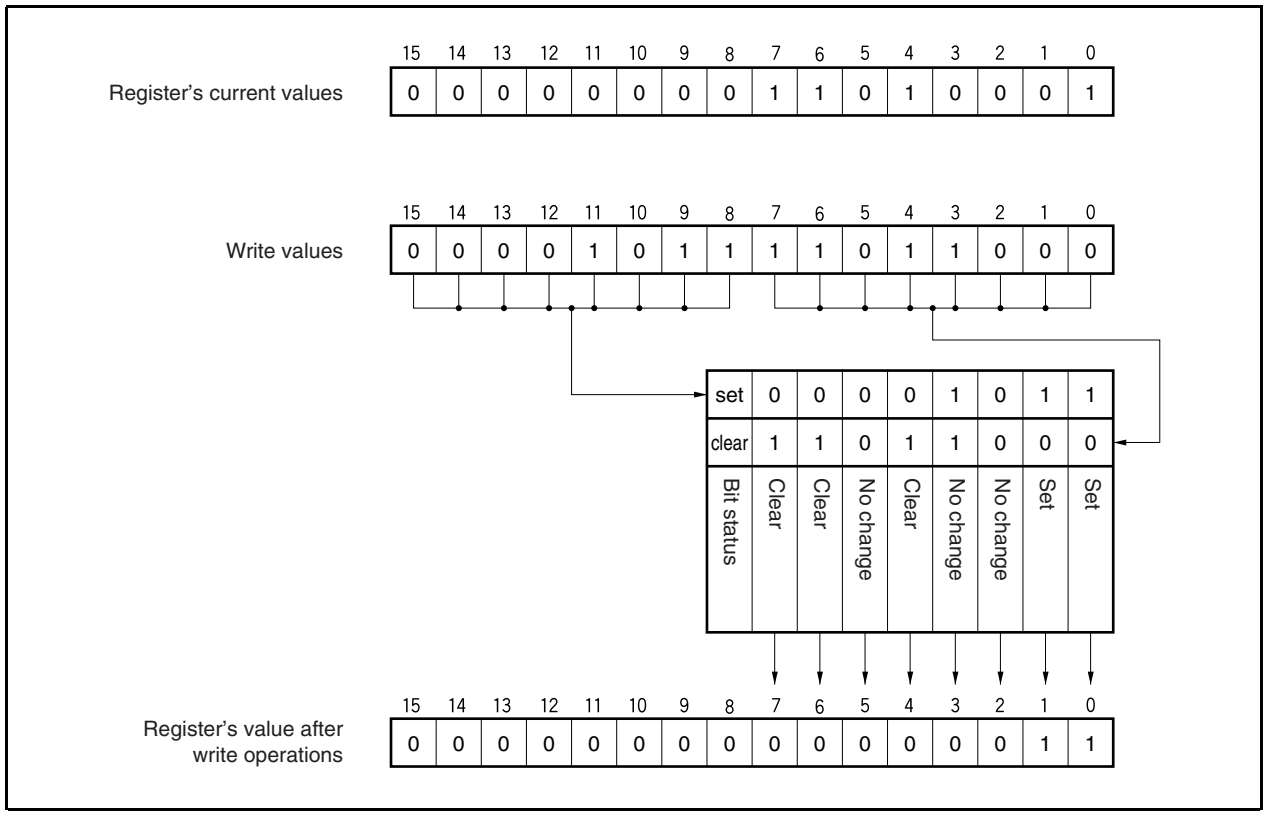


Figure 19-31. 16-Bit Data During Write Operation

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 bit n after bit set/clear operation
0	0	No change
0	1	0
1	0	1
1	1	No change

19.8 CAN Controller Initialization

19.8.1 Initialization of CAN module

Before the CAN module operation is enabled, the CAN module system clock needs to be set in CCP[3:0] of the C0GMCS register by software. Do not change the setting of the CAN module system clock after CAN module operation is enabled. The CAN module is enabled by setting (1) the GOM bit of the C0GMCTRL register.

Regarding initialization processing, see **19.16 Operation of CAN Controller**

19.8.2 Initialization of message buffer

After the CAN module is enabled, the message buffers contain undefined values. A minimum initialization for all message buffers, even for those not used in the application, is necessary before switching the CAN module from INIT mode to one of the operational modes.

The minimum initialization has to include the proper configuration of the following bits and bit strings to avoid unexpected behavior from the CAN module in the operational modes.

Table 19-22. Minimum Configuration of Message Buffer Even When Unused in Application

Register	Bit	Minimum Initialization Value
C0MCTRLm	RDY	0B
C0MCTRLm	TRQ	0B
C0MCTRLm	DN	0B
C0MCONFm	MA0	0B

19.8.3 Transition from INIT mode to operational mode

The CAN module in each CAN I/F channel can be switched to the following operational modes.

“Normal operating mode”

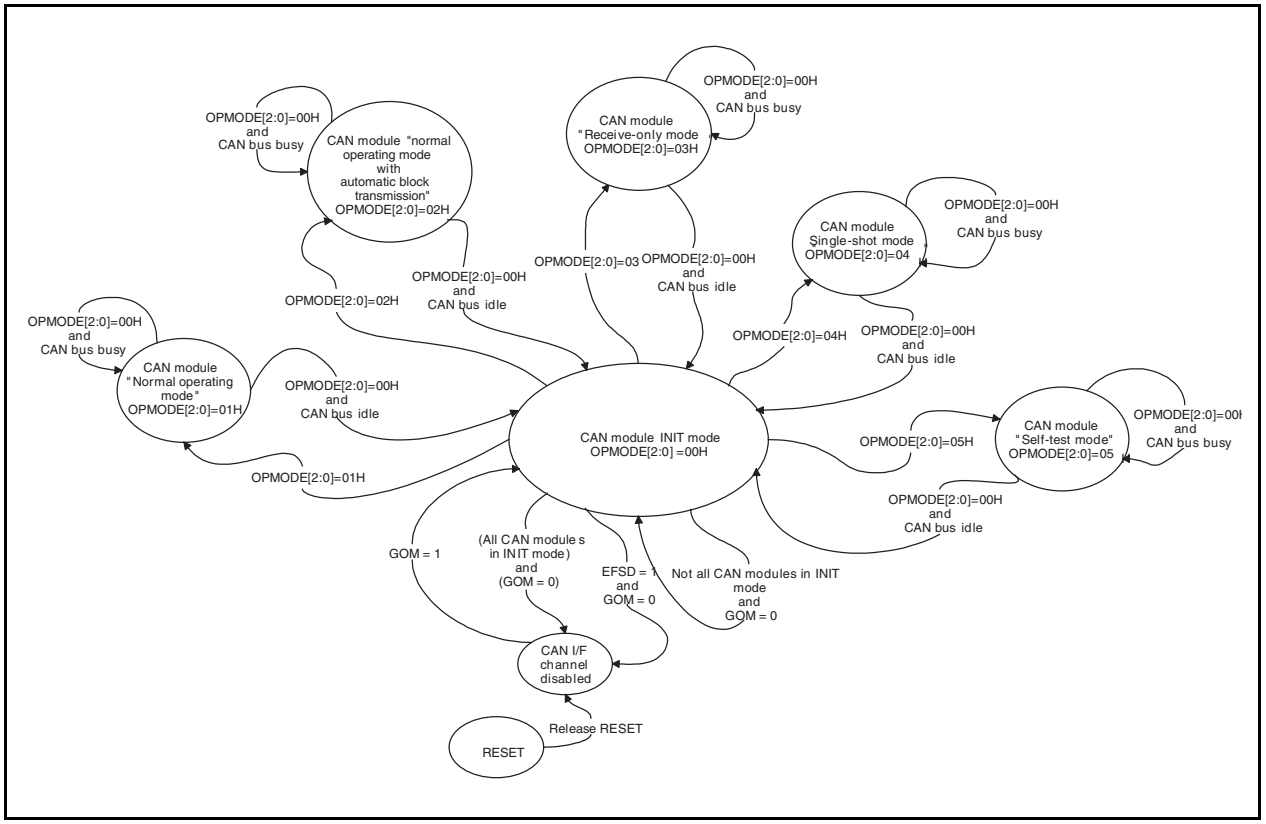
“Normal operating mode with automatic block transmission”

“Receive-only mode”

“Single-shot mode”

“Self-test mode”

Figure 19-32. Transition to CAN Module Operational Modes



The transition from INIT mode to the operational modes is controlled by the bit strings OPMODE2 to OPMODE0 in the C0CTRL register.

Changing from one operational mode to another requires shifting to INIT mode in between. The CAN module refuses CPU attempts to change from one operational mode to another directly.

Transition requests from the operational modes to the INIT mode are not directly accepted by the CAN module when the CAN bus is not idle (i.e. frame reception or transmission is ongoing), but it is kept until the CAN module detects the first bit of intermission. As soon the above mentioned condition is detected, the transition from the operational mode to the INIT mode is executed and the OPMODE2 to OPMODE0 bit string values change to 00H. The CPU has to confirm the proper transition to INIT mode by reading the OPMODE2 to OPMODE0 bit strings until OPMODE2 to OPMODE0 = 000B.

19.8.4 Resetting of CAN module error counter C0ERC in INIT mode

For evaluation purposes, it is necessary to reset the CAN module error counter C0ERC and the CAN module information register C0INFO. Therefore it is possible to set the CCERC bit in the C0CTRL register (1) in INIT mode of the CAN module. As a result, the CAN module error counter C0ERC and the CAN module information register C0INFO register are cleared to their default values when the CAN module shifts to any operational mode.

19.9 Message Reception

19.9.1 Message reception

In all the operational modes of the CAN module, when a data frame is received, whether the received data frame has to be stored in one of the message buffers that satisfy the following conditions is checked.

- The message buffer has to be assigned to receive data frames (MA0 bit set (1) in C0MCONFm register)
- The message buffer has to be configured as a receive message buffer (MT[2:0] bit string in C0MCONFm register has to hold the values 001B, 010B, 011B, 100B or 101B)
- The message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in C0MCTRLm register)

When two or more message buffers of the CAN module receive a message, the message (data frame or remote frame) is always stored in the receive message buffer with the highest priority.

For example, when an unmasked receive message buffer and a message buffer linked to mask 0 have the same identifier, the message is always stored in the unmasked receive message buffer even if this unmasked receive message buffer has already received a message earlier (DN flag is set already).

Priority	Conditions
1 (highest)	Unmasked message buffer
2	Message buffer linked to mask 1
3	Message buffer linked to mask 2
4	Message buffer linked to mask 3
5 (lowest)	Message buffer linked to mask 4

19.9.2 Receive history list function

The receive history list (RHL) function records the number of the receive message buffer in which each data frame or remote frame was stored. The RHL consists of 23 elements and two pointers, the last in-message pointer (LIPT pointer) with the corresponding C0LIPT register and the receive history list get pointer (RGPT pointer) with the corresponding C0RGPT register. After the transition from INIT mode to one of the operational modes, the RHL elements contain undefined values.

The LIPT pointer is utilized as a write pointer for the message buffer numbers stored in the RHL. Anytime a data frame or remote frame is stored, a message buffer number is recorded to the RHL element. The LIPT pointer points to this element. The LIPT pointer is incremented automatically when new elements are entered in the RHL. In this way, the number of message buffers that received new messages recently will be recorded chronologically.

The RGPT pointer is utilized as a read pointer for message buffer numbers stored in the RHL. The RGPT pointer points to the first RHL element that the CPU did not read yet. A stored message buffer number is read by reading the C0RGPT register. The RGPT pointer is incremented automatically each time a message buffer number is read from the C0RGPT register.

The receive history list pointer match bit (RHPM bit) in the C0RGPT register is set (1) whenever the RGPT pointer matches the LIPT pointer. The RHPM bit set (1) signals to the application software that no more message frames have been received. When a new message buffer number is recorded, the LIPT pointer is incremented again, and the RHPM bit is cleared (0). In other words, the message buffer numbers that are stored in RHL indicate unread message buffers.

The receive history list overflow bit (ROVF bit) is set (1) from the CAN module whenever the LIPT pointer points to the RGPT pointer-1. In this state, the RHL is completely filled with message buffer numbers that have not been read yet. When further reception of messages occurs, the last record in RHL is overwritten by the buffer number of the

newly stored message. Therefore, after the ROVF bit is set (1), a recorded message buffer number in the RHL does not completely reflect the chronological order.

Figure 19-33. Receive History List

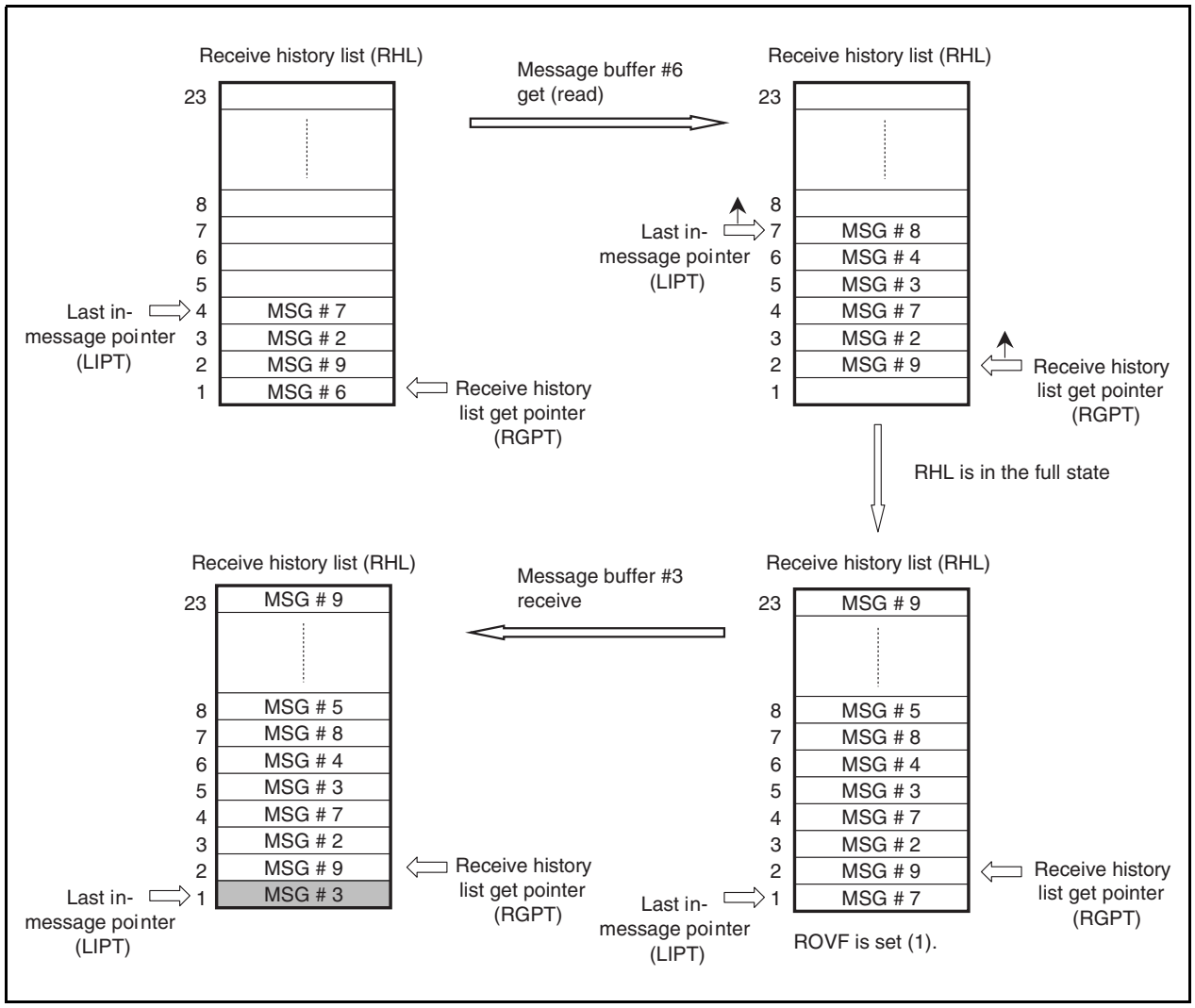


Figure 19-34. Message Reception Procedure Using Receive History List

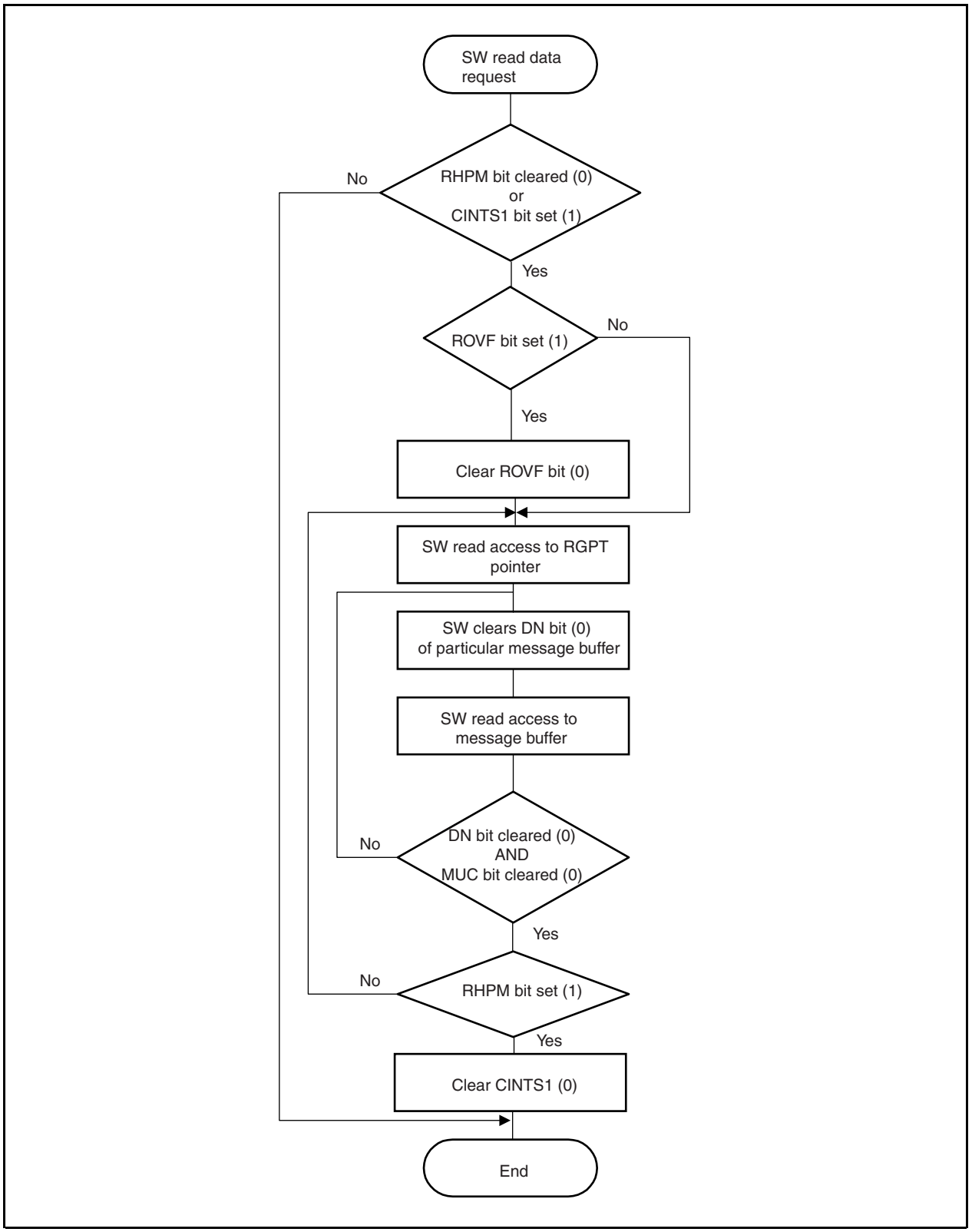
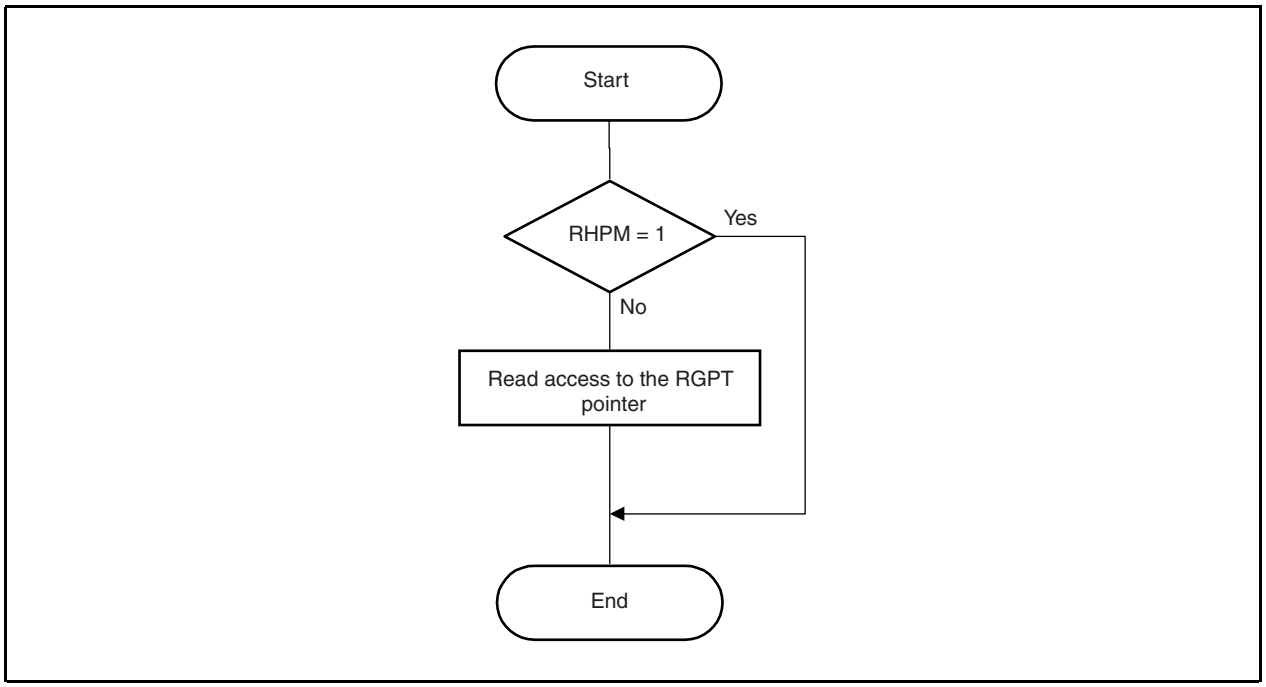


Figure 19-35. C0RGPT Pointer Handling with Respect to RHPM Bit



19.9.3 Mask function

A mask function can be linked to each receive message buffer.

This means that there is no need to distinguish between local masks and global masks.

When the mask function is used, the identifier of a received message is compared with the identifier of the particular message buffer. Bits in the mask configured as “don’t care” will prevent any comparison. Thus, the respective bits of the identifier of the receive message will be stored regardless of the corresponding values in the message buffer.

When the mask function is linked to a receive message buffer, a bit whose value is defined as “1” by masking is not subject to the above-mentioned comparison between the identifier of the received message and the identifier in the message buffer.

However, this comparison is performed for any bit whose value is defined as “0” by the mask.

For example, let us assume that all messages that have a standard-format identifier, in which bits ID27 to ID25 = 0 and bits ID24 and ID22 = 1, are to be stored in message buffer 14. The procedure for this example is shown below.

EXAMPLE:

<1> Identifier bits to be stored in message buffer 14

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.

<2> Identifier bits to be configured in message buffer 14 (example)

(Using CAN0 message ID registers L14 and H14 (C0MIDL14 and C0MIDH14))

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

x = don’t care

Message buffer 14 is set as a standard-format identifier linked to mask 1.

<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, bits CMID27 to CMID24 and CMID22 = 0 and bits CMID28, CMID23, and CMID21 to CMID0 = 1.

19.9.4 Multi buffer receive block function

Two or more receive message buffers can be grouped as a multi buffer receive block (MBRB) by setting the same ID to two or more message buffers. The MBRB can store two or more data frames from the CAN bus without overwriting previously received messages. Support by the CPU is not necessary for this operation.

The user can determine which data block reception completion the CPU has to be informed of by setting the IE bit in the C0MCTRLm register of the message buffer. If a data block consists of message k, the user could initialize the message k buffer for the reception of the data block. In message buffers 0 to (k-1) the IE bits are cleared (0) (i.e. interrupts disabled) and in message buffer k the IE bit is set (1) (interrupts enabled). This kind of configuration establishes a ring buffer that provides received messages to the CPU in a FIFO manner when the CPU reads the messages before the MBRB overflows.

19.9.5 Remote frame reception

In the operational modes “normal operating mode”, “normal operating mode with ABT”, “receive-only mode”, “single-shot mode” and “self-test mode” the receive message acceptance filtering machine evaluates all message buffers that satisfy the following conditions as to whether the received remote frame should be accepted.

- The message buffer has to be assigned to the CAN I/F channel (MA0 bit in C0MCONFm register is set to 1B or larger).
- The message buffer has to be configured as a transmit message buffer (MT2 to MT0 bits in C0MCONFm register are set to 000B).
- The message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in C0MCTRLm register)
- The RTR bit in the transmit message buffer has to be cleared (0).
- The TRQ bit in the transmit message buffer has to be cleared (0).

Upon acceptance of a remote frame, the following actions are executed if the identifier of the received remote frame matches the identifier of a message buffer that satisfies the above conditions.

- The MDLC[3:0] bit string in the C0MDLCm register is overwritten by the DLC value of the received remote frame.
- The DN flag is set (1)
- The interrupt status bit CINTS1 in the C0INTS register is set (1)

- When it has been enabled by setting the CIE1 enable bit in the C0IE register (1), the interrupt request signal INTC0TREC is generated.
- The reception history list is updated with the message buffer number for which the received remote frame has been accepted
- The setting of the OWS bit in the C0MCONFm register has no meaning for the acceptance of a remote frame. It means the remote frame is accepted regardless of whether the OWS bit is cleared (0) or set (1) and the DN flag has already been set (1).

If more than one transmit message buffer with the same identifier is assigned to a CAN I/F channel, the remote frame acceptance evaluation is just executed for the transmit message buffer with the lowest message buffer number.

19.10 Message Transmission

19.10.1 Message transmission

In the operational modes “normal operating mode”, “normal operating mode with ABT” and “single-shot mode” and “self-test mode” of the CAN module, the transmit message search machine is triggered when the TRQ bit is set to 1 in a message buffer that satisfies the following conditions.

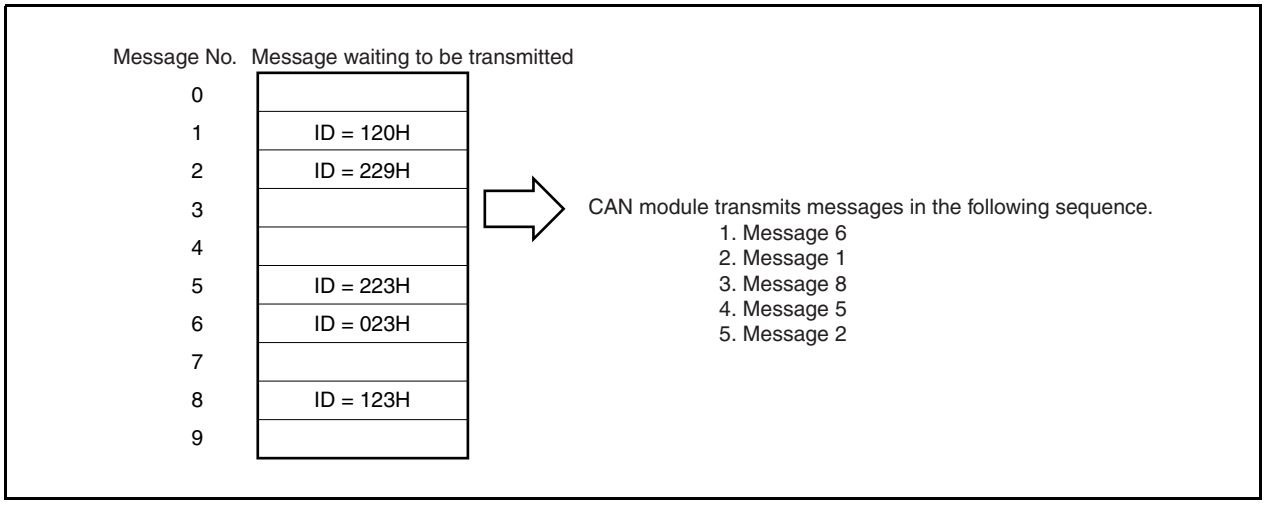
- The message buffer has to be assigned to the particular CAN I/F channel (MA0 bit in C0MCONFm register is set to 1B or larger).
- The message buffer has to be configured as a transmit message buffer (MT2 to MT0 bits in C0MCONFm register are set to 000B).
- The message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in C0MCONFm register).

The CAN 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 priority identifiers and automatically determine which message has to be sent out first.

This eliminates the need for software-based priority control.

Figure 19-36. Message Processing Example (When PBB Bit = 0)



The transmit message search machine compares the message buffer that has a new transmit request with the message buffers in which the TRQ bit is set to 1.

If the new transmit request has the highest priority, the transmit message search machine overwrites the temporary transmit buffer with the message frame, which is stored in the corresponding message buffer. Overwriting is only possible as long the transmit process has not already been started for the message frame currently occupying the temporary transmit buffer. If overwriting of the temporary transmit buffer is impossible, the new transmit request is sent at a later point in time.

The highest priority is determined according the following rules.

Priority	Conditions	Description
1 (highest)	11 MSB identifier value rule [ID28 to ID18]	The first 11 bits of the identifier (i.e. ID28 to ID18) are the first criteria used to judge which message frame has to be sent first. As a result, message frames with the lowest value represented by the 11 most significant bits of the identifier have to be sent first. 11-bit standard identifiers have a higher priority than message frames with 29-bit extended identifiers if the value of the 11-bit standard identifier is equal to or smaller than the 11 most significant bits (11 MSB) of the 29-bit extended identifier.
2	Frame type rule	The priority 1 rule does not provide an unambiguous result when the 11 most significant bits of the identifier are equal. Then the frame type represented by the RTR bit of 11-bit standard identifier message frames and the SRR bit of the 29-bit extended identifier message frames is the next criteria to judge which message frame has to be sent first. In this case data frames with 11-bit standard identifiers (i.e. RTR bit cleared (0)) have a higher priority than remote frames with a standard identifier and message frames with an extended identifier.
3	Identifier type rule	If even the priority 2 rule cannot deliver an unambiguous result, the identifier type represented by the IDE bit is the next criteria in the decision process. In this case a standard identifier message frame (i.e. IDE bit cleared (0)) has a higher priority than a message frame with an extended identifier.
4	18 LSB identifier value rule [ID17 to ID0]	The next criteria to find the message that has to be sent first are the 18 least significant bits of the extended identifier. The message frame with the lowest value represented by those bits is sent first. The priority 4 rule applies in cases where two transmission-pending extended identifier message frames have equal values in the 11 most significant bits of the identifier and have the same frame type (the RTR bit value is the same).
5 (lowest)	Message buffer number	The last criteria to find the message that has to be sent first is the message buffer number. The priority 5 criteria applies when 2 or more message buffers try to send message frames with exactly the same identifier. In this case the message from the message buffer with the lowest message buffer number is sent first.

Remark When the CAN module operates in “normal operating mode with ABT” only one message buffer from the ABT message buffer group has the TRQ bit set to 1 at a time. This buffer competes with transmit message buffers, which do not belong to the ABT message buffers.
Within the ABT message buffers, a fixed order determines which message buffer is sent next.

Upon successful transmission of a message frame, the TRQ flag in the corresponding transmit message buffer is automatically cleared (0) and the corresponding “transmit successful” interrupt status bit CINTS0 in the C0INTS register is set (1). In addition, an interrupt request signal will be released, when it has been enabled by setting the CIE0 enable bit in the C0IE register (1).

19.10.2 Transmit history list function

The transmit history list (THL) function records the number of the message buffer for each transmitted message (data frame or remote frame). The THL consists of 7 elements and two pointers, the last out-message pointer (LOPT pointer) with the corresponding C0LOPT register and the transmit history list get pointer (TGPT pointer) with the corresponding C0TGPT register.

After the transition from INIT mode to one of the operational modes, the THL elements contain undefined values.

The LOPT pointer is utilized as a write pointer for message buffer numbers in the THL. When a data frame or remote frame was transmitted successfully, a message buffer number is recorded to the THL element referenced by the LOPT pointer. When the record to the THL is completed, the LOPT pointer is incremented automatically. In this way, message buffer numbers of transmitted messages are recorded chronologically.

The TGPT pointer is utilized as a read pointer for message buffer numbers in the THL. The TGPT pointer points to the first THL element that CPU has not read yet. A message buffer number is read by reading the C0TGPT register. The TGPT pointer is incremented automatically anytime a message buffer number from the C0TGPT register is read.

The transmit history list pointer match bit (THPM bit) in the C0TGPT register is set (1) whenever the TGPT pointer matches the LOPT pointer. The THPM bit setting (1) signals to the application software that no more message frames have been sent. When a new message is sent, the LOPT pointer is incremented, the pointers do not match anymore, and the THPM bit is cleared (0). In other words, numbers of message buffers that have not been read yet are stored in the THL.

The transmit history list overflow bit (TOVF bit) is set (1) from the CAN module whenever the LOPT pointer points to the TGPT pointer-1. In this state, the THL is completely filled with message buffer numbers that have not been read yet. When further transmission occurs, the last entry in the THL is overwritten, and a record of the message buffer number from where the last transmission was executed is stored. Therefore, after the TOVF bit is set (1), a recorded message buffer number in the THL does not completely reflect the chronological order.

Figure 19-37. Transmit History List

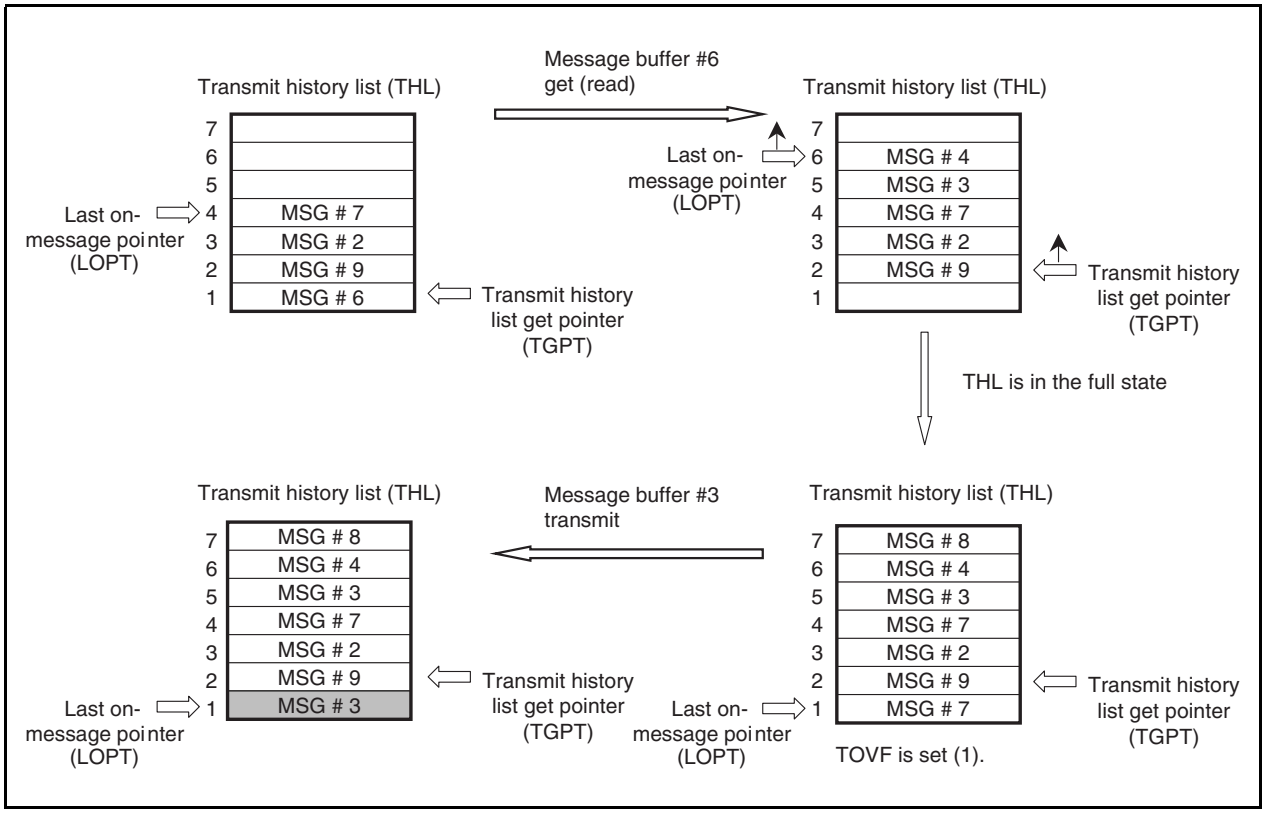


Figure 19-38. Message Transmission Procedure Using Transmit History List

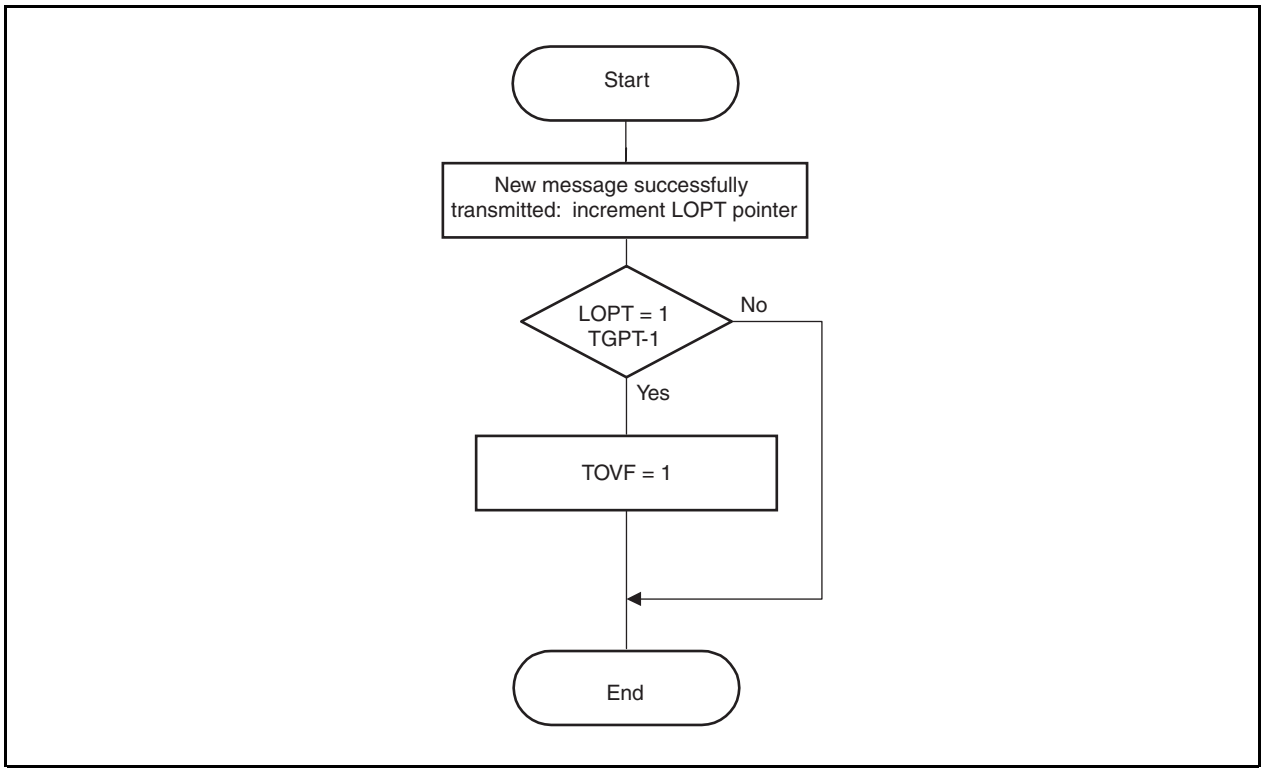
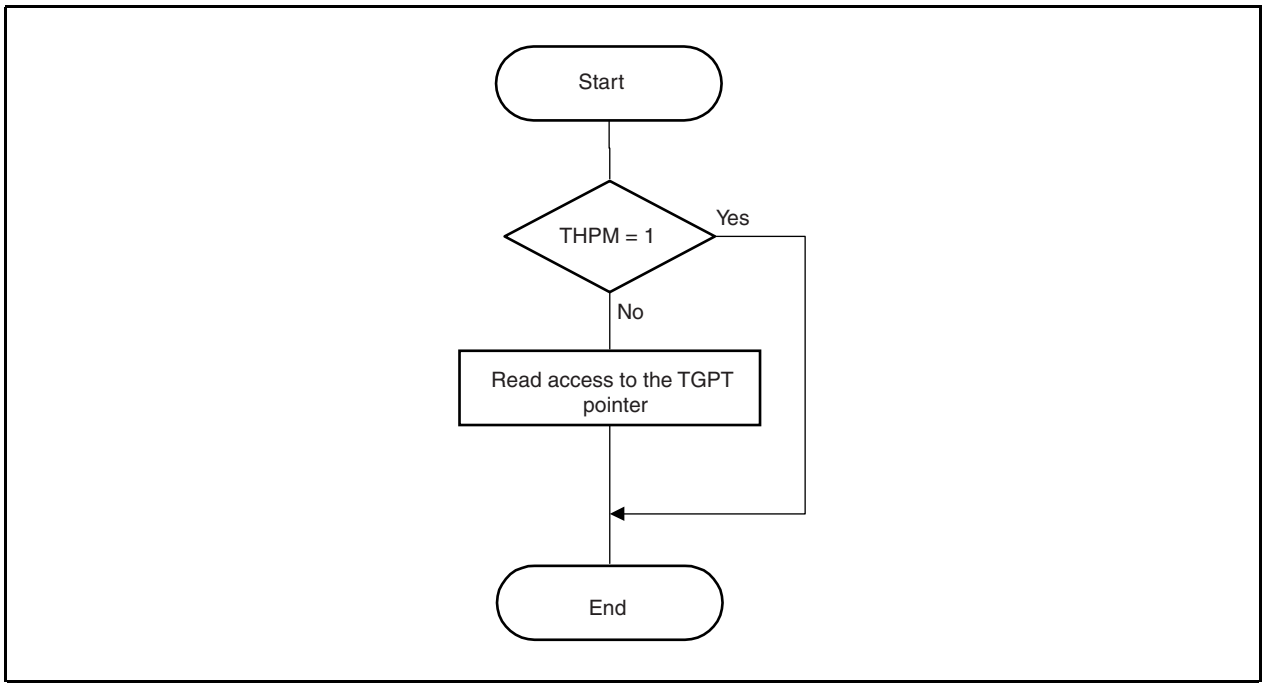


Figure 19-39. C0TGPT Pointer Handling with Respect to THPM Bit



19.10.3 Automatic block transmission (ABT)

The automatic block transmission (ABT) function can transfer data frames successively with no CPU interaction. The maximum number of transmit message buffers assigned to the ABT function is 8 (message buffer numbers 0 to 7). The ABT function is selected by setting the OPMODE[2:0] bit string.

For the ABT function, the message buffers must be initialized before a transmission request by the CPU is submitted in the ABT mode. For all message buffers used for ABT, the CAN module assignment (MAx) has to be set and the MT[2:0] bits have to be set to 000B. If several data blocks are sent in separate messages but with the same identifier, all message buffers in the ABT area have to be set with the same identifier. If two or more data frames with different identifiers are sent, the identifier needs to be set by the C0MIDLm/C0MIDHm register. The C0MDLCm register and the C0MDATA0m to C0MDATA7m registers have to be set before a transmission request for the ABT function is submitted. After initialization of a message buffer for ABT is finished, the RDY bit needs to be set (1). In the ABT mode, the TRQ bit does not have to be manipulated by software.

After all the data for the message buffer in the ABT area has been prepared, the ABTTRG bit can be set (1). Then the automatic block transmission of the message buffer with RDY = 1 is started. The ABT function sets TRQ of the first message buffer (message buffer 0) automatically, and the transmission of a message is started. After the transmission of message buffer 0 has finished, TRQ of message buffer 1 is set (1) automatically, and the message is transmitted. However the ABT function enables delaying the setting of the next TRQ automatically by a programmable period. This delay is defined by the C0GMABTD register. The unit of the LSB of that register is the DBT (data bit time). The DBT depends on the setting in the C0BRP register and the C0BTR register.

When the ABT function encounters a message buffer with RDY = 0, the ABTTRG bit is cleared (0) and the ABT operation is finished. In this situation, the RDY bit of the buffer where the ABT mode stopped can be set and the ABTTRG bit can be set (1) again. Then the ABT function will continue the ABT mode from the message buffer at which it stopped previously. If it is not necessary to continue the ABT mode at the buffer where the ABT mode previously stopped, clear (0) the ABTTRG bit via the ABTCLR bit. The internal ABT engine is reset by when the ABTCLR bit is set (1). If ABTTRG is set (1) again now, the transmission is started from message buffer 0.

The transmission of data frames from all the message buffers in the ABT area can be controlled by using the transmission complete interrupt. The IE bit of the C0MCTRLm register of each message buffer in use except the last message buffer needs to be cleared (0). In this case, the application will get a transmission complete interrupt when all messages in the ABT area have been sent.

Normally, the transmission of message buffers except those used by the ABT function (message buffer 8 to n_{max}) is defined by the priority of the identifier. In this case the sequence of transmitted messages is evaluated between the identifier of the message buffer in the ABT area waiting for transmission and all other identifiers not belonging to the ABT area.

When the ABT function is activated, the transmission of data frames from message buffers in the ABT area is not recorded in the transmission history list (THL).

19.10.4 Transmission request abort process

(1) Transmission request abort in normal operation mode

The user can clear the TRQ bit in the C0MCTRLm register to abort a transmit request. The TRQ bit will be cleared immediately if the abort was successful. Whether the message was really transmitted or not can be checked using the TSTAT bit in the C0CTRL register or the THL.

