To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

RENESAS

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

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

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

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

User's Manual

V850E/CG3™ CarGate+

32-bit RISC Microcontroller

Hardware

μPD70F3178

Document No. U16881EE2V1UM00
Date Published April 2006

© NEC Electronics Corporation 2006
Printed in Germany

NOTES FOR CMOS DEVICES

① **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN**

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (MAX) and V_{IH} (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (MAX) and V_{IH} (MIN).

② **HANDLING OF UNUSED INPUT PINS**

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

③ **PRECAUTION AGAINST ESD**

A 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 when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up 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 benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

④ **STATUS BEFORE INITIALIZATION**

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

⑤ **POWER ON/OFF SEQUENCE**

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ **INPUT OF SIGNAL DURING POWER OFF STATE**

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements.

Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

- **The information in this document is current as of April, 2006. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without 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 cannot 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 product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC 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).

M8E 02. 11-1

All (other) product, brand, or trade names used in this pamphlet are the trademarks or registered trademarks of their respective owners.
Product specifications are subject to change without notice. To ensure that you have the latest product data, please contact your local NEC Electronics sales office.

For further information, please contact:

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668,
Japan
Tel: 044-435-5111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554, U.S.A.
Tel: 408-588-6000
800-366-9782
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211-65030
http://www.eu.necel.com/

Hanover Office
Podbielski Strasse 166 B
30177 Hannover
Tel: 0 511 33 40 2-0

Munich Office
Werner-Eckert-Strasse 9
81829 München
Tel: 0 89 92 10 03-0

Stuttgart Office
Industriestrasse 3
70565 Stuttgart
Tel: 0 711 99 01 0-0

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908-691-133

Succursale Française
9, rue Paul Dautier, B.P. 52180
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091-504-2787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 638 72 00

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02-667541

Branch The Netherlands
Limburglaan 5
5616 HR Eindhoven
The Netherlands
Tel: 040 265 40 10

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
TEL: 010-8235-1155
http://www.cn.necel.com/

NEC Electronics Shanghai Ltd.
Room 2509-2510, Bank of China Tower,
200 Yincheng Road Central,
Pudong New Area, Shanghai P.R. China P.C:200120
Tel: 021-5888-5400
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
12/F., Cityplaza 4,
12 Taikoo Wan Road, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

Seoul Branch
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R. O. C.
Tel: 02-2719-2377

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253-8311
http://www.sg.necel.com/

G05.11-1A

Preface

Readers

This manual is intended for users who want to understand the functions of the V850E/CG3 (nickname CarGate+).

Purpose

This manual presents the hardware manual of V850E/CG3.

Organization

This system specification describes the following sections:

- Pin function
- CPU function
- Internal peripheral function
- Flash memory

Legend

Symbols and notation are used as follows:

Weight in data notation	:	Left is high-order column, right is low order column
Active low notation	:	$\overline{\text{xxx}}$ (pin or signal name is over-scored) or /xxx (slash before signal name)
Memory map address:	:	High order at high stage and low order at low stage
Note	:	Explanation of (Note) in the text
Caution	:	Item deserving extra attention
Remark	:	Supplementary explanation to the text
Numeric notation	:	Binary... xxxx or xxxB Decimal... xxxx Hexadecimal... xxxxH or 0x xxxx

Prefixes representing powers of 2 (address space, memory capacity)

K (kilo): $2^{10} = 1024$

M (mega): $2^{20} = 1024^2 = 1,048,576$

G (giga): $2^{30} = 1024^3 = 1,073,741,824$

Table of Contents

Preface 5

Chapter 1 Introduction 25
- **1.1 General** 25
- **1.2 Device Features** 26
- **1.3 Application Fields** 27
- **1.4 Ordering Information** 27
- **1.5 Pin Configuration (Top View)** 28
- **1.6 Configuration of Function Block** 30
  - 1.6.1 Block Diagram of μPD70F3178 30
  - 1.6.2 On-chip units 31

Chapter 2 Pin Functions 33
- **2.1 List of Pin Functions** 33
- **2.2 Description of Pin Functions** 39
- **2.3 Types of Pin I/O Circuit and Connection of Unused Pins** 52

Chapter 3 CPU Function 55
- **3.1 Features** 55
- **3.2 CPU Register Set** 56
  - 3.2.1 Program register set 57
  - 3.2.2 System register set 58
- **3.3 Operation Modes** 62
  - 3.3.1 Operation modes 62
- **3.4 Address Space** 63
  - 3.4.1 CPU address space 63
  - 3.4.2 Image 64
  - 3.4.3 Wrap-around of CPU address space 65
- **3.5 Memory Map** 66
  - 3.5.1 Area 67
  - 3.5.2 Recommended use of address space 71
- **3.6 Peripheral I/O Registers** 72
- **3.7 PPA related SFR Register** 79
  - 3.7.1 Programmable peripheral I/O registers 185
- **3.8 Specific Registers** 187
  - 3.8.1 Command register (PRCMD) 188
  - 3.8.2 Internal peripheral function wait control register (VSWC) 189

Chapter 4 Bus Control Function 191
- **4.1 Features** 191
- **4.2 Bus Control Pins** 191
- **4.3 Memory Block Function** 192
  - 4.3.1 Chip select control function 193
- **4.4 Bus Cycle Type Control Function** 195
  - 4.4.1 Bus cycle type configuration 195
- **4.5 Bus Access** 196
  - 4.5.1 Number of access clocks 196
  - 4.5.2 Bus sizing function 197
  - 4.5.3 Endian control function 198
  - 4.5.4 Bus width 200
- **4.6 Wait Function** 212
  - 4.6.1 Programmable wait function 212
- **4.7 Idle State Insertion Function** 214
- **4.8 Bus Priority Order** 215

4.9 Boundary Operation Conditions . . . 216
4.9.1 Program space . . . 216
4.9.2 Data space . . . 216

Chapter 5 Clock Generator . . . 217
5.1 Features . . . 217
5.2 Configuration . . . 217
5.3 Control Registers . . . 218
5.3.1 Clock control register (CKC) . . . 218
5.3.2 PLL status register (PSTAT) . . . 219
5.3.3 PLL control register (PLC) . . . 220
5.4 Power Saving Functions . . . 221
5.4.1 General . . . 221
5.4.2 Power save modes outline . . . 223
5.4.3 HALT mode . . . 224
5.4.4 IDLE mode . . . 226
5.4.5 Software STOP mode . . . 227
5.5 Register Description . . . 229
5.5.1 Power save control register (PSC) . . . 229
5.5.2 Power save mode register (PSM) . . . 230

Chapter 6 16-Bit Timer/Event Counter P . . . 231
6.1 Features . . . 231
6.2 Functional Outline . . . 231
6.3 Configuration . . . 232
6.4 Control Registers . . . 237
6.5 Operation . . . 248
6.5.1 Anytime write and reload . . . 249
6.5.2 Interval timer mode (TPnMD2 to TPnMD0 = 000) . . . 253
6.5.3 External trigger pulse mode (TPnMD2 to TPnMD0 = 010) . . . 256
6.5.4 One-shot pulse mode (TPnMD2 to TPnMD0 = 011) . . . 259
6.5.5 PWM mode (TPnMD2 to TPnMD0 = 100) . . . 262
6.5.6 Free-running mode (TPnMD2 to TPnMD0 = 101) . . . 267
6.5.7 Pulse width measurement mode (TPnMD2 to TPnMD0 = 110) . . . 272
6.6 Timer Synchronized Operation Function . . . 274

Chapter 7 Memory Access Control Function . . . 277
7.1 SRAM, External ROM, External I/O Interface . . . 277
7.1.1 Features . . . 277
7.1.2 SRAM connections . . . 278
7.1.3 SRAM, external ROM, external I/O access . . . 279
7.2 Page ROM Controller (ROMC) . . . 283
7.2.1 Features . . . 283
7.2.2 Page ROM connections . . . 284
7.2.3 On-page/off-page judgment . . . 285
7.2.4 Page ROM configuration register (PRC) . . . 287
7.2.5 Page ROM access . . . 288

Chapter 8 Interrupt/Exception Processing Function . . . 293
8.1 Features . . . 293
8.2 Non-Maskable Interrupts . . . 296
8.2.1 Operation . . . 297
8.2.2 Restore . . . 298
8.2.3 Non-maskable interrupt status flag (NP) . . . 299
8.2.4 Edge detection function . . . 299

8.3 Maskable Interrupts . . . 300
8.3.1 Operation . . . 300
8.3.2 Restore . . . 302
8.3.3 Priorities of maskable interrupts . . . 303
8.3.4 Interrupt control register (xxIC) . . . 307
8.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3) . . . 310
8.3.6 In-service priority register (ISPR) . . . 311
8.3.7 Maskable interrupt status flag (ID) . . . 312
8.4 Noise Elimination Circuit . . . 313
8.4.1 Analog filter . . . 314
8.4.2 Interrupt trigger mode selection . . . 314
8.4.3 Interrupt edge detection control registers . . . 315
8.5 Software Exception . . . 319
8.5.1 Operation . . . 319
8.5.2 Restore . . . 320
8.5.3 Exception status flag (EP) . . . 321
8.6 Exception Trap . . . 322
8.6.1 Illegal opcode definition . . . 322
8.7 Multiple Interrupt Processing Control . . . 324
8.8 Interrupt Response Time . . . 326
8.9 Periods in Which Interrupts Are Not Acknowledged . . . 327

Chapter 9 A/D Converter . . . 329
9.1 Functions . . . 329
9.2 Configuration . . . 331
9.3 Control Registers . . . 333
9.4 Operation . . . 343
9.4.1 Basic operation . . . 343
9.4.2 Trigger mode . . . 345
9.4.3 Operation mode . . . 346
9.4.4 Power-fail compare mode . . . 348
9.5 Cautions . . . 351
9.6 How to Read A/D Converter Characteristics Table . . . 354

Chapter 10 Asynchronous Serial Interface A (UARTA) . . . 359
10.1 Features . . . 359
10.2 Configuration . . . 360
10.2.1 I/O pins . . . 361
10.2.2 Control registers . . . 361
10.3 Control Registers . . . 363
10.4 Interrupt Request Signals . . . 372
10.5 Operation . . . 373
10.5.1 Data format . . . 373
10.5.2 SBF transmission/reception format . . . 375
10.5.3 SBF transmission . . . 377
10.5.4 SBF reception . . . 378
10.5.5 UART transmission . . . 379
10.5.6 Procedure of continuous transmission . . . 380
10.5.7 UART reception . . . 382
10.5.8 Reception errors . . . 383
10.5.9 Types and operation of parity . . . 384
10.5.10 Noise filter of receive data . . . 385
10.5.11 Dedicated Baud Rate Generator . . . 386

Chapter 11 3-Wire Serial Interface (CSIB) 393
11.1 Features 393
11.2 Configuration 394
11.3 I/O Pins 395
11.3.1 Control registers 395
11.4 Control Registers 396
11.4.1 Transfer data length change function 402
11.4.2 Interrupt request signals 403
11.5 Operation 404
11.5.1 Single transfer (master mode, transmission/reception mode) 404
11.5.2 Single transfer mode (master mode, reception mode) 405
11.5.3 Continuous mode (master mode, transmission/reception mode) 406
11.5.4 Continuous mode (master mode, reception mode) 407
11.5.5 Continuous reception mode (error) 408
11.5.6 Continuous mode (slave mode, transmission/reception mode) 409
11.5.7 Continuous mode (slave mode, reception mode) 410
11.5.8 Clock timing 411
11.5.9 Output pin status with operation disabled 413
11.6 Operation Flow 414
11.7 Prescaler 3 420
11.7.1 Control registers of prescaler 3 420

Chapter 12 I²C Bus 423
12.1 Mode Switching of I²C Bus and Other Serial Interfaces 423
12.1.1 CSIB2 and I²C0 mode switching 423
12.2 Features 424
12.3 Configuration 425
12.4 Control Registers 429
12.5 I²C Bus Mode Functions 445
12.5.1 Pin configuration 445
12.6 I²C Bus Definitions and Control Methods 446
12.6.1 Start condition 446
12.6.2 Addresses 447
12.6.3 Transfer direction specification 448
12.6.4 Acknowledge signal ($\overline{ACK}$) 449
12.6.5 Stop condition 450
12.6.6 Wait signal ($\overline{WAIT}$) 451
12.7 I²C Interrupt Request Signals (INTIIC0) 453
12.7.1 Master device operation 453
12.7.2 Slave device operation (when receiving slave address data (address match)) 456
12.7.3 Slave device operation (when receiving extension code) 460
12.7.4 Operation without communication 463
12.7.5 Arbitration loss operation (operation as slave after arbitration loss) 464
12.7.6 Operation when arbitration loss occurs (no communication after arbitration loss) 466
12.8 Interrupt Request Signal (INTIIC0) Generation Timing and Wait Control 471
12.9 Address Match Detection Method 473
12.10 Error Detection 473
12.11 Extension Code 473
12.12 Arbitration 474
12.13 Wake-up Function 475
12.14 Communication Reservation 476
12.14.1 When communication reservation function is enabled (IICF0.IICRSV0 bit = 0) 476
12.14.2 When communication reservation function is disabled (IICF0.IICRSV0 bit = 1) 480
12.15 Cautions 481

12.16 Communication Operations . . . 481
12.16.1 Master operation 1 . . . 481
12.16.2 Master operation 2 . . . 483
12.16.3 Slave operation . . . 484
12.17 Timing of Data Communication . . . 487

Chapter 13 AFCAN Controller . . . 495
13.1 Outline Description . . . 495
13.1.1 Features . . . 495
13.1.2 Implementation . . . 495
13.1.3 Overview of functions . . . 496
13.1.4 Configuration . . . 497
13.2 CAN Protocol . . . 498
13.2.1 Frame format . . . 498
13.2.2 Frame types . . . 499
13.2.3 Data frame and remote frame . . . 500
13.2.4 Error frame . . . 508
13.2.5 Overload frame . . . 509
13.3 Functions . . . 510
13.3.1 Determining bus priority . . . 510
13.3.2 Bit stuffing . . . 510
13.3.3 Multi masters . . . 511
13.3.4 Multi cast . . . 511
13.3.5 CAN SLEEP mode/CAN STOP mode function . . . 511
13.3.6 Error control function . . . 511
13.3.7 Baud rate control function . . . 517
13.4 Connection With Target System . . . 521
13.5 Internal Registers of CAN Controller . . . 522
13.5.1 CAN Controller configuration . . . 522
13.5.2 Register access type . . . 523
13.5.3 Register bit configuration . . . 526
13.6 Control Registers . . . 530
13.6.1 CAN Global Control Register (CnGMCTRL) . . . 530
13.6.2 CAN Global Clock Selection Register (CnGMCS) . . . 532
13.6.3 CAN Global Automatic Block Transmission Control Register (CnGMABT) . . . 533
13.6.4 CAN Global Configuration Register (CnGMCONF) . . . 535
13.6.5 CAN Global Automatic Block Transmission Delay Register (CnGMABTD) . . . 536
13.6.6 CAN Module Mask Control Register (CnMASKaL, CnMASKaH) (a = 1, 2, 3, or 4) . . . 537
13.6.7 CAN Module Control Register (CnCTRL) . . . 539
13.6.8 CAN Module Last Error Code Register (CnLEC) . . . 543
13.6.9 CAN Module Information Register (CnINFO) . . . 544
13.6.10 CAN Module Error Counter Register (CnERC) . . . 545
13.6.11 CAN Module Interrupt Enable Register (CnIE) . . . 546
13.6.12 CAN Module Interrupt Status Register (CnINTS) . . . 548
13.6.13 CAN Module Bit Rate Prescaler Register (CnBRP) . . . 550
13.6.14 CAN Module Bit Rate Register (CnBTR) . . . 551
13.6.15 CAN Module Last In-Pointer Register (CnLIPT) . . . 553
13.6.16 CAN Module Receive History List Register (CnRGPT) . . . 554
13.6.17 CAN Module Last Out-Pointer Register (CnLOPT) . . . 556
13.6.18 CAN Module Transmit History List Register (CnTGPT) . . . 557
13.6.19 CAN Module Time Stamp Register (CnTS) . . . 559
13.6.20 CAN Message Data Byte Register (CnMDATAxm) (x = 0 to 7), (CnMDATAzm) (z = 01, 23, 45, 67) . . . 561
13.6.21 CAN Message Data Length Register m (CnMDLCm) . . . 563
13.6.22 CAN Message Configuration Register (CnMCONFm) . . . 564
13.6.23 CAN Message ID Register m (CnMIDLm, CnMIDHm) . . . 565
13.6.24 CAN Message Control Register m (CnMCTRLm) . . . 566

13.7 Bit Set/Clear Function . . . 569
13.8 CAN Controller Initialization . . . 571
13.8.1 Initialization of CAN module . . . 571
13.8.2 Initialization of message buffer . . . 571
13.8.3 Redefinition of message buffer . . . 571
13.8.4 Transition from Initialization mode to Operation mode . . . 573
13.8.5 Resetting Error Counter CnERC of CAN module . . . 573
13.9 Message Reception . . . 574
13.9.1 Message reception . . . 574
13.9.2 Receive history list function . . . 575
13.9.3 Mask function . . . 577
13.9.4 Multi buffer receive block function . . . 579
13.9.5 Remote frame reception . . . 580
13.10 Message Transmission . . . 581
13.10.1 Message transmission . . . 581
13.10.2 Transmit history list function . . . 583
13.10.3 Automatic block transmission (ABT) . . . 584
13.10.4 Transmission abort process . . . 586
13.10.5 Remote frame transmission . . . 587
13.11 Power Save Modes. . . . 587
13.11.1 CAN SLEEP mode . . . 587
13.11.2 CAN STOP Mode . . . 589
13.11.3 Example of using Power Saving modes . . . 590
13.12 Interrupt Function . . . 591
13.12.1 Interrupts generated by CAN module . . . 591
13.13 Diagnosis Functions and Special Operational Modes . . . 592
13.13.1 Receive-Only mode . . . 592
13.13.2 Single-Shot mode . . . 593
13.13.3 Self-Test mode . . . 594
13.14 Time Stamp Function. . . . 595
13.15 Baud Rate Settings . . . 596
13.15.1 Representative examples of baud rate settings . . . 600
13.16 Operation of CAN Controller . . . 604

Chapter 14 DAFCAN . . . 629
14.1 Introduction . . . 629
14.2 Overview of Functions. . . . 630
14.3 Architecture . . . 631
14.3.1 CPU I/F . . . 632
14.3.2 Global Macro Control . . . 632
14.3.3 CAN Interrupt Generator. . . . 632
14.3.4 Message Control (MSG Ctrl) . . . 633
14.3.5 Arbitration Logic . . . 633
14.3.6 RXONLY-CH CAN machine . . . 633
14.3.7 DIAG-CH CAN machine . . . 634
14.3.8 Memory and register layout . . . 634
14.4 Macro Initialisation and Control . . . 635
14.4.1 Global Macro initialisation and control . . . 636
14.4.2 Message buffer initialisation and configuration. . . . 643
14.4.3 Message buffer to CAN I/F channel assignment . . . 645
14.4.4 Configuration of a transmit message buffer . . . 650
14.4.5 DIAG Macro initialisation and control . . . 654
14.4.6 CAN bit time programming . . . 654
14.5 Macro Interrupts. . . . 659
14.6 Message Reception . . . 662
14.6.1 Principal reception process. . . . 662
14.6.2 Reception into a multi-buffer receive block. . . . 671
14.6.3 Reception of remote frames . . . 672

14.7 Message Transmission . . . 673
14.7.1 Principal transmission process . . . 673
14.7.2 Transmit history . . . 676
14.7.3 Automatic block transmission (ABT) . . . 679
14.7.4 Transmission request abort process . . . 681
14.7.5 Transmission of remote frames . . . 683
14.8 Operational Modes of RXONLY_CH . . . 684
14.8.1 Receive–only mode . . . 684
14.8.2 Mirror mode . . . 688
14.8.3 Mirror Mode with TIF . . . 690
14.9 Transitions for Buffer Assignment . . . 691
14.10 Register Description . . . 693
14.10.1 Registers of global macro control . . . 693
14.10.2 CAN Global Macro Control Register (CGMCTRL) . . . 697
14.10.3 CAN Global Macro Clock Selection Register (CGMCS) . . . 698
14.10.4 CAN Global Macro Automatic Block Transmission Register (CGMABT) . . . 699
14.10.5 CAN Global Configuration Register (CnGMCONF) . . . 700
14.10.6 CAN Global Macro Automatic Block Transmission Delay Register (CGMABTD) . . . 701
14.10.7 CAN Transfer ID Reference Registers (CnTIDRmL, CnTIDRmH; m = 0 - 7) . . . 702
14.11 CAN Transfer ID Mask Registers (CnTIDML, CnTIDMH) . . . 707
14.11.1 Registers of the CAN Module . . . 708
14.11.2 DIAG_CH CAN Module Mask 1 Registers (CnMASK1L, CnMASK1H) . . . 712
14.11.3 DIAG_CH CAN Module Mask 2 Registers (CnMASK2L, CnMASK2H) . . . 712
14.11.4 DIAG_CH CAN Module Mask 3 Registers (CnMASK3L, CnMASK3H) . . . 713
14.11.5 DIAG_CH CAN Module Mask 4 Registers (CnMASK4L, CnMASK4H) . . . 714
14.11.6 DIAG_CH CAN Module Control Register (CnCTRL) . . . 715
14.11.7 DIAG_CH CAN Module Last Error Code Register (CnLEC) . . . 718
14.11.8 DIAG_CH CAN Module Information Register (CnINFO) . . . 719
14.11.9 DIAG_CH CAN Module Error Counter (CnERC) . . . 720
14.11.10 DIAG_CH CAN Module Interrupt Enable Register (CnIE) . . . 720
14.11.11 DIAG-CAN Module Interrupt Status Register (CnINTS) . . . 721
14.11.12 DIAG_CH CAN Module Bit-Rate Prescaler Register (CnBRP) . . . 722
14.11.13 DIAG_CH CAN Bit Rate Register (CnBTR) . . . 723
14.11.14 DIAG_CH CAN Module Last In-Pointer Register (CnLIPT) . . . 724
14.11.15 DIAG_CH CAN Module Receive History List Get Pointer Register (CnRGPT) 725
14.11.16 DIAG_CH CAN Module Last Out-Pointer Register (CnLOPT) . . . 726
14.11.17 CAN Module Transmit History List Get Pointer Register (CnTGPT) . . . 727
14.11.18 DIAG_CH CAN Module Time Stamp Register (CnTS) . . . 728
14.11.19 RXONLY_CH CAN Module Control Register (CnCTRL_R) . . . 729
14.11.20 RXONLY_CH CAN Module Last Error Code Register (CnLEC_R) . . . 731
14.11.21 RXONLY_CH CAN Module Error Counter (CnERC_R) . . . 732
14.11.22 RXONLY_CH CAN Module Interrupt Enable Register (CnIE_R) . . . 732
14.11.23 RXONLY-CAN Module Interrupt Status Register (CnINTS_R) . . . 733
14.11.24 RXONLY_CH CAN Module Bit-Rate Prescaler Register (CnBRP_R) . . . 734
14.11.25 RXONLY_CH CAN Bit Rate Register (CnBTR_R) . . . 735
14.11.26 RXONLY-CH CAN Module Last In-Pointer Register (CnLIPT_R) . . . 736
14.11.27 RXONLY_CH CAN Module Time Stamp Register (CnTS_R) . . . 737
14.11.28 RXONLY_CH CAN Module Bus Selector (CnBSEL_R) . . . 738
14.11.29 Registers of Message Buffers . . . 739
14.11.30 Message Data Byte 0 Register (MDATA0m) . . . 741
14.11.31 Message Data Byte 1 Register (MDATA1m) . . . 741
14.11.32 Message Data Byte 2 Register (MDATA2m) . . . 741
14.11.33 Message Data Byte 3 Register (MDATA3m) . . . 741
14.11.34 Message Data Byte 4 Register (MDATA4m) . . . 741
14.11.35 Message Data Byte 5 Register (MDATA5m) . . . 742
14.11.36 Message Data Byte 6 Register (MDATA6m) . . . 742
14.11.37 Message Data Byte 7 Register (MDATA7m) . . . 742

14.11.38 Message Data Length Code Register (MDLCm) 743
14.11.39 Message Configuration Register (MCONFm) 744
14.11.40 Message Identifier Registers (MIDLm, MIDHm) 746
14.11.41 Message Control Register (MCTRLm) 747

Chapter 15 Port Functions 749
15.1 Features 749
15.2 Port Configuration 750
15.3 Pin Functions of Each Port 759
15.3.1 Port 0 759
15.3.2 Port 1 761
15.3.3 Port 2 764
15.3.4 Port 3 767
15.3.5 Port 4 771
15.3.6 Port 7 774
15.3.7 Port AH 775
15.3.8 Port AL 777
15.3.9 Port CS 779
15.3.10 Port CT 781
15.3.11 Port DL 783

Chapter 16 Flash Memory 785
16.1 Features 785
16.2 Erase Unit 786
16.3 Writing with Flash Programmer 786
16.4 Programming Environment 787
16.5 Communication Mode 788
16.6 Pin Connection 790
16.6.1 MODE0 pin 790
16.6.2 MODE1 pin 791
16.6.3 Serial interface pins 792
16.6.4 $\overline{\text{RESET}}$ pin 794
16.6.5 Port pins (including NMI) 794
16.6.6 Other signal pins 794
16.6.7 Power supply 794
16.7 Programming Method 795
16.7.1 Flash memory control 795
16.7.2 Flash memory programming mode 796
16.7.3 Selecting communication mode 797
16.7.4 Communication commands 798

Appendix A Instruction Set List 801

Appendix B Transition Diagrams RXONLY Channel and DIAG Channel 809

Appendix C Index 813

Appendix D Revision History 819

List of Figures

Figure 1-1: Pin Configuration of the μPD70F3178 ... 28
Figure 1-2: Block Diagram of the μPD70F3178 Microcontroller ... 30
Figure 2-1: Pin I/O Circuits ... 54
Figure 3-1: CPU Register Set ... 56
Figure 3-2: Program Counter (PC) ... 57
Figure 3-3: Interrupt Source Register (ECR) ... 59
Figure 3-4: Program Status Word (PSW) ... 60
Figure 3-5: CPU Address Space ... 63
Figure 3-6: Image on Address Space ... 64
Figure 3-7: Wrap-around of Program Space ... 65
Figure 3-8: Wrap-around of Data Space ... 65
Figure 3-9: Memory Map (μPD70F3178 (A)) ... 66
Figure 3-10: Internal RAM Area ... 69
Figure 3-11: Internal Peripheral I/O Area ... 70
Figure 3-12: Example Application of wrap-around (μPD70F3178) ... 71
Figure 3-13: Programmable Peripheral I/O Register (Outline) ... 185
Figure 3-14: Peripheral Area Selection Control Register (BPC) ... 186
Figure 3-15: Command Register (PRCMD) Format ... 188
Figure 3-16: Internal Peripheral Function Wait Control Register (VSWC) Format ... 189
Figure 4-1: Memory Block Function ... 192
Figure 4-2: Chip Area Select Control Registers 0, 1 (1/2) ... 193
Figure 4-3: Bus cycle configuration registers 0, 1 (BCT0, BCT1) Format (1/2) ... 195
Figure 4-4: Bus Size Configuration Register (BSC) Format ... 197
Figure 4-5: Endian configuration register (BEC) Format ... 198
Figure 4-6: Big Endian Addresses within Word ... 199
Figure 4-7: Little Endian Addresses within Word ... 199
Figure 4-8: Data wait control registers 0, 1 (DWC0, DWC1) Format ... 212
Figure 4-9: Address setup wait control register (ASC) Format ... 213
Figure 4-10: Bus cycle control register (BCC) Format ... 214
Figure 5-1: Block Diagram of the Clock Generator ... 217
Figure 5-2: Clock Control Register (CKC) ... 218
Figure 5-3: PLL Status Register (PSTAT) ... 219
Figure 5-4: PLL Control Register (PLC) ... 220
Figure 5-5: Power Save Mode State Transition Diagram ... 222
Figure 5-6: STOP mode released by $\overline{\text{RESET}}$ input ... 228
Figure 5-7: Power Save Control Register (PSC) ... 229
Figure 5-8: Power Save Mode Register (PSM) ... 230
Figure 6-1: Block Diagram of Timer P ... 233
Figure 6-2: TMPn capture/compare register 0 (TPnCCR0) Format ... 234
Figure 6-3: TMPn capture/compare register 1 (TPnCCR1) Format ... 235
Figure 6-4: TMPn counter read buffer register (TPnCNT) Format ... 236
Figure 6-5: TMPn control register 0 (TPnCTL0) Format (1/2) ... 237
Figure 6-6: TMPn timer control register 1 (TPnCTL1) Format (1/2) ... 239
Figure 6-7: TMPn I/O control register 0 (TPnIOC0) Format ... 241
Figure 6-8: TMPn I/O control register 1 (TPnIOC1) Format ... 242
Figure 6-9: TMPn I/O control register 2 (TPnIOC2) Format ... 243
Figure 6-10: TMPn option register 0 (TPnOPT0) Format ... 244
Figure 6-11: Selector operation control register 0 (SELCNT0) Format ... 245
Figure 6-12: Selector operation control register 1 (SELCNT1) Format ... 246
Figure 6-13: Selector operation control register 2 (SELCNT2) Format ... 247
Figure 6-14: Flowchart of Basic Operation of Anytime Write ... 249
Figure 6-15: Timing Chart of Anytime Write ... 250
Figure 6-16: Flowchart of Basic Operation of Reload ... 251
Figure 6-17: Timing Chart of Reload ... 252
Figure 6-18: Flowchart of Basic Operation in Interval Timer Mode ... 253

Figure 6-19: Timing of Basic Operation in Interval Timer Mode (1/2) 254
Figure 6-20: Flowchart of Basic Operation in External Trigger Pulse Output Mode 257
Figure 6-21: Timing of Basic Operation in External Trigger Pulse Output Mode 258
Figure 6-22: Flowchart of Basic Operation in One-Shot Pulse Mode 260
Figure 6-23: Timing of Basic Operation in One-Shot Pulse Mode 261
Figure 6-24: Flowchart of Basic Operation in PWM Mode (1/2) 263
Figure 6-25: Timing of Basic Operation in PWM Mode (1/2) 265
Figure 6-26: Flowchart of Basic Operation in Free-Running Mode 268
Figure 6-27: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 0) .. 269
Figure 6-28: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 1) .. 270
Figure 6-29: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 1) .. 271
Figure 6-30: Flowchart of Basic Operation in Pulse Width Measurement Mode 272
Figure 6-31: Timing of Basic Operation in Pulse Width Measurement Mode 273
Figure 6-32: Tuned Operation Image (TMP0, TMP1, TMP2) 275
Figure 6-33: Basic Operation Timing of Tuned PWM Function (TMP0, TMP1, TMP2) 276
Figure 7-1: Example of Connection to SRAM 278
Figure 7-2: SRAM, External ROM, External I/O Access Timing (1/4) 279
Figure 7-3: Example of Page ROM Connections 284
Figure 7-4: On-Page/Off-Page Judgment during Page ROM Connection (1/2) 285
Figure 7-5: Page ROM Configuration Register (PRC) 287
Figure 7-6: Page ROM Access Timing (1/4) 288
Figure 8-1: NMI0 request during NMI0 serving 296
Figure 8-2: Processing Configuration of Non-Maskable Interrupt 297
Figure 8-3: RETI Instruction Processing 298
Figure 8-4: Non-maskable Interrupt Status Flag (NP) 299
Figure 8-5: Interrupt Mode Register 3 (INTM3) 299
Figure 8-6: Maskable Interrupt Processing 301
Figure 8-7: RETI Instruction Processing 302
Figure 8-8: Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed (1/2) 304
Figure 8-9: Example of Processing Interrupt Requests Simultaneously Generated 306
Figure 8-10: Interrupt Control Register (xxIC) 307
Figure 8-11: Interrupt Mask Registers 0 to 3 (IMR0 to IMR3) 310
Figure 8-12: In-Service Priority Register (ISPR) 311
Figure 8-13: Maskable Interrupt Status Flag (ID) 312
Figure 8-14: Port Interrupt Input Circuit (P01, P27) 313
Figure 8-15: TimerP0 - 2 Input Circuit (P20-P25) 313
Figure 8-16: TimerP2 TTRGP2 Input Circuit (P26) 313
Figure 8-17: UARTA0-1 Input Circuit (P30, P32) 313
Figure 8-18: NMI Input Circuit (P00) 313
Figure 8-19: Interrupt Mode Register 0 (INTM0) 315
Figure 8-20: Interrupt Mode Register 1 (INTM1) 316
Figure 8-21: Interrupt Mode Register 2 (INTM2) 317
Figure 8-22: Interrupt Mode Register 3 (INTM3) 318
Figure 8-23: Software Exception Processing 319
Figure 8-24: RETI Instruction Processing 320
Figure 8-25: Exception Status Flag (EP) 321
Figure 8-26: Exception Trap Processing 322
Figure 8-27: Restore Processing from Exception Trap 323
Figure 8-28: Pipeline Operation at Interrupt Request Acknowledgment (Outline) 326
Figure 9-1: Block Diagram of A/D Converter 330
Figure 9-2: A/D converter mode register 0 (ADA0M0) Format 334
Figure 9-3: A/D converter mode register 1 (ADA0M1) Format 335
Figure 9-4: A/D converter mode register 2 (ADA0M2) Format 337
Figure 9-5: A/D converter channel specification register 0 (ADA0S) Format 338
Figure 9-6: A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH) Format 339
Figure 9-7: Relationship Between Analog Input Voltage and A/D Conversion Results 340
Figure 9-8: Power-fail compare mode register (ADA0PFM) Format 341

Figure 9-9: Power-fail compare threshold value register (ADA0PFT) Format 342
Figure 9-10: A/D Converter Basic Operation 344
Figure 9-11: Timing Example of Continuous Select Mode Operation (ADA0S = 01H) 346
Figure 9-12: Timing Example of Continuous Scan Mode Operation (ADA0S Register = 03H) 347
Figure 9-13: Timing example of continuous select mode operation (when power-fail comparison is made: ADA0S Register = 01H) 349
Figure 9-14: Timing example of continuous scan mode operation (when power-fail comparison is made: ADA0S Register = 03H) 350
Figure 9-15: Processing of Analog Input Pin 351
Figure 9-16: Generation Timing of A/D Conversion End Interrupt Request 352
Figure 9-17: AV_{DD} Pin Processing Example 353
Figure 9-18: Overall Error 354
Figure 9-19: Quantization Error 355
Figure 9-20: Zero-Scale Error 355
Figure 9-21: Full-Scale Error 356
Figure 9-22: Differential Linearity Error 356
Figure 9-23: Integral Linearity Error 357
Figure 9-24: Sampling Time 357
Figure 10-1: Block Diagram of Asynchronous Serial Interface A 360
Figure 10-2: UARTAn control register 0 (UAnCTL0) Format (1/2) 363
Figure 10-3: UARTAn control register 1 (UAnCTL1) Format 365
Figure 10-4: UARTAn control register 2 (UAnCTL2) Format 366
Figure 10-5: UARTAn option control register 0 (UAnOPT0) Format (1/2) 367
Figure 10-6: UARTAn status register (UAnSTR) Format (1/2) 369
Figure 10-7: UARTAn receive data register (UAnRX) Format 371
Figure 10-8: UARTAn transmit data register (UAnTX) Format 371
Figure 10-9: Format of Transmit/Receive Data of UARTA (1/2) 373
Figure 10-10: Outline of Transmission Operation of LIN 375
Figure 10-11: Outline of Reception Operation of LIN 376
Figure 10-12: SBF Transmission 377
Figure 10-13: SBF Reception 378
Figure 10-14: UART Transmission 379
Figure 10-15: Processing Flow of Continuous Transfer 380
Figure 10-16: Timing of Continuous Transmission Operation 381
Figure 10-17: UART Reception 382
Figure 10-18: Noise Filter Circuit 385
Figure 10-19: Configuration of Baud Rate Generator 386
Figure 10-20: Permissible Baud Rate Range for Reception 389
Figure 10-21: Transfer Rate for Continuous Transmission 391
Figure 11-1: Block Diagram of 3-Wire Serial Interface 394
Figure 11-2: CSIBn control register 0 (CBnCTL0) Format (1/2) 396
Figure 11-3: CSIBn control register 1 (CBnCTL1) Format 398
Figure 11-4: CSIBn control register 2 (CBnCTL2) Format 399
Figure 11-5: CSIBn status register (CBnSTR) Format 400
Figure 11-6: CSIBn receive data register (CBnRX) Format 401
Figure 11-7: CSIBn transmit data register (CBnTX) Format 401
Figure 11-8: Changing Transfer Data Length 402
Figure 11-9: Single Transfer Timing (Master Mode, Transmission/Reception Mode) 404
Figure 11-10: Single Transfer Timing (Master Mode, Reception Mode) 405
Figure 11-11: Continuous Transfer Timing (Master Mode, Transmission/Reception Mode) 406
Figure 11-12: Continuous Transfer Timing (Master Mode, Reception Mode) 407
Figure 11-13: Continuous Transfer Timing (Error) 408
Figure 11-14: Continuous Transfer Timing (Slave Mode, Transmission/Reception Mode) 409
Figure 11-15: Continuous Transfer Timing (Slave Mode, Reception Mode) 410
Figure 11-16: Clock Timing (1/2) 411
Figure 11-17: Single Transmission Flow 414
Figure 11-18: Single Reception Flow (Master) 415
Figure 11-19: Single Reception Flow (Slave) 416

Figure 11-20: Continuous Transmission Flow 417
Figure 11-21: Continuous Reception Flow (Master) 418
Figure 11-22: Continuous Reception Flow (Slave) 419
Figure 11-23: Prescaler mode register 0 (PRSM0) Format 420
Figure 11-24: Prescaler compare register 0 (PRSCM0) Format 421
Figure 12-1: CSIB2 and I²C00 Mode Switch Settings 423
Figure 12-2: Block Diagram of I²C0 425
Figure 12-3: Serial Bus Configuration Example Using I²C Bus 426
Figure 12-4: IIC control registers 0 (IICC0) Format (1/4) 430
Figure 12-5: IIC status registers 0 (IICS0 to IICS2) Format (1/3) 434
Figure 12-6: IIC flag registers 0 (IICF0) Format (1/2) 437
Figure 12-7: IIC clock select registers 0 (IICCL0) Format (1/2) 439
Figure 12-8: IIC function expansion registers 0 (IICX0) Format 440
Figure 12-9: I²C0 transfer clock timing 441
Figure 12-11: IIC division clock select registers 0 (OCKS0) Format 443
Figure 12-12: IIC shift registers 0 (IIC0 to IIC2) Format 444
Figure 12-13: Slave address registers 0 (SVA0) Format 444
Figure 12-14: Pin Configuration Diagram 445
Figure 12-15: I²C Bus Serial Data Transfer Timing 446
Figure 12-16: Start Condition 446
Figure 12-17: Address 447
Figure 12-18: Transfer Direction Specification 448
Figure 12-19: $\overline{\text{ACK}}$ Signal 449
Figure 12-20: Stop Condition 450
Figure 12-21: Wait Signal (1/2) 451
Figure 12-22: Arbitration Timing Example 474
Figure 12-23: Communication Reservation Timing 478
Figure 12-24: Timing for Accepting Communication Reservations 478
Figure 12-25: Communication Reservation Flowchart 479
Figure 12-26: Master Operation Flowchart (1/2) 482
Figure 12-27: Software Outline During Slave Operation 484
Figure 12-28: Slave Operation Flowchart (1/2) 485
Figure 12-29: Example of Master to Slave Communication
when 9-Clock wait Is selected for both Master and Slave) (1/3) 488
Figure 12-30: Example of Slave to Master Communication
(when 9-Clock wait Is selected for both Master and Slave) (1/3) 491
Figure 13-1: Block Diagram of CAN Module 497
Figure 13-2: Composition of Layers 498
Figure 13-3: Data Frame 500
Figure 13-4: Remote Frame 501
Figure 13-5: Start of Frame (SOF) 502
Figure 13-6: Arbitration Field (in Standard Format Mode) 503
Figure 13-7: Arbitration Field (in Extended Format Mode) 503
Figure 13-8: Control Field 504
Figure 13-9: Data Field 505
Figure 13-10: CRC Field 505
Figure 13-11: ACK Field 506
Figure 13-12: End of Frame (EOF) 506
Figure 13-13: Interframe Space (Error Active Node) 507
Figure 13-14: Interframe Space (Error Passive Node) 507
Figure 13-15: Error Frame 508
Figure 13-16: Overload Frame 509
Figure 13-17: Recovery Operation from Bus-off State through Normal Recovery Sequence 516
Figure 13-18: Segment Setting 517
Figure 13-19: Configuration of Data Bit Time Defined by CAN Specification 518
Figure 13-20: Hard-synchronization at Recognition of Dominant Level during Bus Idle 519
Figure 13-21: Re-synchronization 520
Figure 13-22: Connection to CAN Bus 521

Figure 13-23: CAN Global Control Register (CnGMCTRL) Format (1/2) 530
Figure 13-24: CAN Global Clock Selection Register (CnGMCS) Format 532
Figure 13-25: CAN global automatic block transmission control register (CnGMABT) (1/2) 533
Figure 13-26: CAN Global Configuration Register (CnGMCONF) Format 535
Figure 13-27: CAN Global Automatic Block Transmission Delay Register (CnGMABTD) Format . 536
Figure 13-28: CANn Module Mask1 Registers (CnMASK1L, CnMASK1H) Format 537
Figure 13-29: CANn Module Mask2 Registers (CnMASK2L, CnMASK2H) Format 537
Figure 13-30: CANn Module Mask3 Registers (CnMASK3L, CnMASK3H) Format 538
Figure 13-31: CANn Module Mask4 Registers (CnMASK4L, CnMASK4H) Format 538
Figure 13-32: CAN Module Control Register (CnCTRL) Format (1/4) 539
Figure 13-33: CAN Module Last Error Code Register (CnLEC) Format 543
Figure 13-34: CAN Module Information Register (CnINFO) Format 544
Figure 13-35: CAN Module Error Counter Register (CnERC) Format 545
Figure 13-36: CAN Module Interrupt Enable Register (CnIE) Format (1/2) 546
Figure 13-37: CAN Module Interrupt Status Register (CnINTS) Format (1/2) 548
Figure 13-38: CAN Module Bit Rate Prescaler Register (CnBRP) 550
Figure 13-39: CAN Module Bit Rate Register (CnBTR) (1/2) 551
Figure 13-40: CAN Module Last In-Pointer Register (CnLIPT) Format 553
Figure 13-41: CAN Module Receive History List Register (CnRGPT) Format (1/2) 554
Figure 13-42: CAN Module Last Out-Pointer Register (CnLOPT) Format 556
Figure 13-43: CAN Module Transmit History List Register (CnTGPT) Format (1/2) 557
Figure 13-44: CAN Module Time Stamp Register (CnTS) Format (1/2) 559
Figure 13-45: CAN Message Data Byte Register (CnMDATA x m) (x = 0 to 7) Format (1/2) 561
Figure 13-46: CAN Message Data Length Register m (CnMDLCm) Format 563
Figure 13-47: CAN Message Configuration Register (CnMCONFm) Format 564
Figure 13-48: CAN Message ID Register m (CnMIDLm, CnMIDHm) Format 565
Figure 13-49: CAN Message Control Register m (CnMCTRLm) Format (1/3) 566
Figure 13-50: Example of Bit Setting/Clearing Operations 569
Figure 13-51: 16-Bit Data during Write Operation 570
Figure 13-52: Setting Transmission Request (TRQ) to Transmit Message Buffer after Redefining 572
Figure 13-53: Transition to Operation Modes 573
Figure 13-54: Receive History List 576
Figure 13-55: Mask Function Identifier Examples (1/2) 577
Figure 13-56: Message Processing Example 581
Figure 13-57: Transmit History List 584
Figure 13-58: CAN Module Terminal Connection in Receive-Only Mode 592
Figure 13-59: CAN Module Terminal Connection in Self-Test Mode 594
Figure 13-60: Timing Diagram of Capture Signal TSOUT 595
Figure 13-61: Initialization 604
Figure 13-62: Re-initialization 605
Figure 13-63: Message Buffer Initialization 606
Figure 13-64: Message Buffer Redefinition 607
Figure 13-65: Message Buffer Redefinition during Transmission 608
Figure 13-66: Message Transmit Processing 609
Figure 13-67: Message Transmit Processing (Normal Operation Mode with ABT) 610
Figure 13-68: Transmission via Interrupt (Using CnLOPT register) 611
Figure 13-69: Transmit via Interrupt (Using CnTGPT register) 612
Figure 13-70: Transmission via Software Polling 613
Figure 13-71: Transmission Abort Processing (Except Normal Operation Mode with ABT) 614
Figure 13-72: Transmission Abort Processing Except for ABT Transmission (Normal Operation Mode with ABT) 615
Figure 13-73: ABT Transmission Abort Processing (Normal Operation Mode with ABT) 616
Figure 13-74: ABT Transmission Request Abort Processing (Normal Operation Mode with ABT) 617
Figure 13-75: Reception via Interrupt (Using CnLIPT Register) 618
Figure 13-76: Reception via Interrupt (Using CnRGPT Register) 619
Figure 13-77: Reception via Software Polling 620

Figure 13-78: Setting CAN Sleep Mode/Stop Mode 621
Figure 13-79: Clear CAN Sleep/Stop Mode 622
Figure 13-80: Bus-Off Recovery 623
Figure 13-81: Normal Shutdown Process 624
Figure 13-82: Forced Shutdown Process 624
Figure 13-83: Error Handling 625
Figure 13-84: Setting CPU Standby (from CAN Sleep Mode) 626
Figure 13-85: Setting CPU Standby (from CAN Stop Mode) 627
Figure 14-1: Diagnosis concept using DAFCAN and 4 other CAN channels 629
Figure 14-2: Architecture of DIAG Macro 631
Figure 14-3: Register Address Layout 634
Figure 14-4: State transitions of Global Macro switching 635
Figure 14-5: Shutdown Process (Normal Shutdown) 637
Figure 14-6: Shutdown Process (Forced Shutdown) 638
Figure 14-7: DIAG Macro Initialisation 639
Figure 14-8: Re-initialisation of the DIAG-CH CAN Module using 48 buffers 640
Figure 14-9: Re-initialisation of the DIAG-CH CAN Module using TX ID Filter function 641
Figure 14-10: Re-initialisation of the DIAG-CH CAN Module using first 32 buffers only 642
Figure 14-11: CPU write access to a message buffer 644
Figure 14-12: Operation in Mirror or Receive-only Mode of RXONLY-CH 645
Figure 14-13: Operation during INIT or STOP of RXONLY-CH 646
Figure 14-14: Set up of Mirror Mode, Mirror Mode w/ TIF or RECONLY Mode for RXONLY-CH ... 648
Figure 14-15: Cancellation of Mirror Mode or Receive-only Mode for RXONLY_CH 649
Figure 14-16: Transitions for operational modes of the DIAG-CH CAN module 656
Figure 14-17: Transitions for operational modes of the RXONLY_CH CAN module 658
Figure 14-18: Schematic of the CAN macro interrupt generation (1/2) 660
Figure 14-19: Message buffer reading with semaphore handling 663
Figure 14-20: Message-Acceptance-Sorting-Rule 665
Figure 14-21: Message reception procedure using the Reception History List 667
Figure 14-22: ROVF setting depending of the LIPT pointer and the RGPT pointer 668
Figure 14-23: RGPT pointer handling with respect to RHPM bit 669
Figure 14-24: Example on acceptance filtering for a message buffer with a link to a mask 670
Figure 14-25: Transmit preparation 675
Figure 14-26: TOVF setting depending of the LOPT pointer and the TGPT pointer 677
Figure 14-27: TGTP pointer handling with respect to THPM bit 678
Figure 14-28: Transmission Request Abort Process 681
Figure 14-29: Transmission Request Abort Process with ABT mode 683
Figure 14-30: Reception Process of RXONLY 684
Figure 14-31: Receive Operation for RXONLY_CH in RECONLY Mode 687
Figure 14-32: Mirror Mode without Application Communication on DIAG_CH 689
Figure 14-33: Mirror Mode with interleaved Application Communication on DIAG_CH 690
Figure 14-34: Changing the Assignment of the upper 16 buffer 691
Figure 14-35: CAN Global Configuration Register (CnGMCONF) Format 700
Figure 15-1: Port Configuration 750
Figure 15-2: Type A Block Diagram 754
Figure 15-3: Type B Block Diagram 755
Figure 15-4: Type C & D Block Diagram 756
Figure 15-5: Type I Block Diagram 757
Figure 15-6: Type L Block Diagram 758
Figure 15-7: Port 0 (P0) 759
Figure 15-8: Port 0 Mode Register (PM0) 760
Figure 15-9: Port 0 Mode Control Register (PMC0) 760
Figure 15-10: Port 1 (P1) 761
Figure 15-11: Port 1 Mode Register (PM1) 762
Figure 15-12: Port 1 Mode Control Register (PMC1) 763
Figure 15-13: Port 2 (P2) 764
Figure 15-14: Port 2 Mode Register (PM2) 765
Figure 15-15: Port 2 Mode Control Register (PMC2) 766

Figure 15-16: Port 3 (P3) 767
Figure 15-17: Port 3 Mode Register (PM3) 768
Figure 15-18: Port 3 Mode Control Register (PMC3) 769
Figure 15-19: Port 3 Function Control Register (PFC3) 770
Figure 15-20: Port 4 (P4) 771
Figure 15-21: Port 4 Mode Register (PM4) 772
Figure 15-22: Port 4 Mode Control Register (PMC4) 773
Figure 15-23: Port Function Register 7 (P7) 774
Figure 15-24: Port AH (PAH) 775
Figure 15-25: Port AH Mode Register (PMAH) 776
Figure 15-26: Port AH Mode Control Register (PMCAH) 776
Figure 15-27: Port AL (PAL) 777
Figure 15-28: Port AL Mode Register (PMAL) 778
Figure 15-29: Port AL Mode Control Register (PMCAL) 778
Figure 15-30: Port CS (PCS) 779
Figure 15-31: Port CS Mode Register (PMCS) 780
Figure 15-32: Port CS Mode Control Register (PMCCS) 780
Figure 15-33: Port CT (PCT) 781
Figure 15-34: Port CT Mode Register (PMCT) 782
Figure 15-35: Port CT Mode Control Register (PMCCT) 782
Figure 15-36: Port DL (PDL) 783
Figure 15-37: Port DL Mode Register (PMDL) 784
Figure 15-38: Port DL Mode Control Register (PMCDL) 784
Figure 16-1: Environment to Write Program to Flash Memory 787
Figure 16-2: Communication with Dedicated Flash Programmer (UARTA0) 788
Figure 16-3: Communication with Dedicated Flash Programmer (CSIB0) 788
Figure 16-4: Communication with Dedicated Flash Programmer (CSIB0+HS) 789
Figure 16-5: Example of Connection of MODE0 Pin 790
Figure 16-6: Example of Connection of MODE1 Pin 791
Figure 16-7: Signal Conflict (Input Pin of Serial Interface) 792
Figure 16-8: Abnormal Operation of Other Device 793
Figure 16-9: Signal Conflict ($\overline{\text{RESET}}$ Pin) 794
Figure 16-10: Flash Memory Manipulation Procedure 795
Figure 16-11: Flash Memory Programming Mode 796
Figure 16-12: Communication Commands 798

List of Tables

Table 1-1: Product Versions ... 26
Table 2-1: Pin Functions ... 33
Table 2-2: Non-Port Pins ... 36
Table 2-3: Pin Status in Reset and Standby Mode ... 38
Table 2-1: Types of Pin I/O Circuit and Connection of Unused Pins ... 52
Table 3-1: Program Registers ... 57
Table 3-2: System Register Numbers ... 58
Table 3-3: Saturation-Processed Operation Result ... 61
Table 3-4: Operation Modes ... 62
Table 3-5: Interrupt/Exception Table ... 67
Table 3-6: Fixed SFR Table ... 72
Table 3-7: PPA related SFR Table ... 79
Table 3-8: The Values of VSWC Register depending on System Clock ... 190
Table 4-1: Number of Bus Access Clocks ... 196
Table 4-2: Bus Priority Order ... 215
Table 5-1: Power Saving Modes Overview ... 221
Table 5-2: Operating states in HALT mode ... 224
Table 5-3: Operation after HALT mode release by interrupt request ... 225
Table 5-4: Operating States in IDLE Mode ... 226
Table 5-5: Operating States in STOP Mode ... 227
Table 6-1: Configuration of TMP0 to TMP3 ... 232
Table 6-2: TMP Pin List ... 232
Table 6-3: Tuned Operation Mode of Timers ... 274
Table 8-1: Interrupt Vector Table ... 294
Table 8-2: Interrupt Control Register Overview ... 308
Table 8-3: Interrupt Response Time ... 327
Table 9-1: Configuration of A/D Converter (V850E/CG3) ... 331
Table 9-2: Conversion Mode Setting Example ... 336
Table 10-1: List of Pins of Asynchronous Serial Interface A ... 361
Table 10-2: Interrupts and their Default Priority ... 372
Table 10-3: Reception Error Causes ... 383
Table 10-4: Baud Rate Generator Set Data ... 388
Table 10-5: Permissible Maximum/Minimum Baud Rate Error ... 390
Table 11-1: List of 3-Wire Serial Interface Pins ... 395
Table 11-2: Interrupts and their Default Priority ... 403
Table 12-1: Configuration of I²C0 ... 426
Table 12-10: Clock Settings ... 442
Table 12-1: INTIIC0 Generation Timing and Wait Control ... 471
Table 12-2: Extension Code Bit Definitions ... 473
Table 12-3: Status During Arbitration and Interrupt Request Signal Generation Timing ... 475
Table 12-4: Wait Periods ... 477
Table 12-5: Wait Periods ... 480
Table 13-1: CAN Channel Offsets ... 495
Table 13-2: Overview of Functions ... 496
Table 13-3: Frame Types ... 499
Table 13-4: RTR Frame Settings ... 503
Table 13-5: Frame Format Setting (IDE Bit) and Number of Identifier (ID) Bits ... 503
Table 13-6: Data Length Setting ... 504
Table 13-7: Operation in Error Status ... 508
Table 13-8: Definition Error Frame Fields ... 508
Table 13-9: Definition of Overload Frame Fields ... 509
Table 13-10: Determining Bus Priority ... 510
Table 13-11: Bit Stuffing ... 510
Table 13-12: Error Types ... 511
Table 13-13: Output Timing of Error Frame ... 512

Table 13-14: Types of Error States ... 513
Table 13-15: Error Counter ... 514
Table 13-16: Segment Setting ... 517
Table 13-17: Configuration of Data Bit Time Defined by CAN Specification ... 518
Table 13-18: List of CAN Controller Registers ... 522
Table 13-19: CAN Global Register Access Types ... 523
Table 13-20: CAN Module Register Access Types ... 524
Table 13-21: Message Buffer Access Types ... 525
Table 13-22: Bit Configuration of CAN Global Registers ... 526
Table 13-23: Bit Configuration of CAN Module Registers ... 527
Table 13-24: Bit Configuration of Message Buffer Registers ... 529
Table 13-25: Message Reception ... 574
Table 13-26: Message Transmission ... 582
Table 13-27: Transmission Abort ... 586
Table 13-28: List of CAN Module Interrupt Sources ... 591
Table 13-29: Settable Bit Rate Combinations ... 597
Table 13-30: Representative Examples of Baud Rate Settings (f_{CANMOD} = 8 MHz) ... 600
Table 13-31: Representative Examples of Baud Rate Settings (f_{CANMOD} = 16 MHz) ... 602
Table 14-1: Outline of DIAG Macro Functions ... 630
Table 14-2: Minimum configuration of message buffer even when unused in the application ... 643
Table 14-3: Message buffer assignment versus PS/OPMODE ... 646
Table 14-4: ABT message buffer allocation scheme ... 651
Table 14-5: Message buffer configuration with mask link defined by MT[2:0] ... 652
Table 14-6: List of all macro interrupt sources ... 659
Table 14-7: Differences in reception process between regular AFCAN and RXONLY_CH ... 685
Table 14-8: Global Macro Register (CPU read access) ... 693
Table 14-9: Global Macro Register (CPU write access) ... 695
Table 14-10: CAN Module Register (CPU read access) ... 708
Table 14-11: CAN Module Register (CPU write access) ... 710
Table 14-12: CAN Message Buffer Register (CPU read access) ... 739
Table 14-13: CAN Message Buffer Register (CPU write access) ... 740
Table 15-1: Functions of each port ... 751
Table 15-2: Port Pin Functions ... 752
Table 16-1: Signals Generated by Dedicated Flash Programmer (PG-FP4) ... 789
Table 16-2: Relationship Between MODE0 and MODE1 Pins and Operation Mode ... 791
Table 16-3: Pins Used by Each Serial Interface ... 792
Table 16-4: Communication Modes ... 797
Table 16-5: Flash Memory Control Commands ... 798
Table 16-6: Response Commands ... 799
Table B-1: Transition Diagram with upper 16 message buffer assigned to DIAG channel ... 809
Table B-2: Transition Diagram with upper 16 message buffer assigned to DIAG channel ... 810
Table B-3: Transition Diagram with upper 16 message buffer assigned to RXONLY channel ... 811
Table B-4: Transition Diagram with upper 16 message buffer assigned to RXONLY channel ... 812

Chapter 1 Introduction

1.1 General

The CarGate+ Flash microcontroller, is a member of NEC's V850 32-bit RISC family, which match the performance gains attainable with RISC-based controllers to the needs of embedded control applications. The V850 CPU offers easy pipeline handling and programming, resulting in compact code size comparable to 16-bit CISC CPUs.
The CarGate+ offers an excellent combination of general purpose peripheral functions, like serial communication interfaces (UART, clocked SI, I²C) and measurement inputs (A/D converter), with dedicated CAN network support.
The device offers power-saving modes to manage the power consumption effectively under varying conditions.
Thus equipped, the CarGate+ is ideally suited for automotive applications, like CAN gateways. It is also an excellent choice for other applications where a combination of sophisticated peripheral functions and CAN network support is required.

(1) V850E CPU

The V850E CPU supports the RISC instruction set, and through the use of basic instructions that can each be executed in 1-clock period and an optimized pipeline, achieves marked improvements in instruction execution speed. In addition a 32-bit hardware multiplier enables this CPU to support multiply instructions, saturated multiply instructions, bit operation instructions, etc.
Also, through 2-byte basic instructions and instructions compatible with high level languages, etc., object code efficiency in a C compiler is increased, and program size can be made more compact. Further, since the on-chip interrupt controller provides high speed interrupt response, including processing, this device is suited for high level real time control fields.

(2) On-chip flash memory

The µPD70F3178 has on-chip an high speed flash memory, which is able to fetch one instruction within one clock cycle. It is possible to program the user application directly on the target board, on which the µPD70F3178 is mounted. In such case system development time can be reduced and system maintainability after shipping can be markedly improved.

(3) A full range of development environment products

A development environment system that includes an optimized C compiler, debugger, in-circuit emulator, simulator, system performance analyzer, and other elements is also available.

1.2 Device Features

- CPU

  - Core: V850E
  - Number of instructions: 81
  - Min. instruction execution time: 31.2 ns (@ 32 MHz operation, 4.5 V to 5.5 V)
  - General-purpose registers: 32 bits × 32 registers:

- Instruction set:
  - V850E
  - Signed multiplication
    (16 bits × 16 bits → 32 bits or 32 bits × 32 bits → 64 bits): 1 to 2 clocks
  - Saturated operation instructions (with overflow/underflow detection function)
  - 32-bit shift instructions: 1 clock
  - Bit manipulation instructions
  - Load/store instructions with long/short format
  - Signed load instructions

- Internal memory

Table 1-1: Product Versions

Part Number	Main Clock Type	Program Memory	Internal RAM	Full-CAN (2.0b active)
μPD70F3178	32 MHz	256 KB Flash	12 KB	5 Channels

- Flash secure selfprogramming support

- Clock Generator
  - Internal PLL (Peripheral clock supply) 4/8 fold PLL
  - Frequency range: 20-24 to 32 MHz
  - Crystal frequency range: 4 MHz - 6 MHz

- Built-in power saving modes: HALT, IDLE, STOP

- Power supply voltage range $4.3\ V \leq V_{DD} \leq 5.5\ V$

- Temperature range: T_A = -40°C to +85°C @ 32 MHz (A grade)

- I/O lines: 80
  - Input only lines: 4
  - Input/output lines: 76

- A/D Converter: 10-bit resolution; 4 channels
  - Select Mode
  - Scan mode
  - Analog input channels shared with input port functionality

- Serial Interfaces
  - 3-wire mode CSI: 3 channels
  - 2-wire I²C 1 channel (1 × shared with CSI)
  - UART (LIN) mode: 2 channels

- Additional Baud Rate Generator 1 channel

- CAN Interface (2.0B active): 5 channels

- Timers
  - 16-bit multi purpose timer/event counter: 2 channels

- Interrupts
  - Vectored interrupt sources 68
  - Internal interrupt sources 57
  - External interrupt sources 11
  - Non maskable interrupts 1
  - Software exceptions 3
  - Exception trap 1
  - Eight levels of priorities can be set (maskable interrupts)

- Bus control unit:
  - Address/data separated bus 20-bit address // 8 &16-bit data bus
  - Supply voltage power for Bus Interface 5.0 V
  - Chip Select Signals 3

- Package 100-pin plastic LQFP, 0.5 mm pin-pitch, 14 × 14 mm

Note: The CAN macro of this device fulfils the requirements according ISO 11898. Additionally the CAN macro was tested according to the test procedures required by ISO 16845. The CAN macro successfully passed all test patterns. Beyond these test patterns, other tests like robustness tests and processor interface tests as recommended by C&S/FH Wolfenbuettel have successfully been issued.

1.3 Application Fields

The V850E/CG3 CarGate+ is ideally tailored to automotive applications, like gateway applications. It is also an excellent choice for other applications where a combination of sophisticated peripheral functions and CAN network support is required.

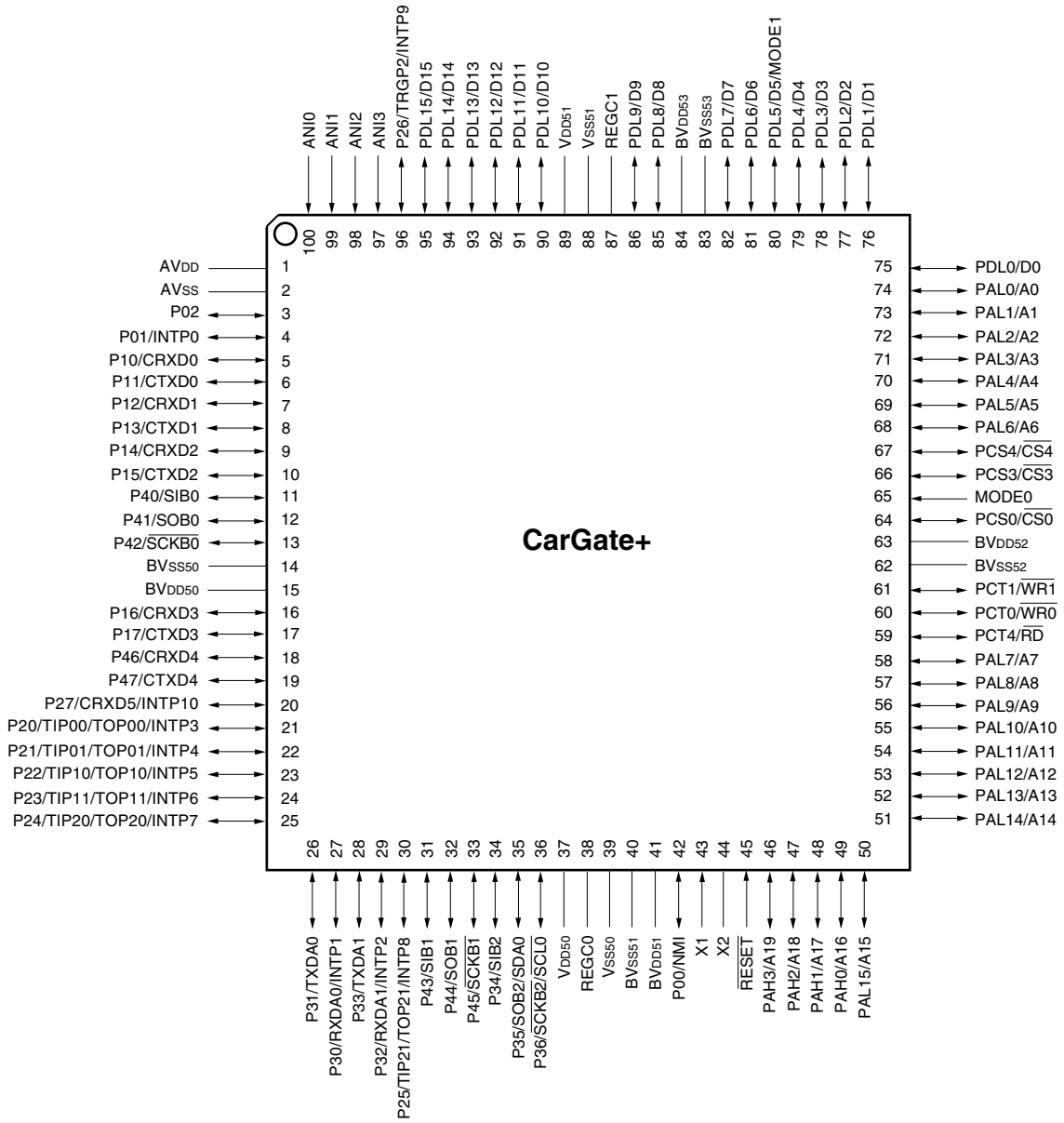
1.4 Ordering Information

Part number	Package	Internal ROM
μPD70F3178	100-pin plastic LQFP, 0.5 mm pin-pitch, 14 × 14 mm	Flash memory

1.5 Pin Configuration (Top View)

100-pin LQFP (0.5 mm pin pitch) (14 × 14 mm)

Figure 1-1: Pin Configuration of the μPD70F3178



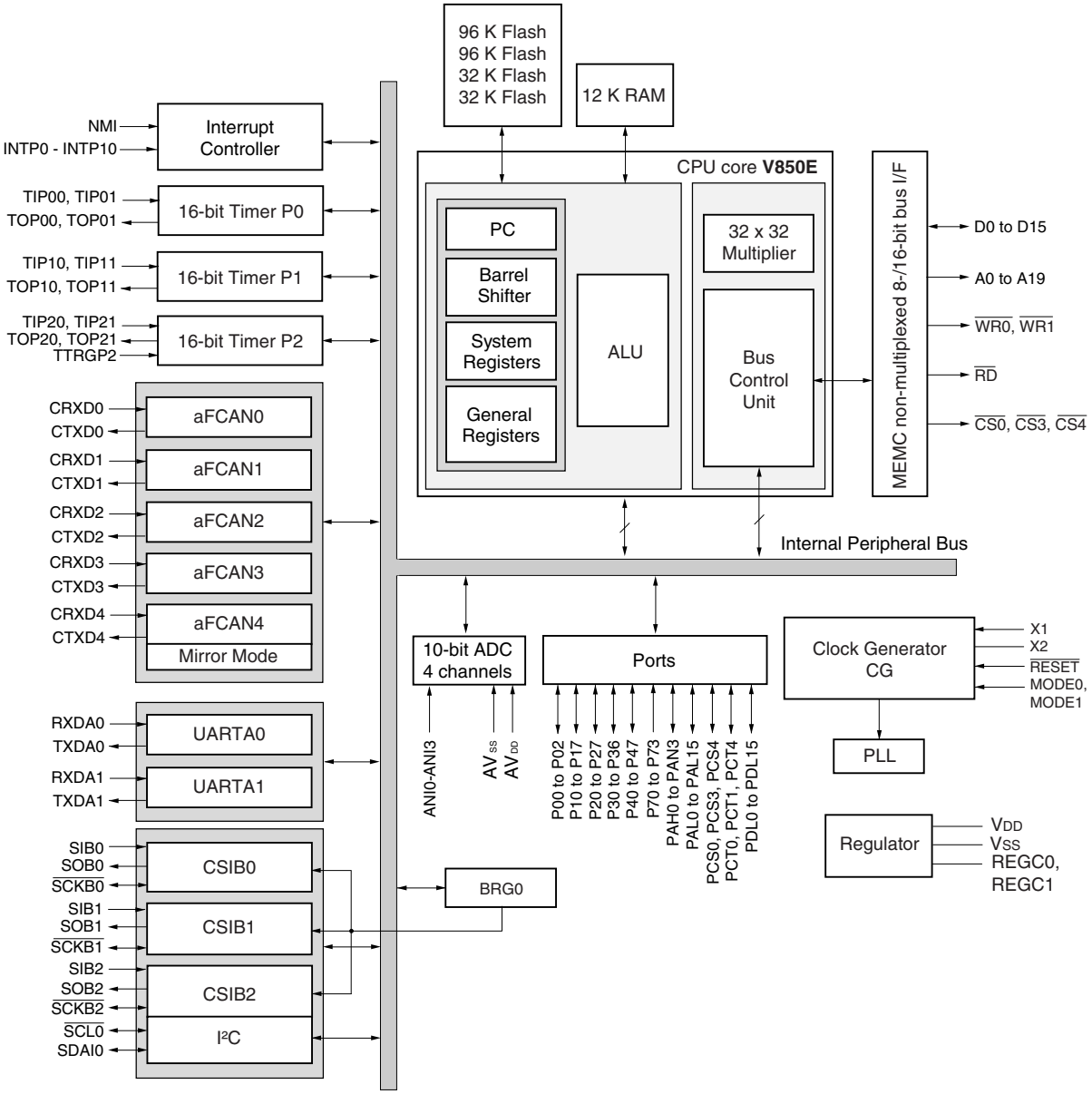
Pin Identification

A0 to A19	: Address Bus	PAL0 to PAL 15	Port AL
D0 to D15	: Data Bus	PCS0, PCS3, PCS4	: Port CS
ANI0 to ANI3	: Analog Input	PCT0, PCT1, PCT4	: Port CT
AV_{DD}	: Analog Power Supply	PDL0 to PDL15	Port PDL
AV_{SS}	: Analog Ground	$\overline{RESET}$	: Reset
CRXD0 to CRXD5	: CAN Receive Line Input	RXDA0 to RXDA1	: UART Receive Data Input
CTXD0 to CTXD4	: CAN Transmit Line Output	$\overline{SCKB0}$, $\overline{SCKB1}$, $\overline{SCKB2}$	: Serial Clock
CV_{DD}	: Clock Generator Power Supply	$\overline{SCL}$	I²C Clock
CV_{SS}	: Clock Generator Ground	SDA	I²C Data
$BVSS_{50}$ to $BVSS_{53}$	Ground for 5 V Power Supply	SIB0, SIB1, SIB2	: Serial Input
VSS_{50} to VSS_{51}	Ground for 5 V Power Supply	SOB0, SOB1, SOB2	: Serial Output
INTP0 to INTP10	External interrupt request	TIP00 to TIP01, TIP10 to TIP11, TIP20 to TIC21	: Timer Input
INTPn0, INTPn5, INTP2n	: Interrupt Request from Peripherals	TOP00 to TOP01, TOP10 to TOP11, TOP20 to TOP21	Timer Output
MODE0, MODE1	: Mode Inputs	TXDA0 to TXDA1	: Transmit Data Output
NMI	: Non-Maskable Interrupt Request	TTRGP2	Timer Trigger Input
P00 to P02	Port 0	BV_{DD50} to BV_{DD53}	: 5 V Power Supply
P10 to P17	: Port 1	V_{DD50} to V_{DD51}	: 5 V Power Supply
P20 to P27	: Port 2	$\overline{WR0}$, $\overline{WR1}$	Write Enable
P30 to P36	: Port 3	$\overline{RD}$	: Read
P40 to P47	: Port 4	$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$	Chip Select
P70 to P73	Port 7	X1, X2	: Crystal (Main-OSC)
PAH0 to PAH3	: Port AH		

1.6 Configuration of Function Block

1.6.1 Block Diagram of µPD70F3178

Figure 1-2: Block Diagram of the µPD70F3178 Microcontroller



1.6.2 On-chip units

(1) CPU

The CPU uses five-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.
Other dedicated on-chip hardware, such as the multiplier (16 bits × 16 bits → 32 bits or 32 bits × 32 bits → 64 bits) and the barrel shifter (32 bits), help accelerate processing of complex instructions.

(2) ROM

The μPD70F3178 has an on-chip flash memory of 256 kBytes.
During instruction fetch, flash memory can be accessed from the CPU in 1-clock cycles.
The memory mapping is done from address 00000000H.

(3) RAM

The RAM area is mapped from address FFFFC000H for the μPD70F3178.
Data in RAM can be accessed from the CPU in 1-clock cycles.

(4) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP0 to INTP10) from on-chip peripheral I/O and external hardware. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiple-interrupt servicing control can be performed for interrupt sources.

(5) Clock generator (CG)

This clock generator supplies frequencies for the CPU and the built-in peripherals. As the input clock, an external crystal is connected to pins X1 and X2.

(6) Serial interface (SIO)

A 2-channel asynchronous serial interface (UART), 3-channel clocked serial interface (CSI), 2-channel clocked serial interface (I^2C) shared with one CSI and 5-channel FCAN are provided as serial interface.

UART transfers data by using the TXDn and RXDn pins. (n = 0 - 1)
CSI transfers data by using the SOn, SIn, and SCKn pins. (n = 0 - 2)
I^2C transfer data by using the SDA and SCL pins.
FCAN performs data transfer using CTXDn and CRXDn pins. (n = 0 - 4)

(7) Real time pulse unit

This unit has 3 channels of 16-bit multi purpose timer/ever, and it is possible to measure pulse widths or frequency and to output a programmable pulse.

(8) A/D converter (ADC)

One high-resolution 10-bit A/D converter, it includes 4 analog input pins. Conversion uses the successive approximation method.

(9) Ports

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

Port	Port Function	Control Function
Port 0	3-bit input/output	NMI, external interrupt
Port 1	8-bit input/output	Serial interface input/output
Port 2	8-bit input/output	Real-time pulse unit input/output, external interrupt input, PWM output, Timer trigger input, serial interface input
Port 3	7-bit input/output	Serial interface input/output
Port 4	8-bit input/output	Serial interface input/output
Port DL	16-bit input/output	External Data input/output
Port AL	16-bit input/output	External address bus
Port AH	4-bit input/output	External address bus
Port CS	3-bit input/output	External bus interface control signal output ($\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$)
Port CT	3-bit input/output	External bus interface control signal output ($\overline{WR0}$, $\overline{WR1}$, $\overline{RD}$)

Chapter 2 Pin Functions

2.1 List of Pin Functions

The names and functions of this product's pins are listed below. These pins can be divided into port pins and non-port pins according to their functions.

(1) Port pins

Table 2-1: Pin Functions (1/3)

Port	I/O	Function	Alternate
P00	I/O	Port 0 3-bit input/output port	NMI (I)
P01			INTP0 (I)
P02			
P10	I/O	Port 1 8-bit input/output port	CRXD0 (I)
P11			CTXD0 (O)
P12			CRXD1 (I)
P13			CTXD1 (O)
P14			CRXD2 (I)
P15			CTXD2 (O)
P16			CRXD3 (I)
P17			CTXD3 (O)
P20	I/O	Port 2 8-bit input/output port	TOP00 (O), TIP00 (I), INTP3 (I)
P21			INTP4 (I), TOP01 (O), TIP01 (I)
P22			INTP5 (I), TOP10 (O), TIP10 (I)
P23			INTP6 (I), TOP11 (O), TIP11 (I)
P24			INTP7 (I), TOP20 (O), TIP20 (I)
P25			INTP8 (I), TOP21 (O), TIP21 (I)
P26			TTRGP2 (I), INTP9 (I)
P27			INTP10 (I)
P30	I/O	Port 3 7-bit input/output port	RXDA0 (I), INTP1 (I)
P31			TXDA0 (O)
P32			RXDA1 (I), INTP2 (I)
P33			TXDA1 (O)
P34			SIB2 (I)
P35			SDAO0 (O), SDAI0 (I), SOB2 (O)
P36			$\overline{\text{SCLO0}}$ (O), $\overline{\text{SCKIB2}}$ (I), SCLI0 (I), $\overline{\text{SCKOB2}}$ (O)

Table 2-1: Pin Functions (2/3)

Port	I/O	Function	Alternate
P40	I/O	Port 4 8-bit input/output port	SIB0 (I)
P41			SOB0 (O)
P42			$\overline{\text{SCKIB0}}$ (I), $\overline{\text{SCKOB0}}$ (O)
P43			SIB1 (I)
P44			SOB1 (O)
P45			$\overline{\text{SCKIB1}}$ (I), $\overline{\text{SCKOB1}}$ (O)
P46			CRXD4 (I)
P47			CTXD4 (O)
P70	I	Port 7 4-bit input port	ANI0
P71			ANI1
P72			ANI2
P73			ANI3
PAH0	I/O	Port PAH 4-bit input/output	A16 (O)
PAH1			A17 (O)
PAH2			A18 (O)
PAH3			A19 (O)
PAL0	I/O	Port PAL 16-bit input/output	A0 (O)
PAL1			A1 (O)
PAL2			A2 (O)
PAL3			A3 (O)
PAL4			A4 (O)
PAL5			A5 (O)
PAL6			A6 (O)
PAL7			A7 (O)
PAL8			A8 (O)
PAL9			A9 (O)
PAL10			A10 (O)
PAL11			A11 (O)
PAL12			A12 (O)
PAL13			A13 (O)
PAL14			A14 (O)
PAL15			A15 (O)
PCS0	I/O	Port PCS 3-bit input/output	$\overline{\text{CS0}}$ (O)
PCS3			$\overline{\text{CS3}}$ (O)
PCS4			$\overline{\text{CS4}}$ (O)
PCT0	I/O	Port PCT 3-bit input/output	$\overline{\text{WR0}}$ (O)
PCT1			$\overline{\text{WR1}}$ (O)
PCT4			$\overline{\text{RD}}$ (O)

Table 2-1: Pin Functions (3/3)

Port	I/O	Function	Alternate
PDL0	I/O	Port PDL 16-bit input/output	DO0 (O), DI0 (I)
PDL1			DO1 (O), DI1 (I)
PDL2			DO2 (O), DI2 (I)
PDL3			DO3 (O), DI3 (I)
PDL4			DI4 (I), DO4 (O)
PDL5			DI5 (I), DO5 (O), MODE1
PDL6			DI6 (I), DO6 (O)
PDL7			DI7 (I), DO7 (O)
PDL8			DO8 (O), DI8 (I)
PDL9			DO9 (O), DI9 (I)
PDL10			DI10 (I), DO10 (O)
PDL11			DI11 (I), DO11 (O)
PDL12			DO12 (O), DI12 (I)
PDL13			DO13 (O), DI13 (I)
PDL14			DI14 (I), DO14 (O)
PDL15			DI15 (I), DO15 (O)

(2) Non-port pins

Table 2-2: Non-Port Pins (1/2)

Pin Name	I/O	Function	Port
A0 - A15	O	Address bus of external bus	PAL0 - PAL15
A16 - A19	O		PAH0 - PAH3
ANI0 - ANI3	I	Analog input for A/D converter	P70 - P73
CRXD0	I	Serial receive data input for aFCAN0 to aFCAN4	P10
CRXD1	I		P12
CRXD2	I		P14
CRXD3	I		P16
CRXD4	I		P46
$\overline{CS0}$	O	Chip select output for external bus	PCS0
$\overline{CS3}$	O		PCS3
$\overline{CS4}$	O		PCS4
CTXD0	O	Serial transmit data for aFCAN0 to aFCAN4	P11
CTXD1	O		P13
CTXD2	O		P15
CTXD3	O		P17
CTXD4	O		P47
D0 - D15	I/O	Data bus of external bus	PDL0 - PDL15
INTP0	I	External interrupt request	P01
INTP1	I		P30
INTP2	I		P32
INTP3	I		P20
INTP4	I		P21
INTP5	I		P22
INTP6	I		P23
INTP7	I		P24
INTP8	I		P25
INTP9	I		P26
INTP10	I		P27
NMI	I	Non maskable interrupt	P00
$\overline{RD}$	O	Read strobe signal	PCT4
RXDA0	I	Serial receive data UARTA0 & UARTA1	P30
RXDA1	I		P32
$\overline{SCKB0}$	I/O	Serial clock I/O from CSIB0 - CSIB2	P42
$\overline{SCKB1}$	I/O		P45
$\overline{SCKB2}$	I/O		P36
$\overline{SCL0}$	I/O	Serial clock line I²C	P36
SDAI0	I/O	Serial data line I²C	P35
SIB0	I	Serial data input CSIB0 - CSIB2	P40
SIB1	I		P43
SIB2	I		P34

Table 2-2: Non-Port Pins (2/2)

Pin Name	I/O	Function	Port
SOB0	O	Serial data output CSIB0- CSIB2	P41
SOB1	O		P44
SOB2	O		P35
TIP00	I	Capture input 0-1 TimerP0 - TimerP2	P20
TIP01	I		P21
TIP10	I		P22
TIP11	I		P23
TIP20	I		P24
TIP21	I		P25
TOP00	O	Compare output 0-1 Timer P0 - Timer P2	P20
TOP01	O		P21
TOP10	O		P22
TOP11	O		P23
TOP20	O		P24
TOP21	O		P25
TTRGP2	I	Timer Trigger Input Timer P2	P26
TXDA0	O	Serial transmit data output UARTA0 - UARTA1	P31
TXDA1	O		P33
$\overline{WR0}$	O	Write strobe signal for external bus	PCT0
$\overline{WR1}$	O		PCT1
AV_{DD}	–	5 V power supply ADC	–
AV_{SS}	–	GND potential for 5 V power supply ADC	–
V_{DD50}-V_{DD51}	–	5 V power supply[Note]	–
BV_{DD50}-BV_{DD53}	–		–
V_{SS50}-V_{SS50}	–	GND potential for 5 V power supply	–
BV_{SS50}-BV_{SS53}	–		–
MODE0	I	Specify Operation Mode	–
MODE1	I		PDL5
REGC0	–	Connection of regulator stabilization capacitance	–
REGC1	–		–
$\overline{RESET}$	I	System reset input	–
X1	I	Connection of external oscillator	–
X2	0		–

Note: All V_{DD5} pins have to be connected to each other. On each pin of V_{DD5}, a capacitor containing a very low serial impedance has to be attached as tight as possible to the pin.

(3) Pin status in RESET and STANDBY mode

Table 2-3: Pin Status in Reset and Standby Mode

Operating Status / Pin	RESET	STOP	IDLE	HALT
D0 to D15	Hi-Z[Note 1]	Hold[Note 2]	Hold[Note 2]	Hold[Note 2]
A0 to A19				
$\overline{CS4}$, $\overline{CS3}$, $\overline{CS0}$				
$\overline{WR0}$, $\overline{WR1}$				
$\overline{RD}$				
TIC01 to TIC00				
INTP05 to INTP00				
INTP15 to INTP10				
INTP21 to INTP20				
INTP5 to INTP0				
NMI				
TOG04 to TOG01				
TOG14 to TOG11				
TOC0				
SO02, SO01, SO00				
SI02, SI01, SI00				
$\overline{SCK2}$, $\overline{SCK1}$, $\overline{SCK0}$				
RXD51 to RXD50				
TXD51 to TXD50				
CRXD0 to CRXD5				
CTXD0 to FCTXD5				
ANI3 to ANI0				
P1, P2, P3, P4, P7				
PAH7 to PAH0				
PCS4, PCS3, PCS0				
PCT4, PCT1, PCT0				

Notes: 1. PDL0-7 are input if MODE0 is High

2. Due to the fact that there is no bus access during stop mode, the bus IF pins are Hi-Z

Remarks: 1. N.A. : This configuration is not available

2. -- : Input data is not sampled

3. 1 : During output / input

4. Hi-Z : High Impedance

2.2 Description of Pin Functions

(1) P00 to P02 (Port 0) … Input/output

Port 0 is a 3-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, P00 to P02 operate as interrupt (NMI, INTP0) input.
An operation mode of port or control mode can be selected for each bit and specified by the port 0 mode control register (PMC0).

(a) Port mode
P00 to P02 can be set to input or output in 1-bit units using the port 1 mode register (PM0).

(b) Control mode
P00 to P02 can be set to port or control mode in 1-bit units using PMC0.

(c) NMI (Non Maskable Interrupt) … Input
This pin inputs non maskable interrupts.

(d) INTP0 (Interrupt) … Input
This pin inputs maskable interrupts.

(2) P10 to P17 (Port 1) … Input/output

Port 1 is an 8-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, P10 to P17 operate as the serial interface (CAN0, CAN1, CAN2, CAN3) input/output.
An operation mode of port or control mode can be selected for each bit and specified by the port 1 mode control register (PMC1).

(a) Port mode

P10 to P17 can be set to input or output in 1-bit units using the port 1 mode register (PM1).

(b) Control mode

P10 to P17 can be set to port or control mode in 1-bit units using PMC1.

(c) CTXD0 to CTXD3 (Transmit data for controller area network) … Output

This pin outputs CAN serial transmit data.

(d) CRXD0 to CRXD3 (Receive data for controller area network) … Input

This pin inputs CAN serial receive data.

(3) P20 to P27 (Port 2) … Input/output

Port 2 is an 8-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as an input/output port, in control mode, P20 to P27 operate as real time pulse unit (RPU) input/output or external interrupt request input.
An operation mode of port or control mode can be selected for each bit and specified by the port 2 mode control register (PMC2).

(a) Port mode

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

(b) Control mode

P20 to P27 can be set to port or control mode in 1-bit units using PMC2.

(c) TOP00,TOP01 to TOP20,TOP21 (Timer output) … Output

These pins output Timer P 0 to Timer P 2 pulse signals.

(d) TIP00, TIP01 to TIP20, TIP21 (Timer input) … Input

These pins input Timer P0 to Timer P1 capture trigger inputs.

(e) TTRGP2 (Timer trigger input) … Input

This pin input Timer P2 trigger.

(f) INTP3 to INTP10 (External interrupt input) … Input

These pin input external interrupt request signals.

(g) CRXD5 (Receive data for controller area network) … Input

This pin inputs CAN serial receive data (for Mirror Mode only).

(4) P30 to P36 (Port 3) … Input/output

Port 3 is a 7-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as an input/output port, in control mode, P30 to P36 operate as serial interface (UART, CSI, I²C) input/output and external interrupt request input.
An operation mode of port or control mode can be selected for each bit and specified by the port 3 mode control register (PMC3).

(a) Port mode
P30 to P36 can be set to input or output in 1-bit units using the port 3 mode register (PM3).

(b) Control mode
P30 to P36 can be set to port or control mode in 1-bit units using PMC3.

(c) RXDA0 and RXDA1 (UART input) … Input
These pins input serial receive data of UART0 and UART1.

(d) TXDA0 and TXDA1 (UART output) … Output
These pins output serial receive data of UART0 and UART1.

(e) INPT1, INTP2 (Interrupt request from peripherals) … Input
These are external interrupt request input pins.

(f) SIB2 (CSIB2 input) … Input
This pin input serial receive data of CSIB2.

(g) SOB2 (CSIB2 output) … Output
This pin input serial send data of CSIB2.

(h) $\overline{\text{SCKB2}}$ (CSIB2 clock) … Input/Output
This pin is the CSIB2 serial clock input/output.

(i) SDA0 (I²C data) … Input/Output
This pin is the I²C0 serial data input/output.

(j) $\overline{\text{SCL}}$ (I²C clock) … Input/Output
This pin is the I²C0 serial clock input/output.

(5) P40 to P47 (Port 4) … Input/output

Port 4 is a 8-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as an input/output port, in control mode, P40 to P47 operate as serial interface (CSI, CAN) input/output.
An operation mode of port or control mode can be selected for each bit and specified by the port 4 mode control register (PMC4).

(a) Port mode

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

(b) Control mode

P40 to P47 can be set to port or control mode in 1-bit units using PMC4.

(c) SIB0 and SIB1 (CSI input) … Input

These pins input serial data of CSIB0 and CSIB1.

(d) SOB0 and SOB1 (CSI output) … Output

These pins output serial data of CSIB0 and CSIB1.

(e) $\overline{\text{SCKB0}}$ and $\overline{\text{SCKB1}}$ (CSI clock) … Input/Output

This pin is the CSIB0 and CSIB1 serial clock input/output.

(f) CTXD4 (Transmit data for controller area network) … Output

This pin outputs CAN serial transmit data.

(g) CRXD4 (Receive data for controller area network) … Input

This pin inputs CAN serial receive data.

(6) P70 to P73 (Port 7) … Input

Port 7 is an 4-bit input-only port in which all pins are fixed as input pins. P70 to P73 can function as input ports and as analog input pins for the A/D converter in control mode. However, they cannot be switched between these input port and analog input pin.

(a) Port mode

P70 to P73 are input-only pins.

(b) Control mode

P70 to P73 also function as pins ANI0 to ANI3, but these alternate functions are not switchable.

(c) ANI0 to ANI3 (Analog Input 0 to 3)

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

(7) PAH0 to PAH3 (Port AH) … Input/output

Port AH is an 4-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, this port operates as the address bus (A16 to A19) for when memory is accessed externally.
An operation mode of port or control mode can be selected for each bit and specified by the port AH mode control register (PMCAH).

(a) Port mode

PAH0 to PAH3 can be set to input or output in 1-bit units using the port AH mode register (PMAH).

(b) Control mode

PAH0 to PAH3 can be used as A16 to A19 by using PMCAH.

(c) A16 to A20 (Address) … Output

This pin outputs the upper 4-bit address of the 20-bit address in the address bus on an external access.

(8) PAL0 to PAL15 (Port AL) … Input/output

Port AL is an 16-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, this port operates as the address bus (A0 to A15) for when memory is accessed externally.
An operation mode of port or control mode can be selected for each bit and specified by the port AL mode control register (PMCAL).

(a) Port mode

PAL0 to PAL15 can be set to input or output in 1-bit units using the port AL mode register (PMAL).

(b) Control mode

PAL0 to PAL15 can be used as A0 to A15 by using PMCAL.

(c) A0 to A15 (Address) … Output

This pin outputs the lower 16-bit address of the 20-bit address in the address bus on an external access.

(9) PCS0, PCS3, PCS4 (Port CS) … Input/output

Port CS is a 3-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, it operates as a chip-select control signal output when memory is accessed externally.
An operation mode of port or control mode can be selected for each bit and specified by the port CS mode control register (PMCCS).

(a) Port mode

PCS0, PCS3, PCS4 can be set to input or output in 1-bit units using the port CS mode register (PMCS).

(b) Control mode

PCS0, PCS3, PCS4 can be used as CS0, CS3, CS4 by using PMCCS.

(c) $\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$ (Chip select) … Output

This is the chip select signal for external SRAM, external ROM, or external peripheral I/O.
The signal CSn is assigned to memory block n (n = 0, 3, 4).
This is active for the period during which a bus cycle that accesses the corresponding memory block is activated.
It is inactive in an idle state (TI).

(10) PDL0 to PDL15 (Port DL) … Input/output

Port DL is an 16-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, this port operates as the data bus (D0 to D15) when memory is accessed externally.
An operation mode of port or control mode can be selected for each bit and specified by the port DL mode control register (PMCDL).
During reset PDL5 is mode select input pin.

(a) Port mode

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

(b) Control mode

PDL0 to PDL15 can be used as D0 to D15 by using PMCDL.

(c) D0 to D15 (Data) … Output

These pins are the 16-bit data input/output of an external access.

(d) PDL5 (MODE1) … Input

This pin is mode select input during reset.

(11) PCT0, PCT1, PCT4 (Port CT) … Input/output

Port CT is a 3-bit input/output port in which input or output can be set in 1-bit units.
Besides functioning as a port, in control mode, it operates as control signal output when memory is accessed externally.
An operation mode of port or control mode can be selected for each bit and specified by the port CT code control register (PMCCT).

(a) Port mode

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

(b) Control mode

PCT0, PCT1, PCT4 can be used as $\overline{\text{WR0}}$, $\overline{\text{WR1}}$, $\overline{\text{RD}}$ by using PMCCT.

(c) $\overline{\text{WR0}}$ (Lower byte write strobe) … Output

This is a strobe signal that shows that the executing bus cycle is a write cycle for SRAM, external ROM, or an external peripheral I/O area.

(d) $\overline{\text{WR1}}$ (Upper byte write strobe) … Output

This is a strobe signal that shows that the executing bus cycle is a write cycle for SRAM, external ROM, or an external peripheral I/O area.

(e) $\overline{\text{RD}}$ (Read strobe) … Output

This is a strobe signal that shows that the executing bus cycle is a read cycle for SRAM, external ROM, or external peripheral I/O. It is inactive in an idle state (TI).

(12) ANI0 to ANI3 (Analog input) … Input

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

(13) MODE0 and MODE1 (Mode) … Input

These are the input pins that specify the operation mode. Operation modes are broadly divided into normal operation modes and flash memory programming mode. The operation mode is determined by sampling the status of each of pins MODE0 to MODE1 on a reset.
Fix MODE so that the input level does not change during operation. MODE1 is shared with PDL5 functionality.

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

$\overline{\text{RESET}}$ input is asynchronous input. When a signal having a certain low level width is input in asynchronous with the operation clock, a system reset that takes precedence over all operations occurs.
Besides a normal initialize or start, this signal is also used to release a standby mode (HALT, IDLE, WATCH, Sub-Watch, software STOP).

(15) NMI (NON-Maskable Interrupt Request)... input

This is the non-maskable interrupt request input pin.

(16) X1, X2 (Crystal)

These pins connect a resonator for system main-clock generation.

(17) BV_{DD50} to BV_{DD53} (Power supply for peripherals)

These are the positive power supply pins for peripheral I/Os.

(18) BV_{SS50} to BV_{SS53} (Ground)

These are the ground pins for peripheral I/Os.

(19) V_{DD50} to V_{DD51} (Power supply)

These are the positive 5 V power supply pins for the voltage regulators.

(20) V_{SS50} to V_{SS51} (Ground)

These are the ground pins for the voltage regulators.

(21) AV_{DD} (Analog power supply)

This is the analog positive power supply pin for the A/D converter.

(22) AV_{SS} (Analog ground)

This is the ground pin for the A/D converter.

(23) A0 to A19 (Address output) … Output

These pins are the address output pins.

(24) D0 to D15 (Data input/output) … Input/output

These pins are the data input/output pins.

(25) REGC0 and REGC1 () … Input/output

These pins are the voltage regulator capacitance input/output pins.

2.3 Types of Pin I/O Circuit and Connection of Unused Pins

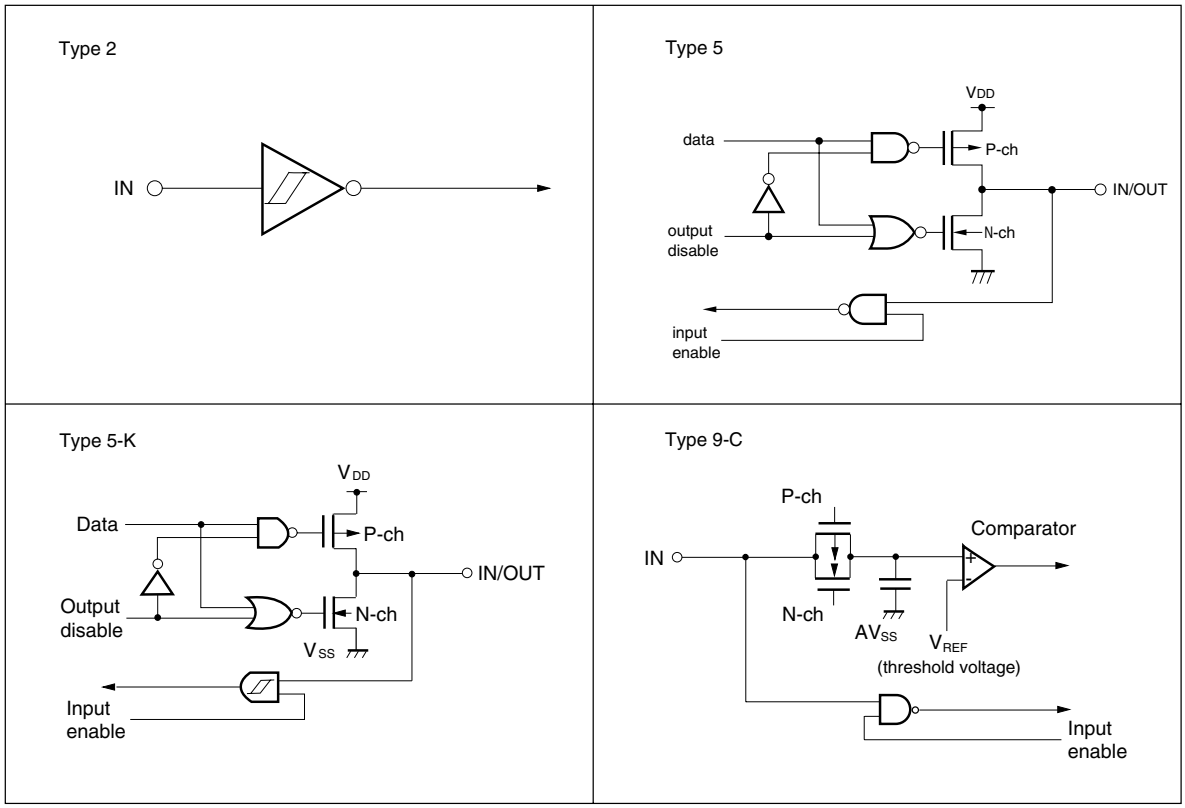
Table 2-1: Types of Pin I/O Circuit and Connection of Unused Pins (1/2)

Pin			I/O Circuit Type	Recommended connection
P00	NMI		5-K	For input: individually connect to V_{DD5} or V_{SS5} via a resistor. For output: leave open.
P01				
P02	INTP0			
P10	CRXD0		5-K	
P11	CTXD0			
P12	CRXD1			
P13	CTXD1			
P14	CRXD2			
P15	CTXD2			
P16	CRXD3			
P17	CTXD3			
P20	TXP00	INTP3	5-K	
P21	TXP01	INTP4		
P22	TXP10	INTP5		
P23	TXP11	INTP6		
P24	TXP20	INTP7		
P25	TXP21	INTP8		
P26	TTRGP2	INTP9		
P27	CRXD5	INTP10		
P30	RXDA0	INTP1	5-K	
P31	TXDA0			
P32	RXDA1	INTP2		
P33	TXDA1			
P34	SIB2			
P35	SOB2	SDA0		
P36	$\overline{\text{SCKB2}}$	$\overline{\text{SCL0}}$		
P40	SIB0		5-K	
P41	SOB0			
P42	$\overline{\text{SCKB0}}$			
P43	SIB1			
P44	SOB1			
P45	$\overline{\text{SCKB1}}$			
P46	CRXD4			
P47	CTXD4			
P70	ANI0		9-C	For input: individually connect to V_{DD5} or V_{SS5} via a resistor.
P71	ANI1			
P72	ANI2			
P73	ANI3			

Table 2-1: Types of Pin I/O Circuit and Connection of Unused Pins (2/2)

Pin			I/O Circuit Type	Recommended connection
PAH0	A16		5	For input: individually connect to V_{DD5} or V_{SS5} via a resistor. For output: leave open.
PAH1	A17			
PAH2	A18			
PAH3	A19			
PCS0	$\overline{CS0}$		5	For input: individually connect to V_{DD5} or V_{SS5} via a resistor. For output: leave open.
PCS3	$\overline{CS3}$			
PCS4	$\overline{CS4}$			
PCT0	$\overline{WR0}$		5	For input: individually connect to V_{DD5} or V_{SS5} via a resistor. For output: leave open.
PCT1	$\overline{WR0}$			
PCT4	$\overline{RD}$			
PDL0	D0		5	
PDL1	D1			
PDL2	D2			
PDL3	D3			
PDL4	D4			
PDL5	D5	MODE1		
PDL6	D6			
PDL7	D7			
PDL8	D8			
PDL9	D9			
PDL10	D10			
PDL11	D11			
PDL12	D12			
PDL13	D13			
PDL14	D14			
PDL15	D15			
ANI0-ANI3			9-C	
MODE0			2	
MODE1	PDL5	D5	5	connect to V_{SS5X} via a resistor.
$\overline{RESET}$			2	-
X1			-	Please refer to the datasheet
X2			-	
AV_{DD}			-	V_{DD5X}
AV_{SS}			-	V_{SS5X}
A0 to A19			5	
D0 to D15			5	

Figure 2-1: Pin I/O Circuits



Chapter 3 CPU Function

The CPU of the CarGate+ is based on a RISC architecture and executes almost all the instructions which can be accessed from the Flash in one clock cycle, using a 5-stage pipeline control.

3.1 Features

- Minimum instruction cycle: 31.25 ns (@ internal 32 MHz operation)
- Memory space
  - Program space: 64 MB linear
  - Data space: 4 GB linear
- Thirty-two 32-bit general registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- One-clock 32-bit shift instruction (barrel shifter)
- Long/short instruction format
- Four types of bit manipulation instructions
  - Set
  - Clear
  - Not
  - Test

3.2 CPU Register Set

The registers of the μPD70F3178 CarGate+ can be classified into two categories: a general program register set and a dedicated system register set. All the registers are 32-bit width. For details, refer to V850E User's Manual Architecture.

Figure 3-1: CPU Register Set

(1) Program register set

31 0

r0	(Zero Register)
r1	(Reserved for Assembler)
r2	(Interrupt Stack Pointer)
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	(Status Saving Register during interrupt)
EIPSW	(Status Saving Register during interrupt)

FEPC	(Status Saving Register during NMI)
FEPSW	(Status Saving Register during NMI)

ECR	(Interrrupt Source Register)

PSW	(Program Status Word)

CTPC	(Status Saving Register during CALLT execution)
CTPSW	(Status Saving Register during CALLT execution)

DBPC	(Status Saving Register during exception/debug trap)
DBPSW	(Status Saving Register during exception/debug trap)

CTBP	(CALLT Base Pointer)

3.2.1 Program register set

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

(1) General registers

Thirty-two general registers, r0 to r31, are available. Any of these registers can be used as a data variable or address variable.
However, r0 and r30 are implicitly used by instructions, and care must be exercised when using these registers. r0 is a register that always holds 0, and is used for operations using 0 and offset 0 addressing. r30 is used, by means of the SLD and SST instructions, as a base pointer for when memory is accessed. Also, r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler. Therefore, before using these registers, their contents must be saved so that they are not lost. The contents must be restored to the registers after the registers have been used.

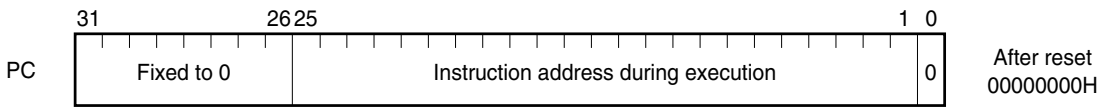
Table 3-1: Program Registers

Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating 32-bit immediate data
r2	Address/data variable registers	
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	Used to access global variable in data area
r5	Text pointer	Register to indicate the start of the text area (where program code is located)
r6 to r29	Address/data variable registers	
r30	Element pointer	Base pointer when memory is accessed
r31	Link pointer	Used by compiler when calling function
PC	Program counter	Holds instruction address during program execution

(2) Program counter

This register holds the instruction address during program execution. The lower 26 bits of this register are valid, and bits 31 to 26 are fixed to 0. If a carry occurs from bit 25 to 26, it is ignored. Bit 0 is fixed to 0, and branching to an odd address cannot be performed.

Figure 3-2: Program Counter (PC)



3.2.2 System register set

System registers control the status of the CPU and hold interrupt information.
To read/write these system registers, use the system register load/store instruction (LDSR or STSR instruction) with a specific system register number indicated below.

Table 3-2: System Register Numbers

No.	System Register Name	Operand Specification	
		LDSR Instruction	STSR Instruction
0	Status saving register during interrupt (EIPC)[Note 1]	O	O
1	Status saving register during interrupt (EIPSW)	O	O
2	Status saving register during NMI (FEPC)	O	O
3	Status saving register during NMI (FEPSW)	O	O
4	Interrupt source register (ECR)	×	O
5	Program status word (PSW)	O	O
6 to 15	Reserved number for future function expansion (operations that access these register numbers cannot be guaranteed).	×	×
16	Status saving register during CALLT execution (CTPC)	O	O
17	Status saving register during CALLT execution (CTPSW)	O	O
18	Status saving register during exception/debug trap (DBPC)	O[Note 2]	O
19	Status saving register during exception/debug trap (DBPSW)	O[Note 2]	O
20	CALLT base pointer (CTBP)	O	O
21 to 31	Reserved number for future function expansion (operations that access these register numbers cannot be guaranteed).	×	×

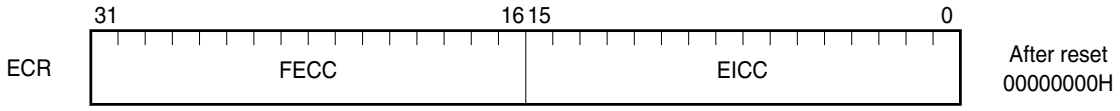
Notes: 1. Because this register has only one set, to approve multiple interrupts, it is necessary to save this register by program.

2. Access is only possible while the DBTRAP instruction is executed.

Caution: Even if bit 0 of EIPC, FEPC, or CTPC is set to 1 with the LDSR instruction, bit 0 will be ignored when the program returned by RETI instruction after interrupt servicing (because bit 0 of the PC is fixed to 0). When setting the value of EIPC, FEPC, or CTPC, use the even value (bit 0 = 0).

Remark: O: Access allowed
×: Access prohibited

Figure 3-3: Interrupt Source Register (ECR)



Bit Position	Bit Name	Function
31 to 16	FECC	Exception code of non-maskable interrupt (NMI)
15 to 0	EICC	Exception code of exception/maskable interrupt

Figure 3-4: Program Status Word (PSW)

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

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

Table 3-3: Saturation-Processed Operation Result

Status of Operation Result	Flag Status			Saturation-Processed Operation Result
	SAT	OV	S	
Maximum positive value exceeded	1	1	0	7FFF FFFFH
Maximum negative value exceeded	1	1	1	8000 0000H
Positive (maximum not exceeded)	Retains the value before operation	0	0	Operation result itself
Negative (maximum not exceeded)			1	

3.3 Operation Modes

3.3.1 Operation modes

The µPD70F3178 CarGate+ has the following operations modes. Mode specification is carried out by the MODE0 and MODE1 pins.

Table 3-4: Operation Modes

Operation Mode	Pins	
	MODE0	MODE1 (PDL5)
Normal operation mode	L	×
Flash programming mode memory	H	L
Forbidden	H	H

Remark: × = don't care
L = low level
H = high level

(1) Normal operation mode (S0)

After external reset release the CPU starts opcode execution from address 0x00000000 of the embedded FLASH memory. The external memory interface signals are in port mode. External interface can be activated by application software and external memory can be accessed. However, the lower 1MB memory area is always mapped to the embedded FLASH.

(2) Flash memory programming mode (P0)

This mode is used for erasing and programming the embedded FLASH memory of CarGate+. After external reset release CPU starts execution of internal firmware.CSIB0 or UART0 can be selected as FLASH writer interface. Selection is done by pulses applied to the MODE0 pin after reset release.

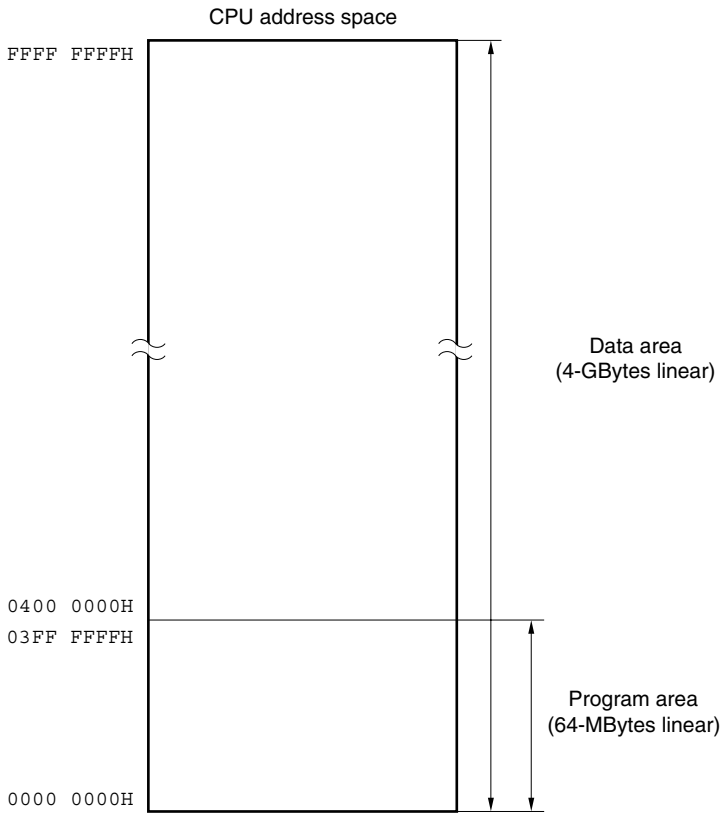
Caution: Be sure to connect a pull down resistor to pin PDL5. Otherwise the CarGate+ device may not enter the FLASH programming mode properly.

3.4 Address Space

3.4.1 CPU address space

The CPU of the μPD70F3178 is of 32-bit architecture and supports up to 4 GB of linear address space (data space) during operand addressing (data access).
Also, in instruction address addressing, a maximum of 64 MB of linear address space (program space) is supported.
Figure 3-5 shows the CPU address space.

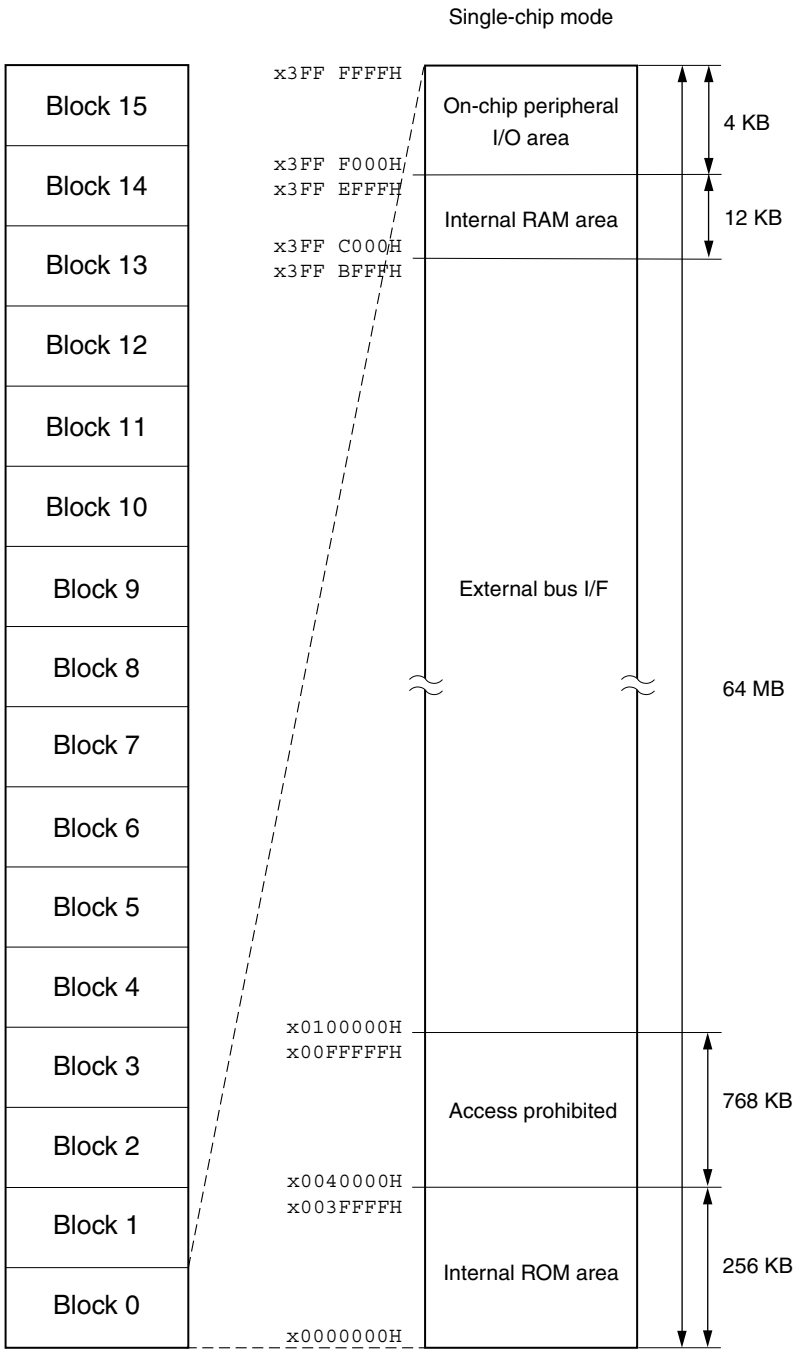
Figure 3-5: CPU Address Space



3.4.2 Image

64 MB physical address space is seen as 64 images in the 4 GB CPU address space. In actuality, the same 64 MB physical address space is accessed regardless of the values of bits 31 to 26 of the CPU address. Figure 3-6 shows the image of the virtual addressing space.
Physical address x000 0000H can be seen as CPU address 0000 0000H, and in addition, can be seen as address 0400 0000H, address 0800 0000H, …, address F800 0000H, or address FC00 0000H.

Figure 3-6: Image on Address Space

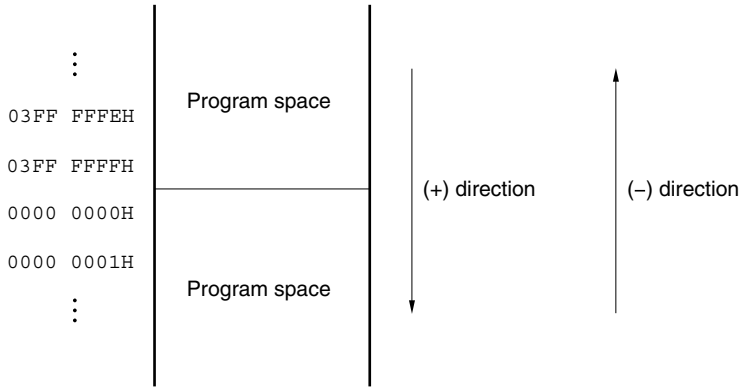


3.4.3 Wrap-around of CPU address space

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are set to “0”, and only the lower 26 bits are valid. Even if a carry or borrow occurs from bit 25 to 26 as a result of branch address calculation, the higher 6 bits ignore the carry or borrow.
Therefore, the lower-limit address of the program space, address 0000 0000H, and the upper-limit address 03FF FFFFH become contiguous addresses. Wrap-around refers to the situation that the lower-limit address and upper-limit address become contiguous like this.

Figure 3-7: Wrap-around of Program Space

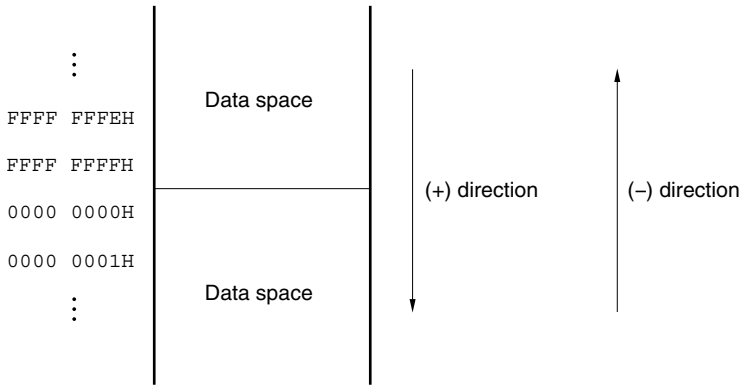


Caution: No instruction can be fetched from the 4 KB area of 03FF F000H to 03FF FFFFH because this area is defined as peripheral I/O area. Therefore, do not execute any branch address calculation in which the result will reside in any part of this area.

(2) Data space

The result of operand address calculation that exceeds 32 bits is ignored.
Therefore, the lower-limit address of the program space, address 0000 0000H, and the upper-limit address FFFF FFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.

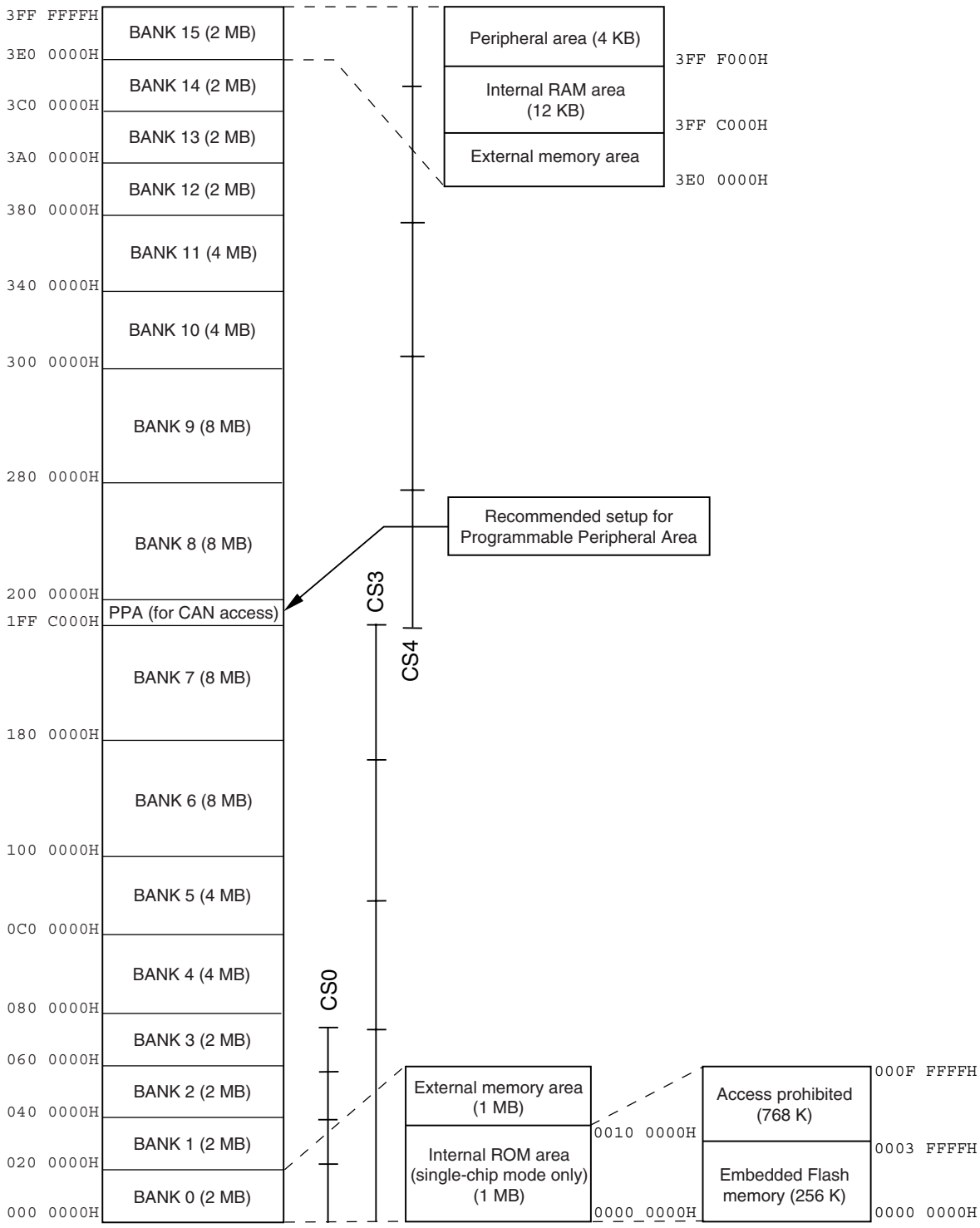
Figure 3-8: Wrap-around of Data Space



3.5 Memory Map

The μPD70F3178 reserves areas as shown in Figure 3-9.

Figure 3-9: Memory Map (μPD70F3178 (A))



3.5.1 Area

(1) Internal FLASH memory area

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

256 KB, addresses 000 0000H to 003 FFFFH, are provided in CarGate+.

(2) Interrupt/exception table

The µPD70F3178 increases the interrupt response speed by assigning handler addresses corresponding to interrupts/exceptions.
The collection of these handler addresses is called an interrupt/exception table, which is located in the external ROM area. When an interrupt/exception request is accepted, execution jumps to the handler address, and the program written at that memory is executed.
Table 3-5 shows the sources of interrupts/exceptions, and the corresponding addresses.

Table 3-5: Interrupt/Exception Table (1/3)

Start Address of Interrupt/ Exception Table	Interrupt/Exception Source
0000 0000H	$\overline{\text{RESET}}$ input
0000 0010H	P60 NMI Input
0000 0040H	TRAP0 instruction(n=0...F)
0000 0050H	TRAP1 instruction(n=0...F)
0000 0060H	Illegal opcode
0000 0080H	P01
0000 0090H	P30
0000 00A0H	P32
0000 00B0H	P20
0000 00C0H	P21
0000 00D0H	P22
0000 00E0H	P23
0000 00F0H	P24
0000 0100H	P25
0000 0110H	P26
0000 0120H	P27
0000 0130H	TMP0 overflow
0000 0140H	TMP0 capture compare channel 0
0000 0150H	TMP0 capture compare channel 1
0000 0160H	TMP1 overflow
0000 0170H	TMP1 capture compare channel 0
0000 0180H	TMP1 capture compare channel 1
Note: Reserved for internal use only please leave at $\overline{\text{RESET}}$ value.	

Table 3-5: Interrupt/Exception Table (2/3)

Start Address of Interrupt/ Exception Table	Interrupt/Exception Source
0000 0190H	TMP2 overflow
0000 01A0H	TMP2 capture compare channel 0
0000 01B0H	TMP2 capture compare channel 1
0000 01C0H	BRG0 overflow
0000 01D0H	A/D conversion end
0000 01E0H	CAN0 error
0000 01F0H	CAN0 wake up
0000 0200H	CAN0 receive
0000 0210H	CAN0 transmit
0000 0220H	CAN1 error
0000 0230H	CAN1 wake up
0000 0240H	CAN1 receive
0000 0250H	CAN1 transmit
0000 0260H	CAN2 error
0000 0270H	CAN2 wake up
0000 0280H	CAN2 receive
0000 0290H	CAN2 transmit
0000 02A0H	CAN3 error
0000 02B0H	CAN3 wake up
0000 02C0H	CAN3 receive
0000 02D0H	CAN3 transmit
0000 02E0H	CAN4 error
0000 02F0H	CAN4 wake up
0000 0300H	CAN4 receive
0000 0310H	CAN4 transmit
0000 0320H	CSIB0 transmit
0000 0330H	CSIB0 receive
0000 0340H	CSIB0 receive error
0000 0350H	CSIB1 transmit
0000 0360H	CSIB1 receive
0000 0370H	CSIB1 receive error
0000 0380H	CSIB2 transmit
0000 0390H	CSIB2 receive
0000 03A0H	CSIB2 receive error
0000 03B0H	UARTA0 receive error
Note: Reserved for internal use only please leave at $\overline{\text{RESET}}$ value.	