(2) Transmission request abort in normal operation mode with automatic block transmission (ABT)

It can become necessary to abort an already started automatic block transmission (ABT). In this case, the user has to clear the ABTTRG bit in the CGMABT register (0).

If the last transmission was successful, the ABT mode is left with the internal ABT pointer pointing to the next message buffer to be transmitted.

In the case of an erroneous transmission, the position of the internal ABT pointer depends on the status of the TRQ bit of the last transmitted message buffer.

- If the TRQ bit was cleared (0) in addition to the clear of ABTTRG (0), the internal ABT pointer points to the next message buffer.
- If the TRQ bit remains set at the time when ABTTRG is cleared(0), the internal ABT pointer points to the last transmitted message buffer.

In the case of a restart of the ABT mode (ABTTRG is set (1)), the next message to be transmitted can be determined from the following table.

TRQ	Abort After Successful Transmission	Abort After Erroneous Transmission
Set (1)	Next message buffer in the ABT area[Note]	Same message buffer in the ABT area
Cleared (0)	Next message buffer in the ABT area[Note]	Next message buffer in the ABT area[Note]

Note If the last message buffer in the ABT area (buffer 7) is reached or if all the subsequent buffers in the ABT area have their RDY bit cleared (0), the internal ABT pointer points to buffer 0.

(3) Transmission of remote frames

Remote frames are only sent from message buffers defined as transmit message buffers.

To distinguish a transmission request for a data frame and for a remote frame, the RTR bit in the C0MCONFm register has to be programmed accordingly. Setting (1) the RTR bit defines a remote frame transmission request.

19.11 Power Saving Modes

19.11.1 CAN sleep mode

The CAN sleep mode can be used to set the CAN controller to standby mode in order to reduce power consumption. The CAN sleep mode can be entered from all operational modes of the CAN module. A release of the CAN sleep mode returns the CAN module to exactly the same operational mode from which the CAN sleep mode was entered.

In the CAN sleep mode, the CAN module does not transmit messages, even when transmit requests are issued or pending.

(1) Entering CAN sleep mode

The CPU issues a CAN sleep mode transition request by writing 01B to the PSMODE1, PSMODE0 bit string in the C0CTRL register.

That transition request is only accepted under the following conditions.

(i) The CAN module is already in one of the following operational modes.
- "Normal operating mode"
- "Normal operating mode with automatic block transmission"
- "Receive-only mode"
- "Single-shot mode"
- "Self-test mode"
- "CAN Stop mode"

(ii) The CAN bus state is bus idle.

If one of the conditions mentioned above is not met, the CAN module will operate as follows.

- If CAN sleep mode is requested from INIT mode, the CAN sleep mode transition request is ignored and the CAN module remains in INIT mode.
- If the CAN bus state is not bus idle (i.e., the CAN bus state is either "transmitting" or "receiving") when CAN sleep mode is requested, immediate transition to CAN sleep mode is not possible. The CAN sleep mode transition request has to be held pending until the CAN bus state becomes bus idle. In the time from CAN sleep mode transition request to successful transition, the PSMODE1, PSMOD0 bit string remains 00H.

(2) Releasing CAN sleep mode

The CAN sleep mode is released by the following events.
- When the CPU writes 00B to the PSMODE1, PSMODE0 bit string in the C0CTRL register.
- A falling edge on the CAN0RX pin (i.e., the CAN bus level shifts from recessive to dominant)

After releasing the CAN sleep mode, the CAN module returns to the operational mode from which CAN sleep mode was requested and PSMODE1, PSMODE0 in the C0CTRL register are reset to 00B.

The interrupt status bit CINTS5 in the C0INTS register is also set (1), if the interrupt enable bit CIE5 is set (1). When INIT mode is requested while the CAN module is in CAN sleep mode, that request is ignored; the CPU has to release sleep mode first before entering the INIT mode.

19.11.2 CAN stop mode

The CAN stop mode can be used to set the CAN controller to standby mode to reduce power consumption, but without the ability to wake up again autonomously. The CAN stop mode can only be entered from the CAN sleep mode of the CAN module. A release of the CAN stop mode puts the CAN module in the sleep mode. In the CAN stop mode, the CAN module does not transmit messages, even when transmit requests are issued or pending.

(1) Entering CAN stop mode

The CPU issues a CAN stop mode transition request by writing 10B to the PSMODE1, PSMODE0 bit string in the C0CTRL register. The CAN stop mode transition request is only accepted when the CAN module is in CAN sleep mode. In all other modes, the CAN stop mode transition request is ignored.

(2) Releasing CAN stop mode

The CAN stop mode can only be released by writing 01B to the PSMODE1, PSMODE0 bit string in the C0CTRL register.

When INIT mode is requested while the CAN module is in CAN stop mode, that request is ignored; the CPU has to release stop mode and subsequently CAN sleep mode before entering the INIT mode.

19.12 Interrupt Function

19.12.1 Interrupts generated by CAN module

Each CAN module of a CAN I/F channel provides 6 different interrupt sources.

The occurrence of these interrupt sources is stored in interrupt status registers. Four separate interrupt request signals are generated from the six sources. After determination of the interrupt source, the user has to clear the corresponding interrupt status bit.

Table 19-23. List of CAN Module Interrupt Sources

No.	Interrupt Status Bit		Interrupt Enable Bit		Interrupt Request Signal	Interrupt Source Description
	Name	Register	Name	Register		
1	CINTS0	C0INTS	CIE0[Note]	C0IE	INTC0TRX	CAN module interrupt status bit for interrupt event 'Message frame successfully transmitted from message buffer m'
2	CINTS1	C0INTS	CIE1[Note]	C0IE	INTC0REC	CAN module interrupt status bit for interrupt event 'Valid message frame reception in message buffer m'
3	CINTS2	C0INTS	CIE2	C0IE	INTC0ERR	CAN module error state interrupt status
4	CINTS3	C0INTS	CIE3	C0IE		CAN module protocol error interrupt status
5	CINTS4	C0INTS	CIE4	C0IE		CAN module arbitration loss interrupt status
6	CINTS5	C0INTS	CIE5	C0IE	INTC0WUP	CAN module wakeup from CAN sleep mode by CAN bus interrupt status bit

Note The IE bit (message buffer interrupt enable bit) in the C0MCTRL register of the corresponding message buffer has to be set to 1 for that message buffer to participate in the interrupt generation process.

19.13 Diagnosis Functions and Special Operational Modes

The CAN module provides the receive-only mode, single-shot mode, and self-test mode to support CAN bus diagnosis or the operation of specific CAN communication methods.

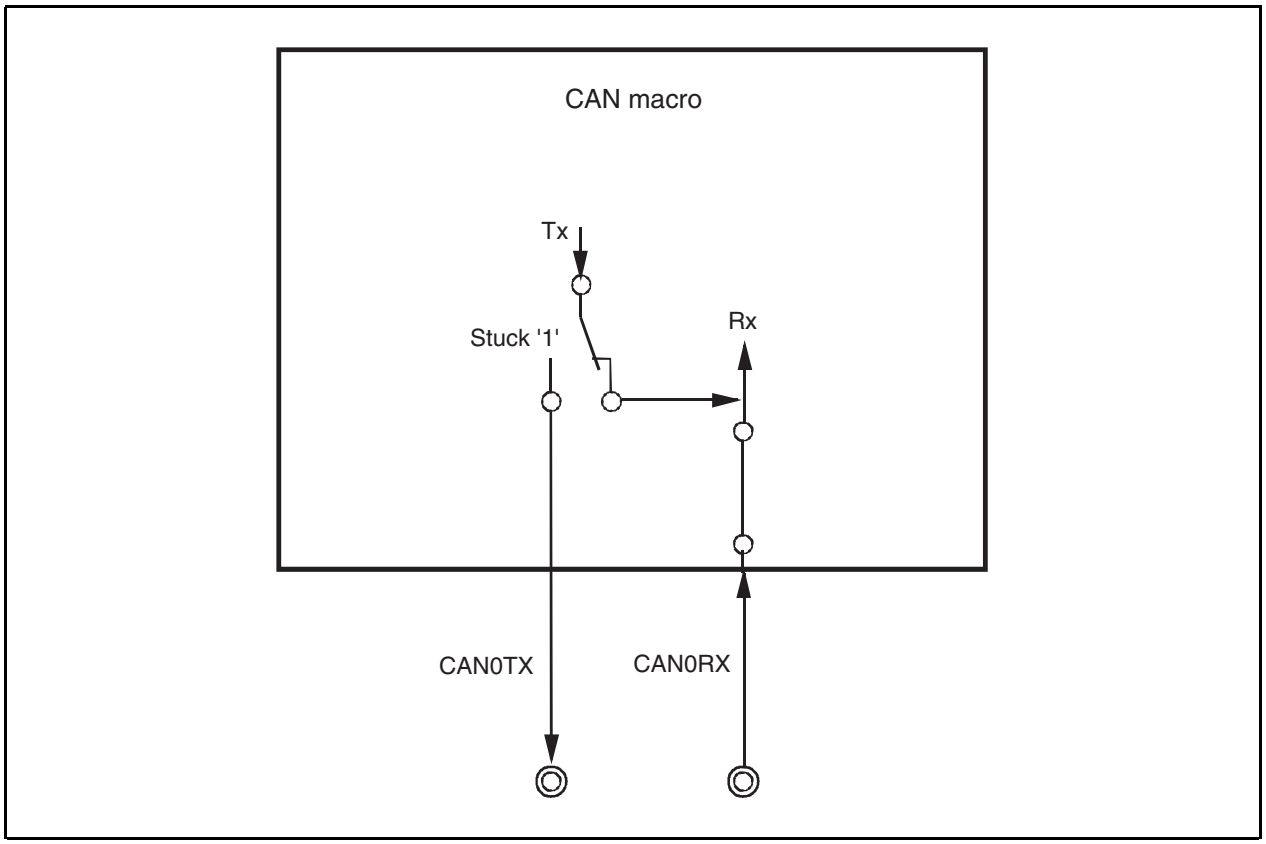
19.13.1 Receive-only mode

The “receive-only mode” can be used for CAN bus analysis nodes, which have to receive all messages without causing any interference on the CAN bus.

For example, receive-only mode is used for automatic baud-rate detection. For automatic bit-rate detection, the bit-rate in the CAN module is changed until a valid reception is detected. A valid reception means that a message frame has been received in the CAN protocol transfer layer without occurrence of an error. A valid reception does not require acceptance of the message frame in a receive message buffer (data frame) or transmit message buffer (remote frame).

The event of a valid reception is indicated by setting the VALID bit in the C0CTRL register (1).

Figure 19-40. CAN Module Terminal Connection in Receive-Only Mode



In the operational mode “receive-only mode”, no message frames can be sent from the CAN module to the CAN bus. Transmit requests issued for message buffers defined as transmit message buffers are ignored.

In “receive-only mode”, the CAN0TX output terminal of the CAN module is stuck at the recessive level. Therefore, no active error flag can be sent from the CAN module when a CAN bus error is detected while receiving a message frame. Since no transmission can be issued from the CAN module, the transmit error counter TEC is never updated. A CAN module in “receive-only mode” never enters the CAN protocol error state bus-off.

A CAN module in the “receive-only mode” is also not able to generate a dominant bit on the CAN bus in the ACK slot of the ACK field upon a valid reception of a message frame. Furthermore, in “receive-only mode”, no overload frames can be generated.

Caution If only 2 CAN nodes are connected to the CAN bus and one of the CAN nodes is operating in “receive-only mode”, there is no acknowledgment on the CAN bus. Due to the missing acknowledgment, the transmitter will send an active error flag and repeat sending the message frame. The transmitter becomes »error passive« after sending the message frame 16 times (assuming the error counters were zero in the beginning and no other errors have occurred). When the message frame is sent for the 17th time, the transmitter generates a passive error flag, so the transmitter in “receive-only mode” gets the first “valid” message frame and the VALID bit is set (1) for the first time.

19.13.2 Single-shot mode

In the operational mode “single-shot mode” automatic re-transmission as defined in the CAN protocol is switched off. (According to the CAN protocol, a message frame transmission that has been aborted by either ‘arbitration loss’ or ‘error occurrence’ has to be repeated without any interaction by software.)

The “single-shot mode” disables the re-transmission of an aborted message frame transmission according the setting of the AL bit in C0CTRL register. When the AL bit is cleared (0) re-transmission upon ‘arbitration loss’ and upon ‘error occurrence’ is disabled. If the AL bit is set (1), re-transmission upon ‘error occurrence’ is disabled, but re-transmission upon ‘arbitration loss’ is enabled. As a consequence, the TRQ flag in a message buffer defined as a transmit message buffer may be cleared (0) by the following events.

- Successful transmission of the message frame
- ‘Arbitration loss’ while sending the arbitration field of the message frame
- ‘Error occurrence’ while sending the message frame

The events ‘arbitration loss’ and ‘error occurrence’ have to be distinguished by checking the LEC2 to LEC0 bits in the C0INTS register.

Upon a successful transmission, the ‘transmit successful’ interrupt status bit CINTS0 in the C0INTS register is set (1). In addition, an interrupt request signal will be generated, when it has been enabled by setting the CIE0 enable bit in the C0IE register (1).

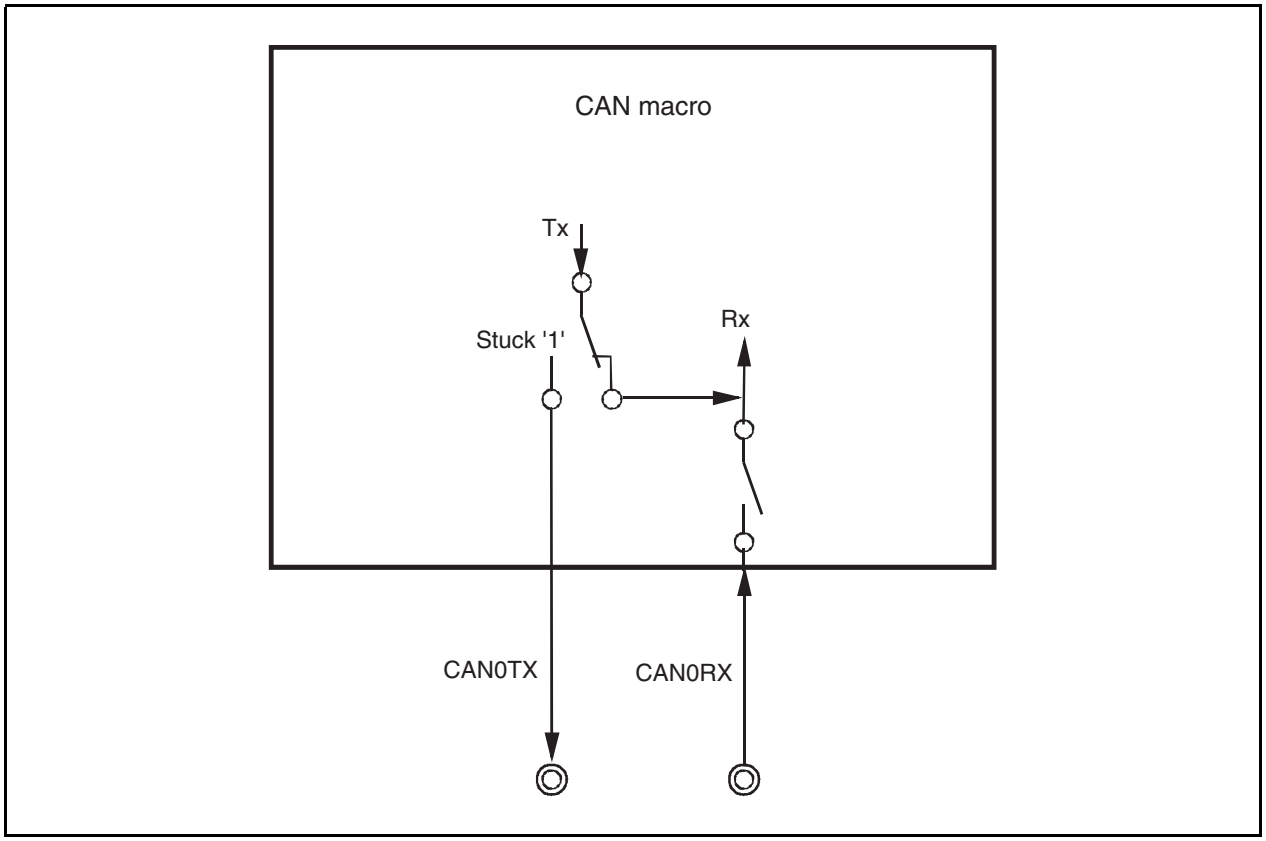
The “single-shot mode” can be used when emulating time-triggered communication methods (ex. TTCAN level 1).

19.13.3 Self-test mode

In the “self-test mode”, message frame transmission and message frame reception can be tested without connecting the CAN node to the CAN bus or without affecting the CAN bus.

In the operational mode “self-test mode”, the CAN module of a CAN I/F channel is completely disconnected from the CAN bus, but internal switches connect the transmit path with the receive path. The CAN0TX pin of the CAN I/F channel is stuck to a ‘recessive’ CAN bus level.

Figure 19-41. CAN Module Terminal Connection in Self-Test Mode



19.14 Time Stamp Function

CAN is an asynchronous, serial protocol. All nodes connected to the bus have a local, autonomous clock. As a consequence, the clocks of the nodes have no relation (i.e., the clocks are asynchronous and may even have different frequencies). But in some applications, a common time base over the network (global time base) is needed. In order to build up a global time base, a time stamp (TS) function is used. The essential mechanism of a TS function is the capture of timer values triggered by signals on the CAN bus.

19.14.1 Basic time stamp function

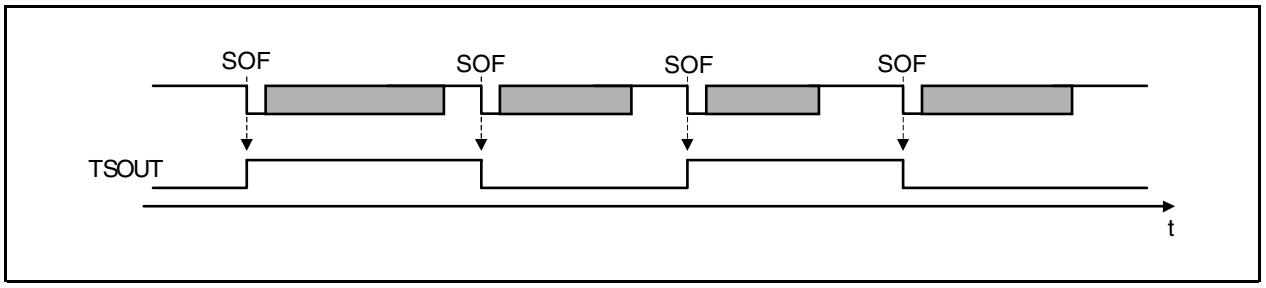
The CAN module supports the capturing of successfully received data frames upon occurrence of an event (basic time stamp function). For the basic TS, a 16-bit timer with a capture and compare register in a microcontroller system is used. In this case, the 16-bit capture timer unit captures the timer value according to a trigger signal (TSOUT) provided by the CAN module. The CPU can retrieve the point in time when the event happened by reading the captured value. In other words, the CPU can get the time stamp of a message received from the CAN bus.

TSOUT is generated by two source events selected by the TSSEL bit in the C0TS register.

- SOF event (start of frame): TSSEL = 0
- EOF event (last bit of end of frame): TSSEL = 1

The TSOUT signal has to be enabled by setting the TSEN bit in the C0TS register.

Figure 19-42. Timing Diagram of Capture Output Signal TSOUT



TSOUT toggles its level upon occurrence of the selected source event (in the timing diagram in Figure 19-42, the SOF event is selected) during data frame reception. As a consequence, the function “capture on falling or rising edge” has to be selected in the 16-bit timer in the microcontroller system.

The time stamp function is controlled by the TSLOCK bit in the C0TS register. When TSLOCK is cleared (0) upon the selected source event, a TSOUT pulse is generated. If TSLOCK is set (1) upon the selected source event, a TSOUT pulse is generated, but as soon as a data frame has been successfully received in message buffer 0, the TSEN signal is automatically cleared (0) to prevent further time stamp pulses. Thus, the toggle of TSOUT is suppressed, and the time stamp value references a data frame in message buffer 0. The CPU does not need to react quickly as the value will not be overwritten until TSEN is set (1) again. The basic TS function cannot be used while the CAN module operates in “normal operating mode with automatic block transmission”.

19.15 Rules for Setting Baud Rate

Always make sure that the settings are within the range of limit values for ensuring correct operation of the CAN controller as follows.

(a) 5 TQ ≤ SPT (sampling point) ≤ 17 TQ
SPT = TSEG1 + 1
(b) 8 TQ ≤ DBT (data bit time) ≤ 25 TQ
DBT = TSEG1 + TSEG2 + 1TQ = TSEG2 + SPT
(c) 1TQ ≤ SJW (synchronization jump width) ≤ 4TQ
SJW ≤ DBT−SPT
(d) 4 ≤ TSEG1 ≤ 16 [3 ≤ Setting Value of TSEG1[3:0] ≤ 15]
(e) 1 ≤ TSEG2 ≤ 8 [0 ≤ Setting Value of TSEG2[2:0]] ≤ 7]

Remark TQ = 1/f_{TQ} (f_{TQ}: CAN protocol layer basic system clock)
TSEG1[3:0] (bits 3 to 0 of the CAN0 bit rate register (C0BTR))
TSEG2[2:0] (bits 10 to 8 of the CAN0 bit rate register (C0BTR))

(1) Example of CAN baud rate setting

The following is an example of how correct settings for the C0BRP register and C0BTR register can be calculated.

Conditions from CAN bus:

<1> CAN module system clock (f_{CANMOD}): 8 MHz
<2> CAN bus baud rate: 500 kbps
<3> Sample point: 80% or more
<4> Synchronization jump width: 3 TQ

First, calculate the ratio between the CAN module system clock frequency and the CAN bus baud rate frequency as shown below.

f_{CANMOD}/CAN bus band rate = 8 MHz/500 kHz = 16

Set a number between 1 and 256 to the C0BRP register's TQPRS7 to TQPRS0 bits as the setting for the prescaler (CAN protocol layer basic system clock: f_{TQ}), then set a value between 8 and 25 to the C0SYNC register's DTBR4 to DBTR0 bits as the data bit time.
Since it is assumed that the SJW (synchronization jump width) value is 3, the maximum setting for SPT (sample point) is the "data bit time setting minus 3" or less and 17 or less.

(SPT ≤ DBT – 3 and SPT ≤ 17)

Given the above limit values, the following two settings are possible.

Prescaler	DBT	SPT(MAX.)	TSEG1	TSEG2	Calculated SPT
1	16	13	12	3	13/16 = 81%
2	8	5	4	3	5/8 = 62.5%

8 MHz/500 kHz = 16

$= 1 \times 16$	<1>
$= 2 \times 8$	<2>
$= 4 \times 4$	<3>
$= 8 \times 2$	<4>
$= 16 \times 1$	<5>

The settings that can actually be made for the device are in the range from <1> to <2> above (the section enclosed in broken lines).
Among these options in the range from <1> to <2> above, option <1> is the ideal setting when actually setting the register.

(i) Prescaler (CAN protocol layer basic system clock: f_{TQ}) setting

f_{TQ} is calculated as shown below.

- $f_{TQ} = f_{CANMOD}/(a + 1) : [0 \leq a \leq 255]$

Value a is set using bits 7 to 0 (TQPRS[7:0]) of the C0BRP register.

f_{TQ} = 8 MHz/1
= 8 MHz/(0 + 1)
thus a = 5

Therefore, C0BRP register = 0005H

(ii) TSEG1 setting

TSEG1 is calculated as shown below.

• TSEG1 = (a + 1)TQ : [1 ≤ a ≤ 15]

Value a is set using bits 3 to 0 (TSEG1[3:0]) of the C0BTR register.

TSEG1 = 12TQ
= (a + 1)TQ
thus a = 11

Therefore, the C0BTR register's bits 3 to 0 = 1011B

(iii) TSEG2 setting

TSEG2 is calculated as shown below.

• TSEG2 = (a + 1)TQ : [0 ≤ a ≤ 7]

Value a is set using bits 10 to 8 (TSEG2[2:0]) of the C0BTR register.

TSEG2 = 3TQ
= (a + 1)TQ
thus a = 11

Therefore, the C0BTR register's bits 10 to 8 = 010B

(iv) SJW (synchronization jump width) setting

SJW is calculated as shown below.

• SJW = (a + 1)TQ : [0 ≤ a ≤ 3]

Value a is set using bits13 and 12 (SJW1, SJW0) of the C0BTR register.

SJW= 3TQ
= (a + 1)TQ
thus a = 2

Therefore, the C0BTR register's bits 13 and 12 = 10B

The C0BTR register settings based on these results are shown in Figure 19-43 below.

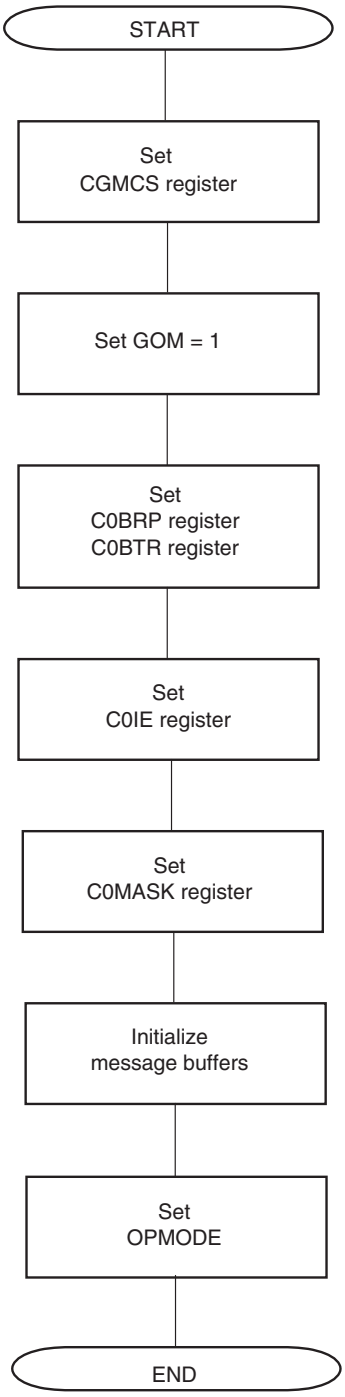
Figure 19-43. C0BTR Register Settings

	15	14	13	12	11	10	9	8
C0BTR	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10
Setting	0	0	0	0	1	0	1	1

	7	6	5	4	3	2	1	0
	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
Setting	1	0	1	0	0	0	1	0

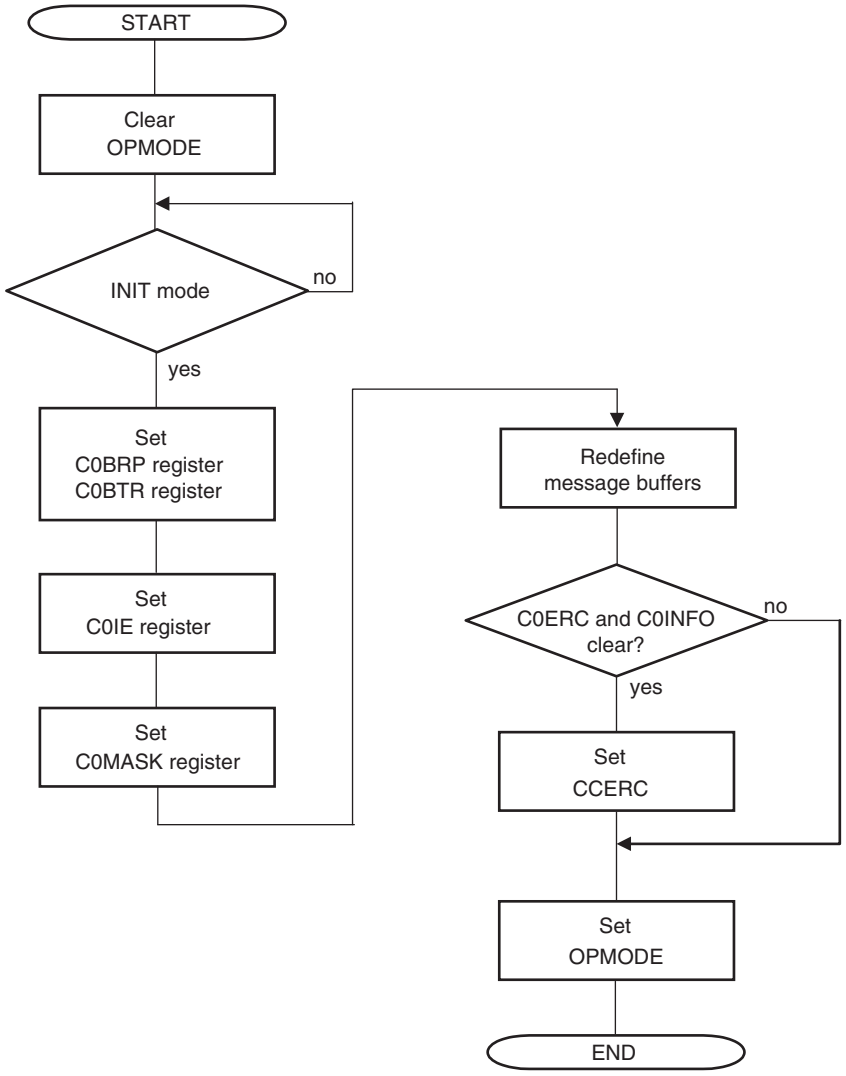
19.16 Operation of CAN Controller

Figure 19-44. Initialization



Remark OPMODE: Normal operating mode, normal operating mode with ABT, receive-only mode, single-shot mode, self-test mode.

Figure 19-45. Reinitialization



Caution Before a message buffer is initialized, the RDY bit must be cleared.

Remark OPMODE: Normal operating mode, normal operating mode with ABT, receive-only mode, single-shot mode, self-test mode.

Figure 19-46. Message Buffer Initialization

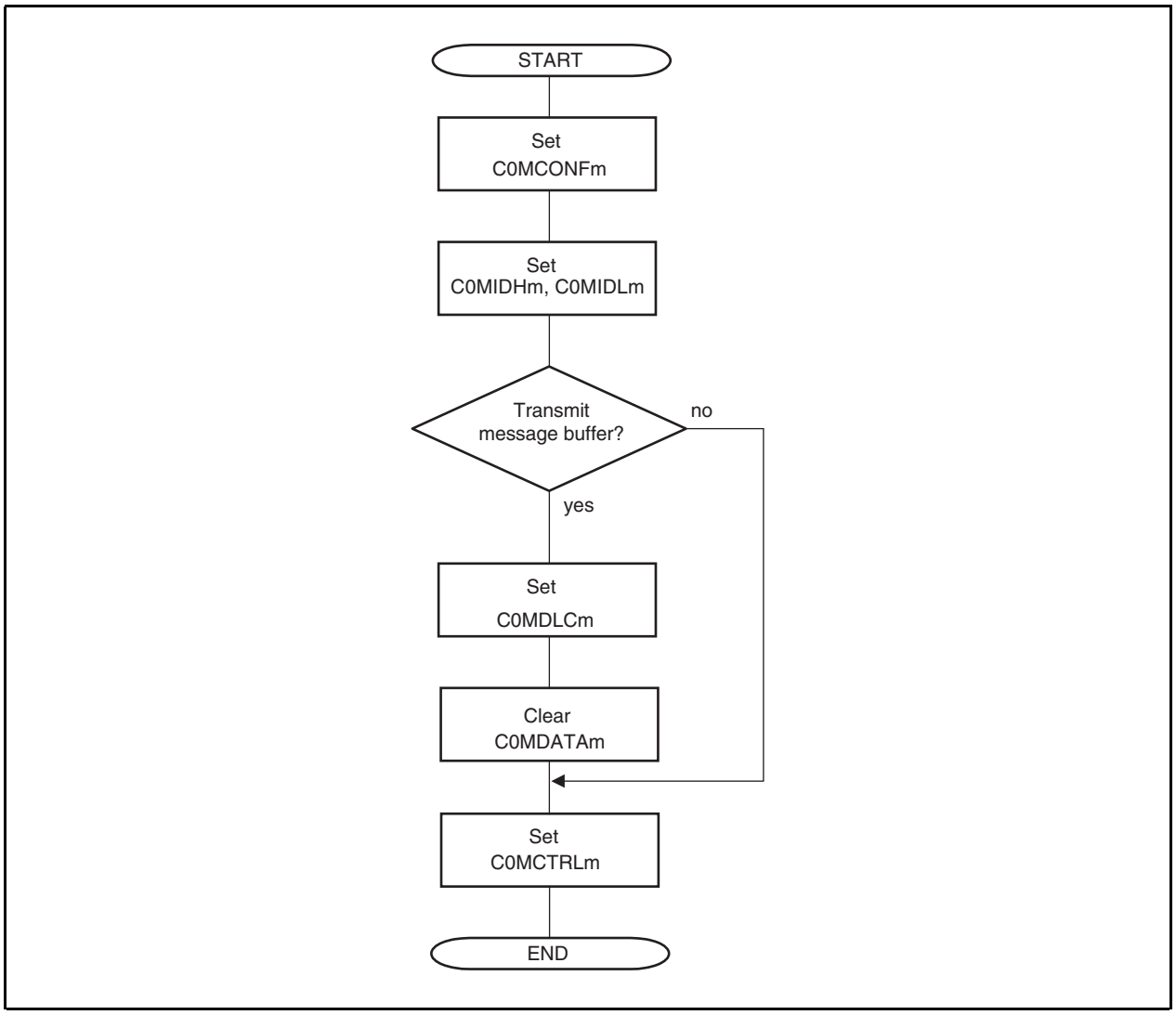


Figure 19-47. Message Buffer Redefinition

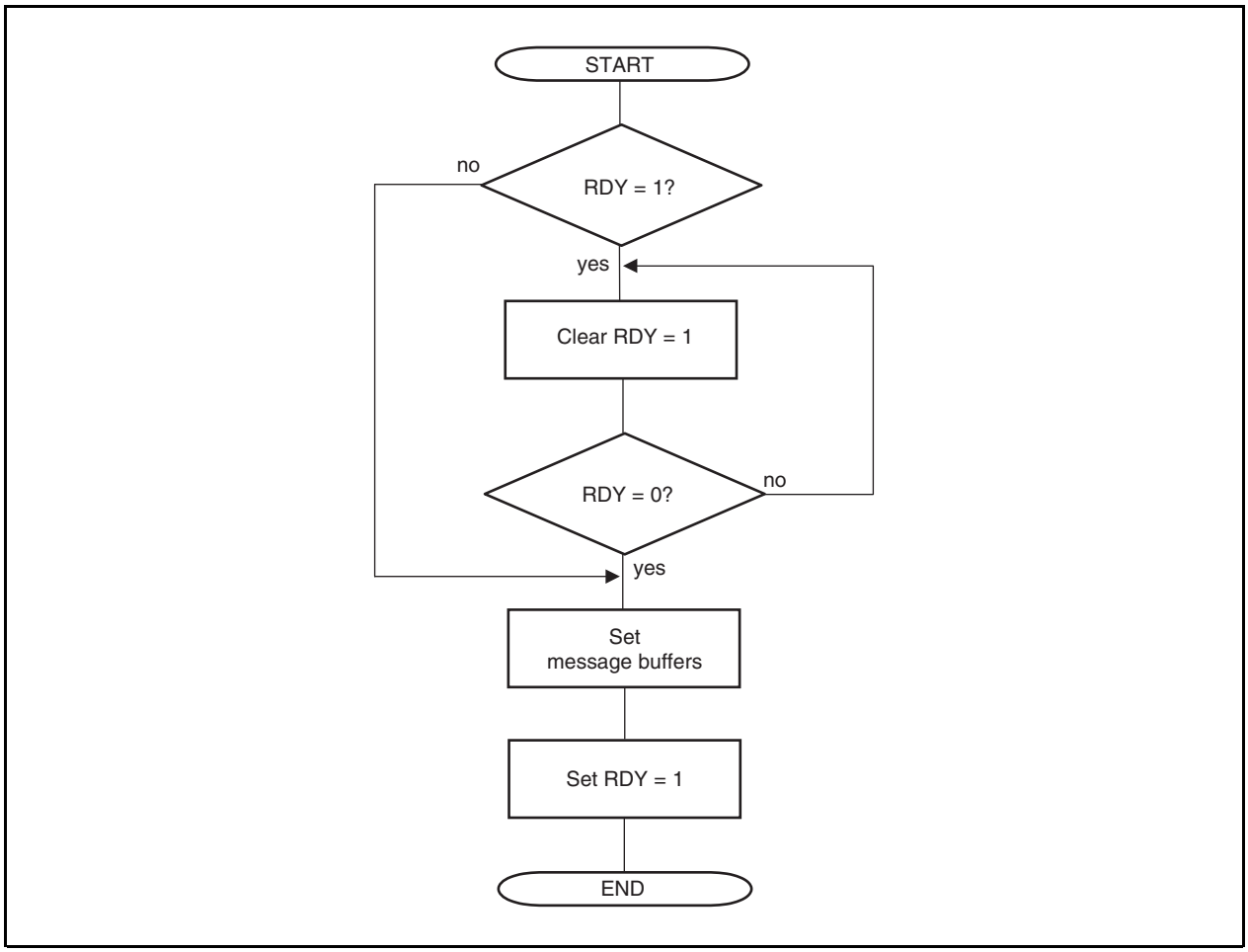
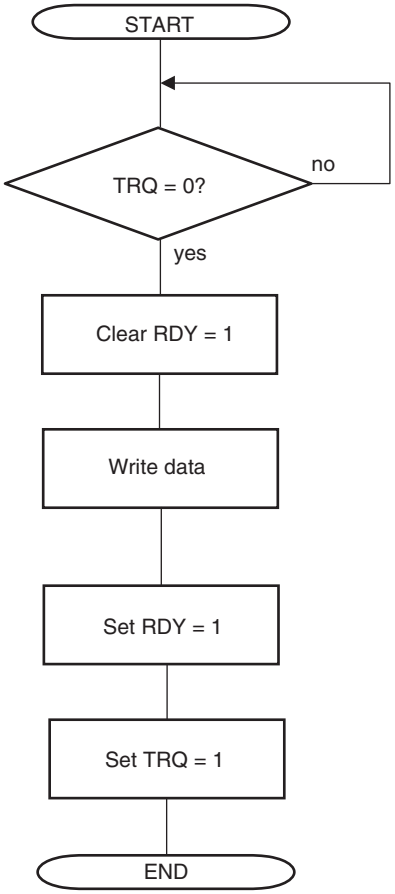


Figure 19-48. Transmit Preparation (Normal Mode)



Caution The TRQ bit should be set after the RDY bit is set.
The TRQ bit and RDY bit should not be set at the same time.

Figure 19-49. Transmit Preparation (ABT Mode)

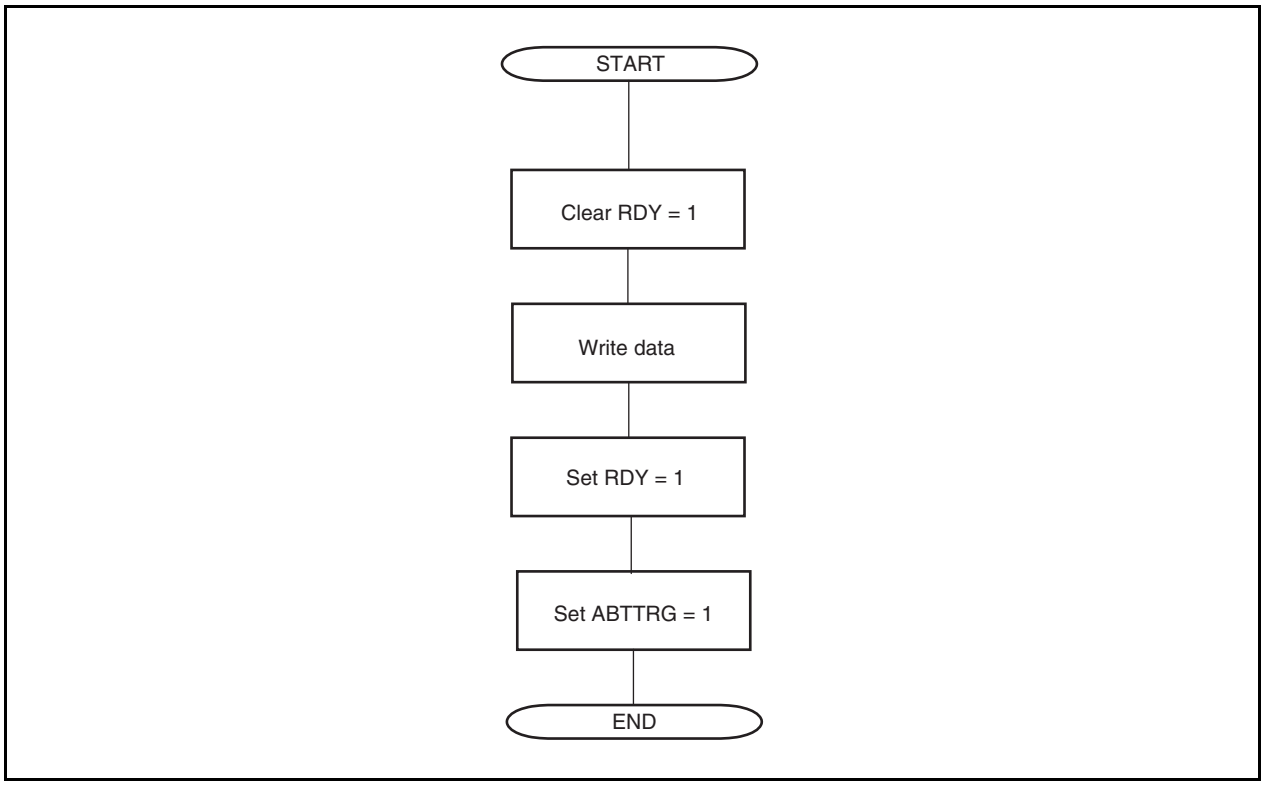
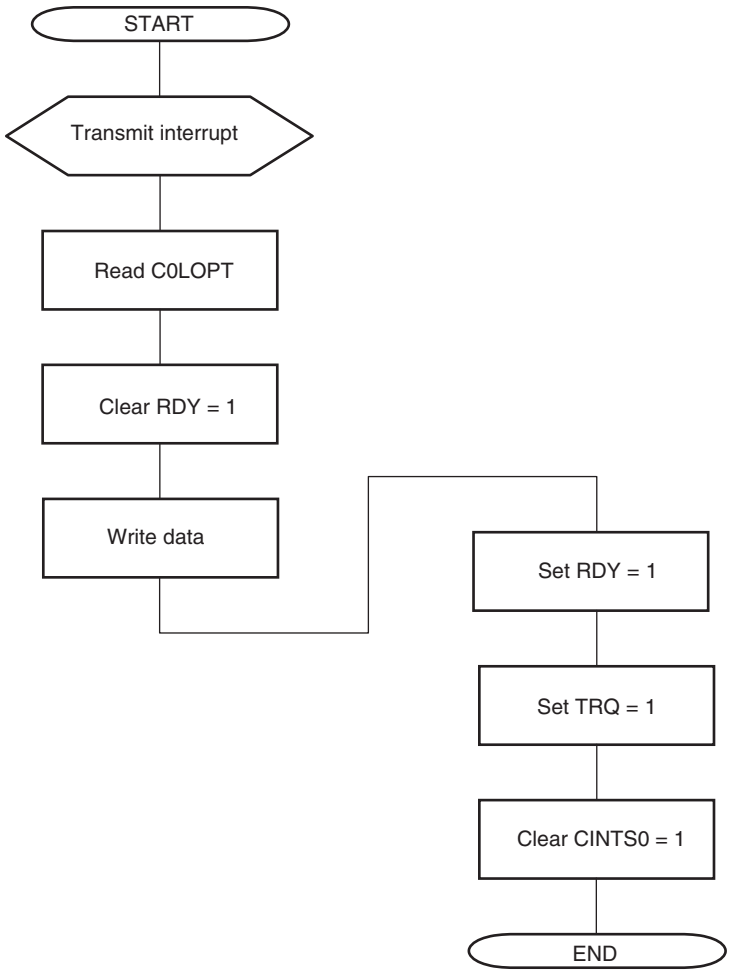
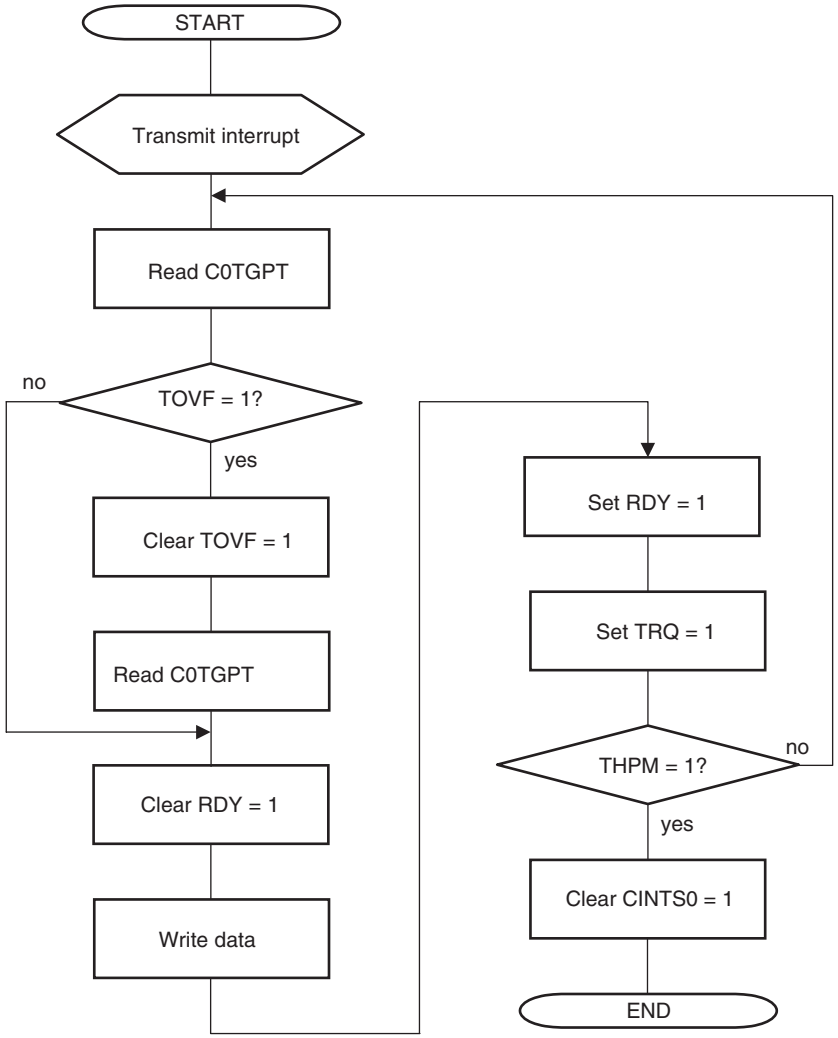


Figure 19-50. Transmission via Interrupt (C0LOPT)



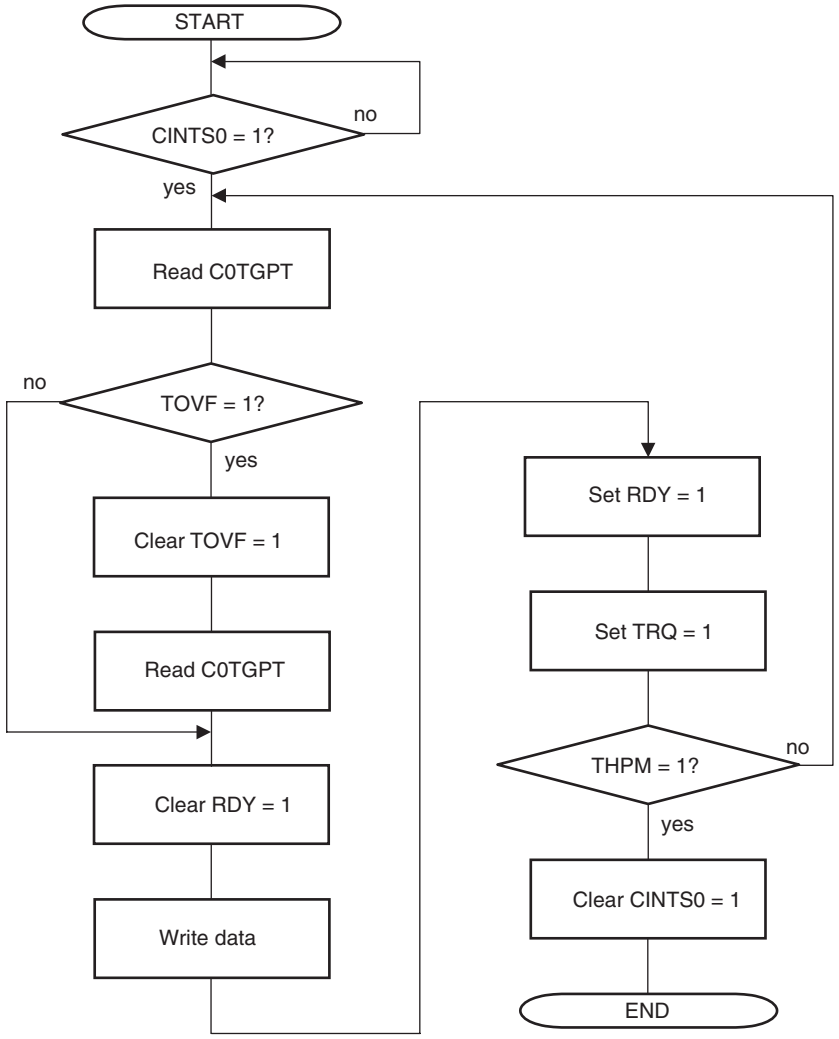
Caution The TRQ bit should be set after the RDY bit is set.
The TRQ bit and RDY bit should not be set at the same time.