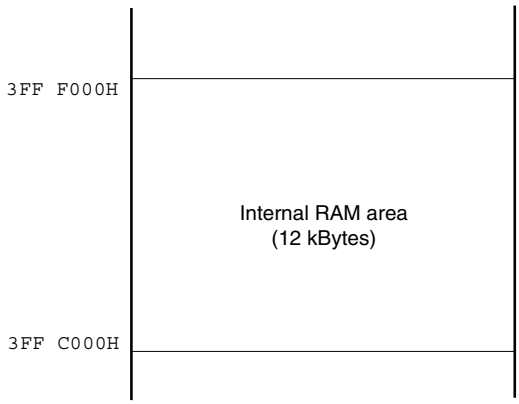
Table 3-5: Interrupt/Exception Table (3/3)

Start Address of Interrupt/ Exception Table	Interrupt/Exception Source
0000 03C0H	UARTA0 receive
0000 03D0H	UARTA0 transmit
0000 03E0H	UARTA1 receive error
0000 03F0H	UARTA1 receive
0000 0400H	UARTA1 transmit
0000 0410H	IIC0 interrupt
0000 0420H	**Note**
0000 0430H	CSIB0 receive & error interrupt
0000 0440H	CSIB1 receive & error interrupt
0000 0450H	CSIB2 receive & error interrupt
0000 0460H	UARTA0 receive & error interrupt
0000 0470H	UARTA1 receive & error interrupt
Note: Reserved for internal use only please leave at $\overline{\text{RESET}}$ value.	

(3) Internal RAM area

For the μPD70F3178 12 KB of memory, addresses 3FF C000H to 3FF F000H, are reserved for the internal RAM area.

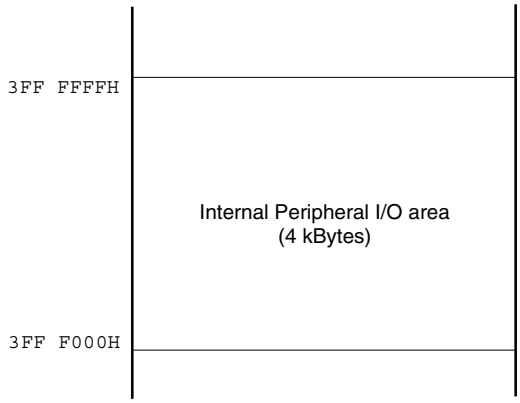
Figure 3-10: Internal RAM Area



(4) Internal peripheral I/O area

4 KB of memory, addresses 3FF F000H to 3FF FFFFH, is provided as an internal peripheral I/O area.

Figure 3-11: Internal Peripheral I/O Area



Peripheral I/O registers associated with the operation mode specification and the state monitoring for the internal peripherals I/O are all memory-mapped to the internal peripheral I/O area. Program fetches cannot be executed from this area.

Cautions: 1. In the μPD70F3178, no registers exist which are capable of word access. But if a register is word accessed, half word access is performed twice in the order of lower address, then higher address of the word area, ignoring the lower 2 bits of the address.

2. For registers in which byte access is possible, if half word access is executed, the higher 8 bits become undefined during the read operation, and the lower 8 bits of data are written to the register during the write operation.

3. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed

Additionally to the peripheral I/O area, a 16 KB area is provided as a programmable peripheral I/O area (refer to **3.7.1 ”Programmable peripheral I/O registers” on page 185**).

3.5.2 Recommended use of address space

The architecture of the μPD70F3178 requires that a register is utilized for address generation when accessing operand data in the data space. Operand data access from instruction can be directly executed at the address in this pointer register ±32 KB. However, the use of general registers as pointer registers decreases the number of usable general registers for handling variables, but minimizes the deterioration of address calculation performance when changing the pointer value and minimizes the program size as well.
To enhance the efficiency of using the pointer in consideration of the memory map of the μPD70F3178, the following points are recommended:

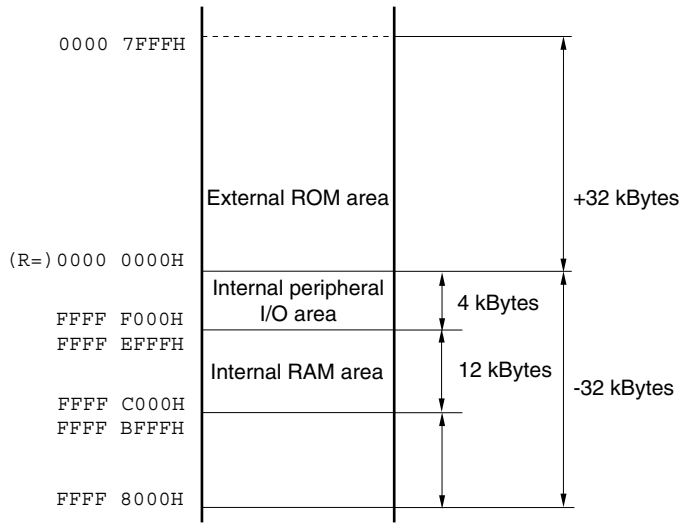
(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to zero (0), and only the lower 26 bits are valid. Therefore, a contiguous 64 MB space, starting from address 0000 0000H, unconditionally corresponds to the memory map of the program space.

(2) Data space

For the efficient use of resources to be performed through the wrap-around feature of the data space, the continuous 16 MB address spaces 0000 0000H to 00FF FFFFH and FF00 0000H to FFFF FFFFH of the 4 GB CPU address space are used as the data space. With the μPD70F3178, 64 MB physical address space is seen as 64 images in the 4 GB CPU address space. The highest bit (bit 25) of this 26-bit address is assigned as address sign-extended to 32 bits.

Figure 3-12: Example Application of wrap-around (μPD70F3178)



When R = r0 (zero register) is specified with the LD/ST disp16 [R] instruction, an addressing range of 0000 0000H ±32 KB can be referenced with the sign-extended, 16-bit displacement value.
The zero register (r0) is a register set to 0 by hardware, and eliminates the need for additional registers for the pointer.

3.6 Peripheral I/O Registers

Table 3-6: Fixed SFR Table (1/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF000	PALL	Port AL low byte	R/W	R/W	-	-	undefined
0xFFFFF000	PAL	Port AL	-	-	R/W	-	undefined
0xFFFFF001	PALH	Port AL high byte	R/W	R/W	-	-	undefined
0xFFFFF002	PAH	Port AH	R/W	R/W	-	-	undefined
0xFFFFF004	PDLL	Port DL low byte	R/W	R/W	-	-	undefined
0xFFFFF004	PDL	Port DL	-	-	R/W	-	undefined
0xFFFFF005	PDLH	Port DL high byte	R/W	R/W	-	-	undefined
0xFFFFF008	PCS	Port CS	R/W	R/W	-	-	undefined
0xFFFFF00A	PCT	Port CT	R/W	R/W	-	-	undefined
0xFFFFF020	PMALL	Port AL mode low byte	R/W	R/W	-	-	0xFF
0xFFFFF020	PMAL	Port AL mode	-	-	R/W	-	0xFFFF
0xFFFFF021	PMALH	Port AL mode high byte	R/W	R/W	-	-	0xFF
0xFFFFF022	PMAH	Port AH mode	R/W	R/W	-	-	0x0F
0xFFFFF024	PMDLL	Port DH mode low byte	R/W	R/W	-	-	0xFF
0xFFFFF024	PMDL	Port DL mode	-	-	R/W	-	0xFFFF
0xFFFFF025	PMDLH	Port DH mode high byte	R/W	R/W	-	-	0xFF
0xFFFFF028	PMCS	Port CS mode	R/W	R/W	-	-	0x19
0xFFFFF02A	PMCT	Port CT mode	R/W	R/W	-	-	0x13
0xFFFFF040	PMCALL	Port AL mode control low byte	R/W	R/W	-	-	0x00
0xFFFFF040	PMCAL	Port AL mode control	-	-	R/W	-	0x0000
0xFFFFF041	PMCALH	Port AL mode control high byte	R/W	R/W	-	-	0x00
0xFFFFF042	PMCAH	Port AH mode control	R/W	R/W	-	-	0x00
0xFFFFF044	PMCDLL	Port DL mode control low byte	R/W	R/W	-	-	0x00
0xFFFFF044	PMCDL	Port DL mode control	-	-	R/W	-	0x0000
0xFFFFF045	PMCDLH	Port DL mode control high byte	R/W	R/W	-	-	0x00
0xFFFFF048	PMCCS	Port CS mode control	R/W	R/W	-	-	0x00
0xFFFFF04A	PMCCT	Port CT mode control	R/W	R/W	-	-	0x00
0xFFFFF060	CSC0	CPU: Chip Area Select Control register 0	-	-	R/W	-	0x2C11
0xFFFFF062	CSC1	CPU: Chip Area Select Control register 1	-	-	R/W	-	0x2C11
0xFFFFF064	BPC	CPU: Peripheral Area Select Control register	-	-	R/W	-	0x0000
0xFFFFF066	BSC	CPU: Bus Size Configuration register	-	-	R/W	-	0x5555
0xFFFFF068	BEC	CPU: Endian Configuration register	-	-	R/W	-	0x0000
0xFFFFF06E	VSWC	CPU: VPB Strobe Wait Control register	R/W	R/W	-	-	0x77
0xFFFFF100	IMR0L	Interrupt Mask register 0L	R/W	R/W	-	-	0xFF
0xFFFFF100	IMR0	Interrupt Mask register 0	-	-	R/W	-	0xFFFF

Table 3-6: Fixed SFR Table (2/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF101	IMR0H	Interrupt Mask register 0H	R/W	R/W	-	-	0xFF
0xFFFFF102	IMR1L	Interrupt Mask register 1L	R/W	R/W	-	-	0xFF
0xFFFFF102	IMR1	Interrupt Mask register 1	-	-	R/W	-	0xFFFF
0xFFFFF103	IMR1H	Interrupt Mask register 1H	R/W	R/W	-	-	0xFF
0xFFFFF104	IMR2L	Interrupt Mask register 2L	R/W	R/W	-	-	0xFF
0xFFFFF104	IMR2	Interrupt Mask register 2	-	-	R/W	-	0xFFFF
0xFFFFF105	IMR2H	Interrupt Mask register 2H	R/W	R/W	-	-	0xFF
0xFFFFF106	IMR3L	Interrupt Mask register 3L	R/W	R/W	-	-	0xFF
0xFFFFF106	IMR3	Interrupt Mask register 3	-	-	R/W	-	0xFFFF
0xFFFFF107	IMR3H	Interrupt Mask register 3H	R/W	R/W	-	-	0xFF
0xFFFFF10E	IMR7H	Interrupt Mask register 7H	R/W	R/W	-	-	0x00
0xFFFFF10E	IMR7L	Interrupt Mask register 7L	R/W	R/W	-	-	0x00
0xFFFFF10E	IMR7	Interrupt Mask register 7	-	-	R/W	-	0x0000
0xFFFFF110	P0IC	Interrupt control register for INTP0	R/W	R/W	-	-	0x47
0xFFFFF112	P1IC	Interrupt control register for INTP1	R/W	R/W	-	-	0x47
0xFFFFF114	P2IC	Interrupt control register for INTP2	R/W	R/W	-	-	0x47
0xFFFFF116	PIC3	Interrupt control register 3	R/W	R/W	-	-	0x47
0xFFFFF116	P3IC	Interrupt control register for INTP3	R/W	R/W	-	-	0x47
0xFFFFF118	P4IC	Interrupt control register for INTP4	R/W	R/W	-	-	0x47
0xFFFFF11A	P5IC	Interrupt control register for INTP5	R/W	R/W	-	-	0x47
0xFFFFF11C	P6IC	Interrupt control register for INTP6	R/W	R/W	-	-	0x47
0xFFFFF11E	P7IC	Interrupt control register for INTP7	R/W	R/W	-	-	0x47
0xFFFFF120	P8IC	Interrupt control register for INTP8	R/W	R/W	-	-	0x47
0xFFFFF122	P9IC	Interrupt control register for INTP9	R/W	R/W	-	-	0x47
0xFFFFF124	P10IC	Interrupt control register for INTP10	R/W	R/W	-	-	0x47
0xFFFFF126	TP0OVIC	Interrupt control register for INTTP0OV	R/W	R/W	-	-	0x47
0xFFFFF128	TP0CC0IC	Interrupt control register for INTTP0CC0	R/W	R/W	-	-	0x47
0xFFFFF12A	TP0CC1IC	Interrupt control register for INTTP0CC1	R/W	R/W	-	-	0x47
0xFFFFF12C	TP1OVIC	Interrupt control register for INTTP1OV	R/W	R/W	-	-	0x47
0xFFFFF12E	TP1CC0IC	Interrupt control register for INTTP1CC0	R/W	R/W	-	-	0x47
0xFFFFF130	TP1CC1IC	Interrupt control register for INTTP1CC1	R/W	R/W	-	-	0x47
0xFFFFF132	TP2OVIC	Interrupt control register for INTTP2OV	R/W	R/W	-	-	0x47
0xFFFFF134	TP2CC0IC	Interrupt control register for INTTP2CC0	R/W	R/W	-	-	0x47
0xFFFFF136	TP2CC1IC	Interrupt control register for INTTP2CC1	R/W	R/W	-	-	0x47
0xFFFFF138	BRG0IC	Interrupt control register for INTBRG0	R/W	R/W	-	-	0x47
0xFFFFF13A	ADIC	Interrupt control register for INTAD	R/W	R/W	-	-	0x47
0xFFFFF13C	C0ERRIC	Interrupt control register for INTC0ERR	R/W	R/W	-	-	0x47
0xFFFFF13E	C0WUPIC	Interrupt control register for INTC0WUP	R/W	R/W	-	-	0x47
0xFFFFF140	C0RECIC	Interrupt control register for INTC0REC	R/W	R/W	-	-	0x47
0xFFFFF142	C0TRXIC	Interrupt control register for INTC0TRX	R/W	R/W	-	-	0x47
0xFFFFF144	C1ERRIC	Interrupt control register for INTC1ERR	R/W	R/W	-	-	0x47

Table 3-6: Fixed SFR Table (3/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF146	C1WUPIC	Interrupt control register for INTC1WUP	R/W	R/W	-	-	0x47
0xFFFFF148	C1RECIC	Interrupt control register for INTC1REC	R/W	R/W	-	-	0x47
0xFFFFF14A	C1TRXIC	Interrupt control register for INTC1TRX	R/W	R/W	-	-	0x47
0xFFFFF14C	C2ERRIC	Interrupt control register for INTC2ERR	R/W	R/W	-	-	0x47
0xFFFFF14E	C2WUPIC	Interrupt control register for INTC2WUP	R/W	R/W	-	-	0x47
0xFFFFF150	C2RECIC	Interrupt control register for INTC2REC	R/W	R/W	-	-	0x47
0xFFFFF152	C2TRXIC	Interrupt control register for INTC2TRX	R/W	R/W	-	-	0x47
0xFFFFF154	C3ERRIC	Interrupt control register for INTC3ERR	R/W	R/W	-	-	0x47
0xFFFFF156	C3WUPIC	Interrupt control register for INTC3WUP	R/W	R/W	-	-	0x47
0xFFFFF158	C3RECIC	Interrupt control register for INTC3REC	R/W	R/W	-	-	0x47
0xFFFFF15A	C3TRXIC	Interrupt control register for INTC3TRX	R/W	R/W	-	-	0x47
0xFFFFF15C	C4ERRIC	Interrupt control register for INTC4ERR	R/W	R/W	-	-	0x47
0xFFFFF15E	C4WUPIC	Interrupt control register for INTC4WUP	R/W	R/W	-	-	0x47
0xFFFFF160	C4RECIC	Interrupt control register for INTC4REC	R/W	R/W	-	-	0x47
0xFFFFF162	C4TRXIC	Interrupt control register for INTC4TRX	R/W	R/W	-	-	0x47
0xFFFFF164	CB0TIC	Interrupt control register for INTCB0T	R/W	R/W	-	-	0x47
0xFFFFF166	CB0RIC	Interrupt control register for INTCB0R	R/W	R/W	-	-	0x47
0xFFFFF168	CB0REIC	Interrupt control register for INTCB0RE	R/W	R/W	-	-	0x47
0xFFFFF16A	CB1TIC	Interrupt control register for INTCB1T	R/W	R/W	-	-	0x47
0xFFFFF16C	CB1RIC	Interrupt control register for INTCB1R	R/W	R/W	-	-	0x47
0xFFFFF16E	CB1REIC	Interrupt control register for INTCB1RE	R/W	R/W	-	-	0x47
0xFFFFF170	CB2TIC	Interrupt control register for INTCB2T	R/W	R/W	-	-	0x47
0xFFFFF172	CB2RIC	Interrupt control register for INTCB2R	R/W	R/W	-	-	0x47
0xFFFFF174	CB2REIC	Interrupt control register for INTCB2RE	R/W	R/W	-	-	0x47
0xFFFFF176	UA0REIC	Interrupt control register for INTUA0RE	R/W	R/W	-	-	0x47
0xFFFFF178	UA0RIC	Interrupt control register for INTUA0R	R/W	R/W	-	-	0x47
0xFFFFF17A	UA0TIC	Interrupt control register for INTUA0T	R/W	R/W	-	-	0x47
0xFFFFF17C	UA1REIC	Interrupt control register for INTUA1RE	R/W	R/W	-	-	0x47
0xFFFFF17E	UA1RIC	Interrupt control register for INTUA1R	R/W	R/W	-	-	0x47
0xFFFFF180	UA1TIC	Interrupt control register for INTUA1T	R/W	R/W	-	-	0x47
0xFFFFF182	IIC0IC	Interrupt control register for INTIIC0	R/W	R/W	-	-	0x47
0xFFFFF1FA	ISPR	In-service Priority register	R	R	-	-	0x00
0xFFFFF1FC	PRCMD	Command register	-	W	-	-	undefined
0xFFFFF1FE	PSC	Power Save Control register	R/W	R/W	-	-	0x00
0xFFFFF200	ADA0M0	ADC mode register 0	R/W	R/W	-	-	0x00
0xFFFFF201	ADA0M1	ADC mode register 1	R/W	R/W	-	-	0x00
0xFFFFF202	ADA0S	ADC channel select register	R/W	R/W	-	-	0x00
0xFFFFF203	ADA0M2	ADC mode register 2	R/W	R/W	-	-	0x00
0xFFFFF204	ADA0PFM	ADC power fail comparison mode register	R/W	R/W	-	-	0x00
0xFFFFF205	ADA0PFT	ADC power fail threshold register	R/W	R/W	-	-	0x00

Table 3-6: Fixed SFR Table (4/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF210	ADCR0	ADC result register channel 0	-	-	R	-	undefined
0xFFFFF211	ADCRH0	ADC result register high byte channel 0	R	R	-	-	undefined
0xFFFFF212	ADCR1	ADC result register channel 1	-	-	R	-	undefined
0xFFFFF213	ADCRH1	ADC result register high byte channel 1	R	R	-	-	undefined
0xFFFFF214	ADCR2	ADC result register channel 2	-	-	R	-	undefined
0xFFFFF215	ADCRH2	ADC result register high byte channel 2	R	R	-	-	undefined
0xFFFFF216	ADCR3	ADC result register channel 3	-	-	R	-	undefined
0xFFFFF217	ADCRH3	ADC result register high byte channel 3	R	R	-	-	undefined
0xFFFFF300	SELCNT0	! Description not available!	R/W	R/W	-	-	0x00
0xFFFFF302	SELCNT1	! Description not available!	R/W	R/W	-	-	0x00
0xFFFFF304	SELCNT2	! Description not available!	R/W	R/W	-	-	0x00
0xFFFFF340	OCKS0	Clock selection register	-	R/W	-	-	0x00
0xFFFFF400	P0	Port 0 register	R/W	R/W	-	-	undefined
0xFFFFF402	P1	Port 1 register	R/W	R/W	-	-	undefined
0xFFFFF404	P2	Port 2 register	R/W	R/W	-	-	undefined
0xFFFFF406	P3	Port 3 register	R/W	R/W	-	-	undefined
0xFFFFF408	P4	Port 4 register	R/W	R/W	-	-	undefined
0xFFFFF40E	P7	ADC port register ANI0 to ANI7	R/W	R/W	-	-	undefined
0xFFFFF420	PM0	Port 0 mode register	R/W	R/W	-	-	0x07
0xFFFFF422	PM1	Port 1 mode register	R/W	R/W	-	-	0xFF
0xFFFFF424	PM2	Port 2 mode register	R/W	R/W	-	-	0xFF
0xFFFFF426	PM3	Port 3 mode register	R/W	R/W	-	-	0x7F
0xFFFFF428	PM4	Port 4 mode register	R/W	R/W	-	-	0xFF
0xFFFFF440	PMC0	Port 0 mode control register	R/W	R/W	-	-	0x00
0xFFFFF442	PMC1	Port 1 mode control register	R/W	R/W	-	-	0x00
0xFFFFF444	PMC2	Port 2 mode control register	R/W	R/W	-	-	0x00
0xFFFFF446	PMC3	Port 3 mode control register	R/W	R/W	-	-	0x00
0xFFFFF448	PMC4	Port 4 mode control register	R/W	R/W	-	-	0x00
0xFFFFF466	PFC3	Port 3 function control register	R/W	R/W	-	-	0x00
0xFFFFF480	BCT0L	Bus cycle type configuration register 0 low byte	R/W	R/W	-	-	0x88
0xFFFFF480	BCT0	Bus cycle type configuration register 0	-	-	R/W	-	0x8888
0xFFFFF481	BCT0H	Bus cycle type configuration register 0 high byte	R/W	R/W	-	-	0x88
0xFFFFF482	BCT1L	Bus cycle type configuration register 1 low byte	R/W	R/W	-	-	0x88
0xFFFFF482	BCT1	Bus cycle type configuration register 1	-	-	R/W	-	0x8888
0xFFFFF483	BCT1H	Bus cycle type configuration register 1 high byte	R/W	R/W	-	-	0x88

Table 3-6: Fixed SFR Table (5/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF484	DWC0L	Data Wait Control Register 0 low byte	R/W	R/W	-	-	0x77
0xFFFFF484	DWC0	Data Wait Control Register 0	-	-	R/W	-	0x7777
0xFFFFF485	DWC0H	Data Wait Control Register 0 high byte	R/W	R/W	-	-	0x77
0xFFFFF486	DWC1L	Data Wait Control Register 1 low byte	R/W	R/W	-	-	0x77
0xFFFFF486	DWC1	Data Wait Control Register 1	-	-	R/W	-	0x7777
0xFFFFF487	DWC1H	Data Wait Control Register 1 high byte	R/W	R/W	-	-	0x77
0xFFFFF488	BCCL	Bus cycle control register low byte	R/W	R/W	-	-	0xFF
0xFFFFF488	BCC	Bus cycle control register	-	-	R/W	-	0xFFFF
0xFFFFF489	BCCH	Bus cycle control register high byte	R/W	R/W	-	-	0xFF
0xFFFFF48A	ASCL	Address Setting Wait Control Register	R/W	R/W	-	-	0xFF
0xFFFFF48A	ASC	Address Setting Wait Control Register	-	-	R/W	-	0xFFFF
0xFFFFF48B	ASCH	Address Setting Wait Control Register	R/W	R/W	-	-	0xFF
0x3FFFF49A	PRC	Page-ROM Configuration register	-	-	R/W	-	0x7000
0xFFFFF590	TP0CTL0	TMP0 timer control register 0	R/W	R/W	-	-	0x00
0xFFFFF591	TP0CTL1	TMP0 timer control register 1	R/W	R/W	-	-	0x00
0xFFFFF592	TP0IOC0	TMP0 timer-specific I/O control register 0	R/W	R/W	-	-	0x00
0xFFFFF593	TP0IOC1	TMP0 timer-specific I/O control register 1	R/W	R/W	-	-	0x00
0xFFFFF594	TP0IOC2	TMP0 timer-specific I/O control register 2	R/W	R/W	-	-	0x00
0xFFFFF595	TP0OPT0	TMP0 option register	R/W	R/W	-	-	0x00
0xFFFFF596	TP0CCR0	TMP0 capture/compare register 0	-	-	R/W	-	0x0000
0xFFFFF598	TP0CCR1	TMP0 capture/compare register 1	-	-	R/W	-	0x0000
0xFFFFF59A	TP0CNT	TMP0 count register	-	-	R	-	0x0000
0xFFFFF5A0	TP1CTL0	TMP1 timer control register 0	R/W	R/W	-	-	0x00
0xFFFFF5A1	TP1CTL1	TMP1 timer control register 1	R/W	R/W	-	-	0x00
0xFFFFF5A2	TP1IOC0	TMP1 timer-specific I/O control register 0	R/W	R/W	-	-	0x00
0xFFFFF5A3	TP1IOC1	TMP1 timer-specific I/O control register 1	R/W	R/W	-	-	0x00
0xFFFFF5A4	TP1IOC2	TMP1 timer-specific I/O control register 2	R/W	R/W	-	-	0x00
0xFFFFF5A5	TP1OPT0	TMP1 option register	R/W	R/W	-	-	0x00
0xFFFFF5A6	TP1CCR0	TMP1 capture/compare register 0	-	-	R/W	-	0x0000
0xFFFFF5A8	TP1CCR1	TMP1 capture/compare register 1	-	-	R/W	-	0x0000
0xFFFFF5AA	TP1CNT	TMP1 count register	-	-	R	-	0x0000
0xFFFFF5B0	TP2CTL0	TMP2 timer control register 0	R/W	R/W	-	-	0x00
0xFFFFF5B1	TP2CTL1	TMP2 timer control register 1	R/W	R/W	-	-	0x00
0xFFFFF5B2	TP2IOC0	TMP2 timer-specific I/O control register 0	R/W	R/W	-	-	0x00
0xFFFFF5B3	TP2IOC1	TMP2 timer-specific I/O control register 1	R/W	R/W	-	-	0x00
0xFFFFF5B4	TP2IOC2	TMP2 timer-specific I/O control register 2	R/W	R/W	-	-	0x00
0xFFFFF5B5	TP2OPT0	TMP2 option register	R/W	R/W	-	-	0x00

Table 3-6: Fixed SFR Table (6/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFF5B6	TP2CCR0	TMP2 capture/compare register 0	-	-	R/W	-	0x0000
0xFFFFF5B8	TP2CCR1	TMP2 capture/compare register 1	-	-	R/W	-	0x0000
0xFFFFF5BA	TP2CNT	TMP2 count register	-	-	R	-	0x0000
0xFFFFF820	PSM	Power save mode register	R/W	R/W	-	-	0x00
0xFFFFF822	CKC	Clock control register	R/W	R/W	-	-	0x00
0xFFFFF824	PSTAT	PLL status register	R	R	-	-	0x01
0xFFFFF840	PLC	Prescaler mode register	R/W	R/W	-	-	0x00
0xFFFFF880	INTM0	External interrupt edge select register 0	R/W	R/W	-	-	0x00
0xFFFFF882	INTM1	External interrupt edge select register 1	R/W	R/W	-	-	0x00
0xFFFFF884	INTM2	External interrupt edge select register 2	R/W	R/W	-	-	0x00
0xFFFFF886	INTM3	External interrupt edge select register 3	R/W	R/W	-	-	0x00
0xFFFFF8B0	PRSM0	Prescaler mode register	R/W	R/W	-	-	0x00
0xFFFFF8B1	PRSCM0	Prescaler compare register	R/W	R/W	-	-	0x00
0xFFFFFA00	UA0CTL0	UARTA0 Control Register 0	R/W	R/W	-	-	0x10
0xFFFFFA01	UA0CTL1	UARTA0 Control Register 1	R/W	R/W	-	-	0x00
0xFFFFFA02	UA0CTL2	UARTA0 Control Register 2	-	R/W	-	-	0xFF
0xFFFFFA03	UA0OPT0	UARTA0 Option Register	R/W	R/W	-	-	0x14
0xFFFFFA04	UA0STR	UARTA0 Status Register	R/W	R/W	-	-	0x00
0xFFFFFA06	UA0RX	UARTA0 Reception data Register	-	R	-	-	0xFF
0xFFFFFA07	UA0TX	UARTA0 Transfer data Register	R/W	R/W	-	-	0xFF
0xFFFFFA10	UA1CTL0	UARTA1 Control Register 0	R/W	R/W	-	-	0x10
0xFFFFFA11	UA1CTL1	UARTA1 Control Register 1	R/W	R/W	-	-	0x00
0xFFFFFA12	UA1CTL2	UARTA1 Control Register 2	-	R/W	-	-	0xFF
0xFFFFFA13	UA1OPT0	UARTA1 Option Register	R/W	R/W	-	-	0x14
0xFFFFFA14	UA1STR	UARTA1 Status Register	R/W	R/W	-	-	0x00
0xFFFFFA16	UA1RX	UARTA1 Reception data Register	-	R	-	-	0xFF
0xFFFFFA17	UA1TX	UARTA1 Transfer data Register	R/W	R/W	-	-	0xFF
0xFFFFFD00	CB0CTL0	CSIB0 control register 0	R/W	R/W	-	-	0x01
0xFFFFFD01	CB0CTL1	CSIB0 control register 1	R/W	R/W	-	-	0x00
0xFFFFFD02	CB0CTL2	CSIB0 control register 2	-	R/W	-	-	0x00
0xFFFFFD03	CB0STR	CSIB0 state register	R/W	R/W	-	-	0x00
0xFFFFFD04	CB0RX0	CSIB0 received data register	-	-	R	-	0x0000
0xFFFFFD04	CB0RX0L	CSIB0 received data register low byte	-	R	-	-	0x00

Table 3-6: Fixed SFR Table (7/7)

Address	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0xFFFFFD06	CB0TX0L	CSIB0 send data register low byte	-	R/W	-	-	0x00
0xFFFFFD06	CB0TX0	CSIB0 send data register	-	-	R/W	-	0x0000
0xFFFFFD10	CB1CTL0	CSIB1 control register 0	R/W	R/W	-	-	0x01
0xFFFFFD11	CB1CTL1	CSIB1 control register 1	R/W	R/W	-	-	0x00
0xFFFFFD12	CB1CTL2	CSIB1 control register 2	-	R/W	-	-	0x00
0xFFFFFD13	CB1STR	CSIB1 state register	R/W	R/W	-	-	0x00
0xFFFFFD14	CB1RX0	CSIB1 received data register	-	-	R	-	0x0000
0xFFFFFD14	CB1RX0L	CSIB1 received data register low byte	-	R	-	-	0x00
0xFFFFFD16	CB1TX0L	CSIB1 send data register low byte	-	R/W	-	-	0x00
0xFFFFFD16	CB1TX0	CSIB1 send data register	-	-	R/W	-	0x0000
0xFFFFFD20	CB2CTL0	CSIB2 control register 0	R/W	R/W	-	-	0x01
0xFFFFFD21	CB2CTL1	CSIB2 control register 1	R/W	R/W	-	-	0x00
0xFFFFFD22	CB2CTL2	CSIB2 control register 2	-	R/W	-	-	0x00
0xFFFFFD23	CB2STR	CSIB2 state register	R/W	R/W	-	-	0x00
0xFFFFFD24	CB2RX0	CSIB2 received data register	-	-	R	-	0x0000
0xFFFFFD24	CB2RX0L	CSIB2 received data register low byte	-	R	-	-	0x00
0xFFFFFD26	CB2TX0L	CSIB2 send data register low byte	-	R/W	-	-	0x00
0xFFFFFD26	CB2TX0	CSIB2 send data register	-	-	R/W	-	0x0000
0xFFFFFD80	IIC0	IIC0 shift register	-	R/W	-	-	0x00
0xFFFFFD82	IICC0	IIC0 control register	R/W	R/W	-	-	0x00
0xFFFFFD82	IICC0SVA0	IIC0 combined IICC0 and SVA0 register	-	-	R/W	-	0x0000
0xFFFFFD83	SVA0	IIC0 Slave address register	-	R/W	-	-	0x00
0xFFFFFD84	IICCL0	IIC0 clock selection register	R/W	R/W	-	-	0x00
0xFFFFFD84	IICCL0IICX0	IIC0 combined IICCL0 and IICX0 register	-	-	R/W	-	0x0000
0xFFFFFD85	IICX0	IIC0 function expansion register	R/W	R/W	-	-	0x00
0xFFFFFD86	IICS0	IIC0 state register	R	R	-	-	0x00
0xFFFFFD87	IICSE0	IIC0 state register (for emulation only)	R	R	-	-	0x00
0xFFFFFD8A	IICF0	IIC0 flag register	R/W	R/W	-	-	0x00

3.7 PPA related SFR Register

The following table lists all SFR registers with SFR addresses. relative to the Programmable Peripheral Area PPA. The PPA is allocated by the BPC register of CPU. SFR registers, which can be accessed by the peripheral area at the top of the memory space, are not included in this list.

Table 3-7: PPA related SFR Table (1/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000000	C0GMCTRLL	CAN0 global macro control register low byte	R/W	R/W	-	-	0x00
0x00000000	C0GMCTRL	CAN0 global macro control register	-	-	R/W	-	0x0000
0x00000001	C0GMCTRLH	CAN0 global macro control register high byte	R/W	R/W	-	-	0x00
0x00000002	C0GMCS	CAN0 global macro clock selection register	R/W	R/W	-	-	0x0F
0x00000004	C0GMCONF	CAN0 global configuration register	-	-	R	-	0x0019
0x00000006	C0GMABTL	CAN0 global macro automatic block transmission register low byte	R/W	R/W	-	-	0x00
0x00000006	C0GMABT	CAN0 global macro automatic block transmission register	-	-	R/W	-	0x0000
0x00000007	C0GMABTH	CAN0 global macro automatic block transmission register high byte	R/W	R/W	-	-	0x00
0x00000008	C0GMABTD	CAN0 global macro automatic block transmission delay register	R/W	R/W	-	-	0x00
0x00000040	C0MASK1L	CAN0 Module Mask 1 Register lower half word	-	-	R/W	-	undef.
0x00000042	C0MASK1H	CAN0 Module Mask 1 Register upper half word	-	-	R/W	-	undef.
0x00000044	C0MASK2L	CAN0 Module Mask 2 Register lower half word	-	-	R/W	-	undef.
0x00000046	C0MASK2H	CAN0 Module Mask 2 Register upper half word	-	-	R/W	-	undef.
0x00000048	C0MASK3L	CAN0 Module Mask 3 Register lower half word	-	-	R/W	-	undef.
0x0000004A	C0MASK3H	CAN0 Module Mask 3 Register upper half word	-	-	R/W	-	undef.
0x0000004C	C0MASK4L	CAN0 Module Mask 4 Register lower half word	-	-	R/W	-	undef.
0x0000004E	C0MASK4H	CAN0 Module Mask 4 Register upper half word	-	-	R/W	-	undef.
0x00000050	C0CTRL	CAN0 Module Control Register	-	-	R/W	-	0x0000
0x00000052	C0LEC	CAN0 Module Last Error Code Register	R/W	R/W	-	-	0x00
0x00000053	C0INFO	CAN0 Module Information Register	R	R	-	-	0x00
0x00000054	C0ERC	CAN0 Module Error Counter	-	-	R	-	0x0000
0x00000056	C0IEL	CAN0 Module Interrupt Enable Register low byte	R/W	R/W	-	-	0x00
0x00000056	C0IE	CAN0 Module Interrupt Enable Register	-	-	R/W	-	0x0000
0x00000057	C0IEH	CAN0 Module Interrupt Enable Register high byte	R/W	R/W	-	-	0x00
0x00000058	C0INTSL	CAN0 Module Interrupt Status Register low byte	R/W	R/W	-	-	0x00
0x00000058	C0INTS	CAN0 Module Interrupt Status Register	-	-	R/W	-	0x0000
0x0000005A	C0BRP	CAN0 Module Bit-Rate Prescaler Register	R/W	R/W	-	-	0xFF
0x0000005C	C0BTR	CAN0 Bit Rate Register	-	-	R/W	-	0x370F
0x0000005E	C0LIPT	CAN0 Module Last In-Pointer Register	-	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (2/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000060	C0RGPTL	CAN0 Module Receive History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00000060	C0RGPT	CAN0 Module Receive History List Get Pointer Register	-	-	R/W	-	undef.
0x00000062	C0LOPT	CAN0 Module Last Out-Pointer Register	-	R	-	-	undef.
0x00000064	C0TGPTL	CAN0 Module Transmit History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00000064	C0TGPT	CAN0 Module Transmit History List Get Pointer Register	-	-	R/W	-	undef.
0x00000066	C0TSL	CAN0 Module Time Stamp Register low byte	R/W	R/W	-	-	0x00
0x00000066	C0TS	CAN0 Module Time Stamp Register	-	-	R/W	-	0x0000
0x00000067	C0TSH	CAN0 Module Time Stamp Register high byte	R/W	R/W	-	-	0x00
0x00000100	C0MDATA000	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000100	C0MDATA0100	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000101	C0MDATA100	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000102	C0MDATA200	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000102	C0MDATA2300	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000103	C0MDATA300	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000104	C0MDATA400	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000104	C0MDATA4500	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000105	C0MDATA500	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000106	C0MDATA600	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000106	C0MDATA6700	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000107	C0MDATA700	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000108	C0MDLC00	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000109	C0MCONF00	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000010A	C0MIDL00	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000010C	C0MIDH00	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000010E	C0MCTRL00	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000120	C0MDATA001	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000120	C0MDATA0101	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000121	C0MDATA101	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000122	C0MDATA201	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000122	C0MDATA2301	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000123	C0MDATA301	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000124	C0MDATA401	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000124	C0MDATA4501	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000125	C0MDATA501	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000126	C0MDATA601	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000126	C0MDATA6701	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000127	C0MDATA701	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (3/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000128	C0MDLC01	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000129	C0MCONF01	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000012A	C0MIDL01	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000012C	C0MIDH01	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000012E	C0MCTRL01	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000140	C0MDATA002	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000140	C0MDATA0102	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000141	C0MDATA102	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000142	C0MDATA202	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000142	C0MDATA2302	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000143	C0MDATA302	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000144	C0MDATA402	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000144	C0MDATA4502	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000145	C0MDATA502	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000146	C0MDATA602	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000146	C0MDATA6702	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000147	C0MDATA702	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000148	C0MDLC02	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000149	C0MCONF02	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000014A	C0MIDL02	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000014C	C0MIDH02	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000014E	C0MCTRL02	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000160	C0MDATA003	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000160	C0MDATA0103	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000161	C0MDATA103	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000162	C0MDATA203	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000162	C0MDATA2303	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000163	C0MDATA303	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000164	C0MDATA403	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000164	C0MDATA4503	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000165	C0MDATA503	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000166	C0MDATA603	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000166	C0MDATA6703	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000167	C0MDATA703	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000168	C0MDLC03	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000169	C0MCONF03	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000016A	C0MIDL03	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000016C	C0MIDH03	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000016E	C0MCTRL03	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000180	C0MDATA004	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000180	C0MDATA0104	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (4/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000181	C0MDATA104	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000182	C0MDATA204	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000182	C0MDATA2304	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000183	C0MDATA304	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000184	C0MDATA404	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000184	C0MDATA4504	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000185	C0MDATA504	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000186	C0MDATA604	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000186	C0MDATA6704	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000187	C0MDATA704	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000188	C0MDLC04	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000189	C0MCONF04	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000018A	C0MIDL04	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000018C	C0MIDH04	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000018E	C0MCTRL04	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001A0	C0MDATA005	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A0	C0MDATA0105	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001A1	C0MDATA105	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A2	C0MDATA205	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A2	C0MDATA2305	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001A3	C0MDATA305	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A4	C0MDATA405	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A4	C0MDATA4505	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001A5	C0MDATA505	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A6	C0MDATA605	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A6	C0MDATA6705	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001A7	C0MDATA705	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A8	C0MDLC05	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001A9	C0MCONF05	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001AA	C0MIDL05	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001AC	C0MIDH05	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001AE	C0MCTRL05	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001C0	C0MDATA006	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C0	C0MDATA0106	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001C1	C0MDATA106	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C2	C0MDATA206	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C2	C0MDATA2306	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001C3	C0MDATA306	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C4	C0MDATA406	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C4	C0MDATA4506	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001C5	C0MDATA506	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (5/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000001C6	C0MDATA606	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C6	C0MDATA6706	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001C7	C0MDATA706	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C8	C0MDLC06	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001C9	C0MCONF06	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001CA	C0MIDL06	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001CC	C0MIDH06	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001CE	C0MCTRL06	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001E0	C0MDATA007	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E0	C0MDATA0107	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001E1	C0MDATA107	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E2	C0MDATA207	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E2	C0MDATA2307	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001E3	C0MDATA307	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E4	C0MDATA407	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E4	C0MDATA4507	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001E5	C0MDATA507	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E6	C0MDATA607	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E6	C0MDATA6707	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001E7	C0MDATA707	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E8	C0MDLC07	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001E9	C0MCONF07	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000001EA	C0MIDL07	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001EC	C0MIDH07	CAN0 data buffer register	-	-	R/W	-	undef.
0x000001EE	C0MCTRL07	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000200	C0MDATA008	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000200	C0MDATA0108	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000201	C0MDATA108	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000202	C0MDATA208	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000202	C0MDATA2308	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000203	C0MDATA308	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000204	C0MDATA408	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000204	C0MDATA4508	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000205	C0MDATA508	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000206	C0MDATA608	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000206	C0MDATA6708	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000207	C0MDATA708	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000208	C0MDLC08	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000209	C0MCONF08	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000020A	C0MIDL08	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000020C	C0MIDH08	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (6/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000020E	C0MCTRL08	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000220	C0MDATA009	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000220	C0MDATA0109	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000221	C0MDATA109	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000222	C0MDATA209	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000222	C0MDATA2309	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000223	C0MDATA309	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000224	C0MDATA409	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000224	C0MDATA4509	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000225	C0MDATA509	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000226	C0MDATA609	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000226	C0MDATA6709	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000227	C0MDATA709	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000228	C0MDLC09	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000229	C0MCONF09	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000022A	C0MIDL09	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000022C	C0MIDH09	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000022E	C0MCTRL09	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000240	C0MDATA010	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000240	C0MDATA0110	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000241	C0MDATA110	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000242	C0MDATA210	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000242	C0MDATA2310	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000243	C0MDATA310	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000244	C0MDATA410	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000244	C0MDATA4510	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000245	C0MDATA510	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000246	C0MDATA610	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000246	C0MDATA6710	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000247	C0MDATA710	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000248	C0MDLC10	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000249	C0MCONF10	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000024A	C0MIDL10	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000024C	C0MIDH10	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000024E	C0MCTRL10	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000260	C0MDATA011	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000260	C0MDATA0111	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000261	C0MDATA111	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000262	C0MDATA211	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000262	C0MDATA2311	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000263	C0MDATA311	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (7/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000264	C0MDATA411	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000264	C0MDATA4511	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000265	C0MDATA511	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000266	C0MDATA611	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000266	C0MDATA6711	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000267	C0MDATA711	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000268	C0MDLC11	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000269	C0MCONF11	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000026A	C0MIDL11	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000026C	C0MIDH11	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000026E	C0MCTRL11	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000280	C0MDATA012	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000280	C0MDATA0112	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000281	C0MDATA112	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000282	C0MDATA212	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000282	C0MDATA2312	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000283	C0MDATA312	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000284	C0MDATA412	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000284	C0MDATA4512	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000285	C0MDATA512	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000286	C0MDATA612	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000286	C0MDATA6712	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000287	C0MDATA712	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000288	C0MDLC12	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000289	C0MCONF12	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000028A	C0MIDL12	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000028C	C0MIDH12	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000028E	C0MCTRL12	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002A0	C0MDATA013	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A0	C0MDATA0113	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002A1	C0MDATA113	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A2	C0MDATA213	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A2	C0MDATA2313	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002A3	C0MDATA313	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A4	C0MDATA413	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A4	C0MDATA4513	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002A5	C0MDATA513	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A6	C0MDATA613	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A6	C0MDATA6713	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002A7	C0MDATA713	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002A8	C0MDLC13	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (8/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000002A9	C0MCONF13	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002AA	C0MIDL13	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002AC	C0MIDH13	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002AE	C0MCTRL13	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002C0	C0MDATA014	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C0	C0MDATA0114	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002C1	C0MDATA114	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C2	C0MDATA214	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C2	C0MDATA2314	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002C3	C0MDATA314	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C4	C0MDATA414	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C4	C0MDATA4514	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002C5	C0MDATA514	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C6	C0MDATA614	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C6	C0MDATA6714	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002C7	C0MDATA714	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C8	C0MDLC14	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002C9	C0MCONF14	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002CA	C0MIDL14	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002CC	C0MIDH14	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002CE	C0MCTRL14	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002E0	C0MDATA015	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E0	C0MDATA0115	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002E1	C0MDATA115	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E2	C0MDATA215	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E2	C0MDATA2315	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002E3	C0MDATA315	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E4	C0MDATA415	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E4	C0MDATA4515	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002E5	C0MDATA515	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E6	C0MDATA615	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E6	C0MDATA6715	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002E7	C0MDATA715	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E8	C0MDLC15	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002E9	C0MCONF15	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000002EA	C0MIDL15	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002EC	C0MIDH15	CAN0 data buffer register	-	-	R/W	-	undef.
0x000002EE	C0MCTRL15	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000300	C0MDATA016	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000300	C0MDATA0116	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000301	C0MDATA116	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (9/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000302	C0MDATA216	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000302	C0MDATA2316	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000303	C0MDATA316	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000304	C0MDATA416	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000304	C0MDATA4516	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000305	C0MDATA516	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000306	C0MDATA616	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000306	C0MDATA6716	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000307	C0MDATA716	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000308	C0MDLC16	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000309	C0MCONF16	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000030A	C0MIDL16	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000030C	C0MIDH16	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000030E	C0MCTRL16	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000320	C0MDATA017	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000320	C0MDATA0117	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000321	C0MDATA117	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000322	C0MDATA217	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000322	C0MDATA2317	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000323	C0MDATA317	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000324	C0MDATA417	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000324	C0MDATA4517	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000325	C0MDATA517	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000326	C0MDATA617	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000326	C0MDATA6717	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000327	C0MDATA717	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000328	C0MDLC17	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000329	C0MCONF17	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000032A	C0MIDL17	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000032C	C0MIDH17	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000032E	C0MCTRL17	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000340	C0MDATA018	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000340	C0MDATA0118	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000341	C0MDATA118	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000342	C0MDATA218	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000342	C0MDATA2318	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000343	C0MDATA318	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000344	C0MDATA418	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000344	C0MDATA4518	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000345	C0MDATA518	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000346	C0MDATA618	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (10/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000346	C0MDATA6718	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000347	C0MDATA718	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000348	C0MDLC18	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000349	C0MCONF18	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000034A	C0MIDL18	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000034C	C0MIDH18	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000034E	C0MCTRL18	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000360	C0MDATA019	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000360	C0MDATA0191	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000361	C0MDATA191	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000362	C0MDATA219	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000362	C0MDATA2319	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000363	C0MDATA319	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000364	C0MDATA419	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000364	C0MDATA4519	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000365	C0MDATA519	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000366	C0MDATA619	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000366	C0MDATA6719	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000367	C0MDATA719	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000368	C0MDLC19	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000369	C0MCONF19	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000036A	C0MIDL19	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000036C	C0MIDH19	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000036E	C0MCTRL19	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000380	C0MDATA020	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000380	C0MDATA0120	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000381	C0MDATA120	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000382	C0MDATA220	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000382	C0MDATA2320	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000383	C0MDATA320	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000384	C0MDATA420	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000384	C0MDATA4520	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000385	C0MDATA520	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000386	C0MDATA620	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000386	C0MDATA6720	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000387	C0MDATA720	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000388	C0MDLC20	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000389	C0MCONF20	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000038A	C0MIDL20	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000038C	C0MIDH20	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000038E	C0MCTRL20	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (11/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000003A0	C0MDATA021	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A0	C0MDATA0121	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003A1	C0MDATA121	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A2	C0MDATA221	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A2	C0MDATA2321	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003A3	C0MDATA321	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A4	C0MDATA421	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A4	C0MDATA4521	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003A5	C0MDATA521	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A6	C0MDATA621	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A6	C0MDATA6721	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003A7	C0MDATA721	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A8	C0MDLC21	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003A9	C0MCONF21	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003AA	C0MIDL21	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003AC	C0MIDH21	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003AE	C0MCTRL21	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003C0	C0MDATA022	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C0	C0MDATA0122	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003C1	C0MDATA122	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C2	C0MDATA222	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C2	C0MDATA2322	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003C3	C0MDATA322	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C4	C0MDATA422	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C4	C0MDATA4522	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003C5	C0MDATA522	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C6	C0MDATA622	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C6	C0MDATA6722	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003C7	C0MDATA722	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C8	C0MDLC22	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003C9	C0MCONF22	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003CA	C0MIDL22	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003CC	C0MIDH22	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003CE	C0MCTRL22	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003E0	C0MDATA023	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E0	C0MDATA0123	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003E1	C0MDATA123	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E2	C0MDATA223	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E2	C0MDATA2323	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003E3	C0MDATA323	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E4	C0MDATA423	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (12/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000003E4	C0MDATA4523	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003E5	C0MDATA523	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E6	C0MDATA623	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E6	C0MDATA6723	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003E7	C0MDATA723	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E8	C0MDLC23	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003E9	C0MCONF23	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000003EA	C0MIDL23	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003EC	C0MIDH23	CAN0 data buffer register	-	-	R/W	-	undef.
0x000003EE	C0MCTRL23	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000400	C0MDATA024	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000400	C0MDATA0124	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000401	C0MDATA124	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000402	C0MDATA224	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000402	C0MDATA2324	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000403	C0MDATA324	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000404	C0MDATA424	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000404	C0MDATA4524	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000405	C0MDATA524	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000406	C0MDATA624	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000406	C0MDATA6724	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000407	C0MDATA724	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000408	C0MDLC24	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000409	C0MCONF24	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000040A	C0MIDL24	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000040C	C0MIDH24	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000040E	C0MCTRL24	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000420	C0MDATA025	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000420	C0MDATA0125	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000421	C0MDATA125	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000422	C0MDATA225	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000422	C0MDATA2325	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000423	C0MDATA325	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000424	C0MDATA425	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000424	C0MDATA4525	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000425	C0MDATA525	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000426	C0MDATA625	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000426	C0MDATA6725	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000427	C0MDATA725	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000428	C0MDLC25	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000429	C0MCONF25	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (13/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000042A	C0MIDL25	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000042C	C0MIDH25	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000042E	C0MCTRL25	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000440	C0MDATA026	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000440	C0MDATA0126	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000441	C0MDATA126	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000442	C0MDATA226	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000442	C0MDATA2326	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000443	C0MDATA326	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000444	C0MDATA426	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000444	C0MDATA4526	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000445	C0MDATA526	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000446	C0MDATA626	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000446	C0MDATA6726	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000447	C0MDATA726	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000448	C0MDLC26	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000449	C0MCONF26	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000044A	C0MIDL26	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000044C	C0MIDH26	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000044E	C0MCTRL26	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000460	C0MDATA027	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000460	C0MDATA0271	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000461	C0MDATA271	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000462	C0MDATA227	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000462	C0MDATA2327	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000463	C0MDATA327	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000464	C0MDATA427	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000464	C0MDATA4527	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000465	C0MDATA527	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000466	C0MDATA627	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000466	C0MDATA6727	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000467	C0MDATA727	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000468	C0MDLC27	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000469	C0MCONF27	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000046A	C0MIDL27	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000046C	C0MIDH27	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000046E	C0MCTRL27	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000480	C0MDATA028	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000480	C0MDATA0128	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000481	C0MDATA128	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000482	C0MDATA228	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (14/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000482	C0MDATA2328	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000483	C0MDATA328	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000484	C0MDATA428	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000484	C0MDATA4528	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000485	C0MDATA528	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000486	C0MDATA628	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000486	C0MDATA6728	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000487	C0MDATA728	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000488	C0MDLC28	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000489	C0MCONF28	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000048A	C0MIDL28	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000048C	C0MIDH28	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000048E	C0MCTRL28	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004A0	C0MDATA029	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A0	C0MDATA0129	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004A1	C0MDATA129	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A2	C0MDATA229	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A2	C0MDATA2329	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004A3	C0MDATA329	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A4	C0MDATA429	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A4	C0MDATA4529	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004A5	C0MDATA529	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A6	C0MDATA629	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A6	C0MDATA6729	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004A7	C0MDATA729	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A8	C0MDLC29	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004A9	C0MCONF29	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004AA	C0MIDL29	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004AC	C0MIDH29	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004AE	C0MCTRL29	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004C0	C0MDATA030	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C0	C0MDATA0130	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004C1	C0MDATA130	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C2	C0MDATA230	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C2	C0MDATA2330	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004C3	C0MDATA330	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C4	C0MDATA430	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C4	C0MDATA4530	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004C5	C0MDATA530	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C6	C0MDATA630	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C6	C0MDATA6730	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (15/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000004C7	C0MDATA730	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C8	C0MDLC30	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004C9	C0MCONF30	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004CA	C0MIDL30	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004CC	C0MIDH30	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004CE	C0MCTRL30	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004E0	C0MDATA031	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E0	C0MDATA0131	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004E1	C0MDATA131	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E2	C0MDATA231	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E2	C0MDATA2331	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004E3	C0MDATA331	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E4	C0MDATA431	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E4	C0MDATA4531	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004E5	C0MDATA531	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E6	C0MDATA631	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E6	C0MDATA6731	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004E7	C0MDATA731	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E8	C0MDLC31	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004E9	C0MCONF31	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000004EA	C0MIDL31	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004EC	C0MIDH31	CAN0 data buffer register	-	-	R/W	-	undef.
0x000004EE	C0MCTRL31	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000500	C0MDATA032	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000500	C0MDATA0132	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000501	C0MDATA132	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000502	C0MDATA232	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000502	C0MDATA2332	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000503	C0MDATA332	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000504	C0MDATA432	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000504	C0MDATA4532	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000505	C0MDATA532	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000506	C0MDATA632	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000506	C0MDATA6732	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000507	C0MDATA732	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000508	C0MDLC32	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000509	C0MCONF32	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000050A	C0MIDL32	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000050C	C0MIDH32	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000050E	C0MCTRL32	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000520	C0MDATA033	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (16/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000520	C0MDATA0133	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000521	C0MDATA133	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000522	C0MDATA233	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000522	C0MDATA2333	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000523	C0MDATA333	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000524	C0MDATA433	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000524	C0MDATA4533	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000525	C0MDATA533	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000526	C0MDATA633	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000526	C0MDATA6733	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000527	C0MDATA733	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000528	C0MDLC33	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000529	C0MCONF33	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000052A	C0MIDL33	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000052C	C0MIDH33	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000052E	C0MCTRL33	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000540	C0MDATA034	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000540	C0MDATA0134	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000541	C0MDATA134	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000542	C0MDATA234	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000542	C0MDATA2334	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000543	C0MDATA334	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000544	C0MDATA434	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000544	C0MDATA4534	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000545	C0MDATA534	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000546	C0MDATA634	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000546	C0MDATA6734	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000547	C0MDATA734	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000548	C0MDLC34	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000549	C0MCONF34	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000054A	C0MIDL34	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000054C	C0MIDH34	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000054E	C0MCTRL34	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000560	C0MDATA035	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000560	C0MDATA0351	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000561	C0MDATA351	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000562	C0MDATA235	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000562	C0MDATA2335	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000563	C0MDATA335	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000564	C0MDATA435	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000564	C0MDATA4535	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (17/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000565	C0MDATA535	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000566	C0MDATA635	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000566	C0MDATA6735	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000567	C0MDATA735	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000568	C0MDLC35	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000569	C0MCONF35	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000056A	C0MIDL35	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000056C	C0MIDH35	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000056E	C0MCTRL35	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000580	C0MDATA036	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000580	C0MDATA0136	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000581	C0MDATA136	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000582	C0MDATA236	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000582	C0MDATA2336	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000583	C0MDATA336	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000584	C0MDATA436	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000584	C0MDATA4536	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000585	C0MDATA536	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000586	C0MDATA636	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000586	C0MDATA6736	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000587	C0MDATA736	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000588	C0MDLC36	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000589	C0MCONF36	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000058A	C0MIDL36	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000058C	C0MIDH36	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000058E	C0MCTRL36	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005A0	C0MDATA037	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A0	C0MDATA0137	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005A1	C0MDATA137	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A2	C0MDATA237	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A2	C0MDATA2337	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005A3	C0MDATA337	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A4	C0MDATA437	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A4	C0MDATA4537	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005A5	C0MDATA537	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A6	C0MDATA637	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A6	C0MDATA6737	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005A7	C0MDATA737	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A8	C0MDLC37	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005A9	C0MCONF37	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005AA	C0MIDL37	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (18/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000005AC	C0MIDH37	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005AE	C0MCTRL37	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005C0	C0MDATA038	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C0	C0MDATA0138	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005C1	C0MDATA138	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C2	C0MDATA238	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C2	C0MDATA2338	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005C3	C0MDATA338	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C4	C0MDATA438	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C4	C0MDATA4538	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005C5	C0MDATA538	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C6	C0MDATA638	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C6	C0MDATA6738	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005C7	C0MDATA738	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C8	C0MDLC38	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005C9	C0MCONF38	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005CA	C0MIDL38	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005CC	C0MIDH38	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005CE	C0MCTRL38	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005E0	C0MDATA039	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E0	C0MDATA0139	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005E1	C0MDATA139	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E2	C0MDATA239	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E2	C0MDATA2339	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005E3	C0MDATA339	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E4	C0MDATA439	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E4	C0MDATA4539	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005E5	C0MDATA539	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E6	C0MDATA639	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E6	C0MDATA6739	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005E7	C0MDATA739	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E8	C0MDLC39	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005E9	C0MCONF39	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000005EA	C0MIDL39	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005EC	C0MIDH39	CAN0 data buffer register	-	-	R/W	-	undef.
0x000005EE	C0MCTRL39	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000600	C0MDATA040	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000600	C0MDATA0140	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000601	C0MDATA140	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000602	C0MDATA240	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000602	C0MDATA2340	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (19/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000603	C0MDATA340	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000604	C0MDATA440	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000604	C0MDATA4540	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000605	C0MDATA540	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000606	C0MDATA640	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000606	C0MDATA6740	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000607	C0MDATA740	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000608	C0MDLC40	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000609	C0MCONF40	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000060A	C0MIDL40	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000060C	C0MIDH40	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000060E	C0MCTRL40	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000620	C0MDATA041	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000620	C0MDATA0141	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000621	C0MDATA141	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000622	C0MDATA241	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000622	C0MDATA2341	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000623	C0MDATA341	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000624	C0MDATA441	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000624	C0MDATA4541	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000625	C0MDATA541	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000626	C0MDATA641	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000626	C0MDATA6741	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000627	C0MDATA741	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000628	C0MDLC41	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000629	C0MCONF41	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000062A	C0MIDL41	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000062C	C0MIDH41	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000062E	C0MCTRL41	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000640	C0MDATA042	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000640	C0MDATA0142	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000641	C0MDATA142	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000642	C0MDATA242	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000642	C0MDATA2342	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000643	C0MDATA342	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000644	C0MDATA442	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000644	C0MDATA4542	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000645	C0MDATA542	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000646	C0MDATA642	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000646	C0MDATA6742	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000647	C0MDATA742	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (20/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000648	C0MDLC42	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000649	C0MCONF42	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000064A	C0MIDL42	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000064C	C0MIDH42	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000064E	C0MCTRL42	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000660	C0MDATA043	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000660	C0MDATA0143	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000661	C0MDATA143	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000662	C0MDATA243	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000662	C0MDATA2343	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000663	C0MDATA343	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000664	C0MDATA443	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000664	C0MDATA4543	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000665	C0MDATA543	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000666	C0MDATA643	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000666	C0MDATA6743	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000667	C0MDATA743	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000668	C0MDLC43	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000669	C0MCONF43	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000066A	C0MIDL43	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000066C	C0MIDH43	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000066E	C0MCTRL43	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000680	C0MDATA044	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000680	C0MDATA0144	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000681	C0MDATA144	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000682	C0MDATA244	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000682	C0MDATA2344	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000683	C0MDATA344	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000684	C0MDATA444	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000684	C0MDATA4544	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000685	C0MDATA544	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000686	C0MDATA644	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000686	C0MDATA6744	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000687	C0MDATA744	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000688	C0MDLC44	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x00000689	C0MCONF44	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x0000068A	C0MIDL44	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000068C	C0MIDH44	CAN0 data buffer register	-	-	R/W	-	undef.
0x0000068E	C0MCTRL44	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006A0	C0MDATA045	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A0	C0MDATA0145	CAN0 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (21/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000006A1	C0MDATA145	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A2	C0MDATA245	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A2	C0MDATA2345	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006A3	C0MDATA345	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A4	C0MDATA445	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A4	C0MDATA4545	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006A5	C0MDATA545	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A6	C0MDATA645	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A6	C0MDATA6745	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006A7	C0MDATA745	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A8	C0MDLC45	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006A9	C0MCONF45	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006AA	C0MIDL45	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006AC	C0MIDH45	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006AE	C0MCTRL45	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006C0	C0MDATA046	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C0	C0MDATA0146	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006C1	C0MDATA146	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C2	C0MDATA246	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C2	C0MDATA2346	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006C3	C0MDATA346	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C4	C0MDATA446	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C4	C0MDATA4546	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006C5	C0MDATA546	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C6	C0MDATA646	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C6	C0MDATA6746	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006C7	C0MDATA746	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C8	C0MDLC46	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006C9	C0MCONF46	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006CA	C0MIDL46	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006CC	C0MIDH46	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006CE	C0MCTRL46	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006E0	C0MDATA047	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E0	C0MDATA0147	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006E1	C0MDATA147	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E2	C0MDATA247	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E2	C0MDATA2347	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006E3	C0MDATA347	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E4	C0MDATA447	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E4	C0MDATA4547	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006E5	C0MDATA547	CAN0 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (22/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000006E6	C0MDATA647	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E6	C0MDATA6747	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006E7	C0MDATA747	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E8	C0MDLC47	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006E9	C0MCONF47	CAN0 data buffer register	R/W	R/W	-	-	undef.
0x000006EA	C0MIDL47	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006EC	C0MIDH47	CAN0 data buffer register	-	-	R/W	-	undef.
0x000006EE	C0MCTRL47	CAN0 data buffer register	-	-	R/W	-	undef.
0x00000800	C1GMCTRLL	CAN1 Global Macro Control Register low byte	R/W	R/W	-	-	0x00
0x00000800	C1GMCTRL	CAN1 Global Macro Control Register	-	-	R/W	-	0x0000
0x00000801	C1GMCTRLH	CAN1 Global Macro Control Register high byte	R/W	R/W	-	-	0x00
0x00000802	C1GMCS	CAN1 Global Macro Clock Selection Register	R/W	R/W	-	-	0x0F
0x00000004	C1GMCONF	CAN1 global configuration register	-	-	R	-	0x0019
0x00000806	C1GMABTL	CAN1 Global Macro Automatic Block Transmission Register low byte	R/W	R/W	-	-	0x00
0x00000806	C1GMABT	CAN1 Global Macro Automatic Block Transmission Register	-	-	R/W	-	0x0000
0x00000807	C1GMABTH	CAN1 Global Macro Automatic Block Transmission Register high byte	R/W	R/W	-	-	0x00
0x00000808	C1GMABTD	CAN1 Global Macro Automatic Block Transmission Delay Register	R/W	R/W	-	-	0x00
0x00000840	C1MASK1L	CAN1 Module Mask 1 Register lower half word	-	-	R/W	-	undef.
0x00000842	C1MASK1H	CAN1 Module Mask 1 Register upper half word	-	-	R/W	-	undef.
0x00000844	C1MASK2L	CAN1 Module Mask 2 Register lower half word	-	-	R/W	-	undef.
0x00000846	C1MASK2H	CAN1 Module Mask 2 Register upper half word	-	-	R/W	-	undef.
0x00000848	C1MASK3L	CAN1 Module Mask 3 Register lower half word	-	-	R/W	-	undef.
0x0000084A	C1MASK3H	CAN1 Module Mask 3 Register upper half word	-	-	R/W	-	undef.
0x0000084C	C1MASK4L	CAN1 Module Mask 4 Register lower half word	-	-	R/W	-	undef.
0x0000084E	C1MASK4H	CAN1 Module Mask 4 Register upper half word	-	-	R/W	-	undef.
0x00000850	C1CTRL	CAN1 Module Control Register	-	-	R/W	-	0x0000
0x00000852	C1LEC	CAN1 Module Last Error Code Register	R/W	R/W	-	-	0x00
0x00000853	C1INFO	CAN1 Module Information Register	R	R	-	-	0x00
0x00000854	C1ERC	CAN1 Module Error Counter	-	-	R	-	0x0000
0x00000856	C1IEL	CAN1 Module Interrupt Enable Register low byte	R/W	R/W	-	-	0x00
0x00000856	C1IE	CAN1 Module Interrupt Enable Register	-	-	R/W	-	0x0000
0x00000857	C1IEH	CAN1 Module Interrupt Enable Register high byte	R/W	R/W	-	-	0x00
0x00000858	C1INTSL	CAN1 Module Interrupt Status Register low byte	R/W	R/W	-	-	0x00
0x00000858	C1INTS	CAN1 Module Interrupt Status Register	-	-	R/W	-	0x0000

Table 3-7: PPA related SFR Table (23/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000085A	C1BRP	CAN1 Module Bit-Rate Prescaler Register	R/W	R/W	-	-	0xFF
0x0000085C	C1BTR	CAN1 Bit Rate Register	-	-	R/W	-	0x370F
0x0000085E	C1LIPT	CAN1 Module Last In-Pointer Register	-	R/W	-	-	undef.
0x00000860	C1RGPTL	CAN1 Module Receive History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00000860	C1RGPT	CAN1 Module Receive History List Get Pointer Register	-	-	R/W	-	undef.
0x00000862	C1LOPT	CAN1 Module Last Out-Pointer Register	-	R	-	-	undef.
0x00000864	C1TGPTL	CAN1 Module Transmit History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00000864	C1TGPT	CAN1 Module Transmit History List Get Pointer Register	-	-	R/W	-	undef.
0x00000866	C1TSL	CAN1 Module Time Stamp Register low byte	R/W	R/W	-	-	0x00
0x00000866	C1TS	CAN1 Module Time Stamp Register	-	-	R/W	-	0x0000
0x00000867	C1TSH	CAN1 Module Time Stamp Register high byte	R/W	R/W	-	-	0x00
0x00000900	C1MDATA000	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000900	C1MDATA0100	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000901	C1MDATA100	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000902	C1MDATA200	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000902	C1MDATA2300	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000903	C1MDATA300	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000904	C1MDATA400	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000904	C1MDATA4500	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000905	C1MDATA500	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000906	C1MDATA600	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000906	C1MDATA6700	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000907	C1MDATA700	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000908	C1MDLC00	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000909	C1MCONF00	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x0000090A	C1MIDL00	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000090C	C1MIDH00	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000090E	C1MCTRL00	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000920	C1MDATA001	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000920	C1MDATA0101	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000921	C1MDATA101	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000922	C1MDATA201	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000922	C1MDATA2301	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000923	C1MDATA301	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000924	C1MDATA401	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000924	C1MDATA4501	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000925	C1MDATA501	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (24/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000926	C1MDATA601	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000926	C1MDATA6701	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000927	C1MDATA701	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000928	C1MDLC01	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000929	C1MCONF01	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x0000092A	C1MIDL01	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000092C	C1MIDH01	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000092E	C1MCTRL01	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000940	C1MDATA002	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000940	C1MDATA0102	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000941	C1MDATA102	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000942	C1MDATA202	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000942	C1MDATA2302	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000943	C1MDATA302	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000944	C1MDATA402	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000944	C1MDATA4502	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000945	C1MDATA502	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000946	C1MDATA602	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000946	C1MDATA6702	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000947	C1MDATA702	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000948	C1MDLC02	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000949	C1MCONF02	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x0000094A	C1MIDL02	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000094C	C1MIDH02	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000094E	C1MCTRL02	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000960	C1MDATA003	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000960	C1MDATA0103	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000961	C1MDATA103	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000962	C1MDATA203	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000962	C1MDATA2303	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000963	C1MDATA303	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000964	C1MDATA403	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000964	C1MDATA4503	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000965	C1MDATA503	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000966	C1MDATA603	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000966	C1MDATA6703	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000967	C1MDATA703	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000968	C1MDLC03	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000969	C1MCONF03	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x0000096A	C1MIDL03	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000096C	C1MIDH03	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (25/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000096E	C1MCTRL03	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000980	C1MDATA004	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000980	C1MDATA0104	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000981	C1MDATA104	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000982	C1MDATA204	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000982	C1MDATA2304	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000983	C1MDATA304	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000984	C1MDATA404	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000984	C1MDATA4504	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000985	C1MDATA504	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000986	C1MDATA604	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000986	C1MDATA6704	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000987	C1MDATA704	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000988	C1MDLC04	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000989	C1MCONF04	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x0000098A	C1MIDL04	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000098C	C1MIDH04	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000098E	C1MCTRL04	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009A0	C1MDATA005	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A0	C1MDATA0105	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009A1	C1MDATA105	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A2	C1MDATA205	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A2	C1MDATA2305	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009A3	C1MDATA305	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A4	C1MDATA405	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A4	C1MDATA4505	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009A5	C1MDATA505	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A6	C1MDATA605	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A6	C1MDATA6705	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009A7	C1MDATA705	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A8	C1MDLC05	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009A9	C1MCONF05	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009AA	C1MIDL05	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009AC	C1MIDH05	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009AE	C1MCTRL05	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009C0	C1MDATA006	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C0	C1MDATA0106	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009C1	C1MDATA106	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C2	C1MDATA206	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C2	C1MDATA2306	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009C3	C1MDATA306	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (26/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000009C4	C1MDATA406	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C4	C1MDATA4506	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009C5	C1MDATA506	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C6	C1MDATA606	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C6	C1MDATA6706	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009C7	C1MDATA706	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C8	C1MDLC06	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009C9	C1MCONF06	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009CA	C1MIDL06	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009CC	C1MIDH06	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009CE	C1MCTRL06	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009E0	C1MDATA007	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E0	C1MDATA0107	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009E1	C1MDATA107	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E2	C1MDATA207	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E2	C1MDATA2307	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009E3	C1MDATA307	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E4	C1MDATA407	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E4	C1MDATA4507	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009E5	C1MDATA507	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E6	C1MDATA607	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E6	C1MDATA6707	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009E7	C1MDATA707	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E8	C1MDLC07	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009E9	C1MCONF07	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x000009EA	C1MIDL07	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009EC	C1MIDH07	CAN1 data buffer register	-	-	R/W	-	undef.
0x000009EE	C1MCTRL07	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A00	C1MDATA008	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A00	C1MDATA0108	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A01	C1MDATA108	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A02	C1MDATA208	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A02	C1MDATA2308	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A03	C1MDATA308	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A04	C1MDATA408	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A04	C1MDATA4508	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A05	C1MDATA508	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A06	C1MDATA608	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A06	C1MDATA6708	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A07	C1MDATA708	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A08	C1MDLC08	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (27/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000A09	C1MCONF08	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A0A	C1MIDL08	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A0C	C1MIDH08	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A0E	C1MCTRL08	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A20	C1MDATA009	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A20	C1MDATA0109	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A21	C1MDATA109	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A22	C1MDATA209	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A22	C1MDATA2309	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A23	C1MDATA309	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A24	C1MDATA409	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A24	C1MDATA4509	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A25	C1MDATA509	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A26	C1MDATA609	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A26	C1MDATA6709	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A27	C1MDATA709	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A28	C1MDLC09	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A29	C1MCONF09	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A2A	C1MIDL09	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A2C	C1MIDH09	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A2E	C1MCTRL09	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A40	C1MDATA010	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A40	C1MDATA0110	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A41	C1MDATA110	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A42	C1MDATA210	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A42	C1MDATA2310	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A43	C1MDATA310	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A44	C1MDATA410	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A44	C1MDATA4510	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A45	C1MDATA510	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A46	C1MDATA610	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A46	C1MDATA6710	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A47	C1MDATA710	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A48	C1MDLC10	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A49	C1MCONF10	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A4A	C1MIDL10	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A4C	C1MIDH10	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A4E	C1MCTRL10	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A60	C1MDATA011	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A60	C1MDATA0111	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A61	C1MDATA111	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (28/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000A62	C1MDATA211	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A62	C1MDATA2311	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A63	C1MDATA311	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A64	C1MDATA411	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A64	C1MDATA4511	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A65	C1MDATA511	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A66	C1MDATA611	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A66	C1MDATA6711	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A67	C1MDATA711	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A68	C1MDLC11	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A69	C1MCONF11	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A6A	C1MIDL11	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A6C	C1MIDH11	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A6E	C1MCTRL11	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A80	C1MDATA012	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A80	C1MDATA0112	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A81	C1MDATA112	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A82	C1MDATA212	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A82	C1MDATA2312	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A83	C1MDATA312	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A84	C1MDATA412	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A84	C1MDATA4512	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A85	C1MDATA512	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A86	C1MDATA612	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A86	C1MDATA6712	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A87	C1MDATA712	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A88	C1MDLC12	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A89	C1MCONF12	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000A8A	C1MIDL12	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A8C	C1MIDH12	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000A8E	C1MCTRL12	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AA0	C1MDATA013	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA0	C1MDATA0113	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AA1	C1MDATA113	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA2	C1MDATA213	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA2	C1MDATA2313	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AA3	C1MDATA313	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA4	C1MDATA413	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA4	C1MDATA4513	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AA5	C1MDATA513	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA6	C1MDATA613	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (29/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000AA6	C1MDATA6713	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AA7	C1MDATA713	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA8	C1MDLC13	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AA9	C1MCONF13	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AAA	C1MIDL13	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AAC	C1MIDH13	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AAE	C1MCTRL13	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AC0	C1MDATA014	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC0	C1MDATA0114	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AC1	C1MDATA114	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC2	C1MDATA214	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC2	C1MDATA2314	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AC3	C1MDATA314	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC4	C1MDATA414	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC4	C1MDATA4514	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AC5	C1MDATA514	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC6	C1MDATA614	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC6	C1MDATA6714	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AC7	C1MDATA714	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC8	C1MDLC14	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AC9	C1MCONF14	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000ACA	C1MIDL14	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000ACC	C1MIDH14	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000ACE	C1MCTRL14	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AE0	C1MDATA015	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE0	C1MDATA0115	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AE1	C1MDATA115	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE2	C1MDATA215	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE2	C1MDATA2315	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AE3	C1MDATA315	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE4	C1MDATA415	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE4	C1MDATA4515	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AE5	C1MDATA515	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE6	C1MDATA615	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE6	C1MDATA6715	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AE7	C1MDATA715	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE8	C1MDLC15	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AE9	C1MCONF15	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000AEA	C1MIDL15	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AEC	C1MIDH15	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000AEE	C1MCTRL15	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (30/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000B00	C1MDATA016	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B00	C1MDATA0116	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B01	C1MDATA116	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B02	C1MDATA216	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B02	C1MDATA2316	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B03	C1MDATA316	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B04	C1MDATA416	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B04	C1MDATA4516	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B05	C1MDATA516	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B06	C1MDATA616	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B06	C1MDATA6716	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B07	C1MDATA716	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B08	C1MDLC16	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B09	C1MCONF16	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B0A	C1MIDL16	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B0C	C1MIDH16	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B0E	C1MCTRL16	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B20	C1MDATA017	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B20	C1MDATA0117	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B21	C1MDATA117	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B22	C1MDATA217	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B22	C1MDATA2317	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B23	C1MDATA317	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B24	C1MDATA417	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B24	C1MDATA4517	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B25	C1MDATA517	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B26	C1MDATA617	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B26	C1MDATA6717	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B27	C1MDATA717	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B28	C1MDLC17	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B29	C1MCONF17	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B2A	C1MIDL17	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B2C	C1MIDH17	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B2E	C1MCTRL17	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B40	C1MDATA018	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B40	C1MDATA0118	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B41	C1MDATA118	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B42	C1MDATA218	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B42	C1MDATA2318	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B43	C1MDATA318	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B44	C1MDATA418	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (31/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000B44	C1MDATA4518	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B45	C1MDATA518	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B46	C1MDATA618	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B46	C1MDATA6718	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B47	C1MDATA718	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B48	C1MDLC18	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B49	C1MCONF18	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B4A	C1MIDL18	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B4C	C1MIDH18	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B4E	C1MCTRL18	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B60	C1MDATA019	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B60	C1MDATA0191	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B61	C1MDATA191	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B62	C1MDATA219	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B62	C1MDATA2319	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B63	C1MDATA319	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B64	C1MDATA419	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B64	C1MDATA4519	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B65	C1MDATA519	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B66	C1MDATA619	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B66	C1MDATA6719	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B67	C1MDATA719	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B68	C1MDLC19	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B69	C1MCONF19	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B6A	C1MIDL19	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B6C	C1MIDH19	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B6E	C1MCTRL19	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B80	C1MDATA020	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B80	C1MDATA0120	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B81	C1MDATA120	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B82	C1MDATA220	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B82	C1MDATA2320	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B83	C1MDATA320	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B84	C1MDATA420	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B84	C1MDATA4520	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B85	C1MDATA520	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B86	C1MDATA620	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B86	C1MDATA6720	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B87	C1MDATA720	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B88	C1MDLC20	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000B89	C1MCONF20	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (32/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000B8A	C1MIDL20	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B8C	C1MIDH20	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000B8E	C1MCTRL20	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BA0	C1MDATA021	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA0	C1MDATA0121	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BA1	C1MDATA121	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA2	C1MDATA221	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA2	C1MDATA2321	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BA3	C1MDATA321	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA4	C1MDATA421	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA4	C1MDATA4521	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BA5	C1MDATA521	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA6	C1MDATA621	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA6	C1MDATA6721	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BA7	C1MDATA721	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA8	C1MDLC21	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BA9	C1MCONF21	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BAA	C1MIDL21	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BAC	C1MIDH21	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BAE	C1MCTRL21	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BC0	C1MDATA022	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC0	C1MDATA0122	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BC1	C1MDATA122	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC2	C1MDATA222	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC2	C1MDATA2322	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BC3	C1MDATA322	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC4	C1MDATA422	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC4	C1MDATA4522	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BC5	C1MDATA522	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC6	C1MDATA622	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC6	C1MDATA6722	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BC7	C1MDATA722	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC8	C1MDLC22	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BC9	C1MCONF22	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BCA	C1MIDL22	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BCC	C1MIDH22	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BCE	C1MCTRL22	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BE0	C1MDATA023	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE0	C1MDATA0123	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BE1	C1MDATA123	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE2	C1MDATA223	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (33/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000BE2	C1MDATA2323	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BE3	C1MDATA323	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE4	C1MDATA423	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE4	C1MDATA4523	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BE5	C1MDATA523	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE6	C1MDATA623	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE6	C1MDATA6723	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BE7	C1MDATA723	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE8	C1MDLC23	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BE9	C1MCONF23	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000BEA	C1MIDL23	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BEC	C1MIDH23	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000BEE	C1MCTRL23	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C00	C1MDATA024	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C00	C1MDATA0124	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C01	C1MDATA124	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C02	C1MDATA224	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C02	C1MDATA2324	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C03	C1MDATA324	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C04	C1MDATA424	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C04	C1MDATA4524	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C05	C1MDATA524	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C06	C1MDATA624	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C06	C1MDATA6724	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C07	C1MDATA724	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C08	C1MDLC24	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C09	C1MCONF24	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C0A	C1MIDL24	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C0C	C1MIDH24	CAN1 data buffer register	-	-	R/W	-	undef.
0x0000000E	C1MCTRL24	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000020	C1MDATA025	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C20	C1MDATA0125	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C21	C1MDATA125	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C22	C1MDATA225	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C22	C1MDATA2325	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C23	C1MDATA325	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C24	C1MDATA425	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C24	C1MDATA4525	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C25	C1MDATA525	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C26	C1MDATA625	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C26	C1MDATA6725	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (34/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000C27	C1MDATA725	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C28	C1MDLC25	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C29	C1MCONF25	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C2A	C1MIDL25	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C2C	C1MIDH25	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C2E	C1MCTRL25	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C40	C1MDATA026	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C40	C1MDATA0126	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C41	C1MDATA126	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C42	C1MDATA226	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C42	C1MDATA2326	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C43	C1MDATA326	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C44	C1MDATA426	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C44	C1MDATA4526	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C45	C1MDATA526	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C46	C1MDATA626	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C46	C1MDATA6726	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C47	C1MDATA726	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C48	C1MDLC26	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C49	C1MCONF26	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C4A	C1MIDL26	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C4C	C1MIDH26	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C4E	C1MCTRL26	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C60	C1MDATA027	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C60	C1MDATA0271	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C61	C1MDATA271	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C62	C1MDATA227	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C62	C1MDATA2327	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C63	C1MDATA327	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C64	C1MDATA427	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C64	C1MDATA4527	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C65	C1MDATA527	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C66	C1MDATA627	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C66	C1MDATA6727	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C67	C1MDATA727	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C68	C1MDLC27	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C69	C1MCONF27	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C6A	C1MIDL27	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C6C	C1MIDH27	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C6E	C1MCTRL27	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C80	C1MDATA028	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (35/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000C80	C1MDATA0128	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C81	C1MDATA128	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C82	C1MDATA228	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C82	C1MDATA2328	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C83	C1MDATA328	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C84	C1MDATA428	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C84	C1MDATA4528	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C85	C1MDATA528	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C86	C1MDATA628	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C86	C1MDATA6728	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C87	C1MDATA728	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C88	C1MDLC28	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C89	C1MCONF28	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000C8A	C1MIDL28	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C8C	C1MIDH28	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000C8E	C1MCTRL28	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CA0	C1MDATA029	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA0	C1MDATA0129	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CA1	C1MDATA129	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA2	C1MDATA229	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA2	C1MDATA2329	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CA3	C1MDATA329	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA4	C1MDATA429	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA4	C1MDATA4529	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CA5	C1MDATA529	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA6	C1MDATA629	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA6	C1MDATA6729	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CA7	C1MDATA729	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA8	C1MDLC29	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CA9	C1MCONF29	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CAA	C1MIDL29	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CAC	C1MIDH29	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CAE	C1MCTRL29	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CC0	C1MDATA030	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC0	C1MDATA0130	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CC1	C1MDATA130	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC2	C1MDATA230	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC2	C1MDATA2330	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CC3	C1MDATA330	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC4	C1MDATA430	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC4	C1MDATA4530	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (36/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000CC5	C1MDATA530	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC6	C1MDATA630	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC6	C1MDATA6730	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CC7	C1MDATA730	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC8	C1MDLC30	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CC9	C1MCONF30	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CCA	C1MIDL30	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CCC	C1MIDH30	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CCE	C1MCTRL30	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CE0	C1MDATA031	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE0	C1MDATA0131	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CE1	C1MDATA131	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE2	C1MDATA231	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE2	C1MDATA2331	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CE3	C1MDATA331	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE4	C1MDATA431	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE4	C1MDATA4531	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CE5	C1MDATA531	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE6	C1MDATA631	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE6	C1MDATA6731	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CE7	C1MDATA731	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE8	C1MDLC31	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CE9	C1MCONF31	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000CEA	C1MIDL31	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CEC	C1MIDH31	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000CEE	C1MCTRL31	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D00	C1MDATA032	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D00	C1MDATA0132	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D01	C1MDATA132	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D02	C1MDATA232	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D02	C1MDATA2332	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D03	C1MDATA332	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D04	C1MDATA432	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D04	C1MDATA4532	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D05	C1MDATA532	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D06	C1MDATA632	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D06	C1MDATA6732	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D07	C1MDATA732	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D08	C1MDLC32	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D09	C1MCONF32	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D0A	C1MIDL32	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (37/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000D0C	C1MIDH32	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D0E	C1MCTRL32	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D20	C1MDATA033	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D20	C1MDATA0133	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D21	C1MDATA133	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D22	C1MDATA233	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D22	C1MDATA2333	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D23	C1MDATA333	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D24	C1MDATA433	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D24	C1MDATA4533	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D25	C1MDATA533	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D26	C1MDATA633	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D26	C1MDATA6733	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D27	C1MDATA733	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D28	C1MDLC33	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D29	C1MCONF33	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D2A	C1MIDL33	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D2C	C1MIDH33	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D2E	C1MCTRL33	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D40	C1MDATA034	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D40	C1MDATA0134	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D41	C1MDATA134	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D42	C1MDATA234	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D42	C1MDATA2334	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D43	C1MDATA334	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D44	C1MDATA434	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D44	C1MDATA4534	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D45	C1MDATA534	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D46	C1MDATA634	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D46	C1MDATA6734	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D47	C1MDATA734	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D48	C1MDLC34	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D49	C1MCONF34	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D4A	C1MIDL34	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D4C	C1MIDH34	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D4E	C1MCTRL34	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D60	C1MDATA035	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D60	C1MDATA0351	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D61	C1MDATA351	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D62	C1MDATA235	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D62	C1MDATA2335	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (38/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000D63	C1MDATA335	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D64	C1MDATA435	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D64	C1MDATA4535	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D65	C1MDATA535	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D66	C1MDATA635	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D66	C1MDATA6735	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D67	C1MDATA735	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D68	C1MDLC35	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D69	C1MCONF35	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D6A	C1MIDL35	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D6C	C1MIDH35	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D6E	C1MCTRL35	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D80	C1MDATA036	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D80	C1MDATA0136	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D81	C1MDATA136	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D82	C1MDATA236	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D82	C1MDATA2336	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D83	C1MDATA336	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D84	C1MDATA436	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D84	C1MDATA4536	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D85	C1MDATA536	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D86	C1MDATA636	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D86	C1MDATA6736	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D87	C1MDATA736	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D88	C1MDLC36	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D89	C1MCONF36	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000D8A	C1MIDL36	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D8C	C1MIDH36	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000D8E	C1MCTRL36	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DA0	C1MDATA037	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA0	C1MDATA0137	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DA1	C1MDATA137	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA2	C1MDATA237	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA2	C1MDATA2337	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DA3	C1MDATA337	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA4	C1MDATA437	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA4	C1MDATA4537	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DA5	C1MDATA537	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA6	C1MDATA637	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA6	C1MDATA6737	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DA7	C1MDATA737	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (39/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000DA8	C1MDLC37	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DA9	C1MCONF37	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DAA	C1MIDL37	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DAC	C1MIDH37	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DAE	C1MCTRL37	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DC0	C1MDATA038	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC0	C1MDATA0138	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DC1	C1MDATA138	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC2	C1MDATA238	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC2	C1MDATA2338	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DC3	C1MDATA338	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC4	C1MDATA438	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC4	C1MDATA4538	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DC5	C1MDATA538	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC6	C1MDATA638	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC6	C1MDATA6738	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DC7	C1MDATA738	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC8	C1MDLC38	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DC9	C1MCONF38	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DCA	C1MIDL38	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DCC	C1MIDH38	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DCE	C1MCTRL38	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DE0	C1MDATA039	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE0	C1MDATA0139	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DE1	C1MDATA139	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE2	C1MDATA239	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE2	C1MDATA2339	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DE3	C1MDATA339	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE4	C1MDATA439	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE4	C1MDATA4539	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DE5	C1MDATA539	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE6	C1MDATA639	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE6	C1MDATA6739	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DE7	C1MDATA739	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE8	C1MDLC39	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DE9	C1MCONF39	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000DEA	C1MIDL39	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DEC	C1MIDH39	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000DEE	C1MCTRL39	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E00	C1MDATA040	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E00	C1MDATA0140	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (40/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000E01	C1MDATA140	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E02	C1MDATA240	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E02	C1MDATA2340	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E03	C1MDATA340	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E04	C1MDATA440	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E04	C1MDATA4540	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E05	C1MDATA540	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E06	C1MDATA640	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E06	C1MDATA6740	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E07	C1MDATA740	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E08	C1MDLC40	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E09	C1MCONF40	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E0A	C1MIDL40	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E0C	C1MIDH40	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E0E	C1MCTRL40	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E20	C1MDATA041	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E20	C1MDATA0141	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E21	C1MDATA141	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E22	C1MDATA241	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E22	C1MDATA2341	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E23	C1MDATA341	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E24	C1MDATA441	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E24	C1MDATA4541	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E25	C1MDATA541	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E26	C1MDATA641	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E26	C1MDATA6741	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E27	C1MDATA741	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E28	C1MDLC41	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E29	C1MCONF41	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E2A	C1MIDL41	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E2C	C1MIDH41	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E2E	C1MCTRL41	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E40	C1MDATA042	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E40	C1MDATA0142	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E41	C1MDATA142	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E42	C1MDATA242	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E42	C1MDATA2342	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E43	C1MDATA342	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E44	C1MDATA442	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E44	C1MDATA4542	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E45	C1MDATA542	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (41/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000E46	C1MDATA642	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E46	C1MDATA6742	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E47	C1MDATA742	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E48	C1MDLC42	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E49	C1MCONF42	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E4A	C1MIDL42	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E4C	C1MIDH42	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E4E	C1MCTRL42	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E60	C1MDATA043	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E60	C1MDATA0143	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E61	C1MDATA143	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E62	C1MDATA243	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E62	C1MDATA2343	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E63	C1MDATA343	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E64	C1MDATA443	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E64	C1MDATA4543	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E65	C1MDATA543	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E66	C1MDATA643	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E66	C1MDATA6743	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E67	C1MDATA743	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E68	C1MDLC43	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E69	C1MCONF43	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E6A	C1MIDL43	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E6C	C1MIDH43	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E6E	C1MCTRL43	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E80	C1MDATA044	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E80	C1MDATA0144	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E81	C1MDATA144	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E82	C1MDATA244	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E82	C1MDATA2344	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E83	C1MDATA344	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E84	C1MDATA444	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E84	C1MDATA4544	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E85	C1MDATA544	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E86	C1MDATA644	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E86	C1MDATA6744	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E87	C1MDATA744	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E88	C1MDLC44	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E89	C1MCONF44	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000E8A	C1MIDL44	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000E8C	C1MIDH44	CAN1 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (42/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000E8E	C1MCTRL44	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EA0	C1MDATA045	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA0	C1MDATA0145	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EA1	C1MDATA145	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA2	C1MDATA245	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA2	C1MDATA2345	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EA3	C1MDATA345	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA4	C1MDATA445	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA4	C1MDATA4545	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EA5	C1MDATA545	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA6	C1MDATA645	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA6	C1MDATA6745	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EA7	C1MDATA745	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA8	C1MDLC45	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EA9	C1MCONF45	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EAA	C1MIDL45	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EAC	C1MIDH45	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EAE	C1MCTRL45	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EC0	C1MDATA046	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC0	C1MDATA0146	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EC1	C1MDATA146	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC2	C1MDATA246	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC2	C1MDATA2346	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EC3	C1MDATA346	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC4	C1MDATA446	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC4	C1MDATA4546	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EC5	C1MDATA546	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC6	C1MDATA646	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC6	C1MDATA6746	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EC7	C1MDATA746	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC8	C1MDLC46	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EC9	C1MCONF46	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000ECA	C1MIDL46	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000ECC	C1MIDH46	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000ECE	C1MCTRL46	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EE0	C1MDATA047	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE0	C1MDATA0147	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EE1	C1MDATA147	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE2	C1MDATA247	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE2	C1MDATA2347	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EE3	C1MDATA347	CAN1 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (43/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00000EE4	C1MDATA447	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE4	C1MDATA4547	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EE5	C1MDATA547	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE6	C1MDATA647	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE6	C1MDATA6747	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EE7	C1MDATA747	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE8	C1MDLC47	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EE9	C1MCONF47	CAN1 data buffer register	R/W	R/W	-	-	undef.
0x00000EEA	C1MIDL47	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EEC	C1MIDH47	CAN1 data buffer register	-	-	R/W	-	undef.
0x00000EEE	C1MCTRL47	CAN1 data buffer register	-	-	R/W	-	undef.
0x00001000	C2GMCTRLL	CAN2 Global Macro Control Register low byte	R/W	R/W	-	-	0x00
0x00001000	C2GMCTRL	CAN2 Global Macro Control Register	-	-	R/W	-	0x0000
0x00001001	C2GMCTRLH	CAN2 Global Macro Control Register high byte	R/W	R/W	-	-	0x00
0x00001002	C2GMCS	CAN2 Global Macro Clock Selection Register	R/W	R/W	-	-	0x0F
0x00001004	C2GMCONF	CAN2 global configuration register	-	-	R	-	0x0019
0x00001006	C2GMABTL	CAN2 Global Macro Automatic Block Transmission Register low byte	R/W	R/W	-	-	0x00
0x00001006	C2GMABT	CAN2 Global Macro Automatic Block Transmission Register	-	-	R/W	-	0x0000
0x00001007	C2GMABTH	CAN2 Global Macro Automatic Block Transmission Register high byte	R/W	R/W	-	-	0x00
0x00001008	C2GMABTD	CAN2 Global Macro Automatic Block Transmission Delay Register	R/W	R/W	-	-	0x00
0x00001040	C2MASK1L	CAN2 Module Mask 1 Register lower half word	-	-	R/W	-	undef.
0x00001042	C2MASK1H	CAN2 Module Mask 1 Register upper half word	-	-	R/W	-	undef.
0x00001044	C2MASK2L	CAN2 Module Mask 2 Register lower half word	-	-	R/W	-	undef.
0x00001046	C2MASK2H	CAN2 Module Mask 2 Register upper half word	-	-	R/W	-	undef.
0x00001048	C2MASK3L	CAN2 Module Mask 3 Register lower half word	-	-	R/W	-	undef.
0x0000104A	C2MASK3H	CAN2 Module Mask 3 Register upper half word	-	-	R/W	-	undef.
0x0000104C	C2MASK4L	CAN2 Module Mask 4 Register lower half word	-	-	R/W	-	undef.
0x0000104E	C2MASK4H	CAN2 Module Mask 4 Register upper half word	-	-	R/W	-	undef.
0x00001050	C2CTRL	CAN2 Module Control Register	-	-	R/W	-	0x0000
0x00001052	C2LEC	CAN2 Module Last Error Code Register	R/W	R/W	-	-	0x00
0x00001053	C2INFO	CAN2 Module Information Register	R	R	-	-	0x00
0x00001054	C2ERC	CAN2 Module Error Counter	-	-	R	-	0x0000
0x00001056	C2IEL	CAN2 Module Interrupt Enable Register low byte	R/W	R/W	-	-	0x00
0x00001056	C2IE	CAN2 Module Interrupt Enable Register	-	-	R/W	-	0x0000
0x00001057	C2IEH	CAN2 Module Interrupt Enable Register high byte	R/W	R/W	-	-	0x00

Table 3-7: PPA related SFR Table (44/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001058	C2INTSL	CAN2 Module Interrupt Status Register low byte	R/W	R/W	-	-	0x00
0x00001058	C2INTS	CAN2 Module Interrupt Status Register	-	-	R/W	-	0x0000
0x0000105A	C2BRP	CAN2 Module Bit-Rate Prescaler Register	R/W	R/W	-	-	0xFF
0x0000105C	C2BTR	CAN2 Bit Rate Register	-	-	R/W	-	0x370F
0x0000105E	C2LIPT	CAN2 Module Last In-Pointer Register	-	R/W	-	-	undef.
0x00001060	C2RGPTL	CAN2 Module Receive History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00001060	C2RGPT	CAN2 Module Receive History List Get Pointer Register	-	-	R/W	-	undef.
0x00001062	C2LOPT	CAN2 Module Last Out-Pointer Register	-	R	-	-	undef.
0x00001064	C2TGPTL	CAN2 Module Transmit History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00001064	C2TGPT	CAN2 Module Transmit History List Get Pointer Register	-	-	R/W	-	undef.
0x00001066	C2TSL	CAN2 Module Time Stamp Register low byte	R/W	R/W	-	-	0x00
0x00001066	C2TS	CAN2 Module Time Stamp Register	-	-	R/W	-	0x0000
0x00001067	C2TSH	CAN2 Module Time Stamp Register high byte	R/W	R/W	-	-	0x00
0x00001100	C2MDATA000	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001100	C2MDATA0100	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001101	C2MDATA100	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001102	C2MDATA200	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001102	C2MDATA2300	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001103	C2MDATA300	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001104	C2MDATA400	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001104	C2MDATA4500	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001105	C2MDATA500	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001106	C2MDATA600	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001106	C2MDATA6700	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001107	C2MDATA700	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001108	C2MDLC00	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001109	C2MCONF00	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000110A	C2MIDL00	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000110C	C2MIDH00	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000110E	C2MCTRL00	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001120	C2MDATA001	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001120	C2MDATA0101	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001121	C2MDATA101	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001122	C2MDATA201	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001122	C2MDATA2301	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001123	C2MDATA301	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (45/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001124	C2MDATA401	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001124	C2MDATA4501	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001125	C2MDATA501	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001126	C2MDATA601	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001126	C2MDATA6701	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001127	C2MDATA701	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001128	C2MDLC01	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001129	C2MCONF01	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000112A	C2MIDL01	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000112C	C2MIDH01	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000112E	C2MCTRL01	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001140	C2MDATA002	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001140	C2MDATA0102	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001141	C2MDATA102	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001142	C2MDATA202	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001142	C2MDATA2302	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001143	C2MDATA302	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001144	C2MDATA402	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001144	C2MDATA4502	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001145	C2MDATA502	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001146	C2MDATA602	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001146	C2MDATA6702	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001147	C2MDATA702	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001148	C2MDLC02	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001149	C2MCONF02	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000114A	C2MIDL02	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000114C	C2MIDH02	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000114E	C2MCTRL02	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001160	C2MDATA003	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001160	C2MDATA0103	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001161	C2MDATA103	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001162	C2MDATA203	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001162	C2MDATA2303	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001163	C2MDATA303	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001164	C2MDATA403	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001164	C2MDATA4503	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001165	C2MDATA503	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001166	C2MDATA603	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001166	C2MDATA6703	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001167	C2MDATA703	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001168	C2MDLC03	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (46/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001169	C2MCONF03	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000116A	C2MIDL03	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000116C	C2MIDH03	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000116E	C2MCTRL03	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001180	C2MDATA004	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001180	C2MDATA0104	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001181	C2MDATA104	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001182	C2MDATA204	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001182	C2MDATA2304	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001183	C2MDATA304	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001184	C2MDATA404	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001184	C2MDATA4504	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001185	C2MDATA504	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001186	C2MDATA604	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001186	C2MDATA6704	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001187	C2MDATA704	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001188	C2MDLC04	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001189	C2MCONF04	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000118A	C2MIDL04	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000118C	C2MIDH04	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000118E	C2MCTRL04	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011A0	C2MDATA005	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A0	C2MDATA0105	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011A1	C2MDATA105	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A2	C2MDATA205	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A2	C2MDATA2305	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011A3	C2MDATA305	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A4	C2MDATA405	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A4	C2MDATA4505	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011A5	C2MDATA505	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A6	C2MDATA605	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A6	C2MDATA6705	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011A7	C2MDATA705	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A8	C2MDLC05	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011A9	C2MCONF05	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011AA	C2MIDL05	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011AC	C2MIDH05	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011AE	C2MCTRL05	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011C0	C2MDATA006	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C0	C2MDATA0106	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011C1	C2MDATA106	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (47/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000011C2	C2MDATA206	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C2	C2MDATA2306	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011C3	C2MDATA306	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C4	C2MDATA406	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C4	C2MDATA4506	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011C5	C2MDATA506	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C6	C2MDATA606	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C6	C2MDATA6706	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011C7	C2MDATA706	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C8	C2MDLC06	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011C9	C2MCONF06	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011CA	C2MIDL06	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011CC	C2MIDH06	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011CE	C2MCTRL06	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011E0	C2MDATA007	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E0	C2MDATA0107	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011E1	C2MDATA107	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E2	C2MDATA207	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E2	C2MDATA2307	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011E3	C2MDATA307	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E4	C2MDATA407	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E4	C2MDATA4507	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011E5	C2MDATA507	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E6	C2MDATA607	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E6	C2MDATA6707	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011E7	C2MDATA707	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E8	C2MDLC07	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011E9	C2MCONF07	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000011EA	C2MIDL07	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011EC	C2MIDH07	CAN2 data buffer register	-	-	R/W	-	undef.
0x000011EE	C2MCTRL07	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001200	C2MDATA008	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001200	C2MDATA0108	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001201	C2MDATA108	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001202	C2MDATA208	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001202	C2MDATA2308	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001203	C2MDATA308	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001204	C2MDATA408	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001204	C2MDATA4508	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001205	C2MDATA508	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001206	C2MDATA608	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (48/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001206	C2MDATA6708	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001207	C2MDATA708	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001208	C2MDLC08	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001209	C2MCONF08	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000120A	C2MIDL08	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000120C	C2MIDH08	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000120E	C2MCTRL08	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001220	C2MDATA009	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001220	C2MDATA0109	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001221	C2MDATA109	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001222	C2MDATA209	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001222	C2MDATA2309	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001223	C2MDATA309	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001224	C2MDATA409	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001224	C2MDATA4509	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001225	C2MDATA509	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001226	C2MDATA609	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001226	C2MDATA6709	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001227	C2MDATA709	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001228	C2MDLC09	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001229	C2MCONF09	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000122A	C2MIDL09	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000122C	C2MIDH09	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000122E	C2MCTRL09	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001240	C2MDATA010	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001240	C2MDATA0110	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001241	C2MDATA110	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001242	C2MDATA210	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001242	C2MDATA2310	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001243	C2MDATA310	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001244	C2MDATA410	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001244	C2MDATA4510	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001245	C2MDATA510	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001246	C2MDATA610	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001246	C2MDATA6710	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001247	C2MDATA710	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001248	C2MDLC10	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001249	C2MCONF10	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000124A	C2MIDL10	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000124C	C2MIDH10	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000124E	C2MCTRL10	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (49/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001260	C2MDATA011	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001260	C2MDATA0111	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001261	C2MDATA111	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001262	C2MDATA211	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001262	C2MDATA2311	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001263	C2MDATA311	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001264	C2MDATA411	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001264	C2MDATA4511	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001265	C2MDATA511	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001266	C2MDATA611	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001266	C2MDATA6711	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001267	C2MDATA711	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001268	C2MDLC11	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001269	C2MCONF11	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000126A	C2MIDL11	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000126C	C2MIDH11	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000126E	C2MCTRL11	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001280	C2MDATA012	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001280	C2MDATA0112	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001281	C2MDATA112	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001282	C2MDATA212	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001282	C2MDATA2312	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001283	C2MDATA312	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001284	C2MDATA412	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001284	C2MDATA4512	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001285	C2MDATA512	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001286	C2MDATA612	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001286	C2MDATA6712	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001287	C2MDATA712	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001288	C2MDLC12	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001289	C2MCONF12	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000128A	C2MIDL12	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000128C	C2MIDH12	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000128E	C2MCTRL12	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012A0	C2MDATA013	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A0	C2MDATA0113	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012A1	C2MDATA113	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A2	C2MDATA213	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A2	C2MDATA2313	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012A3	C2MDATA313	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A4	C2MDATA413	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (50/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000012A4	C2MDATA4513	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012A5	C2MDATA513	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A6	C2MDATA613	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A6	C2MDATA6713	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012A7	C2MDATA713	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A8	C2MDLC13	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012A9	C2MCONF13	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012AA	C2MIDL13	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012AC	C2MIDH13	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012AE	C2MCTRL13	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012C0	C2MDATA014	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C0	C2MDATA0114	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012C1	C2MDATA114	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C2	C2MDATA214	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C2	C2MDATA2314	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012C3	C2MDATA314	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C4	C2MDATA414	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C4	C2MDATA4514	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012C5	C2MDATA514	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C6	C2MDATA614	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C6	C2MDATA6714	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012C7	C2MDATA714	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C8	C2MDLC14	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012C9	C2MCONF14	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012CA	C2MIDL14	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012CC	C2MIDH14	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012CE	C2MCTRL14	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012E0	C2MDATA015	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E0	C2MDATA0115	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012E1	C2MDATA115	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E2	C2MDATA215	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E2	C2MDATA2315	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012E3	C2MDATA315	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E4	C2MDATA415	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E4	C2MDATA4515	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012E5	C2MDATA515	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E6	C2MDATA615	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E6	C2MDATA6715	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012E7	C2MDATA715	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E8	C2MDLC15	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000012E9	C2MCONF15	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (51/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000012EA	C2MIDL15	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012EC	C2MIDH15	CAN2 data buffer register	-	-	R/W	-	undef.
0x000012EE	C2MCTRL15	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001300	C2MDATA016	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001300	C2MDATA0116	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001301	C2MDATA116	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001302	C2MDATA216	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001302	C2MDATA2316	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001303	C2MDATA316	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001304	C2MDATA416	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001304	C2MDATA4516	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001305	C2MDATA516	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001306	C2MDATA616	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001306	C2MDATA6716	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001307	C2MDATA716	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001308	C2MDLC16	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001309	C2MCONF16	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000130A	C2MIDL16	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000130C	C2MIDH16	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000130E	C2MCTRL16	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001320	C2MDATA017	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001320	C2MDATA0117	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001321	C2MDATA117	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001322	C2MDATA217	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001322	C2MDATA2317	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001323	C2MDATA317	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001324	C2MDATA417	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001324	C2MDATA4517	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001325	C2MDATA517	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001326	C2MDATA617	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001326	C2MDATA6717	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001327	C2MDATA717	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001328	C2MDLC17	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001329	C2MCONF17	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000132A	C2MIDL17	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000132C	C2MIDH17	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000132E	C2MCTRL17	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001340	C2MDATA018	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001340	C2MDATA0118	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001341	C2MDATA118	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001342	C2MDATA218	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (52/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001342	C2MDATA2318	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001343	C2MDATA318	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001344	C2MDATA418	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001344	C2MDATA4518	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001345	C2MDATA518	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001346	C2MDATA618	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001346	C2MDATA6718	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001347	C2MDATA718	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001348	C2MDLC18	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001349	C2MCONF18	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000134A	C2MIDL18	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000134C	C2MIDH18	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000134E	C2MCTRL18	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001360	C2MDATA019	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001360	C2MDATA0191	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001361	C2MDATA191	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001362	C2MDATA219	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001362	C2MDATA2319	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001363	C2MDATA319	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001364	C2MDATA419	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001364	C2MDATA4519	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001365	C2MDATA519	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001366	C2MDATA619	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001366	C2MDATA6719	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001367	C2MDATA719	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001368	C2MDLC19	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001369	C2MCONF19	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000136A	C2MIDL19	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000136C	C2MIDH19	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000136E	C2MCTRL19	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001380	C2MDATA020	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001380	C2MDATA0120	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001381	C2MDATA120	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001382	C2MDATA220	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001382	C2MDATA2320	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001383	C2MDATA320	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001384	C2MDATA420	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001384	C2MDATA4520	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001385	C2MDATA520	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001386	C2MDATA620	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001386	C2MDATA6720	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (53/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001387	C2MDATA720	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001388	C2MDLC20	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001389	C2MCONF20	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000138A	C2MIDL20	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000138C	C2MIDH20	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000138E	C2MCTRL20	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013A0	C2MDATA021	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A0	C2MDATA0121	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013A1	C2MDATA121	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A2	C2MDATA221	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A2	C2MDATA2321	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013A3	C2MDATA321	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A4	C2MDATA421	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A4	C2MDATA4521	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013A5	C2MDATA521	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A6	C2MDATA621	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A6	C2MDATA6721	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013A7	C2MDATA721	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A8	C2MDLC21	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013A9	C2MCONF21	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013AA	C2MIDL21	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013AC	C2MIDH21	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013AE	C2MCTRL21	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013C0	C2MDATA022	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C0	C2MDATA0122	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013C1	C2MDATA122	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C2	C2MDATA222	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C2	C2MDATA2322	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013C3	C2MDATA322	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C4	C2MDATA422	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C4	C2MDATA4522	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013C5	C2MDATA522	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C6	C2MDATA622	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C6	C2MDATA6722	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013C7	C2MDATA722	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C8	C2MDLC22	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013C9	C2MCONF22	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013CA	C2MIDL22	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013CC	C2MIDH22	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013CE	C2MCTRL22	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013E0	C2MDATA023	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (54/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000013E0	C2MDATA0123	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013E1	C2MDATA123	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E2	C2MDATA223	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E2	C2MDATA2323	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013E3	C2MDATA323	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E4	C2MDATA423	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E4	C2MDATA4523	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013E5	C2MDATA523	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E6	C2MDATA623	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E6	C2MDATA6723	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013E7	C2MDATA723	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E8	C2MDLC23	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013E9	C2MCONF23	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000013EA	C2MIDL23	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013EC	C2MIDH23	CAN2 data buffer register	-	-	R/W	-	undef.
0x000013EE	C2MCTRL23	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001400	C2MDATA024	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001400	C2MDATA0124	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001401	C2MDATA124	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001402	C2MDATA224	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001402	C2MDATA2324	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001403	C2MDATA324	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001404	C2MDATA424	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001404	C2MDATA4524	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001405	C2MDATA524	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001406	C2MDATA624	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001406	C2MDATA6724	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001407	C2MDATA724	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001408	C2MDLC24	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001409	C2MCONF24	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000140A	C2MIDL24	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000140C	C2MIDH24	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000140E	C2MCTRL24	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001420	C2MDATA025	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001420	C2MDATA0125	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001421	C2MDATA125	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001422	C2MDATA225	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001422	C2MDATA2325	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001423	C2MDATA325	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001424	C2MDATA425	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001424	C2MDATA4525	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (55/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001425	C2MDATA525	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001426	C2MDATA625	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001426	C2MDATA6725	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001427	C2MDATA725	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001428	C2MDLC25	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001429	C2MCONF25	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000142A	C2MIDL25	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000142C	C2MIDH25	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000142E	C2MCTRL25	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001440	C2MDATA026	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001440	C2MDATA0126	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001441	C2MDATA126	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001442	C2MDATA226	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001442	C2MDATA2326	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001443	C2MDATA326	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001444	C2MDATA426	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001444	C2MDATA4526	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001445	C2MDATA526	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001446	C2MDATA626	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001446	C2MDATA6726	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001447	C2MDATA726	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001448	C2MDLC26	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001449	C2MCONF26	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000144A	C2MIDL26	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000144C	C2MIDH26	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000144E	C2MCTRL26	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001460	C2MDATA027	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001460	C2MDATA0271	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001461	C2MDATA271	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001462	C2MDATA227	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001462	C2MDATA2327	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001463	C2MDATA327	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001464	C2MDATA427	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001464	C2MDATA4527	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001465	C2MDATA527	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001466	C2MDATA627	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001466	C2MDATA6727	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001467	C2MDATA727	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001468	C2MDLC27	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001469	C2MCONF27	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000146A	C2MIDL27	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (56/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000146C	C2MIDH27	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000146E	C2MCTRL27	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001480	C2MDATA028	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001480	C2MDATA0128	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001481	C2MDATA128	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001482	C2MDATA228	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001482	C2MDATA2328	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001483	C2MDATA328	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001484	C2MDATA428	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001484	C2MDATA4528	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001485	C2MDATA528	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001486	C2MDATA628	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001486	C2MDATA6728	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001487	C2MDATA728	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001488	C2MDLC28	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001489	C2MCONF28	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000148A	C2MIDL28	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000148C	C2MIDH28	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000148E	C2MCTRL28	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014A0	C2MDATA029	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A0	C2MDATA0129	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014A1	C2MDATA129	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A2	C2MDATA229	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A2	C2MDATA2329	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014A3	C2MDATA329	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A4	C2MDATA429	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A4	C2MDATA4529	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014A5	C2MDATA529	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A6	C2MDATA629	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A6	C2MDATA6729	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014A7	C2MDATA729	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A8	C2MDLC29	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014A9	C2MCONF29	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014AA	C2MIDL29	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014AC	C2MIDH29	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014AE	C2MCTRL29	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014C0	C2MDATA030	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C0	C2MDATA0130	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014C1	C2MDATA130	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C2	C2MDATA230	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C2	C2MDATA2330	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (57/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000014C3	C2MDATA330	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C4	C2MDATA430	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C4	C2MDATA4530	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014C5	C2MDATA530	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C6	C2MDATA630	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C6	C2MDATA6730	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014C7	C2MDATA730	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C8	C2MDLC30	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014C9	C2MCONF30	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014CA	C2MIDL30	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014CC	C2MIDH30	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014CE	C2MCTRL30	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014E0	C2MDATA031	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E0	C2MDATA0131	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014E1	C2MDATA131	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E2	C2MDATA231	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E2	C2MDATA2331	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014E3	C2MDATA331	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E4	C2MDATA431	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E4	C2MDATA4531	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014E5	C2MDATA531	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E6	C2MDATA631	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E6	C2MDATA6731	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014E7	C2MDATA731	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E8	C2MDLC31	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014E9	C2MCONF31	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000014EA	C2MIDL31	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014EC	C2MIDH31	CAN2 data buffer register	-	-	R/W	-	undef.
0x000014EE	C2MCTRL31	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001500	C2MDATA032	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001500	C2MDATA0132	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001501	C2MDATA132	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001502	C2MDATA232	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001502	C2MDATA2332	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001503	C2MDATA332	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001504	C2MDATA432	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001504	C2MDATA4532	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001505	C2MDATA532	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001506	C2MDATA632	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001506	C2MDATA6732	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001507	C2MDATA732	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (58/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001508	C2MDLC32	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001509	C2MCONF32	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000150A	C2MIDL32	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000150C	C2MIDH32	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000150E	C2MCTRL32	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001520	C2MDATA033	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001520	C2MDATA0133	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001521	C2MDATA133	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001522	C2MDATA233	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001522	C2MDATA2333	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001523	C2MDATA333	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001524	C2MDATA433	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001524	C2MDATA4533	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001525	C2MDATA533	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001526	C2MDATA633	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001526	C2MDATA6733	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001527	C2MDATA733	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001528	C2MDLC33	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001529	C2MCONF33	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000152A	C2MIDL33	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000152C	C2MIDH33	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000152E	C2MCTRL33	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001540	C2MDATA034	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001540	C2MDATA0134	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001541	C2MDATA134	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001542	C2MDATA234	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001542	C2MDATA2334	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001543	C2MDATA334	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001544	C2MDATA434	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001544	C2MDATA4534	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001545	C2MDATA534	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001546	C2MDATA634	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001546	C2MDATA6734	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001547	C2MDATA734	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001548	C2MDLC34	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001549	C2MCONF34	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000154A	C2MIDL34	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000154C	C2MIDH34	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000154E	C2MCTRL34	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001560	C2MDATA035	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001560	C2MDATA0351	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (59/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001561	C2MDATA351	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001562	C2MDATA235	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001562	C2MDATA2335	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001563	C2MDATA335	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001564	C2MDATA435	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001564	C2MDATA4535	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001565	C2MDATA535	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001566	C2MDATA635	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001566	C2MDATA6735	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001567	C2MDATA735	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001568	C2MDLC35	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001569	C2MCONF35	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000156A	C2MIDL35	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000156C	C2MIDH35	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000156E	C2MCTRL35	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001580	C2MDATA036	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001580	C2MDATA0136	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001581	C2MDATA136	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001582	C2MDATA236	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001582	C2MDATA2336	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001583	C2MDATA336	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001584	C2MDATA436	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001584	C2MDATA4536	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001585	C2MDATA536	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001586	C2MDATA636	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001586	C2MDATA6736	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001587	C2MDATA736	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001588	C2MDLC36	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001589	C2MCONF36	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000158A	C2MIDL36	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000158C	C2MIDH36	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000158E	C2MCTRL36	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015A0	C2MDATA037	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A0	C2MDATA0137	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015A1	C2MDATA137	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A2	C2MDATA237	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A2	C2MDATA2337	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015A3	C2MDATA337	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A4	C2MDATA437	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A4	C2MDATA4537	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015A5	C2MDATA537	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (60/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000015A6	C2MDATA637	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A6	C2MDATA6737	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015A7	C2MDATA737	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A8	C2MDLC37	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015A9	C2MCONF37	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015AA	C2MIDL37	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015AC	C2MIDH37	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015AE	C2MCTRL37	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015C0	C2MDATA038	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C0	C2MDATA0138	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015C1	C2MDATA138	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C2	C2MDATA238	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C2	C2MDATA2338	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015C3	C2MDATA338	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C4	C2MDATA438	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C4	C2MDATA4538	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015C5	C2MDATA538	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C6	C2MDATA638	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C6	C2MDATA6738	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015C7	C2MDATA738	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C8	C2MDLC38	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015C9	C2MCONF38	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015CA	C2MIDL38	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015CC	C2MIDH38	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015CE	C2MCTRL38	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015E0	C2MDATA039	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E0	C2MDATA0139	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015E1	C2MDATA139	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E2	C2MDATA239	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E2	C2MDATA2339	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015E3	C2MDATA339	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E4	C2MDATA439	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E4	C2MDATA4539	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015E5	C2MDATA539	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E6	C2MDATA639	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E6	C2MDATA6739	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015E7	C2MDATA739	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E8	C2MDLC39	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015E9	C2MCONF39	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000015EA	C2MIDL39	CAN2 data buffer register	-	-	R/W	-	undef.
0x000015EC	C2MIDH39	CAN2 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (61/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000015EE	C2MCTRL39	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001600	C2MDATA040	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001600	C2MDATA0140	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001601	C2MDATA140	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001602	C2MDATA240	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001602	C2MDATA2340	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001603	C2MDATA340	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001604	C2MDATA440	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001604	C2MDATA4540	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001605	C2MDATA540	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001606	C2MDATA640	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001606	C2MDATA6740	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001607	C2MDATA740	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001608	C2MDLC40	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001609	C2MCONF40	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000160A	C2MIDL40	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000160C	C2MIDH40	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000160E	C2MCTRL40	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001620	C2MDATA041	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001620	C2MDATA0141	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001621	C2MDATA141	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001622	C2MDATA241	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001622	C2MDATA2341	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001623	C2MDATA341	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001624	C2MDATA441	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001624	C2MDATA4541	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001625	C2MDATA541	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001626	C2MDATA641	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001626	C2MDATA6741	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001627	C2MDATA741	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001628	C2MDLC41	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001629	C2MCONF41	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000162A	C2MIDL41	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000162C	C2MIDH41	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000162E	C2MCTRL41	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001640	C2MDATA042	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001640	C2MDATA0142	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001641	C2MDATA142	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001642	C2MDATA242	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001642	C2MDATA2342	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001643	C2MDATA342	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (62/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001644	C2MDATA442	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001644	C2MDATA4542	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001645	C2MDATA542	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001646	C2MDATA642	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001646	C2MDATA6742	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001647	C2MDATA742	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001648	C2MDLC42	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001649	C2MCONF42	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000164A	C2MIDL42	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000164C	C2MIDH42	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000164E	C2MCTRL42	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001660	C2MDATA043	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001660	C2MDATA0143	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001661	C2MDATA143	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001662	C2MDATA243	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001662	C2MDATA2343	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001663	C2MDATA343	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001664	C2MDATA443	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001664	C2MDATA4543	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001665	C2MDATA543	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001666	C2MDATA643	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001666	C2MDATA6743	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001667	C2MDATA743	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001668	C2MDLC43	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001669	C2MCONF43	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000166A	C2MIDL43	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000166C	C2MIDH43	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000166E	C2MCTRL43	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001680	C2MDATA044	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001680	C2MDATA0144	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001681	C2MDATA144	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001682	C2MDATA244	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001682	C2MDATA2344	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001683	C2MDATA344	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001684	C2MDATA444	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001684	C2MDATA4544	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001685	C2MDATA544	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001686	C2MDATA644	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001686	C2MDATA6744	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001687	C2MDATA744	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x00001688	C2MDLC44	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (63/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001689	C2MCONF44	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x0000168A	C2MIDL44	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000168C	C2MIDH44	CAN2 data buffer register	-	-	R/W	-	undef.
0x0000168E	C2MCTRL44	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016A0	C2MDATA045	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A0	C2MDATA0145	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016A1	C2MDATA145	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A2	C2MDATA245	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A2	C2MDATA2345	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016A3	C2MDATA345	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A4	C2MDATA445	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A4	C2MDATA4545	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016A5	C2MDATA545	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A6	C2MDATA645	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A6	C2MDATA6745	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016A7	C2MDATA745	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A8	C2MDLC45	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016A9	C2MCONF45	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016AA	C2MIDL45	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016AC	C2MIDH45	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016AE	C2MCTRL45	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016C0	C2MDATA046	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C0	C2MDATA0146	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016C1	C2MDATA146	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C2	C2MDATA246	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C2	C2MDATA2346	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016C3	C2MDATA346	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C4	C2MDATA446	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C4	C2MDATA4546	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016C5	C2MDATA546	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C6	C2MDATA646	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C6	C2MDATA6746	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016C7	C2MDATA746	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C8	C2MDLC46	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016C9	C2MCONF46	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016CA	C2MIDL46	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016CC	C2MIDH46	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016CE	C2MCTRL46	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016E0	C2MDATA047	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E0	C2MDATA0147	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016E1	C2MDATA147	CAN2 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (64/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000016E2	C2MDATA247	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E2	C2MDATA2347	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016E3	C2MDATA347	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E4	C2MDATA447	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E4	C2MDATA4547	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016E5	C2MDATA547	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E6	C2MDATA647	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E6	C2MDATA6747	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016E7	C2MDATA747	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E8	C2MDLC47	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016E9	C2MCONF47	CAN2 data buffer register	R/W	R/W	-	-	undef.
0x000016EA	C2MIDL47	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016EC	C2MIDH47	CAN2 data buffer register	-	-	R/W	-	undef.
0x000016EE	C2MCTRL47	CAN2 data buffer register	-	-	R/W	-	undef.
0x00001800	C3GMCTRLL	CAN3 Global Macro Control Register low byte	R/W	R/W	-	-	0x00
0x00001800	C3GMCTRL	CAN3 Global Macro Control Register	-	-	R/W	-	0x0000
0x00001801	C3GMCTRLH	CAN3 Global Macro Control Register high byte	R/W	R/W	-	-	0x00
0x00001802	C3GMCS	CAN3 Global Macro Clock Selection Register	R/W	R/W	-	-	0x0F
0x00001804	C3GMCONF	CAN3 global configuration register	-	-	R	-	0x0019
0x00001806	C3GMABTL	CAN3 Global Macro Automatic Block Transmission Register low byte	R/W	R/W	-	-	0x00
0x00001806	C3GMABT	CAN3 Global Macro Automatic Block Transmission Register	-	-	R/W	-	0x0000
0x00001807	C3GMABTH	CAN3 Global Macro Automatic Block Transmission Register high byte	R/W	R/W	-	-	0x00
0x00001808	C3GMABTD	CAN3 Global Macro Automatic Block Transmission Delay Register	R/W	R/W	-	-	0x00
0x00001840	C3MASK1L	CAN3 Module Mask 1 Register lower half word	-	-	R/W	-	undef.
0x00001842	C3MASK1H	CAN3 Module Mask 1 Register upper half word	-	-	R/W	-	undef.
0x00001844	C3MASK2L	CAN3 Module Mask 2 Register lower half word	-	-	R/W	-	undef.
0x00001846	C3MASK2H	CAN3 Module Mask 2 Register upper half word	-	-	R/W	-	undef.
0x00001848	C3MASK3L	CAN3 Module Mask 3 Register lower half word	-	-	R/W	-	undef.
0x0000184A	C3MASK3H	CAN3 Module Mask 3 Register upper half word	-	-	R/W	-	undef.
0x0000184C	C3MASK4L	CAN3 Module Mask 4 Register lower half word	-	-	R/W	-	undef.
0x0000184E	C3MASK4H	CAN3 Module Mask 4 Register upper half word	-	-	R/W	-	undef.
0x00001850	C3CTRL	CAN3 Module Control Register	-	-	R/W	-	0x0000
0x00001852	C3LEC	CAN3 Module Last Error Code Register	R/W	R/W	-	-	0x00
0x00001853	C3INFO	CAN3 Module Information Register	R	R	-	-	0x00
0x00001854	C3ERC	CAN3 Module Error Counter	-	-	R	-	0x0000