Figure 19-51. Transmission via Interrupt (C0TGPT)



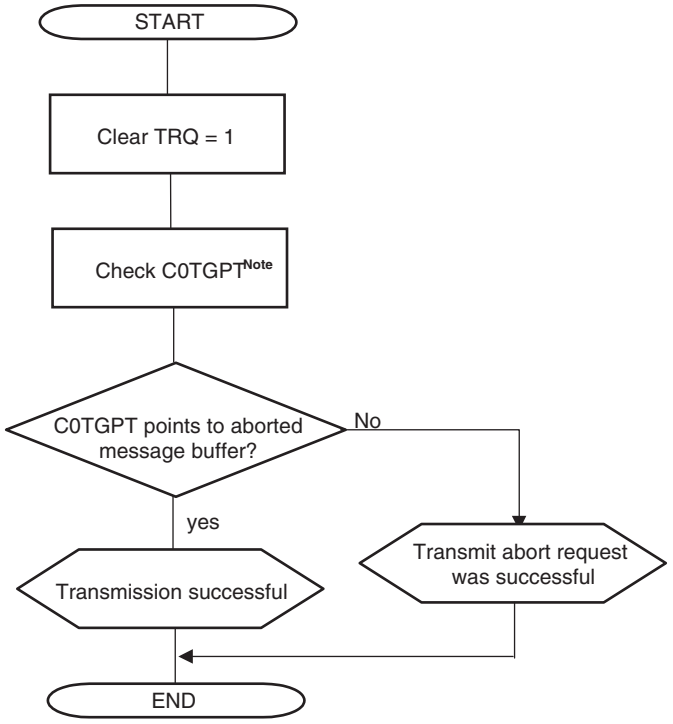
Caution The TRQ bit should be set after the RDY bit is set.
The TRQ bit and RDY bit should not be set at the same time.

Figure 19-52. Transmit Software Polling



Caution The TRQ bit should be set after the RDY bit is set.
The TRQ bit and RDY bit should not be set at the same time.

Figure 19-53. Transmission Request Abort Process (Normal Mode)



Note The user must decide when to check.

Figure 19-54. Transmission Request Abort Process (ABT Mode)

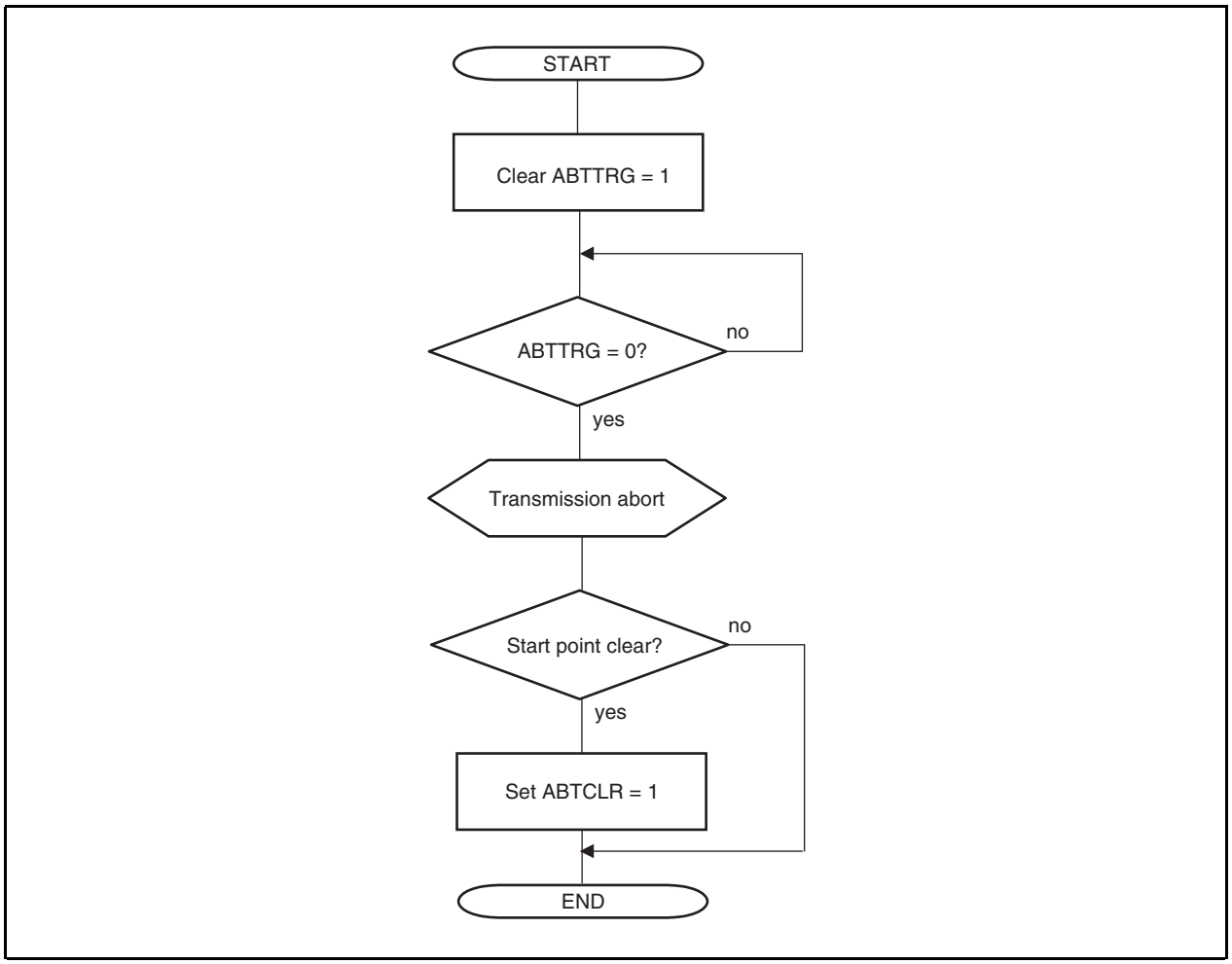
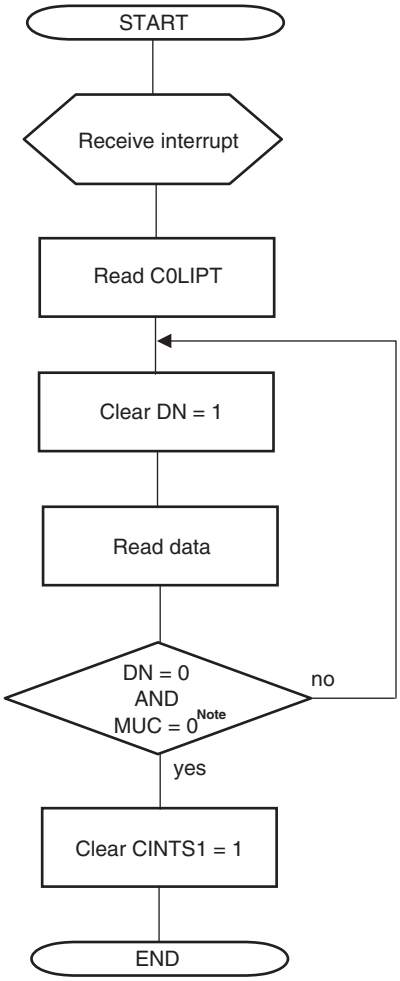
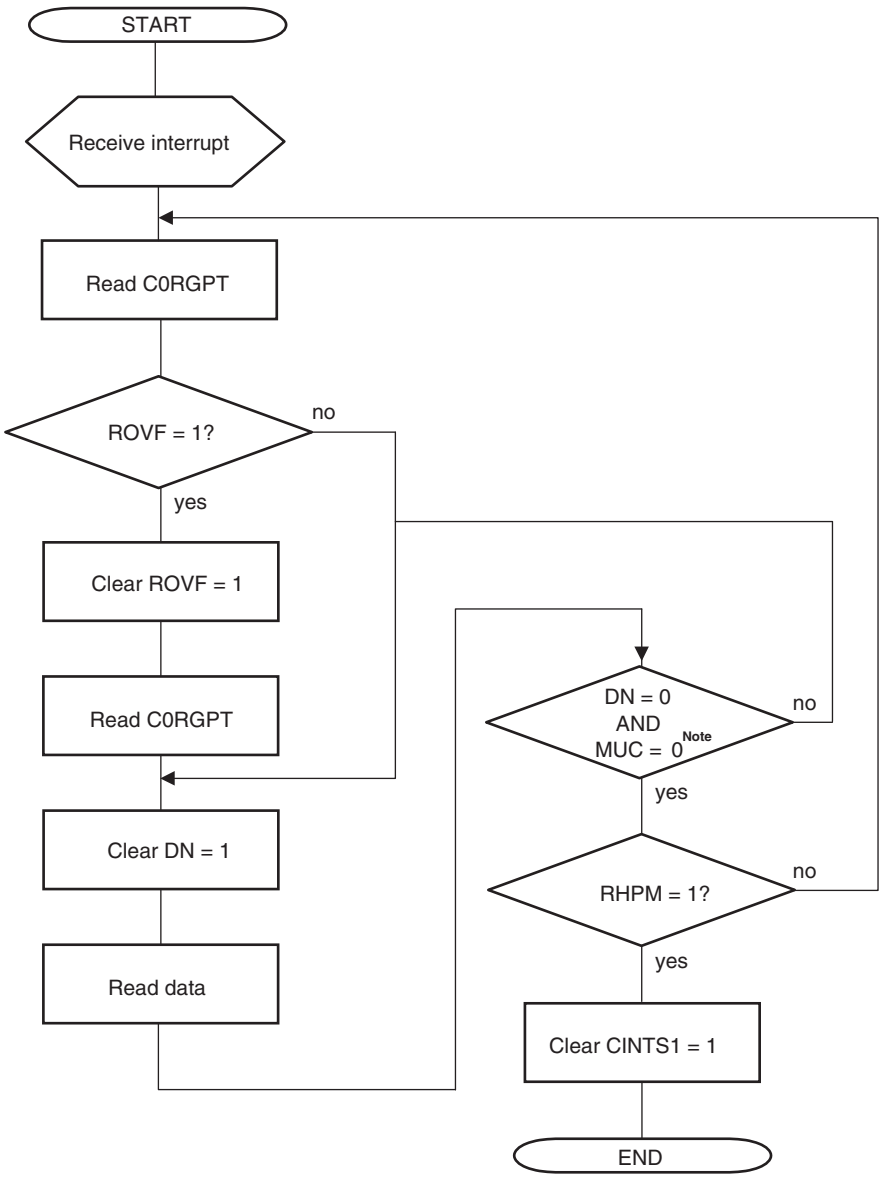


Figure 19-55. Reception via Interrupt (C0LIPT)



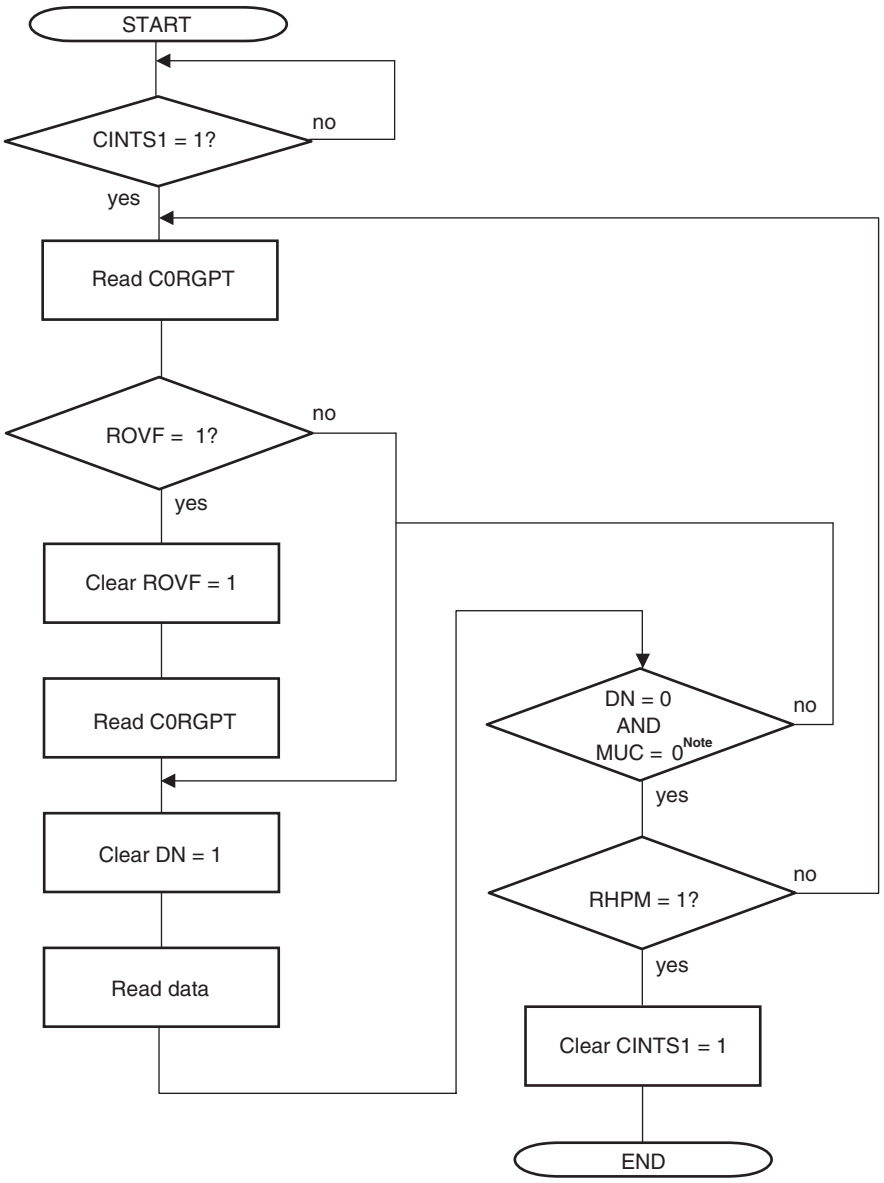
Note Check the MUC bit and DN bit using one read access.

Figure 19-56. Reception via Interrupt (C0RGPT)



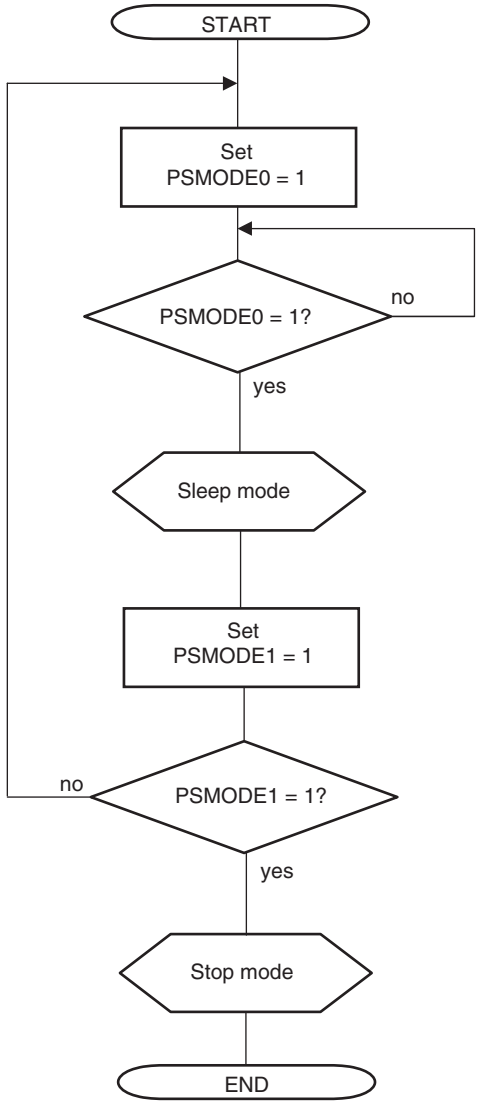
Note Check the MUC bit and DN bit using one read access.

Figure 19-57. Receive Software Polling



Note Check the MUC bit and DN bit using one read access.

Figure 19-58. Setting CAN Sleep/Stop Mode



Caution When accessing a message buffer after requesting shift to sleep mode, confirm that the MBON bit is set. If MBON is cleared, the read value is undefined.

Figure 19-59. Clear CAN Sleep/Stop Mode

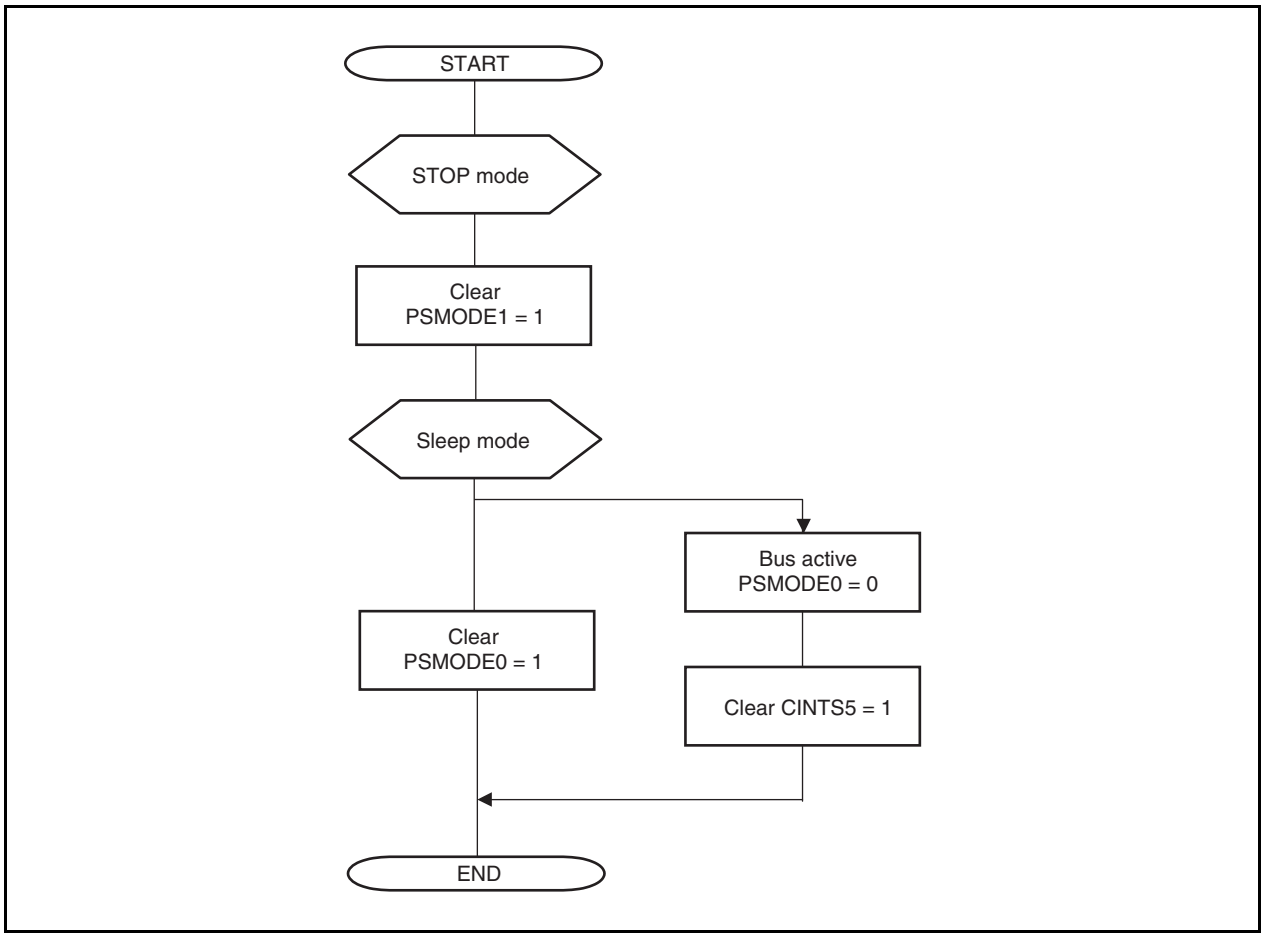


Figure 19-60. Bus-Off Recovery

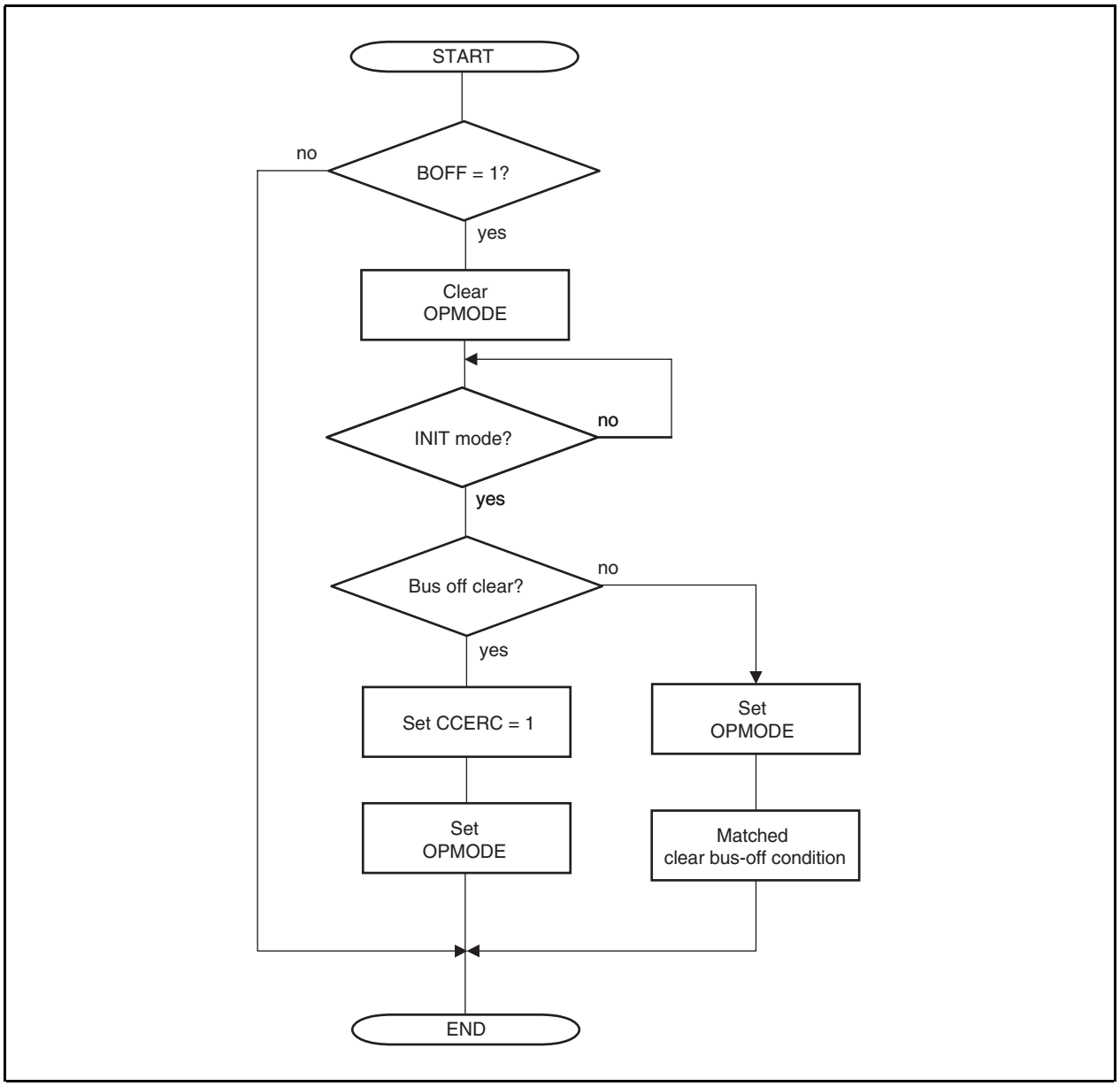
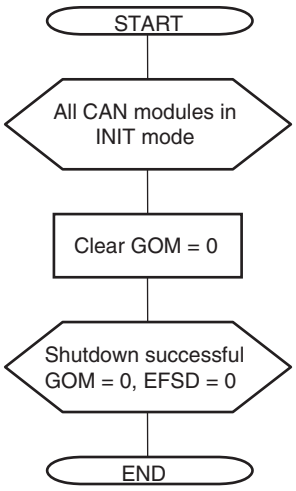
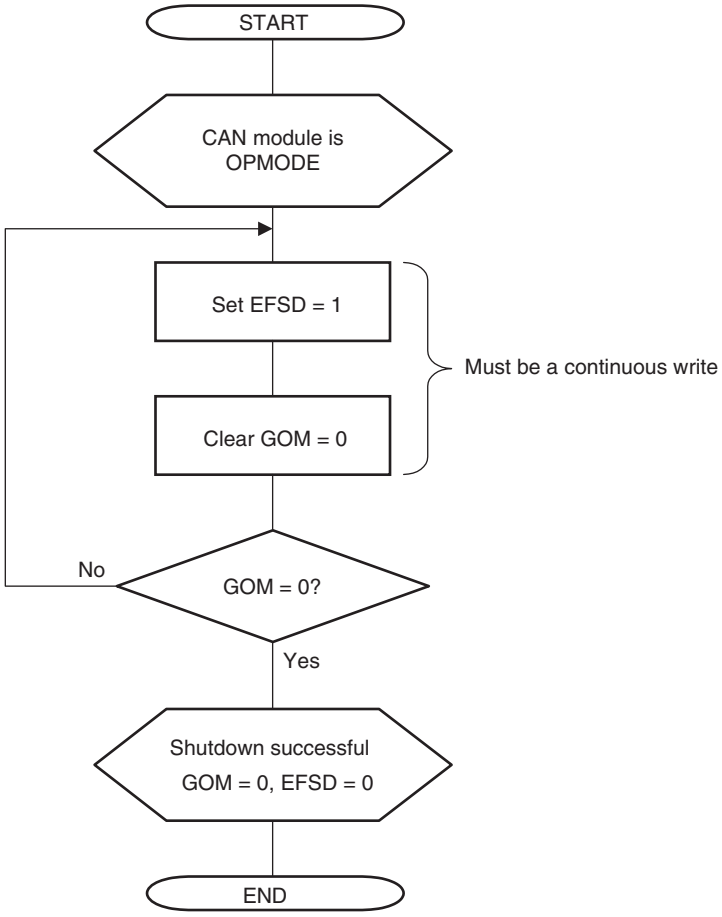


Figure 19-61. Shutdown Process (Normal Shutdown)



Remark If the GOM bit is set after shutdown, the message buffer must be reinitialized before it is shifted to any operation mode.

Figure 19-62. Shutdown Process (Forcible Shutdown)



Cautions 1. Do not read- or write-access any register between setting the EFSD bit and clearing the GOM bit.

2. If the GOM bit is not cleared after clearing the GOM bit in the shutdown sequence, set the EFSD bit again. Otherwise, interrupts will be set to disabled between when the EFSD bit is set and the GOM bit is cleared.

Remark OPMODE: Normal operating mode, normal operating mode with ABT, receive-only mode, single-shot mode, self-test mode.

Figure 19-63. Error Handling

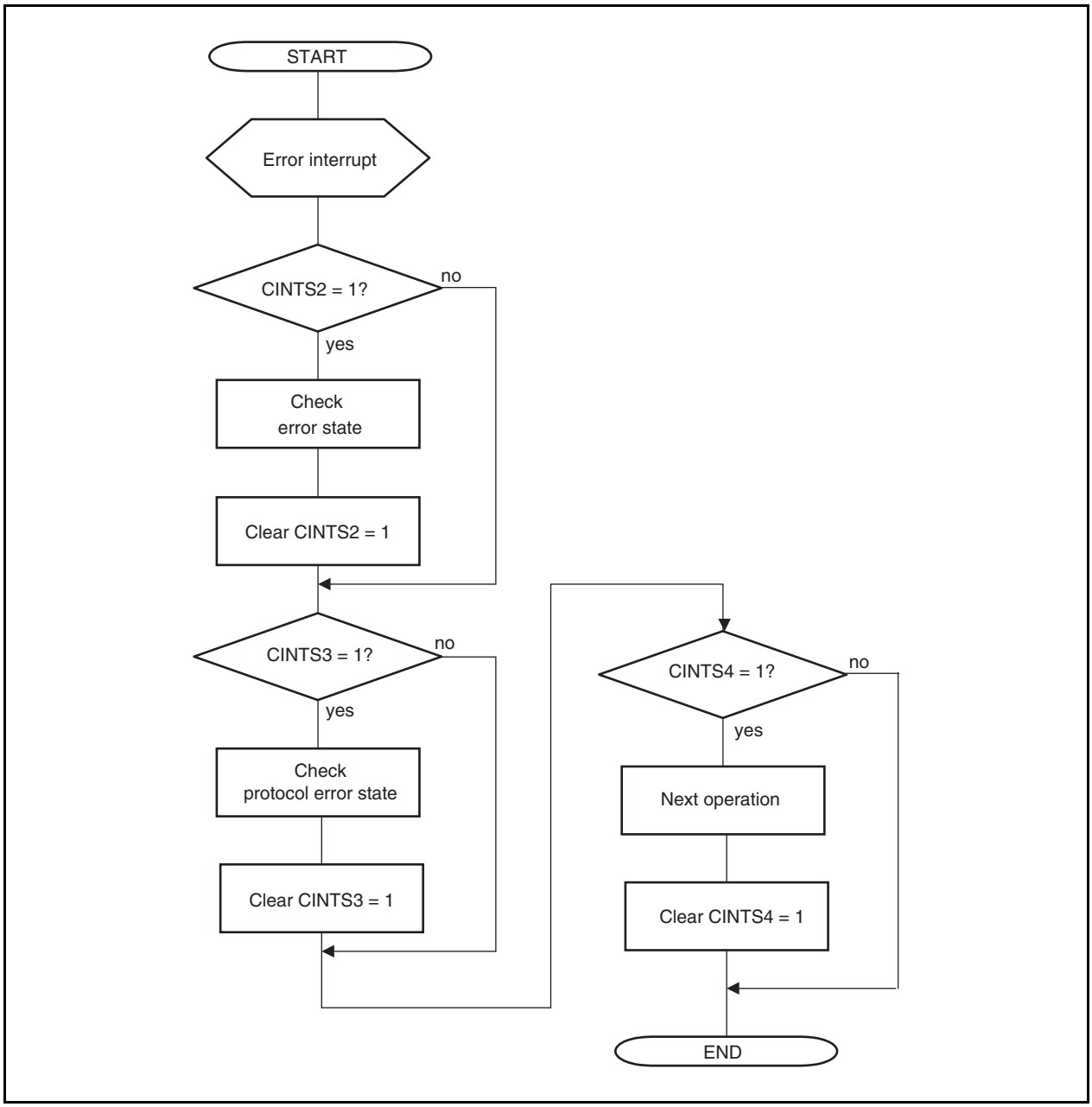


Figure 19-64. Setting CPU Standby (from CAN Sleep Mode)

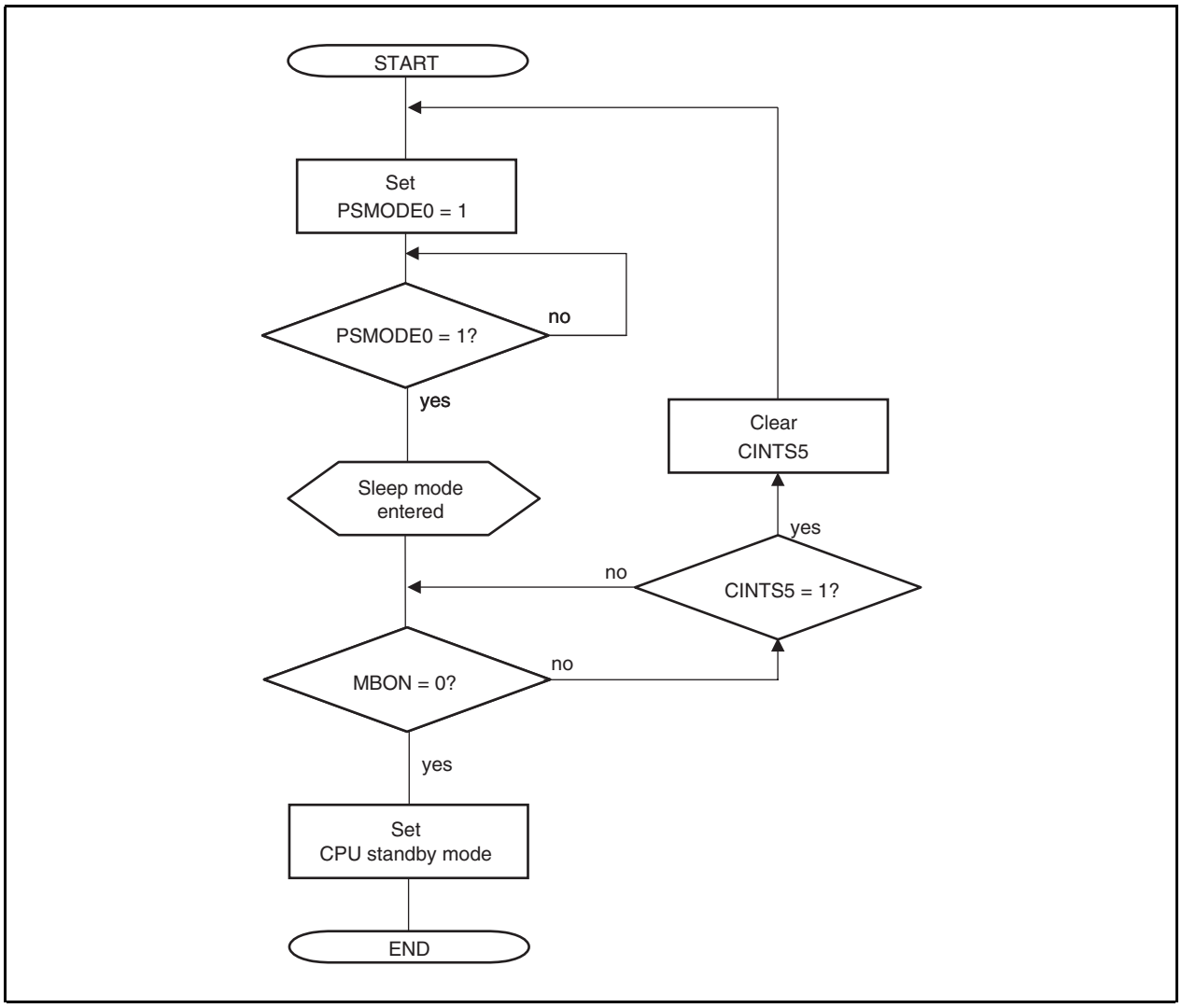
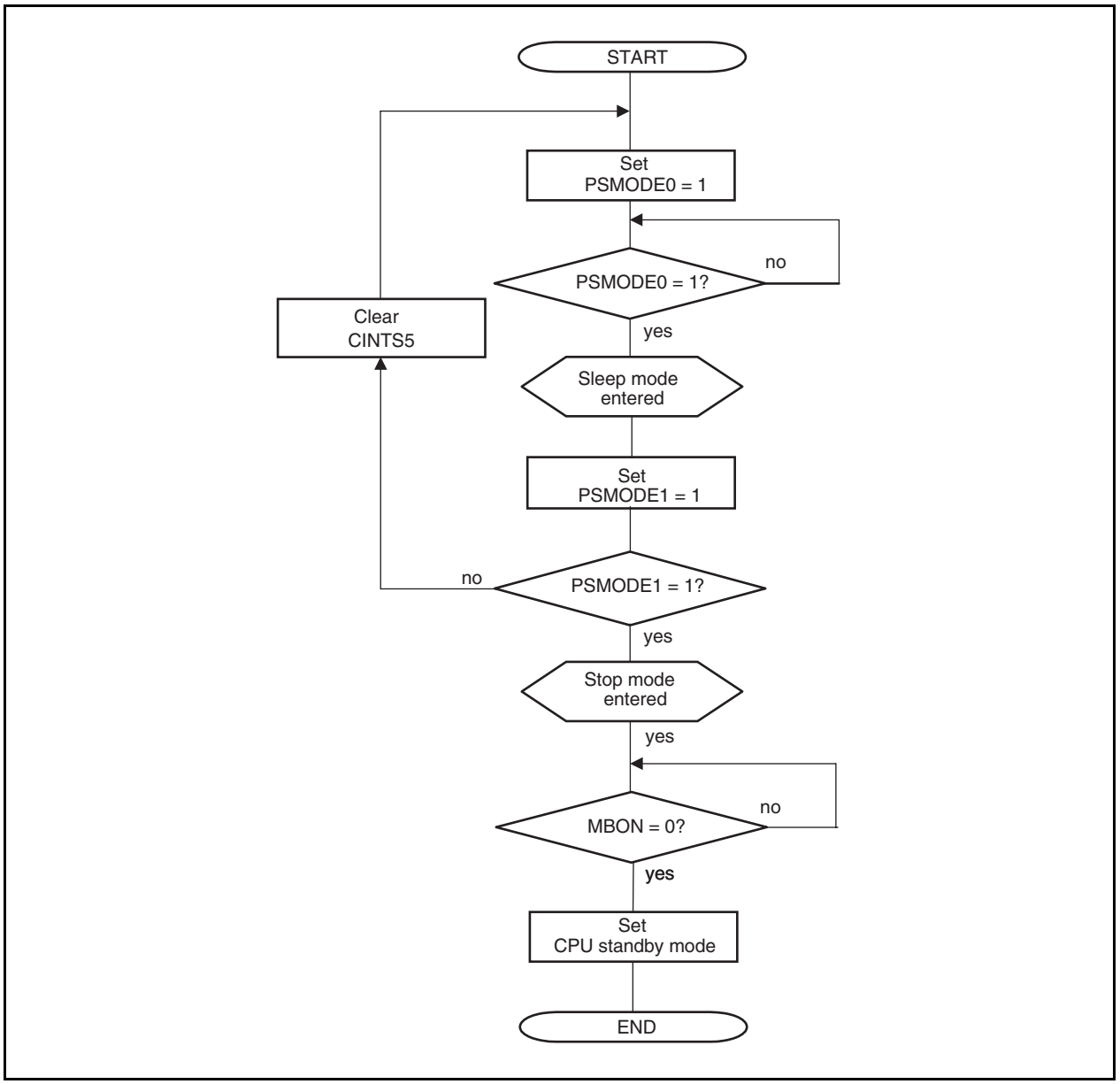


Figure 19-65. Setting CPU Standby (from CAN Stop Mode)



CHAPTER 20 DMA FUNCTIONS (DMA CONTROLLER)

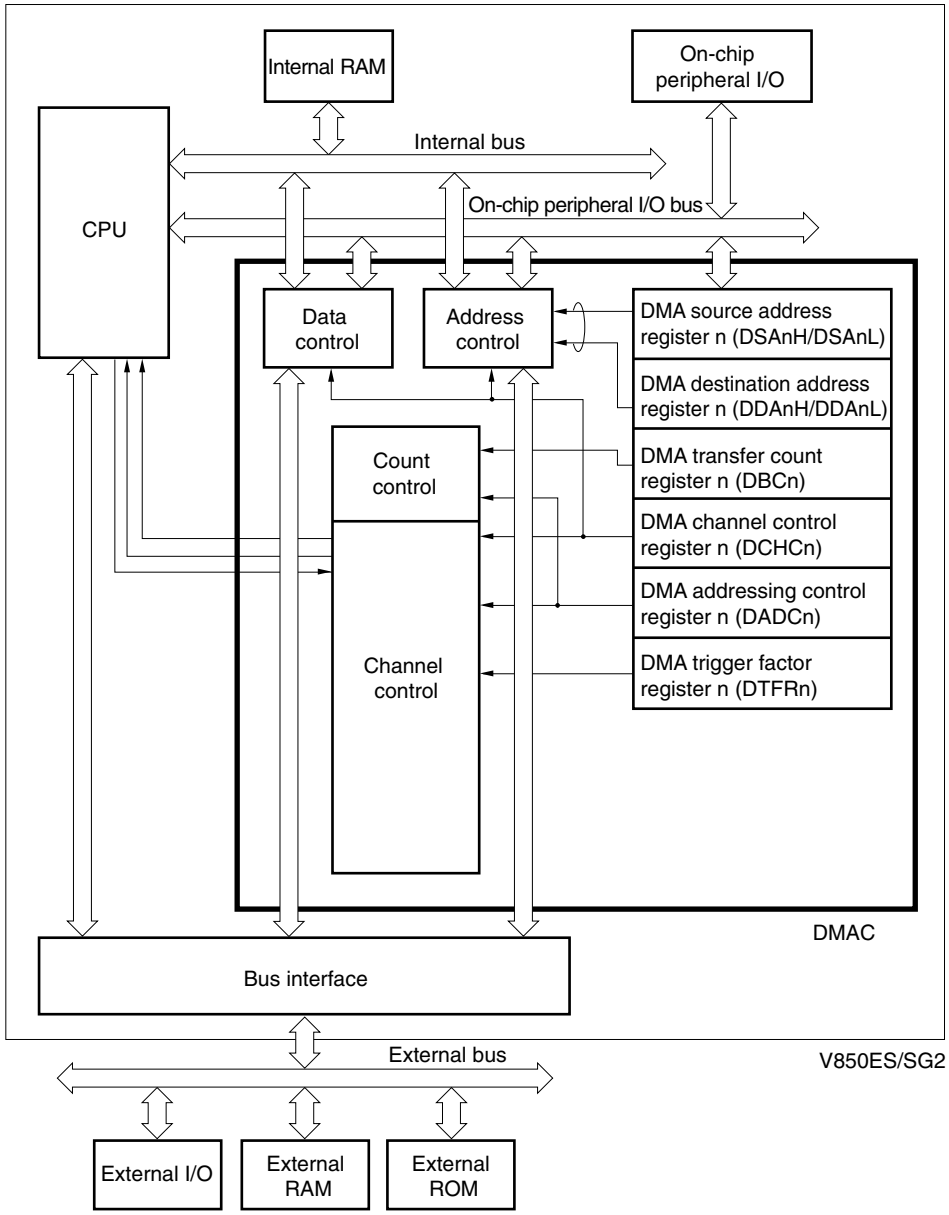
The V850ES/SG2 include a direct memory access (DMA) controller (DMAC) that executes and controls DMA transfer.