Table 3-7: PPA related SFR Table (65/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001856	C3IEL	CAN3 Module Interrupt Enable Register low byte	R/W	R/W	-	-	0x00
0x00001856	C3IE	CAN3 Module Interrupt Enable Register	-	-	R/W	-	0x0000
0x00001857	C3IEH	CAN3 Module Interrupt Enable Register high byte	R/W	R/W	-	-	0x00
0x00001858	C3INTSL	CAN3 Module Interrupt Status Register low byte	R/W	R/W	-	-	0x00
0x00001858	C3INTS	CAN3 Module Interrupt Status Register	-	-	R/W	-	0x0000
0x0000185A	C3BRP	CAN3 Module Bit-Rate Prescaler Register	R/W	R/W	-	-	0xFF
0x0000185C	C3BTR	CAN3 Bit Rate Register	-	-	R/W	-	0x370F
0x0000185E	C3LIPT	CAN3 Module Last In-Pointer Register	-	R/W	-	-	undef.
0x00001860	C3RGPTL	CAN3 Module Receive History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00001860	C3RGPT	CAN3 Module Receive History List Get Pointer Register	-	-	R/W	-	undef.
0x00001862	C3LOPT	CAN3 Module Last Out-Pointer Register	-	R	-	-	undef.
0x00001864	C3TGPTL	CAN3 Module Transmit History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00001864	C3TGPT	CAN3 Module Transmit History List Get Pointer Register	-	-	R/W	-	undef.
0x00001866	C3TSL	CAN3 Module Time Stamp Register low byte	R/W	R/W	-	-	0x00
0x00001866	C3TS	CAN3 Module Time Stamp Register	-	-	R/W	-	0x0000
0x00001867	C3TSH	CAN3 Module Time Stamp Register high byte	R/W	R/W	-	-	0x00
0x00001900	C3MDATA000	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001900	C3MDATA0100	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001901	C3MDATA100	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001902	C3MDATA200	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001902	C3MDATA2300	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001903	C3MDATA300	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001904	C3MDATA400	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001904	C3MDATA4500	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001905	C3MDATA500	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001906	C3MDATA600	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001906	C3MDATA6700	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001907	C3MDATA700	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001908	C3MDLC00	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001909	C3MCONF00	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000190A	C3MIDL00	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000190C	C3MIDH00	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000190E	C3MCTRL00	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001920	C3MDATA001	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001920	C3MDATA0101	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (66/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001921	C3MDATA101	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001922	C3MDATA201	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001922	C3MDATA2301	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001923	C3MDATA301	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001924	C3MDATA401	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001924	C3MDATA4501	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001925	C3MDATA501	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001926	C3MDATA601	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001926	C3MDATA6701	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001927	C3MDATA701	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001928	C3MDLC01	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001929	C3MCONF01	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000192A	C3MIDL01	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000192C	C3MIDH01	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000192E	C3MCTRL01	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001940	C3MDATA002	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001940	C3MDATA0102	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001941	C3MDATA102	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001942	C3MDATA202	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001942	C3MDATA2302	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001943	C3MDATA302	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001944	C3MDATA402	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001944	C3MDATA4502	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001945	C3MDATA502	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001946	C3MDATA602	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001946	C3MDATA6702	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001947	C3MDATA702	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001948	C3MDLC02	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001949	C3MCONF02	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000194A	C3MIDL02	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000194C	C3MIDH02	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000194E	C3MCTRL02	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001960	C3MDATA003	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001960	C3MDATA0103	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001961	C3MDATA103	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001962	C3MDATA203	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001962	C3MDATA2303	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001963	C3MDATA303	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001964	C3MDATA403	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001964	C3MDATA4503	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001965	C3MDATA503	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (67/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001966	C3MDATA603	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001966	C3MDATA6703	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001967	C3MDATA703	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001968	C3MDLC03	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001969	C3MCONF03	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000196A	C3MIDL03	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000196C	C3MIDH03	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000196E	C3MCTRL03	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001980	C3MDATA004	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001980	C3MDATA0104	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001981	C3MDATA104	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001982	C3MDATA204	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001982	C3MDATA2304	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001983	C3MDATA304	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001984	C3MDATA404	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001984	C3MDATA4504	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001985	C3MDATA504	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001986	C3MDATA604	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001986	C3MDATA6704	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001987	C3MDATA704	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001988	C3MDLC04	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001989	C3MCONF04	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000198A	C3MIDL04	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000198C	C3MIDH04	CAN3 data buffer register	-	-	R/W	-	undef.
0x0000198E	C3MCTRL04	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019A0	C3MDATA005	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A0	C3MDATA0105	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019A1	C3MDATA105	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A2	C3MDATA205	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A2	C3MDATA2305	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019A3	C3MDATA305	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A4	C3MDATA405	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A4	C3MDATA4505	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019A5	C3MDATA505	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A6	C3MDATA605	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A6	C3MDATA6705	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019A7	C3MDATA705	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A8	C3MDLC05	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019A9	C3MCONF05	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019AA	C3MIDL05	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019AC	C3MIDH05	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (68/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000019AE	C3MCTRL05	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019C0	C3MDATA006	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C0	C3MDATA0106	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019C1	C3MDATA106	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C2	C3MDATA206	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C2	C3MDATA2306	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019C3	C3MDATA306	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C4	C3MDATA406	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C4	C3MDATA4506	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019C5	C3MDATA506	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C6	C3MDATA606	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C6	C3MDATA6706	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019C7	C3MDATA706	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C8	C3MDLC06	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019C9	C3MCONF06	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019CA	C3MIDL06	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019CC	C3MIDH06	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019CE	C3MCTRL06	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019E0	C3MDATA007	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E0	C3MDATA0107	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019E1	C3MDATA107	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E2	C3MDATA207	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E2	C3MDATA2307	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019E3	C3MDATA307	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E4	C3MDATA407	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E4	C3MDATA4507	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019E5	C3MDATA507	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E6	C3MDATA607	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E6	C3MDATA6707	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019E7	C3MDATA707	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E8	C3MDLC07	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019E9	C3MCONF07	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x000019EA	C3MIDL07	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019EC	C3MIDH07	CAN3 data buffer register	-	-	R/W	-	undef.
0x000019EE	C3MCTRL07	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A00	C3MDATA008	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A00	C3MDATA0108	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A01	C3MDATA108	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A02	C3MDATA208	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A02	C3MDATA2308	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A03	C3MDATA308	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (69/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001A04	C3MDATA408	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A04	C3MDATA4508	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A05	C3MDATA508	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A06	C3MDATA608	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A06	C3MDATA6708	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A07	C3MDATA708	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A08	C3MDLC08	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A09	C3MCONF08	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A0A	C3MIDL08	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A0C	C3MIDH08	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A0E	C3MCTRL08	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A20	C3MDATA009	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A20	C3MDATA0109	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A21	C3MDATA109	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A22	C3MDATA209	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A22	C3MDATA2309	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A23	C3MDATA309	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A24	C3MDATA409	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A24	C3MDATA4509	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A25	C3MDATA509	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A26	C3MDATA609	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A26	C3MDATA6709	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A27	C3MDATA709	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A28	C3MDLC09	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A29	C3MCONF09	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A2A	C3MIDL09	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A2C	C3MIDH09	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A2E	C3MCTRL09	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A40	C3MDATA010	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A40	C3MDATA0110	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A41	C3MDATA110	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A42	C3MDATA210	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A42	C3MDATA2310	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A43	C3MDATA310	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A44	C3MDATA410	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A44	C3MDATA4510	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A45	C3MDATA510	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A46	C3MDATA610	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A46	C3MDATA6710	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A47	C3MDATA710	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A48	C3MDLC10	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (70/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001A49	C3MCONF10	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A4A	C3MIDL10	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A4C	C3MIDH10	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A4E	C3MCTRL10	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A60	C3MDATA011	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A60	C3MDATA0111	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A61	C3MDATA111	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A62	C3MDATA211	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A62	C3MDATA2311	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A63	C3MDATA311	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A64	C3MDATA411	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A64	C3MDATA4511	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A65	C3MDATA511	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A66	C3MDATA611	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A66	C3MDATA6711	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A67	C3MDATA711	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A68	C3MDLC11	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A69	C3MCONF11	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A6A	C3MIDL11	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A6C	C3MIDH11	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A6E	C3MCTRL11	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A80	C3MDATA012	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A80	C3MDATA0112	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A81	C3MDATA112	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A82	C3MDATA212	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A82	C3MDATA2312	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A83	C3MDATA312	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A84	C3MDATA412	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A84	C3MDATA4512	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A85	C3MDATA512	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A86	C3MDATA612	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A86	C3MDATA6712	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A87	C3MDATA712	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A88	C3MDLC12	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A89	C3MCONF12	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001A8A	C3MIDL12	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A8C	C3MIDH12	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001A8E	C3MCTRL12	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AA0	C3MDATA013	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA0	C3MDATA0113	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AA1	C3MDATA113	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (71/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001AA2	C3MDATA213	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA2	C3MDATA2313	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AA3	C3MDATA313	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA4	C3MDATA413	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA4	C3MDATA4513	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AA5	C3MDATA513	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA6	C3MDATA613	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA6	C3MDATA6713	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AA7	C3MDATA713	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA8	C3MDLC13	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AA9	C3MCONF13	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AAA	C3MIDL13	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AAC	C3MIDH13	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AAE	C3MCTRL13	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AC0	C3MDATA014	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC0	C3MDATA0114	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AC1	C3MDATA114	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC2	C3MDATA214	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC2	C3MDATA2314	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AC3	C3MDATA314	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC4	C3MDATA414	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC4	C3MDATA4514	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AC5	C3MDATA514	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC6	C3MDATA614	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC6	C3MDATA6714	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AC7	C3MDATA714	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC8	C3MDLC14	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AC9	C3MCONF14	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001ACA	C3MIDL14	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001ACC	C3MIDH14	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001ACE	C3MCTRL14	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AE0	C3MDATA015	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE0	C3MDATA0115	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AE1	C3MDATA115	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE2	C3MDATA215	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE2	C3MDATA2315	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AE3	C3MDATA315	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE4	C3MDATA415	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE4	C3MDATA4515	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AE5	C3MDATA515	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE6	C3MDATA615	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (72/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001AE6	C3MDATA6715	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AE7	C3MDATA715	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE8	C3MDLC15	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AE9	C3MCONF15	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001AEA	C3MIDL15	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AEC	C3MIDH15	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001AEE	C3MCTRL15	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B00	C3MDATA016	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B00	C3MDATA0116	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B01	C3MDATA116	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B02	C3MDATA216	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B02	C3MDATA2316	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B03	C3MDATA316	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B04	C3MDATA416	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B04	C3MDATA4516	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B05	C3MDATA516	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B06	C3MDATA616	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B06	C3MDATA6716	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B07	C3MDATA716	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B08	C3MDLC16	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B09	C3MCONF16	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B0A	C3MIDL16	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B0C	C3MIDH16	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B0E	C3MCTRL16	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B20	C3MDATA017	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B20	C3MDATA0117	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B21	C3MDATA117	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B22	C3MDATA217	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B22	C3MDATA2317	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B23	C3MDATA317	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B24	C3MDATA417	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B24	C3MDATA4517	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B25	C3MDATA517	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B26	C3MDATA617	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B26	C3MDATA6717	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B27	C3MDATA717	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B28	C3MDLC17	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B29	C3MCONF17	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B2A	C3MIDL17	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B2C	C3MIDH17	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B2E	C3MCTRL17	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (73/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001B40	C3MDATA018	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B40	C3MDATA0118	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B41	C3MDATA118	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B42	C3MDATA218	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B42	C3MDATA2318	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B43	C3MDATA318	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x0000DB44	C3MDATA418	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B44	C3MDATA4518	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B45	C3MDATA518	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B46	C3MDATA618	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B46	C3MDATA6718	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B47	C3MDATA718	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B48	C3MDLC18	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B49	C3MCONF18	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B4A	C3MIDL18	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B4C	C3MIDH18	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B4E	C3MCTRL18	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B60	C3MDATA019	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B60	C3MDATA0191	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B61	C3MDATA191	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B62	C3MDATA219	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B62	C3MDATA2319	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B63	C3MDATA319	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B64	C3MDATA419	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B64	C3MDATA4519	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B65	C3MDATA519	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B66	C3MDATA619	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B66	C3MDATA6719	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B67	C3MDATA719	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B68	C3MDLC19	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B69	C3MCONF19	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B6A	C3MIDL19	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B6C	C3MIDH19	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B6E	C3MCTRL19	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B80	C3MDATA020	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B80	C3MDATA0120	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B81	C3MDATA120	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B82	C3MDATA220	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B82	C3MDATA2320	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B83	C3MDATA320	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B84	C3MDATA420	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (74/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001B84	C3MDATA4520	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B85	C3MDATA520	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B86	C3MDATA620	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B86	C3MDATA6720	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B87	C3MDATA720	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B88	C3MDLC20	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B89	C3MCONF20	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001B8A	C3MIDL20	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B8C	C3MIDH20	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001B8E	C3MCTRL20	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BA0	C3MDATA021	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA0	C3MDATA0121	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BA1	C3MDATA121	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA2	C3MDATA221	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA2	C3MDATA2321	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BA3	C3MDATA321	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA4	C3MDATA421	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA4	C3MDATA4521	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BA5	C3MDATA521	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA6	C3MDATA621	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA6	C3MDATA6721	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BA7	C3MDATA721	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA8	C3MDLC21	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BA9	C3MCONF21	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BAA	C3MIDL21	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BAC	C3MIDH21	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BAE	C3MCTRL21	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BC0	C3MDATA022	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC0	C3MDATA0122	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BC1	C3MDATA122	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC2	C3MDATA222	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC2	C3MDATA2322	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BC3	C3MDATA322	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC4	C3MDATA422	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC4	C3MDATA4522	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BC5	C3MDATA522	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC6	C3MDATA622	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC6	C3MDATA6722	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BC7	C3MDATA722	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC8	C3MDLC22	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BC9	C3MCONF22	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (75/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001BCA	C3MIDL22	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BCC	C3MIDH22	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BCE	C3MCTRL22	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BE0	C3MDATA023	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE0	C3MDATA0123	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BE1	C3MDATA123	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE2	C3MDATA223	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE2	C3MDATA2323	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BE3	C3MDATA323	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE4	C3MDATA423	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE4	C3MDATA4523	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BE5	C3MDATA523	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE6	C3MDATA623	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE6	C3MDATA6723	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BE7	C3MDATA723	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE8	C3MDLC23	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BE9	C3MCONF23	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001BEA	C3MIDL23	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BEC	C3MIDH23	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001BEE	C3MCTRL23	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C00	C3MDATA024	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C00	C3MDATA0124	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C01	C3MDATA124	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C02	C3MDATA224	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C02	C3MDATA2324	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C03	C3MDATA324	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C04	C3MDATA424	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C04	C3MDATA4524	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C05	C3MDATA524	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C06	C3MDATA624	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C06	C3MDATA6724	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C07	C3MDATA724	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C08	C3MDLC24	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C09	C3MCONF24	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C0A	C3MIDL24	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C0C	C3MIDH24	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C0E	C3MCTRL24	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C20	C3MDATA025	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C20	C3MDATA0125	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C21	C3MDATA125	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C22	C3MDATA225	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (76/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001C22	C3MDATA2325	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C23	C3MDATA325	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C24	C3MDATA425	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C24	C3MDATA4525	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C25	C3MDATA525	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C26	C3MDATA625	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C26	C3MDATA6725	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C27	C3MDATA725	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C28	C3MDLC25	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C29	C3MCONF25	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C2A	C3MIDL25	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C2C	C3MIDH25	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C2E	C3MCTRL25	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C40	C3MDATA026	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C40	C3MDATA0126	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C41	C3MDATA126	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C42	C3MDATA226	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C42	C3MDATA2326	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C43	C3MDATA326	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C44	C3MDATA426	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C44	C3MDATA4526	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C45	C3MDATA526	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C46	C3MDATA626	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C46	C3MDATA6726	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C47	C3MDATA726	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C48	C3MDLC26	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C49	C3MCONF26	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C4A	C3MIDL26	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C4C	C3MIDH26	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C4E	C3MCTRL26	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C60	C3MDATA027	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C60	C3MDATA0271	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C61	C3MDATA271	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C62	C3MDATA227	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C62	C3MDATA2327	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C63	C3MDATA327	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C64	C3MDATA427	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C64	C3MDATA4527	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C65	C3MDATA527	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C66	C3MDATA627	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C66	C3MDATA6727	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (77/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001C67	C3MDATA727	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C68	C3MDLC27	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C69	C3MCONF27	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C6A	C3MIDL27	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C6C	C3MIDH27	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C6E	C3MCTRL27	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C80	C3MDATA028	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C80	C3MDATA0128	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C81	C3MDATA128	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C82	C3MDATA228	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C82	C3MDATA2328	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C83	C3MDATA328	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C84	C3MDATA428	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C84	C3MDATA4528	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C85	C3MDATA528	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C86	C3MDATA628	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C86	C3MDATA6728	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C87	C3MDATA728	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C88	C3MDLC28	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C89	C3MCONF28	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001C8A	C3MIDL28	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C8C	C3MIDH28	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001C8E	C3MCTRL28	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CA0	C3MDATA029	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA0	C3MDATA0129	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CA1	C3MDATA129	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA2	C3MDATA229	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA2	C3MDATA2329	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CA3	C3MDATA329	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA4	C3MDATA429	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA4	C3MDATA4529	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CA5	C3MDATA529	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA6	C3MDATA629	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA6	C3MDATA6729	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CA7	C3MDATA729	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA8	C3MDLC29	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CA9	C3MCONF29	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CAA	C3MIDL29	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CAC	C3MIDH29	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CAE	C3MCTRL29	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CC0	C3MDATA030	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (78/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001CC0	C3MDATA0130	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CC1	C3MDATA130	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC2	C3MDATA230	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC2	C3MDATA2330	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CC3	C3MDATA330	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC4	C3MDATA430	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC4	C3MDATA4530	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CC5	C3MDATA530	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC6	C3MDATA630	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC6	C3MDATA6730	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CC7	C3MDATA730	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC8	C3MDLC30	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CC9	C3MCONF30	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CCA	C3MIDL30	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CCC	C3MIDH30	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CCE	C3MCTRL30	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CE0	C3MDATA031	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE0	C3MDATA0131	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CE1	C3MDATA131	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE2	C3MDATA231	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE2	C3MDATA2331	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CE3	C3MDATA331	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE4	C3MDATA431	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE4	C3MDATA4531	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CE5	C3MDATA531	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE6	C3MDATA631	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE6	C3MDATA6731	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CE7	C3MDATA731	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE8	C3MDLC31	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CE9	C3MCONF31	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001CEA	C3MIDL31	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CEC	C3MIDH31	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001CEE	C3MCTRL31	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D00	C3MDATA032	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D00	C3MDATA0132	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D01	C3MDATA132	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D02	C3MDATA232	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D02	C3MDATA2332	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D03	C3MDATA332	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D04	C3MDATA432	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D04	C3MDATA4532	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (79/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001D05	C3MDATA532	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D06	C3MDATA632	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D06	C3MDATA6732	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D07	C3MDATA732	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D08	C3MDLC32	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D09	C3MCONF32	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D0A	C3MIDL32	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D0C	C3MIDH32	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D0E	C3MCTRL32	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D20	C3MDATA033	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D20	C3MDATA0133	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D21	C3MDATA133	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D22	C3MDATA233	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D22	C3MDATA2333	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D23	C3MDATA333	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D24	C3MDATA433	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D24	C3MDATA4533	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D25	C3MDATA533	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D26	C3MDATA633	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D26	C3MDATA6733	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D27	C3MDATA733	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D28	C3MDLC33	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D29	C3MCONF33	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D2A	C3MIDL33	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D2C	C3MIDH33	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D2E	C3MCTRL33	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D40	C3MDATA034	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D40	C3MDATA0134	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D41	C3MDATA134	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D42	C3MDATA234	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D42	C3MDATA2334	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D43	C3MDATA334	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D44	C3MDATA434	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D44	C3MDATA4534	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D45	C3MDATA534	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D46	C3MDATA634	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D46	C3MDATA6734	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D47	C3MDATA734	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D48	C3MDLC34	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D49	C3MCONF34	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D4A	C3MIDL34	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (80/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001D4C	C3MIDH34	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D4E	C3MCTRL34	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D60	C3MDATA035	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D60	C3MDATA0351	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D61	C3MDATA351	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D62	C3MDATA235	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D62	C3MDATA2335	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D63	C3MDATA335	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D64	C3MDATA435	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D64	C3MDATA4535	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D65	C3MDATA535	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D66	C3MDATA635	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D66	C3MDATA6735	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D67	C3MDATA735	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D68	C3MDLC35	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D69	C3MCONF35	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D6A	C3MIDL35	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D6C	C3MIDH35	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D6E	C3MCTRL35	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D80	C3MDATA036	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D80	C3MDATA0136	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D81	C3MDATA136	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D82	C3MDATA236	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D82	C3MDATA2336	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D83	C3MDATA336	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D84	C3MDATA436	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D84	C3MDATA4536	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D85	C3MDATA536	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D86	C3MDATA636	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D86	C3MDATA6736	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D87	C3MDATA736	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D88	C3MDLC36	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D89	C3MCONF36	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001D8A	C3MIDL36	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D8C	C3MIDH36	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001D8E	C3MCTRL36	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DA0	C3MDATA037	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA0	C3MDATA0137	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DA1	C3MDATA137	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA2	C3MDATA237	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA2	C3MDATA2337	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (81/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001DA3	C3MDATA337	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA4	C3MDATA437	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA4	C3MDATA4537	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DA5	C3MDATA537	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA6	C3MDATA637	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA6	C3MDATA6737	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DA7	C3MDATA737	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA8	C3MDLC37	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DA9	C3MCONF37	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DAA	C3MIDL37	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DAC	C3MIDH37	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DAE	C3MCTRL37	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DC0	C3MDATA038	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC0	C3MDATA0138	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DC1	C3MDATA138	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC2	C3MDATA238	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC2	C3MDATA2338	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DC3	C3MDATA338	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC4	C3MDATA438	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC4	C3MDATA4538	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DC5	C3MDATA538	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC6	C3MDATA638	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC6	C3MDATA6738	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DC7	C3MDATA738	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC8	C3MDLC38	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DC9	C3MCONF38	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DCA	C3MIDL38	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DCC	C3MIDH38	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DCE	C3MCTRL38	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DE0	C3MDATA039	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE0	C3MDATA0139	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DE1	C3MDATA139	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE2	C3MDATA239	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE2	C3MDATA2339	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DE3	C3MDATA339	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE4	C3MDATA439	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE4	C3MDATA4539	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DE5	C3MDATA539	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE6	C3MDATA639	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE6	C3MDATA6739	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DE7	C3MDATA739	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (82/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001DE8	C3MDLC39	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DE9	C3MCONF39	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001DEA	C3MIDL39	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DEC	C3MIDH39	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001DEE	C3MCTRL39	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E00	C3MDATA040	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E00	C3MDATA0140	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E01	C3MDATA140	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E02	C3MDATA240	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E02	C3MDATA2340	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E03	C3MDATA340	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E04	C3MDATA440	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E04	C3MDATA4540	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E05	C3MDATA540	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E06	C3MDATA640	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E06	C3MDATA6740	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E07	C3MDATA740	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E08	C3MDLC40	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E09	C3MCONF40	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E0A	C3MIDL40	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E0C	C3MIDH40	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E0E	C3MCTRL40	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E20	C3MDATA041	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E20	C3MDATA0141	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E21	C3MDATA141	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E22	C3MDATA241	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E22	C3MDATA2341	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E23	C3MDATA341	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E24	C3MDATA441	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E24	C3MDATA4541	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E25	C3MDATA541	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E26	C3MDATA641	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E26	C3MDATA6741	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E27	C3MDATA741	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E28	C3MDLC41	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E29	C3MCONF41	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E2A	C3MIDL41	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E2C	C3MIDH41	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E2E	C3MCTRL41	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E40	C3MDATA042	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E40	C3MDATA0142	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (83/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001E41	C3MDATA142	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E42	C3MDATA242	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E42	C3MDATA2342	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E43	C3MDATA342	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E44	C3MDATA442	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E44	C3MDATA4542	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E45	C3MDATA542	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E46	C3MDATA642	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E46	C3MDATA6742	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E47	C3MDATA742	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E48	C3MDLC42	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E49	C3MCONF42	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E4A	C3MIDL42	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E4C	C3MIDH42	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E4E	C3MCTRL42	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E60	C3MDATA043	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E60	C3MDATA0143	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E61	C3MDATA143	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E62	C3MDATA243	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E62	C3MDATA2343	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E63	C3MDATA343	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E64	C3MDATA443	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E64	C3MDATA4543	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E65	C3MDATA543	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E66	C3MDATA643	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E66	C3MDATA6743	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E67	C3MDATA743	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E68	C3MDLC43	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E69	C3MCONF43	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E6A	C3MIDL43	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E6C	C3MIDH43	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E6E	C3MCTRL43	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E80	C3MDATA044	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E80	C3MDATA0144	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E81	C3MDATA144	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E82	C3MDATA244	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E82	C3MDATA2344	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E83	C3MDATA344	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E84	C3MDATA444	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E84	C3MDATA4544	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E85	C3MDATA544	CAN3 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (84/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001E86	C3MDATA644	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E86	C3MDATA6744	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E87	C3MDATA744	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E88	C3MDLC44	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E89	C3MCONF44	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001E8A	C3MIDL44	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E8C	C3MIDH44	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001E8E	C3MCTRL44	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EA0	C3MDATA045	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA0	C3MDATA0145	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EA1	C3MDATA145	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA2	C3MDATA245	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA2	C3MDATA2345	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EA3	C3MDATA345	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA4	C3MDATA445	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA4	C3MDATA4545	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EA5	C3MDATA545	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA6	C3MDATA645	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA6	C3MDATA6745	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EA7	C3MDATA745	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA8	C3MDLC45	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EA9	C3MCONF45	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EAA	C3MIDL45	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EAC	C3MIDH45	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EAE	C3MCTRL45	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EC0	C3MDATA046	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC0	C3MDATA0146	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EC1	C3MDATA146	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC2	C3MDATA246	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC2	C3MDATA2346	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EC3	C3MDATA346	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC4	C3MDATA446	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC4	C3MDATA4546	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EC5	C3MDATA546	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC6	C3MDATA646	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC6	C3MDATA6746	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EC7	C3MDATA746	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC8	C3MDLC46	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EC9	C3MCONF46	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001ECA	C3MIDL46	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001ECC	C3MIDH46	CAN3 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (85/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00001ECE	C3MCTRL46	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EE0	C3MDATA047	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE0	C3MDATA0147	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EE1	C3MDATA147	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE2	C3MDATA247	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE2	C3MDATA2347	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EE3	C3MDATA347	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE4	C3MDATA447	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE4	C3MDATA4547	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EE5	C3MDATA547	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE6	C3MDATA647	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE6	C3MDATA6747	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EE7	C3MDATA747	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE8	C3MDLC47	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EE9	C3MCONF47	CAN3 data buffer register	R/W	R/W	-	-	undef.
0x00001EEA	C3MIDL47	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EEC	C3MIDH47	CAN3 data buffer register	-	-	R/W	-	undef.
0x00001EEE	C3MCTRL47	CAN3 data buffer register	-	-	R/W	-	undef.
0x00002000	C4GMCTRLL	CAN4 Global Macro Control Register low byte	R/W	R/W	-	-	0x00
0x00002000	C4GMCTRL	CAN4 Global Macro Control Register	-	-	R/W	-	0x0000
0x00002001	C4GMCTRLH	CAN4 Global Macro Control Register high byte	R/W	R/W	-	-	0x00
0x00002002	C4GMCS	CAN4 Global Macro Clock Selection Register	R/W	R/W	-	-	0x0F
0x00002004	C4GMCONF	CAN4 global configuration register	-	-	R	-	0x0119
0x00002006	C4GMABTL	CAN4 Global Macro Automatic Block Transmission Register low byte	R/W	R/W	-	-	0x00
0x00002006	C4GMABT	CAN4 Global Macro Automatic Block Transmission Register	-	-	R/W	-	0x0000
0x00002007	C4GMABTH	CAN4 Global Macro Automatic Block Transmission Register high byte	R/W	R/W	-	-	0x00
0x00002008	C4GMABTD	CAN4 Global Macro Automatic Block Transmission Delay Register	R/W	R/W	-	-	0x00
0x00002040	C4MASK1L	CAN4 Module Mask 1 Register lower half word	-	-	R/W	-	undef.
0x00002042	C4MASK1H	CAN4 Module Mask 1 Register upper half word	-	-	R/W	-	undef.
0x00002044	C4MASK2L	CAN4 Module Mask 2 Register lower half word	-	-	R/W	-	undef.
0x00002046	C4MASK2H	CAN4 Module Mask 2 Register upper half word	-	-	R/W	-	undef.
0x00002048	C4MASK3L	CAN4 Module Mask 3 Register lower half word	-	-	R/W	-	undef.
0x0000204A	C4MASK3H	CAN4 Module Mask 3 Register upper half word	-	-	R/W	-	undef.
0x0000204C	C4MASK4L	CAN4 Module Mask 4 Register lower half word	-	-	R/W	-	undef.
0x0000204E	C4MASK4H	CAN4 Module Mask 4 Register upper half word	-	-	R/W	-	undef.
0x00002050	C4CTRL	CAN4 Module Control Register	-	-	R/W	-	0x0000

Table 3-7: PPA related SFR Table (86/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002052	C4LEC	CAN4 Module Last Error Code Register	R/W	R/W	-	-	0x00
0x00002053	C4INFO	CAN4 Module Information Register	R	R	-	-	0x00
0x00002054	C4ERC	CAN4 Module Error Counter	-	-	R	-	0x0000
0x00002056	C4IEL	CAN4 Module Interrupt Enable Register low byte	R/W	R/W	-	-	0x00
0x00002056	C4IE	CAN4 Module Interrupt Enable Register	-	-	R/W	-	0x0000
0x00002057	C4IEH	CAN4 Module Interrupt Enable Register high byte	R/W	R/W	-	-	0x00
0x00002058	C4INTSL	CAN4 Module Interrupt Status Register low byte	R/W	R/W	-	-	0x00
0x00002058	C4INTS	CAN4 Module Interrupt Status Register	-	-	R/W	-	0x0000
0x0000205A	C4BRP	CAN4 Module Bit-Rate Prescaler Register	R/W	R/W	-	-	0xFF
0x0000205C	C4BTR	CAN4 Bit Rate Register	-	-	R/W	-	0x370F
0x0000205E	C4LIPT	CAN4 Module Last In-Pointer Register	-	R/W	-	-	undef.
0x00002060	C4RGPTL	CAN4 Module Receive History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00002060	C4RGPT	CAN4 Module Receive History List Get Pointer Register	-	-	R/W	-	undef.
0x00002062	C4LOPT	CAN4 Module Last Out-Pointer Register	-	R	-	-	undef.
0x00002064	C4TGPTL	CAN4 Module Transmit History List Get Pointer Register low byte	R/W	R/W	-	-	0x02
0x00002064	C4TGPT	CAN4 Module Transmit History List Get Pointer Register	-	-	R/W	-	undef.
0x00002066	C4TSL	CAN4 Module Time Stamp Register low byte	R/W	R/W	-	-	0x00
0x00002066	C4TS	CAN4 Module Time Stamp Register	-	-	R/W	-	0x0000
0x00002067	C4TSH	CAN4 Module Time Stamp Register high byte	R/W	R/W	-	-	0x00
0x00002100	C4MDATA000	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002101	C4MDATA100	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002102	C4MDATA200	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002102	C4MDATA2300	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002103	C4MDATA300	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002104	C4MDATA400	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002104	C4MDATA4500	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002105	C4MDATA500	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002106	C4MDATA600	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002106	C4MDATA6700	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002107	C4MDATA700	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002108	C4MDLC00	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002109	C4MCONF00	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000210A	C4MIDL00	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000210C	C4MIDH00	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002102	C4MCTRL00	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (87/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002120	C4MDATA001	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002120	C4MDATA0101	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002121	C4MDATA101	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002122	C4MDATA201	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002122	C4MDATA2301	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002123	C4MDATA301	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002124	C4MDATA401	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002124	C4MDATA4501	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002125	C4MDATA501	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002126	C4MDATA601	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002126	C4MDATA6701	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002127	C4MDATA701	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002128	C4MDLC01	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002129	C4MCONF01	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000212A	C4MIDL01	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000212C	C4MIDH01	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000212E	C4MCTRL01	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002140	C4MDATA002	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002140	C4MDATA0102	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002141	C4MDATA102	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002142	C4MDATA202	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002142	C4MDATA2302	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002143	C4MDATA302	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002144	C4MDATA402	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002144	C4MDATA4502	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002145	C4MDATA502	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002146	C4MDATA602	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002146	C4MDATA6702	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002147	C4MDATA702	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002148	C4MDLC02	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002149	C4MCONF02	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000214A	C4MIDL02	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000214C	C4MIDH02	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000214E	C4MCTRL02	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002160	C4MDATA003	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002160	C4MDATA0103	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002161	C4MDATA103	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002162	C4MDATA203	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002162	C4MDATA2303	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002163	C4MDATA303	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002164	C4MDATA403	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (88/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002164	C4MDATA4503	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002165	C4MDATA503	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002166	C4MDATA603	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002166	C4MDATA6703	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002167	C4MDATA703	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002168	C4MDLC03	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002169	C4MCONF03	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000216A	C4MIDL03	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000216C	C4MIDH03	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000216E	C4MCTRL03	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002180	C4MDATA004	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002180	C4MDATA0104	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002181	C4MDATA104	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002182	C4MDATA204	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002182	C4MDATA2304	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002183	C4MDATA304	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002184	C4MDATA404	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002184	C4MDATA4504	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002185	C4MDATA504	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002186	C4MDATA604	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002186	C4MDATA6704	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002187	C4MDATA704	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002188	C4MDLC04	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002189	C4MCONF04	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000218A	C4MIDL04	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000218C	C4MIDH04	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000218E	C4MCTRL04	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021A0	C4MDATA005	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A0	C4MDATA0105	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021A1	C4MDATA105	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A2	C4MDATA205	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A2	C4MDATA2305	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021A3	C4MDATA305	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A4	C4MDATA405	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A4	C4MDATA4505	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021A5	C4MDATA505	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A6	C4MDATA605	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A6	C4MDATA6705	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021A7	C4MDATA705	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A8	C4MDLC05	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021A9	C4MCONF05	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (89/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000021AA	C4MIDL05	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021AC	C4MIDH05	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021AE	C4MCTRL05	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021C0	C4MDATA006	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C0	C4MDATA0106	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021C1	C4MDATA106	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C2	C4MDATA206	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C2	C4MDATA2306	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021C3	C4MDATA306	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C4	C4MDATA406	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C4	C4MDATA4506	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021C5	C4MDATA506	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C6	C4MDATA606	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C6	C4MDATA6706	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021C7	C4MDATA706	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C8	C4MDLC06	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021C9	C4MCONF06	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021CA	C4MIDL06	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021CC	C4MIDH06	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021CE	C4MCTRL06	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021E0	C4MDATA007	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E0	C4MDATA0107	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021E1	C4MDATA107	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E2	C4MDATA207	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E2	C4MDATA2307	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021E3	C4MDATA307	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E4	C4MDATA407	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E4	C4MDATA4507	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021E5	C4MDATA507	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E6	C4MDATA607	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E6	C4MDATA6707	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021E7	C4MDATA707	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E8	C4MDLC07	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021E9	C4MCONF07	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000021EA	C4MIDL07	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021EC	C4MIDH07	CAN4 data buffer register	-	-	R/W	-	undef.
0x000021EE	C4MCTRL07	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002200	C4MDATA008	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002200	C4MDATA0108	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002201	C4MDATA108	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002202	C4MDATA208	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (90/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002202	C4MDATA2308	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002203	C4MDATA308	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002204	C4MDATA408	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002204	C4MDATA4508	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002205	C4MDATA508	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002206	C4MDATA608	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002206	C4MDATA6708	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002207	C4MDATA708	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002208	C4MDLC08	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002209	C4MCONF08	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000220A	C4MIDL08	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000220C	C4MIDH08	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002202	C4MCTRL08	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002220	C4MDATA009	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002220	C4MDATA0109	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002221	C4MDATA109	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002222	C4MDATA209	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002222	C4MDATA2309	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002223	C4MDATA309	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002224	C4MDATA409	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002224	C4MDATA4509	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002225	C4MDATA509	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002226	C4MDATA609	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002226	C4MDATA6709	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002227	C4MDATA709	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002228	C4MDLC09	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002229	C4MCONF09	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000222A	C4MIDL09	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000222C	C4MIDH09	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000222E	C4MCTRL09	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002240	C4MDATA010	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002240	C4MDATA0110	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002241	C4MDATA110	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002242	C4MDATA210	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002242	C4MDATA2310	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002243	C4MDATA310	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002244	C4MDATA410	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002244	C4MDATA4510	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002245	C4MDATA510	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002246	C4MDATA610	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002246	C4MDATA6710	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (91/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002247	C4MDATA710	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002248	C4MDLC10	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002249	C4MCONF10	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000224A	C4MIDL10	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000224C	C4MIDH10	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000224E	C4MCTRL10	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002260	C4MDATA011	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002260	C4MDATA0111	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002261	C4MDATA111	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002262	C4MDATA211	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002262	C4MDATA2311	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002263	C4MDATA311	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002264	C4MDATA411	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002264	C4MDATA4511	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002265	C4MDATA511	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002266	C4MDATA611	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002266	C4MDATA6711	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002267	C4MDATA711	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002268	C4MDLC11	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002269	C4MCONF11	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000226A	C4MIDL11	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000226C	C4MIDH11	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000226E	C4MCTRL11	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002280	C4MDATA012	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002280	C4MDATA0112	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002281	C4MDATA112	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002282	C4MDATA212	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002282	C4MDATA2312	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002283	C4MDATA312	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002284	C4MDATA412	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002284	C4MDATA4512	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002285	C4MDATA512	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002286	C4MDATA612	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002286	C4MDATA6712	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002287	C4MDATA712	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002288	C4MDLC12	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002289	C4MCONF12	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000228A	C4MIDL12	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000228C	C4MIDH12	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000228E	C4MCTRL12	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022A0	C4MDATA013	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (92/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000022A0	C4MDATA0113	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022A1	C4MDATA113	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A2	C4MDATA213	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A2	C4MDATA2313	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022A3	C4MDATA313	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A4	C4MDATA413	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A4	C4MDATA4513	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022A5	C4MDATA513	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A6	C4MDATA613	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A6	C4MDATA6713	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022A7	C4MDATA713	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A8	C4MDLC13	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022A9	C4MCONF13	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022AA	C4MIDL13	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022AC	C4MIDH13	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022AE	C4MCTRL13	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022C0	C4MDATA014	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C0	C4MDATA0114	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022C1	C4MDATA114	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C2	C4MDATA214	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C2	C4MDATA2314	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022C3	C4MDATA314	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C4	C4MDATA414	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C4	C4MDATA4514	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022C5	C4MDATA514	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C6	C4MDATA614	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C6	C4MDATA6714	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022C7	C4MDATA714	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C8	C4MDLC14	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022C9	C4MCONF14	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022CA	C4MIDL14	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022CC	C4MIDH14	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022CE	C4MCTRL14	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022E0	C4MDATA015	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E0	C4MDATA0115	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022E1	C4MDATA115	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E2	C4MDATA215	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E2	C4MDATA2315	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022E3	C4MDATA315	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E4	C4MDATA415	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E4	C4MDATA4515	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (93/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000022E5	C4MDATA515	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E6	C4MDATA615	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E6	C4MDATA6715	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022E7	C4MDATA715	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E8	C4MDLC15	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022E9	C4MCONF15	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000022EA	C4MIDL15	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022EC	C4MIDH15	CAN4 data buffer register	-	-	R/W	-	undef.
0x000022EE	C4MCTRL15	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002300	C4MDATA016	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002300	C4MDATA0116	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002301	C4MDATA116	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002302	C4MDATA216	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002302	C4MDATA2316	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002303	C4MDATA316	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002304	C4MDATA416	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002304	C4MDATA4516	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002305	C4MDATA516	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002306	C4MDATA616	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002306	C4MDATA6716	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002307	C4MDATA716	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002308	C4MDLC16	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002309	C4MCONF16	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000230A	C4MIDL16	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000230C	C4MIDH16	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002302	C4MCTRL16	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002320	C4MDATA017	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002320	C4MDATA0117	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002321	C4MDATA117	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002322	C4MDATA217	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002322	C4MDATA2317	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002323	C4MDATA317	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002324	C4MDATA417	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002324	C4MDATA4517	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002325	C4MDATA517	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002326	C4MDATA617	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002326	C4MDATA6717	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002327	C4MDATA717	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002328	C4MDLC17	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002329	C4MCONF17	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000232A	C4MIDL17	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (94/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x0000232C	C4MIDH17	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000232E	C4MCTRL17	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002340	C4MDATA018	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002340	C4MDATA0118	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002341	C4MDATA118	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002342	C4MDATA218	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002342	C4MDATA2318	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002343	C4MDATA318	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002344	C4MDATA418	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002344	C4MDATA4518	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002345	C4MDATA518	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002346	C4MDATA618	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002346	C4MDATA6718	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002347	C4MDATA718	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002348	C4MDLC18	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002349	C4MCONF18	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000234A	C4MIDL18	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000234C	C4MIDH18	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000234E	C4MCTRL18	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002360	C4MDATA019	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002360	C4MDATA0191	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002361	C4MDATA191	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002362	C4MDATA219	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002362	C4MDATA2319	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002363	C4MDATA319	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002364	C4MDATA419	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002364	C4MDATA4519	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002365	C4MDATA519	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002366	C4MDATA619	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002366	C4MDATA6719	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002367	C4MDATA719	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002368	C4MDLC19	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002369	C4MCONF19	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000236A	C4MIDL19	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000236C	C4MIDH19	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000236E	C4MCTRL19	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002380	C4MDATA020	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002380	C4MDATA0120	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002381	C4MDATA120	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002382	C4MDATA220	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002382	C4MDATA2320	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (95/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002383	C4MDATA320	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002384	C4MDATA420	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002384	C4MDATA4520	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002385	C4MDATA520	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002386	C4MDATA620	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002386	C4MDATA6720	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002387	C4MDATA720	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002388	C4MDLC20	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002389	C4MCONF20	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000238A	C4MIDL20	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000238C	C4MIDH20	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000238E	C4MCTRL20	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023A0	C4MDATA021	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A0	C4MDATA0121	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023A1	C4MDATA121	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A2	C4MDATA221	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A2	C4MDATA2321	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023A3	C4MDATA321	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A4	C4MDATA421	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A4	C4MDATA4521	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023A5	C4MDATA521	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A6	C4MDATA621	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A6	C4MDATA6721	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023A7	C4MDATA721	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A8	C4MDLC21	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023A9	C4MCONF21	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023AA	C4MIDL21	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023AC	C4MIDH21	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023AE	C4MCTRL21	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023C0	C4MDATA022	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C0	C4MDATA0122	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023C1	C4MDATA122	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C2	C4MDATA222	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C2	C4MDATA2322	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023C3	C4MDATA322	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C4	C4MDATA422	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C4	C4MDATA4522	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023C5	C4MDATA522	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C6	C4MDATA622	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C6	C4MDATA6722	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023C7	C4MDATA722	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (96/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000023C8	C4MDLC22	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023C9	C4MCONF22	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023CA	C4MIDL22	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023CC	C4MIDH22	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023CE	C4MCTRL22	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023E0	C4MDATA023	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E0	C4MDATA0123	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023E1	C4MDATA123	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E2	C4MDATA223	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E2	C4MDATA2323	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023E3	C4MDATA323	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E4	C4MDATA423	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E4	C4MDATA4523	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023E5	C4MDATA523	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E6	C4MDATA623	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E6	C4MDATA6723	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023E7	C4MDATA723	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E8	C4MDLC23	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023E9	C4MCONF23	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000023EA	C4MIDL23	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023EC	C4MIDH23	CAN4 data buffer register	-	-	R/W	-	undef.
0x000023EE	C4MCTRL23	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002400	C4MDATA024	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002400	C4MDATA0124	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002401	C4MDATA124	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002402	C4MDATA224	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002402	C4MDATA2324	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002403	C4MDATA324	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002404	C4MDATA424	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002404	C4MDATA4524	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002405	C4MDATA524	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002406	C4MDATA624	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002406	C4MDATA6724	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002407	C4MDATA724	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002408	C4MDLC24	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002409	C4MCONF24	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000240A	C4MIDL24	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000240C	C4MIDH24	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002402	C4MCTRL24	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002420	C4MDATA025	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002420	C4MDATA0125	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (97/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002421	C4MDATA125	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002422	C4MDATA225	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002422	C4MDATA2325	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002423	C4MDATA325	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002424	C4MDATA425	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002424	C4MDATA4525	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002425	C4MDATA525	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002426	C4MDATA625	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002426	C4MDATA6725	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002427	C4MDATA725	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002428	C4MDLC25	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002429	C4MCONF25	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000242A	C4MIDL25	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000242C	C4MIDH25	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000242E	C4MCTRL25	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002440	C4MDATA026	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002440	C4MDATA0126	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002441	C4MDATA126	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002442	C4MDATA226	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002442	C4MDATA2326	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002443	C4MDATA326	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002444	C4MDATA426	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002444	C4MDATA4526	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002445	C4MDATA526	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002446	C4MDATA626	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002446	C4MDATA6726	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002447	C4MDATA726	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002448	C4MDLC26	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002449	C4MCONF26	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000244A	C4MIDL26	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000244C	C4MIDH26	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000244E	C4MCTRL26	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002460	C4MDATA027	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002460	C4MDATA0271	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002461	C4MDATA271	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002462	C4MDATA227	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002462	C4MDATA2327	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002463	C4MDATA327	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002464	C4MDATA427	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002464	C4MDATA4527	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002465	C4MDATA527	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (98/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002466	C4MDATA627	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002466	C4MDATA6727	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002467	C4MDATA727	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002468	C4MDLC27	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002469	C4MCONF27	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000246A	C4MIDL27	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000246C	C4MIDH27	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000246E	C4MCTRL27	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002480	C4MDATA028	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002480	C4MDATA0128	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002481	C4MDATA128	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002482	C4MDATA228	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002482	C4MDATA2328	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002483	C4MDATA328	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002484	C4MDATA428	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002484	C4MDATA4528	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002485	C4MDATA528	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002486	C4MDATA628	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002486	C4MDATA6728	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002487	C4MDATA728	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002488	C4MDLC28	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002489	C4MCONF28	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000248A	C4MIDL28	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000248C	C4MIDH28	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000248E	C4MCTRL28	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024A0	C4MDATA029	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A0	C4MDATA0129	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024A1	C4MDATA129	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A2	C4MDATA229	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A2	C4MDATA2329	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024A3	C4MDATA329	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A4	C4MDATA429	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A4	C4MDATA4529	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024A5	C4MDATA529	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A6	C4MDATA629	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A6	C4MDATA6729	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024A7	C4MDATA729	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A8	C4MDLC29	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024A9	C4MCONF29	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024AA	C4MIDL29	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024AC	C4MIDH29	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (99/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000024AE	C4MCTRL29	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024C0	C4MDATA030	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C0	C4MDATA0130	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024C1	C4MDATA130	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C2	C4MDATA230	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C2	C4MDATA2330	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024C3	C4MDATA330	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C4	C4MDATA430	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C4	C4MDATA4530	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024C5	C4MDATA530	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C6	C4MDATA630	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C6	C4MDATA6730	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024C7	C4MDATA730	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C8	C4MDLC30	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024C9	C4MCONF30	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024CA	C4MIDL30	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024CC	C4MIDH30	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024CE	C4MCTRL30	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024E0	C4MDATA031	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E0	C4MDATA0131	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024E1	C4MDATA131	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E2	C4MDATA231	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E2	C4MDATA2331	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024E3	C4MDATA331	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E4	C4MDATA431	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E4	C4MDATA4531	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024E5	C4MDATA531	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E6	C4MDATA631	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E6	C4MDATA6731	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024E7	C4MDATA731	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E8	C4MDLC31	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024E9	C4MCONF31	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000024EA	C4MIDL31	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024EC	C4MIDH31	CAN4 data buffer register	-	-	R/W	-	undef.
0x000024EE	C4MCTRL31	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002500	C4MDATA032	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002500	C4MDATA0132	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002501	C4MDATA132	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002502	C4MDATA232	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002502	C4MDATA2332	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002503	C4MDATA332	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (100/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002504	C4MDATA432	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002504	C4MDATA4532	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002505	C4MDATA532	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002506	C4MDATA632	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002506	C4MDATA6732	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002507	C4MDATA732	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002508	C4MDLC32	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002509	C4MCONF32	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000250A	C4MIDL32	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000250C	C4MIDH32	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002502	C4MCTRL32	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002520	C4MDATA033	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002520	C4MDATA0133	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002521	C4MDATA133	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002522	C4MDATA233	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002522	C4MDATA2333	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002523	C4MDATA333	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002524	C4MDATA433	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002524	C4MDATA4533	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002525	C4MDATA533	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002526	C4MDATA633	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002526	C4MDATA6733	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002527	C4MDATA733	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002528	C4MDLC33	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002529	C4MCONF33	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000252A	C4MIDL33	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000252C	C4MIDH33	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000252E	C4MCTRL33	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002540	C4MDATA034	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002540	C4MDATA0134	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002541	C4MDATA134	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002542	C4MDATA234	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002542	C4MDATA2334	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002543	C4MDATA334	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002544	C4MDATA434	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002544	C4MDATA4534	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002545	C4MDATA534	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002546	C4MDATA634	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002546	C4MDATA6734	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002547	C4MDATA734	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002548	C4MDLC34	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (101/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002549	C4MCONF34	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000254A	C4MIDL34	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000254C	C4MIDH34	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000254E	C4MCTRL34	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002560	C4MDATA035	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002560	C4MDATA0351	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002561	C4MDATA351	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002562	C4MDATA235	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002562	C4MDATA2335	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002563	C4MDATA335	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002564	C4MDATA435	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002564	C4MDATA4535	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002565	C4MDATA535	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002566	C4MDATA635	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002566	C4MDATA6735	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002567	C4MDATA735	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002568	C4MDLC35	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002569	C4MCONF35	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000256A	C4MIDL35	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000256C	C4MIDH35	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000256E	C4MCTRL35	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002580	C4MDATA036	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002580	C4MDATA0136	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002581	C4MDATA136	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002582	C4MDATA236	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002582	C4MDATA2336	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002583	C4MDATA336	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002584	C4MDATA436	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002584	C4MDATA4536	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002585	C4MDATA536	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002586	C4MDATA636	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002586	C4MDATA6736	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002587	C4MDATA736	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002588	C4MDLC36	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002589	C4MCONF36	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000258A	C4MIDL36	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000258C	C4MIDH36	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000258E	C4MCTRL36	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025A0	C4MDATA037	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A0	C4MDATA0137	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025A1	C4MDATA137	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (102/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000025A2	C4MDATA237	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A2	C4MDATA2337	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025A3	C4MDATA337	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A4	C4MDATA437	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A4	C4MDATA4537	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025A5	C4MDATA537	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A6	C4MDATA637	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A6	C4MDATA6737	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025A7	C4MDATA737	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A8	C4MDLC37	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025A9	C4MCONF37	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025AA	C4MIDL37	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025AC	C4MIDH37	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025AE	C4MCTRL37	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025C0	C4MDATA038	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C0	C4MDATA0138	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025C1	C4MDATA138	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C2	C4MDATA238	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C2	C4MDATA2338	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025C3	C4MDATA338	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C4	C4MDATA438	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C4	C4MDATA4538	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025C5	C4MDATA538	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C6	C4MDATA638	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C6	C4MDATA6738	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025C7	C4MDATA738	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C8	C4MDLC38	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025C9	C4MCONF38	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025CA	C4MIDL38	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025CC	C4MIDH38	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025CE	C4MCTRL38	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025E0	C4MDATA039	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E0	C4MDATA0139	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025E1	C4MDATA139	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E2	C4MDATA239	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E2	C4MDATA2339	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025E3	C4MDATA339	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E4	C4MDATA439	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E4	C4MDATA4539	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025E5	C4MDATA539	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E6	C4MDATA639	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (103/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000025E6	C4MDATA6739	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025E7	C4MDATA739	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E8	C4MDLC39	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025E9	C4MCONF39	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000025EA	C4MIDL39	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025EC	C4MIDH39	CAN4 data buffer register	-	-	R/W	-	undef.
0x000025EE	C4MCTRL39	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002600	C4MDATA040	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002600	C4MDATA0140	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002601	C4MDATA140	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002602	C4MDATA240	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002602	C4MDATA2340	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002603	C4MDATA340	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002604	C4MDATA440	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002604	C4MDATA4540	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002605	C4MDATA540	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002606	C4MDATA640	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002606	C4MDATA6740	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002607	C4MDATA740	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002608	C4MDLC40	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002609	C4MCONF40	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000260A	C4MIDL40	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000260C	C4MIDH40	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002602	C4MCTRL40	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002620	C4MDATA041	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002620	C4MDATA0141	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002621	C4MDATA141	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002622	C4MDATA241	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002622	C4MDATA2341	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002623	C4MDATA341	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002624	C4MDATA441	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002624	C4MDATA4541	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002625	C4MDATA541	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002626	C4MDATA641	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002626	C4MDATA6741	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002627	C4MDATA741	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002628	C4MDLC41	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002629	C4MCONF41	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000262A	C4MIDL41	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000262C	C4MIDH41	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000262E	C4MCTRL41	CAN4 data buffer register	-	-	R/W	-	undef.

Table 3-7: PPA related SFR Table (104/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002640	C4MDATA042	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002640	C4MDATA0142	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002641	C4MDATA142	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002642	C4MDATA242	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002642	C4MDATA2342	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002643	C4MDATA342	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002644	C4MDATA442	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002644	C4MDATA4542	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002645	C4MDATA542	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002646	C4MDATA642	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002646	C4MDATA6742	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002647	C4MDATA742	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002648	C4MDLC42	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002649	C4MCONF42	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000264A	C4MIDL42	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000264C	C4MIDH42	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000264E	C4MCTRL42	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002660	C4MDATA043	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002660	C4MDATA0143	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002661	C4MDATA143	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002662	C4MDATA243	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002662	C4MDATA2343	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002663	C4MDATA343	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002664	C4MDATA443	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002664	C4MDATA4543	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002665	C4MDATA543	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002666	C4MDATA643	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002666	C4MDATA6743	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002667	C4MDATA743	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002668	C4MDLC43	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002669	C4MCONF43	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000266A	C4MIDL43	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000266C	C4MIDH43	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000266E	C4MCTRL43	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002680	C4MDATA044	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002680	C4MDATA0144	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002681	C4MDATA144	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002682	C4MDATA244	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002682	C4MDATA2344	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002683	C4MDATA344	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002684	C4MDATA444	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (105/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x00002684	C4MDATA4544	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002685	C4MDATA544	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002686	C4MDATA644	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002686	C4MDATA6744	CAN4 data buffer register	-	-	R/W	-	undef.
0x00002687	C4MDATA744	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002688	C4MDLC44	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x00002689	C4MCONF44	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x0000268A	C4MIDL44	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000268C	C4MIDH44	CAN4 data buffer register	-	-	R/W	-	undef.
0x0000268E	C4MCTRL44	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026A0	C4MDATA045	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A0	C4MDATA0145	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026A1	C4MDATA145	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A2	C4MDATA245	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A2	C4MDATA2345	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026A3	C4MDATA345	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A4	C4MDATA445	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A4	C4MDATA4545	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026A5	C4MDATA545	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A6	C4MDATA645	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A6	C4MDATA6745	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026A7	C4MDATA745	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A8	C4MDLC45	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026A9	C4MCONF45	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026AA	C4MIDL45	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026AC	C4MIDH45	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026AE	C4MCTRL45	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026C0	C4MDATA046	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C0	C4MDATA0146	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026C1	C4MDATA146	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C2	C4MDATA246	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C2	C4MDATA2346	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026C3	C4MDATA346	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C4	C4MDATA446	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C4	C4MDATA4546	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026C5	C4MDATA546	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C6	C4MDATA646	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C6	C4MDATA6746	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026C7	C4MDATA746	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C8	C4MDLC46	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026C9	C4MCONF46	CAN4 data buffer register	R/W	R/W	-	-	undef.

Table 3-7: PPA related SFR Table (106/106)

Address Offset	SFR	Description	1-bit	8-bit	16-bit	32-bit	Reset
0x000026CA	C4MIDL46	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026CC	C4MIDH46	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026CE	C4MCTRL46	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026E0	C4MDATA047	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E0	C4MDATA0147	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026E1	C4MDATA147	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E2	C4MDATA247	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E2	C4MDATA2347	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026E3	C4MDATA347	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E4	C4MDATA447	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E4	C4MDATA4547	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026E5	C4MDATA547	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E6	C4MDATA647	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E6	C4MDATA6747	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026E7	C4MDATA747	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E8	C4MDLC47	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026E9	C4MCONF47	CAN4 data buffer register	R/W	R/W	-	-	undef.
0x000026EA	C4MIDL47	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026EC	C4MIDH47	CAN4 data buffer register	-	-	R/W	-	undef.
0x000026EE	C4MCTRL47	CAN4 data buffer register	-	-	R/W	-	undef.

3.7.1 Programmable peripheral I/O registers

In the μPD70F3178, the 16 KB area of x0000H to x3FFFH is provided as a programmable peripheral I/O area. In this area, the area between x0000H and x1200H is used exclusively for the FCAN controller.

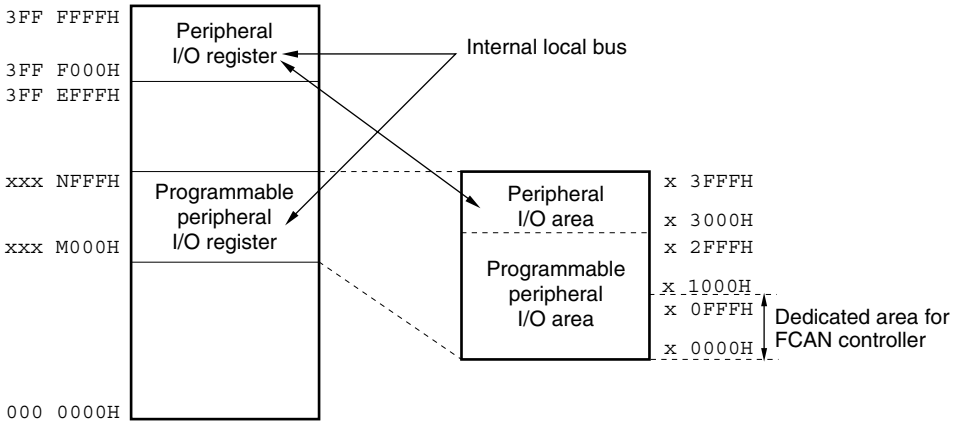
The internal bus of the μPD70F3178 becomes active when
- the peripheral I/O register area (3FF F000H to 3FF FFFFH) or
- the programmable peripheral I/O register area (xxxx m000H to xxxx nFFFH)

is accessed (m = xx00B, n = xx11B).

Note that when data is written to the peripheral I/O register area, the written contents are reflected also on the upper 4 KB area of the programmable peripheral I/O area.

The base address of the programmable peripheral I/O area is specified by the initialization of the peripheral area selection control register (BPC).

Figure 3-13: Programmable Peripheral I/O Register (Outline)



Cautions: 1. The CAN message buffer register can allocate address xxxx freely as a programmable peripheral I/O register. but once the address xxxx is set, it cannot be changed.

2. If the programmable peripheral I/O area overlaps the following areas, the programmable peripheral I/O area becomes ineffective.
- **Peripheral I/O area**
- **ROM area**
- **RAM area**

Remark: M = xx00B
N = xx11B

(1) Peripheral area selection control register (BPC)

The BPC register is a 16-bit register that specifies the base address or the programmable peripheral area.

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

Figure 3-14: Peripheral Area Selection Control Register (BPC)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BPC	PA15	0	PA13	PA12	PA11	PA10	PA9	PA8	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	FFFF F064H	0000H

Bit Position	Bit Name	Function
15	PA15	Enables/disables usage of programmable peripheral I/O area. PA15 — Usage of Programmable Peripheral I/O Area 0 — Disables usage of programmable peripheral I/O area 1 — Enables usage of programmable peripheral I/O area
13 to 0	PA13 to PA0	Specifies an address in programmable peripheral I/O area (corresponds to A27 to A14, respectively).

Remark: For μPD70F3178, the recommended setting of the BPC setting is 87FFH.
With that initialization the base address of the programmable peripheral area is located at 01FF C000H.
Therefore the FCAN macro is mapped to the memory location 1FF C000H to 1FF E700H.

3.8 Specific Registers

Specific registers are registers that are protected from being written with illegal data due to erroneous program execution, etc. The write access of these specific registers is executed in a specific sequence, and if abnormal store operations occur, it is notified by the peripheral status register (PHS).
The µPD70F3178 CarGate+ has 1 specific register- the power save control register (PSC).
For details of the PSC register please refer to the chapter **5.4 ”Power Saving Functions” on page 221**.

The access sequence to the specified registers is shown below.
The following sequence shows the data setting of the specific registers.

- Store instruction (ST/SST instruction)
- Bit operation instruction (SET1/CLR1/NOT1 instruction)

Please see the following example for initialization of a power save mode. The PSC register is a specific register and therefore the PRCMD register has to be written first. The following 5 NOPs are necessary for waken from the STOP mode.

Example

```
<1>  MOV    0x02,r10
<2>  ST.B   r10,PRCMD[r0]
<3>  ST.B   r10,PSC[r0]
<4>  NOP    dummy instruction (5 times NOP required)
```

No special sequence is required when reading the specific registers.

Remarks:

1. A store instruction to a command register will not be received with an interrupt.
   This presupposes that this is done with the continuous store instructions in <1> and <2> above in the program. If another instruction is placed between <1> and <2>, when an interrupt is received by that instruction, the above sequence may not be established, and cause a malfunction, so caution is necessary.
2. The data written in the PRCMD register is dummy data, but use the same general purpose register for writing to the PRCMD register (<2> in the example above) as was used in setting data in the specified register (<3> in the example above). Addressing is the same in the case where a general purpose register is used.
3. In a store instruction to the PSC register for setting it in the software STOP mode or IDLE mode, it is necessary to insert 1 or more NOP instructions just after. When clearing each power save mode by interrupt, or when resetting after executing interrupt processing, start executing from the next instruction without executing 1 instruction just after the store instruction.

3.8.1 Command register (PRCMD)

This command register (PRCMD) is to protect the registers that may have a significant influence on the application system (PSC, PSM) from an inadvertent write access, so that the system does not stop in case of a program hang-up.
This register can only be written in 8-bit units (undefined data is used when this register is read).
Only the first write access to a specific on-chip register (hereafter referred to as a “specific register”) after data has been written to the PRCMD register is valid.
In this way, the value of the specific register can be rewritten only in a specified sequence, and an illegal write access is inhibited.

Figure 3-15: Command Register (PRCMD) Format

	7	6	5	4	3	2	1	0	Address	R/W	At Reset
PRCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0	FFFFF1FCH	R/W	xxH

Remark: REG7 to REG0: registration code, same as used for PSC.

Caution: **Caution: The register must be written with store instruction execution by CPU. DMA transfer is prohibited.**

3.8.2 Internal peripheral function wait control register (VSWC)

This register inserts wait states to the internal access of peripheral SFRs.
This register can be read or written in 1-bit and 8-bit units.

Figure 3-16: Internal Peripheral Function Wait Control Register (VSWC) Format

	7	6	5	4	3	2	1	0	Address	R/W	Reset Value
VSWC	0	SUWL2	SUWL1	SUWL0	0	VSWL2	VSWL1	VSWL0	FFFFF06EH	R/W	77H
	0	1	1	1	0	1	1	1			

Bit Name	Description
SUWL2, SUWL1, SUWL0	Setup wait for internal peripheral bus length
VSWL2, VSWL1, VSWL0	Internal peripheral bus wait length

SUWL2	SUWL1	SUWL0	Number of data wait states (n = 7 - 0)
0	0	0	0
0	0	1	1 system clock
0	1	0	2 system clock
0	1	1	3 system clock
1	0	0	4 system clock
1	0	1	5 system clock
1	1	0	6 system clock
1	1	1	7 system clock (default)

VSWL2	VSWL1	VSWL0	Number of data wait states (n = 7 - 0)
0	0	0	0
0	0	1	1 system clock
0	1	0	2 system clock
0	1	1	3 system clock
1	0	0	4 system clock
1	0	1	5 system clock
1	1	0	6 system clock
1	1	1	7 system clock (default)

Caution: With respect to the specified operation frequency the following register settings for VSWC are recommended.

Table 3-8: The Values of VSWC Register depending on System Clock

System Clock	Setup Wait	Strobe Wait	VSWC
4.0 MHz < f_{CPU} < 16.6 MHz	0	0	00H
16.6 MHz < f_{CPU} < 25.0 MHz	0	1	01H
25.0 MHz < f_{CPU} < 32.0 MHz	1	1	11H

Chapter 4 Bus Control Function

The μPD70F3178 CarGate+ is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

4.1 Features

- 16-bit/8-bit data bus sizing function
- 8 chip areas select function
- 3 chip area select signals externally available ($\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$)
- Wait function
- Programmable wait function, capable of inserting up to 7 wait states for each memory block
- Idle state insertion function
- External device connection can be enabled via bus control/port alternate function pins

4.2 Bus Control Pins

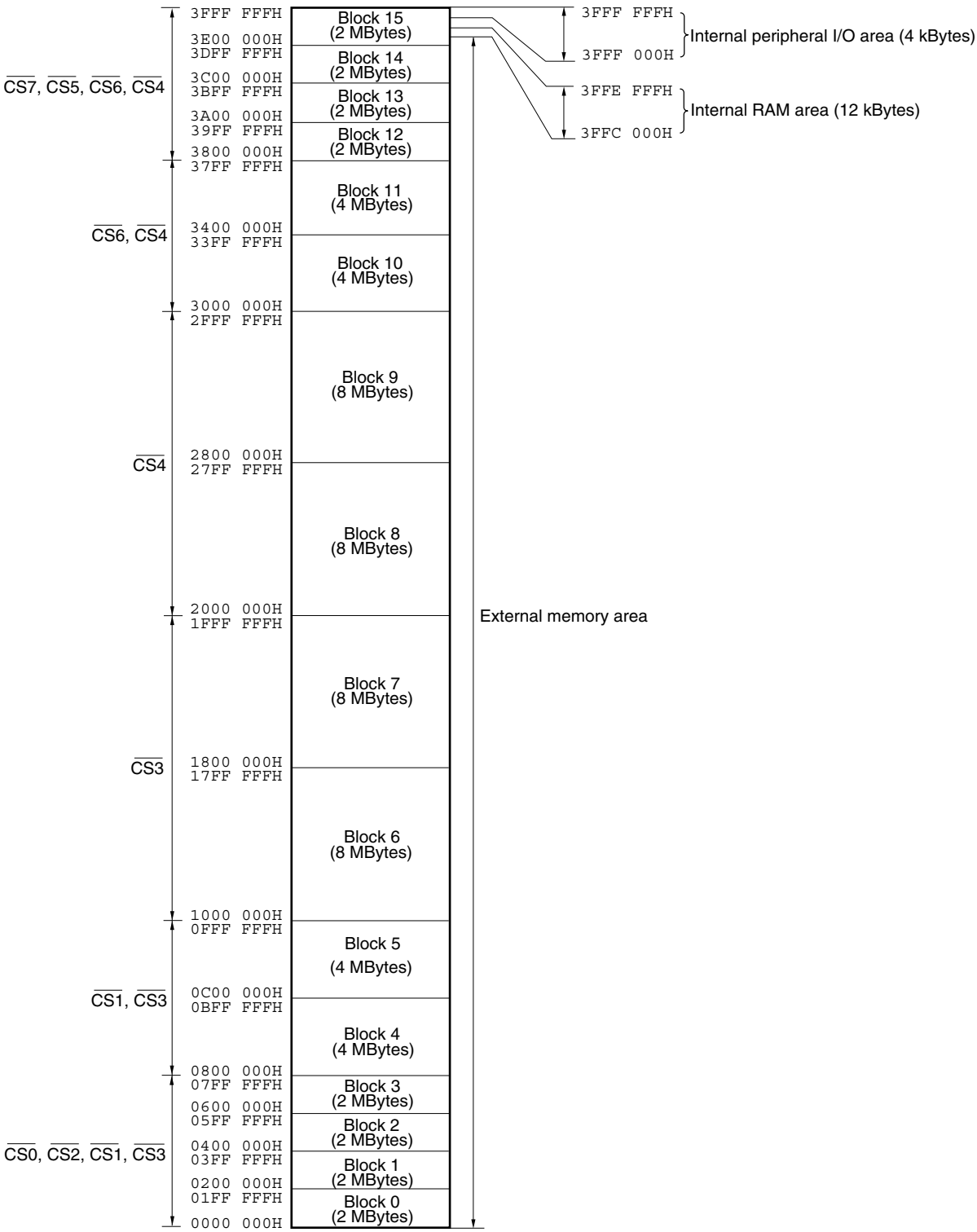
The following pins are used for connecting to external devices.

Bus Control Pin (Function when in Control Mode)	Function when in Port Mode	Register for Port/Control Mode Switching
Address data Data bus (D0 to D15)	PDL0 to PDL15 (Port DL)	PMCDL
Address bus (A0 to A15)	PAL0 to PAL15	PMCAL
Address bus (A16 to A19)	PAH0 to PAH3 (Port AH)	PMCAH
Chip select ($\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$)	PCS0, PCS3 and PCS4 (Port CS)	PMCCS
Read/write control ($\overline{WR0}$, $\overline{WR1}$, $\overline{RD}$)	PCT0, PCT1, PCT4 (Port CT)	PMCCT

4.3 Memory Block Function

The 64 MB memory space is divided into memory blocks of 2 MB, 4 MB, and 8 MB units.

Figure 4-1: Memory Block Function



4.3.1 Chip select control function

The 64 MB memory area can be divided into 2 MB, 4 MB and 8 MB memory blocks by the chip area selection control registers 0 and 1 (CSC0, CSC1) to control the chip select signals.
The memory area can be effectively used by dividing the memory area into memory blocks using the chip select control function. The priority order is described below.

(1) Chip area selection control registers 0, 1(CSC0, CSC1)

These registers can be read/written in 16-bit units. Valid by setting each bit (to 1). If different chip area select signals are set to the same block, the priority order is controlled as follows.

CSC0: Peripheral I/O area > $\overline{CS0}$ > $\overline{CS2}$ > $\overline{CS1}$ > $\overline{CS3}$ **Note**
CSC1: Peripheral I/O area > $\overline{CS7}$ > $\overline{CS5}$ > $\overline{CS6}$ > $\overline{CS4}$ **Note**

Note: Not all the chip area select signals are externally available on output pins. Even so, enabling chip area select signals other than CS0, CS3 or CS4, the setting for the corresponding memory blocks will be effective too, regardless of an external chip select output pin.

Figure 4-2: Chip Area Select Control Registers 0, 1 (1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
CSC0	CS33	CS32	CS31	CS30	CS23	CS22	CS21	CS20	CS13	CS12	CS11	CS10	CS03	CS02	CS01	CS00	3FFFFF060H	2C11H
	$\overline{CS3}$				$\overline{CS2}$				$\overline{CS1}$				$\overline{CS0}$					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
CSC1	CS43	CS42	CS41	CS40	CS53	CS52	CS51	CS50	CS63	CS62	CS61	CS60	CS73	CS72	CS71	CS70	3FFFFF062H	2C11H
	$\overline{CS4}$				$\overline{CS5}$				$\overline{CS6}$				$\overline{CS7}$					

Figure 4-2: Chip Area Select Control Registers 0, 1 (2/2)

Bit Position	Bit Name	Function
15 to 0	CSn0 to CSn3 (n = 0 to 7)	Chip Select Enables chip select.

CSnm	CS Operation
CS00	$\overline{CS0}$ active during block 0 access
CS01	$\overline{CS0}$ active during block 1 access.
CS02	$\overline{CS0}$ active during block 2 access.
CS03	$\overline{CS0}$ active during block 3 access.
CS10	$\overline{CS1}$ active during block 0 or 1 access.
CS11	$\overline{CS1}$ active during block 2 or 3 access.
CS12	$\overline{CS1}$ active during block 4 access.
CS13	$\overline{CS1}$ active during block 5 access.
CS20	$\overline{CS2}$ active during block 0 access.
CS21	$\overline{CS2}$ active during block 1 access.
CS22	$\overline{CS2}$ active during block 2 access.
CS23	$\overline{CS2}$ active during block 3 access.
CS30	$\overline{CS3}$ active during block 0, 1, 2, or 3 access.
CS31	$\overline{CS3}$ active during block 4 or 5 access.
CS32	$\overline{CS3}$ active during block 6 access.
CS33	$\overline{CS3}$ active during block 7 access.
CS40	$\overline{CS4}$ active during block 12, 13, 14, or 15 access.
CS41	$\overline{CS4}$ active during block 10 or 11 access.
CS42	$\overline{CS4}$ active during block 9 access.
CS43	$\overline{CS4}$ active during block 8 access.
CS50	$\overline{CS5}$ active during block 15 access.
CS51	$\overline{CS5}$ active during block 14 access.
CS52	$\overline{CS5}$ active during block 13 access.
CS53	$\overline{CS5}$ active during block 12 access.
CS60	$\overline{CS6}$ active during block 14 or 15 access.
CS61	$\overline{CS6}$ active during block 12 or 13 access.
CS62	CS6 active during block 11 access.
CS63	$\overline{CS6}$ active during block 10 access.
CS70	$\overline{CS7}$ active during block 15 access.
CS71	$\overline{CS7}$ active during block 14 access.
CS72	$\overline{CS7}$ active during block 13 access.
CS73	$\overline{CS7}$ active during block 12 access.

4.4 Bus Cycle Type Control Function

In the μPD70F3178 CarGate+, the following external devices can be connected directly to each memory block.

- SRAM, external ROM, external I/O
- Page ROM

Connected external devices are specified by the bus cycle type configuration registers 0, 1 (BCT0, BCT1).

4.4.1 Bus cycle type configuration

(1) Bus cycle configuration registers 0, 1(BCT0, BCT1)

These registers can be read/written in 16-bit units

Figure 4-3: Bus cycle configuration registers 0, 1 (BCT0, BCT1) Format (1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BCT0	ME3	0	0	BT30	ME2	0	0	BT20	ME1	0	0	BT10	ME0	0	0	BT00	FFFFF480H	8888H
	$\overline{CS3}$				$\overline{CS2}$				$\overline{CS1}$				$\overline{CS0}$					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BCT1	ME7	0	0	BT70	ME6	0	0	BT60	ME5	0	0	BT50	ME4	0	0	BT40	FFFFF482H	8888H
	$\overline{CS7}$				$\overline{CS6}$				$\overline{CS5}$				$\overline{CS4}$					

Figure 4-3: Bus cycle configuration registers 0, 1 (BCT0, BCT1) Format (2/2)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>15, 11, 7, 3
(BCT0)
15, 11, 7, 3
(BCT1)</td><td>MEn
(n = 0 to 7)</td><td>Memory Controller Enable
Sets memory controller operation enable for each chip select signal $\overline{CSn}$.
<table>
<tr><th>ME</th><th>Memory Controller Operation Enable</th></tr>
<tr><td>0</td><td>Operation disable</td></tr>
<tr><td>1</td><td>Operation enable</td></tr>
</table></td></tr>
<tr><td>12, 8, 4, 0
(BCT0),
(BCT1)</td><td>BTn0
(n = 0 to 7)</td><td>Bus Cycle Type
Specifies the device to be connected to the $\overline{CSn}$ signal...
<table>
<tr><th>BTn0</th><th>External Device Connected Directly to $\overline{CSn}$ signal</th></tr>
<tr><td>0</td><td>SRAM, external I/O</td></tr>
<tr><td>1</td><td>Page ROM</td></tr>
</table></td></tr>
</table>

Cautions: 1. Write to the BCT0 and BCT1 registers after reset, and then do not change the set value. Also, do not access an external memory area other than that for this initialization routine until initial setting of the BCT0 and BCT1 registers is finished. However, it is possible to access external memory areas whose initialization has been finished.

2. The bits marked as 0 are reserved. They have to leave to 0.

4.5 Bus Access

4.5.1 Number of access clocks

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

Table 4-1: Number of Bus Access Clocks

Resources (Bus width) Bus Cycle Configuration		Internal RAM (32 bits)	Peripheral I/O (16 bits)	External memory (16 bits)
Instruction fetch	Normal access	1 Note 1	-	2 Note 2
	Branch	1	-	2 Note 2
Operand data access		1	3 Note 2	2 Note 2

Notes: 1. The instruction fetch becomes 2 clocks, in case of contention with data access.

2. This is the minimum value.

4.5.2 Bus sizing function

The bus sizing function controls data bus width for each CS area. The data bus width is specified by using the bus size configuration register (BSC).

(1) Bus size configuration register (BSC)

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

Figure 4-4: Bus Size Configuration Register (BSC) Format

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BSC	0	BS70	0	BS60	0	BS50	0	BS40	0	BS30	0	BS20	0	BS10	0	BS00	FFFFF066H	5555H
	$\overline{CS7}$		$\overline{CS6}$		$\overline{CS5}$		$\overline{CS4}$		$\overline{CS3}$		$\overline{CS2}$		$\overline{CS1}$		$\overline{CS0}$			

Bit Position	Bit Name	Function
15 to 0	BSn1, BSn0 (n = 0 to 7)	Data Bus Width Sets the data bus width of CSn area. BSn0 — Data Bus Width of CSn area 0 — 8 bits 1 — 16 bits

Cautions: 1. Write to the BSC register after reset, and then do not change the set value. Also, do not access an external memory area other than that for this initialization routine until initial setting of the BSC register is finished. However, it is possible to access external memory areas whose initialization has been finished.

2. When the data bus width is specified as 8 bits, only the $\overline{WR0}$ signal becomes active.

4.5.3 Endian control function

The endian control function can be used to set processing of word data in memory either by the Big Endian method or the Little Endian method for each CS area selected with the chip select signal ($\overline{CS0}$ to $\overline{CS7}$). Switching of the endian method is specified with the endian configuration register (BEC).

(1) Endian configuration register (BEC)

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

Figure 4-5: Endian configuration register (BEC) Format

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BEC	0	BE70	0	BE60	0	BE50	0	BE40	0	BE30	0	BE20	0	BE10	0	BE00	FFFFF068H	0000H
	$\overline{CS7}$		$\overline{CS6}$		$\overline{CS5}$		$\overline{CS4}$		$\overline{CS3}$		$\overline{CS2}$		$\overline{CS1}$		$\overline{CS0}$			

Bit Position	Bit Name	Function
14, 12, 10, 8, 6, 4, 2, 0	BEn0 (n = 0 to 7)	Big Endian Specifies the endian method. BEn0 — Endian Control 0 — Little Endian method 1 — Big Endian method

Cautions: 1. Bits 15, 13, 11, 9, 7, 5, 3, and 1 of the BEC register must be cleared (0). If these bits are set to 1, the operation is not guaranteed.

2. Set the CSn area specified as the programmable peripheral I/O area to Little Endian format (n = 0 to 7).

3. In the following areas, the data processing method is fixed to Little Endian method. Any setting of Big Endian method for these areas according to the BEC register is invalid.
- On-chip peripheral I/O area
- Internal RAM area
- Fetch area of external memory

Figure 4-6: Big Endian Addresses within Word

31 – 24	23 – 16	17 – 8	7 – 0
0008H	0009H	000AH	000BH
0004H	0005H	0006H	0007H
0000H	0001H	0002H	0003H

Figure 4-7: Little Endian Addresses within Word

31 – 24	23 – 16	17 – 8	7 – 0
000BH	000AH	0009H	0008H
0007H	0006H	0005H	0004H
0003H	0002H	0001H	0000H

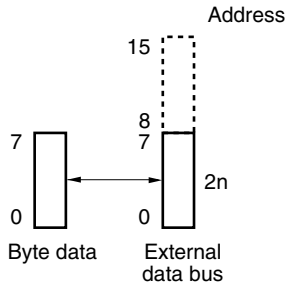
4.5.4 Bus width

The μPD70F3178 CarGate+ accesses peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The following shows the operation for each type of access. Access all data in order starting from the lower order side.

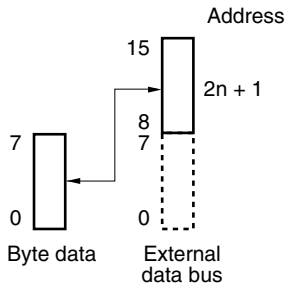
(1) Byte access (8 bits)

(a) When the data bus width is 16 bits (Little Endian)

<1> Access to even address (2n)

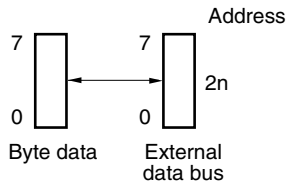


<2> Access to odd address (2n + 1)

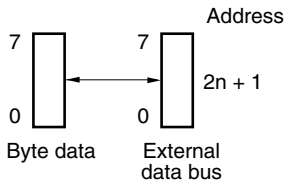


(b) When the data bus width is 8 bits (Little Endian)

<1> Access to even address (2n)

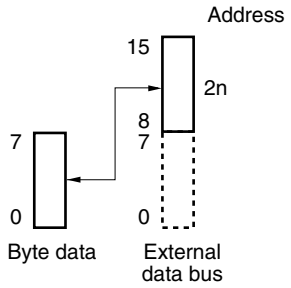


<2> Access to odd address (2n + 1)

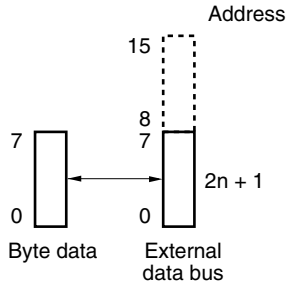


(c) When the data bus width is 16 bits (Big Endian)

<1> Access to even address (2n)

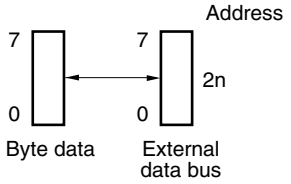


<2> Access to odd address (2n + 1)

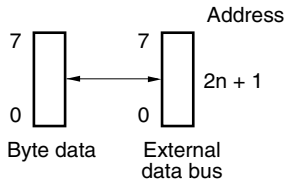


(d) When the data bus width is 8 bits (Big Endian)

<1> Access to even address (2n)



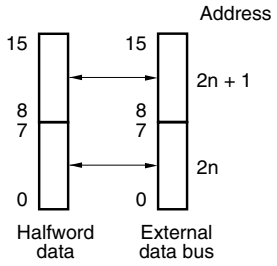
<2> Access to odd address (2n + 1)



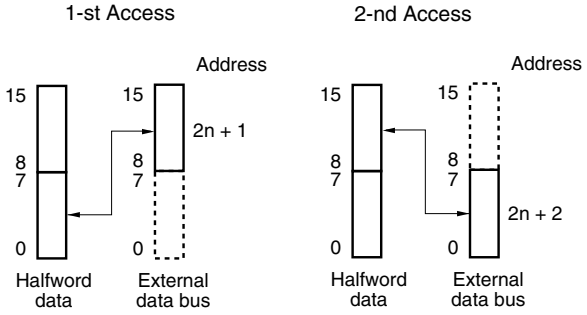
(2) Halfword access (16 bits)

(a) When the bus width is 16 bits (Little Endian)

<1> Access to even address (2n)

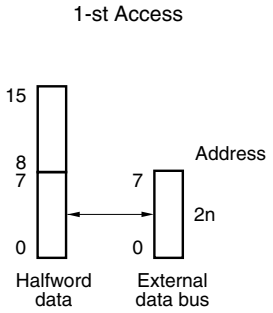


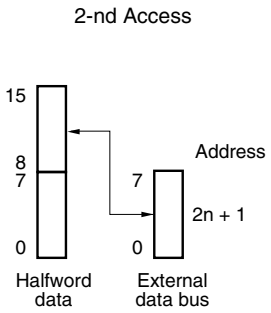
<2> Access to odd address (2n + 1)



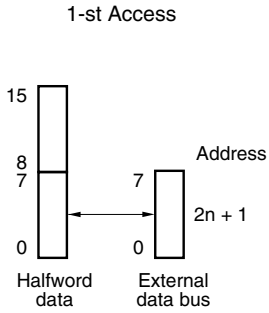
(b) When the data bus width is 8 bits (Little Endian)

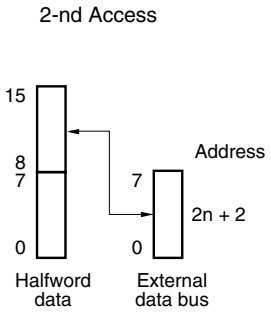
<1> Access to even address (2n)





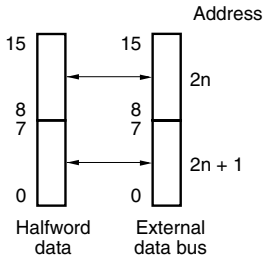
<2> Access to odd address (2n + 1)



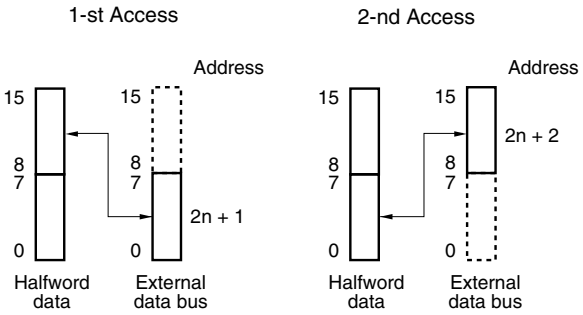


(c) When the data bus width is 16 bits (Big Endian)

<1> Access to even address (2n)

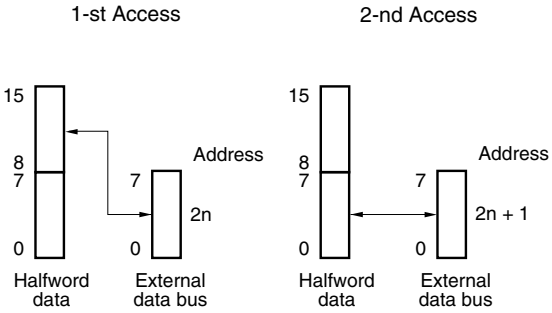


<2> Access to odd address (2n + 1)



(d) When the data bus width is 8 bits (Big Endian)

<1> Access to even address (2n)

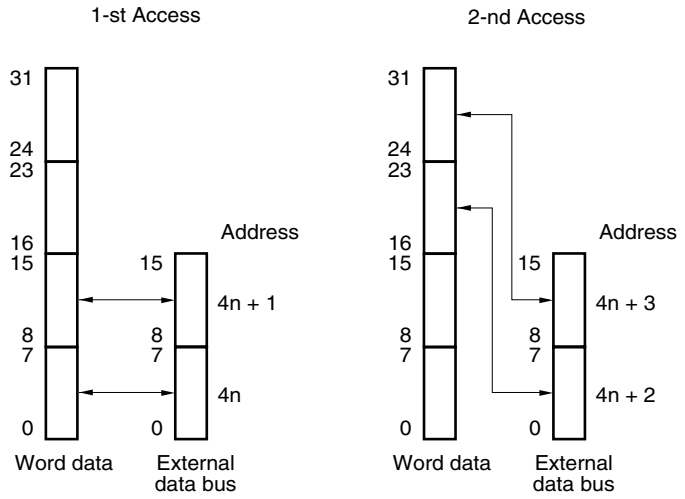


<2> Access to odd address (2n + 1)

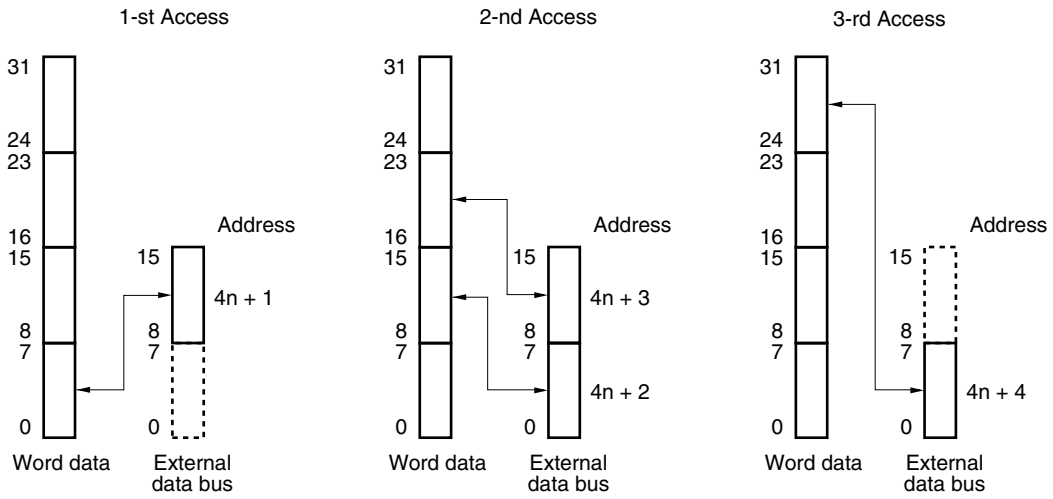
(3) Word access (32 bits)

(a) When the bus width is 16 bits (Little Endian)

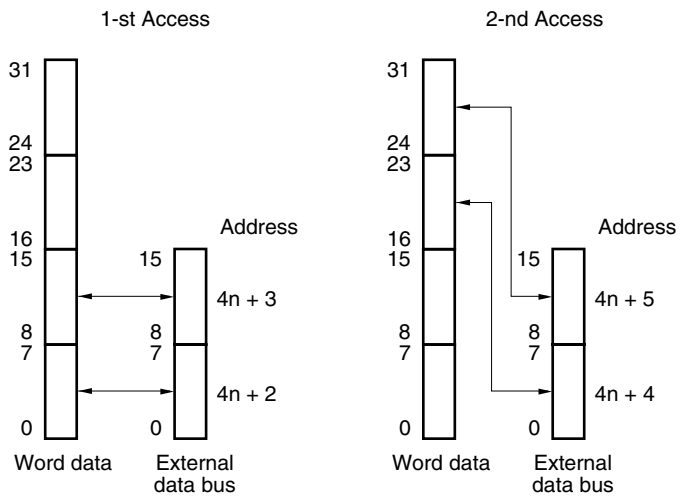
<1> Access to address 4n



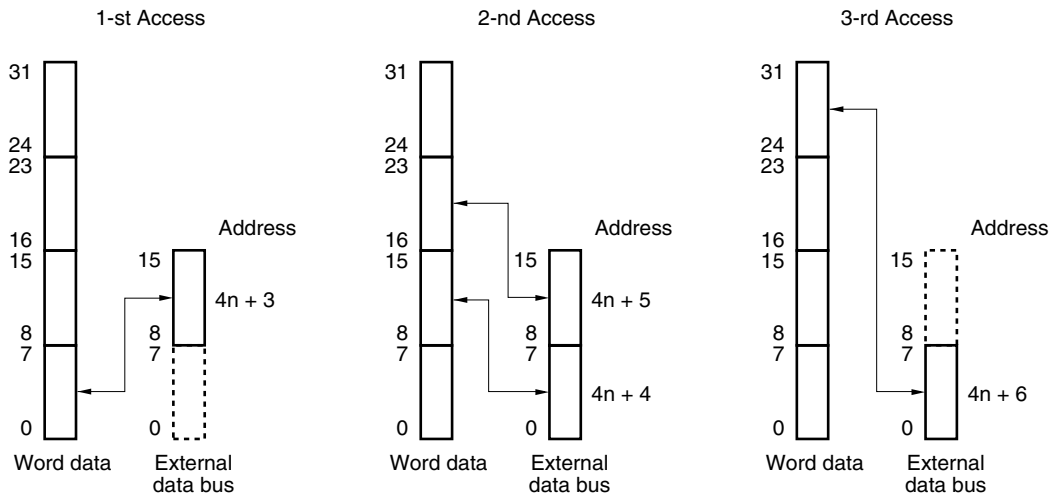
<2> Access to address 4n + 1



<3> Access to address 4n + 2

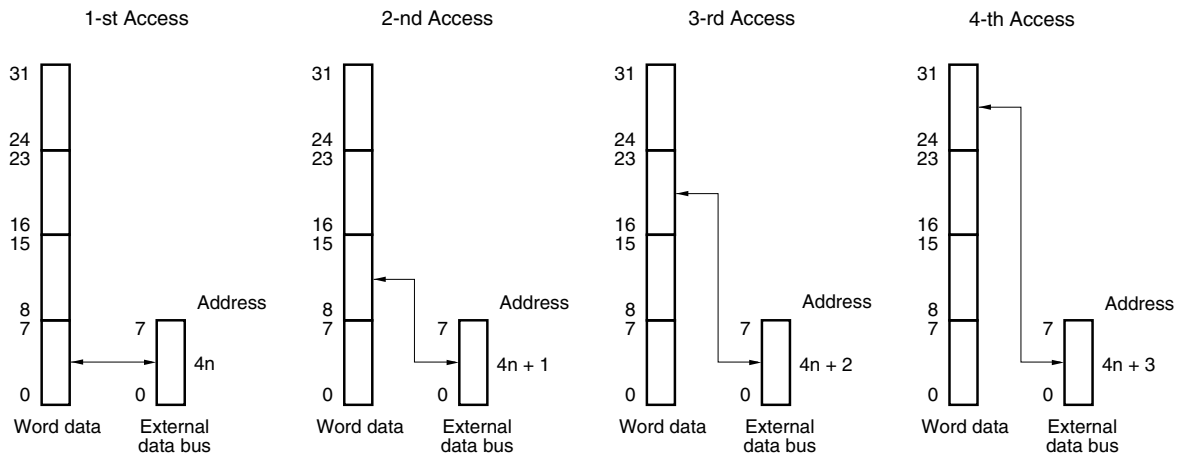


<4> Access to address 4n + 3

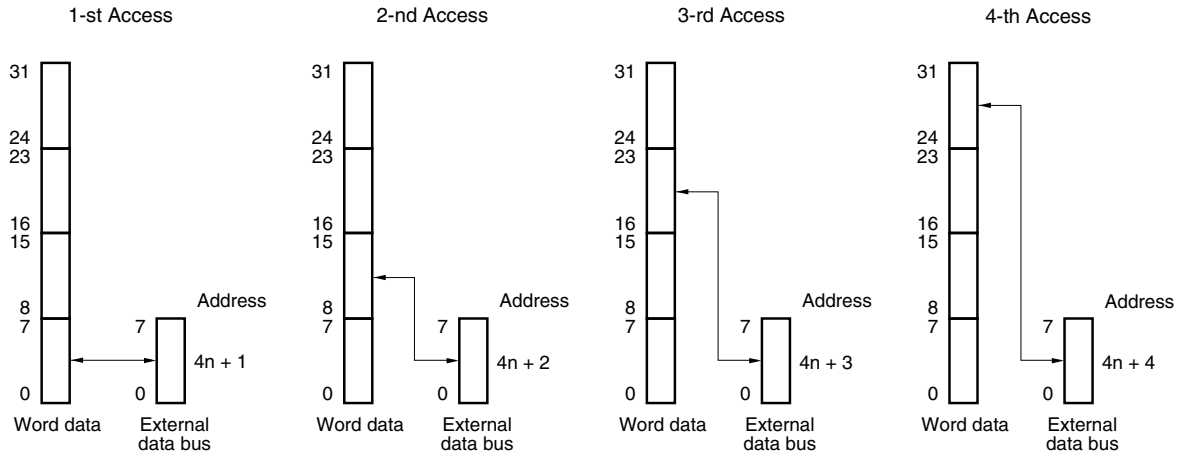


(b) When the bus width is 8 bits (Little Endian)

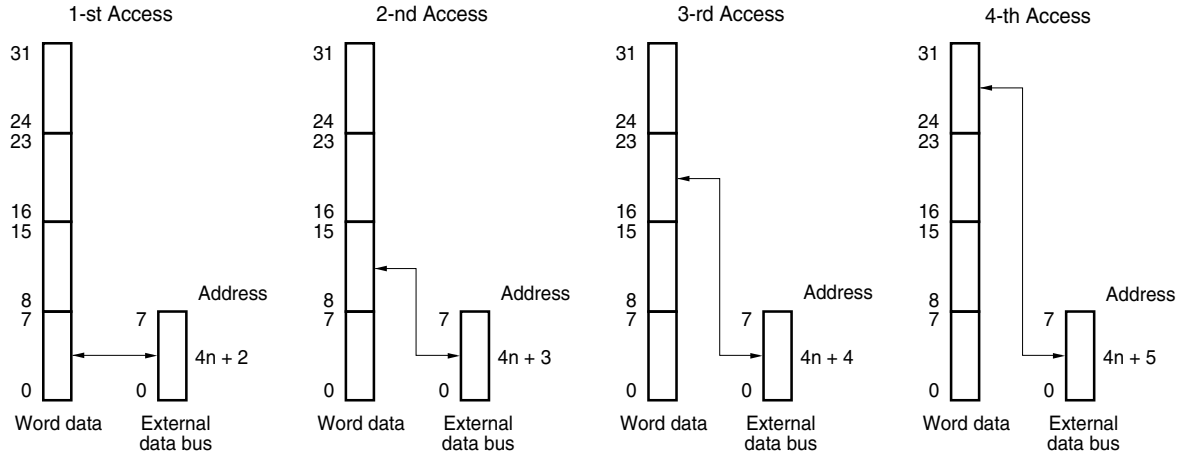
<1> Access to address 4n



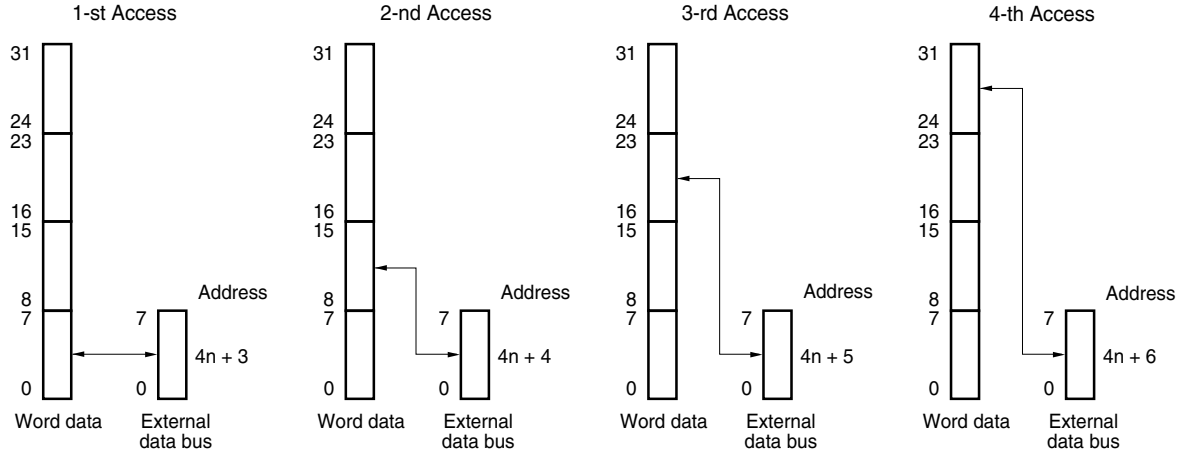
<2> Access to address 4n + 1



<3> Access to address 4n + 2

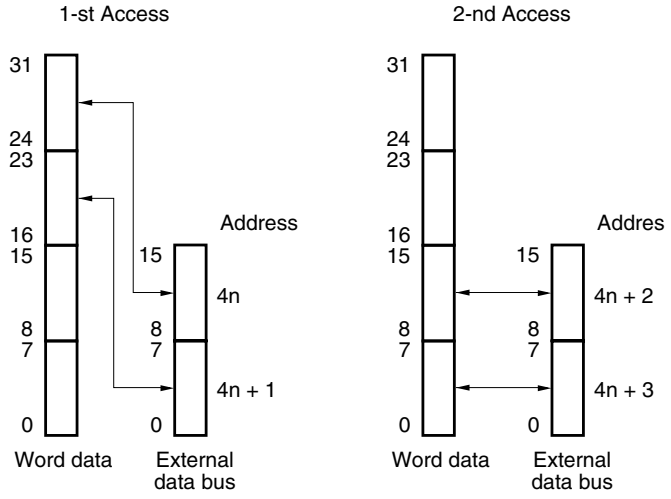


<4> Access to address 4n + 3

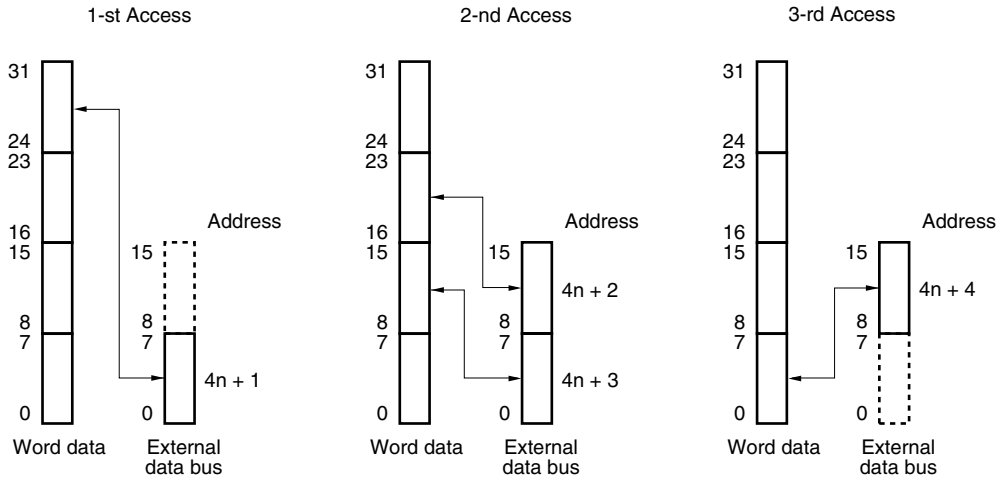


(c) When the data bus width is 16 bits (Big Endian)

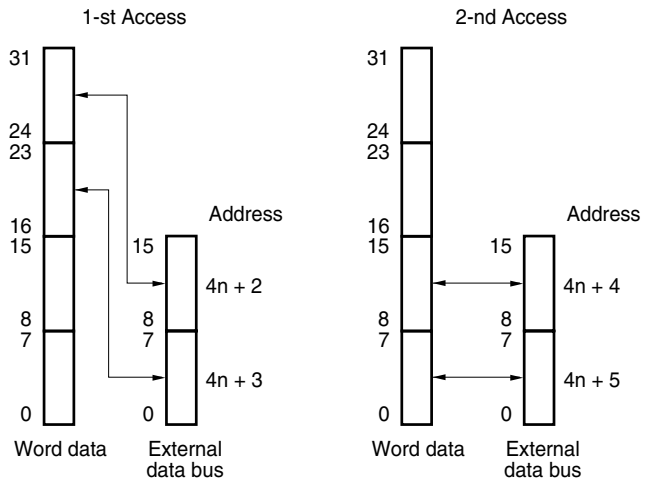
<1> Access to address 4n



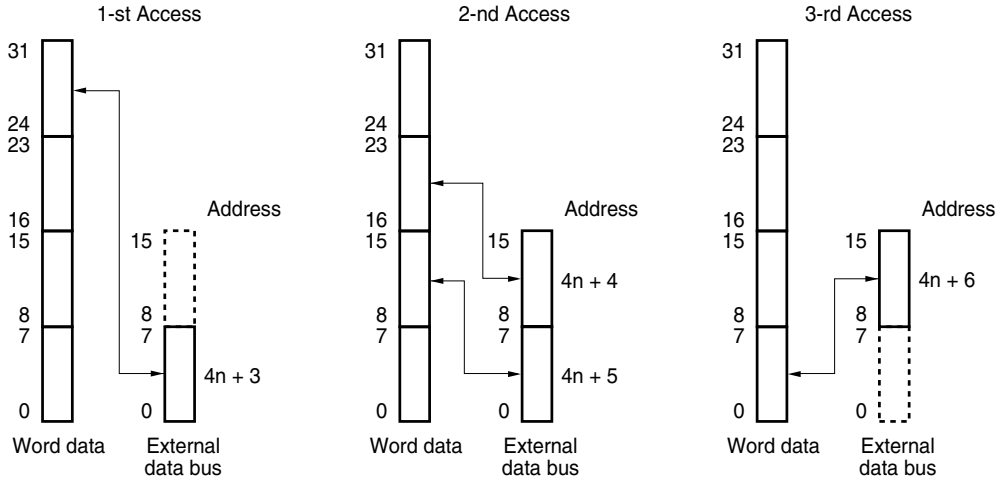
<2> Access to address 4n + 1



<3> Access to address 4n + 2

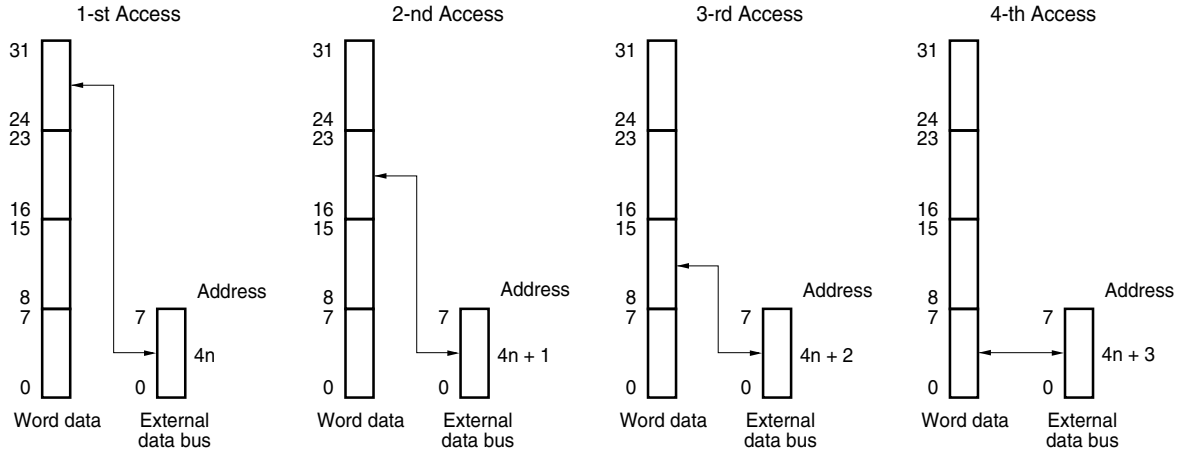


<4> Access to address 4n + 3

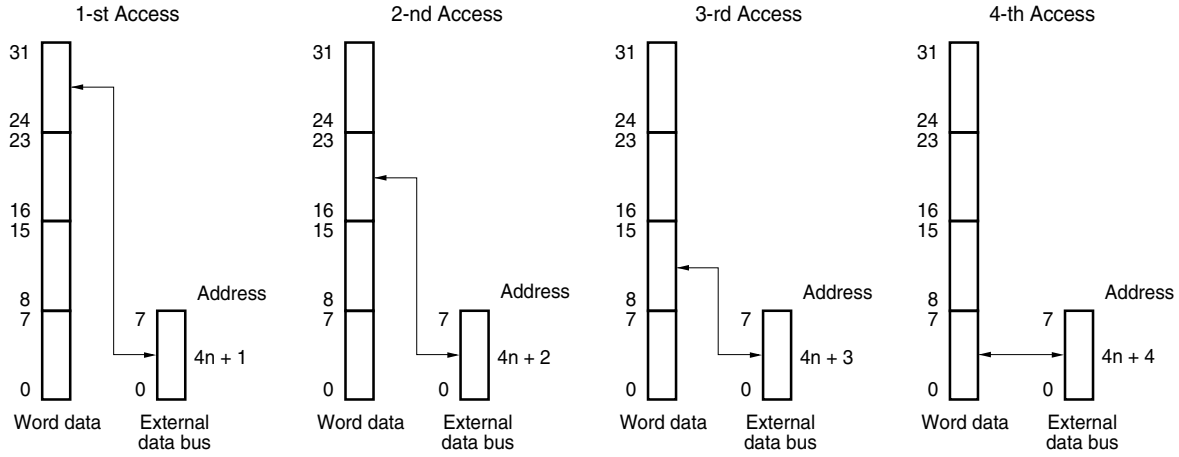


(d) When the data bus width is 8 bits (Big Endian)

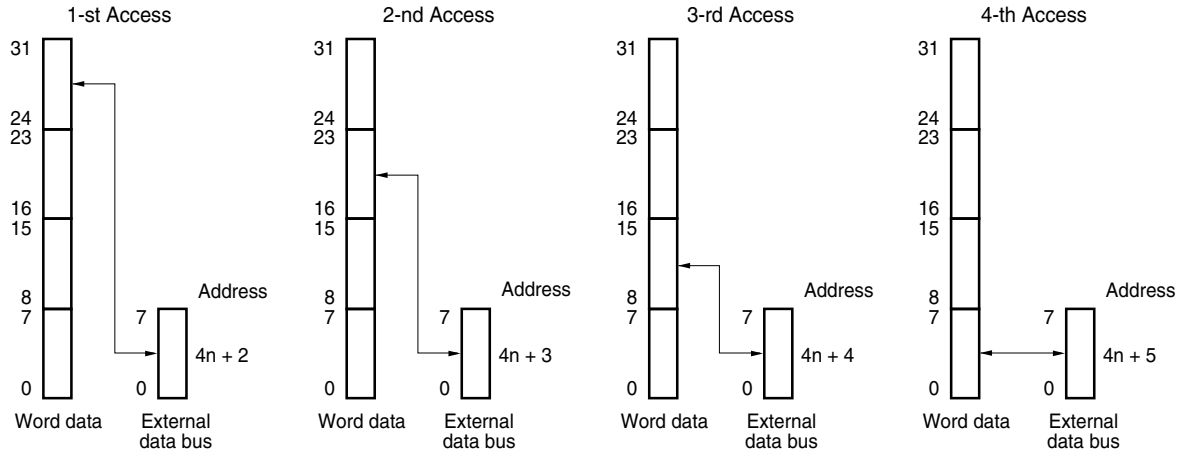
<1> Access to address 4n



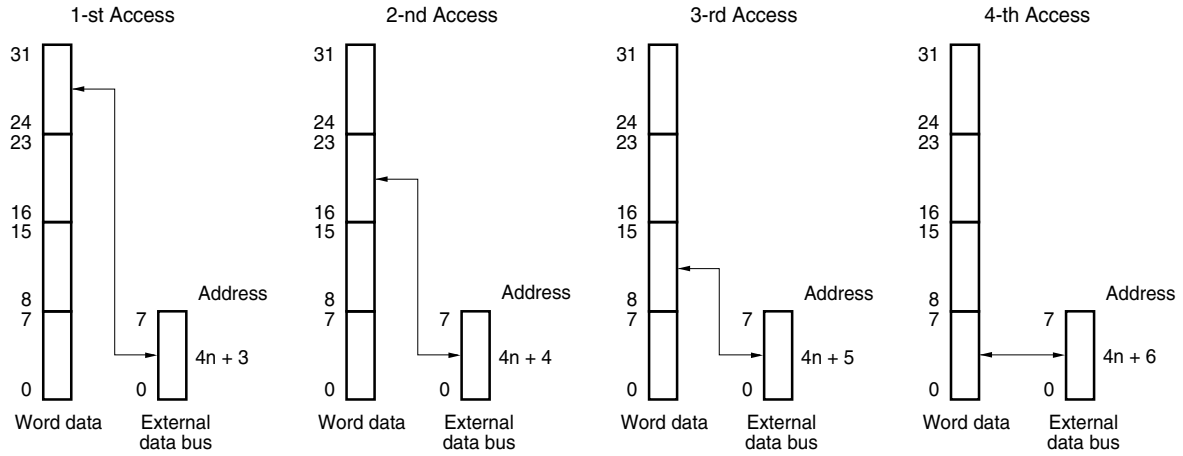
<2> Access to address 4n + 1



<3> Access to address 4n + 2



<4> Access to address 4n + 3



4.6 Wait Function

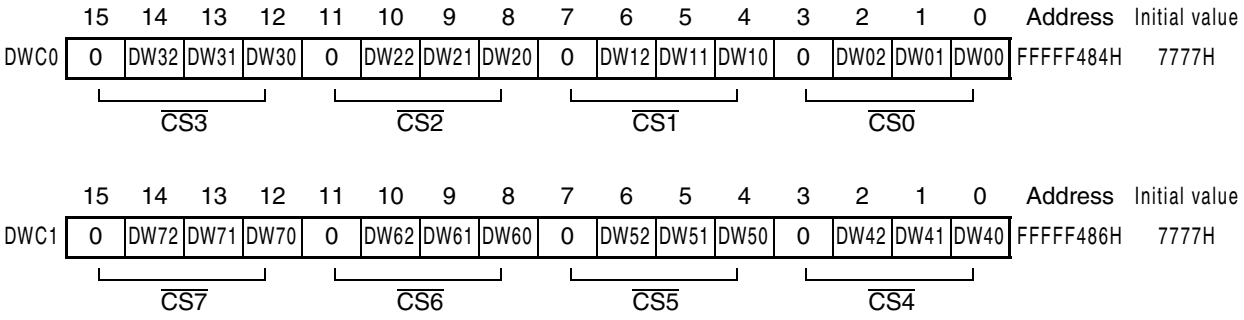
4.6.1 Programmable wait function

(1) Data wait control registers 0, 1 (DWC0, DWC1)

With the purpose of realizing easy interfacing with low-speed memory or with I/Os, it is possible to insert up to 7 data wait states with respect to the starting bus cycle for each CS area.
The number of wait states can be specified by data wait control registers 0 and 1 (DWC0, DWC1) in programming. Just after system reset, all blocks have 7 data wait states inserted.

These registers can be read/written in 16-bit units.

Figure 4-8: Data wait control registers 0, 1 (DWC0, DWC1) Format



Bit Position	Bit Name	Function
14 to 12, 10 to 8, 6 to 4, 2 to 0	DWn2 to DWn0 (n = 0 to 7)	Data Wait Specifies the number of wait states inserted in the $\overline{CSn}$ area.

DWn2	DWn1	DWn0	Number of Wait States Inserted in $\overline{CSn}$ Space
0	0	0	No wait states inserted
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

Cautions: 1. The internal Cache and the internal RAM area are not subject to programmable waits and ordinarily no wait access is carried out. The internal peripheral I/O area is also not subject to programmable wait states, with wait control performed only by each peripheral function.

2. In the following cases, the settings of registers DWC0 and DWC1 are invalid (wait control is performed by each memory controller).
- Page ROM on-page access

3. Write to the DWC0 and DWC1 registers after reset, and then do not change the set values. Also, do not access an external memory area other than that for this initialization routine until initial setting of the DWC0 and DWC1 registers is finished. However, it is possible to access external memory areas whose initialization has been finished.

(2) Address setup wait control register (ASC)

The μPD70F3178 CarGate+ allows insertion of address setup wait states before the T1 cycle of the SRAM or page ROM cycle.
The number of address setup wait states can be set with the ASC register for each CS area.

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

Figure 4-9: Address setup wait control register (ASC) Format

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
ASC	AC71	AC70	AC61	AC60	AC51	AC50	AC41	AC40	AC31	AC30	AC21	AC20	AC11	AC10	AC01	AC00	FFFFF48AH	FFFFH
	$\overline{CS7}$		$\overline{CS6}$		$\overline{CS5}$		$\overline{CS4}$		$\overline{CS3}$		$\overline{CS2}$		$\overline{CS1}$		$\overline{CS0}$			

Bit Position	Bit Name	Function
15 to 0	ACn1, ACn0 (n = 0 to 7)	Address Cycle Specifies the number of address setup wait states inserted before the T1 cycle of SRAM/page ROM cycle for each CS area.

ACn1	ACn0	Number of Wait States
0	0	Not inserted
0	1	1
1	0	2
1	1	3

4.7 Idle State Insertion Function

To facilitate interfacing with low-speed memory devices, an idle state (TI) can be inserted into the current bus cycle after the T2 state to meet the data output float delay time (tdf) on memory read access for each CS space. The bus cycle following the T2 state starts after the idle state is inserted.

An idle state is inserted after read/write cycles for SRAM, external I/O, or external ROM.

In the following cases, an idle state is inserted in the timing.

- after read/write cycles for SRAM, external I/O, or external ROM

The idle state insertion setting can be specified by program using the bus cycle control register (BCC). Immediately after the system reset, idle state insertion is automatically programmed for all memory blocks.

(1) Bus cycle control register (BCC)

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

Figure 4-10: Bus cycle control register (BCC) Format

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
BCC	BC71	BC70	BC61	BC60	BC51	BC50	BC41	BC40	BC31	BC30	BC21	BC20	BC11	BC10	BC01	BC00	FFFFF488H	FFFFH
	$\overline{CS7}$		$\overline{CS6}$		$\overline{CS5}$		$\overline{CS4}$		$\overline{CS3}$		$\overline{CS2}$		$\overline{CS1}$		$\overline{CS0}$			

Bit Position	Bit Name	Function
15 to 0	BCn1, BCn0 (n = 0 to 7)	Data Cycle Specifies the insertion of an idle state when accessing corresponding $\overline{CSn}$ area.

BCn1	BCn0	Idle State in $\overline{CSn}$ Area
0	0	Not inserted
0	1	1
1	0	2
1	1	3

Cautions: 1. The internal iCache area, the internal RAM area and the internal peripheral I/O area are not subject to insertion of an idle state.

2. Write to the BCC register after reset, and then do not change the set value. Also, do not access an external memory area other than that for this initialization routine until initial setting of the BCC register is finished. However, it is possible to access external memory areas whose initialization has been finished.

4.8 Bus Priority Order

There are two external bus cycles: operand data access and instruction fetch.
As for the priority order, the highest priority has the instruction fetch than operand data access.
An instruction fetch may be inserted between read access and write access during read modify write access.
Also, an instruction fetch may be inserted between bus access and bus access during CPU bus clock.

Table 4-2: Bus Priority Order

Priority Order	External Bus Cycle	Bus Master
Low	Operand data access	CPU
↕	Instruction fetch	CPU
High		

4.9 Boundary Operation Conditions

4.9.1 Program space

(1) Branching to the peripheral I/O area or successive fetch from the internal RAM area to the internal peripheral I/O area is inhibited. In terms of hardware, fetching the NOP opcode continues, and fetching from the external memory is not performed.

(2) If a branch instruction exists at the upper limit of the internal RAM area, a pre-fetch operation (invalid fetch) that straddles over the internal peripheral I/O area does not occur when instruction fetch is performed.

4.9.2 Data space

The μPD70F3178 CarGate+ is provided with an address misalign function.
Through this function, regardless of the data format (word data, halfword data, or byte data), data can be placed in all addresses. However, in the case of word data and halfword data, if data are not subjected to boundary alignment, the bus cycle will be generated a minimum of 2 times and bus efficiency will drop.

(1) In the case of halfword length data access

When the address's LSB bit is 1, the byte length bus cycle will be generated 2 times.

(2) In the case of word length data access

(a) When the address's LSB is 1, bus cycles will be generated in the order of byte length bus cycle, halfword length bus cycle, and word length bus cycle.

(b) When the address's lowest 2 bits are 10, the halfword length bus cycle will be generated 2 times.

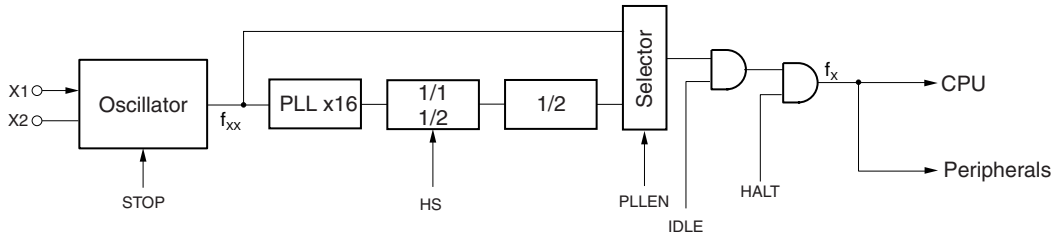
Chapter 5 Clock Generator

5.1 Features

- Multiplication function by PLL synthesizer
- Clock sources
  - Oscillation through oscillator connection
- Power save modes
  - HALT mode
  - IDLE mode
  - STOP mode

5.2 Configuration

Figure 5-1: Block Diagram of the Clock Generator



This block diagram does not necessarily show the exact wiring in hardware but the functional structure.

5.3 Control Registers

5.3.1 Clock control register (CKC)

This is an 8-bit register that controls the clock management. Data can be written to it only in a sequence of specific instructions so that its contents are not easily rewritten in case of program hang-up. See also PHCMD register.

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

Figure 5-2: Clock Control Register (CKC)

	7	6	5	4	3	2	1	0	Address	Initial value
CKC	PLLEN	0	0	0	0	0	0	0	FFFFF822H	00H
R/W	R/W	R	R	R	R	R	R	R		

Bit name	Function
PLLEN	PLLEN enable bit This bit enables or disables PLL operation. 0: PLL disabled 1: PLL enabled

To write data to the CKC register, use the store instruction (ST/SST) and bit manipulation instruction (SET1/CLR1/NOT1).
The contents of this register can be read in the normal sequence.

5.3.2 PLL status register (PSTAT)

This is an 8-bit register that monitors status of the PLL. Reading VBSTAT shows, if system clock is switched to PLL.

This register can be read in 8-bit units.

Figure 5-3: PLL Status Register (PSTAT)

	7	6	5	4	3	2	1	0	Address	Initial value
PSTAT	0	0	0	0	0	0	0	VBSTAT	FFFFF824H	01H
R/W	R	R	R	R	R	R	R	R		

Bit name	Function
VBSTAT	System Clock Status 0: System clock = PLL output 1: System clock = OSC output

5.3.3 PLL control register (PLC)

This is a 8-bit register that controls the PLL multiplication factor.