The DMAC controls data transfer between memory and I/O, between memories, or between I/Os based on DMA requests issued by the on-chip peripheral I/O (serial interface, real-time pulse unit, and A/D converter), interrupts from external input pins, or software triggers (memory refers to internal RAM or external memory).

20.1 Features

- 4 independent DMA channels
- Transfer unit: 8/16 bits
- Maximum transfer count: 65,536 (2^{16})
- Transfer type: Two-cycle transfer
- Transfer mode: Single transfer mode
- Transfer requests
  - Request by interrupts from on-chip peripheral I/O (serial interface, timer/counter, A/D converter) or interrupts from external input pin
  - Requests by software trigger
- Transfer objects
  - Internal RAM ↔ peripheral I/O
  - Peripheral I/O ↔ peripheral I/O
  - Internal RAM ↔ external memory
  - External memory ↔ peripheral I/O
  - External memory ↔ external memory

20.2 Configuration



Remark n = 0 to 3

20.3 Control Registers

20.3.1 DMA source address registers 0 to 3 (DSA0 to DSA3)

The DSA0 to DSA3 registers set the DMA source addresses (28 bits each) for DMA channel n (n = 0 to 3). These registers are divided into two 16-bit registers, DSAnH and DSAnL.

(1) DMA source address registers 0H to 3H (DSA0H to DSA3H)

The DSA0H to DSA3H registers can be read or written in 16-bit units.

After reset: Undefined R/W Address: DSA0H FFFFF082H, DSA1H FFFFF08AH, DSA2H FFFFF092H, DSA3H FFFFF09AH

DSAnH

15	14	13	12	11	10	9	8
IR	0	0	0	0	0	SA25	SA24

7	6	5	4	3	2	1	0
SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16

(n = 0 to 3)

IR	Specification of DMA source address
0	External memory, on-chip peripheral I/O
1	Internal RAM

SA25 to SA16	Sets the DMA source addresses (A25 to A16). During DMA transfer, it stores the next DMA transfer source address.

(2) DMA source address registers 0L to 3L (DSA0L to DSA3L)

The DSA0L to DSA3L registers can be read or written in 16-bit units.

After reset: Undefined R/W Address: DSA0L FFFFF080H, DSA1L FFFFF088H, DSA2L FFFFF090H, DSA3L FFFFF098H

DSAnL

15	14	13	12	11	10	9	8
SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8

7	6	5	4	3	2	1	0
SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0

(n = 0 to 3)

SA15 to SA0	Sets the DMA source addresses (A15 to A0). During DMA transfer, it stores the next DMA transfer source address.

Caution The DSAnH and DSAnL registers cannot be changed during transfer. Set the INIT bit of the DCHCn register to 1 to initialize these registers before changing them.

20.3.2 DMA destination address registers 0 to 3 (DDA0 to DDA3)

The DDA0 to DDA3 registers set the DMA destination address (28 bits each) for DMA channel n (n = 0 to 3). These registers are divided into two 16-bit registers, DDAnH and DDAnL.

(1) DMA destination address registers 0H to 3H (DDA0H to DDA3H)

The DDA0H to DDA3H registers can be read or written in 16-bit units.

After reset: Undefined R/W Address: DDA0H FFFFF086H, DDA1H FFFFF08EH, DDA2H FFFFF096H, DDA3H FFFFF09EH

DDAnH

15	14	13	12	11	10	9	8
IR	0	0	0	0	0	DA25	DA24

7	6	5	4	3	2	1	0
DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16

(n = 0 to 3)

IR	DMA destination address specification
0	External memory, on-chip peripheral I/O
1	Internal RAM

DA25 to DA16	Sets the DMA destination addresses (A25 to A16). During DMA transfer, it stores the next DMA transfer destination address.

(2) DMA destination address registers 0L to 3L (DDA0L to DDA3L)

The DDA0L to DDA3L registers can be read or written in 16-bit units.

After reset: Undefined R/W Address: DDA0L FFFFF084H, DDA1L FFFFF08CH, DDA2L FFFFF094H, DDA3L FFFFF09CH

DDAnL

15	14	13	12	11	10	9	8
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8

7	6	5	4	3	2	1	0
DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0

(n = 0 to 3)

DA15 to DA0	Sets the DMA destination addresses (A15 to A0). During DMA transfer, it stores the next DMA transfer destination address.

Caution The DDAnH and DDAnL registers cannot be changed during transfer. Set the INIT bit of the DCHCn register to 1 to initialize these registers before changing them.

20.3.3 DMA byte count registers 0 to 3 (DBC0 to DBC3)

The DBC0 to DBC3 registers are 16-bit registers that set the byte transfer count for DMA channels n (n = 0 to 3). These registers store the remaining transfer count during DMA transfer.

These registers can be read or written in 16-bit units.

Remark If the DBCn register is read during DMA transfer after a terminal count has occurred without the register being overwritten, the value set immediately before the DMA transfer will be read out (0000H will not be read, even if DMA transfer has ended).

After reset: Undefined R/W Address: DBC0 FFFFF0C0H, DBC1 FFFFF0C2H, DBC2 FFFFF0C4H, DBC3 FFFFF0C6H

	15	14	13	12	11	10	9	8
DBCn	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8

7	6	5	4	3	2	1	0
BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0

(n = 0 to 3)

BC15 to BC0	Byte transfer count setting or remaining byte transfer count during DMA transfer
0000H	Byte transfer count 1 or remaining byte transfer count
0001H	Byte transfer count 2 or remaining byte transfer count
:	:
FFFFH	Byte transfer count 65,536 (2^{16}) or remaining byte transfer count

Caution The DBCn register cannot be changed during transfer. Set the INIT bit of the DCHCn register to 1 to initialize this register before changing it.

20.3.4 DMA addressing control registers 0 to 3 (DADC0 to DADC3)

The DADC0 to DADC3 registers are 16-bit registers that control the DMA transfer mode for DMA channel n (n = 0 to 3). These registers cannot be accessed during DMA operation.

These registers can be read or written in 16-bit units.

After reset: 0000H R/W Address: DADC0 FFFFF0D0H, DADC1 FFFFF0D2H, DADC2 FFFFF0D4H, DADC3 FFFFF0D6H

DADCn

15	14	13	12	11	10	9	8
0	DS0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
SAD1	SAD0	DAD1	DAD0	0	0	0	0

(n = 0 to 3)

DS0	Setting of transfer data size for DMA transfer
0	8 bits
1	16 bits

SAD1	SAD0	Setting of count direction of the source address for DMA channel n
0	0	Increment
0	1	Decrement
1	0	Fixed
1	1	Setting prohibited

DAD1	DAD0	Setting of count direction of the destination address for DMA channel n
0	0	Increment
0	1	Decrement
1	0	Fixed
1	1	Setting prohibited

20.3.5 DMA channel control registers 0 to 3 (DCHC0 to DCHC3)

The DCHC0 to DCHC3 registers are 8-bit registers that control the DMA transfer operating mode for DMA channel n (n = 0 to 3).

These registers can be read or written in 8-bit or 1-bit units. (However, bit 7 is read-only and bits 1 and 2 are write-only. If bit 1 or 2 is read, the read value is always 0.)

Reset input clears these registers to 00H.

After reset: 00H R/W Address: DCHC0 FFFFF0E0H, DCHC1 FFFFF0E2H, DCHC2 FFFFF0E4H, DCHC3 FFFFF0E6H

DCHCn (n = 0 to 3)

<7>	6	5	4	3	<2>	<1>	<0>
TCn[Note 1]	0	0	0	0	INITn[Note 2]	STGn	Enn[Note 2]

TCn	Status flag indicates whether DMA transfer through DMA channel n has ended or not
0	DMA transfer had not ended.
1	DMA transfer had ended.
It is set to 1 when DMA transfer ends and cleared (to 0) when it is read.	

INITn	To change the DDAnH, DDAnL, DSAnH, DSAnL, or DBCn register before the number of the transfers set for DBCn is complete, set this bit to 1 to initialize DMA.
Set the INIT bit to 1 when the Enn bit = 0.	

STGn	If this bit is set to 1 in the DMA transfer enable state (TCn bit = 0, Enn bit = 1), DMA transfer is started.

Enn	Setting of whether DMA transfer through DMA channel n is to be enabled or disabled
0	DMA transfer disabled
1	DMA transfer enabled
This bit is cleared to 0 when DMA transfer ends. It is also cleared to 0 when DMA transfer is forcibly terminated by means of NMI input.	

Notes 1. The TCn bit is read-only.

2. The INITn and STGn bits are write-only.

Cautions 1. Before generating a DMA transfer request by software, make sure that the TCn bit is set to 1 and then clear the TCn bit to 0.

2. If the INIT bit setting and the DMA transfer of another channel conflict, initialization may not performed.

20.3.6 DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)

The DTFR0 to DTFR3 registers are 8-bit registers that control the DMA transfer start trigger through interrupt request signals from on-chip peripheral I/O.

The interrupt request signals set with these registers serve as DMA transfer start factors.

These registers can be read or written in 8-bit units. However, only bit 7 (DFn) can be read or written in 1-bit units.

Reset input clears these registers to 00H.

After reset: 00H R/W Address: DTFR0 FFFFF810H, DTFR1 FFFFF812H, DTFR2 FFFFF814H, DTFR3 FFFFF816H

	<7>	6	5	4	3	2	1	0
DTFRn	DFn	0	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0

(n = 0 to 3)

DFn[Note]	Setting of interrupt source that serves as the DMA start factor
0	No DMA transfer request
1	DMA transfer request

Note The DFn bit is a write-only bit. Write 0 to this bit to clear a DMA transfer request if the interrupt that is specified as the cause of starting DMA transfer while DMA transfer is disabled.

Cautions 1. Be sure to stop DMA operation before making changes to DTFRn register settings.

2. An interrupt request input in a standby mode (IDLE or software STOP mode) cannot be used as a DMA transfer start factor.

3. For details of IFCn5 to IFCn0 bits, see Table 20-1 DMA Start Factor.

Remark n = 0 to 3

Table 20-1. DMA Start Factor (1/2)

IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source
0	0	0	0	0	0	DMA request by interrupt disabled
0	0	0	0	0	1	INTP0
0	0	0	0	1	0	INTP1
0	0	0	0	1	1	INTP2
0	0	0	1	0	0	INTP3
0	0	0	1	0	1	INTP4
0	0	0	1	1	0	INTP5
0	0	0	1	1	1	INTP6
0	0	1	0	0	0	INTP7
0	0	1	0	0	1	INTTQ0OV
0	0	1	0	1	0	INTTQ0CC0
0	0	1	0	1	1	INTTQ0CC1
0	0	1	1	0	0	INTTQ0CC2
0	0	1	1	0	1	INTTQ0CC3
0	0	1	1	1	0	INTTP0OV
0	0	1	1	1	1	INTTP0CC0
0	1	0	0	0	0	INTTP0CC1
0	1	0	0	0	1	INTTP1OV
0	1	0	0	1	0	INTTP1CC0
0	1	0	0	1	1	INTTP1CC1
0	1	0	1	0	0	INTTP2OV
0	1	0	1	0	1	INTTP2CC0
0	1	0	1	1	0	INTTP2CC1
0	1	0	1	1	1	INTTP3CC0
0	1	1	0	0	0	INTTP3CC1
0	1	1	0	0	1	INTTP4CC0
0	1	1	0	1	0	INTTP4CC1
0	1	1	0	1	1	INTTP5CC0
0	1	1	1	0	0	INTTP5CC1
0	1	1	1	0	1	INTTM0EQ0
0	1	1	1	1	0	INTCB0R/INTIIC1[Note]
0	1	1	1	1	1	INTCB0T
1	0	0	0	0	0	INTCB1R
1	0	0	0	0	1	INTCB1T
1	0	0	0	1	0	INTCB2R
1	0	0	0	1	1	INTCB2T
1	0	0	1	0	0	INTCB3R
1	0	0	1	0	1	INTCB3T
1	0	0	1	1	0	INTUA0R/INTCB4R
1	0	0	1	1	1	INTUA0T/INTCB4T
1	0	1	0	0	0	INTUA1R/INTIIC2[Note]
1	0	1	0	0	1	INTUA1T
1	0	1	0	1	0	INTUA2R/INTIIC0[Note]

Note I^2C bus version (Y version) only

Remark n = 0 to 3

Table 20-1. DMA Start Factor (2/2)

IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source
1	0	1	0	1	1	INTUA2T
1	0	1	1	0	0	INTAD
1	0	1	1	0	1	INTKR
1	0	1	1	1	0	INTERR[Note]
1	0	1	1	1	1	INTSTA[Note]
1	1	0	0	0	0	INTIE1[Note]
Other than above						Setting prohibited

Note IEBus controller version only

Remark n = 0 to 3

20.4 DMA Bus States

20.4.1 Types of bus states

The DMAC bus states consist of the following 10 states.

(1) TI state

The TI state is an idle state, during which no access request is issued.

The DMA request signals are sampled at the rising edge of the CLKOUT signal.

(2) T0 state

DMA transfer ready state (state in which a DMA transfer request has been issued and the bus mastership is acquired for the first DMA transfer).

(3) T1R state

The bus enters the T1R state at the beginning of a read operation in the two-cycle transfer mode.

Address driving starts. After entering the T1R state, the bus enters the T2R state.

(4) T1RI state

The T1RI state is a state in which the bus waits for the acknowledge signal corresponding to an external memory read request.

After entering the last T1RI state, the bus invariably enters the T2R state.

(5) T2R state

The T2R state corresponds to the last state of a read operation in the two-cycle transfer mode, or to a wait state.

In the last T2R state, read data is sampled. After entering the last T2R state, the bus enters the T1W state.

(6) T2RI state

State in which the bus is ready for DMA transfer to on-chip peripheral I/O or internal RAM (state in which the bus mastership is acquired for DMA transfer to on-chip peripheral I/O or internal RAM).

After entering the last T2RI state, the bus invariably enters the T1W state.

(7) T1W state

The bus enters the T1W state at the beginning of a write operation in the two-cycle transfer mode.

Address driving starts. After entering the T1W state, the bus enters the T2W state.

(8) T1WI state

State in which the bus waits for the acknowledge signal corresponding to an external memory write request.

After entering the last T1WI state, the bus invariably enters the T2W state.

(9) T2W state

The T2W state corresponds to the last state of a write operation in the two-cycle transfer mode, or to a wait state.

In the last T2W state, the write strobe signal is made inactive.

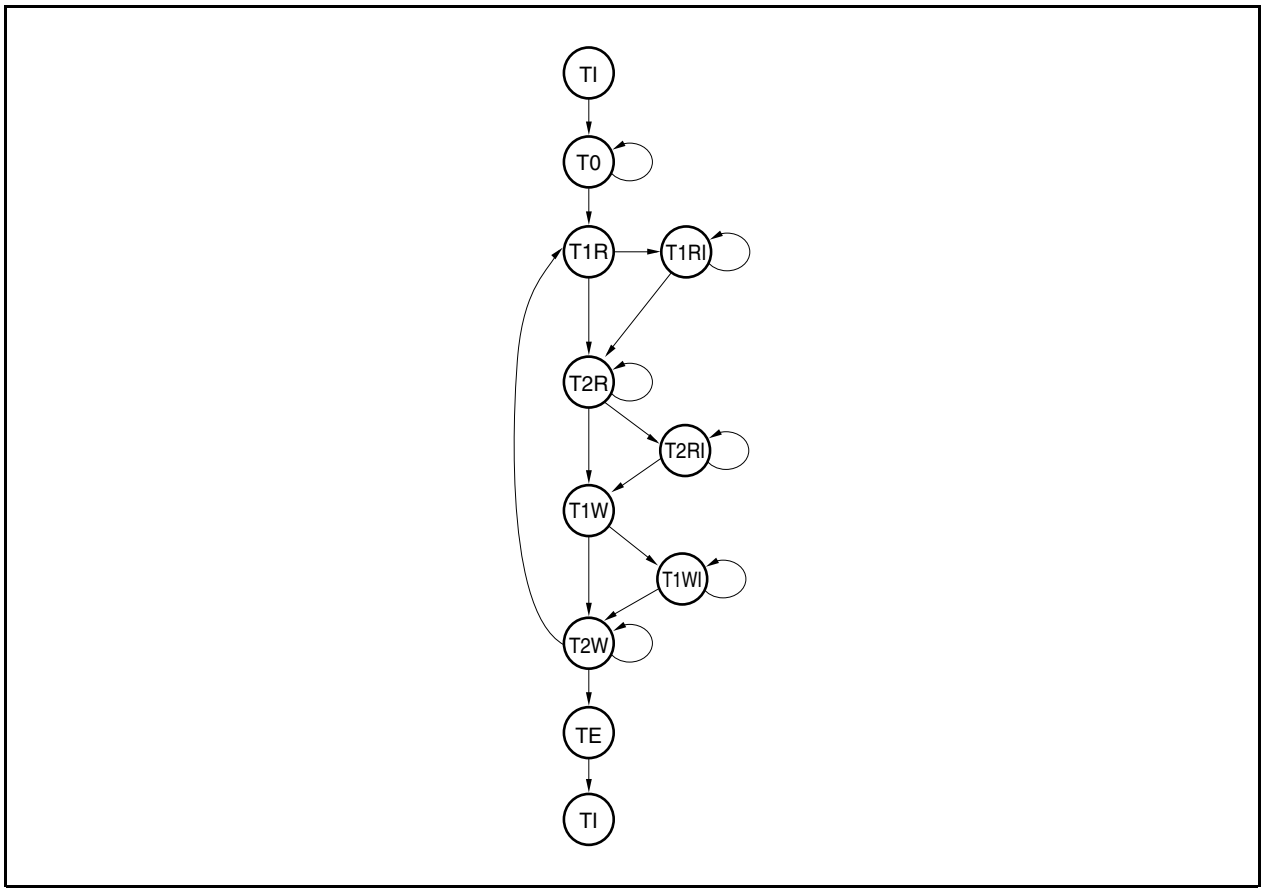
(10) TE state

The TE state corresponds to DMA transfer completion. The DMAC generates the internal DMA transfer completion signal and various internal signals are initialized. After entering the TE state, the bus invariably enters the TI state.

20.4.2 DMAC bus cycle state transition

Each time the processing for a DMA transfer is completed, the bus mastership is released.

Figure 20-1. DMAC Bus Cycle State Transition



20.5 Transfer Mode

20.5.1 Single transfer mode

In single transfer mode, the DMAC releases the bus at each byte/halfword transfer. If there is a subsequent DMA transfer request, transfer is performed again once. This operation continues until a terminal count occurs.

When the DMAC has released the bus, if another higher priority DMA transfer request is issued, the higher priority DMA request always takes precedence.

20.6 Transfer Types

20.6.1 Two-cycle transfer

In two-cycle transfer, data transfer is performed in two cycles, a read cycle (source to DMAC) and a write cycle (DMAC to destination).

In the first cycle, the source address is output and reading is performed from the source to the DMAC. In the second cycle, the destination address is output and writing is performed from the DMAC to the destination.

20.7 Transfer Object

20.7.1 Transfer object

Table 20-2 shows the relationship with transfer object (√: Transfer enabled, ×: Transfer disabled).

Table 20-2. Relationship with Transfer Object

		Destination			
		Internal ROM	On-Chip Peripheral I/O	Internal RAM	External Memory
Source	On-chip peripheral I/O	×	√	√	√
	Internal RAM	×	√	×	√
	External memory	×	√	√	√
	Internal ROM	×	×	×	×

Caution The operation is not guaranteed for combinations of transfer destination and source marked with "×" in Table 20-2.

Remark During two-cycle 16-bit transfer, if the data bus width of the transfer source and that of the transfer destination are different, the operation becomes as follows.

In the case of transfer from a 16-bit bus to an 8-bit bus
A 16-bit read cycle is generated and then an 8-bit write cycle is generated twice.

In the case of transfer from an 8-bit bus to a 16-bit bus
An 8-bit read cycle is generated twice and then a 16-bit write cycle is generated.

20.7.2 External bus cycles during DMA transfer (two-cycle transfer)

The external bus cycles during DMA transfer (two-cycle transfer) are shown below.

Table 20-3. External Bus Cycles During DMA Transfer (Two-Cycle Transfer)

Transfer Object	External Bus Cycle	
On-chip peripheral I/O, internal RAM	None[Note]	–
External I/O	Yes	SRAM cycle
External memory	Yes	Memory access cycle

Note Other external cycles such as a CPU-based bus cycle can be started.

20.8 DMA Channel Priorities

The DMA channel priorities are fixed as follows.

DMA channel 0 > DMA channel 1 > DMA channel 2 > DMA channel 3

These priorities are valid in the TI state only. In the block transfer mode, the channel used for transfer is never switched.

In the single-step transfer mode, if a higher priority DMA transfer request is issued while the bus is released (in the TI state), the higher priority DMA transfer request is acknowledged.

20.9 DMA Transfer Start Factors

There are two types of DMA transfer start factors, as shown below.

(1) Request from software

If the STGn, Enn, and TCn bits of the DCHCn register are set as follows, DMA transfer starts (n = 0 to 3).

- STGn bit = 1
- Enn bit = 1
- TCn bit = 0

(2) Request from on-chip peripheral I/O

If, when the Enn and TCn bits of the DCHCn register are set as shown below, an interrupt request is issued from the on-chip peripheral I/O that is set in the DTFRn register, DMA transfer starts (n = 0 to 3).

- Enn bit = 1
- TCn bit = 0

20.10 DMA Transfer End

20.10.1 DMA transfer end interrupt

When DMA transfer ends and the TCn bit of the DCHCn register is set to 1, a DMA transfer end interrupt (INTDMAn) is issued to the interrupt controller (INTC) (n = 0 to 3).

20.10.2 Terminal count output upon DMA transfer end

The terminal count signal becomes active for one clock during the last DMA transfer cycle.

20.11 Precautions

(1) Memory boundary

The transfer operation is not guaranteed if the source or the destination address exceeds the area of DMA objects (external memory, internal RAM, or on-chip peripheral I/O) during DMA transfer.

(2) Transfer of misaligned data

DMA transfer of 16-bit bus width misaligned data is not supported.

(3) Times related to DMA transfer

The overhead before and after DMA transfer and the minimum execution clock for DMA transfer are shown below.

- Internal RAM access: 2 clocks

Note that for external memory access, the time depends on the type of external memory connected.

(4) Bus arbitration for CPU

The CPU can access external memory, on-chip peripheral I/O, and internal RAM not undergoing DMA transfer. While data transfer among external memories or to and from I/O is being performed, the CPU can access internal RAM.

(5) Notes on INITn bit (n = 0 to 3) of DMA channel control registers 0 to 3 (DCHC0 to DCHC3)

When a channel executing DMA transfer is to be initialized, the channel may not be initialized even if the INITn bit is set to 1. To accurately initialize the channel, execute either of the following two procedures.

(a) To temporarily stop transfer of all DMA channels

[Procedure]

<1> Disable interrupts (DI status).

<2> Read the Enn bit of the DCHCn register of the DMA channels other than the one to be forcibly stopped, and transfer the value of that bit to a general-purpose register.

<3> Clear the Enn bit of the DMA channels being used (including the channel to be forcibly stopped). Execute the instruction that clears the Enn bit twice if the channel is the last DMA channel. If data is to be transferred from or to the internal RAM at this time, execute the instruction three times.
For example, execute the following instructions when channels 0, 1, and 2 are being used.
- Clear E00 bit of DCHC0 register (to 0)
- Clear E11 bit of DCHC1 register (to 0)
- Clear E22 bit of DCHC2 register (to 0)
- Clear E22 bit of DCHC2 register (to 0) again

<4> Set the INITn bit of the channel to be forcibly stopped.

<5> If both the TCn bit and Enn bit of the channels not to be stopped forcibly are 1 (if the result of ANDing is 1) as a result of step <2> above, clear the Enn bit that has been saved (to 0).

<6> Write the Enn bit to the DCHCn register after the operation in step <5> above.

<7> Enable interrupts (EI status).

Caution Be sure to perform step <5> above to prevent the Enn bit of the channels that have been completed normally in steps <2> or <3> from being illegally set again.

(b) Repeatedly setting INITn bit until DMA transfer is forcibly stopped

[Procedure]

<1> Clear the Enn bit of the DCHCn register of the channel to be forcibly stopped to 0.

<2> Clear the Enn bit of the above channel to 0 again. If data is transferred from or to the internal RAM to or from the channel to be forcibly stopped, execute step <2> again.

<3> Copy the initial number of transfers of the channel to be forcibly stopped to a general-purpose register.

<4> Set the INITn bit of the DCHCn register of the channel to be forcibly stopped to 1.

<5> Read the value of the DMA transfer count register (DBCn) of the channel to be forcibly stopped, and compare that value with the value copied in step <3> above. If the two values do not match, repeat steps <4> and <5>.

Cautions 1. If the DBCn register is read in step <5>, and if DMA transfer is stopped due to trouble, the remaining number of transfers will be read. If DMA transfer has been forcibly stopped correctly, the initial number of transfers will be read.

2. This procedure may take time in an application where DMA transfer of a channel other than that to be forcibly stopped is frequently executed until the channel in question is forcibly stopped.

(6) Program execution of internal RAM and DMA transfer

If the following condition is satisfied, the CPU may not operate correctly. If this happens, only the reset signal can be acknowledged.

[Condition]

- When DMA is executed to transfer data to/from the internal RAM

Therefore, take either of the following remedial measures.

[Remedy]

- To execute DMA transfer to transfer data to/from the internal RAM, do not execute the bit manipulation instructions (SET1, CLR1, NOT1) located on the internal RAM, and do not execute data access instruction that accesses a misaligned address.
- When executing the bit manipulation instructions (SET1, CLR1, NOT1) located on the internal RAM or data access instruction that accesses a misaligned address, do not execute DMA transfer to transfer data to/from the internal RAM.

(7) Notes on TCn bit (n = 0 to 3) of DMA channel control registers 0 to 3 (DCHC0 to DCHC3)

The TCn bit is not automatically cleared even if it is read at the specified timing. This can be avoided by the following two methods.

(a) Polling TCn bit to wait for completion of DMA transfer

After confirming that the TCn bit has been set, read the TCn bit three times.

(b) Reading TCn bit by interrupt servicing routine

Read the TCn bit three times.

20.11.1 Interrupt factors

DMA transfer is interrupted if a bus hold is issued.

If the factor (bus hold) interrupting DMA transfer disappears, DMA transfer promptly restarts.

CHAPTER 21 CRC FUNCTION

21.1 Functions

- CRC operation circuit for the detection of data block errors
- Generation of 16-bit CRC code using a CRC-CCITT ($X^{16} + X^{12} + X^5 + 1$) generating function for blocks of data of any length in 8-bit units
- CRC code is set to the CRC data register each time 1-byte data is transferred to the CRCIN register, after the initial value is set to the CRCD register.

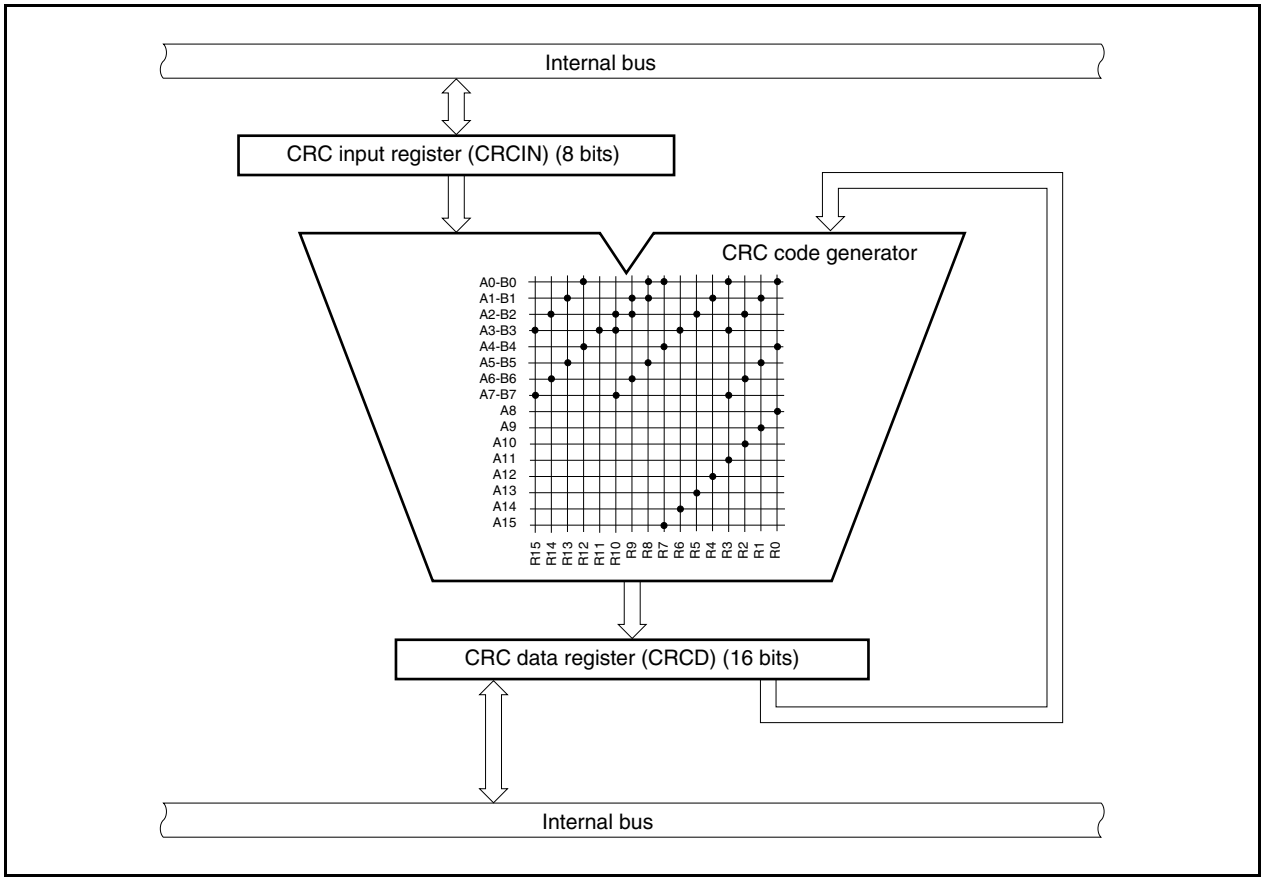
21.2 Configuration

The CRC function includes the following hardware.

Table 21-1. CRC Configuration

Item	Configuration
Control registers	CRC input register (CRCIN) CRC data register (CRCD)

Figure 21-1. Block Diagram of CRC Register



21.3 Control Registers

(1) CRC input register (CRCIN)

The CRCIN register is an 8-bit register for data setting.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF310H

	7	6	5	4	3	2	1	0
CRCIN								

(2) CRC data register (CRCD)

The CRCD register is a 16-bit register that stores CRC-CCITT operation results.

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

Reset input clears this register to 0000H.

Cautions 1. Following write to the CRCD register, do not write the CRC calculator output to the CRCD register during the first write access to the CRCIN register. (Do not load the operation result.)

2. Through the operation of the CRCIN register, the CRC operation results are saved to the CRCD register. Therefore, to write to the CRCD register via the internal bus and then read the written value, read the CRCD register before writing to CRCIN.

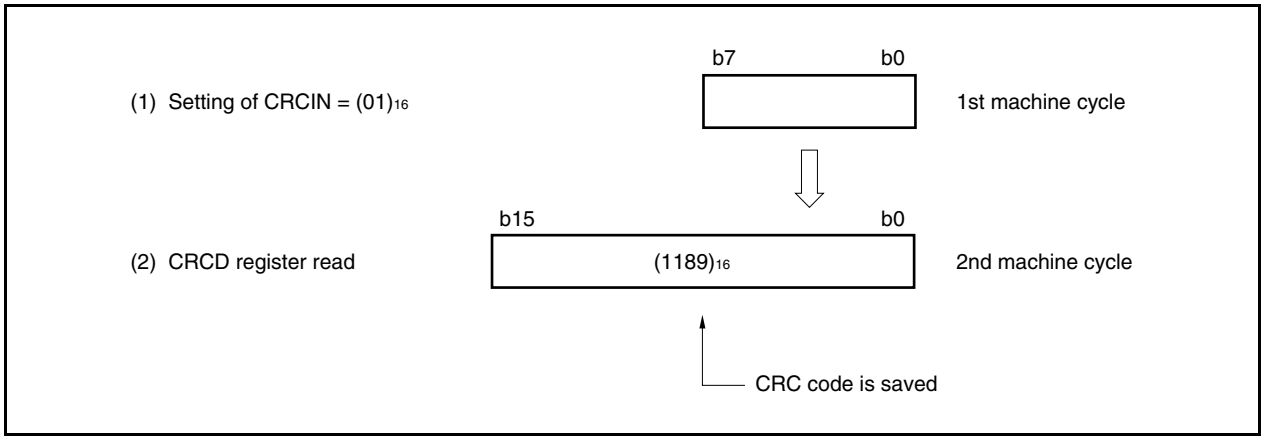
After reset: 0000H R/W Address: FFFFF312H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

21.4 Operation

21.4.1 CRC operation circuit operation example

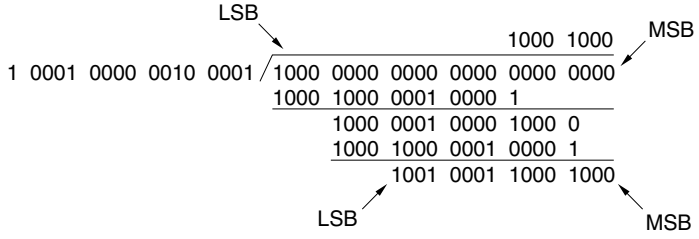
Figure 21-2. CRC Operation Circuit Operation Example (LSB First)



The code when $(01)_{16}$ is send LSB first is (1000 0000). Therefore, the CRC code with generating function $X^{16} + X^{12} + X^5 + 1$ becomes the remainder when (1000 0000) X^{16} is divided with (1 0001 0000 0010 0001) using the modulo-2 operation formula.

The modulo-2 operation is performed based on the following formula.

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0
– 1 = 1

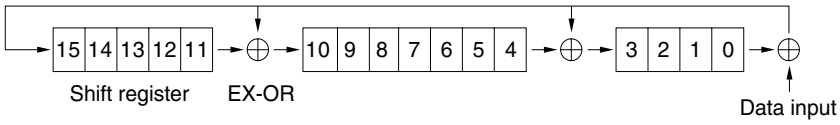


Therefore, CRC sign becomes $\overbrace{1001}^{9}\ \overbrace{0001}^{8}\ \overbrace{1000}^{1}\ \overbrace{1000}^{1}$. Since LSB first is used, this corresponds to $(1189)_{16}$ in hexadecimal notation.

21.4.2 Operation circuit configuration

The CRC operation principle is division, but CRC code can be generated by hardware using a shift register and exclusive OR (EX-OR).

Figure 21-3. Operation Circuit Configuration (CRC Data Register)



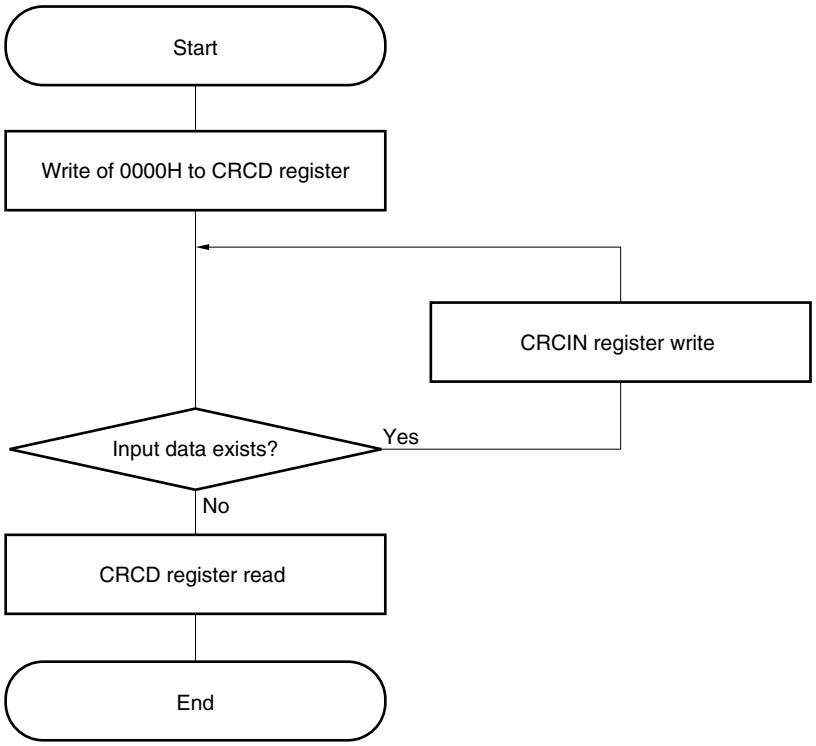
Using the denotation of data A15 to A0 of the CRCD register, data B7 to B0 of the CRCIN register, and CRC operation results R15 to R0, the CRC code generation circuit is configured as follows.

R15 = A7 ⊕ B7 ⊕ A3 ⊕ B3
R14 = A6 ⊕ B6 ⊕ A2 ⊕ B2
R13 = A5 ⊕ B5 ⊕ A1 ⊕ B1
R12 = A4 ⊕ B4 ⊕ A0 ⊕ B0
R11 = A3 ⊕ B3
R10 = A7 ⊕ B7 ⊕ A3 ⊕ B3 ⊕ A2 ⊕ B2
R9 = A6 ⊕ B6 ⊕ A2 ⊕ B2 ⊕ A1 ⊕ B1
R8 = A5 ⊕ B5 ⊕ A1 ⊕ B1 ⊕ A0 ⊕ B0
R7 = A4 ⊕ B4 ⊕ A0 ⊕ B0 ⊕ A15
R6 = A3 ⊕ B3 ⊕ A14
R5 = A2 ⊕ B2 ⊕ A13
R4 = A1 ⊕ B1 ⊕ A12
R3 = A7 ⊕ B7 ⊕ A3 ⊕ B3 ⊕ A0 ⊕ B0 ⊕ A11
R2 = A6 ⊕ B6 ⊕ A2 ⊕ B2 ⊕ A10
R1 = A5 ⊕ B5 ⊕ A1 ⊕ B1 ⊕ A9
R0 = A4 ⊕ B4 ⊕ A0 ⊕ B0 ⊕ A8

21.4.3 Usage method

The usage method of the CRC operation circuit is described below.

Figure 21-4. CRC Operation Flow



[Basic usage method]

<1> Write 0000H to the CRCD register.

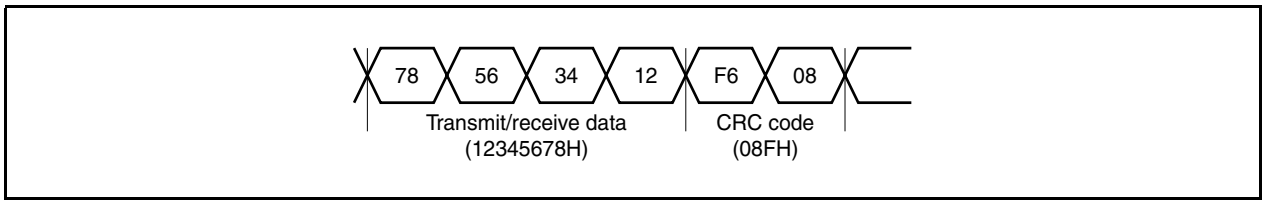
<2> Write the required quantity of data to the CRCIN register.

<3> Read the CRCD register.

Communication errors can easily be detected if the CRC code is transmitted/received along with transmit/receive data when transmitting/receiving data consisting of several bytes.

The following is an illustration using the transmission of 12345678H (0001 0010 0011 0100 0101 0110 0111 1000B) LSB first as an example.

Figure 21-5. CRC Transmission Example



Setting procedure on transmitting side

<1> Write the initial value 0000H to the CRCD register.

<2> Write the 1 byte of data to be transmitted first to the transmit buffer register. (At this time, also write the same data to the CRCIN register.)

<3> When transmitting several bytes of data, write the same data to the CRCIN register each time transmit data is written to the transmit buffer register.

<4> After all the data has been transmitted, write the contents of the CRCD register (CRC code) to the transmit buffer register and transmit them. (Since this is LSB first, transmit the data starting from the lower bytes, then the higher bytes.)

Setting procedure on receiving side

<1> Write the initial value 0000H to the CRCD register.

<2> When the reception of the first 1 byte of data has been completed, write that receive data to the CRCIN register.

<3> If receiving several bytes of data, write the receive data to the CRCIN register upon every transmission completion. (In the case of normal reception, when all the receive data has been written to the CRCIN register, the contents of the CRCD register on the receiving side and the contents of the CRCD register on the transmitting side are the same.)

<4> Next, the CRC code is transmitted from the transmitting side, so write this data to the CRCIN register similarly to receive data. (When reception of all the data, including the CRC code, has been completed, reception was normal if the contents of the CRCD register are 0000H.)

<5> If the contents of the CRCD register are other than 0000H, this indicates a communication error, so transmit a resend request to the transmitting side.

CHAPTER 22 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850ES/SG2 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 54 interrupt requests.

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

The V850ES/SG2 can process interrupt request signals from the on-chip peripheral hardware and external sources. Moreover, exception processing can be started by the TRAP instruction (software exception) or by generation of an exception event (i.e. fetching of an illegal opcode) (exception trap).

22.1 Features

○ Interrupts
- Non-maskable interrupts: 2 sources
- Maskable interrupts: External: 8, Internal: 47/51 sources (see **Table 1-1**)
- 8 levels of programmable priorities (maskable interrupts)
- Multiple interrupt control according to priority
- Masks can be specified for each maskable interrupt request.
- Noise elimination, edge detection, and valid edge specification for external interrupt request signals.

○ Exceptions
- Software exceptions: 32 sources
- Exception trap: 2 sources (illegal opcode exception)

Interrupt/exception sources are listed in Table 22-1.