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

Figure 5-4: PLL Control Register (PLC)

	7	6	5	4	3	2	1	0	Address	Initial value
PLC	0	0	0	0	0	0	0	HS	FFFFF840H	00H
R/W	R	R	R	R	R	R	R	R/W		

Bit name	Function
HS	High Speed flag 0: PLL × 8 => $f_{CPU} = f_{OSC} \times {}^{8}/_{2}$ => f_{CPU}= 24 MHz with 6 MHz external 1: PLL × 16 => $f_{CPU} = f_{OSC} \times {}^{16}/_{2}$ => f_{CPU}= 32 MHz with 4 MHz external

Caution: HS must be set before PLLEN

5.4 Power Saving Functions

5.4.1 General

The device provides the following power saving functions. These modes can be combined and switched to suit the target application, which enables effective implementation of low-power systems.

The device provides the following power saving functions. These modes can be combined and switched to suit the target application, which enables effective implementation of low-power systems.

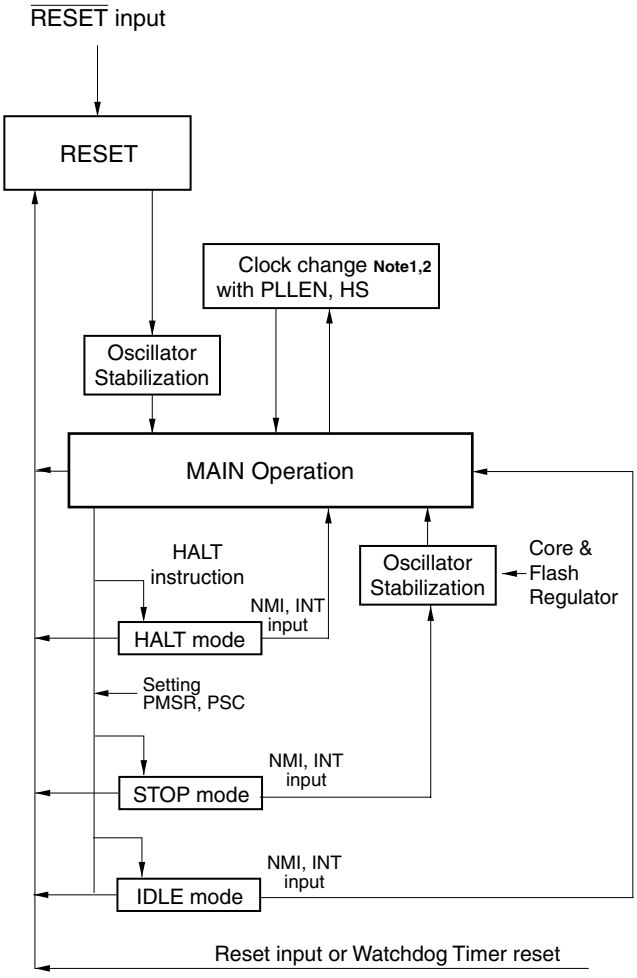
Table 5-1: Power Saving Modes Overview

Clock Source		Mode	Operation of		Clock Supply to	
			Oscillator	PLL	Peripherals	CPU
			Main Osc.			
OSC mode	Initial Mode	Normal	×	–	×	×
	PLL enabled	Normal	×	×	×	×
		HALT	×	×	×	–
		IDLE	–	×	–	–
		STOP	–	–	–	–

Remark: ×: Operates
–: Stopped

Figure 5-5 shows the operation of the clock generator in normal operation mode, HALT mode, IDLE mode and software STOP mode.
An effective low power consumption system can be realized by combining these modes and switching modes according to the required use.

Figure 5-5: Power Save Mode State Transition Diagram



Notes: 1. The PLL is deactivated per hardware.

2. Enable PLL manual.

5.4.2 Power save modes outline

µPD70F3178 CarGate+ is provided with the following standby modes: HALT, IDLE and software STOP. Application systems, which are designed so that these modes are switched appropriately according to operation purposes, reduce power consumption efficiently.

(1) HALT mode:

In this mode supply of the operating clock to the CPU is stopped whereby other on-chip peripheral functions continue to operate. Combining this mode with the normal operating mode to provide intermittent operations enables the overall system power consumption to be reduced.
This mode is entered by executing the dedicated instruction (HALT).

(2) IDLE mode:

In this mode, the clock generator continues to operate but stopping the supply of internal system clock stops the overall system. As it is not necessary to secure the oscillation stabilization time, it is possible to switch to the normal operating mode quickly in response to a release signal.
This mode provides low power consumption, where the power is only consumed from the OSC.
This mode is entered by setting registers with software.

(3) Software STOP mode:

In this mode, the main clock generator is stopped and the entire system stops. This mode provides ultra-low power consumption, where the power consumed is only leakage current.
This mode is entered by setting registers with software.

5.4.3 HALT mode

In this mode, the CPU clock is stopped, though the clock generators (oscillator and PLL synthesizer) continue to operate for supplying clock signals to other peripheral function circuits.
Setting the HALT mode when the CPU is idle reduces the total system power consumption.
In the HALT mode, program execution is stopped but the contents of all registers and internal RAM prior are retained as is.
On-chip peripheral hardware irrelevant to the CPU instruction execution also continues to operate. The state of the various hardware units in the HALT mode is tabulated below.

Table 5-2: Operating states in HALT mode

Items	Operation
Clock generator	Operating
PLL	Operating
Internal system clock	Operating
CPU	Stopped (but CPU clock still operates)
I/O line	Unchanged
Peripheral function	Operating
Internal data	Retains all internal data before entering HALT mode, such as CPU registers, status, data, and on-chip RAM.

Remark: Even after the HALT instruction is executed, instruction fetch operations continue until the internal instruction pre-fetch queue is full. After the queue becomes full, the CPU stops with the items set as tabulated above.

HALT mode release:

The HALT mode can be released by a non-maskable interrupt request, an unmasked maskable interrupt request, or $\overline{\text{RESET}}$ signal input.

(1) Release by interrupt request

The HALT mode is released unconditionally by an unmasked maskable interrupt request regardless of its priority level. However, if the HALT mode is entered during execution of an interrupt handler, the operation differs on interrupt priority levels as follows:

(a) If an interrupt request less prioritized than the currently serviced interrupt request is generated, the HALT mode is released but the interrupt is not acknowledged. The interrupt request itself is retained.
(b) If an interrupt request (including a non-maskable one) prioritized than the currently serviced interrupt request is generated, the interrupt request is acknowledged along with the HALT mode release.

Table 5-3: Operation after HALT mode release by interrupt request

Release cause	EI state	DI state
NMI request	Branches to handler address.	
Maskable interrupt request	Branches to handler address, or executes the next instruction.	Executes the next instruction.

Remark: If HALT mode is entered during execution of a particular interrupt handler and an unmasked interrupt request with a higher priority than the previous one is subsequently generated, the program branches to the vector address for the latter interrupt.

(2) Release by $\overline{\text{RESET}}$ pin input

This operation is the same as normal reset operation.

5.4.4 IDLE mode

In this mode, the CPU clock is stopped resulting in stop of the entire system, though the clock generators (oscillator and PLL synthesizer) continue to operate.
As it is not necessary to secure the oscillator oscillation stabilization time and the PLL lock-up time, it is possible to quickly switch to the normal operating mode in response to a release cause. The IDLE mode can be entered by configuration the PSM and PSC registers.
In the IDLE mode, program execution is stopped but the contents of all registers and internal RAM prior to entering this mode are retained. On-chip peripheral hardware operation is also stopped.

The state of the various hardware units in the IDLE mode is tabulated below.

Table 5-4: Operating States in IDLE Mode

Items	Operation
Clock generator	Operating
PLL	Operating
Internal system clock	Stopped
CPU	Stopped
I/O line	Unchanged
Peripheral function	Stops
Internal data	Retains all internal data before entering IDLE mode, such as CPU registers, status, data, and on-chip RAM.

IDLE mode release:

Release operation is same as release from HALT mode.
The IDLE mode is released by NMI, $\overline{\text{RESET}}$ signal input, or an unmasked maskable interrupt request.

(a) Release by Interrupt input:

When the IDLE mode is released, the NMI request is acknowledged.
If the IDLE mode is entered during the execution of NMI handler, the IDLE mode is released but the interrupt is not acknowledged. The interrupt itself is retained.

(b) Release by $\overline{\text{RESET}}$ input:

This operation is the same as normal reset operation.

5.4.5 Software STOP mode

In this mode, the CPU clock is stopped including the clock generators (oscillator and PLL synthesizer), resulting in stop of the entire system for ultra-low power consumption (the only consumed is device leakage current).
When this mode is released, the oscillation stabilization time for the oscillator should be secured until the system clock is stabilized. However, when the external clock operates this product, securing the oscillation stabilization time for the oscillator until the system clock is stabilized is unnecessary. In the direct mode as well, the lock-up time does not have to be secured.
This mode is entered by setting the PSM & PSC register.
In this mode, the program execution stops, but the contents of all registers and internal RAM prior to entering this mode are retained. µPD70F3178 peripherals operations are also stopped

The state of the various hardware units in the software STOP mode is tabulated below.

Table 5-5: Operating States in STOP Mode

Items	Operation
Clock generator	Stopped (Sub OSC operates, if SOSTP bit = “1”)
PLL	Stopped
Internal system clock	Stopped
CPU	Stopped
I/O line**Note**	Unchanged
Peripheral function	Stopped
Internal data**Note**	Retains all previous internal data, such as CPU registers, status, data, and on-chip RAM.

Note: When the V_{DD} value is within the operating range. However, even if V_{DD} falls below the lowest operating voltage, the internal RAM content is retained as long as the data retention voltage V_{DDDR} is maintained.

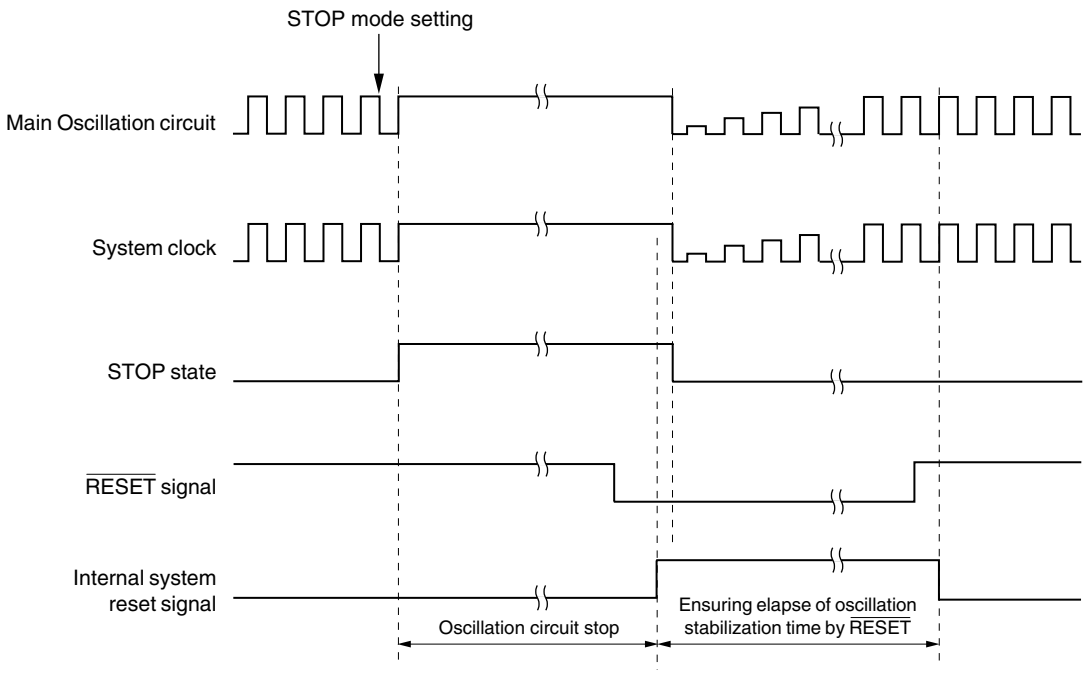
STOP mode release:

The STOP mode can be released by a non-maskable interrupt request, an unmasked maskable interrupt request, or $\overline{\text{RESET}}$ signal input.

(1) When released by $\overline{\text{RESET}}$ input

This operation is the same as normal reset operation. Oscillator stabilization time must be ensured by reset input.

Figure 5-6: STOP mode released by $\overline{\text{RESET}}$ input



5.5 Register Description

5.5.1 Power save control register (PSC)

This is an 8-bit register that controls the power save mode. Data can be written only in a sequence of specific instructions so that its contents are not easily rewritten in case of program hang-up.

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

Figure 5-7: Power Save Control Register (PSC)

	7	6	5	4	3	2	1	0	Address	Initial value
PSC	0[Note]	0	0	0	0	0	STP	0	FFFFF1FEH	00H
R/W	R	R	R	R	R	R	R/W	R		

Bit name	Function
STP	Power save mode specification 0: IDLE, STOP mode are released 1: IDLE, STOP mode are entered

Note: If this bit is set to 1, proper operation can not be guaranteed!

Data is set in the power save control register (PSC) according to the following sequence.

<1> Set the power save mode register (PSM) (with the following instructions).

- Store instruction (ST/SST instruction)
- Bit manipulation instruction (SET1/CLR1/NOT1 instruction)

<2> Prepare data in any one of the general-purpose registers to set to the specific register.
<3> Write arbitrary data to the command register (PRCMD).
<4> Set the power save control register (PSC) (with the following instructions).

- Store instruction (ST/SST instruction)
- Bit manipulation instruction (SET1/CLR1/NOT1 instruction)

<5> Assert the NOP instructions (5 instructions (<5> to <9>).

5.5.2 Power save mode register (PSM)

This is an 8-bit register that control the power save mode and sub-oscillator control.

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

Figure 5-8: Power Save Mode Register (PSM)

	7	6	5	4	3	2	1	0	Address	Initial value
PSM	0	0	0	0	0	0	0	PSM0	FFFFF820H	00H
R/W	R	R	R	R	R	R	R	R/W		

Bit name	Function
PSM0	Standby mode specification bits PSM0 — Standby Mode 0 — IDLE 1 — STOP

Chapter 6 16-Bit Timer/Event Counter P

The V850E/CG3 CarGate+ includes 3 16-bit timer/event counter P.

6.1 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
- One-shot pulse output
- Pulse width measurement function
- Timer synchronized operation function
- Free-running function
- External trigger pulse output function

6.2 Functional Outline

- Capture trigger input signal × 2
- External trigger input signal × 1
- Clock selection × 8
- Readable counter × 1
- Capture/compare reload register × 2
- Capture/compare match interrupt × 2
- Timer output (TOPn0, TOPn1) × 2

Remark: n = 0 to 2

6.3 Configuration

TMP consists of the following hardware.

Table 6-1: Configuration of TMP0 to TMP3

Item	Configuration
Timer register	16-bit counter
Registers	TMPn capture/compare registers 0, 1 (TPnCCR0, TPnCCR1) TMPn counter read buffer register (TPnCNT) CCR0 buffer register, CCR1 buffer register
Timer inputs	2 (TIPn0[Note], TIPn1)
Timer outputs	2 (TOPn0, TOPn1)
Control registers	TMPn control registers 0, 1 (TPnCTL0, TPnCTL1) TMPn I/O control registers 0 to 2 (TPnIOC0 to TPnIOC2) TMPn option register 0 (TPnOPT0) Selector operation control registers 0 to 2 (SELCNT0 to SELCNT2)

Note: TIPn0 functions alternately as a capture trigger input signal and external trigger input signal.

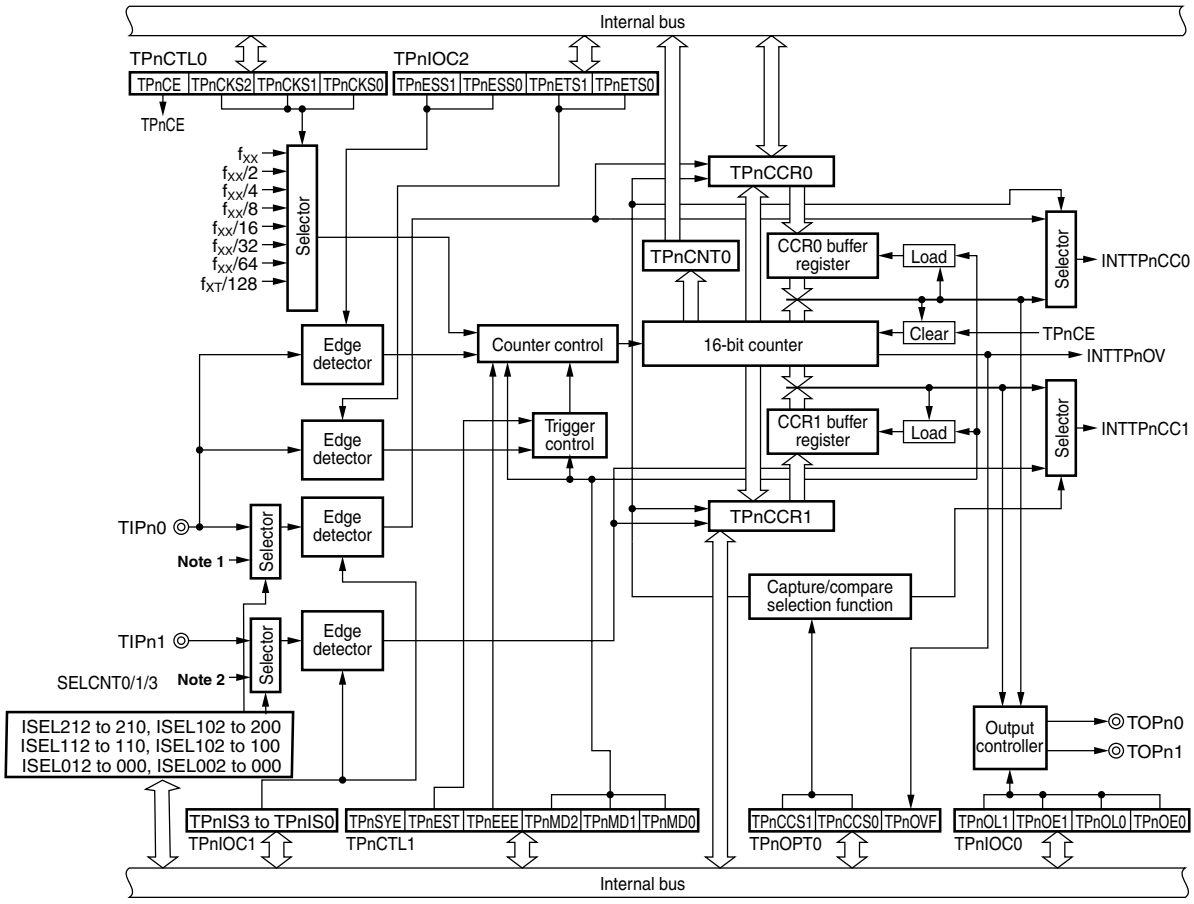
Remark: n = 0 to 2,
m = 0, 1

The pins of TMP function alternately as port pins. For how to set the alternate function, refer to the description of the registers in **Chapter 15 ”Port Functions” on page 749**.

Table 6-2: TMP Pin List

Pin Name		Alternate-Function Pin	I/O	Function
TIP00	TOP00	P20 / INTP3	Input/ Output	External event (TMP0)
TIP01	TOP01	P21 / INTP4		
TIP10	TOP10	P22 / INTP5		External event (TMP1)
TIP11	TOP11	P23 / INTP6		
TIP20	TOP20	P24 / INTP7		External event (TMP2)
TIP21	TOP21	P25 / INTP8		
TTRGP2		P26 / INTP9	Input	

Figure 6-1: Block Diagram of Timer P



Notes: 1. Refer to SELCNT0 to SELCNT2

2. Refer to SELCNT0 to SELCNT2

Remark: n=0 to 2

(1) TMPn capture/compare register 0 (TPnCCR0)

The TPnCCR0 register is a 16-bit register that has a capture function and a compare function. Whether this register is used as a capture register or a compare register can be specified by using the TPnCCS0 bit of the TPnOPT0 register, but only in the free-running mode.
In the pulse width measurement mode, this register can be used only as a capture register (it cannot be used as a compare register).
In all the modes other than the free-running mode and pulse width measurement mode, this register functions as a compare register.
In the default status, the TPnCCR0 register functions as a compare register.
This register can be read or written in 16-bit units.
Reset input clears this register to 0000H.

Figure 6-2: TMPn capture/compare register 0 (TPnCCR0) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCCR0									FFFFF596H		
(n=0 to 2)	7	6	5	4	3	2	1	0	FFFFF5A6H	R/W	0000H
									FFFFF5B6H		

- When used as compare register
  - TPnCCR0 can be rewritten when TPnCE = 1.
  - TPnCCR0 is rewritten as follows when TPnCE = 1.

TMP Operation Mode	Method of Writing TPnCCR0 Register
PWM output mode or external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse output mode, or interval timer mode	Any time write
Pulse width measurement mode	Cannot be used because used only as capture register

- When used as capture register
  - The count value is stored in TPnCCR0 on detection of the edge of the capture trigger (TIPn0) input.

(2) TMPn capture/compare register 1 (TPnCCR1)

The TPnCCR1 register is a 16-bit register that has a capture function and a compare function. Whether this register is used as a capture register or a compare register can be specified by using 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 only as a capture register (it cannot be used as a compare register).
In all the modes other than the free-running mode and pulse width measurement mode, this register functions as a compare register.
In the default status, the TPnCCR1 register functions as a reload register.
This register can be read or written in 16-bit units.
Reset input clears this register to 0000H.

Figure 6-3: TMPn capture/compare register 1 (TPnCCR1) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCCR1									FFFFF598H		
(n=0 to 2)	7	6	5	4	3	2	1	0	FFFFF5A8H	R/W	0000H
									FFFFF5B8H		

- When used as compare register
  - TPnCCR1 can be rewritten when TPnCE = 1.
  - TPnCCR1 is rewritten as follows when TPnCE = 1.

TMP Operation Mode	Method of Writing TPnCCR1 Register
PWM output mode or external trigger pulse output mode	Reload
Free-running mode, external event count mode, one-shot pulse output mode, or interval timer mode	Any time write
Pulse width measurement mode	Cannot be used because used only as capture register

- When used as capture register
  - The count value is stored in TPnCCR1 on detection of the edge of the capture trigger (TIPn1) input.

(3) TMPn counter read buffer register (TPnCNT)

The TPnCNT register is a read buffer register that can read the value of the 16-bit counter.
This register is read-only, in 16-bit units.
Reset input sets this register to FFFFH.
Although the hardware status is FFFFH when TPnCE = 0, 0000H is read from this register.
The value of the register is read when TOCE = 1.

Figure 6-4: TMPn counter read buffer register (TPnCNT) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCNT									FFFFF59AH		
(n=0 to 2)	7	6	5	4	3	2	1	0	FFFFF5AAH	R	FFFFH
									FFFFF5BAH		

6.4 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.

Figure 6-5: TMPn control register 0 (TPnCTL0) Format (1/2)

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnCTL0	TPnCE	0	0	0	0	TPnCKS2	TPnCKS1	TPnCKS0	FFFFF590H		
(n=0 to 2)									FFFFF5A0H	R/W	00H
									FFFFF5B0H		

TPnCE	Control of operation of timer Pn
0	Disable internal operating clock operation (asynchronously reset TMPn).
1	Enable internal operating clock operation.
The TPnCE bit controls the internal operating clock and asynchronously resets TMPn. When this bit is cleared to 0, the internal operating clock of TMPn is stopped (fixed to the low level), and TMPn is asynchronously reset. When the TPnCE bit is set to 1, the internal operating clock is enabled within 2 input clocks, and TMPn counts up.	

TPnCKS2	TPnCKS1	TPnCKS0	Selection of internal count clock
0	0	0	f_{XX}
0	0	1	$f_{XX}/2$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/8$
1	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 TPnCKS2 to TPnCKS0 bits when TPnCE = 0.
When the TPnCE bit setting is changed from 0 to 1, the TPnCKS2 to TPnCKS0 bits can be set at the same time.

Remark: f_{XX} : Main system clock frequency

Figure 6-5: TMPn control register 0 (TPnCTL0) Format (2/2)

Resolution and maximum number of counts:

Internal count clock	Resolution [μs]		Maximum count time [ms]	
	f_{XX} = 32 MHz	f_{XX} = 24 MHz	f_{XX} = 32 MHz	f_{XX} = 24 MHz
f_{XX}	0.031	0.042	2.05	2.72
$f_{XX}/2$	0.063	0.083	4.10	5.50
$f_{XX}/4$	0.125	0.167	8.19	10.94
$f_{XX}/8$	0.250	0.333	16.38	21.82
$f_{XX}/16$	0.500	0.667	32.77	43.64
$f_{XX}/32$	1.000	1.333	65.54	87.36
$f_{XX}/64$	2.000	2.667	131.11	174.78
$f_{XX}/128$	4.000	5.333	262.14	394.50

(2) TMPn timer 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.

Figure 6-6: TMPn timer control register 1 (TPnCTL1) Format (1/2)

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnCTL1	TPnSYE	TPnEST	TPnEEE	0	0	TPnMD2	TPnMD1	TPnMD0	FFFFF591H		
(n=0 to 2)									FFFFF5A1H	R/W	00H
									FFFFF5B1H		

TPnSYE	Tuned operation mode enable control
0	Independent operation mode (asynchronous operation mode)
1	Tuned operation mode (specification of slave operation) In this mode, timer P can operate in synchronization with a master timer Master timer: TMP0; Slave timer: TMP1, TMP2
Remark: For the tuned operation mode, refer to **6.6 "Timer Synchronized Operation Function" on page 274**. **Caution: Be sure to clear the TP0SYE and TP2SYE bits to 0.**	

Caution: TM0SYE bit must always set to 0!

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 or external trigger pulse output mode (this bit is invalid in any other mode). By setting TPnEST to 1 when TPnCE = 1, a software trigger is issued. 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	Selection of count clock
0	Internal clock (clock selected by TPnCKS2 to TPnCKS0 bits)
1	External clock (edge of input to TIPn0)
The valid edge is specified by the TPnEES1 and TPnEES0 bits when TPnEET = 1 (external clock: TIPn0).	

Figure 6-6: TMPn timer control register 1 (TPnCTL1) Format (2/2)

TPnMD2	TPnMD1	TPnMD0	Selection of timer mode
0	0	0	Interval timer mode
0	0	1	forbidden
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). If these bits are rewritten when TPnCE = 1, the operation cannot be guaranteed. If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

(3) TMPn I/O control register 0 (TPnIOC0)

The TPnIOC0 register is an 8-bit register that controls the timer outputs (TOPn0 and TOPn1).
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this register to 00H.

Figure 6-7: TMPn I/O control register 0 (TPnIOC0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnIOC0	0	0	0	0	TPnOL1	TPnOE1	TPnOL0	TPnOE0	FFFFF592H		
(n=0 to 2)									FFFFF5A2H	R/W	00H
									FFFFF5B2H		

TPnOLm	Setting of TOPnm output level (m = 0, 1)
0	Normal output
1	Inverted output

TPnOEm	Setting of TOPnm output (m = 0, 1)
0	Disable timer output (TOPnm pin outputs low level when TPnOLm = 0, and high level when TPnOLm = 1)
1	Enable timer output (TOPnm pin outputs pulses).

Cautions: 1. Rewrite the TPnOL1, TPnOE1, TPnOL0, and TPnOE0 bits when TPnCE0 = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

2. To enable the timer output, be sure to set the corresponding alternate-function pins TPnIS3 to TPnIS0 of the TPnIOC1 register to “Detect no edge” and invalidate the capture operation. Then set the corresponding alternate-function port to output mode.

(4) TMPn I/O control register 1 (TPnIOC1)

The TPnIOC1 register is an 8-bit register that controls the valid edge of the external input signals (TIPn0 and TIPn1).
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this register to 00H.

Figure 6-8: TMPn I/O control register 1 (TPnIOC1) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnIOC1	0	0	0	0	TPnIS3	TPnIS2	TPnIS1	TPnIS0	FFFFF593H		
(n=0 to 2)									FFFFF5A3H	R/W	00H
									FFFFF5B3H		

TPnIS3	TPnIS2	Setting of valid edge of capture input (TIPn1)
0	0	Detect no edge (capture operation is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

TPnIS1	TPnIS0	Setting of valid edge of capture input (TIPn0)
0	0	Detect no edge (capture operation is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

Cautions: 1. Rewrite the TPnIS3 to TPnIS0 bits when TPnCE0 = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

2. The TPnIS3 to TPnIS0 bits are valid only in the free-running mode and pulse width measurement mode. A capture operation is not performed in any other mode.

3. If used as the capture input, be sure to set the corresponding alternate-function pins TPnOE1 and TPnOE0 of the TPnIOC0 register to “Disable timer output” and set the capture input valid edge. Then set the corresponding alternate-function port to input mode.

(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.

Figure 6-9: TMPn I/O control register 2 (TPnIOC2) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnIOC2	0	0	0	0	TPnEES1	TPnEES0	TPnETS1	TPnETS0	FFFFF594H		
(n=0 to 2)									FFFFF5A4H	R/W	00H
									FFFFF5B4H		

TPnEES1	TPnEES0	Setting of valid edge of external event count input (TIP00)
0	0	Detect no edge (external event count is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

TPnETS1	TPnETS0	Setting of valid edge of external trigger input (TIP00)
0	0	Detect no edge (external trigger is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

Cautions: 1. Rewrite the TPnEES1, TPnEES0, TPnEST1, and TPnEST0 bits when TPnCE = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

2. The TPnEES1 and TPnEES0 bits are valid when TPnEEE = 1 or when the external event count mode is set (TPnMD2 to TPnMD0 of TIPnCTL1 register = 001).

(6) TMPn option register 0 (TPnOPT0)

The TPnOPT0 register is an 8-bit register that selects a capture or compare operation, and detects an overflow.
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this register to 00H.

Figure 6-10: TMPn option register 0 (TPnOPT0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnOPT0	0	0	TPnCCS1	TPnCCS0	0	0	0	TPnOVF	FFFFF595H		
(n=0 to 2)									FFFFF5A5H	R/W	00H
									FFFFF5B5H		

TPnCCSm	Selection of capture or compare operation of TPnCCRm register (m = 0, 1)
0	Compare register
1	Capture register
The set value of the TPnCCSm bit is valid only in the free-running mode	

TPnOVF	Detection of overflow of timer P
Set (1)	Overflow occurred
Reset (0)	0 written to TPnOVF bit or TPnCE = 0
• The TPnOVF bit is set when the 16-bit counter overflows from FFFFH to 0000H in the free-running mode and pulse width measurement mode. • As soon as the TPnOVF bit has been set to 1, an interrupt request signal (INTTPnOV) is generated. The INTTPnOV signal is not generated in any mode other than the free-running mode and pulse width measurement mode. • The TPnOVF bit is not cleared even if the TPnOVF bit and TPnOPT0 register are read when TPnOVF = 1. • The TPnOVF bit can be read and written, but 1 cannot be written to the TPnOVF bit. Writing 1 to this bit does not affect the operation of timer P.	

Caution: Rewrite the TPnCCS1 and TPnCCS0 bits when TPnCE0 = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

(7) Selector operation control register 0 (SELCNT0)

SELCNT0 controls the input source for TPTTI0 and TPTTI1 of TMP0.

Register SELCNT0 can be read and written in 8-bit units.

Figure 6-11: Selector operation control register 0 (SELCNT0) Format

	7	6	5	4	3	2	1	0	Address	R/W	Initial value
SELCNT0	0	ISEL012	ISEL011	ISEL010	0	ISEL002	ISEL001	ISEL000	FFFFF300H	R/W	00H

Bit position	Bit name	Function
0…2	ISEL000, ISEL001, ISEL002	Selects input source for TPTTI0 of TMP0

ISEL002	ISEL001	ISEL000	Source of TPTTI00
0	0	0	TIP00
0	0	1	RXDA0
0	1	0	TSOUT0[Note 1]
0	1	1	TSOUT2[Note 1]
1	0	0	TSOUT4[Note 1]
others			Don't use. Decode as TIP00

Bit position	Bit name	Function
4…6	ISEL010, ISEL011, ISEL012	Selects input source for TPTTI1 of TMP0

ISEL012	ISEL011	ISEL010	Source of TPTTI01
0	0	0	TIP01
0	0	1	RXDA1
0	1	0	TSOUT1[Note 1]
0	1	1	TSOUT3[Note 1]
1	0	0	TSOUTR[Notes 1, 2]
others			Don't use. Decode as TIP01

Notes: 1. TSOUTn is a timer trigger signal from the AFCAN macro and can be used for Time Stamp function. For details please refer to 13.14 "Time Stamp Function" on page 595.

2. TSOUTR is the TSOUT signal from the RXONLY_CH of DAFCAN.

(8) Selector operation control register 1 (SELCNT1)

SELCNT1 controls the input source for TPTTI0 and TPTTI1 of TMP1.

Register SELCNT1 can be read and written in 8-bit units.

Figure 6-12: Selector operation control register 1 (SELCNT1) Format

	7	6	5	4	3	2	1	0	Address	R/W	Initial value
SELCNT1	0	ISEL112	ISEL111	ISEL110	0	ISEL102	ISEL101	ISEL100	FFFFF302H	R/W	00H

Bit position	Bit name	Function
0…2	ISEL100, ISEL101, ISEL102	Selects input source for TPTTI0 of TMP1
4…6	ISEL110, ISEL111, ISEL112	Selects input source for TPTTI1 of TMP1

Selects input source for TPTTI0 of TMP1

ISEL102	ISEL101	ISEL100	Source of TPTTI10
0	0	0	TIP10
0	0	1	RXDA0
0	1	0	TSOUT0[Note 1]
0	1	1	TSOUT2[Note 1]
1	0	0	TSOUT4[Note 1]
others			Don't use. Decode as TIP00

Selects input source for TPTTI1 of TMP1

ISEL112	ISEL111	ISEL110	Source of TPTTI11
0	0	0	TIP11
0	0	1	RXDA1
0	1	0	TSOUT1[Note 1]
0	1	1	TSOUT3[Note 1]
1	0	0	TSOUTR[Notes 1, 2]
others			Don't use. Decode as TIP11

Notes: 1. TSOUTn is a timer trigger signal from the AFCAN macro and can be used for Time Stamp function. For details please refer to 13.14 "Time Stamp Function" on page 595.

2. TSOUTR is the TSOUT signal from the RXONLY_CH of DAFCAN.

(9) Selector operation control register 2 (SELCNT2)

SELCNT2 controls the input source for TPTTI0 and TPTTI1 of TMP2.

Register SELCNT2 can be read and written in 8-bit units.

Figure 6-13: Selector operation control register 2 (SELCNT2) Format

	7	6	5	4	3	2	1	0	Address	R/W	Initial value
SELCNT2	0	ISEL212	ISEL211	ISEL210	0	ISEL202	ISEL201	ISEL200	FFFFF304H	R/W	00H

Bit position	Bit name	Function
0…2	ISEL200, ISEL201, ISEL202	Selects input source for TPTTI0 of TMP2
4…6	ISEL210, ISEL211, ISEL212	Selects input source for TPTTI1 of TMP0

Selects input source for TPTTI0 of TMP2

ISEL202	ISEL201	ISEL200	Source of TPTTI20
0	0	0	TIP20
0	0	1	RXDA0
0	1	0	TSOUT0[Note 1]
0	1	1	TSOUT2[Note 1]
1	0	0	TSOUT4[Note 1]
others			Don't use. Decode as TIP20

Selects input source for TPTTI1 of TMP0

ISEL212	ISEL211	ISEL210	Source of TPTTI21
0	0	0	TIP21
0	0	1	RXDA1
0	1	0	TSOUT1[Note 1]
0	1	1	TSOUT3[Note 1]
1	0	0	TSOUTR[Notes 1, 2]
others			Don't use. Decode as TIP21

Notes: 1. TSOUTn is a timer trigger signal from the AFCAN macro and can be used for Time Stamp function. For details please refer to 13.14 "Time Stamp Function" on page 595.

2. TSOUTR is the TSOUT signal from the RXONLY_CH of DAFCAN.

6.5 Operation

Timer P performs the following operations.

Operation	TPnEST (Software Trigger Bit)	TIPn0 (External Trigger Input)	Capture/Compare Selection	Compare Write
Interval timer mode	Invalid	Invalid	Compare only	Anytime write
External trigger pulse output mode[Note]	Valid	Valid	Compare only	Reload
One-shot pulse output mode[Note]	Valid	Valid	Compare only	Anytime write
PWM mode	Invalid	Invalid	Compare only	Reload
Free-running mode	Invalid	Invalid	Capture/compare selectable	Anytime write
Pulse width measurement mode[Note]	Invalid	Invalid	Capture only	Not applicable

Note: To use the external trigger pulse output mode, one-shot pulse mode, or pulse width measurement mode, select a count clock (by clearing the TPnEEE bit of the TPnCTL1 register to 0).

Remark: n = 0 to 2

6.5.1 Anytime write and reload

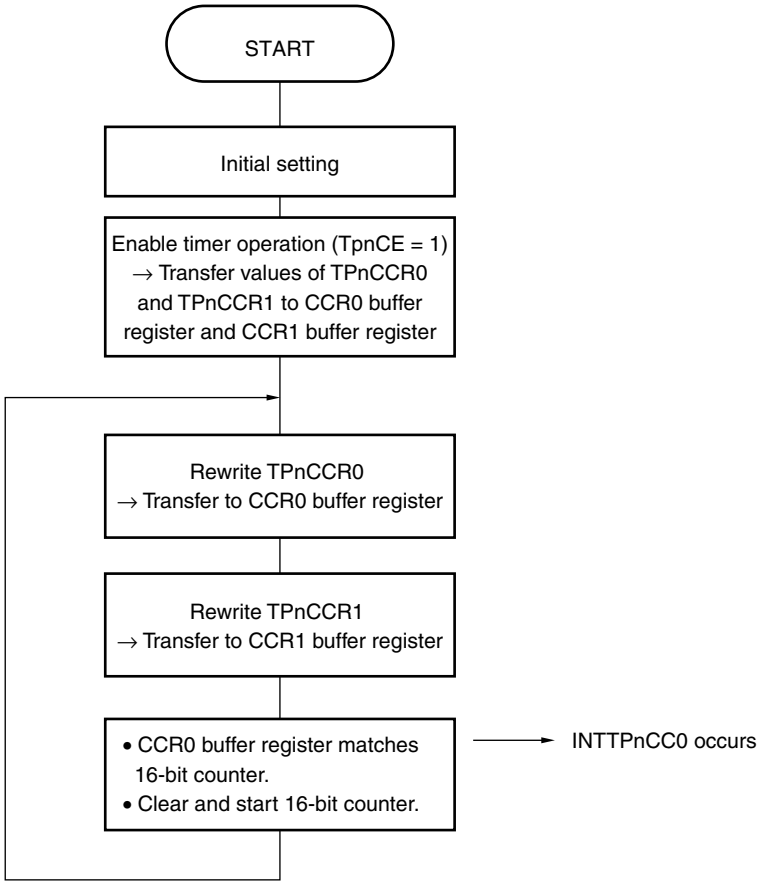
Timer P allows rewriting of the TPnCCR0 and TPnCCR1 registers while the timer is operating (TPnCE = 1). These registers are written differently (anytime write or reload) depending on the mode.

(1) Anytime write

When data is written to the TPnCCRm register during timer operation, it is transferred at any time to the CCRm buffer register and is compared with the value of the 16-bit counter.

Remark: n=0 to 2
m = 0, 1

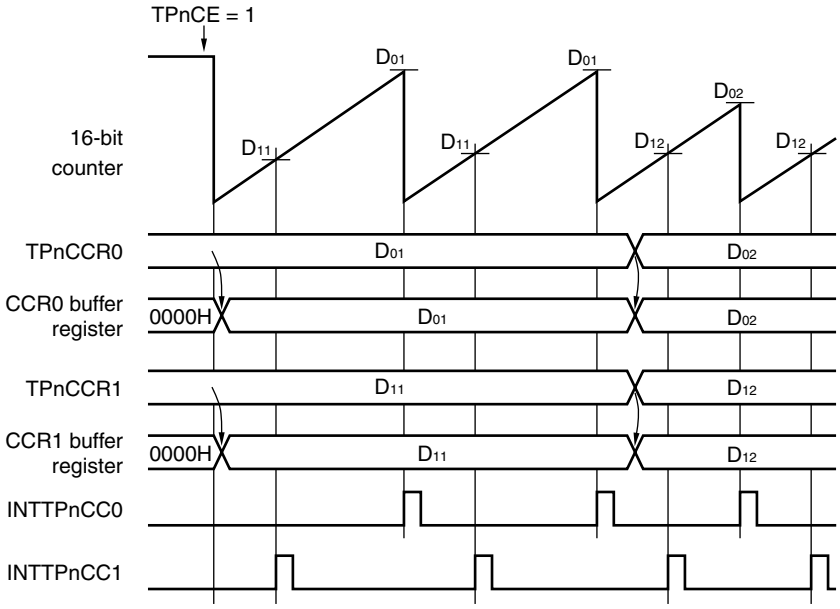
Figure 6-14: Flowchart of Basic Operation of Anytime Write



Remarks: **1.** This is an example in the interval timer mode.

2. n=0 to 2

Figure 6-15: Timing Chart of Anytime Write



Remarks: 1. D_{01}, D_{02}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Set value of TPnCCR1 register (0000H to FFFFH)

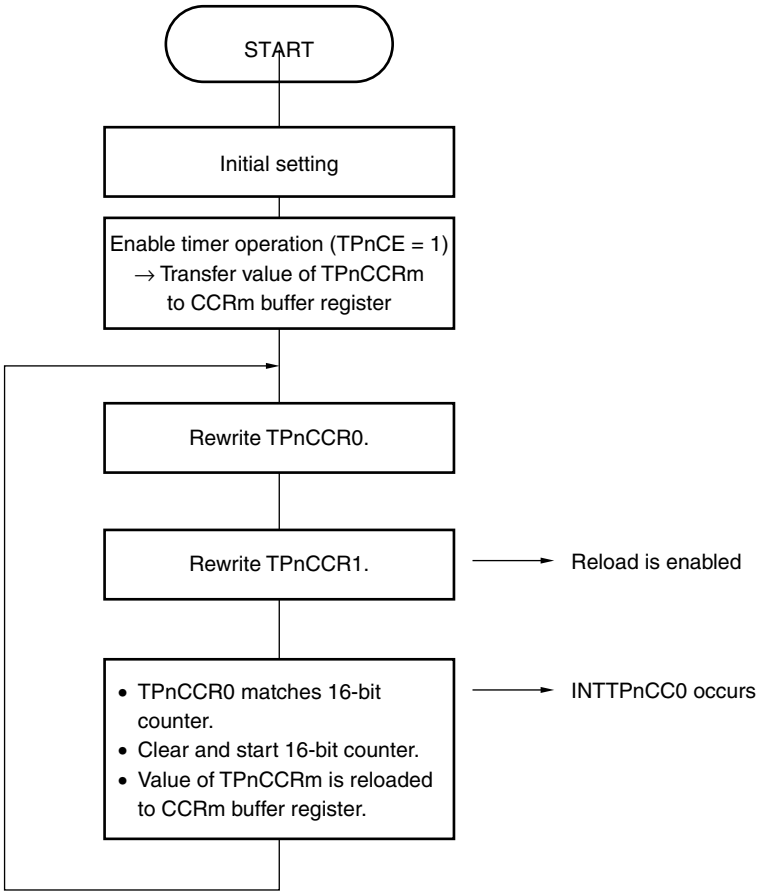
2. This is an example in the interval timer mode.

3. n=0 to 2

(2) Reload

When data is written to the TPnCCR0 and TPnCCR1 registers during timer operation, it is compared with the value of the 16-bit counter via the CCRm buffer register. The values of the TPnCCR0 and TPnCCR1 registers can be rewritten when TPnCE = 1.
So that the set values of the TPnCCR0 and TPnCCR1 registers are compared with the value of the 16-bit counter (the set values are reloaded to the CCRm buffer register), the value of the TPnCCR0 register must be rewritten and then a value must be written to the TPnCCR1 register before the value of the 16-bit counter matches the value of TPnCCR0. When the value of the TPnCCR0 register matches the value of the 16-bit counter, the values of the TPnCCR0 and TPnCCR1 registers are reloaded.
Whether the next reload timing is made valid or not is controlled by writing to the TPnCCR1 register. Therefore, write the same value to the TPnCCR1 register when it is necessary to rewrite the value of only the TPnCCR0 register.

Figure 6-16: Flowchart of Basic Operation of Reload

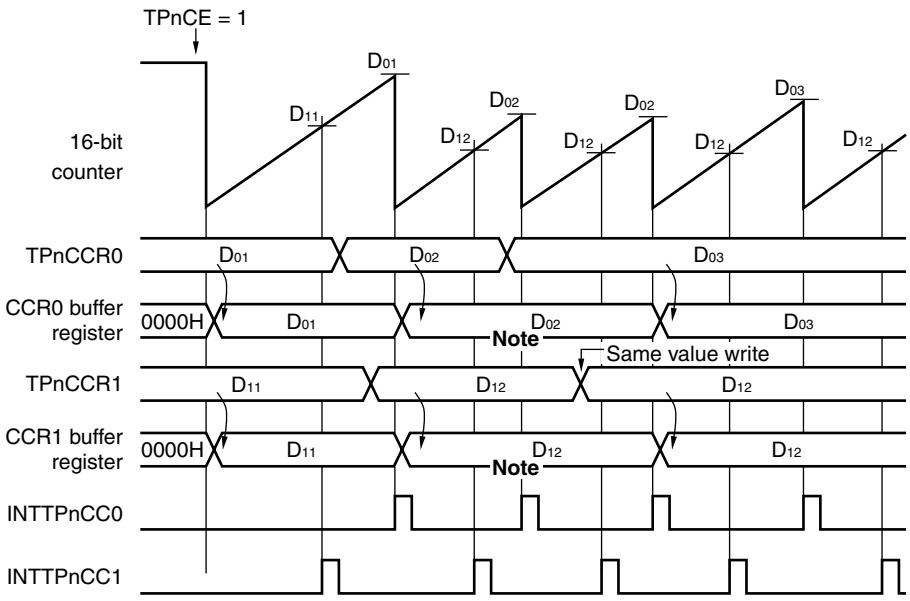


Caution: Writing the TPnCCR1 register includes an operation to enable reload. Therefore, rewrite the TPnCCR1 register after rewriting the TPnCCR0 register.

Remarks: 1. This is an example in the PWM mode.

2. n=0 to 2, m = 0, 1

Figure 6-17: Timing Chart of Reload



Note: The value is not reloaded because the TPnCCR1 register is not written.

Remarks: **1.** D_{01}, D_{02}, D_{03}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Set value of TPnCCR1 register (0000H to FFFFH)

2. This is an example in the PWM mode.

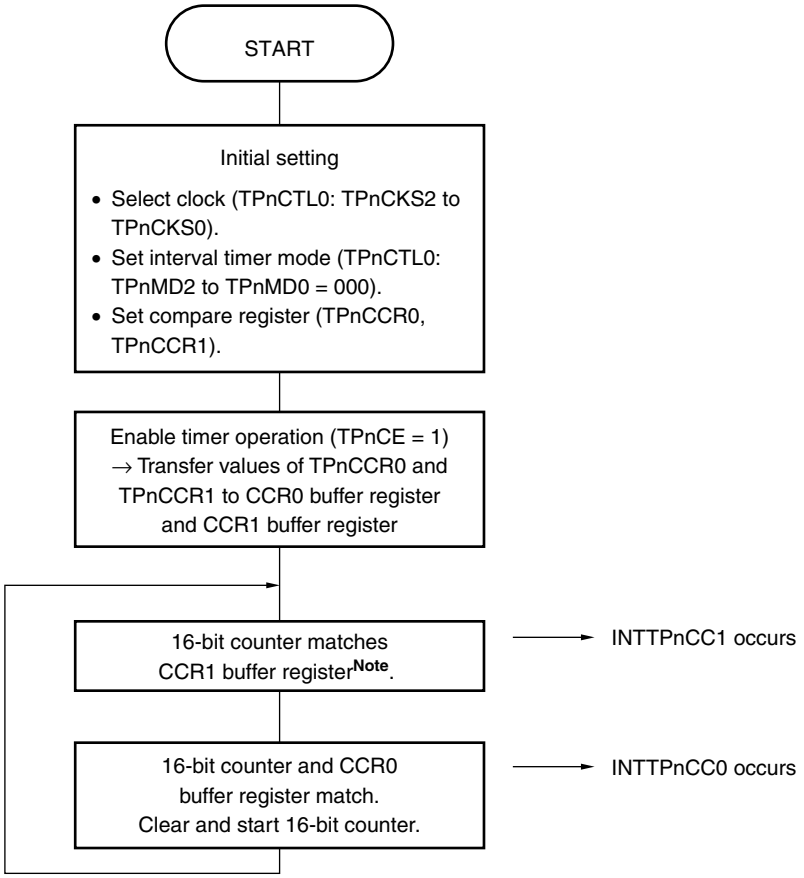
3. n=0 to 2

6.5.2 Interval timer mode (TPnMD2 to TPnMD0 = 000)

In the interval timer mode, an interrupt request signal (INTTPnCC0) is generated when the set value of the TPnCCR0 register matches the value of the 16-bit counter, and the 16-bit counter is cleared. Rewriting the TPnCCR0 register is enabled when TPnCE = 1. When a value is set to the TPnCCR0 register by a write instruction from the CPU, it is transferred to the CCR0 buffer register by means of anytime write, and is compared with the value of the 16-bit counter.
In the interval timer mode, the 16-bit counter can be cleared only when its value matches the value of the CCR0 buffer register.
The 16-bit counter is not cleared by using the TPnCCR1 register.
However, the set value of the TPnCCR1 register is transferred to the CCR1 buffer register and compared with the value of the 16-bit counter. As a result, an interrupt request (INTTPnCC1) is generated.
The value can also be output from the TOPnm pin by setting the TPnOEm bit to 1.
When the TPnCCR1 register is not used, it is recommended to set the TPnCCR1 register to FFFFH.

Remark: n=0 to 2
m = 0, 1

Figure 6-18: Flowchart of Basic Operation in Interval Timer Mode

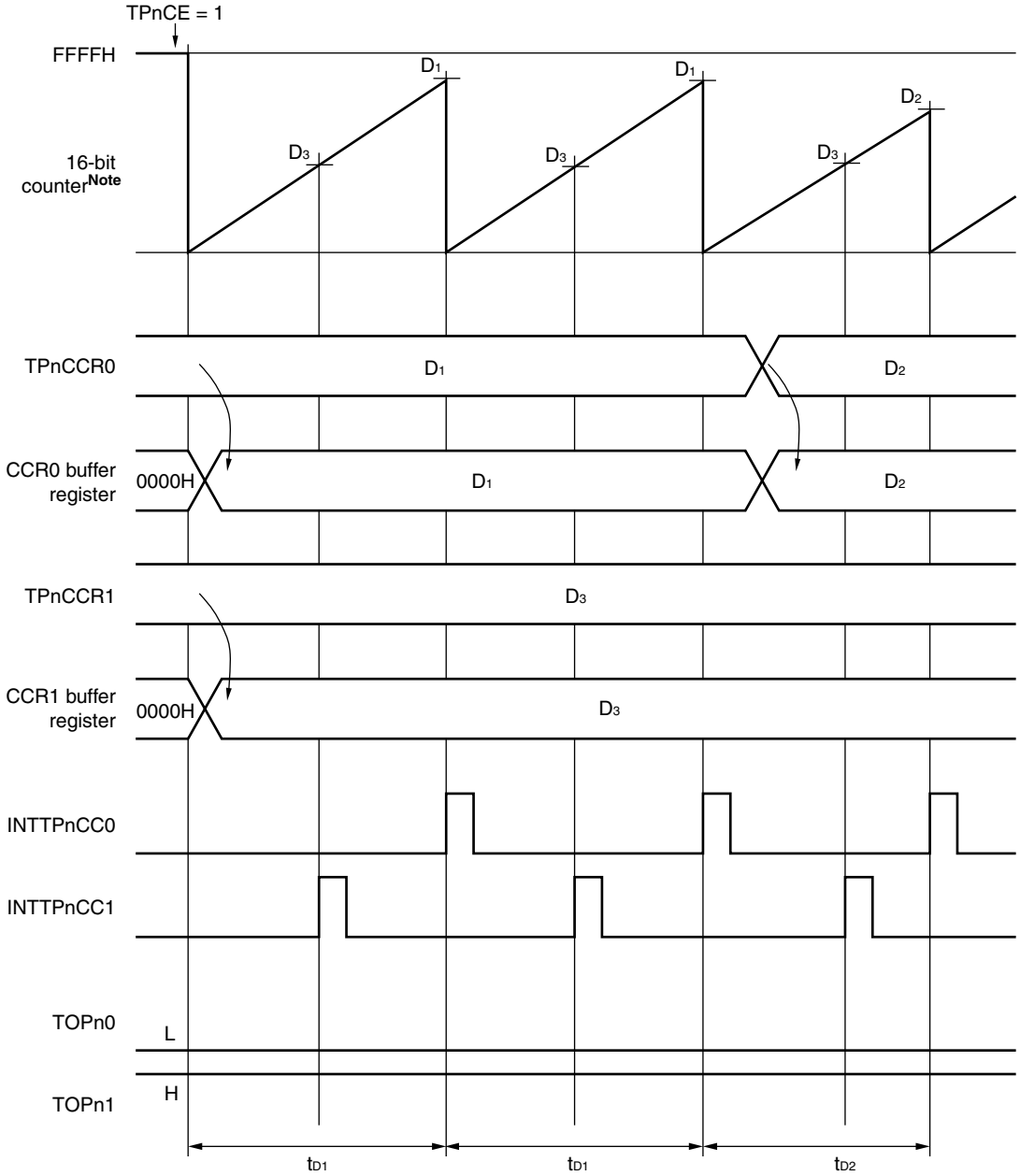


Note: The 16-bit counter is not cleared when its value matches the value of TPnCCR1.

Remark: n=0 to 2

Figure 6-19: Timing of Basic Operation in Interval Timer Mode (1/2)

(a) When $D_1 > D_2 > D_3$, only TPnCCR0 register value is written, and TOPn0 and TOPn1 are not output (TPnOE0 = 0, TPnOE1 = 0, TPnOL0 = 0, TPnOL1 = 1)



Note: The 16-bit counter is not cleared when its value matches the value of TPnCCR1.

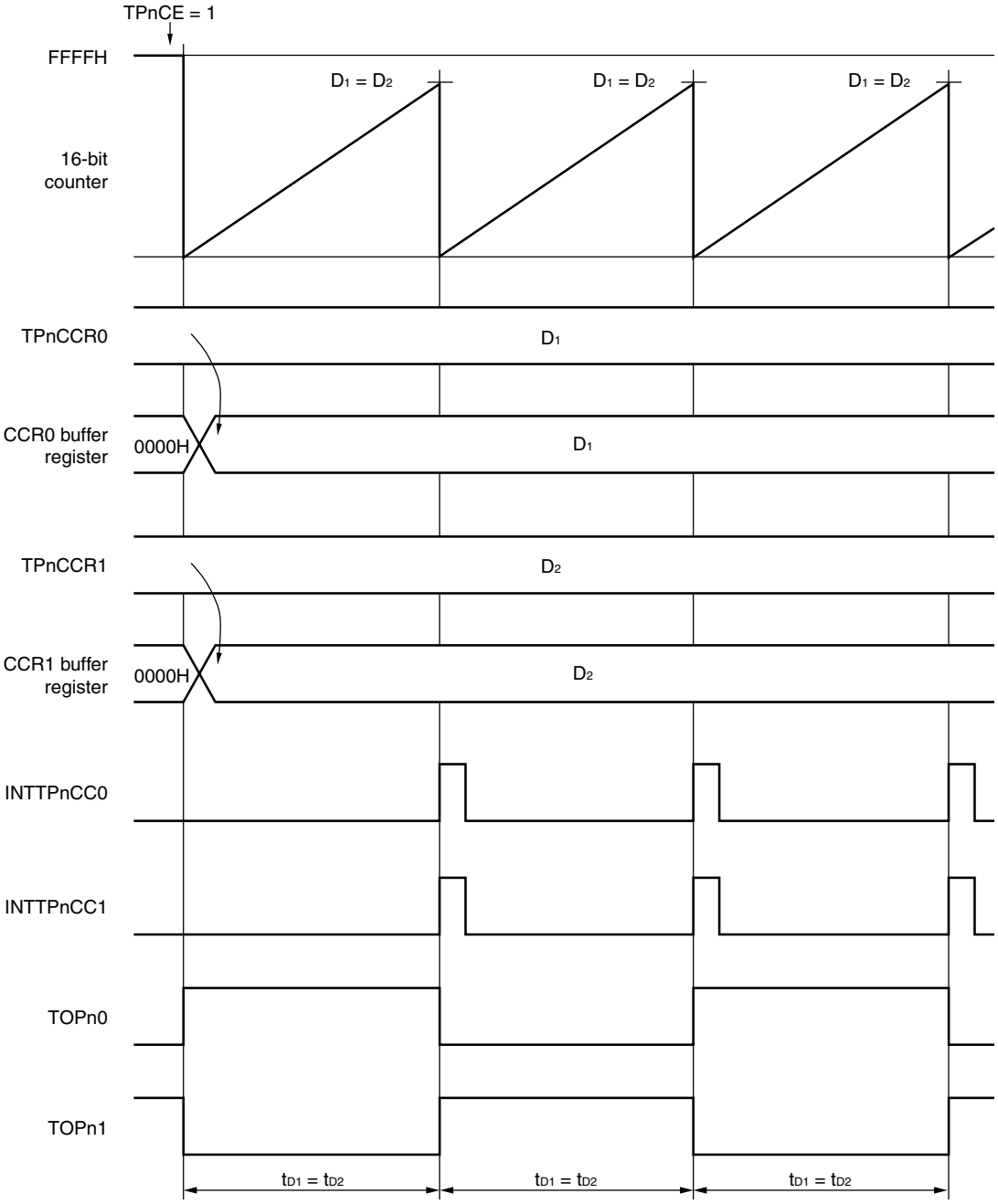
Remarks: 1. D_1, D_2: Set value of TPnCCR0 register (0000H to FFFFH)
D_3: Set value of TPnCCR1 register (0000H to FFFFH)

2. Interval time (t_{Dn}) = (Dn + 1) × (Count clock cycle)

3. n=0 to 2

Figure 6-19: Timing of Basic Operation in Interval Timer Mode (2/2)

(b) When $D_1 = D_2$, TPnCCR0 and TPnCCR1 are not rewritten, and TOPn0 and TOPn1 are output (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 1)



Remarks: 1. D_1: Set value of TPnCCR0 register (0000H to FFFFH)
D_2: Set value of TPnCCR1 register (0000H to FFFFH)

2. Interval time (t_{Dn}) = (Dn + 1) × (Count clock cycle)

3. n=0 to 2

6.5.3 External trigger pulse mode (TPnMD2 to TPnMD0 = 010)

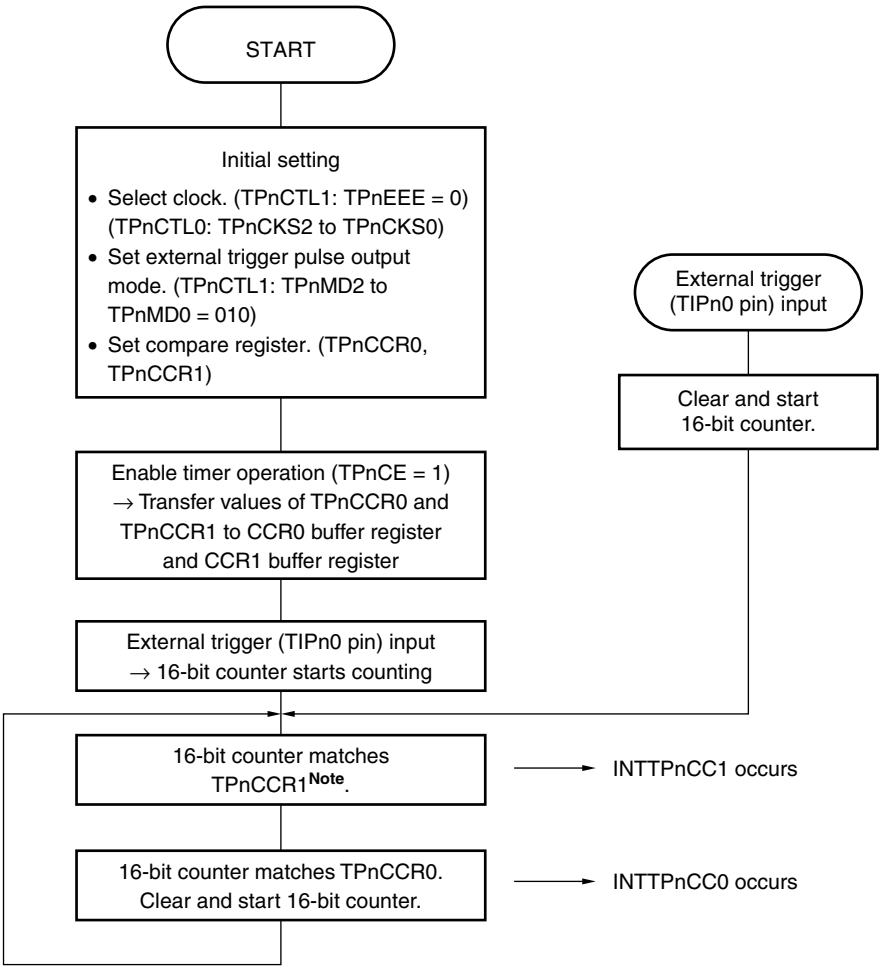
When TPnCE = 1 in the external trigger pulse mode, the 16-bit counter stops at FFFFH and waits for input of an external trigger (TIPn0 pin input). When the counter detects the edge of the external trigger (TIPn0 pin input), it starts counting up.
The duty factor of the signal output from the TOPn1 pin is set by a reload register (TPnCCR1) and the period is set by a compare register (TPnCCR0).
Rewriting the TPnCCR0 and TPnCCR1 registers is enabled when TPnCE = 1.
So that the set values of the TPnCCR0 and TPnCCR1 registers after rewriting are compared with the value of the 16-bit counter (reloaded to the CCRm buffer register), the TPnCCR0 register must be rewritten and then a value is written to the TPnCCR1 register before the value of the 16-bit counter matches the value of the TPnCCR0 register. When the value of the TPnCCR0 register later matches the value of the 16-bit counter, the values of the TPnCCR0 and TPnCCR1 registers are reloaded to the CCRm buffer register.
Whether the next reload timing is made valid or not is controlled by writing to the TPnCCR1 register. Therefore, write the same value to the TPnCCR1 register when it is necessary to rewrite the value of only the TPnCCR0 register. Reload is invalid when only the TPnCCR0 register is rewritten. To stop timer P, clear TPnCE to 0. If the edge of the external trigger (TIPn0 pin input) is detected more than once in the external trigger pulse mode, the 16-bit counter is cleared at the point of edge detection, and resumes counting up. To realize the same function as the external trigger pulse mode by using a software trigger instead of the external trigger input (TIPn0 pin input) (software trigger pulse mode), a software trigger is generated by setting the TPnEST bit of the TPnCTL1 register to 1. The waveform of the external trigger pulse is output from TOPn1. A toggle output is produced from the TOPn0 pin when the value of the TPnCCR0 register matches the value of the 16-bit counter.
In the external trigger pulse mode, the capture function of the TPnCCR0 and TPnCCR1 registers cannot be used because these registers can be used only as compare registers.