Table 22-1. Interrupt Source List (1/3)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Reset	Interrupt	–	RESET	$\overline{\text{RESET}}$ pin input Reset input by internal source	RESET	0000H	00000000H	Undefined	–
Non-maskable	Interrupt	–	NMI	NMI pin valid edge input	Pin	0010H	00000010H	nextPC	–
		–	INTWDT2	WDT2 overflow	WDT2	0020H	00000020H	nextPC	–
Software exception	Exception	–	TRAP0n[Note]	TRAP instruction	–	004nH[Note]	00000040H	nextPC	–
		–	TRAP1n[Note]	TRAP instruction	–	005nH[Note]	00000050H	nextPC	–
Exception trap	Exception	–	ILGOP/ DBG0	Illegal opcode/ DBTRAP instruction	–	0060H	00000060H	nextPC	–
Maskable	Interrupt	0	INTLVI	Low voltage detection	POCLVI	0080H	00000080H	nextPC	LVIIC
		1	INTP0	External interrupt pin input edge detection (INTP0)	Pin	0090H	00000090H	nextPC	PIC0
		2	INTP1	External interrupt pin input edge detection (INTP1)	Pin	00A0H	000000A0H	nextPC	PIC1
		3	INTP2	External interrupt pin input edge detection (INTP2)	Pin	00B0H	000000B0H	nextPC	PIC2
		4	INTP3	External interrupt pin input edge detection (INTP3)	Pin	00C0H	000000C0H	nextPC	PIC3
		5	INTP4	External interrupt pin input edge detection (INTP4)	Pin	00D0H	000000D0H	nextPC	PIC4
		6	INTP5	External interrupt pin input edge detection (INTP5)	Pin	00E0H	000000E0H	nextPC	PIC5
		7	INTP6	External interrupt pin input edge detection (INTP6)	Pin	00F0H	000000F0H	nextPC	PIC6
		8	INTP7	External interrupt pin input edge detection (INTP7)	Pin	0100H	00000100H	nextPC	PIC7
		9	INTTQ0OV	TMQ0 overflow	TMQ0	0110H	00000110H	nextPC	TQ0OVIC
		10	INTTQ0CC0	TMQ0 capture 0/compare 0 match	TMQ0	0120H	00000120H	nextPC	TQ0CCIC0
		11	INTTQ0CC1	TMQ0 capture 1/compare 1 match	TMQ0	0130H	00000130H	nextPC	TQ0CCIC1
		12	INTTQ0CC2	TMQ0 capture 2/compare 2 match	TMQ0	0140H	00000140H	nextPC	TQ0CCIC2
		13	INTTQ0CC3	TMQ0 capture 3/compare 3 match	TMQ0	0150H	00000150H	nextPC	TQ0CCIC3
		14	INTTP0OV	TMP0 overflow	TMP0	0160H	00000160H	nextPC	TP0OVIC
		15	INTTP0CC0	TMP0 capture 0/compare 0 match	TMP0	0170H	00000170H	nextPC	TP0CCIC0
		16	INTTP0CC1	TMP0 capture 1/compare 1 match	TMP0	0180H	00000180H	nextPC	TP0CCIC1
		17	INTTP1OV	TMP1 overflow	TMP1	0190H	00000190H	nextPC	TP1OVIC

Note n = 0 to FH

Table 22-1. Interrupt Source List (2/3)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	18	INTTP1CC0	TMP1 capture 0/compare 0 match	TMP1	01A0H	000001AH	nextPC	TP1CCIC0
		19	INTTP1CC1	TMP1 capture 1/compare 1 match	TMP1	01B0H	000001B0H	nextPC	TP1CCIC1
		20	INTTP2OV	TMP2 overflow	TMP2	01C0H	000001C0H	nextPC	TP2OVIC
		21	INTTP2CC0	TMP2 capture 0/compare 0 match	TMP2	01D0H	000001D0H	nextPC	TP2CCIC0
		22	INTTP2CC1	TMP2 capture 1/compare 1 match	TMP2	01E0H	000001E0H	nextPC	TP2CCIC1
		23	INTTP3OV	TMP3 overflow	TMP3	01F0H	000001F0H	nextPC	TP3OVIC
		24	INTTP3CC0	TMP3 capture 0/compare 0 match	TMP3	0200H	00000200H	nextPC	TP3CCIC0
		25	INTTP3CC1	TMP3 capture 1/compare 1 match	TMP3	0210H	00000210H	nextPC	TP3CCIC1
		26	INTTP4OV	TMP4 overflow	TMP4	0220H	00000220H	nextPC	TP4OVIC
		27	INTTP4CC0	TMP4 capture 0/compare 0 match	TMP4	0230H	00000230H	nextPC	TP4CCIC0
		28	INTTP4CC1	TMP4 capture 1/compare 1 match	TMP4	0240H	00000240H	nextPC	TP4CCIC1
		29	INTTP5OV	TMP5 overflow	TMP5	0250H	00000250H	nextPC	TP5OVIC
		30	INTTP5CC0	TMP5 capture 0/compare 0 match	TMP5	0260H	00000260H	nextPC	TP5CCIC0
		31	INTTP5CC1	TMP5 capture 1/compare 1 match	TMP5	0270H	00000270H	nextPC	TP5CCIC1
		32	INTTM0EQ0	TMM0 compare match	TMM0	0280H	00000280H	nextPC	TM0EQIC0
		33	INTCB0R/ INTIIC1	CSIB0 reception completion/IIC1 transfer completion	CSIB0/ IIC1	0290H	00000290H	nextPC	CB0RIC/ IICIC1
		34	INTCB0T	CSIB0 consecutive transmission write enable	CSIB0	02A0H	000002A0H	nextPC	CB0TIC
		35	INTCB1R	CSIB1 reception completion	CSIB1	02B0H	000002B0H	nextPC	CB1RIC
		36	INTCB1T	CSIB1 consecutive transmission write enable	CSIB1	02C0H	000002C0H	nextPC	CB1TIC
		37	INTCB2R	CSIB2 reception completion	CSIB2	02D0H	000002D0H	nextPC	CB2RIC
		38	INTCB2T	CSIB2 consecutive transmission write enable	CSIB2	02E0H	000002E0H	nextPC	CB2TIC
		39	INTCB3R	CSIB3 reception completion	CSIB3	02F0H	000002F0H	nextPC	CB3RIC
		40	INTCB3T	CSIB3 consecutive transmission write enable	CSIB3	0300H	00000300H	nextPC	CB3TIC
		41	INTUA0R/ INTCB4R	UARTA0 reception completion/CSIB4 reception completion	UARTA0/ CSIB4	0310H	00000310H	nextPC	UA0RIC/ CB4RIC
		42	INTUA0T/ INTCB4T	UARTA0 transmission enable/CSIB4 consecutive transmission write enable	UARTA0/ CSIB4	0320H	00000320H	nextPC	UA0TIC/ CB4TIC

Note I^2C bus version (Y version) only

Table 22-1. Interrupt Source List (3/3)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	43	INTUA1R/ INTIIC2[Note 1]	UARTA1 reception completion/ UARTA1 reception error/ IIC2 transfer completion	UARTA1/ IIC2	0330H	00000330H	nextPC	UA1RIC/ IICIC2
		44	INTUA1T	UARTA1 transmission enable	UARTA1	0340H	00000340H	nextPC	UA1TIC
		45	INTUA2R/ INTIIC0[Note 1]	UARTA2 reception completion/ IIC0 transfer completion	UARTA/ IIC0	0350H	00000350H	nextPC	UA2RIC/ IICIC0
		46	INTUA2T	UARTA2 transmission enable	UARTA2	0360H	00000360H	nextPC	UA2TIC
		47	INTAD	A/D conversion completion	A/D	0370H	00000370H	nextPC	ADIC
		48	INTDMA0	DMA0 transfer completion	DMA	0380H	00000380H	nextPC	DMAIC0
		49	INTDMA1	DMA1 transfer completion	DMA	0390H	00000390H	nextPC	DMAIC1
		50	INTDMA2	DMA2 transfer completion	DMA	03A0H	000003A0H	nextPC	DMAIC2
		51	INTDMA3	DMA3 transfer completion	DMA	03B0H	000003B0H	nextPC	DMAIC3
		52	INTKR	Key return interrupt	KR	03C0H	000003C0H	nextPC	KRIC
		53	INTWTI	Watch timer interval	WT	03D0H	000003D0H	nextPC	WTIIC
		54	INTWT	Watch timer reference time	WT	03E0H	000003E0H	nextPC	WTIC
		55	INTC0ERR[Note 2]/ INTERR[Note 3]	AFCAN0 error/IEBus error	AFCAN0/ IEBus	03F0H	000003F0H	nextPC	ERRIC0/ ERRIC
		56	INTC0WUP[Note 2]/ INTSTA[Note 3]	AFCAN0 wakeup/ IEBus status	AFCAN0/ IEBus	0400H	00000400H	nextPC	WUPIC0/ STSAIC
		57	INTC0REC[Note 2]/ INTIE1[Note 3]	AFCAN0 reception/ IEBus data interrupt	AFCAN0/ IEBus	0410H	00000410H	nextPC	RECIC0/ IEIC1
		58	INTC0TRX[Note 2]/ INTIE2[Note 3]	AFCAN0 transmission/ IEBus error/IEBus status	AFCAN0/ IEBus	0420H	00000420H	nextPC	TRXIC0/ IEIC2

Notes 1. I^2C bus version (Y version) only

2. CAN controller version only

3. IEBus controller version only

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

Restored PC: The value of the program counter (PC) saved to EIPC or FEPC when interrupt servicing is started. Note, however, that the restored PC when a non-maskable or maskable interrupt is acknowledged while one of the following instructions is being executed does not become the nextPC (if an interrupt is acknowledged during interrupt execution, execution stops, and then resumes after the interrupt servicing has finished).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)
- Division instructions (DIV, DIVH, DIVU, DIVHU)
- PREPARE, DISPOSE instructions (only if an interrupt is generated before the stack pointer is updated)

nextPC: The PC value that starts the processing following interrupt/exception processing.

2. The execution address of the illegal instruction when an illegal opcode exception occurs is calculated by (Restored PC – 4).

22.2 Non-Maskable Interrupts

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

This product has the following two non-maskable interrupt request signals.

- NMI pin input (NMI)
- Non-maskable interrupt request signal generated by overflow of watchdog timer (INTWDT2)

The valid edge of the NMI pin can be selected from four types: “rising edge”, “falling edge”, “both edges”, and “no edge detection”.

The non-maskable interrupt request signal generated by overflow of the watchdog timer 2 (INTWDT2) functions when the WDM21 and WDM20 bits of the WDTM2 register are set to “01”.

If two or more non-maskable interrupt request signals occur at the same time, the interrupt with the higher priority is serviced, as follows (the interrupt request signal with the lower priority is ignored).

INTWDT2 > NMI

If a new NMI or INTWDT2 request signal is issued while a NMI is being serviced, it is serviced as follows.

(1) If new NMI request signal is issued while NMI is being serviced

The new NMI request signal is held pending, regardless of the value of the NP bit of PSW in the CPU. The pending NMI request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

(2) If INTWDT2 request signal is issued while NMI is being serviced

The INTWDT2 request signal is held pending if the NP bit of the PSW remains set (1) while the NMI is being serviced. The pending INTWDT2 request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

If the NP bit of PSW is cleared (0) while the NMI is being serviced, the newly generated INTWDT2 request signal is executed (the NMI servicing is stopped).

Caution If a non-maskable interrupt request signal is generated, the values of the PC and PSW are saved to the NMI status save registers (FEPC and FEPSW). At this time, execution can be returned by the RETI instruction only from an NMI signal. Execution cannot be returned while INTWDT2 signal is being serviced. Therefore, reset the system after the interrupt has been serviced.

Figure 22-1. Non-Maskable Interrupt Request Signal Acknowledgment Operation (1/2)

(a) NMI and INTWDT2 request signals generated at the same time

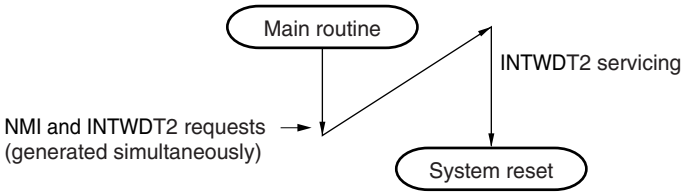
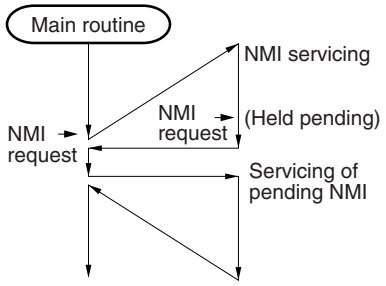
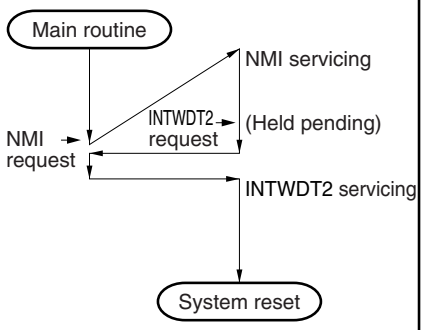
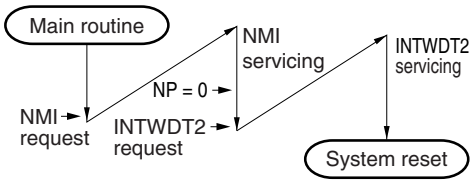
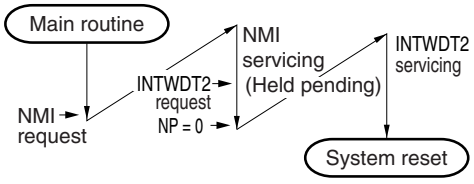
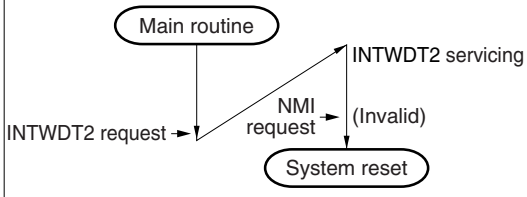
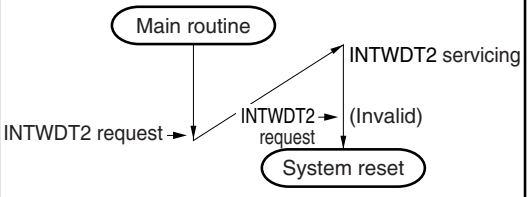


Figure 22-1. Non-Maskable Interrupt Request Signal Acknowledgment Operation (2/2)

(b) Non-maskable interrupt request signal generated during non-maskable interrupt servicing

Non-maskable interrupt being serviced	Non-maskable interrupt request signal generated during non-maskable interrupt servicing	
	NMI	INTWDT2
NMI	• NMI request generated during NMI servicing 	• INTWDT2 request generated during NMI servicing (NP = 1 retained before INTWDT2 request)  • INTWDT2 request generated during NMI servicing (NP = 0 set before INTWDT2 request)  • INTWDT2 request generated during NMI servicing (NP = 0 set after INTWDT2 request) 
INTWDT2	• NMI request generated during INTWDT2 servicing 	• INTWDT2 request generated during INTWDT2 servicing 

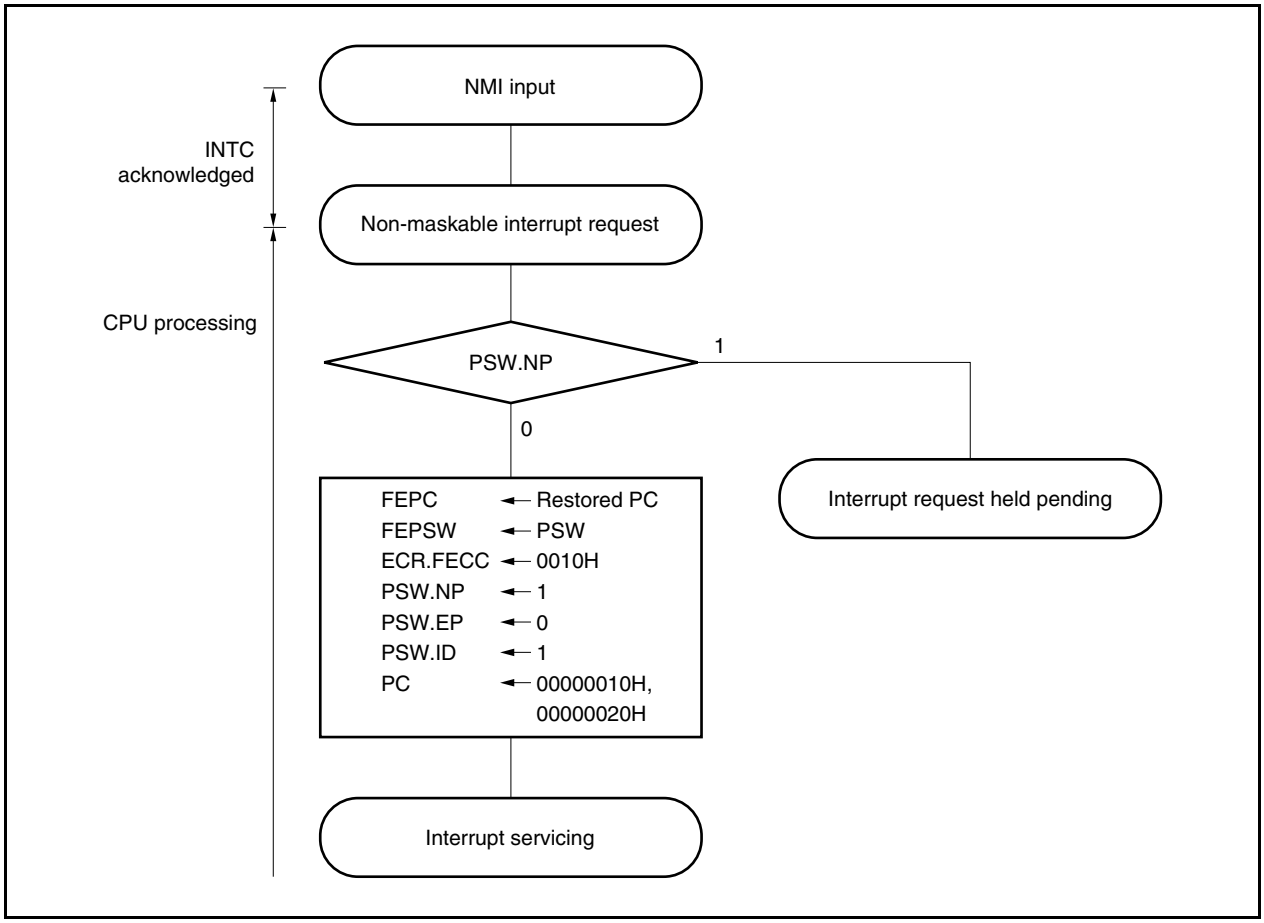
22.2.1 Operation

If a non-maskable interrupt request signal 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, 0020H) to the higher halfword (FECC) of ECR.
<4> Sets the NP and ID bits of the PSW and clears the EP bit.
<5> Sets the handler address (00000010H, 00000020H) corresponding to the non-maskable interrupt to the PC, and transfers control.

The servicing configuration of a non-maskable interrupt is shown in Figure 22-2.

Figure 22-2. Servicing Configuration of Non-Maskable Interrupt



22.2.2 Restore

(1) From NMI input

Execution is restored from the NMI servicing by the RETI instruction.

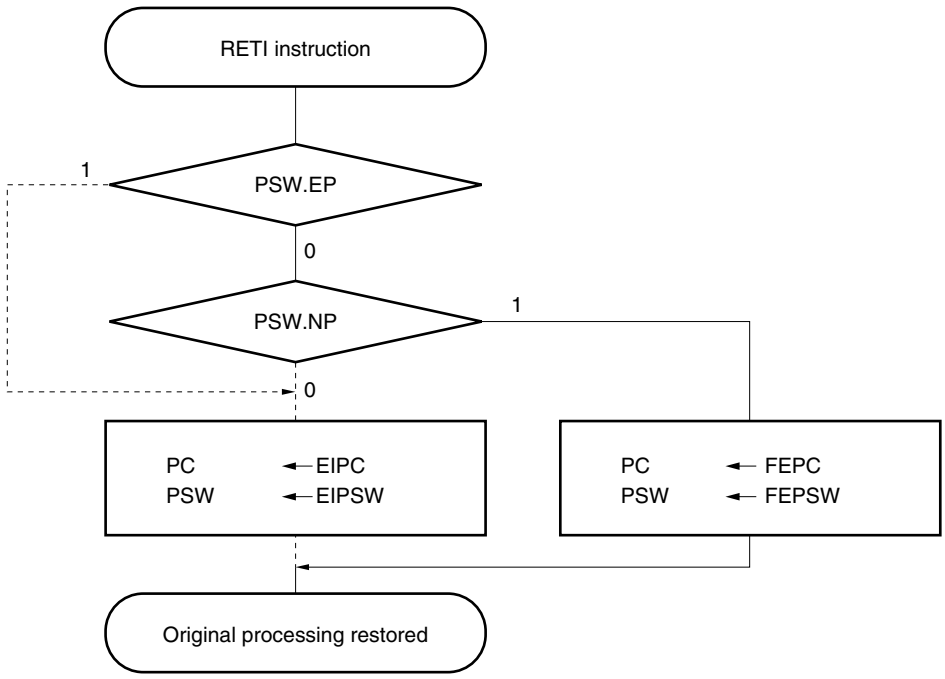
When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

<1> Loads the restored 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.

Figure 22-3 illustrates how the RETI instruction is processed.

Figure 22-3. RETI Instruction Processing



Caution When the PSW.EP bit and PSW.NP bit are changed by the LDSR instruction during non-maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set PSW.EP back to 0 and PSW.NP back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

(2) From INTWDT2 signal

Execution cannot be returned from INTWDT2 by the RETI instruction. Execute a system reset after the interrupt has been serviced.

22.2.3 NP flag

The NP flag is a status flag that indicates that non-maskable interrupt servicing is under execution.

This flag is set when a non-maskable interrupt request signal has been acknowledged, and masks non-maskable interrupt requests to prohibit multiple interrupts from being acknowledged.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	ID	SAT	CY	OV	S	Z

NP	Non-maskable interrupt servicing status
0	No non-maskable interrupt servicing
1	Non-maskable interrupt currently being serviced

22.2.4 Eliminating noise on NMI pin

The NMI pin has a noise eliminator that eliminates noise using analog delay. Unless the level input to the NMI pin is held for a specific time, therefore, it cannot be detected as an edge i.e., the edge is detected after specific time.

The NMI pin is used to release the software STOP mode. Because the internal system clock is stopped in the software STOP mode, noise is not eliminated by using the system clock.

22.2.5 Function to detect edge of NMI pin

The valid edge of the NMI pin can be selected from four types: “rising edge”, “falling edge”, “both edges”, and “no edge detection”.

Specify the valid edge of the NMI pin by using the INTR0 and INTF0 registers.

After reset, it is specified that “no edge is detected” on the NMI pin. Unless the valid edge is specified by the INTF0 and INTR0 registers, therefore, an interrupt request signal is not acknowledged (the NMI pin serves as a port pin).

To use the P00/NMI pin as an I/O port pin, specify that the valid edge of the NMI pin is “no edge detection”.

(1) External interrupt falling edge specification register 0 (INTF0)

The INTF0 register is an 8-bit register that specifies detection of the falling edge of an NMI via bit 2.

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

Reset input cleats this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC00H

	7	6	5	4	3	2	1	0
INTF0	0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0

Remark For how to specify a valid edge, see **Table 22-2**.

(2) External interrupt rising edge specification register 0 (INTR0)

The INTR0 register is an 8-bit register that specifies detection of the rising edge of the NMI pin via bit 2.

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC20H

	7	6	5	4	3	2	1	0
INTR0	0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0

Remark For how to specify a valid edge, see **Table 22-2**.

Table 22-2. NMI Valid Edge Specification

INTF02	INTR02	NMI Valid Edge Specification
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

22.3 Maskable Interrupts

Maskable interrupt request signals can be masked by interrupt control registers. The V850ES/SG2 has 54 maskable interrupt sources.

If two or more maskable interrupt request signals 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 (programmable priority control).

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

When the EI instruction is executed in an interrupt service routine, the interrupt enabled (EI) status is set, which enables servicing of interrupts having a higher priority than the interrupt request signal in progress (specified by the interrupt control register). Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

To enable multiple interrupts, however, save EIPC and EIPSW to memory or registers before executing the EI instruction, and execute the DI instruction before the RETI instruction to restore the original values of EIPC and EIPSW.

22.3.1 Operation

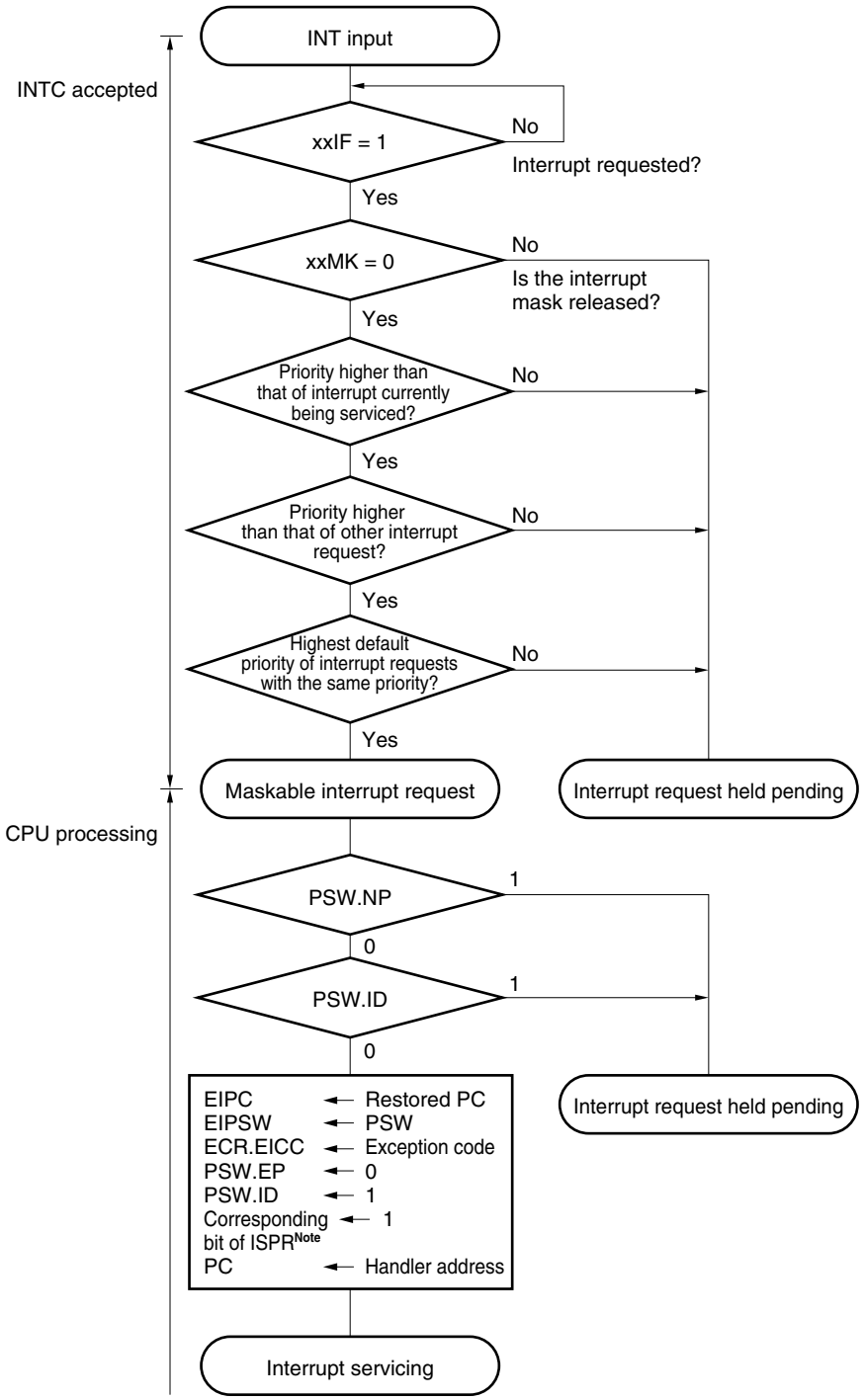
If a maskable interrupt occurs, the CPU performs the following processing, and transfers control to a handler routine.

<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> Sets the handler address corresponding to each interrupt to the PC, and transfers control.

The maskable interrupt request signal masked by INTC and the maskable interrupt request signal generated while another interrupt is being serviced (while PSW.NP = 1 or PSW.ID = 1) are held pending inside INTC. In this case, servicing a new maskable interrupt is started in accordance with the priority of the pending maskable interrupt request signal if either the maskable interrupt is unmasked or PSW.NP and PSW.ID are cleared to 0 by using the RETI or LDSR instruction.

How maskable interrupts are serviced is illustrated below.

Figure 22-4. Maskable Interrupt Servicing



Note For the ISPR register, see **22.3.6 In-service priority register (ISPR)**.

22.3.2 Restore

Recovery from maskable interrupt servicing is carried out by the RETI instruction.

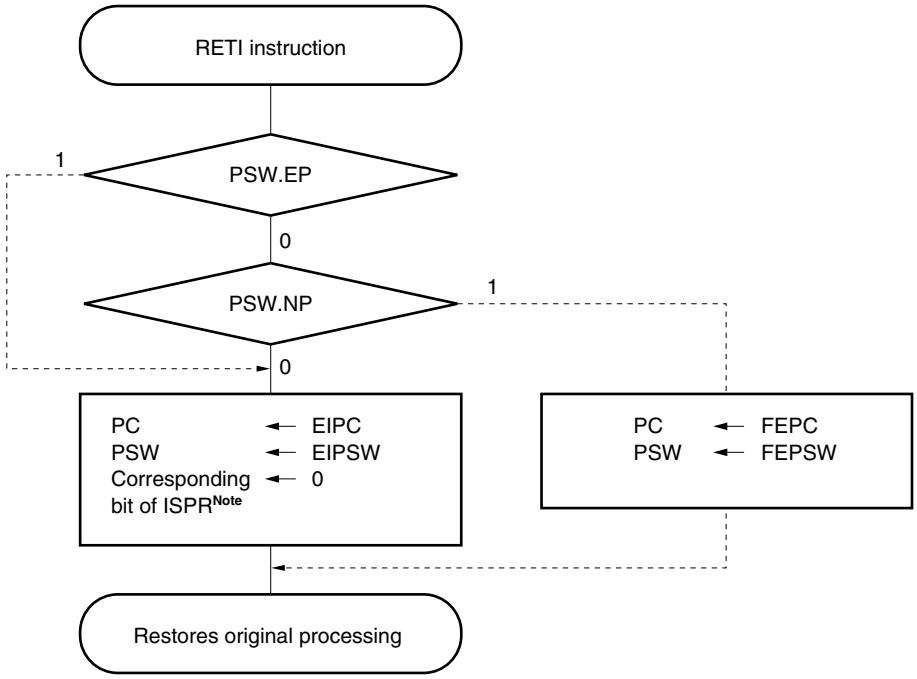
When the RETI instruction is executed, the CPU performs the following steps, and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from EIPC and EIPSW because the EP bit of the PSW is 0 and the NP bit of the PSW is 0.

<2> Transfers control to the address of the restored PC and PSW.

Figure 22-5 illustrates the processing of the RETI instruction.

Figure 22-5. RETI Instruction Processing



Note For the ISPR register, see **22.3.6 In-service priority register (ISPR)**.

Caution When the PSW.EP bit and the PSW.NP bit are changed by the LDSR instruction during maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set PSW.EP back to 0 and PSW.NP back to 0 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

22.3.3 Priorities of maskable interrupts

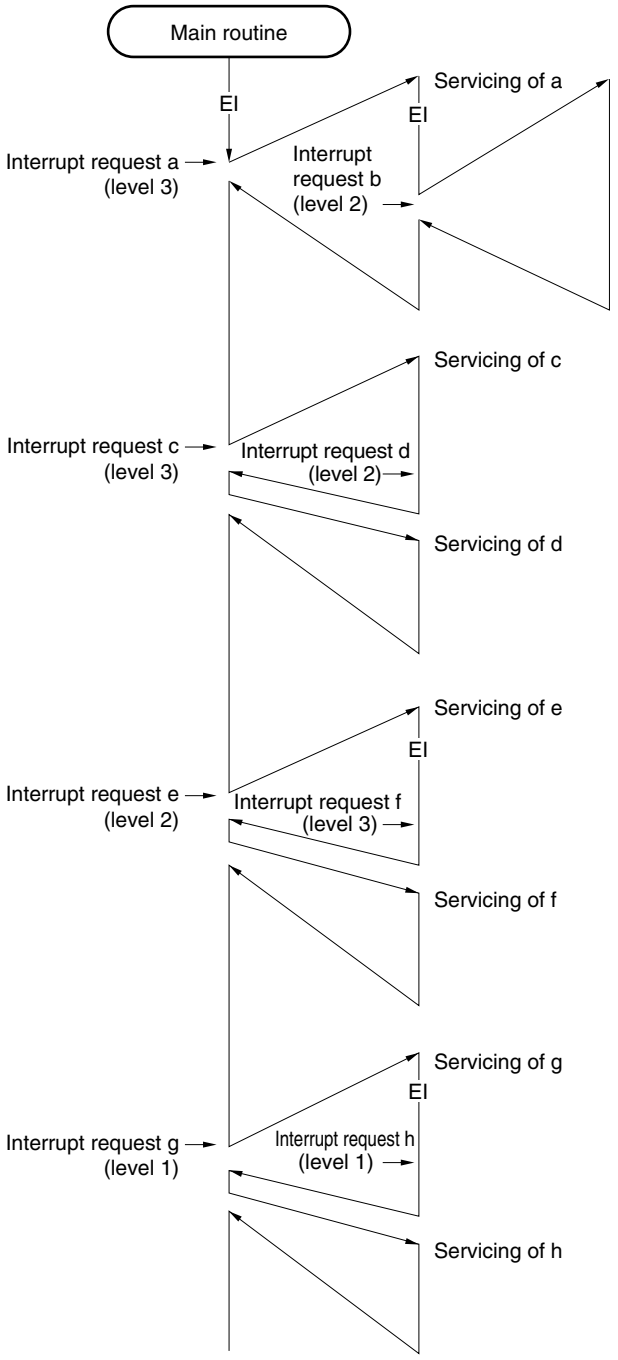
The INTC performs multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification bit (xxPRn) of the interrupt control register (xxICn). When two or more interrupts having the same priority level specified by the xxPRn bit are generated at the same time, interrupt request signals are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, see **Table 22-1 Interrupt/Exception Source List**. The programmable priority control customizes interrupt request signals into eight levels by setting the priority level specification flag.

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

Remark xx: Identification name of each peripheral unit (see **Table 22-3 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 22-3 Interrupt Control Register (xxICn)**).

Figure 22-6. Example of Processing in Which Another Interrupt Request Signal Is Issued While an Interrupt Is Being Serviced (1/2)

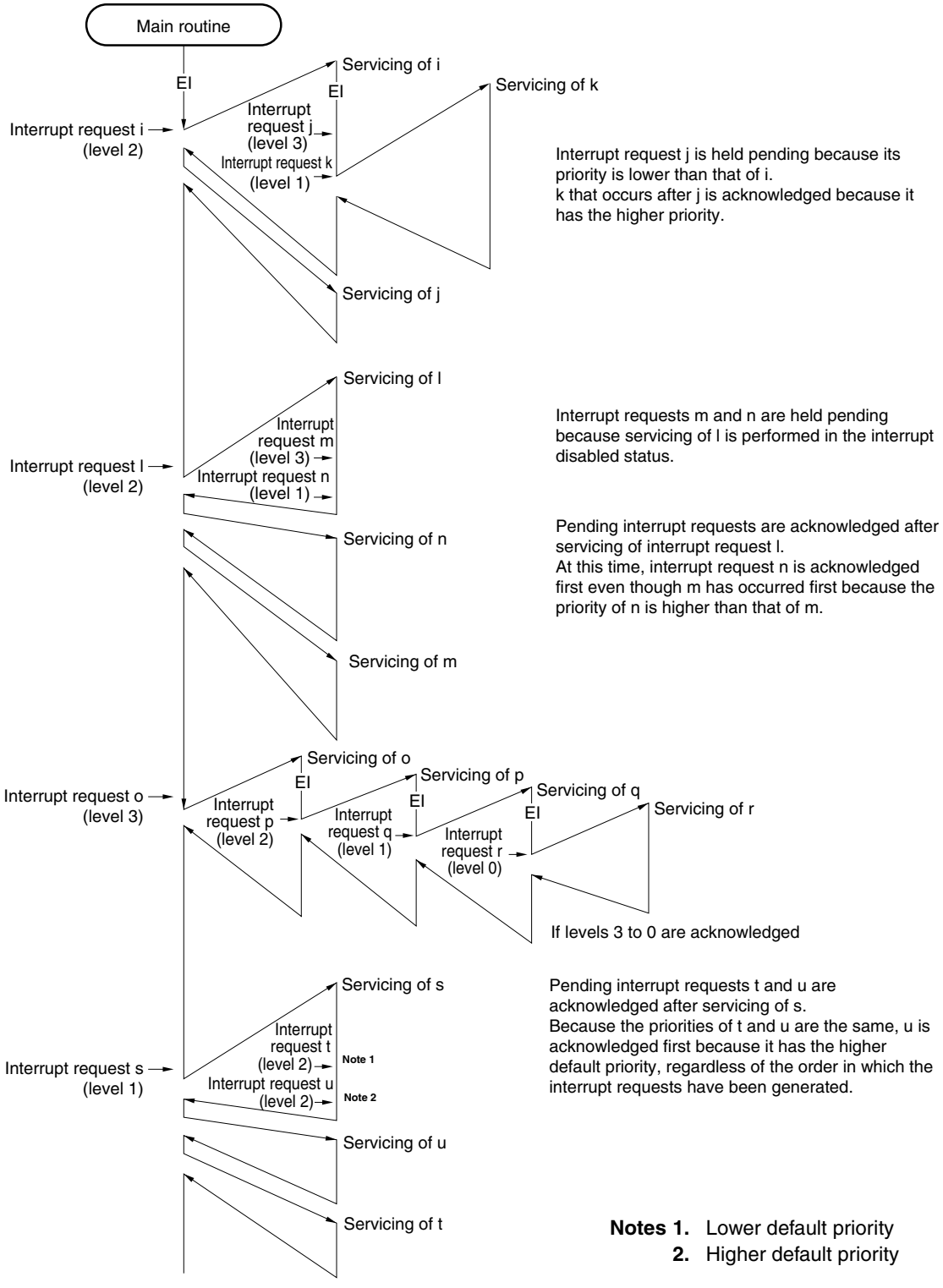


Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remarks 1. a to u in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

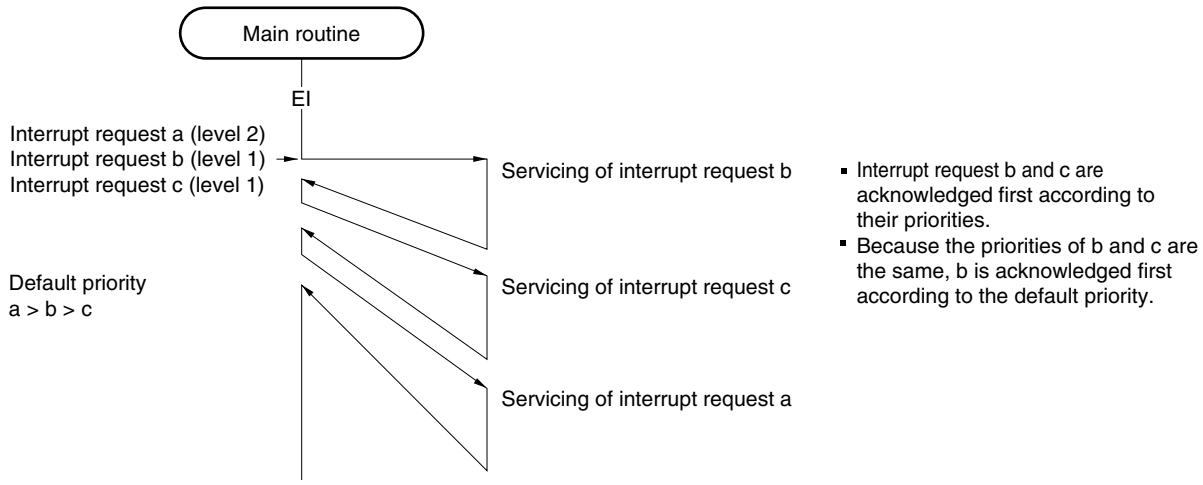
2. The default priority in the figure indicates the relative priority between two interrupt request signals.

Figure 22-6. Example of Processing in Which Another Interrupt Request Signal Is Issued While an Interrupt Is Being Serviced (2/2)



Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Figure 22-7. Example of Servicing Interrupt Request Signals Simultaneously Generated



Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remarks 1. a to c in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

2. The default priority in the figure indicates the relative priority between two interrupt request signals.

22.3.4 Interrupt control register (xxICn)

The xxICn register is assigned to each interrupt request signal (maskable interrupt) and sets the control conditions for each maskable interrupt request.

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

Reset input sets this register to 47H.

Caution Disable interrupts (DI) to read the xxIFn bit of the xxICn register. If the xxIFn bit is read while interrupts are enabled (EI), the correct value may not be read when acknowledging an interrupt and reading the bit conflict.

After reset: 47H R/W Address: FFFFF112H to FFFFF17CH

	<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 issued
1	Interrupt request issued

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 The flag xxIFn is reset automatically by the hardware if an interrupt request signal is acknowledged.

Remark xx: Identification name of each peripheral unit (see **Table 22-3 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 22-3 Interrupt Control Register (xxICn)**).

The addresses and bits of the interrupt control registers are as follows.

Table 22-3. Interrupt Control Register (xxICn) (1/2)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF110H	LVIIC	LVIIF	LVIMK	0	0	0	LVIPR2	LVIPR1	LVIPR0
FFFFF112H	PIC0	PIF0	PMK0	0	0	0	PPR02	PPR01	PPR00
FFFFF114H	PIC1	PIF1	PMK1	0	0	0	PPR12	PPR11	PPR10
FFFFF116H	PIC2	PIF2	PMK2	0	0	0	PPR22	PPR21	PPR20
FFFFF118H	PIC3	PIF3	PMK3	0	0	0	PPR32	PPR31	PPR30
FFFFF11AH	PIC4	PIF4	PMK4	0	0	0	PPR42	PPR41	PPR40
FFFFF11CH	PIC5	PIF5	PMK5	0	0	0	PPR52	PPR51	PPR50
FFFFF11EH	PIC6	PIF6	PMK6	0	0	0	PPR62	PPR61	PPR60
FFFFF120H	PIC7	PIF7	PMK7	0	0	0	PPR72	PPR71	PPR70
FFFFF122H	TQ0OVIC	TQ0OVIF	TQ0OVMK	0	0	0	TQ0OVPR2	TQ0OVPR1	TQ0OVPR0
FFFFF124H	TQ0CCIC0	TQ0CCIF0	TQ0CCMK0	0	0	0	TQ0CCPR02	TQ0CCPR01	TQ0CCPR00
FFFFF126H	TQ0CCIC1	TQ0CCIF1	TQ0CCMK1	0	0	0	TQ0CCPR12	TQ0CCPR11	TQ0CCPR10
FFFFF128H	TQ0CCIC2	TQ0CCIF2	TQ0CCMK2	0	0	0	TQ0CCPR22	TQ0CCPR21	TQ0CCPR20
FFFFF12AH	TQ0CCIC3	TQ0CCIF3	TQ0CCMK3	0	0	0	TQ0CCPR32	TQ0CCPR31	TQ0CCPR30
FFFFF12CH	TP0OVIC	TP0OVIF	TP0OVMK	0	0	0	TP0OVPR2	TP0OVPR1	TP0OVPR0
FFFFF12EH	TP0CCIC0	TP0CCIF0	TP0CCMK0	0	0	0	TP0CCPR02	TP0CCPR01	TP0CCPR00
FFFFF130H	TP0CCIC1	TP0CCIF1	TP0CCMK1	0	0	0	TP0CCPR12	TP0CCPR11	TP0CCPR10
FFFFF132H	TP1OVIC	TP1OVIF	TP1OVMK	0	0	0	TP1OVPR2	TP1OVPR1	TP1OVPR0
FFFFF134H	TP1CCIC0	TP1CCIF0	TP1CCMK0	0	0	0	TP1CCPR02	TP1CCPR01	TP1CCPR00
FFFFF136H	TP1CCIC1	TP1CCIF1	TP1CCMK1	0	0	0	TP1CCPR12	TP1CCPR11	TP1CCPR10
FFFFF138H	TP2OVIC	TP2OVIF	TP2OVMK	0	0	0	TP2OVPR2	TP2OVPR1	TP2OVPR0
FFFFF13AH	TP2CCIC0	TP2CCIF0	TP2CCMK0	0	0	0	TP2CCPR02	TP2CCPR01	TP2CCPR00
FFFFF13CH	TP2CCIC1	TP2CCIF1	TP2CCMK1	0	0	0	TP2CCPR12	TP2CCPR11	TP2CCPR10
FFFFF13EH	TP3OVIC	TP3OVIF	TP3OVMK	0	0	0	TP3OVPR2	TP3OVPR1	TP3OVPR0
FFFFF140H	TP3CCIC0	TP3CCIF0	TP3CCMK0	0	0	0	TP3CCPR02	TP3CCPR01	TP3CCPR00
FFFFF142H	TP3CCIC1	TP3CCIF1	TP3CCMK1	0	0	0	TP3CCPR12	TP3CCPR11	TP3CCPR10
FFFFF144H	TP4OVIC	TP4OVIF	TP4OVMK	0	0	0	TP4OVPR2	TP4OVPR1	TP4OVPR0
FFFFF146H	TP4CCIC0	TP4CCIF0	TP4CCMK0	0	0	0	TP4CCPR02	TP4CCPR01	TP4CCPR00
FFFFF148H	TP4CCIC1	TP4CCIF1	TP4CCMK1	0	0	0	TP4CCPR12	TP4CCPR11	TP4CCPR10
FFFFF14AH	TP5OVIC	TP5OVIF	TP5OVMK	0	0	0	TP5OVPR2	TP5OVPR1	TP5OVPR0
FFFFF14CH	TP5CCIC0	TP5CCIF0	TP5CCMK0	0	0	0	TP5CCPR02	TP5CCPR01	TP5CCPR00
FFFFF14EH	TP5CCIC1	TP5CCIF1	TP5CCMK1	0	0	0	TP5CCPR12	TP5CCPR11	TP5CCPR10
FFFFF150H	TM0EQIC0	TM0EQIF0	TM0EQMK0	0	0	0	TM0EQPR02	TM0EQPR01	TM0EQPR00
FFFFF152H	CB0RIC/ IICIC1[Note]	CB0RIF/ IICIF1	CB0RMK/ IICMK1	0	0	0	CB0RPR2/ IICPR12	CB0RPR1/ IICPR11	CB0RPR0/ IICPR10
FFFFF154H	CB0TIC	CB0TIF	CB0TMK	0	0	0	CB0TPR2	CB0TPR1	CB0TPR0
FFFFF156H	CB1RIC	CB1RIF	CB1RMK	0	0	0	CB1RPR2	CB1RPR1	CB1RPR0
FFFFF158H	CB1TIC	CB1TIF	CB1TMK	0	0	0	CB1TPR2	CB1TPR1	CB1TPR0
FFFFF15AH	CB2RIC	CB2RIF	CB2RMK	0	0	0	CB2RPR2	CB2RPR1	CB2RPR0
FFFFF15CH	CB2TIC	CB2TIF	CB2TMK	0	0	0	CB2TPR2	CB2TPR1	CB2TPR0
FFFFF15EH	CB3RIC	CB3RIF	CB3RMK	0	0	0	CB3RPR2	CB3RPR1	CB3RPR0
FFFFF160H	CB3TIC	CB3TIF	CB3TMK	0	0	0	CB3TPR2	CB3TPR1	CB3TPR0

Note I^2C bus version (Y version) only

Table 22-3. Interrupt Control Register (xxICn) (2/2)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF162H	UA0RIC/ CB4RIC	UA0RIF/ CB4RIF	UA0RMK/ CB4RMK	0	0	0	UA0RPR2/ CB4RPR2	UA0RPR1/ CB4RPR1	UA0RPR0/ CB4RPR0
FFFFF164H	UA0TIC/ CB4TIC	UA0TIF/ CB4TIF	UA0TMK/ CB4TMK	0	0	0	UA0TPR2/ CB4TPR2	UA0TPR1/ CB4TPR1	UA0TPR0/ CB4TPR0
FFFFF166H	UA1RIC/ IICIC2Note 1	UA1RIF/ IICIF2	UA1RMK/ IICMK2	0	0	0	UA1RPR2/ IICPR22	UA1RPR1/ IICPR21	UA1RPR0/ IICPR20
FFFFF168H	UA1TIC	UA1TIF	UA1TMK	0	0	0	UA1TPR2	UA1TPR1	UA1TPR0
FFFFF16AH	UA2RIC/ IICIC0Note 1	UA2RIF/ IICIF0	UA2RMK/ IICMK0	0	0	0	UA2RPR2/ IICPR02	UA2RPR1/ IICPR01	UA2RPR0/ IICPR00
FFFFF16CH	UA2TIC	UA2TIF	UA2TMK	0	0	0	UA2TPR2	UA2TPR1	UA2TPR0
FFFFF16EH	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
FFFFF170H	DMAIC0	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
FFFFF172H	DMAIC1	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
FFFFF174H	DMAIC2	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
FFFFF176H	DMAIC3	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30
FFFFF178H	KRIC	KRIF	KRMK	0	0	0	KRPR2	KRPR1	KRPR0
FFFFF17AH	WTIIC	WTIIF	WTIMK	0	0	0	WTIPR2	WTIPR1	WTIPR0
FFFFF17CH	WTIC	WTIF	WTMK	0	0	0	WTPR2	WTPR1	WTPR0
FFFFF17EH	ERRIC0Note 2/ ERRICNote 3	ERRIF0/ ERRIF	ERRMK0/ ERRMK	0	0	0	ERRPR02/ ERRPR2	ERRPR01/ ERRPR1	ERRPR00/ ERRPR0
FFFFF180H	WUPIC0Note 2/ STAICNote 3	WUPIF0/ STAIF	WUPMK0/ STAMI	0	0	0	WUPPR02/ STAPR2	WUPPR01/ STAPR1	WUPPR00/ STAPR0
FFFFF182H	RECIC0Note 2/ IEIC1Note 3	RECIF0/ IEIF1	RECMK0/ IEMK1	0	0	0	RECPR02/ IEPR12	RECPR01/ IEPR11	RECPR00/ IEPR10
FFFFF184H	TRXIC0Note 2/ IEIC2Note 3	TRXIF0/ IEIF2	TRXMK0/ IEMK2	0	0	0	TRXPR02/ IEPR22	TRXPR01/ IEPR21	TRXPR00/ IEPR20

Notes 1. I^2C bus version (Y version) only

2. CAN controller version only

3. IEBus controller version only

22.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)

The IMR0 to IMR3 registers set the interrupt mask state for the maskable interrupts. The xxMKn bit of the IMR0 to IMR3 registers is equivalent to the xxMKn bit of the xxICn register.

The IMRm register can be read or written in 16-bit units (m = 0 to 3).

If the higher 8 bits of the IMRm register are used as an IMRmH register and the lower 8 bits as an IMRmL register, these registers can be read or written in 8-bit or 1-bit units (m = 0 to 3).

Bits 11 to 15 of the IMR3 register are fixed to 1. If these bits are not 1, the operation cannot be guaranteed.

Reset input sets these registers to FFFFH.

Caution The device file defines the xxMKn bit of the xxICn register as a reserved word. If a bit is manipulated using the name of xxMKn, the contents of the xxICn register, instead of the IMRm register, are rewritten (as a result, the contents of the IMRm register are also rewritten).

After reset: FFFFH R/W Address: FFFFF106H

	15	14	13	12	11	10	9	8
IMR3	1	1	1	1	1	TRXMK0/ IEMK2	RECMK0/ IEMK1	WUPMK0/ STAMK
	7	6	5	4	3	2	1	0
	ERRMK0/ ERRMK	WTMK	WTIMK	KRMK	DMAMK3	DMAMK2	DMAMK1	DMAMK0

After reset: FFFFH R/W Address: FFFFF104H

	15	14	13	12	11	10	9	8
IMR2	ADMK	UA2TMK	UA2RMK/ IICMK0	UA1TMK	UA1RMK/ IIC2MK	UA0TMK/ CB4TMK	UA0RMK/ CB4RMK	CB3TMK
	7	6	5	4	3	2	1	0
	CB3RMK	CB2TMK	CB2RMK	CB1TMK	CB1RMK	CB0TMK	CB0RMK/ IICMK1	TM0EQMK0

After reset: FFFFH R/W Address: FFFFF102H

	15	14	13	12	11	10	9	8
IMR1	TP5CCMK1	TP5CCMK0	TP5OVMK	TP4CCMK1	TP4CCMK0	TP4OVMK	TP3CCMK1	TP3CCMK0
	7	6	5	4	3	2	1	0
	TP3OVMK	TP2CCMK1	TP2CCMK0	TP2OVMK	TP1CCMK1	TP1CCMK0	TP1OVMK	TP0CCMK1

After reset: FFFFH R/W Address: FFFFF100H

	15	14	13	12	11	10	9	8
IMR0	TP0CCMK0	TP0OVMK	TQ0CCMK3	TQ0CCMK2	TQ0CCMK1	TQ0CCMK0	TQ0OVMK	PMK7
	7	6	5	4	3	2	1	0
	PMK6	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	LVIMK

xxMKn	Setting of interrupt mask flag
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Remark xx: Identification name of each peripheral unit (see **Table 22-3 Interrupt Control Register (xxICn)**).

n: Peripheral unit number (see **Table 22-3 Interrupt Control Register (xxICn)**)

22.3.6 In-service priority register (ISPR)

The ISPR register holds the priority level of the maskable interrupt currently acknowledged. When an interrupt request signal is acknowledged, the bit of this register corresponding to the priority level of that interrupt request signal is set to 1 and remains set while the interrupt is serviced.

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

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

Reset input clears this register to 00H.

Caution If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).

After reset: 00H R Address: FFFFF1FAH

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
ISPR	ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0

ISPRn	Priority of interrupt currently acknowledged
0	Interrupt request signal with priority n not acknowledged
1	Interrupt request signal with priority n acknowledged

Remark n = 0 to 7 (priority level)

22.3.7 ID flag

This flag controls the maskable interrupt's operating state, and stores control information regarding enabling or disabling of interrupt request signals. An interrupt disable flag (ID) is incorporated, which is assigned to the PSW.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	ID	SAT	CY	OV	S	Z

ID	Specification of maskable interrupt servicing[Note]
0	Maskable interrupt request signal acknowledgment enabled
1	Maskable interrupt request signal acknowledgment disabled (pending)

Note Interrupt disable flag (ID) function

This bit is set to 1 by the DI instruction and reset to 0 by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW.

Non-maskable interrupt request signals and exceptions are acknowledged regardless of this flag. When a maskable interrupt request signal is acknowledged, the ID flag is automatically set to 1 by hardware.

The interrupt request signal generated during the acknowledgment disabled period (ID = 1) is acknowledged when the xxIFn bit of xxICn is set to 1, and the ID flag is reset to 0.

22.3.8 Watchdog timer mode register 2 (WDTM2)

The WDTM2 register is a special register and write-only in a specific sequence.

This register can be read or written in 8-bit or 1-bit units (for details, see **CHAPTER 12 FUNCTIONS OF WATCHDOG TIMER 2**).

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF6D0H

	7	6	5	4	3	2	1	0
WDTM2	0	WDM21	WDM20	0	0	0	0	0

WDM21	WDM20	Selection of watchdog timer operation mode
0	0	Stops operation
0	1	Non-maskable interrupt request mode
1	×	Reset mode (initial-value)

22.3.9 Eliminating noise on INTP0 to INTP7 pins

The INTP0 to INTP7 pins have a noise eliminator that eliminates noise using analog delay. Unless the level input to each pin is held for a specific time, therefore, it cannot be detected as a signal edge i.e., the edge is detected after specific time.

22.3.10 Function to detect edge of INTP0 to INTP7 pins

The valid edge of the INTP0 to INTP7 pins can be selected from the following four.

- Rising edge
- Falling edge
- Both edges
- No edge detection

(1) External interrupt falling edge specification register 0 (INTF0)

The INTF0 register is an 8-bit register that specifies detection of the falling edge of the external interrupt pins (INTP0 to INTP3) by bits 3 to 6.

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC00H

	7	6	5	4	3	2	1	0
INTF0	0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0

Remark For how to specify a valid edge, see **Table 22-4**.

(2) External interrupt rising edge specification register 0 (INTR0)

The INTR0 register is an 8-bit register that specifies detection of the rising edge of the external interrupt pins (INTP0 to INTP3) via bits 3 to 6.

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC20H

	7	6	5	4	3	2	1	0
INTR0	0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0

Remark For how to specify a valid edge, see **Table 22-4**.

Table 22-4. Valid Edge Specification

INTF0n	INTR0n	Valid Edge Specification (n = 2 to 6)
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Remark n = 3 to 6: Control of INTP0 to INTP3 pins

(3) External interrupt falling edge specification register 3L (INTF3L)

The INTF3L register is an 8-bit register that specifies detection of the falling edge of the external interrupt pin (INTP7).

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

Reset input cleats this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF31 and INTR31 bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC06H

	7	6	5	4	3	2	1	0
INTF3L	0	0	0	0	0	0	INTF31	0

Remark For how to specify a valid edge, see **Table 22-5**.

(4) External interrupt rising edge specification register 3L (INTR3L)

The INTR3L register is an 8-bit register that specifies detection of the rising edge of the external interrupt pin (INTP7).

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF31 and INTR31 bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC26H

	7	6	5	4	3	2	1	0
INTR3	0	0	0	0	0	0	INTR31	0

Remark For how to specify a valid edge, see **Table 22-5**.

Table 22-5. Valid Edge Specification

INTF31	INTR31	Valid Edge Specification
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution Be sure to clear the INTF31 and INTR31 bits to 00 when these registers are not used as INTP7.

(5) External interrupt falling edge specification register 9H (INTF9H)

The INTF9H register is an 8-bit register that specifies detection of the falling edge of the external interrupt pins (INTP4 to INTP6).

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF9n and INTR9n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC13H

	15	14	13	12	11	10	9	8
INTF9H	INTH915	INTF914	INTF913	0	0	0	0	0

Remark For how to specify a valid edge, see **Table 22-6**.

(6) External interrupt rising edge specification register 9H (INTR9H)

The INTR9H register is an 8-bit register that specifies detection of the rising edge of the external interrupt pins (INTP4 to INTP6).

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

Reset input clears this register to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF9n and INTR9n bits to 0, and then set the port mode.

After reset: 00H R/W Address: FFFFFC33H

	15	14	13	12	11	10	9	8
INTR9H	INTR915	INTR914	INTR913	0	0	0	0	0

Remark For how to specify a valid edge, see **Table 22-6**.

Table 22-6. Valid Edge Specification

INTF9n	INTR9n	Valid Edge Specification (n = 13 to 15)
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution Be sure to clear the INTF9n and INTR9n bits to 00 when these registers are not used as INTP4 to INTP6.

Remark n = 13 to 15: Control of INTP4 to INTP6 pins

(7) Noise elimination control register

Digital noise elimination can be selected for the INTP3 pin. The noise elimination settings are performed with the NFC register.

When digital noise elimination is selected, the sampling clock for digital sampling can be selected from among $f_{XX}/64$, $f_{XX}/128$, $f_{XX}/256$, $f_{XX}/512$, $f_{XX}/1,024$, and f_{XT}. Sampling is performed 3 times.

Even when digital noise elimination is selected, using f_{XT} as the sampling clock makes it possible to use the INTP3 interrupt request signal to release the IDLE and STOP modes.

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

Reset input clears this register to 00H.

Caution After the sampling clock has been changed, it takes 3 sampling clocks to initialize the digital noise eliminator. Therefore, if an INTP3 valid edge is input within these 3 sampling clocks after the sampling clock has been changed, an interrupt request signal may occur. Therefore, be careful about the following points when using the DMA function.

- **When using the interrupt function, after the 3 sampling clocks have elapsed, allow the interrupt after the interrupt request flag (bit 7 of PIC3) has been cleared.**
- **When using the DMA function (started with INTP3), enable DMA after 3 sampling clocks have elapsed.**

After reset: 00H R/W Address: FFFFF318H

	7	6	5	4	3	2	1	0
NFC	NFEN	0	0	0	0	NFC2	NFC1	NFC0

NFEN	Settings of INTP3 pin noise elimination
0	Analog noise elimination (60 ns (TYP.))
1	Digital noise elimination

NFC2	NFC1	NFC0	Digital sampling clock
0	0	0	$f_{XX}/64$
0	0	1	$f_{XX}/128$
0	1	0	$f_{XX}/256$
0	1	1	$f_{XX}/512$
1	0	0	$f_{XX}/1,024$
1	0	1	f_{XT} (subclock)
Other than above			Setting prohibited

Remarks 1. Since sampling is performed 3 times, the reliably eliminated noise width is 2 sampling clocks.

2. In the case of noise with a width smaller than 2 sampling clocks, an interrupt request signal is generated if noise synchronized with the sampling clock is input.

22.4 Software Exception

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

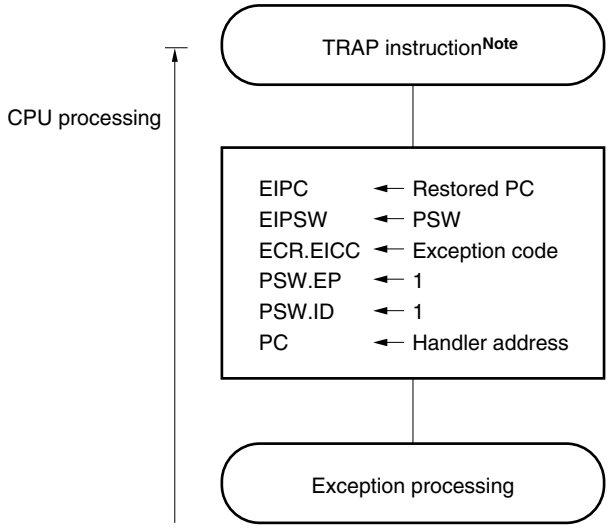
22.4.1 Operation

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> Sets the handler address (00000040H or 00000050H) corresponding to the software exception to the PC, and transfers control.

Figure 22-8 illustrates the processing of a software exception.

Figure 22-8. Software Exception Processing



Note TRAP instruction format: TRAP vector (the vector is a value from 0 to 1FH.)

The handler address is determined by the TRAP instruction's operand (vector). If the vector is 0 to 0FH, it becomes 00000040H, and if the vector is 10H to 1FH, it becomes 00000050H.

22.4.2 Restore

Recovery from software exception processing is carried out by the RETI instruction.

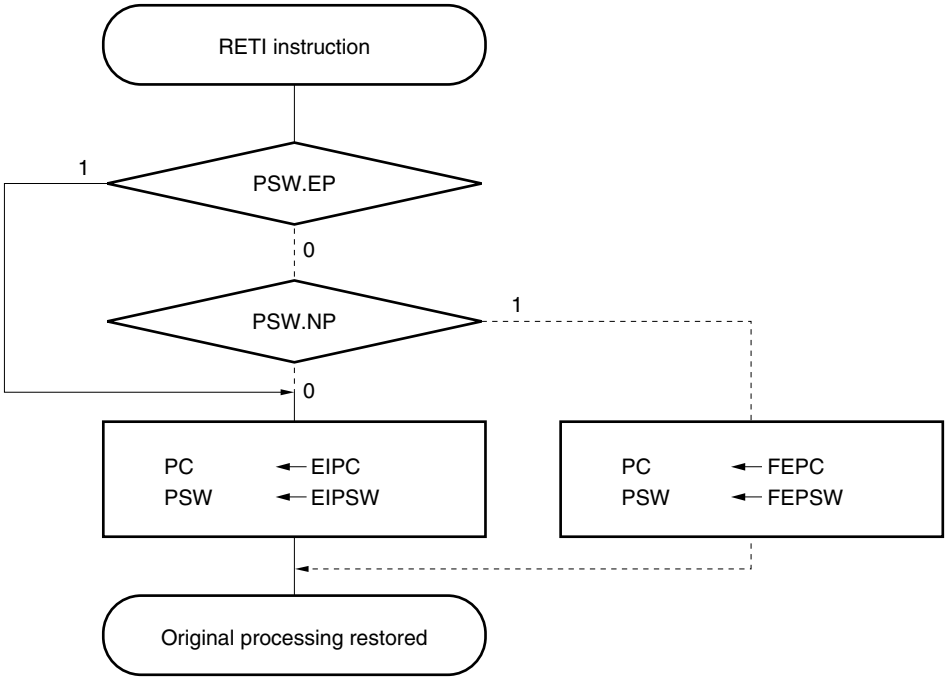
By executing the RETI instruction, the CPU carries out the following processing and shifts control to the restored PC's address.

<1> Loads 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.

Figure 22-9 illustrates the processing of the RETI instruction.

Figure 22-9. RETI Instruction Processing



Caution When the PSW.EP bit and the PSW.NP bit are changed by the LDSR instruction during the software exception processing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set PSW.EP back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

22.4.3 EP flag

The EP flag is bit 6 of the PSW, and is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	**EP**	ID	SAT	CY	OV	S	Z

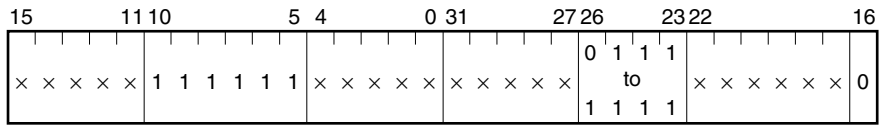
EP	Exception processing status
0	Exception processing not in progress.
1	Exception processing in progress.

22.5 Exception Trap

An exception trap is an interrupt that is requested when the illegal execution of an instruction takes place. In the V850ES/SG2, an illegal opcode exception (ILGOP: Illegal Opcode Trap) is considered as an exception trap.

22.5.1 Illegal opcode definition

The illegal instruction has an opcode (bits 10 to 5) of 111111B, a sub-opcode (bits 26 to 23) of 0111B to 1111B, and a sub-opcode (bit 16) of 0B. An exception trap is generated when an instruction applicable to this illegal instruction is executed.



×: Arbitrary

Caution Since it is possible to assign this instruction to an illegal opcode in the future, it is recommended that it not be used.

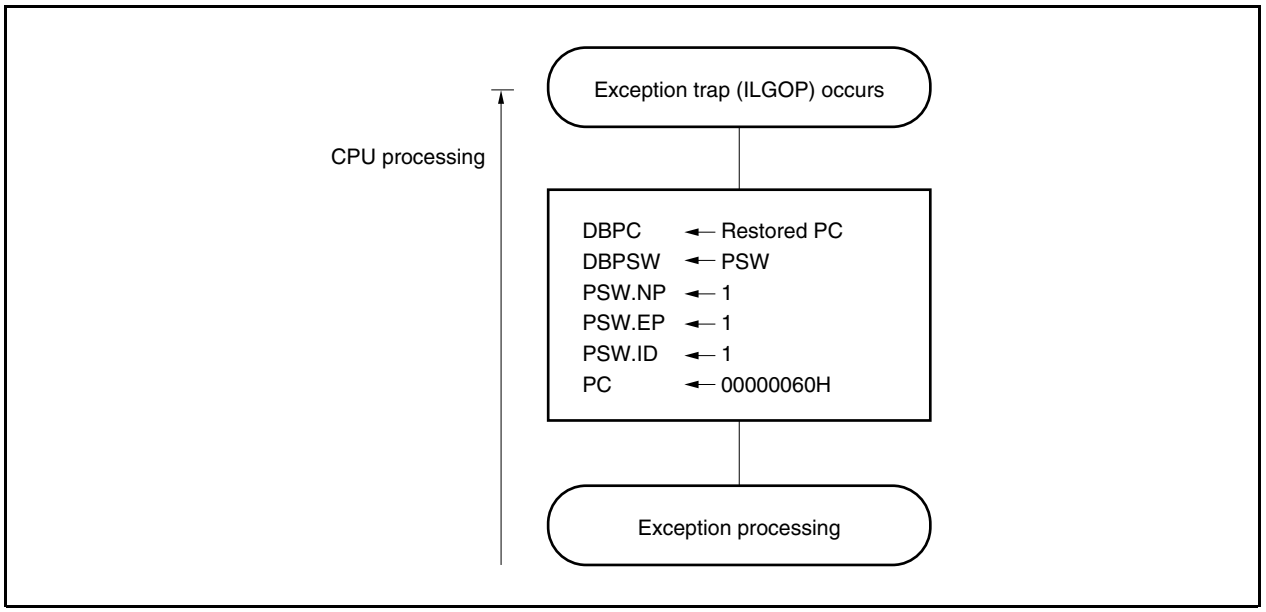
(1) Operation

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

<1> Saves the restored PC to DBPC.
<2> Saves the current PSW to DBPSW.
<3> Sets the NP, EP, and ID bits of the PSW.
<4> Sets the handler address (00000060H) corresponding to the exception trap to the PC, and transfers control.

Figure 22-10 illustrates the processing of the exception trap.

Figure 22-10. Exception Trap Processing



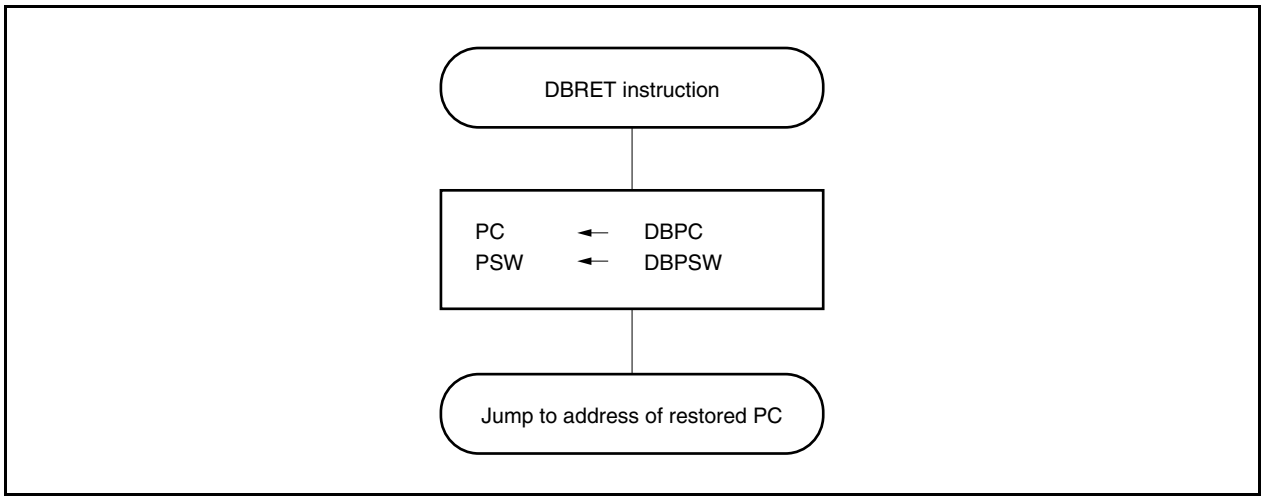
(2) Restore

Recovery from an exception trap is carried out by the DBRET instruction. By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.
<2> Transfers control to the address indicated by the restored PC and PSW.

Figure 22-11 illustrates the restore processing from an exception trap.

Figure 22-11. Restore Processing from Exception Trap



22.5.2 Debug trap

A debug trap is an exception that is generated when the DBTRAP instruction is executed and is always acknowledged.

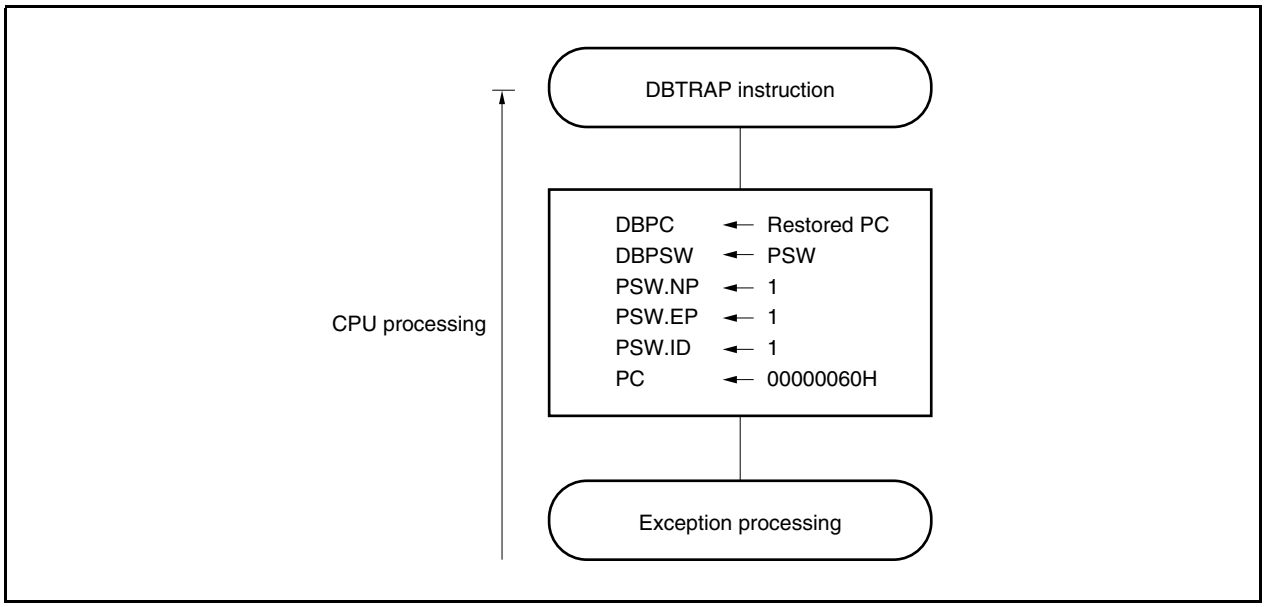
Upon occurrence of a debug trap, the CPU performs the following processing.

(1) Operation

<1> Saves restored PC to DBPC.
<2> Saves current PSW to DBPSW.
<3> Sets the NP, EP, and ID bits of PSW.
<4> Sets handler address (00000060H) for debug trap to PC and transfers control.

Figure 22-12 shows the debug trap processing format.

Figure 22-12. Debug Trap Processing Format



(2) Restoration

Restoration from a debug trap is executed with the DBRET instruction.

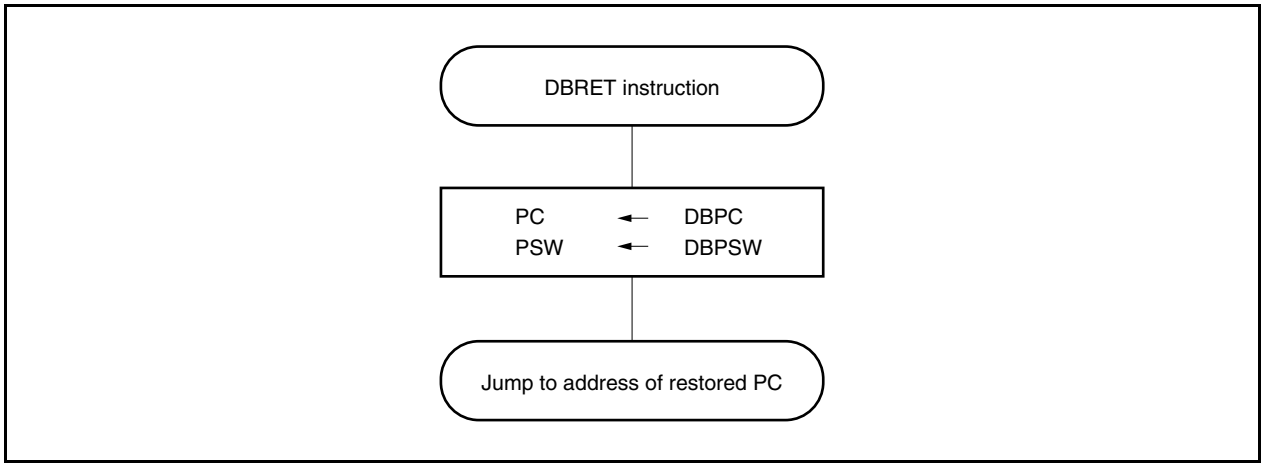
With the DBRET instruction, the CPU performs the following steps and transfers control to the address of the restored PC.

<1> The restored PC and PSW are read from DBPC and DBPSW.

<2> Control is transferred to the fetched address of the restored PC and PSW.

Table 22-13 shows the processing format for restoration from a debug trap.

Figure 22-13. Processing Format of Restoration from Debug Trap

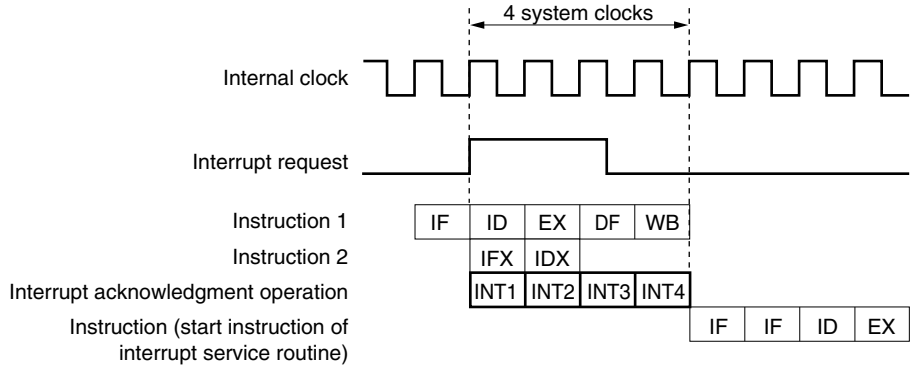


22.6 Interrupt Acknowledge Time of CPU

Except the following cases, the interrupt acknowledge time of the CPU is 5 clocks minimum. To input interrupt request signals successively, input the next interrupt request signal at least 5 clocks after the preceding interrupt.

- In software/hardware STOP mode
- When the external bus is accessed
- When interrupt request non-sampling instructions are successively executed (see **22.7 Periods in Which Interrupts Are Not Acknowledged by CPU**.)
- When the interrupt control register is accessed

Figure 22-14. Pipeline Operation at Interrupt Request Signal Acknowledgment (Outline)



Remark INT1 to INT4: Interrupt acknowledgment processing
IFX: Invalid instruction fetch
IDX: Invalid instruction decode

Interrupt acknowledge time (internal system clock)			Condition
	Internal interrupt	External interrupt	
Minimum	4	4 + Analog delay time	The following cases are exceptions. • In IDLE/software STOP mode • External bus access • Two or more interrupt request non-sample instructions are executed in succession • Access to peripheral I/O register
Maximum	6	6 + Analog delay time	

22.7 Periods in Which Interrupts Are Not Acknowledged by CPU

An interrupt is acknowledged by the CPU while an instruction is being executed. However, no interrupt will be acknowledged between an interrupt request non-sample instruction and the next instruction (interrupt is held pending).

The interrupt request non-sample instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- The store instruction for the command register (PRCMD)
- The load, store, or bit manipulation instructions for the following interrupt-related registers.
  Interrupt control register (xxICn), interrupt mask registers 0 to 3 (IMR0 to IMR3), in-service priority register (ISPR)

Remark xx: Identification name of each peripheral unit (see **Table 22-3 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 22-3 Interrupt Control Register (xxICn)**).

CHAPTER 23 KEY INTERRUPT FUNCTION

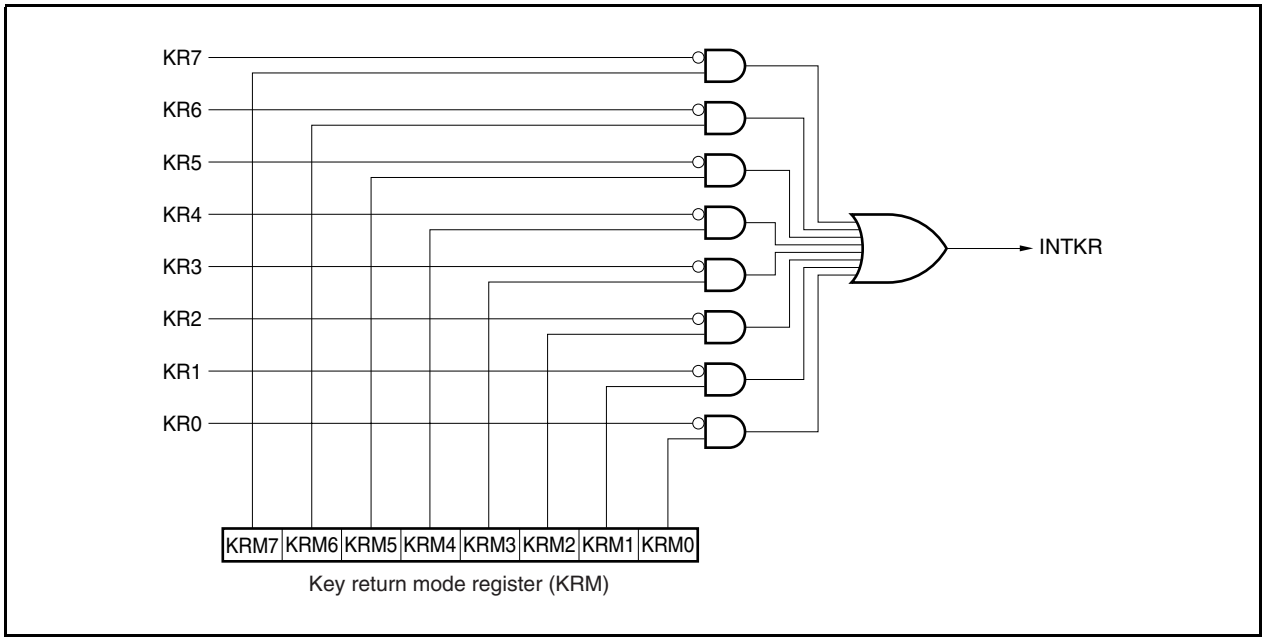
23.1 Function

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the eight key input pins (KR0 to KR7) by setting the key return mode register (KRM).

Table 23-1. Assignment of Key Return Detection Pins

Flag	Pin Description
KRM0	Controls KR0 signal in 1-bit units
KRM1	Controls KR1 signal in 1-bit units
KRM2	Controls KR2 signal in 1-bit units
KRM3	Controls KR3 signal in 1-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 23-1. Key Return Block Diagram



23.2 Control Register

(1) Key return mode register (KRM)

The KRM register controls the KRM0 to KRM7 bits using the KR0 to KR7 signals.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF300H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key return mode
0	Does not detect key return signal
1	Detects key return signal

Caution If the KRM register is changed, an interrupt request flag may be set. To prevent this, change the KRM register after disabling interrupts, and then enable interrupts after clearing the interrupt request flag.

CHAPTER 24 STANDBY FUNCTION

24.1 Overview

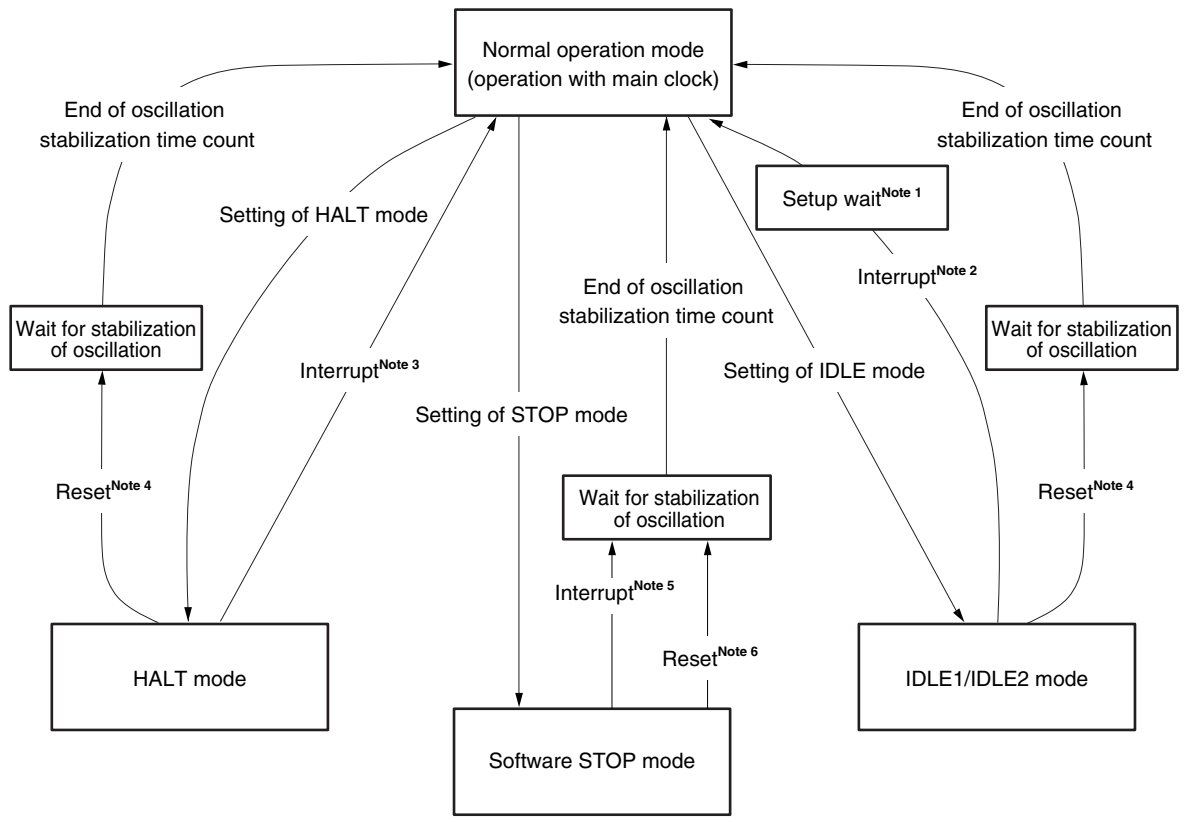
The power consumption of the system can be effectively reduced by using the standby modes in combination and selecting the appropriate mode for the application. The available standby modes are listed in Table 24-1.

Table 24-1. Standby Modes

Mode	Functional Outline
HALT mode	Mode to stop only the operating clock of the CPU
IDLE1 mode	Mode to stop all the internal operations of the chip except the oscillator, PLL operation[Note], and flash memory programming mode
IDLE2 mode	Mode to stop all the internal operations of the chip except the oscillator
Software STOP mode	Mode to stop all the internal operations of the chip except the subclock oscillator
Subclock operation mode	Mode to use the subclock as the internal system clock
Sub-IDLE mode	Mode to stop all the internal operations of the chip except the oscillator, PLL operation[Note], and flash memory programming mode, in the subclock operation mode

Note The PLL holds the previous operation status (in clock-through mode or PLL mode).

Figure 24-1. Status Transition



Notes 1. IDLE2 mode only

2. Non-maskable interrupt request signal (NMI, INTWDT2), unmasked external interrupt request signal, or unmasked internal maskable interrupt request signal operable in IDLE1/IDLE2 mode.

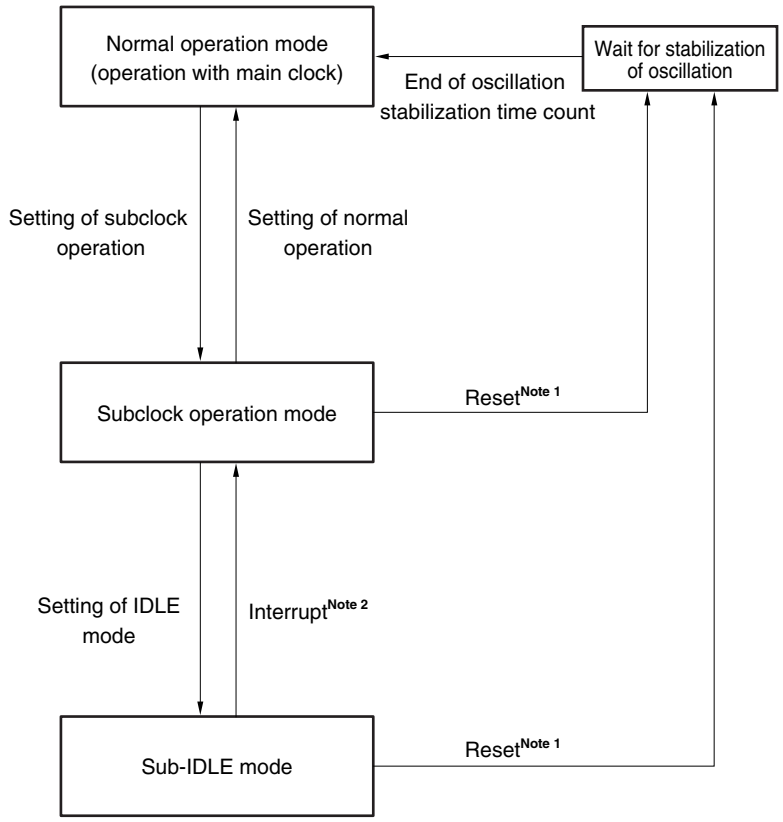
3. Non-maskable interrupt request signal, unmasked external interrupt request signal, or unmasked internal maskable interrupt request signal.

4. Reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM).

5. Non-maskable interrupt request signal (NMI, INTWDT2), unmasked external interrupt request signal, or unmasked internal maskable interrupt request signal operable in software STOP mode.

6. Reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, or low-voltage detector (LVI).

Figure 24-2. Status Transition (During Subclock Operation)



Notes 1. Reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM).

2. Non-maskable interrupt request signal (NMI, INTWDT2), unmasked external interrupt request signal, or unmasked internal maskable interrupt request signal operable in sub-IDLE mode.

24.2 HALT Mode

24.2.1 Setting and operation status

The HALT mode is set when a dedicated instruction (HALT) is executed in the normal operation mode.

In the HALT mode, the clock oscillator continues operating. Only clock supply to the CPU is stopped; clock supply to the other on-chip peripheral functions continues.

As a result, program execution is stopped, and the internal RAM retains the contents before the HALT mode was set. The on-chip peripheral functions that are independent of instruction processing by the CPU continue operating.

Table 24-3 shows the operation status in the HALT mode.

The average power consumption of the system can be reduced by using the HALT mode in combination with the normal operation mode for intermittent operation.

Cautions 1. Insert five or more NOP instructions after the HALT instruction.

2. When the HALT instruction is executed while an interrupt request signal is being held pending, the status shifts to HALT mode, but the HALT mode is then released immediately by the pending interrupt request.

24.2.2 Releasing HALT mode

The HALT mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from a peripheral function operable in the HALT mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).

After the HALT mode has been released, the normal operation mode is restored.

(1) Releasing HALT mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The HALT mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the HALT mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the HALT mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the HALT mode is released and that interrupt request signal is acknowledged.

Table 24-2. Operation After Releasing HALT Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed.	The next instruction is executed.

(2) Releasing HALT mode by reset input

The same operation as the normal reset operation is performed.

Table 24-3. Operation Status in HALT Mode

Setting of HALT Mode / Item		Operation Status	
		When Subclock Is Not Used	When Subclock Is Used
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Ring-OSC generator		Oscillation enabled	
PLL		Operable	
CPU		Stops operation	
DMA		Operable	
Interrupt controller		Operable	
ROM correction		Stops operation	
Timer P (TMP0 to TMP5)		Stops operation	
Timer Q (TMP0)		Stops operation	
Timer M (TMM0)		Operable when a clock other than f_{XT} is selected as the count clock	Operable
Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
Watchdog timer 2		Operable when a clock other than f_{XT} is selected as the count clock	Operable
Serial interface	CSIB0 to CSIB4	Operable	
	I^2C00 to I^2C02	Operable	
	UARTA0 to UARTA2	Operable	
CAN controller		Operable	
IEBus controller		Operable	
A/D converter		Operable	
D/A converter		Operable	
Real-time output function (RTO)		Operable	
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Operable (in the status in which data is not input to CRCIN to stop the CPU)	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION**.	
Port function		Retains status before HALT mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the HALT mode was set.	

24.3 IDLE1 Mode

24.3.1 Setting and operation status

The IDLE1 mode is set by clearing the PSM1 and PSM0 bits of the PSMR register to 00 and setting the STP bit of the PSC register to 1 in the normal operation mode.

In the IDLE1 mode, the clock oscillator, PLL operation, and flash memory continue operating but clock supply to the CPU and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE1 mode was set are retained. The CPU and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 24-5 shows the operation status in the IDLE1 mode.

The IDLE1 mode can reduce the current consumption more than the HALT mode because it stops the operation of the on-chip peripheral functions. The main clock oscillator does not stop, so the normal operation mode can be restored without waiting for the oscillation stabilization time after the IDLE1 mode has been released, in the same manner as when the HALT mode is released.

Caution Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE1 mode.

24.3.2 Releasing IDLE1 mode

The IDLE1 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from a peripheral function operable in the IDLE1 mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)).

After the IDLE1 mode has been released, the normal operation mode is restored.

(1) Releasing IDLE1 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The IDLE1 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the IDLE1 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

Caution An interrupt request signal that is disabled by setting the NMI1M, NMI0M, and INTM bits of the PSC register to 1 becomes invalid and IDLE1 mode is not released.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the IDLE1 mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE1 mode is released and that interrupt request signal is acknowledged.

Table 24-4. Operation After Releasing IDLE1 Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address after securing the prescribed setup time.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed after securing the prescribed setup time.	The next instruction is executed after securing the prescribed setup time.

(2) Releasing IDLE1 mode by reset input

The same operation as the normal reset operation is performed.

Table 24-5. Operation Status in IDLE1 Mode

Item \ Setting of IDLE1 Mode		Operation Status	
		When Subclock Is Not Used	When Subclock Is Used
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Ring-OSC generator		Oscillation enabled	
PLL		Operable	
CPU		Stops operation	
DMA		Stops operation	
Interrupt controller		Stops operation (but standby mode release enabled)	
ROM correction		Stops operation	
Timer P (TMP0 to TMP5)		Stops operation	
Timer Q (TMQ0)		Stops operation	
Timer M (TMM0)		Operable when $f_R/8$ is selected as the count clock	Operable when $f_R/8$ or f_{XT} is selected as the count clock
Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
Serial interface	CSIB0 to CSIB4	Operable when $\overline{\text{SCKBn}}$ input clock is selected as operation clock (n = 0 to 4)	
	I^2C00 to I^2C02	Stops operation	
	UARTA0 to UARTA2	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
CAN controller		Stops operation	
IEBus controller		Stops operation	
A/D converter		Stops operation	
D/A converter		Stops operation	
Real-time output function (RTO)		Stops operation	
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Stops operation	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION**.	
Port function		Retains status before IDLE1 mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE1 mode was set.	

24.4 IDLE2 Mode

24.4.1 Setting and operation status

The IDLE2 mode is set by setting the PSM1 and PSM0 bits of the PSMR register to 10 and setting the STP bit of the PSC register to 1 in the normal operation mode.

In the IDLE2 mode, the clock oscillator continues operation but clock supply to the CPU, PLL operation, flash memory, and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE2 mode was set are retained. The CPU and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 24-7 shows the operation status in the IDLE2 mode.

The IDLE2 mode can reduce the current consumption more than the IDLE1 mode because it stops the operations of the on-chip peripheral functions and flash memory. However, because the PLL operation and flash memory are stopped, a setup time for the PLL operation and flash memory is required when IDLE2 mode is released.

Caution Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE2 mode.

24.4.2 Releasing IDLE2 mode

The IDLE2 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the IDLE2 mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)). The PLL returns to the operation status before the IDLE2 mode was set.

After the IDLE2 mode has been released, the normal operation mode is restored.

(1) Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The IDLE2 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the IDLE2 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

Caution The interrupt request signal that is disabled by setting the NMI1M, NMI0M, and INTM bits of the PSC register to 1 becomes invalid and IDLE2 mode is not released.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the IDLE2 mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE2 mode is released and that interrupt request signal is acknowledged.

Table 24-6. Operation After Releasing IDLE2 Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address after securing the prescribed setup time.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed after securing the prescribed setup time.	The next instruction is executed after securing the prescribed setup time.

(2) Releasing IDLE2 mode by reset input

The same operation as the normal reset operation is performed.

Table 24-7. Operation Status in IDLE2 Mode

Item \ Setting of IDLE2 Mode		Operation Status	
		When Subclock Is Not Used	When Subclock Is Used
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Ring-OSC generator		Oscillation enabled	
PLL		Stops operation (recommended to stopping PLL operation because of reducing consumption current)	
CPU		Stops operation	
DMA		Stops operation	
Interrupt controller		Stops operation	
ROM correction		Stops operation	
Timer P (TMP0 to TMP5)		Stops operation	
Timer Q (TMP0)		Stops operation	
Timer M (TMM0)		Operable when f_R/8 is selected as the count clock	Operable when f_R/8 or f_{XT} is selected as the count clock
Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
Serial interface	CSIB0 to CSIB4	Operable when $\overline{SCKBn}$ input clock is selected as operation clock (n = 0 to 4)	
	I^2C00 to I^2C02	Stops operation	
	UARTA0 to UARTA2	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
CAN controller		Stops operation	
IEBus controller		Stops operation	
A/D converter		Operable when f_{BRG} is selected as operation clock	
D/A converter		Stops operation	
Real-time output function (RTO)		Stops operation	
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Stops operation	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION**.	
Port function		Retains status before IDLE2 mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE2 mode was set.	

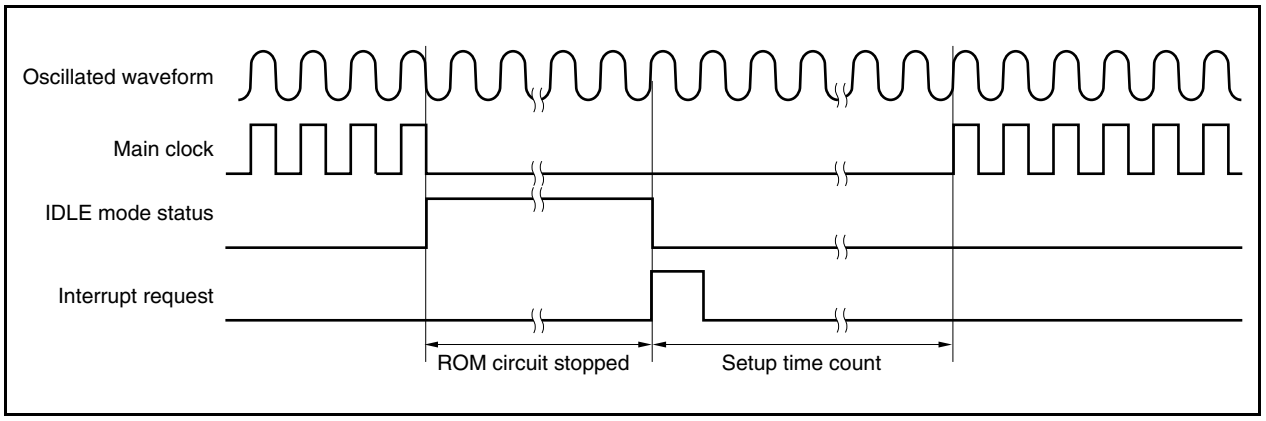
24.4.3 Securing setup time when releasing IDLE2 mode

Secure the setup time for the ROM (flash memory) after releasing the IDLE2 mode because the operation of the blocks other than the main clock oscillator stops after IDLE2 mode is set.

(1) Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

Secure the specified setup time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting. When it overflows, the normal operation mode is restored.



(2) Release by reset input ($\overline{\text{RESET}}$ pin input, WDT2RES generation)

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, 2^{16}/fx.

24.5 Software STOP Mode

24.5.1 Setting and operation status

The software STOP mode is set by setting the PSM1 and PSM0 bits of the PSMR register to 01 and setting the STP bit of the PSC register to 1 in the normal operation mode.

In the software STOP mode, the subclock oscillator continues operation but the main clock oscillator stops. Clock supply to the CPU and the on-chip peripheral functions is stopped.

As a result, program execution stops, and the contents of the internal RAM before the software STOP mode was set are retained. The on-chip peripheral functions that operate with the clock oscillated by the subclock oscillator or an external clock continue operating.

Table 24-9 shows the operation status in the software STOP mode.

Because the software STOP stops operation of the main clock oscillator, it reduces the current consumption to a level lower than the IDLE2 mode. If the subclock oscillator, Ring-OSC, and external clock are not used, the power consumption can be minimized with only leakage current flowing.

Caution Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the software STOP mode.

24.5.2 Releasing software STOP mode

The software STOP mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the software STOP mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, or low-voltage detector (LVI)).

After the software STOP mode has been released, the normal operation mode is restored after the oscillation stabilization time has been secured.

Caution The interrupt request that is disabled by setting the NMI1M, NMI0M, and INTM bits of the PSC register to 1 becomes invalid and software STOP mode is not released.

(1) Releasing software STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The software STOP mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the software STOP mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the software STOP mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the software STOP mode is released and that interrupt request signal is acknowledged.

Table 24-8. Operation After Releasing Software STOP Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address after securing the oscillation stabilization time.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed after securing the oscillation stabilization time.	The next instruction is executed after securing the oscillation stabilization time.

(2) Releasing software STOP mode by reset input

The same operation as the normal reset operation is performed.

Table 24-9. Operation Status in Software STOP Mode

Setting of Software STOP Mode / Item		Operation Status	
		When Subclock Is Not Used	When Subclock Is Used
Main clock oscillator		Stops oscillation	
Subclock oscillator		–	Oscillation enabled
Ring-OSC generator		Oscillation enabled	
PLL		Stops operation	
CPU		Stops operation	
DMA		Stops operation	
Interrupt controller		Stops operation	
ROM correction		Stops operation	
Timer P (TMP0 to TMP5)		Stops operation	
Timer Q (TMP0)		Stops operation	
Timer M (TMM0)		Operable when f_R/8 is selected as the count clock	Operable when f_R/8 or f_{XT} is selected as the count clock
Watch timer		Stops operation	Operable when f_{XT} is selected as the count clock
Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
Serial interface	CSIB0 to CSIB4	Operable when $\overline{SCKBn}$ input clock is selected as operation clock (n = 0 to 4)	
	I^2C00 to I^2C02	Stops operation	
	UARTA0 to UARTA2	Stops operation (but UART0 is operable when the ASCKA0 input clock is selected)	
CAN controller		Stops operation	
IEBus controller		Stops operation	
A/D converter		Stops operation	
D/A converter		Stops operation	
Real-time output function (RTO)		Stops operation	
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Stops operation	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION**.	
Port function		Retains status before software STOP mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the software STOP mode was set.	

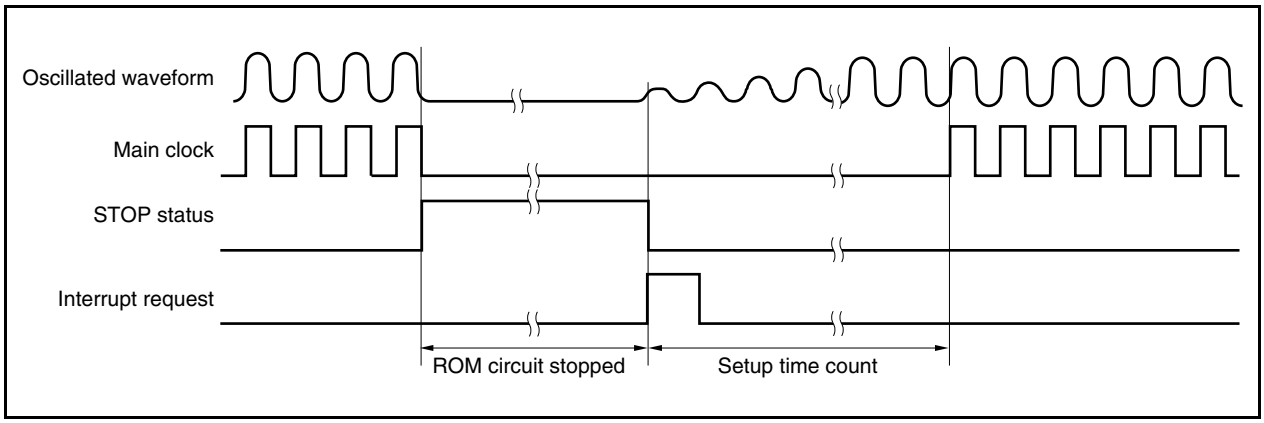
24.5.3 Securing oscillation stabilization time when releasing software STOP mode

Secure the oscillation stabilization time for main clock oscillator after releasing the software STOP mode because the operation of the blocks other than the main clock oscillator stops after software STOP mode is set.

(1) Releasing software STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

Secure the oscillation stabilization time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting. When it overflows, the normal operation mode is restored.



(2) Release by reset

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, 2^{16}/fx.

24.7 Subclock Operation Mode

24.7.1 Setting and operation status

The subclock operation mode is set by setting the CK3 bit of the PCC register to 1 in the normal operation mode.

When the subclock operation mode is set, the internal system clock is changed from the main clock to the subclock. Check whether the clock has been switched by using the CLS bit of the PCC register.

When the MCK bit of the PCC register is set to 1, the operation of the main clock oscillator is stopped. As a result, the system operates only with the subclock. However, watchdog timer 2 stops counting when subclock operation is started (CLS bit of PCC register = 1). (Watchdog timer 2 retains the value before the subclock operation mode was set.)

In the subclock operation mode, the current consumption can be reduced to a level lower than in the normal operation mode because the subclock is used as the internal system clock. In addition, the current consumption can be further reduced to the level of the software STOP mode by stopping the operation of the main system clock oscillator.

Table 24-10 shows the operation status in subclock operation mode.

Caution When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits of the PCC register (using a bit manipulation instruction to manipulate the bit is recommended). For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).

24.7.2 Releasing subclock operation mode

The subclock operation mode is released by reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)) when the CK3 bit is cleared to 0.

If the main clock is stopped (MCK bit = 1), set the MCK bit to 1, secure the oscillation stabilization time of the main clock by software, and clear the CK3 bit to 0.

The normal operation mode is restored when the subclock operation mode is released.

Caution When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended).
For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).

Table 24-10. Operation Status in Subclock Operation Mode

Setting of Subclock Operation Mode / Item		Operation Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
Subclock oscillator		Oscillation enabled	
Ring-OSC generator		Oscillation enabled	
PLL		Operable	Stops operation[Note]
CPU		Operable	
DMA		Operable	
Interrupt controller		Operable	
ROM correction		Operable	
Timer P (TMP0 to TMP5)		Operable	Stops operation
Timer Q (TMP0)		Operable	Stops operation
Timer M (TMM0)		Operable	Operable when $f_R/8$ or f_{XT} is selected as the count clock
Watch timer		Operable	Operable when f_{XT} is selected as the count clock
Watchdog timer 2		Operable	Operable when f_R or f_{XT} is selected as the count clock
Serial interface	CSIB0 to CSIB4	Operable	Operable when $\overline{\text{SCKBn}}$ input clock is selected as operation clock (n = 0 to 4)
	I^2C00 to I^2C02	Operable	Stops operation
	UARTA0 to UARTA2	Operable	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)
CAN controller		Operable	Stops operation
IEBus controller		Operable	Stops operation
A/D converter		Operable	Stops operation
D/A converter		Operable	Stops operation
Real-time output function (RTO)		Operable	Stops operation
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Operable	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION**.	
Port function		Settable	
Internal data		Settable	

Note Be sure to stop the PLL (PLLON bit of PLLCTL register = 0) before stopping the main clock.

24.8 Sub-IDLE Mode

24.8.1 Setting and operation status

The sub-IDLE mode is set by setting the PSM1 and PSM0 bits of the PSMR register to 10 and setting the STP bit of the PSC register to 1 in the subclock operation mode.

In this mode, the clock oscillator continues operation but clock supply to the CPU, flash memory, and the other on-chip peripheral functions is stopped.

As a result, program execution stops and the contents of the internal RAM before the sub-IDLE mode was set are retained. The CPU and the other on-chip peripheral functions are stopped. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Because the sub-IDLE mode stops operation of the CPU, flash memory, and other on-chip peripheral functions, it can reduce the current consumption more than the subclock operation mode. If the sub-IDLE mode is set after the main clock has been stopped, the current consumption can be reduced to a level as low as that in the software STOP mode.

Table 24-12 shows the operation status in the sub-IDLE mode.

24.8.2 Releasing sub-IDLE mode

The sub-IDLE mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the sub-IDLE mode, or reset signal (reset by $\overline{\text{RESET}}$ pin input, WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM)). The PLL returns to the operation status before the sub-IDLE mode was set.

When the sub-IDLE mode is released by an interrupt request signal, the subclock operation mode is set. If it is released by reset signal, the normal operation mode is restored.

(1) Releasing sub-IDLE mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The sub-IDLE mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.

If the sub-IDLE mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

Caution The interrupt request signal that is disabled by setting the NMI1M, NMI0M, and INTM bits of the PSC register to 1 becomes invalid and sub-IDLE mode is not released.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the sub-IDLE mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the sub-IDLE mode is released and that interrupt request signal is acknowledged.

Table 24-11. Operation After Releasing Sub-IDLE Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed.	The next instruction is executed.

(2) Releasing sub-IDLE mode by reset input

The same operation as the normal reset operation is performed.

Table 24-12. Operation Status in Sub-IDLE Mode

Item \ Setting of Sub-IDLE Mode		Operation Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
Subclock oscillator		Oscillation enabled	
Ring-OSC generator		Oscillation enabled	
PLL		Operable	Stops operation[Note]
CPU		Stops operation	
DMA		Stops operation	
Interrupt controller		Stops operation	
ROM correction		Stops operation	
Timer P (TMP0 to TMP5)		Stops operation	
Timer Q (TMP0)		Stops operation	
Timer M (TMM0)		Operable when $f_R/8$ or f_{XT} is selected as the count clock	
Watch timer		Stops operation	Operable when f_{XT} is selected as the count clock
Watchdog timer 2		Operable when f_R or f_{XT} is selected as the count clock	
Serial interface	CSIB0 to CSIB4	Operable when $\overline{SCKBn}$ input clock is selected as operation clock (n = 0 to 4)	
	I^2C00 to I^2C02	Stops operation	
	UARTA0 to UARTA2	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
CAN controller		Stops operation	
IEBus controller		Stops operation	
A/D converter		Stops operation	
D/A converter		Stops operation	
Real-time output function (RTO)		Stops operation	
Key interrupt function (KR)		Operable	
CRC arithmetic circuit		Stops operation	
External bus interface		See **CHAPTER 5 BUS CONTROL FUNCTION** (same operation status as IDLE1, IDLE2 mode).	
Port function		Retains status before sub-IDLE mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the sub-IDLE mode was set.	

Note Be sure to stop the PLL (PLLON bit of PLLCTL register = 0) before stopping the main clock.

24.9 Control Registers

(1) Power save control register (PSC)

The PSC register is an 8-bit register that controls the standby function. The STP bit of this register is used to specify the software STOP mode. This register is a special register (see **3.4.9 Special registers**). Data can be written to this register only in a specific sequence so that its contents are not rewritten by mistake due to a program hang-up.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF1FEH

	7	<6>	<5>	<4>	3	2	<1>	0
PSC	0	NMI1M	NMI0M	INTM	0	0	STP	0

NMI1M	Standby mode release control upon occurrence of INTWDT2 signal
0	Standby mode release by INTWDT2 signal enabled
1	Standby mode release by INTWDT2 signal disabled

NMI0M	Standby mode release control by NMI pin input
0	Standby mode release by NMI pin input enabled
1	Standby mode release by NMI pin input disabled

INTM	Standby mode release control via maskable interrupt request signal
0	Standby mode release by maskable interrupt request signal enabled
1	Standby mode release by maskable interrupt request signal disabled

STP	Standby mode[Note] setting
0	Normal mode
1	Standby mode

Note Standby mode by STP bit: IDLE1 mode, IDLE2 mode, software STOP mode, sub-IDLE mode

Caution Before setting IDLE1, IDLE2, software STOP, or sub IDLE mode, set the PSM1 and PSM0 bits of the PSMR register.

CHAPTER 25 RESET FUNCTIONS

25.1 Overview

The following reset functions are available.

- Reset by $\overline{\text{RESET}}$ pin input
- Reset by watchdog timer 2 overflow (WDT2RES)
- System reset by low-voltage detector (LVI)
- System reset by clock monitor (CLM)

25.2 Registers to Check Reset Source

(1) Reset source flag register (RESF)

The RESF register indicates from which source a reset signal is generated.

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

$\overline{\text{RESET}}$ input clears this register to 00H. The default value differs if the source of reset is other than $\overline{\text{RESET}}$.

After reset: 00H R/W Address: FFFFF888H

	7	6	5	4	3	2	1	0
RESF	0	0	0	WDT2RF	0	0	CLMRF	LVIRF

WDT2RF	Reset signal from WDT2
0	Not generated
1	Generated

CLMRF	Reset signal from CLM
0	Not generated
1	Generated

LVIRF	Reset signal from LVI
0	Not generated
1	Generated

Note The value of this register is cleared to 00H when a reset is executed via the $\overline{\text{RESET}}$ pin. When a reset is executed by the WDT2RES signal, low-voltage detector (LVI), or clock monitor (CLM), the reset flags of this register (WDT2RF bit, CLMRF bit, and LVIRF bit) are set (with the other sources retained).

Caution Only "0" can be written to each bit of this register. If writing "0" conflicts with setting the flag (occurrence of reset), setting the flag takes precedence.

25.3 Operation

25.3.1 Reset operation via $\overline{\text{RESET}}$ pin

When a low level is input to the $\overline{\text{RESET}}$ pin, the system is reset, and each hardware unit is initialized.

When the level of the $\overline{\text{RESET}}$ pin is changed from low to high, the reset status is released.

If the reset status is released by $\overline{\text{RESET}}$ pin input, the oscillation stabilization time elapses (reset value of OSTS register: $2^{16}/f_X$) and then the CPU starts program execution.

Table 25-1. Hardware Status on $\overline{\text{RESET}}$ Pin Input

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Ring-OSC generator	Oscillation stops	Oscillation starts
Peripheral clock (f_X to f_X/1,024)	Operation stops	Operation starts after securing of oscillation stabilization time
Internal system clock (f_{XX}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing of oscillation stabilization time (initialized to f_{XX}/8)
CPU	Initialized	Program execution starts after securing of oscillation stabilization time
Watchdog timer 2	Operation stops	Operation starts
Internal RAM	Undefined if power-on reset or writing data to RAM (by CPU) and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained[Note].	
I/O lines (ports/alternate-function pins)	High impedance	
On-chip peripheral I/O registers	Initialized to specified status, OCDM register is set (01H).	
Other on-chip peripheral functions	Operation stops	Operation can be started after securing oscillation stabilization time

Note The firmware of the V850ES/SG2 uses part of the internal RAM after the internal system reset operation has been released because it supports a boot swapping function. Therefore, the contents of some RAM areas are not retained on power-on reset.

Caution The V850ES/SG2 may enter on-chip debug mode (flash memory version only) after the reset status has been released, depending on the pin status. For details, see CHAPTER 4 PORT FUNCTIONS.

Figure 25-1. Timing of Reset Operation by $\overline{\text{RESET}}$ Pin Input

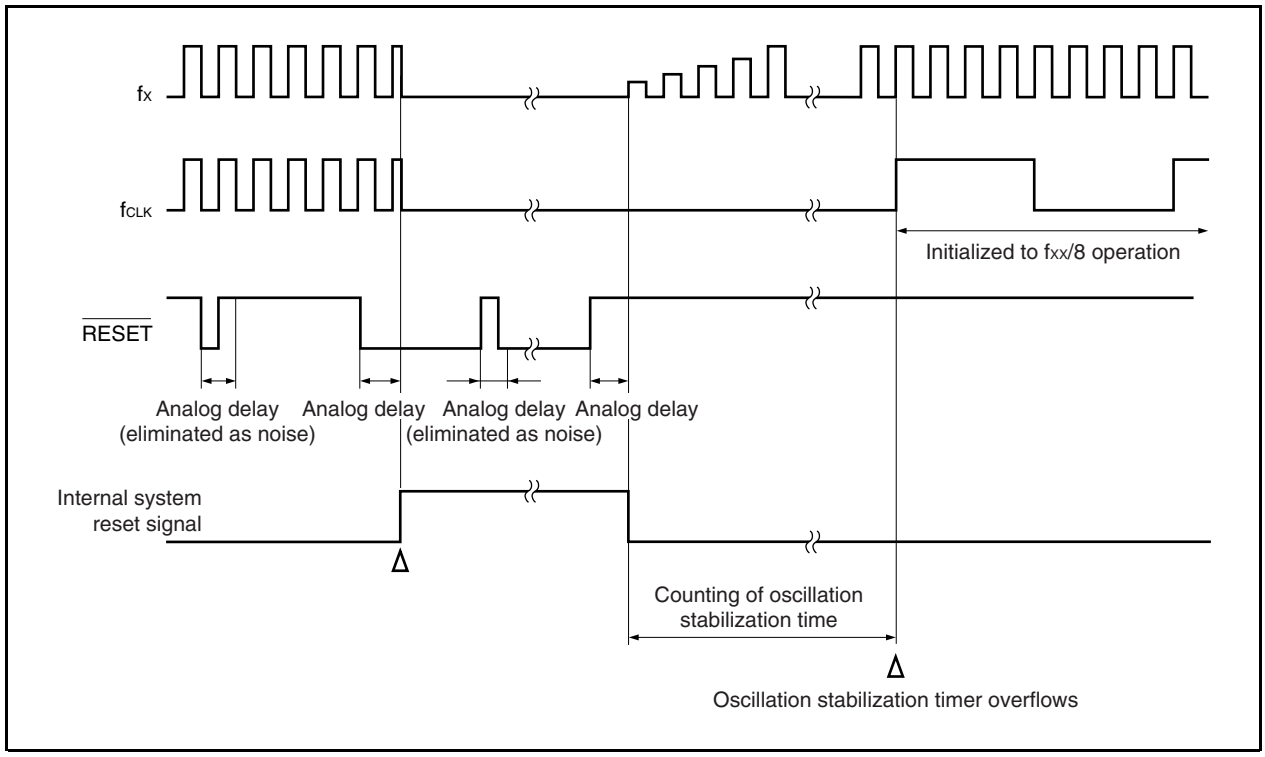
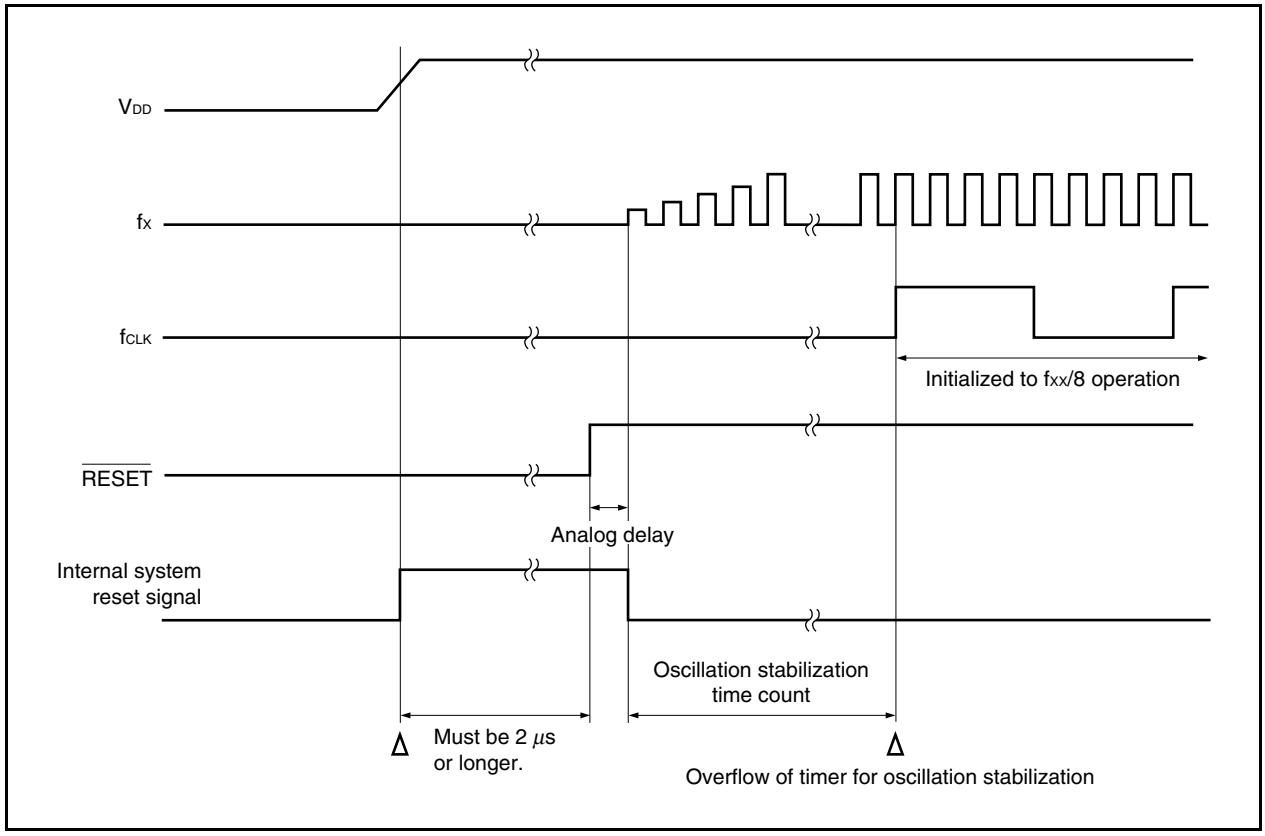


Figure 25-2. Timing of Power-on Reset Operation



25.3.2 Reset operation by WDT2RES signal

When watchdog timer 2 is set to the reset operation mode due to overflow, upon watchdog timer 2 overflow (WDT2RES signal generation), a system reset is executed and the hardware is initialized to the initial status.

Following watchdog timer 2 overflow, the reset status is entered and lasts the predetermined time (analog delay), and the reset status is then automatically released. Following reset release, the CPU starts program execution after securing the oscillation stabilization time (initial value of OSTS register: $2^{16}/f_X$) of the main clock oscillator.

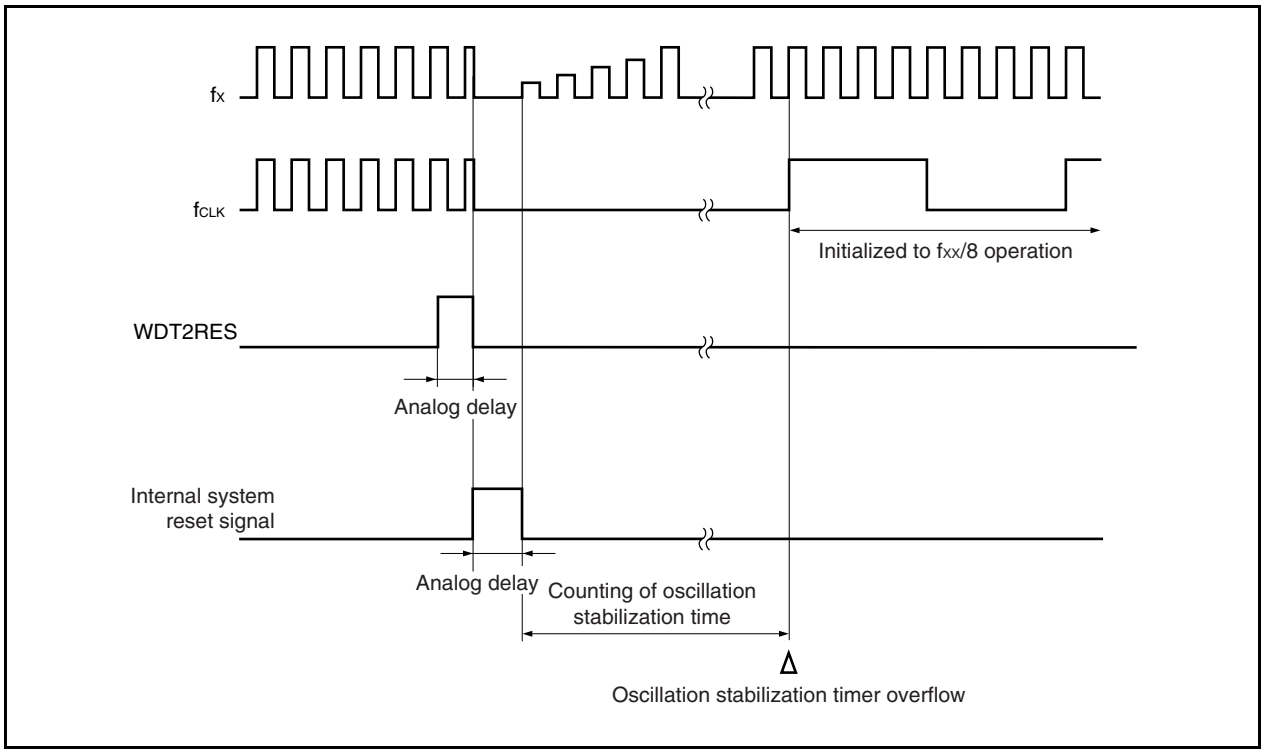
The main clock oscillator is stopped during the reset period.

Table 25-2. Hardware Statuses During WST2RES Signal Generation

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Ring-OSC generator	Oscillation stops	Oscillation starts
Peripheral clock (f_{XX} to f_{XX}/1,024)	Operation stops	Operation starts after securing oscillation stabilization time
Internal system clock (f_{XX}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
CPU	Initialized	Program execution after securing oscillation stabilization time
Watchdog timer 2	Operation stops	Operation starts after securing oscillation stabilization time
Internal RAM	Undefined if power-on reset or writing data to RAM (by CPU) and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained[Note].	
I/O lines (ports/alternate-function pins)	High impedance	
On-chip peripheral I/O register	Initialized to specified status, OCDM register retains its value.	
On-chip peripheral functions other than above	Operation stops	Operation can be started after securing oscillation stabilization time.

Note The firmware of the V850ES/SG2 uses part of the internal RAM after the internal system reset operation has been released because it supports a boot swapping function. Therefore, the contents of some RAM areas are not retained on power-on reset.

Figure 25-3. Timing of Reset Operation by WDT2RES Signal Generation



25.3.3 Reset operation by low-voltage detector

If the supply voltage falls below the voltage detected by the low-voltage detector when LVI operation is enabled, a system reset is executed (when the LVIMD bit of the LVIM register is set to 1), and the hardware is initialized to the initial status.

The reset status lasts from when a supply voltage drop has been detected until the supply voltage rises above the LVI detection voltage detector. Following reset release, the CPU starts program execution after securing the oscillation stabilization time (initial value of OSTS register: $2^{16}/f_X$) of the main clock oscillator.

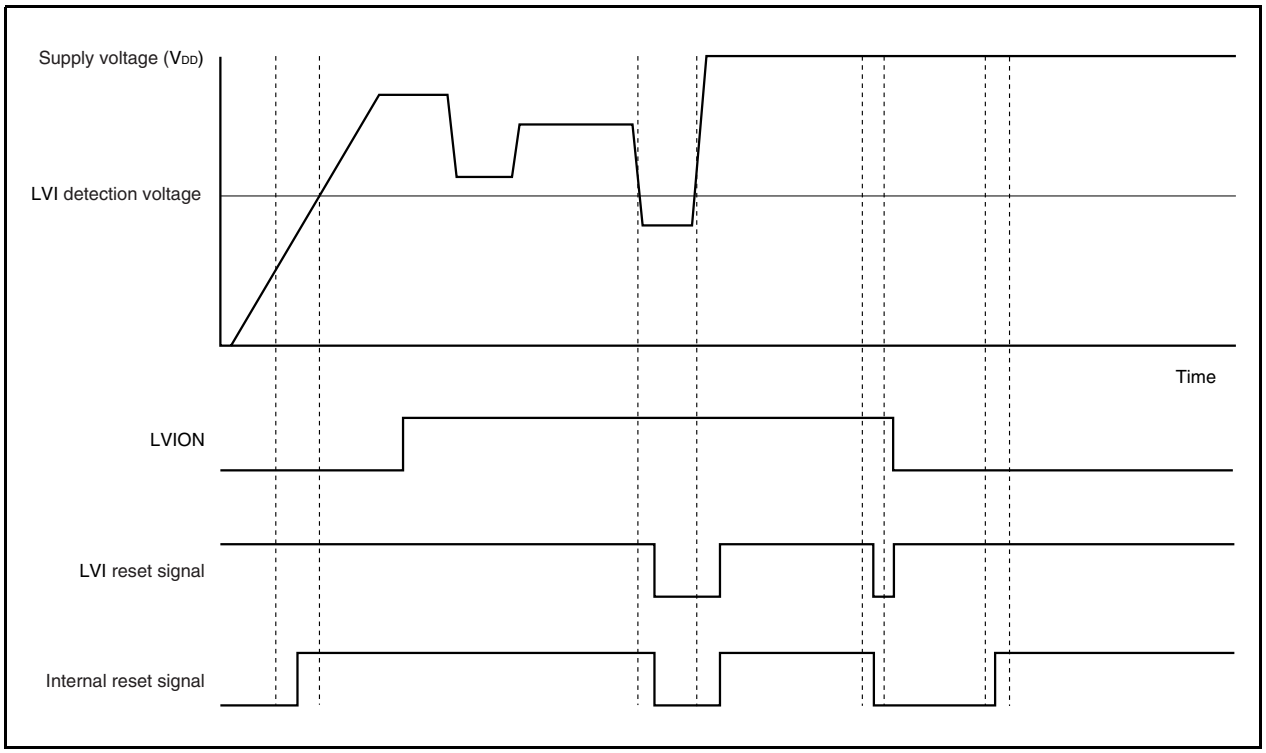
The main clock oscillator is stopped during the reset period.

Table 25-3. Hardware Statuses During Reset Operation by Low-Voltage Detector

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Ring-OSC generator	Oscillation stops	Oscillation starts
Peripheral clock (f_X to f_X/1,024)	Operation stops	Operation starts after securing oscillation stabilization time
Internal system clock (f_{XX}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
CPU	Initialized	Program execution starts after securing oscillation stabilization time
Watchdog timer 2	Operation stops	Operation starts
Internal RAM	Undefined if power-on reset or writing data to RAM (by CPU) and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained[Note].	
I/O lines (ports/alternate-function pins)	High impedance	
On-chip peripheral I/O register	Initialized to specified status, OCDM register retains its value.	
LVI	Operation stops	
On-chip peripheral functions other than above	Operation stops	Operation can be started after securing oscillation stabilization time.

Note The firmware of the V850ES/SG2 uses part of the internal RAM after the internal system reset operation has been released because it supports a boot swapping function. Therefore, the contents of some RAM areas are not retained on power-on reset.

Figure 25-4. Timing of Reset Operation by Low-Voltage Detector



(1) Low-voltage detection register (LVIM)

The LVIM register is used to enable or disable low-voltage detection and select the operation mode.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF890H

	<7>	6	5	4	3	2	<1>	<0>
LVIM	LVION	0	RREN	0	0	0	LVIMD	LVIF

LVION	Low-voltage detection operation enable/disable
0	Operation disable .
1	Operation enable .

LVIMD	Selection of operation mode of low-voltage detection
0	Generate interrupt request signal INTLVI when supply voltage < detection voltage.
1	Generate internal reset signal LVIRES when supply voltage < detection voltage.

LVIF	Low-voltage detection flag
0	When supply voltage > detection voltage or when operation is disabled
1	Supply voltage < detection voltage

Cautions 1. After setting the LVION bit to 1, wait for 0.1 ms (TYP.) (target value) before checking the voltage using the LVIF bit.

2. The value of the LVIF flag is output as the signal INTLVI when the LVION bit = 1 and LVIMD bit = 0.

(2) Low-voltage detection level select register (LVIS)

The LVIS register is used to select the low voltage level to be detected.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF891H

	7	6	5	4	3	2	1	0
LVIS	0	0	0	0	0	0	0	LVIS0

LVIS0	Detection level
0	3.0 V ±0.15 V
1	2.85 V ±0.15 V

(3) Internal RAM data status register (RAMS)

The RAMS register is used to make the data of the internal RAM valid or invalid.

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

Reset input sets this register to 01H.

After reset: 01H R/W Address: FFFFF892H

	7	6	5	4	3	2	1	<0>
RAMS	0	0	0	0	0	0	0	RAMF

RAMF	Internal RAM data valid/invalid
0	Valid
1	Invalid

25.3.4 Clock monitor

(1) Function of clock monitor

The clock monitor samples the main clock using the internal Ring-OSC and generates a reset request signal when oscillation of the main clock is stopped.

Once the operation of the clock monitor has been enabled by the operation enable flag, it can be stopped only by reset.

The clock monitor is automatically stopped under the following conditions.

- When the oscillation stabilization time is counted after the software STOP mode has been released
- When the main clock is stopped (MCK bit of PCC register = 1 when subclock operates and CLS bit of PCC register = 0 when main clock operates)
- When the sampling clock is stopped (Ring-OSC)
- When the CPU operates on Ring-OSC

Table 25-4. Operation Status of Clock Monitor (CLM.CLME Bit = 1, with Ring-OSC Operating)

CPU Operation Clock	Operation Mode	Main Clock Status	Ring-OSC Clock Status	Clock Monitor Status
Main clock	HALT mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	IDLE mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	STOP mode	Stops	Oscillates[Note 1]	Stops
Subclock (MCK bit of PCC register = 0)	HALT mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	IDLE mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	STOP mode	Stops	Oscillates[Note 1]	Stops
Subclock (MCK bit of PCC register = 1)	HALT mode	Stops	Oscillates[Note 1]	Stops
	IDLE mode	Stops	Oscillates[Note 1]	Stops
	STOP mode	Stops	Oscillates[Note 1]	Stops
Ring-OSC clock	–	Stops	Stops[Note 1]	Stops
During reset	–	Stops	Stops	Stops

Notes 1. Ring-OSC can be stopped by setting the RSTOP bit of the RCM register to 1.

2. The clock monitor is stopped when Ring-OSC is stopped.

(2) Clock monitor mode register (CLM)

The CLM register is used to select the operation mode of the clock monitor.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF870H

	7	6	5	4	3	2	1	<0>
CLM	0	0	0	0	0	0	0	CLME

CLME	Clock monitor operation enable disable
0	Operation of clock monitor disabled.
1	Operation of clock monitor enabled.

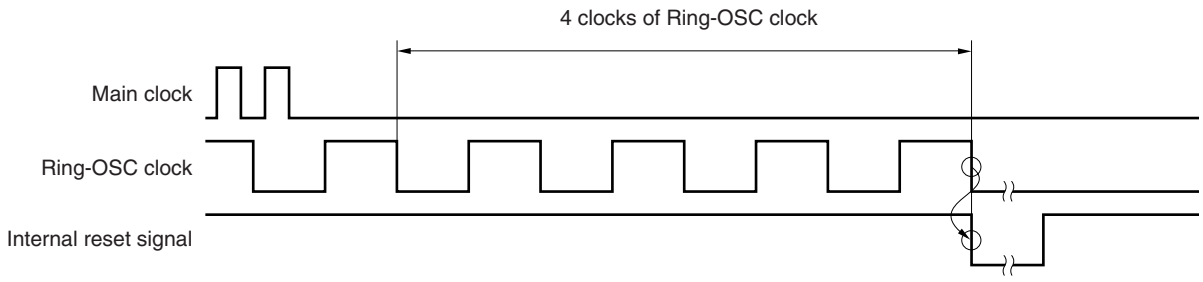
Caution Once the CLME bit has been set to 1, it can be cleared to 0 only by reset.

(3) Operation of clock monitor

(a) Operation when main clock oscillation is stopped (CLME bit = 1)

If oscillation of the main clock is stopped when the CLME bit = 1, an internal reset signal is generated at the timing shown in Figure 25-5.

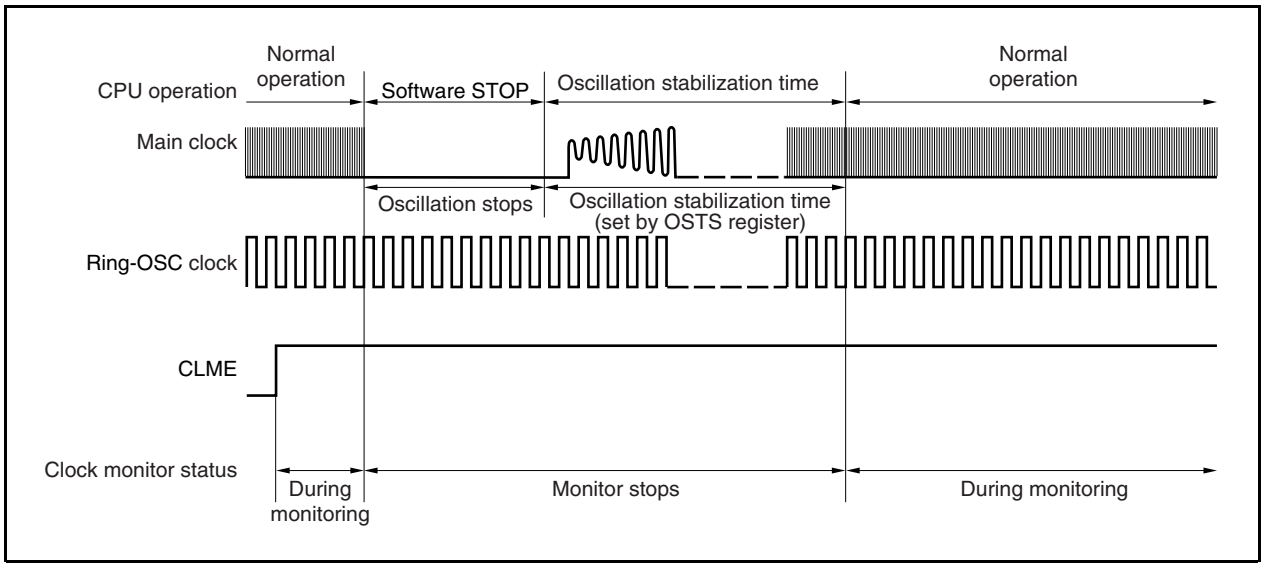
Figure 25-5. When Oscillation of Main Clock Is Stopped



(b) Operation in software STOP mode and after software STOP mode is released

If the software STOP mode is set when the CLME bit = 1, the monitor operation is stopped in the software STOP mode and while the oscillation stabilization time is being counted. The monitor operation is automatically started after the oscillation stabilization time has elapsed.

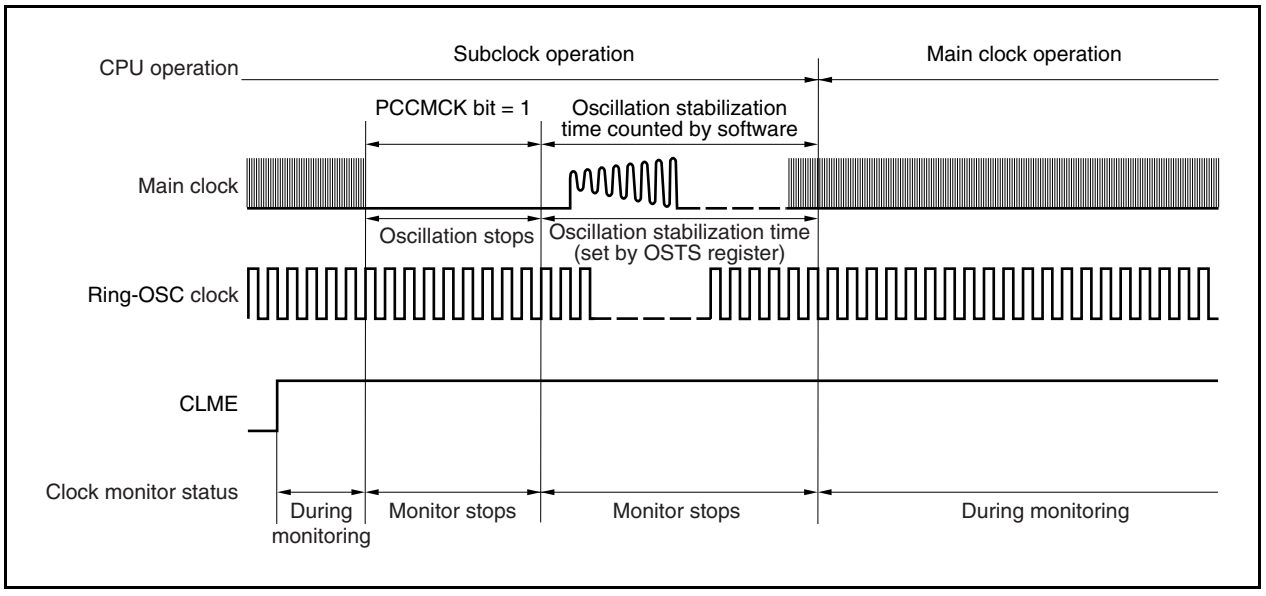
Figure 25-6. Operation in Software STOP Mode and After Software STOP Mode Is Released



(c) Operation when main clock is stopped

If the main clock is stopped by setting the MCK bit of the PCC register to 1 while the subclock is operating (CLS bit of PCC register = 1), the monitor operation is stopped until the main clock operates (CLS bit of PCC register = 0). The monitor operation is automatically started when the main clock starts operating.

Figure 25-7. Operation When Main Clock Is Stopped



(d) Operation when CPU operates on Ring-OSC clock (CCLSF bit of CCLS register = 1)

The monitor operation is not started even if the CLME bit is set to 1 when the CCLSF bit is 1.

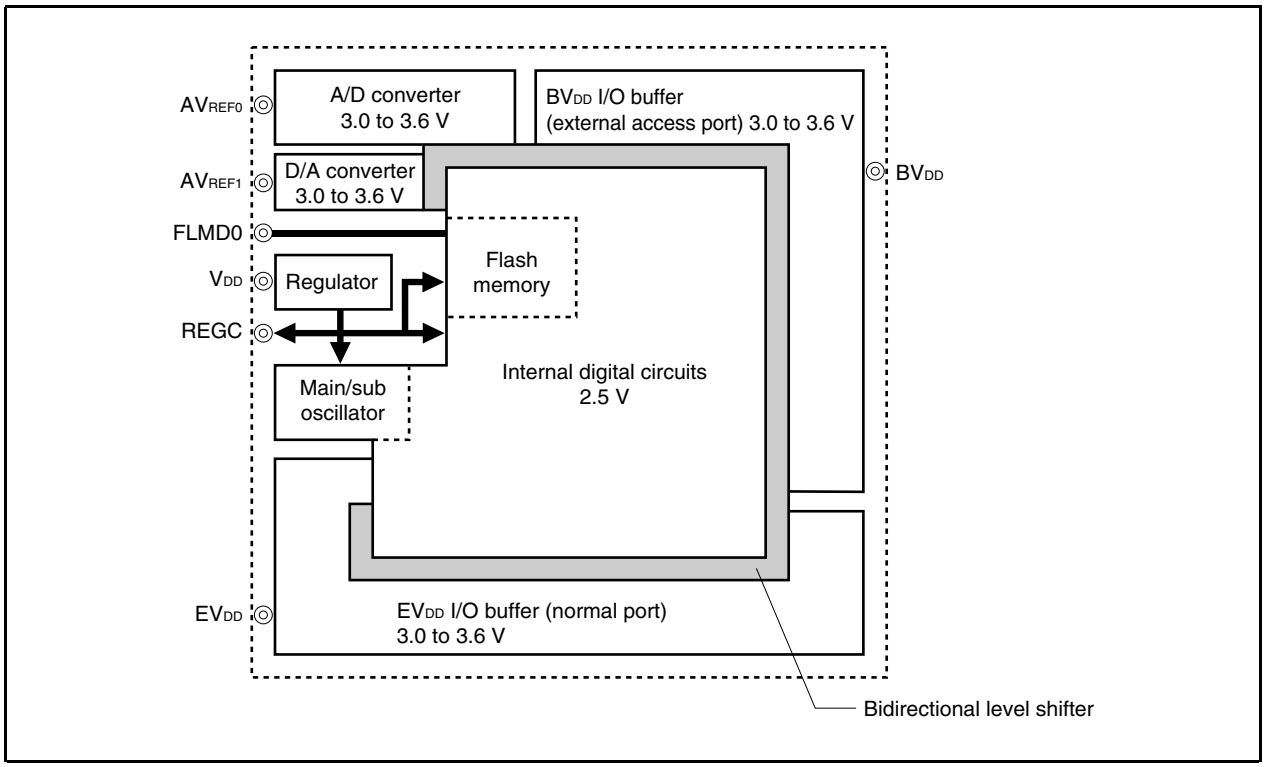
CHAPTER 26 REGULATOR

26.1 Outline

The V850ES/SG2 include a regulator to reduce the power consumption and noise.

This regulator supplies a stepped-down V_{DD} power supply voltage to the oscillator block and internal logic circuits (except the A/D converter, D/A converter, and output buffer). The regulator output voltage is set to 2.5 V (±0.2 V).

Figure 26-1. Regulator



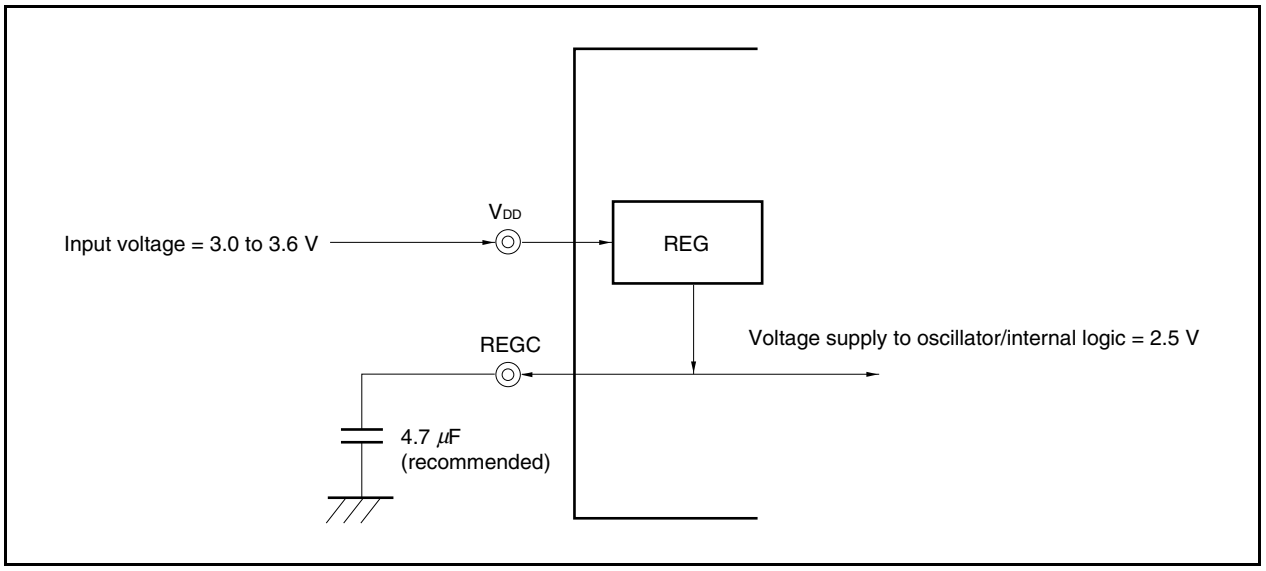
26.2 Operation

The regulator of this product always operates in any mode (normal operation mode, HALT mode, IDLE1 mode, IDLE2 mode, software STOP mode, or during reset).

Be sure to connect a capacitor (4.7 μF (recommended value)) to the REGC pin to stabilize the regulator output.

A diagram of the regulator pin connection method is shown below.

Figure 26-2. REGC Pin Connection (REGC = Capacity)



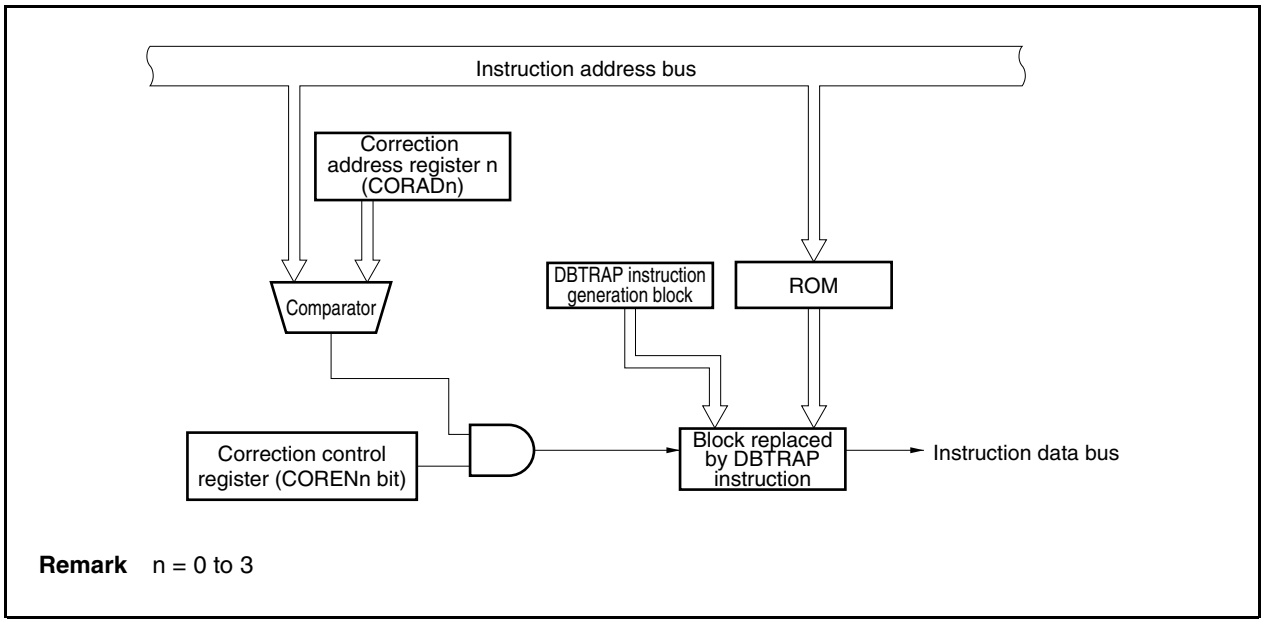
CHAPTER 27 ROM CORRECTION FUNCTION

27.1 Overview

The ROM correction function is used to replace part of the program in the mask ROM with the program of an external RAM or the internal RAM.

By using this function, instruction bugs found in the mask ROM can be corrected at up to four places.

Figure 27-1. Block Diagram of ROM Correction



Remark n = 0 to 3

27.2 Control Registers

(1) Correction address registers 0 to 3 (CORAD0 to CORAD3)

The CORAD0 to CORAD3 registers set the first address (correction address) of the instruction to be corrected in the ROM.

The program can be corrected at up to four places because four CORADn registers are provided (n = 0 to 3).

The CORADn register can be read or written in 32-bit units.

If the higher 16 bits of the CORADn register are used as the CORADnH register, and the lower 16 bits as the CORADnL register, these registers can be read or written in 16-bit units.

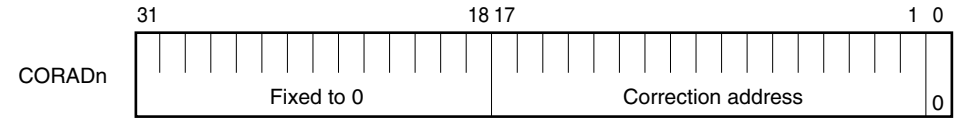
Reset input clears these registers to 00000000H.

Because the ROM capacity differs from one product to another, set the correction addresses in the following ranges.

- μPD703260, 703260Y, 703270, 703270Y, 703280, 703280Y (256 KB): 0000000H to 003FFFFH
- μPD703261, 703261Y, 703271, 703271Y, 703281, 703281Y (384 KB): 0000000H to 005FFFFH
- μPD703262, 703262Y, 703272, 703272Y, 703282, 703282Y (512 KB): 0000000H to 007FFFFH
- μPD703263, 703263Y, 703273, 703273Y, 703283, 703283Y (640 KB): 0000000H to 009FFFFH

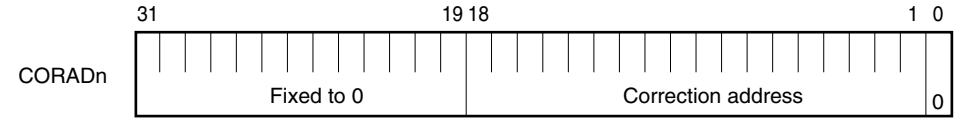
After reset: 00000000H R/W Address: CORAD0 FFFFF840H CORAD2 FFFFF848H
CORAD1 FFFFF844H CORAD3 FFFFF84CH

(a) 256 KB



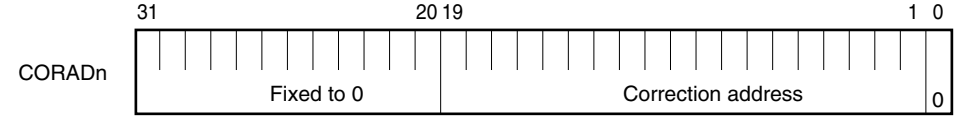
(n = 0 to 3)

(b) 384 KB, 512 KB



(n = 0 to 3)

(c) 640 KB



(n = 0 to 3)

(2) Correction control register (CORCN)

The CORCN register disables or enables the correction operation of each CORADn register (n = 0 to 3).

Each channel can be enabled or disabled by this register.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: FFFFF880H

	7	6	5	4	3	2	1	0
CORCN	0	0	0	0	COREN3	COREN2	COREN1	COREN0

CORENn	Enables/disables correction operation
0	Disabled
1	Enabled

Remark n = 0 to 3

Table 27-1. Correspondence Between CORCN Register Bits and CORADn Registers

CORCN Register Bit	Corresponding CORADn Register
COREN3	CORAD3
COREN2	CORAD2
COREN1	CORAD1
COREN0	CORAD0

27.3 ROM Correction Operation and Program Flow

<1> If the address to be corrected and the fetch address of the internal ROM match, the fetch code is replaced by the DBTRAP instruction.

<2> When the DBTRAP instruction is executed, execution branches to address 00000060H.

<3> Software processing after branching causes the result of ROM correction to be judged (the fetch address and ROM correction operation are confirmed) and execution to branch to the correction software.

<4> After the correction software has been executed, the return address is set, and return processing is started by the DBRET instruction.

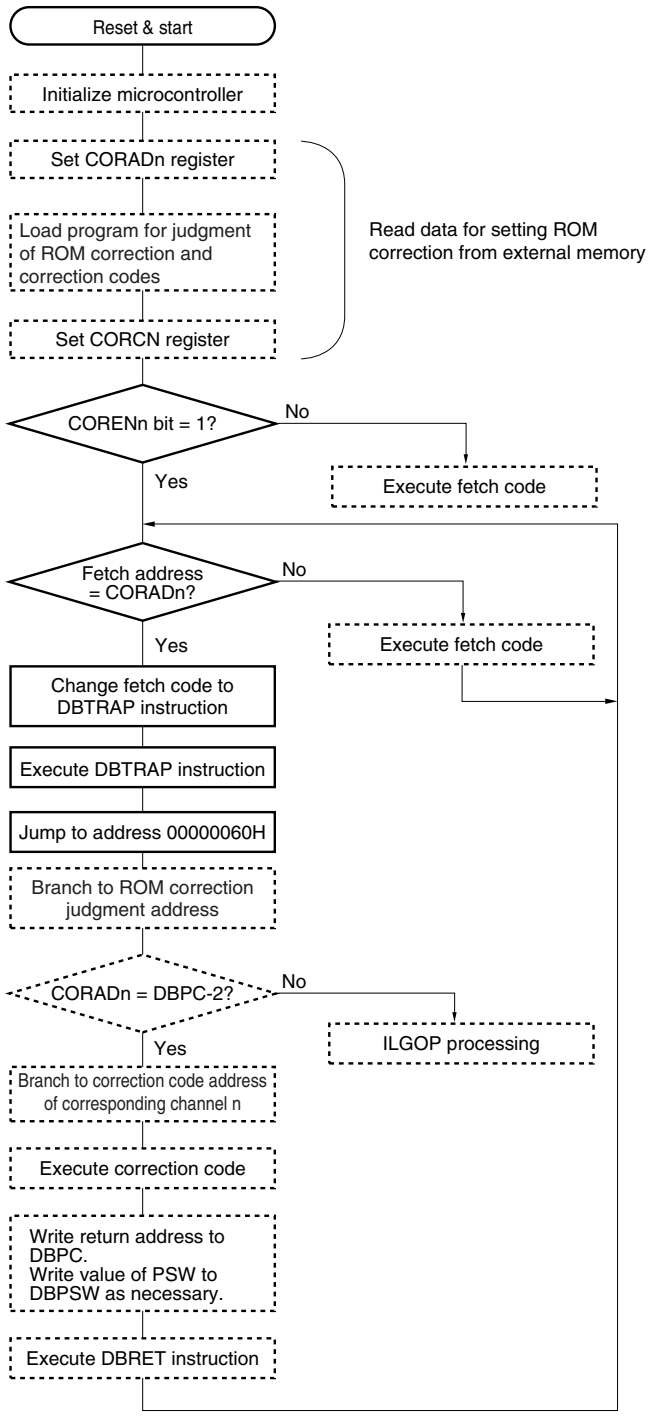
Cautions 1. The software that performs <3> and <4> must be executed in the internal ROM/RAM.

2. Develop the program so that the ROM correction function is not used until data has been completely written to the CORCN register that controls ROM correction.

3. When setting an address to be corrected to the CORADn register, clear the higher bits to 0 in accordance with the capacity of the internal ROM.

4. The ROM correction function cannot be used to correct the data of the internal ROM. It can only be used to correct instruction codes. If ROM correction is used to correct data, that data is replaced with the DBTRAP instruction code.

Figure 27-2. ROM Correction Operation and Program Flow



Remarks 1. : Processing by user program (software)

: Processing by ROM correction (hardware)

2. n = 0 to 3

CHAPTER 28 FLASH MEMORY

The following products are the flash memory versions of the V850ES/SG2.

Caution There are differences in the amount of noise tolerance and noise radiation between flash memory versions and mask ROM versions. When considering changing from a flash memory version to a mask ROM version during the process from experimental manufacturing to mass production, make sure to sufficiently evaluate commercial samples (CS) (not engineering samples (ES)) of the mask ROM versions.

- μPD70F3261, 70F3261Y, 70F3271, 70F3271Y, 70F3281, 70F3281Y: 384 KB flash memory versions
- μPD70F3263, 70F3263Y, 70F3273, 70F3273Y, 70F3283, 70F3283Y: 640 KB flash memory versions

In the instruction fetch to this flash memory, 4 bytes can be accessed by a single clock, the same as in the mask ROM version.

Writing to flash memory can be performed with the 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 as the development environment and the applications using a flash memory.

- Software can be altered after the V850ES/SG2 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.

28.1 Features

- 4-byte/1-clock access (in instruction fetch access)
- Chip erase/block unit erase
- Communication via serial interface with the dedicated flash programmer
- Erase/write voltage: Can be erased/written with a single power supply (FLMD0 = V_{DD}, FLMD1 = V_{SS}).
- On-board programming
- Flash memory programming by self-writing

28.1.1 Erasure unit

The erasure units for 384/640 KB flash memory versions are shown below.

(1) Chip erase

The area of xx000000H to xx05FFFFH and xx000000 to xx09FFFFH can be erased at the same time.

(2) Block erase

Erasure can be performed in block units (28 KB × 4, 4 KB × 4, 64 KB × 8[Note]).

Block 0: 28 KB
Block 1: 28 KB
Block 2: 28 KB
Block 3: 28 KB
Block 4: 8 KB
Block 5: 8 KB
Block 6: 8 KB
Block 7: 8 KB
Block 8: 64 KB
Block 9: 64 KB
Block 10: 64 KB
Block 11: 64 KB
Block 12: 64 KB
Block 13: 64 KB
Block 14: 64 KB
Block 15: 64 KB

Note 384 KB (μPD70F3261, 70F3261Y, 70F3271, 70F3271Y, 70F3281, and 70F3281Y) are divided into four blocks, 8 to 11.

28.2 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 V850ES/SG2 is mounted on the target system. Mount connectors, etc., on the target system to connect the dedicated flash programmer.

(2) Off-board programming

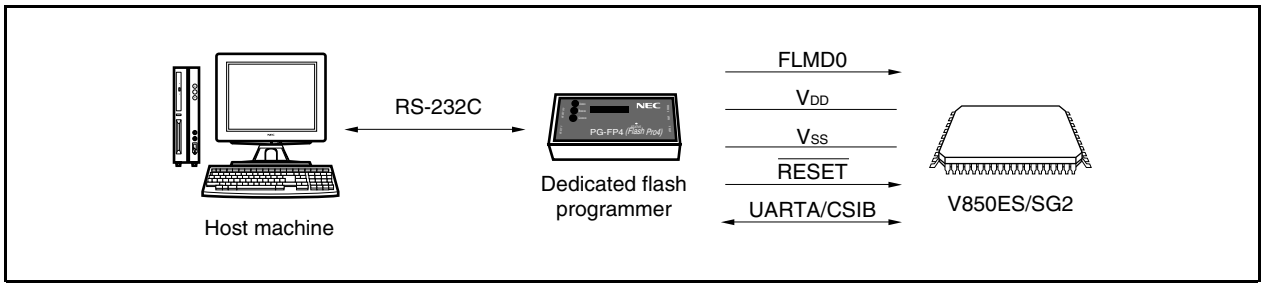
Writing to flash memory is performed by a dedicated program adapter (FA Series), etc., before mounting the V850ES/SG2 on the target system.

Remark FA Series is a product of Naito Densei Machida Mfg. Co., Ltd.

28.3 Programming Environment

The following shows the environment required for writing programs to the flash memory of V850ES/SG2.

Figure 28-1. Environment Required for Writing Programs to Flash Memory



A host machine is required for controlling the dedicated flash programmer.

UARTA0 or CSIB0 is used for the interface between the dedicated flash programmer and the V850ES/SG2 to perform writing, erasing, etc. A dedicated program adapter (FA Series) required for off-board writing.

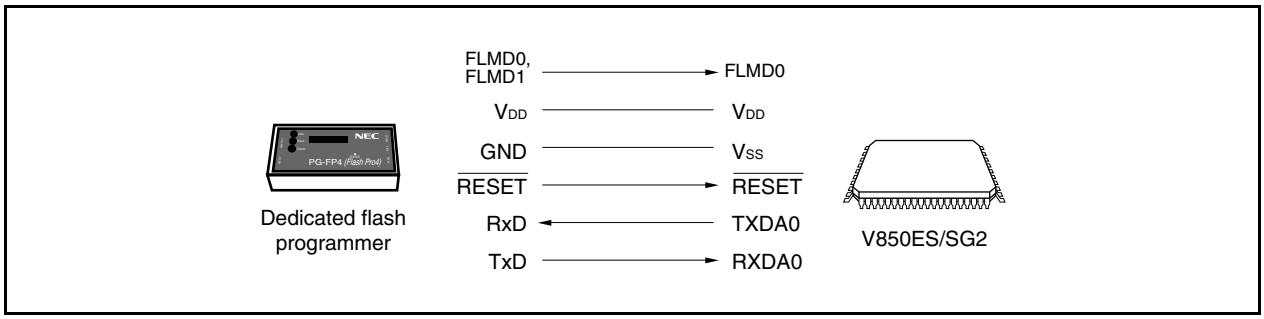
28.4 Communication Mode

The communication between the dedicated flash programmer and the V850ES/SG2 is performed by serial communication using UARTA0 or CSIB0 of the V850ES/SG2.

(1) UARTA0

Transfer rate: 4,800 to 76,800 bps

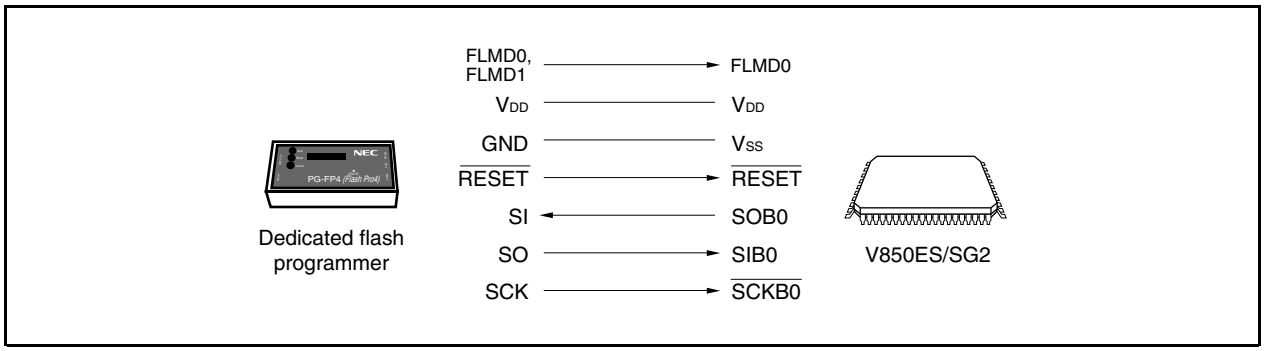
Figure 28-2. Communication with Dedicated Flash Programmer (UARTA0)



(2) CSIB0

Serial clock: Up to 1 MHz (MSB first)

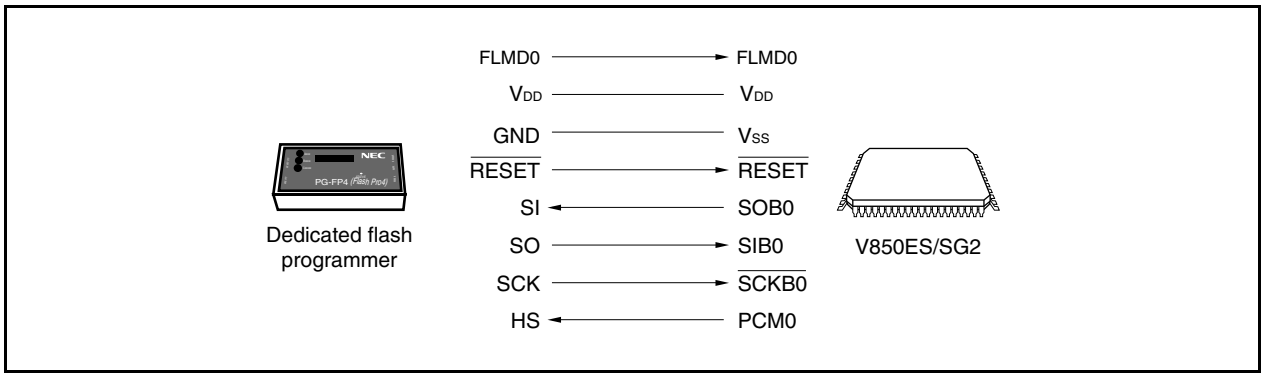
Figure 28-3. Communication with Dedicated Flash Programmer (CSIB0)



(3) CSIB0 + HS

Serial clock: Up to 1 MHz (MSB first)

Figure 28-4. Communication with Dedicated Flash Programmer (CSIB0 + HS)



The dedicated flash programmer outputs the transfer clock, and the V850ES/SG2 operate as slaves.

When the PG-FP4 is used as the dedicated flash programmer, it generates the following signals to the V850ES/SG2. For details, refer to the PG-FP4 manual.

Table 28-1. Signal Generation of Dedicated Flash Programmer (PG-FP4)

PG-FP4			V850ES/SG2	Connection Handling		
Signal Name	I/O	Pin Function	Pin Name	CSIB0	UARTA0	CSIB0 + HS
FLMD0	Output	Writing enable/disable	FLMD0	◎	◎	◎
V_{DD}	I/O	V_{DD} voltage generation/ voltage monitoring	V_{DD}	◎	◎	◎
GND	–	Ground	V_{SS}	◎	◎	◎
CLK	Output	Clock output to V850ES/SG2	X1	×	×	×
$\overline{\text{RESET}}$	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎	◎
SI/RxD	Input	Receive signal	SOB0/TxDA0	◎	◎	◎
SO/TxD	Output	Transmit signal	SIB0/RxDA0	◎	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCKB0}}$	×	×	◎
HS	Input	Handshake signal of CSIB0 + HS	PCM0	×	×	◎

Remark ◎: Always connected

×: Does not need to be connected

28.5 Pin Connection

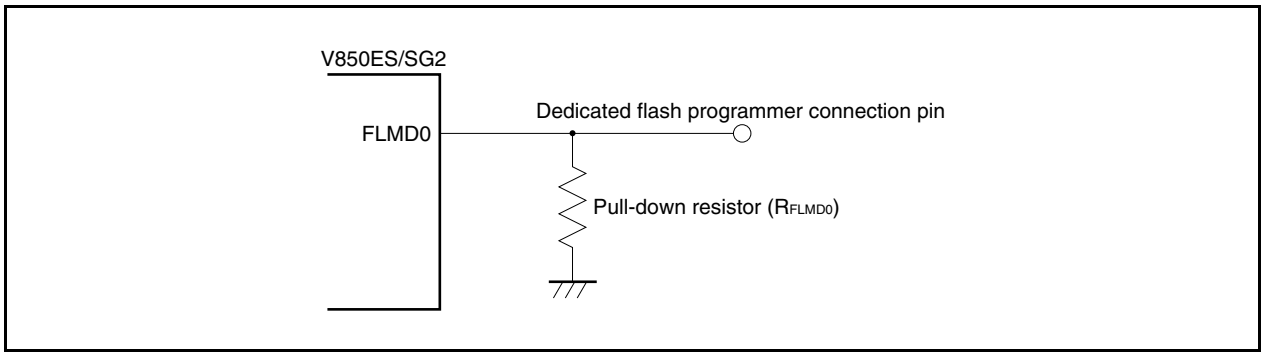
When performing on-board writing, mount a connector on the target system to connect to the dedicated flash programmer. Also, incorporate a function on-board to switch from the normal operation mode to the flash memory programming mode.

When switched to the flash memory programming mode, all the pins not used for flash memory programming become the same status as that immediately after reset. Therefore, all the ports enter the output high-impedance status, so that pin handling is required when the external device does not acknowledge the output high-impedance status.

28.5.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the V_{DD} write voltage is supplied to the FLMD0 pin. The following shows an example of the connection of the FLMD0 pin.

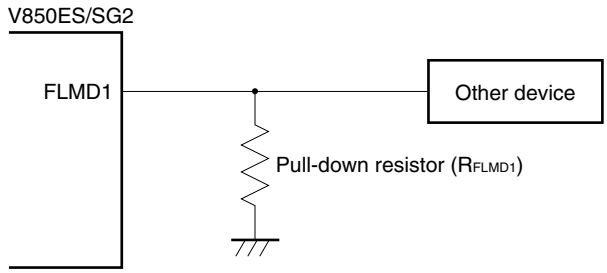
Figure 28-5. FLMD0 Pin Connection Example



28.5.2 FLMD1 pin

When 0 V is input to the FLMD0 pin, the FLMD1 pin does not function. When V_{DD} is supplied to the FLMD0 pin, the flash memory programming mode is entered, so 0 V must be input to the FLMD1 pin. An FLMD1 pin connection example is shown below.

Figure 28-6. FLMD1 Pin Connection Example



Caution If the V_{DD} signal is input to the FLMD1 pin from another device during on-board writing and immediately after reset, isolate this signal.

Table 28-2. Relationship of Operation Mode of FLMD0 and FLMD1 Pins

FLMD0	FLMD1	Operation Mode
0	×	Normal operation mode
V_{DD}	0	Flash memory programming mode
V_{DD}	V_{DD}	Setting prohibited

28.5.3 Serial interface pin

The following shows the pins used by each serial interface.

Table 28-3. Pins Used by Serial Interfaces

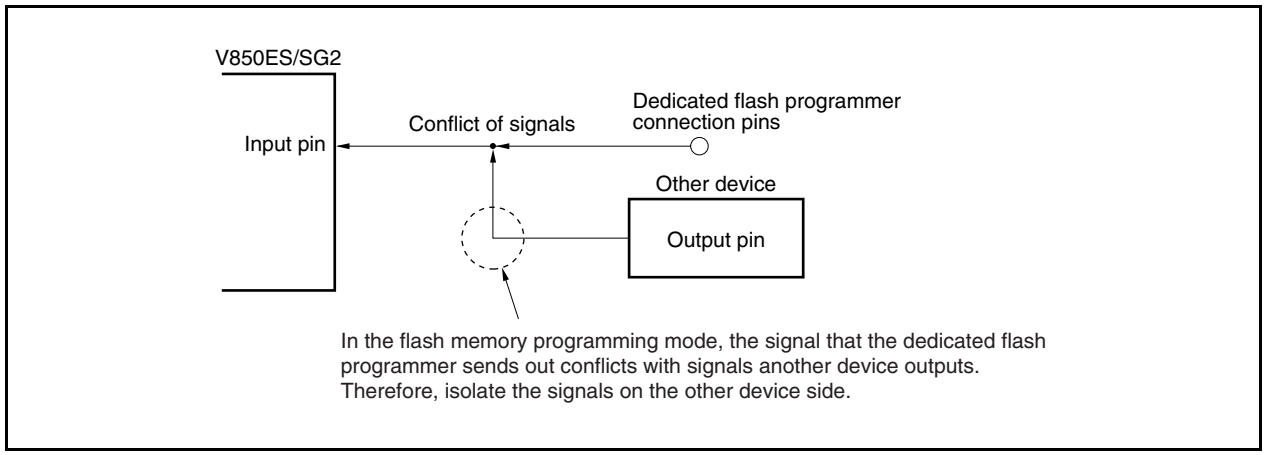
Serial Interface	Pins Used
CSIB0	SOB0, SIB0, $\overline{\text{SCKB0}}$
CSIB0 + HS	SOB0, SIB0, $\overline{\text{SCKB0}}$, PCM0
UARTA0	TXDA0, RXDA0

When connecting a dedicated flash programmer to a serial interface pin that is connected to another device on-board, care should be taken to avoid conflict of signals and malfunction of the other device.

(1) Conflict of signals

When the dedicated flash programmer (output) is connected 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 the output high-impedance status.

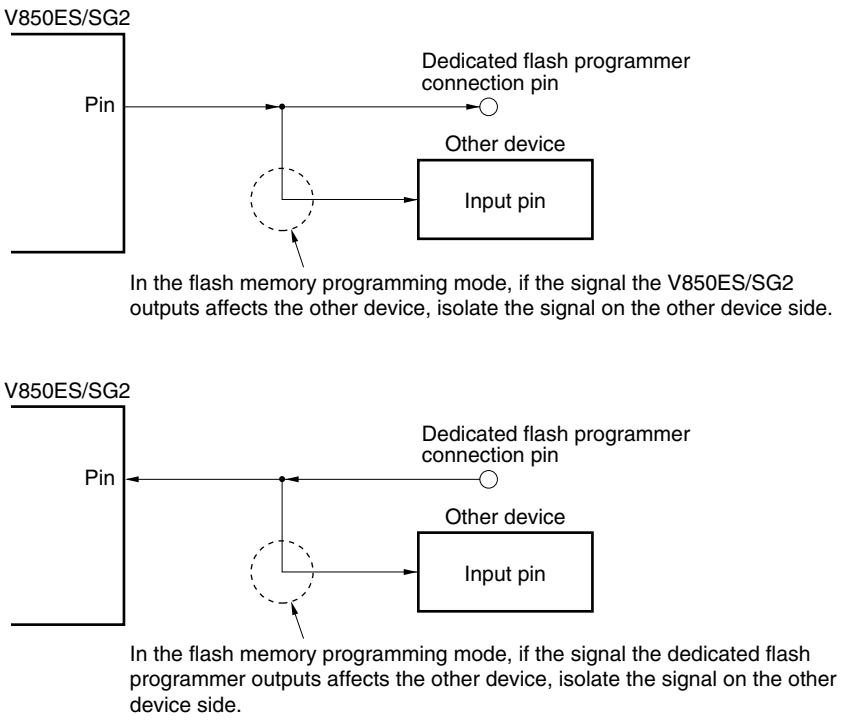
Figure 28-7. Conflict of Signals (Serial Interface Input Pin)



(2) Malfunction of other device

When the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), the signal is output to the other device, causing 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 28-8. Malfunction of Other Device

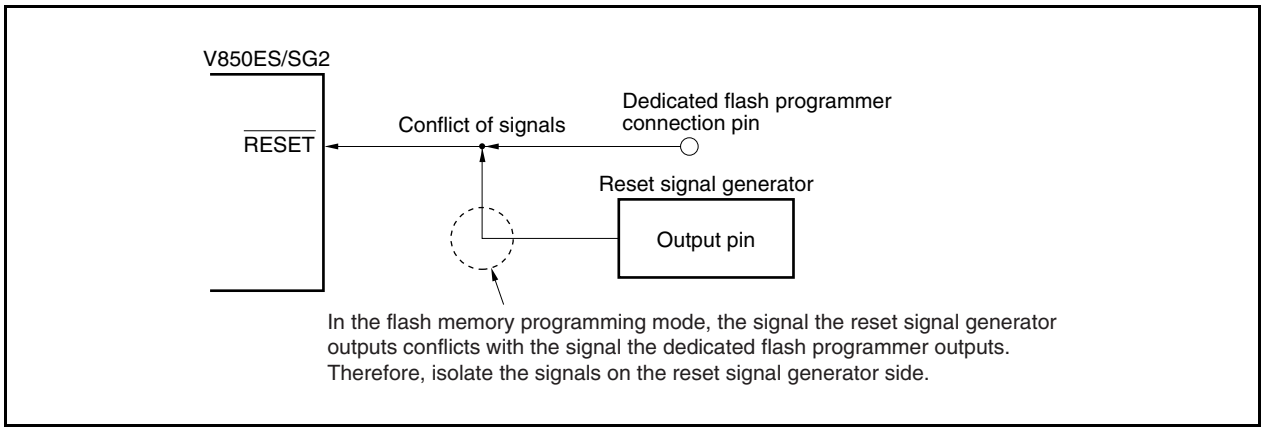


28.5.4 $\overline{\text{RESET}}$ pin

When the reset signals of the dedicated flash programmer are connected 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 28-9. Conflict of Signals ($\overline{\text{RESET}}$ Pin)



28.5.5 Port pins (including NMI)

When the flash memory programming mode is set, all the port pins except the pins that communicate with the dedicated flash programmer enter the output high-impedance status. If problems such as disabling output high-impedance status should occur in the external devices connected to the port, connect the port pins to V_{DD} or V_{SS} via resistors.

28.5.6 Other signal pins

Connect X1, X2, XT1, and XT2 to the same status as that in the normal operation mode.

28.5.7 Power supply

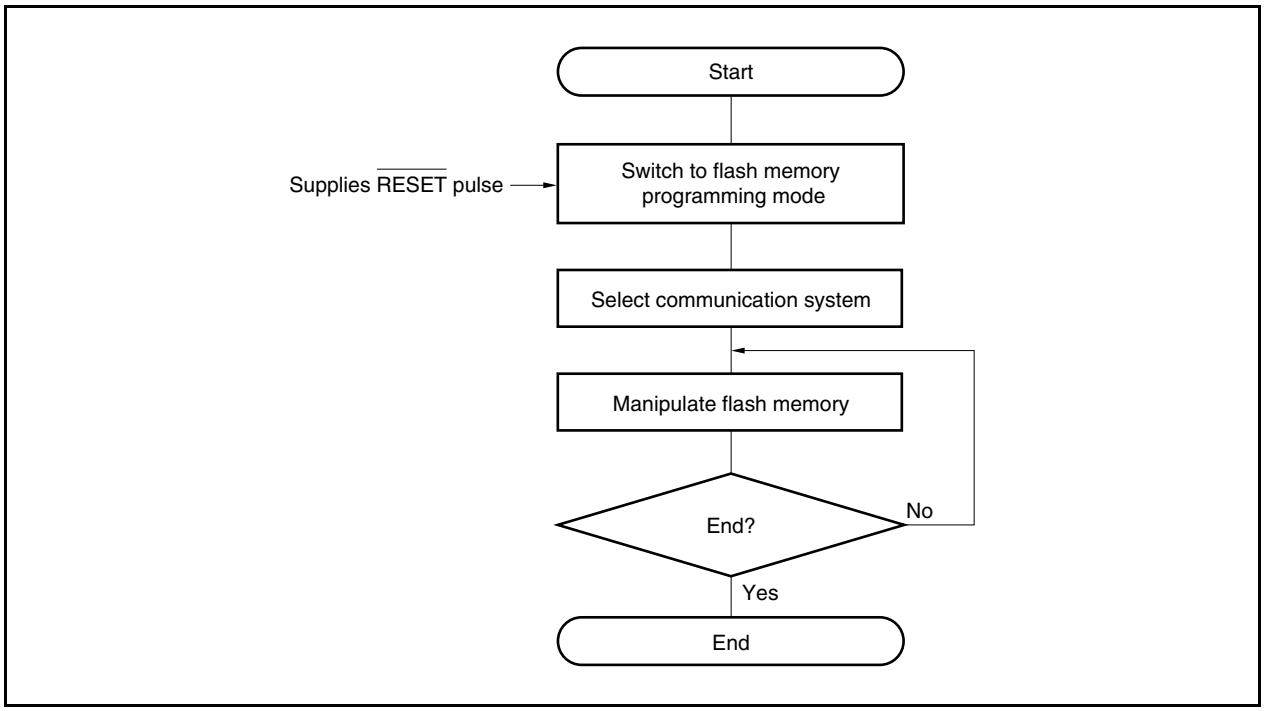
Supply the same power (V_{DD}, V_{SS}, EV_{DD}, EV_{SS}, AV_{SS}, BV_{DD}, BV_{SS}, AV_{REF0}, AV_{REF1}) as in normal operation mode.

28.6 Programming Method

28.6.1 Flash memory control

The following shows the procedure for manipulating the flash memory.

Figure 28-10. Procedure for Manipulating Flash Memory

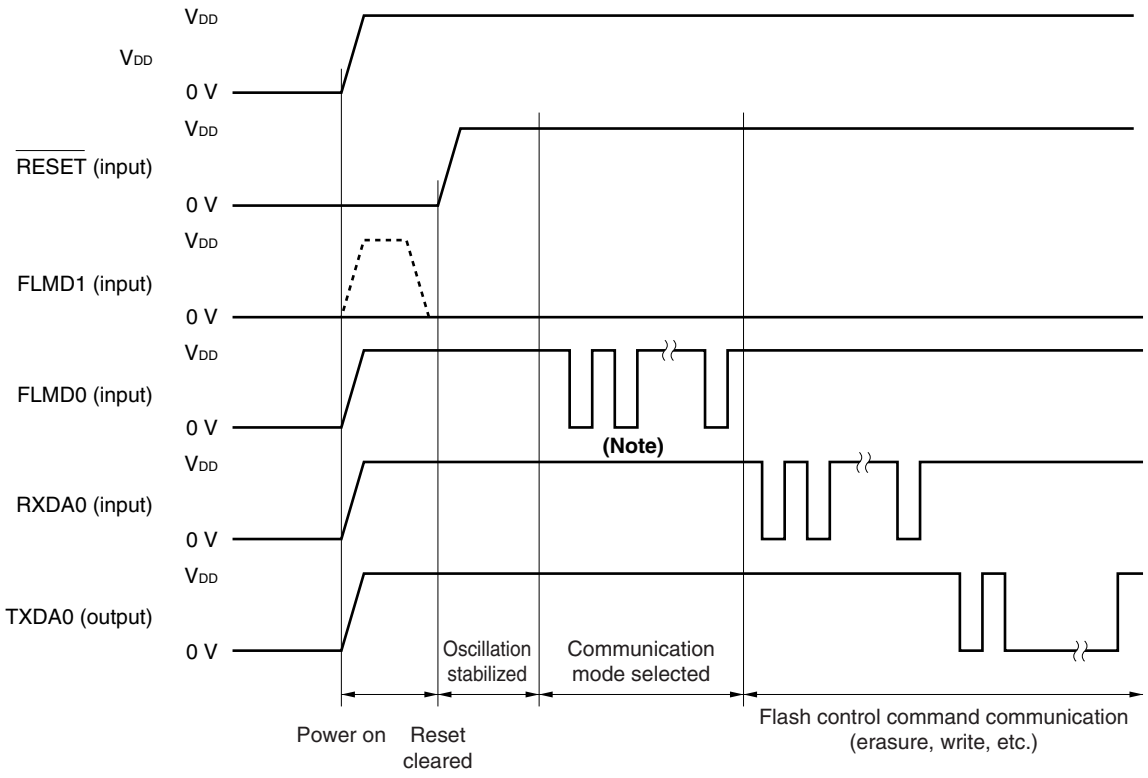


28.6.2 Flash memory programming mode

When rewriting the contents of flash memory using the dedicated flash programmer, set the V850ES/SG2 to the flash memory programming mode. When switching modes, set the FLMD0 and FLMD1 pins before releasing reset.

When performing on-board writing, change modes using a jumper, etc.

Figure 28-11. Flash Memory Programming Mode



Note The number of clocks to be inserted differs depending on the communication mode. For details, see **Table 28-4**.

28.6.3 Selection of communication mode

In the V850ES/SG2, the communication mode is selected by inputting pulses (12 pulses max.) to the FLMD0 pin after switching to the flash memory programming mode. The FLMD0 pulse is generated by the dedicated flash programmer.

The following shows the relationship between the number of pulses and the communication mode.

Table 28-4. List of Communication Modes

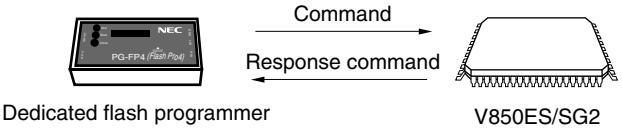
FLMD0 Pulse	Communication Mode	Remarks
8	CSIB0	V850ES/SG2 perform slave operation, MSB first
11	CSIB0 + HS	V850ES/SG2 perform slave operation, MSB first
9	CSIB3	V850ES/SG2 perform slave operation, MSB first
12	CSIB3 + HS	V850ES/SG2 perform slave operation, MSB first
0	UARTA0	Communication rate: 9,600 bps (at reset), LSB first
Others	RFU	Setting prohibited

Caution When UARTA is selected, the receive clock is calculated based on the reset command sent from the dedicated flash programmer after receiving the FLMD0 pulse.

28.6.4 Communication command

The V850ES/SG2 communicate with the dedicated flash programmer by means of commands. The command sent from the dedicated flash programmer to the V850ES/SG2 is called a “command”. The response signal sent from the V850ES/SG2 to the dedicated flash programmer is called a “response command”.

Figure 28-12. Communication Command



The following shows the commands for flash memory control of the V850ES/SG2. All of these commands are issued from the dedicated flash programmer, and the V850ES/SG2 perform the various processing corresponding to the commands.

Table 28-5. Flash Memory Control Command

Category	Command Name	Support			Function
		CSIB	CSIB + HS	UARTA	
Blank check	Block blank check command	√	√	√	Checks the erase state of the entire memory.
Erase	Chip erase command	√	√	√	Erases the contents of the entire memory.
	Block erase command	√	√	√	Erases the contents of the specified block memory.
Write	Write command	√	√	√	Writes data according to the specification of the write address and the number of bytes to be written, and executes a verify check.
Verify	Verify command	√	√	√	Compares the contents of the entire memory and the input data.
System setting and control	Reset command	√	√	√	Escapes from each state.
	Oscillating frequency setting command	√	√	√	Sets the oscillation frequency.
	Baud rate setting command	–	–	√	Sets the baud rate when using UART.
	Silicon signature command	√	√	√	Reads the silicon signature information.
	Version acquisition command	√	√	√	Reads the version information of the device.
	Status command	√	√	–	Acquires the operation status.
	Security setting command	√	√	√	Erases chip and blocks, and sets security of write.

The V850ES/SG2 return response commands to the commands issued from the dedicated flash programmer. The following shows the response commands returned by the V850ES/SG2.

Table 28-6. Response Commands

Response Command Name	Function
ACK	Acknowledges command/data, etc.
NAK	Acknowledges illegal command/data, etc.