Caution: In the external trigger pulse mode, select the internal clock (TPnEEE of TPnCTL1 register = 0) as the count clock.

Remarks: **1.** For the reload operation when TPnCCR0 and TPnCCR1 are rewritten during timer operation, refer to **6.5.5 ”PWM mode (TPnMD2 to TPnMD0 = 100)” on page 262**.

2. n=0 to 2
m = 0, 1

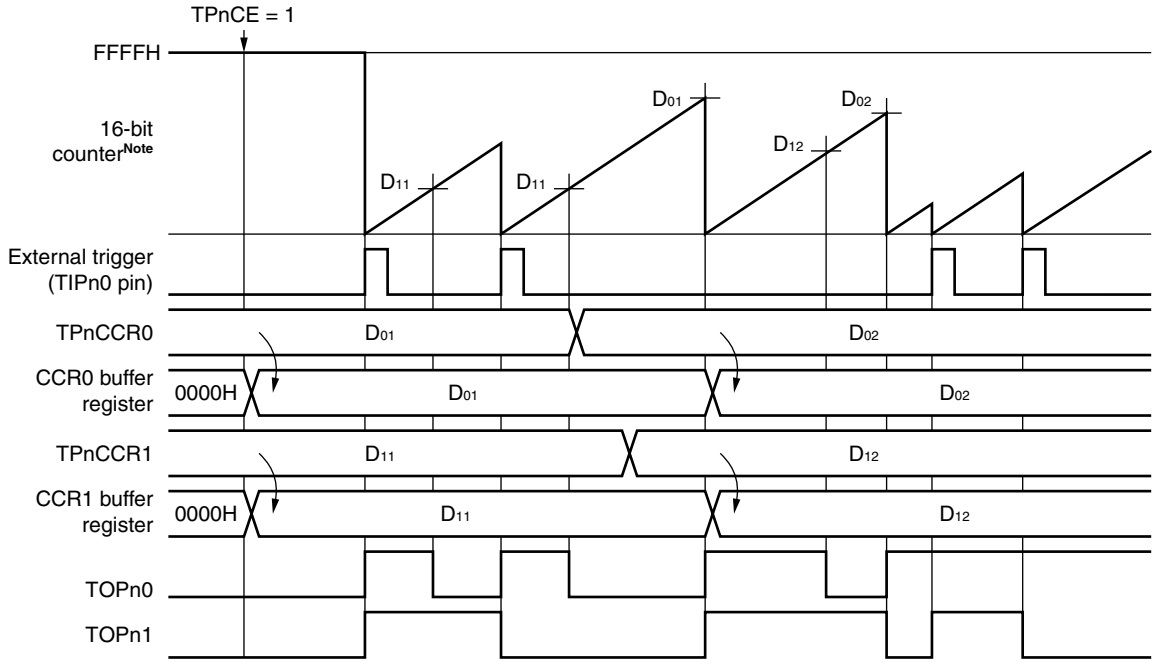
Figure 6-20: Flowchart of Basic Operation in External Trigger Pulse Output Mode



Note: The 16-bit counter is not cleared when it matches the CCR1 buffer register.

Remark: n=0 to 2

Figure 6-21: Timing of Basic Operation in External Trigger Pulse Output Mode (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Note: The 16-bit counter is not cleared when it matches the CCR1 buffer register.

Remarks: 1. D_{01}, D_{02}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Duty of TOPn1 output = (Set value of TPnCCR1 register) / (Set value of TP0CCR0 register)
Cycle of TOPn1 output = (Set value of TPnCCR0 register) × (Count clock cycle)

3. n=0 to 2

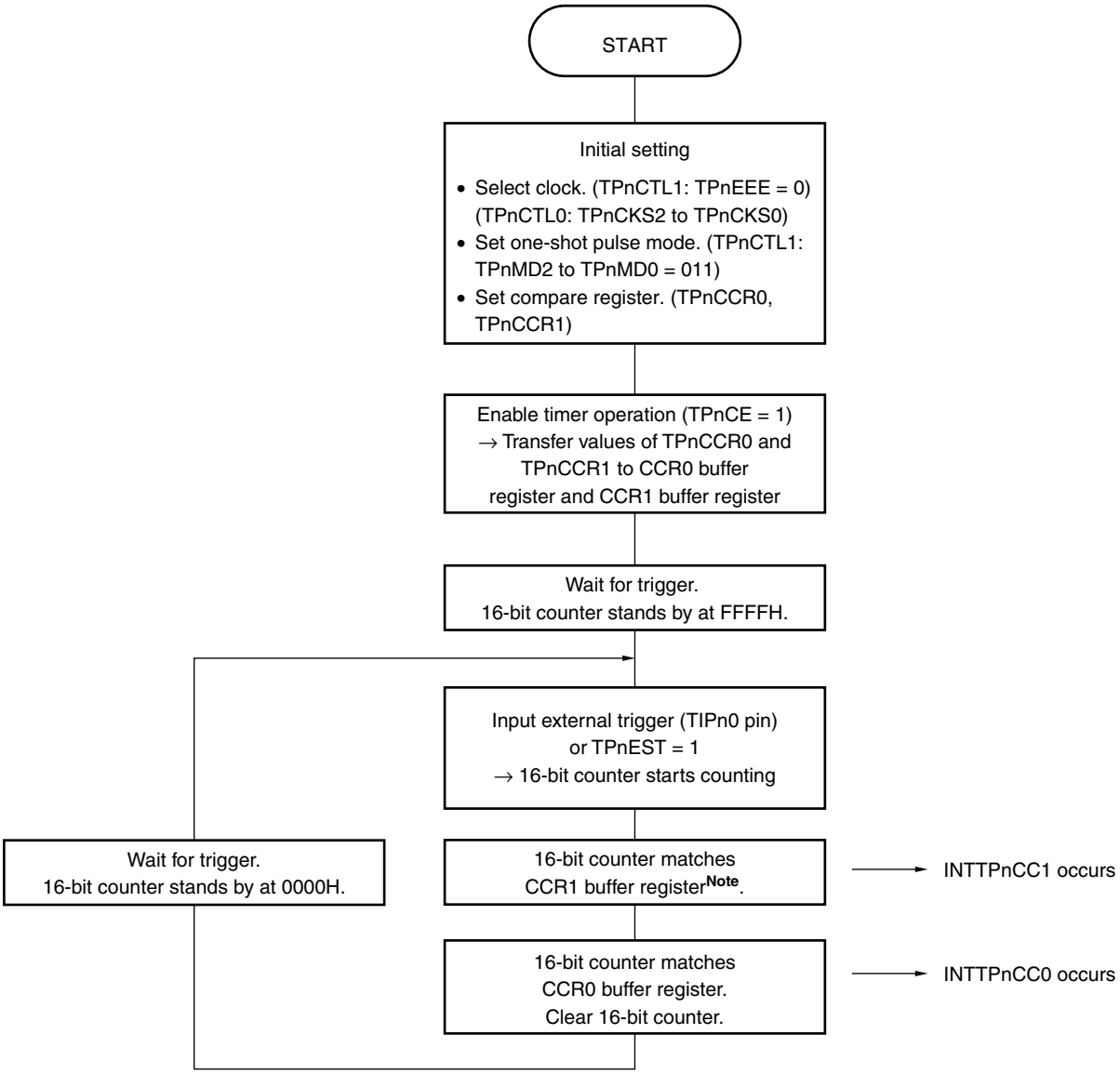
6.5.4 One-shot pulse mode (TPnMD2 to TPnMD0 = 011)

When TPnCE is set to 1 in the one-shot pulse mode, the 16-bit counter waits for the setting of the TPnEST bit (to 1) or a trigger that is input when the edge of the TIPn0 pin is detected, while holding FFFFH. When the trigger is input, the 16-bit counter starts counting up. When the value of the 16-bit counter matches the value of the CCR1 buffer register that has been transferred from the TPnCCR1 register, TOPn1 goes high. When the value of the 16-bit counter matches the value of the CCR0 buffer register that has been transferred from the TPnCCR0 register, TOPn1 goes low, and the 16-bit counter is cleared to 0000H and stops. Input of a second or subsequent trigger is ignored while the 16-bit counter is operating. Be sure to input a second trigger while the 16-bit counter is stopped at 0000H. In the one-shot pulse mode, rewriting the TPnCCR0 and TPnCCR1 registers is enabled when TPnCE = 1. The set values of the TPnCCR0 and TPnCCR1 registers become valid after a write instruction from the CPU is executed. They are then transferred to the CCR0 and CCR1 buffer registers, and compared with the value of the 16-bit counter. The waveform of the one-shot pulse is output from the TOPn1 pin. The TOPn0 pin produces a toggle output when the value of the 16-bit counter matches the value of the TPnCCR0 register.
In the one-shot pulse mode, the TPnCCR0 and TPnCCR1 registers function only as compare registers. They cannot be used as capture registers.

Caution: Select the internal clock (TPnEEE of the TPnCTL1 register = 0) as the count clock in the one-shot pulse mode.

Remark: n=0 to 2

Figure 6-22: Flowchart of Basic Operation in One-Shot Pulse Mode

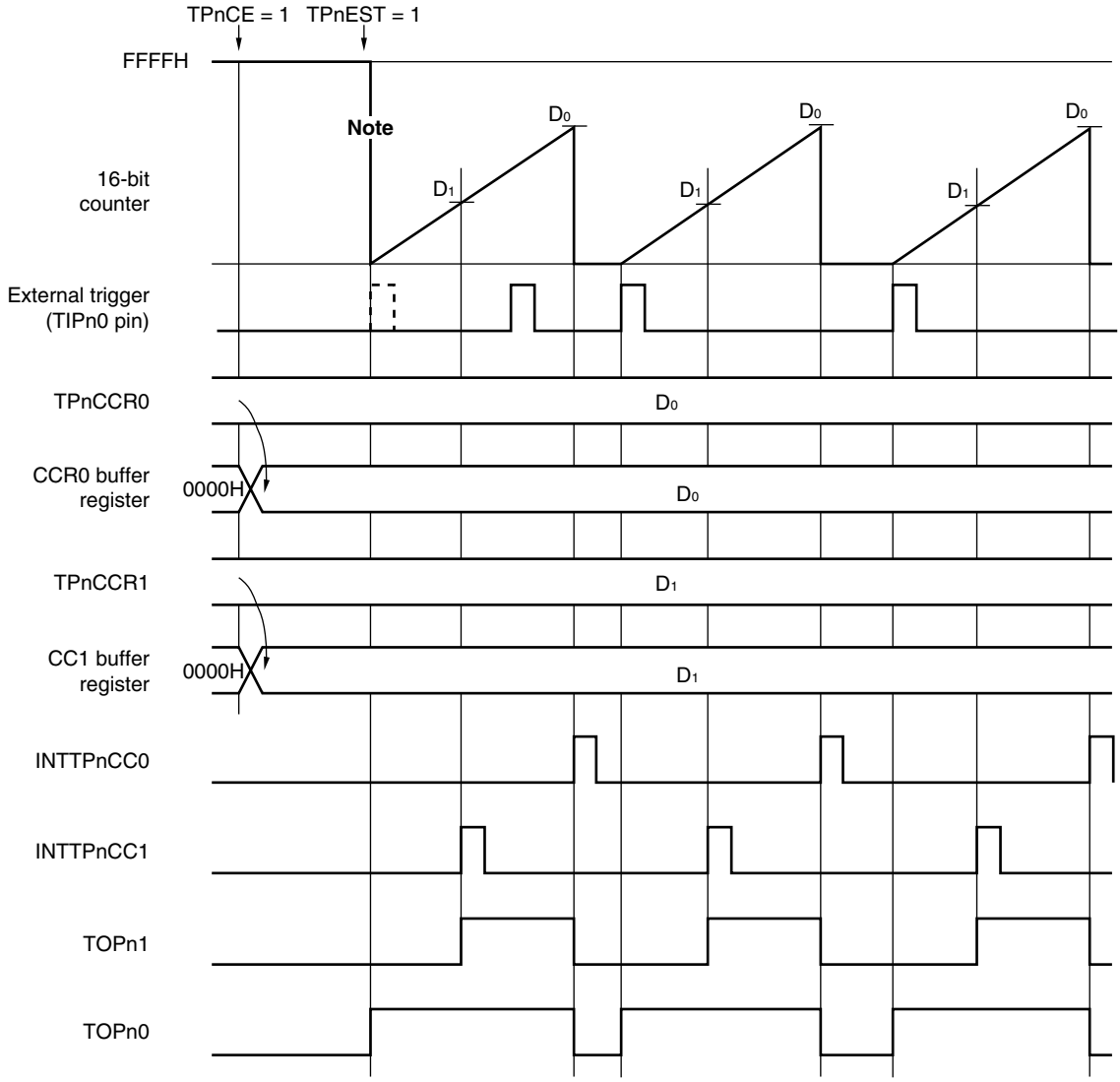


Note: The 16-bit counter is not cleared when it matches the CCR1 buffer register.

Caution: The 16-bit counter is not cleared even if the trigger is input while the counter is counting up, and the trigger input is ignored.

Remark: n=0 to 2

Figure 6-23: Timing of Basic Operation in One-Shot Pulse Mode (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Note: The 16-bit counter starts counting up either when TPnEST = 1 or when TIPn0 is input.

Remarks: 1. D_0: Set value of TPnCCR0 register (0000H to FFFFH)
D_1: Set value of TPnCCR1 register (0000H to FFFFH)

2. n=0 to 2

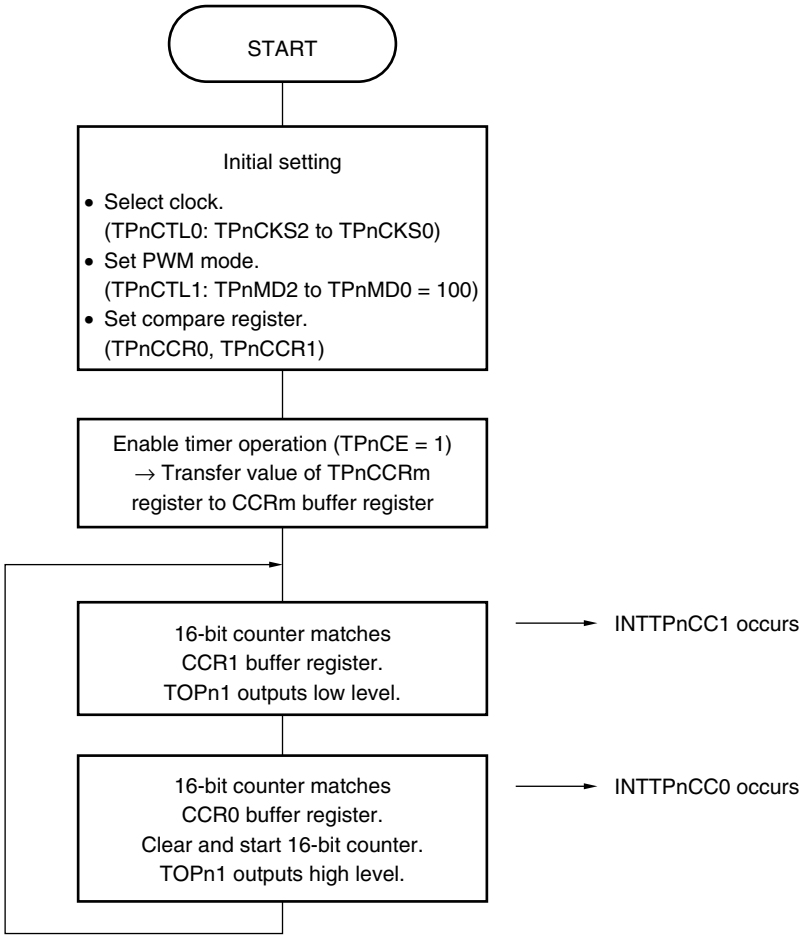
6.5.5 PWM mode (TPnMD2 to TPnMD0 = 100)

In the PWM mode, TMPn capture/compare register 1 (TPnCCR1) is used to set the duty factor and TMPn capture/compare register 0 (TPnCCR0) is used to set the cycle.
By using these two registers and operating the timer, variable-duty PWM is output.
Rewriting the TPnCCR0 and TPnCCR1 registers is enabled when TPnCE = 1.
So that the set values of the TPnCCR0 and TPnCCR1 registers are compared with the value of the 16-bit counter (reloaded to the CCR0 and CCR1 buffer registers), the TPnCCR0 register must be rewritten and then a value must be written to the TPnCCR1 register before the value of the 16-bit counter matches the value of the TPnCCR0 register. The values of the TPnCCR0 and TPnCCR1 registers are reloaded when the value of the TPnCCR0 register later matches the value of the 16-bit counter.
Whether the next reload timing is made valid or not is controlled by writing to the TPnCCR1 register. Therefore, write the same value to the TPnCCR1 register even when only the value of the TPnCCR0 register needs to be rewritten. Reload is invalid when only the value of the TPnCCR0 register is rewritten. To stop timer P, clear TPnCE to 0. The waveform of PWM is output from the TOPn1 pin. The TOPn0 pin produces a toggle output when the 16-bit counter matches the TPnCCR0 register.
In the PWM mode, the TPnCCR0 and TPnCCR1 registers are used only as compare registers. They cannot be used as capture registers.

Remark: n=0 to 2

Figure 6-24: Flowchart of Basic Operation in PWM Mode (1/2)

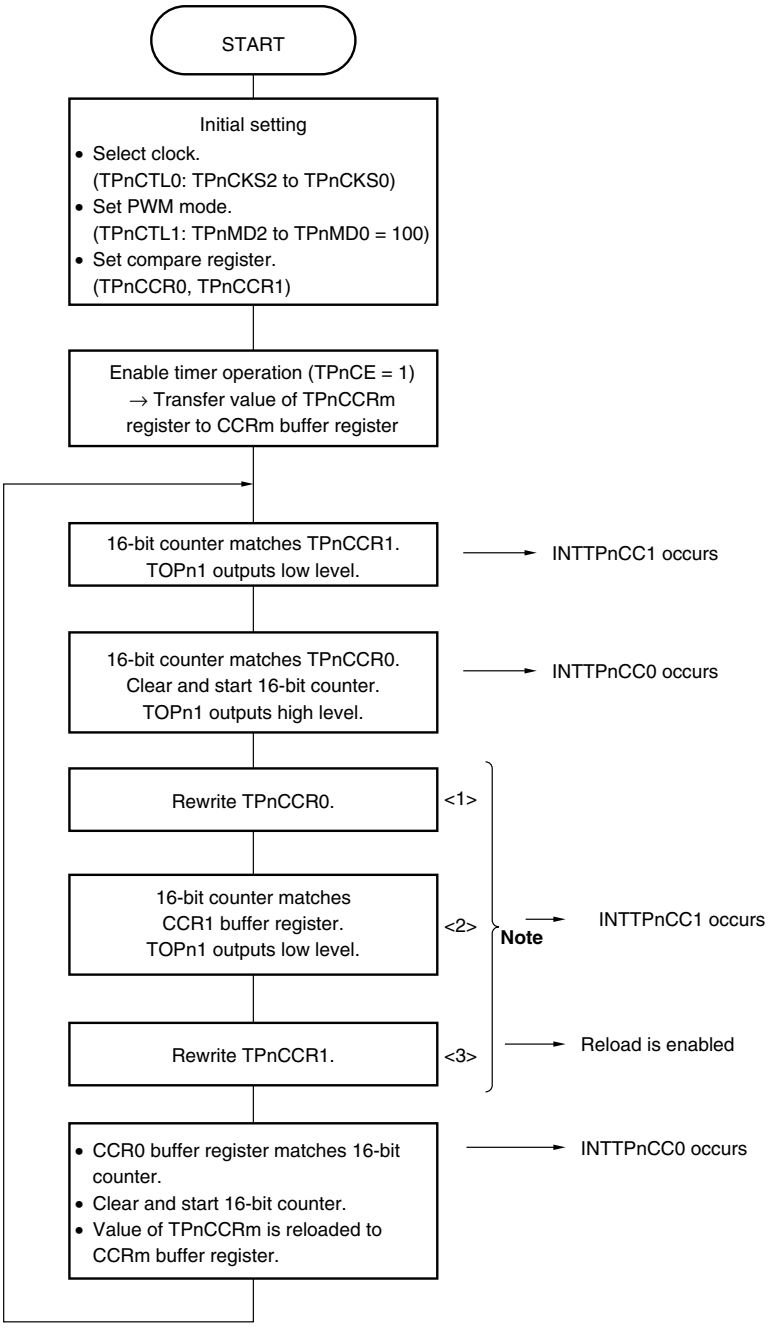
(a) When values of TPnCCR0 and TPnCCR1 registers are not rewritten during timer operation



Remark: n=0 to 2
m = 0, 1

Figure 6-24: Flowchart of Basic Operation in PWM Mode (2/2)

(b) When values of TPnCCR0 and TPnCCR1 registers are rewritten during timer operation

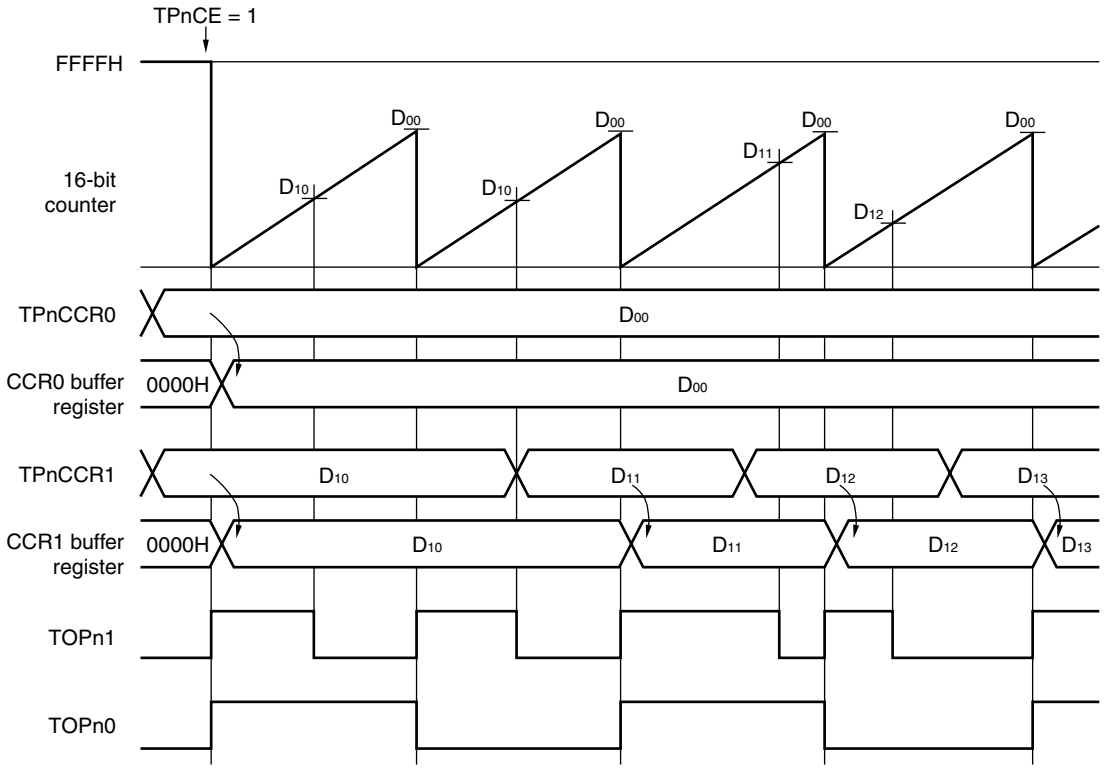


Note: The timing of <2> may differ depending on the rewrite timing of <1> and <3> and the value of TPnCCR1, but make sure that <3> comes after <1>.

Remark: n=0 to 2
m = 0, 1

Figure 6-25: Timing of Basic Operation in PWM Mode (1/2)

(a) When rewriting value of TPnCCR1 (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



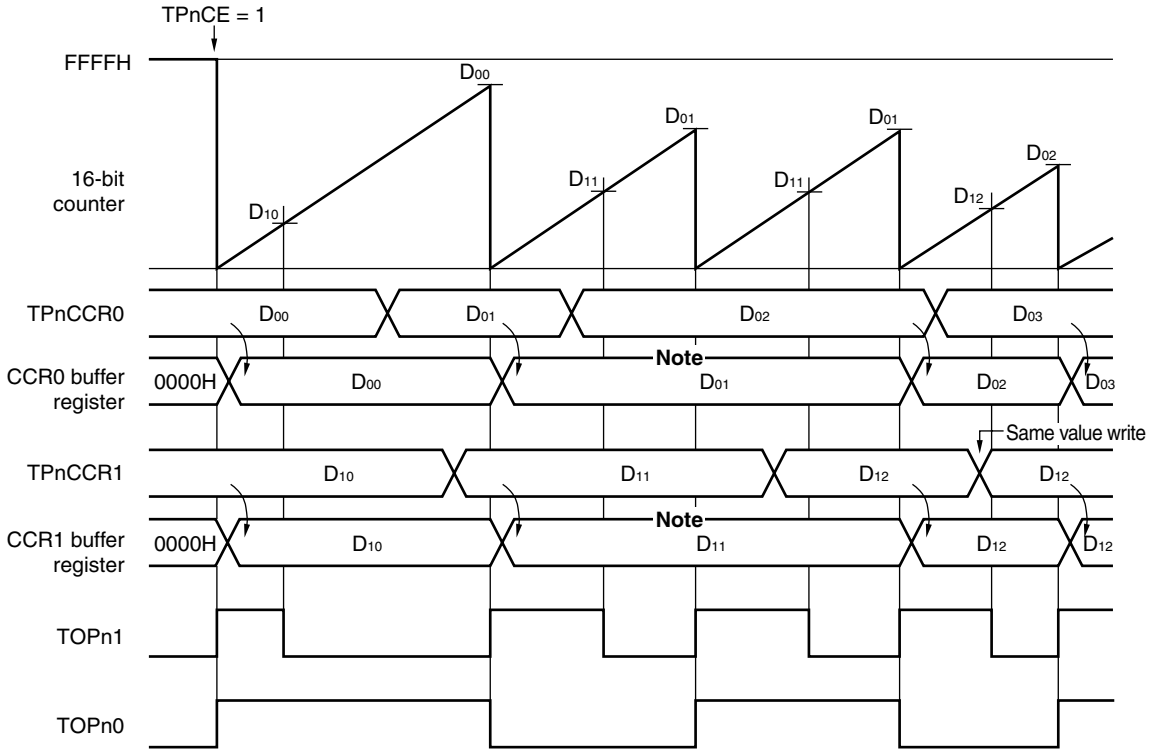
Remarks: 1. D_{00}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Duty of TOPn1 output = (Set value of TPnCCR1 register) / (Set value of TP0CCR0 register)
Cycle of TOPn1 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Toggle width of TOPn0 output = (Set value of TPnCCR0 register + 1) × (Count clock period)

3. n=0 to 2

Figure 6-25: Timing of Basic Operation in PWM Mode (2/2)

(b) When rewriting values of TPnCCR0 and TPnCCR1 (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Note: No value is reloaded because the TPnCCR1 register is not rewritten.

Remarks: **1.** D_{00}, D_{01}, D_{02}, D_{03}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Duty of TOPn1 output = (Set value of TPnCCR1 register) / (Set value of TP0CCR0 register)
Cycle of TOPn1 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Toggle width of TOPn0 output = (Set value of TPnCCR0 register + 1) × (Count clock cycle)

3. n=0 to 2

6.5.6 Free-running mode (TPnMD2 to TPnMD0 = 101)

In the free-running mode, the 16-bit counter free-runs, and the bit that selects the capture or compare register function is made valid by the setting of the TPnCCS1 and TPnCCS0 bits, so that an interval function and a capture function can be realized.
Setting of the TPnCCS1 and TPnCCS0 bits of the TPnOPT0 register is valid only in the free-running mode.

TPnCCS1	Operation
0	TPnCCR1 register is used as compare register.
1	TPnCCR1 register is used as capture register.

TPnCCS0	Operation
0	TPnCCR0 register is used as compare register.
1	TPnCCR0 register is used as capture register.

- When TPnCCR1 register is used as compare register

When the value of the 16-bit counter matches the value of the CCR1 buffer register in the free-running mode, an interrupt is generated (interval function).
Rewriting the value of the compare register is enabled during timer operation, and a value can be written to the register at any time (when the value to be compared is written to the register, it is synchronized with the internal clock and compared with the value of the 16-bit counter).
If timer output (TOPn1) is enabled, TOPn1 produces a toggle output when the value of the 16-bit counter matches the value of the CCR1 buffer register.

- When TPnCCR1 register is used as capture register

The value of the 16-bit counter is stored in the TPnCCR1 register when the edge of the TIPn1 pin is detected.

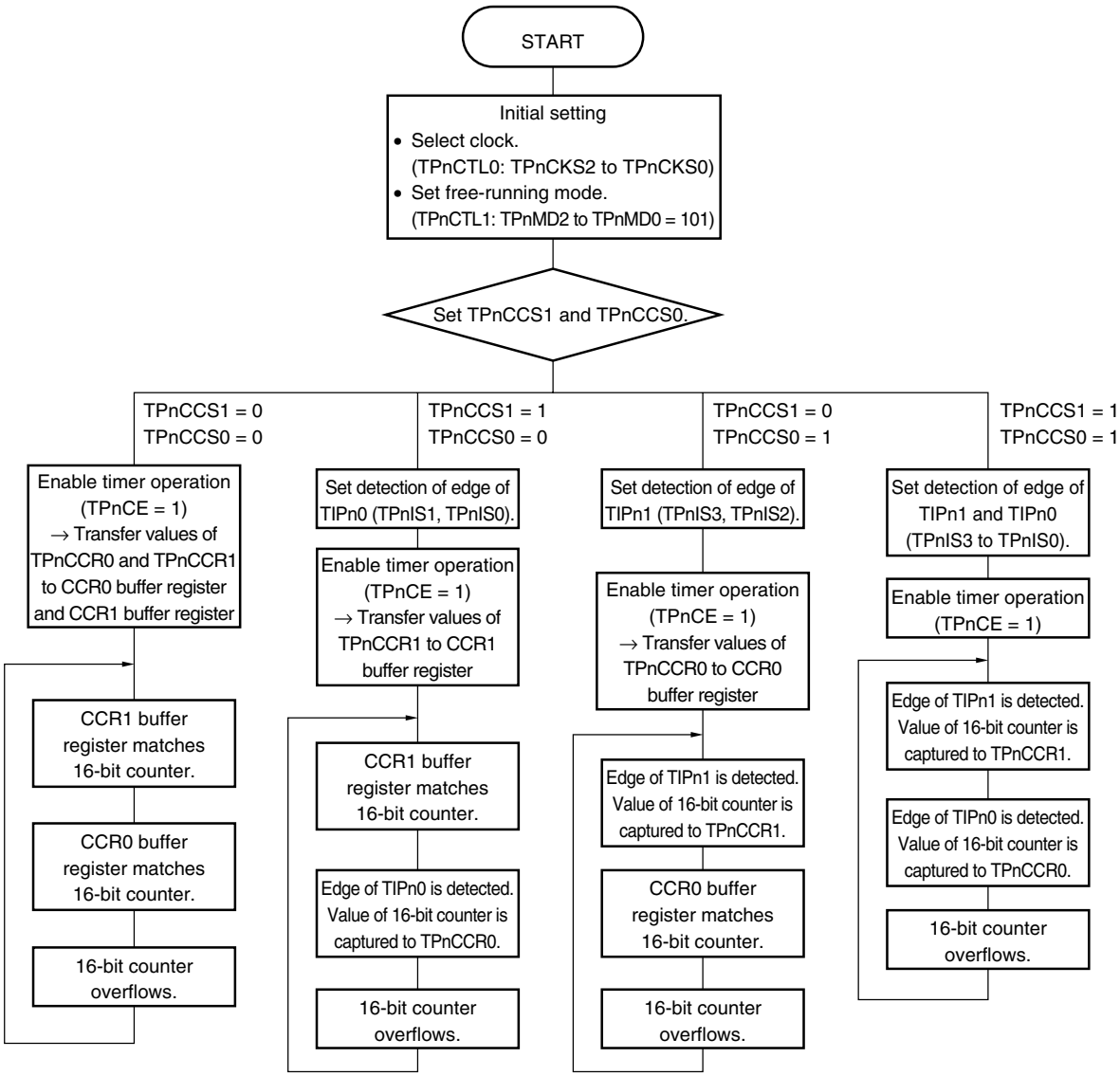
- When TPnCCR0 register is used as compare register

When the value of the 16-bit counter matches the value of the CCR0 buffer register in the free-running mode, an interrupt is generated (interval function).
Rewriting the value of the compare register is enabled during timer operation, and a value can be written to the register at any time (when the value to be compared is written to the register, it is synchronized with the internal clock and compared with the value of the 16-bit counter).
If timer output (TOPn0) is enabled, TOPn0 produces a toggle output when the value of the 16-bit counter matches the value of the CCR0 buffer register.

- When TPnCCR0 register is used as capture register

The value of the 16-bit counter is stored in the TPnCCR0 register when the edge of the TIPn0 pin is detected.

Figure 6-26: Flowchart of Basic Operation in Free-Running Mode



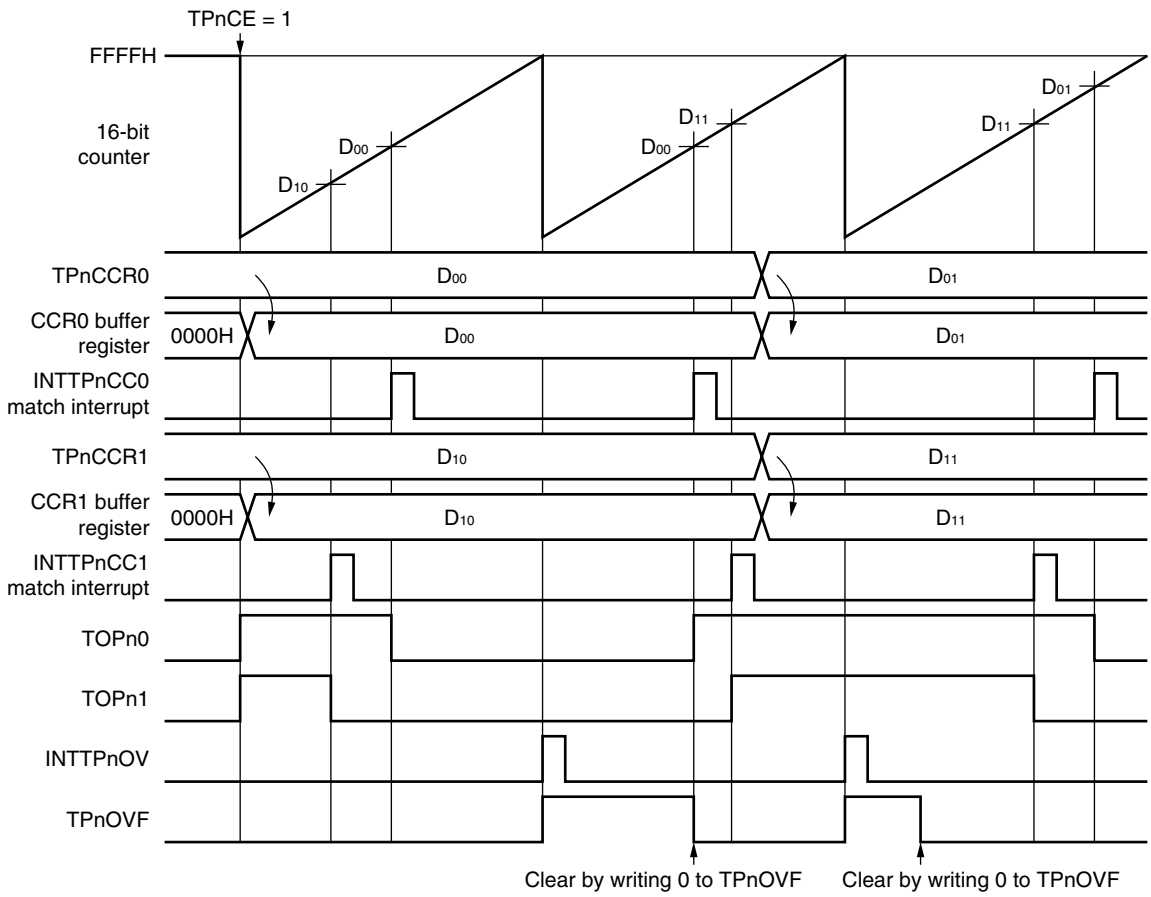
Remark: n=0 to 2

(1) When TPnCCS1 = 0 and TPnCCS0 = 0 (compare function)

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. If a value is written to the TPnCCR0 and TPnCCR1 registers in this mode, it is transferred to the CCR0 and CCR1 buffer registers (anytime write). Even if an one-shot pulse trigger is input in this mode, an one-shot pulse is not generated. If TPnOEm is set to 1, TOPnm produces a toggle output when the value of the 16-bit counter matches the value of the CCRm buffer register.

Remark: n=0 to 2
m = 0, 1

Figure 6-27: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 0) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Toggle width of TOPn0 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Toggle width of TOPn1 output = (Set value of TPnCCR1 register)

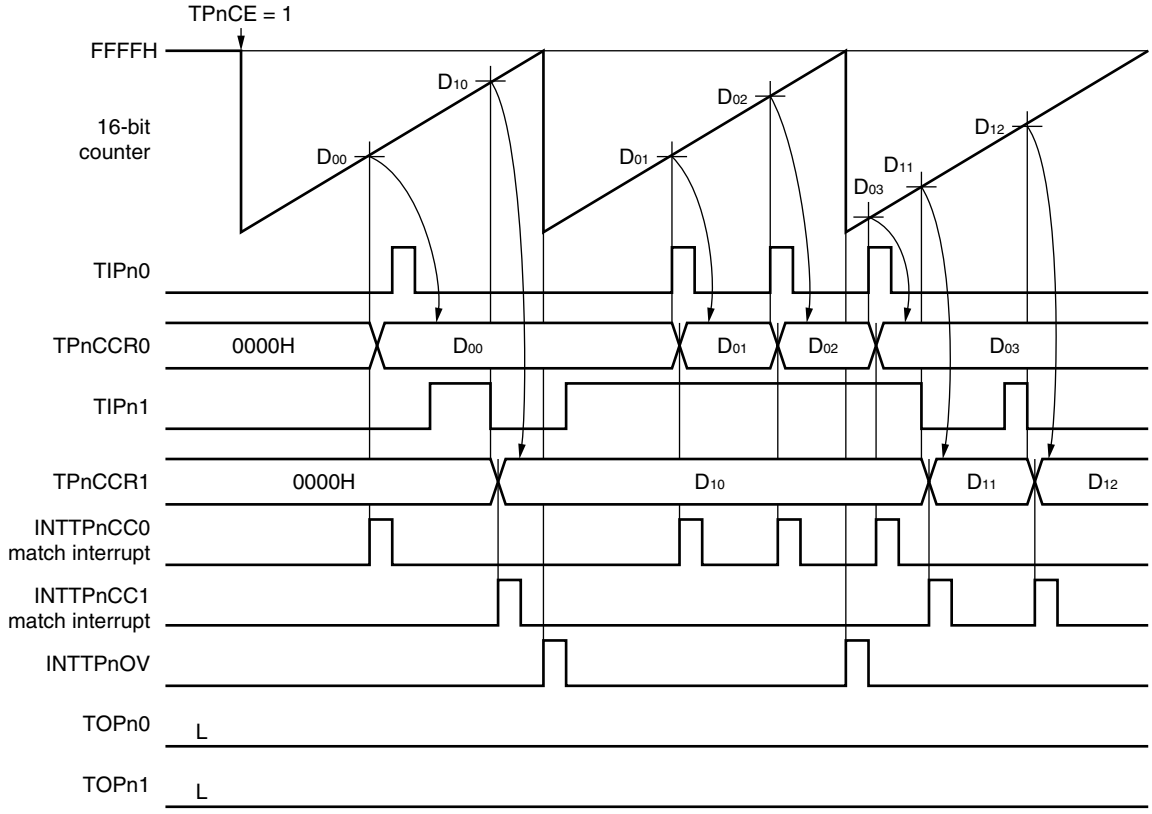
3. TOPnm output goes high when counting is started.

4. n=0 to 2
m = 0, 1

(2) When TPnCCS1 = 1 and TPnCCS0 = 1 (capture function)

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. The value captured by the capture trigger is written to the TPnCCR0 and TPnCCR1 registers.
Capturing close to an overflow (FFFFH) is judged using the overflow flag (TPnOVF).
However, if the interval of the capture trigger is such that the overflow occurs twice (two or more cycles of free-running), the TPnOVF flag cannot be used for judgment.

Figure 6-28: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 1) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Value captured to TPnCCR1 register (0000H to FFFFH)

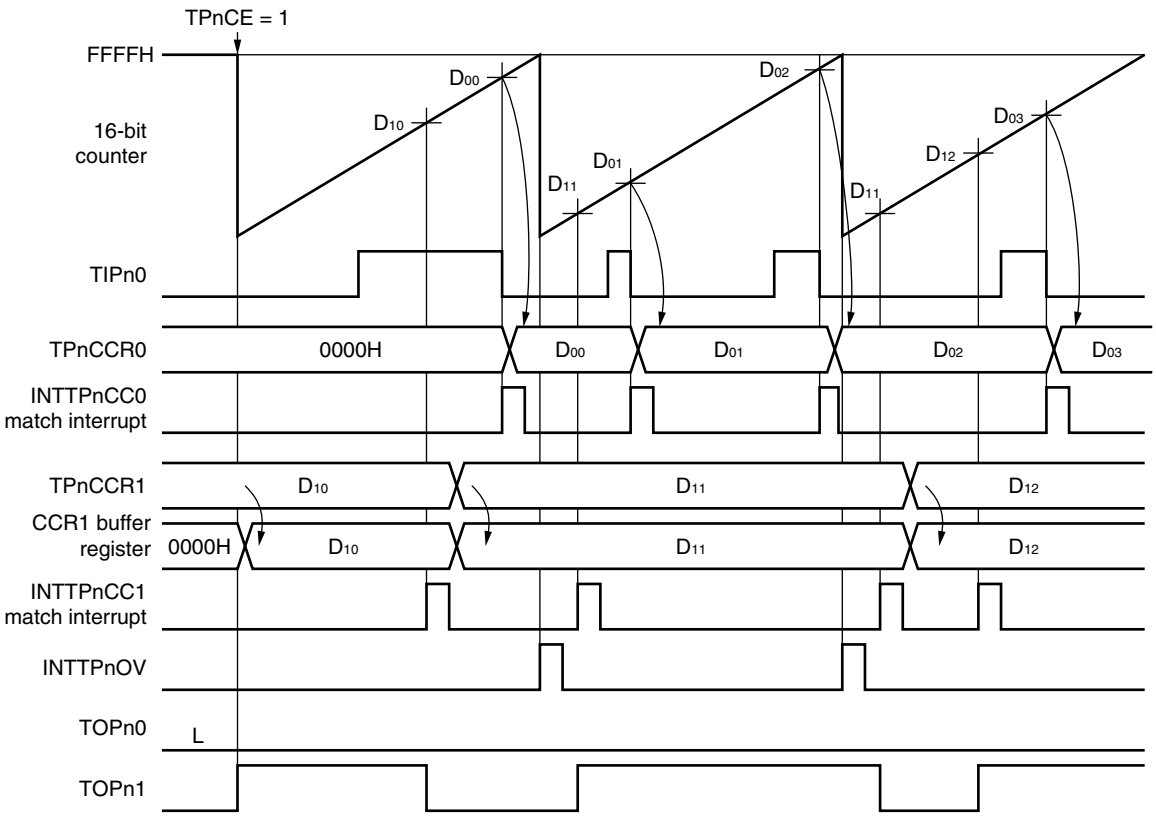
2. TIPn0: Rising edge is detected (TPnIS1, TPnIS0 = 01).
TIPn1: Falling edge is detected (TPnIS3, TPnIS2 = 10).

3. n=0 to 2

(3) When TPnCCS1 = 0 and TPnCCS0 = 1

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. The TPnCCR1 register is used as a compare register. As an interval function, an interrupt signal is output when the value of the 16-bit counter matches the set value of the TPnCCR1 register. If TPnOE1 is set to 1, TOPn1 produces a toggle output when the value of the 16-bit counter matches the set value of the TPnCCR1 register.

Figure 6-29: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 1) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Value captured to TPnCCR1 register (0000H to FFFFH)

2. TIPn0: Falling edge is detected (TPnIS1, TPnIS0 = 10).

3. n=0 to 2

(4) Overflow flag

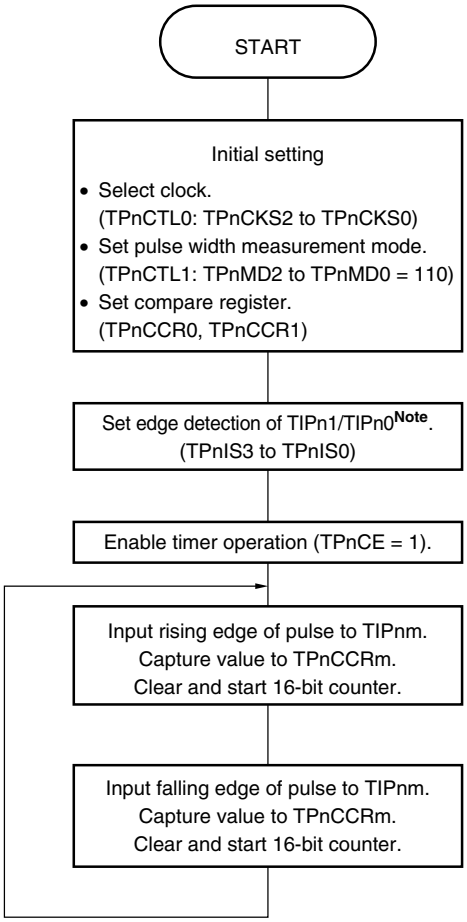
When the counter overflows from FFFFH to 0000H in the free-running mode, the overflow flag (TPnOVF) is set to 1, and an overflow interrupt (INTTPnOV) is generated.
After generation of the overflow interrupt (INTTPnOV), be sure to check if the overflow flag (TPnOVF) is set to 1.
The overflow flag is cleared by the CPU by writing 0 to it.

6.5.7 Pulse width measurement mode (TPnMD2 to TPnMD0 = 110)

In the pulse width measurement mode, free-running counting is performed. The value of the 16-bit counter is captured to capture register 0 (TPnCCR0) when both the rising and falling edges of the TIPn0 pin are detected, and the 16-bit counter is cleared to 0000H. In this way, the external input pulse width can be measured.
To measure a long pulse width that exceeds the overflow of the 16-bit counter, use the overflow flag for detection. A pulse width that causes overflow to occur twice or more cannot be measured. Adjust the operating frequency of the 16-bit counter. When the edge of the TIPn1 pin is detected, the value of the 16-bit counter is stored in capture register 1 (TPnCCR1), and the 16-bit counter is cleared.

Caution: In the pulse width measurement mode, select the internal clock (TPnEEE of the TPnCTL1 register = 0) as the count clock.

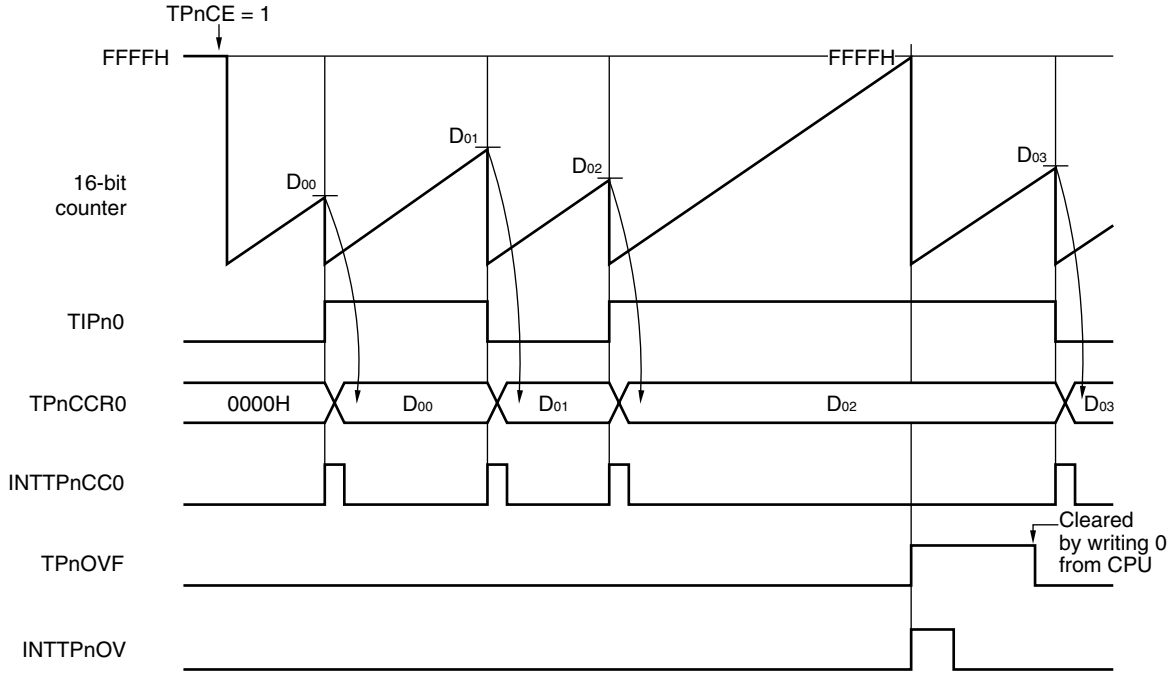
Figure 6-30: Flowchart of Basic Operation in Pulse Width Measurement Mode



Note: An external pulse can be input from either TIPn0 or TIPn1. Only one of them can be used. Specify that both the rising and falling edges are detected. Specify that the input edge of an external pulse input that is not used is not detected.

Remark: n=0 to 2
m = 0, 1

Figure 6-31: Timing of Basic Operation in Pulse Width Measurement Mode (TPnOE0 = 0, TPnOE1 = 0, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)

2. TIPn0: Both the rising and falling edges are detected (TPnIS1, TPnIS0 = 11).

3. n=0 to 2

6.6 Timer Synchronized Operation Function

Timer P has a timer synchronized operation function (tuned operation mode).

Table 6-3: Tuned Operation Mode of Timers

Master Timer	Slave Timer	
TMP0	TMP1	TMP2

Cautions: 1. The tuned operation mode is enabled or disabled by the TPnSYE bit of the TPnCTL1 register.

2. Set the tuned operation mode using the following procedure.

- **Set the TPnSYE bit of the TPnCTL1 register of the slave timer to enable the tuned operation.**
- **Set the TPnMD2 to TPnMD0 bits of the TPnCTL1 register of the slave timer to the free-running mode.**
- **Set the timer mode by using the TPnMD2 to TPnMD0 bits of the TPnCTL1 register.**
- **At this time, do not set the TP0SYE bit of the TP0CTL1of the master timer.**
- **Set the compare register value of the master and slave timers.**
- **Set the TPnCE bit of the TPnCTL0 register of the slave timer to enable operation on the internal operating clock.**
- **Set the TP0CE bit of the TP0CTL0 register of the master timer to enable operation on the internal operating clock.**

Figure 6-32: Tuned Operation Image (TMP0, TMP1, TMP2)

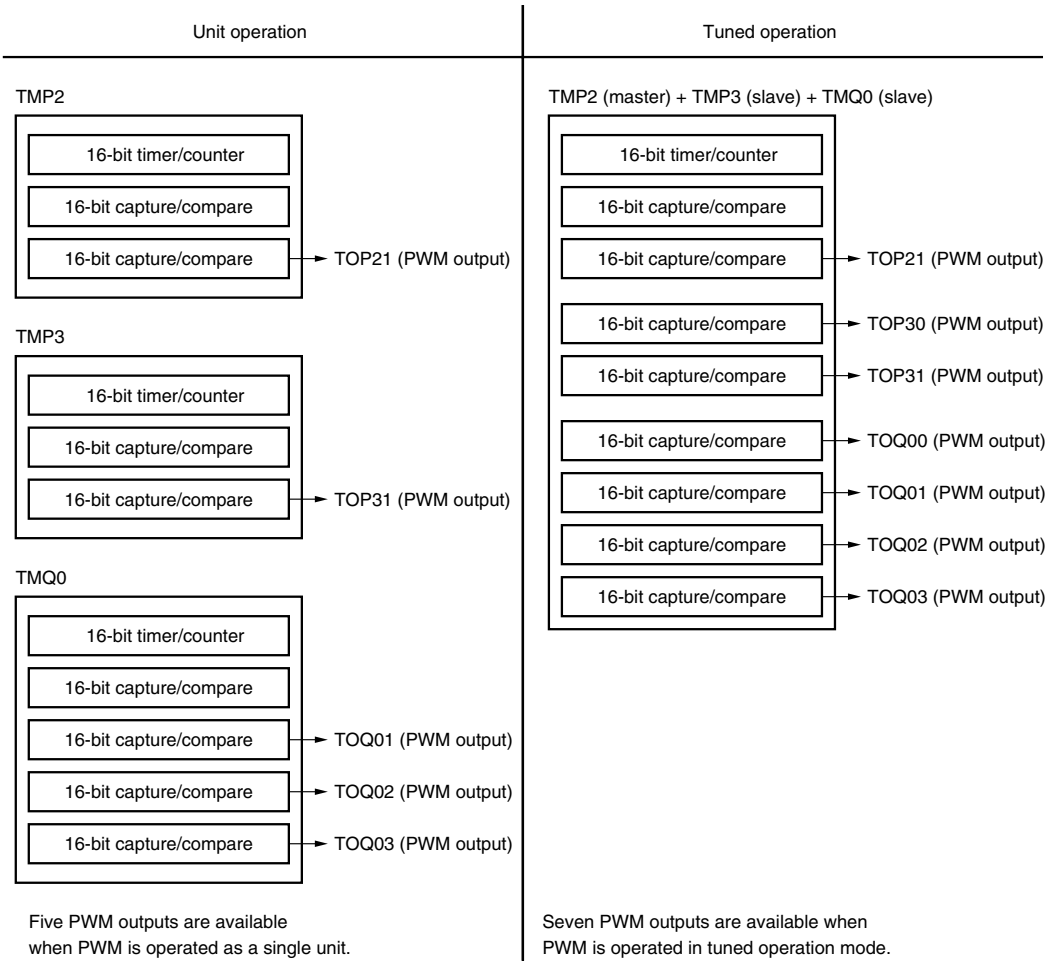
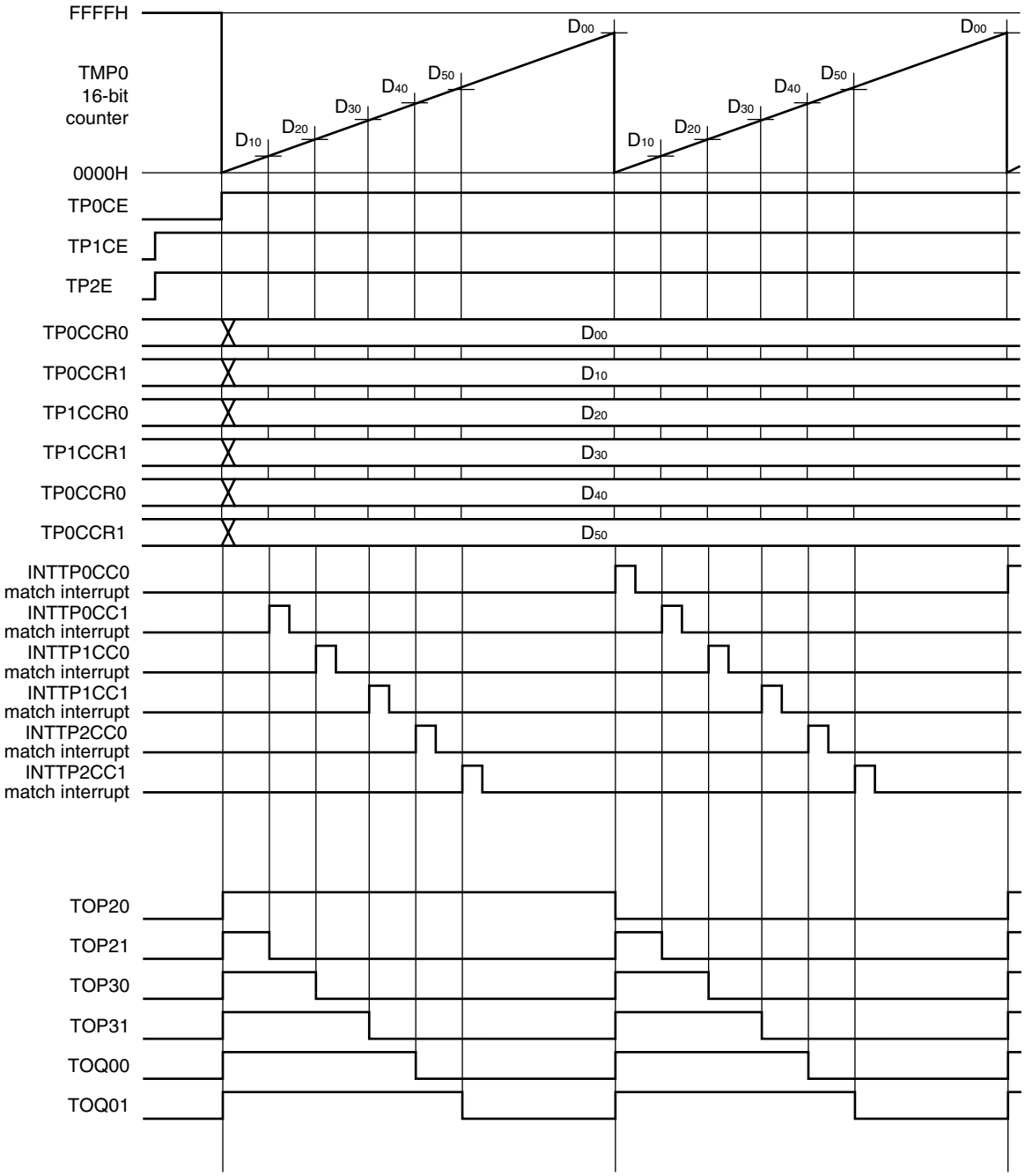


Figure 6-33: Basic Operation Timing of Tuned PWM Function (TMP0, TMP1, TMP2)



Chapter 7 Memory Access Control Function

7.1 SRAM, External ROM, External I/O Interface

7.1.1 Features

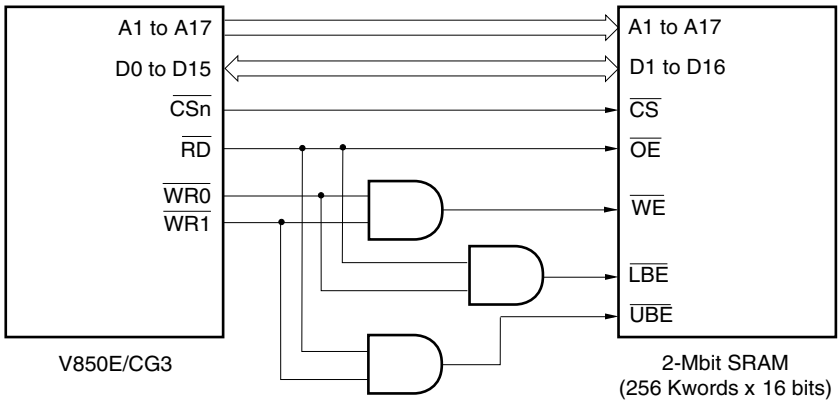
- Access to SRAM takes a minimum of 2 states.
- Up to 7 states of programmable data waits can be inserted through setting of the DWC0 and DWC1 registers.
- Up to 3 idle states can be inserted after the read/write cycle through setting of the BCC register.
- Up to 3 address set up wait states can be inserted through setting of the ASC register.

7.1.2 SRAM connections

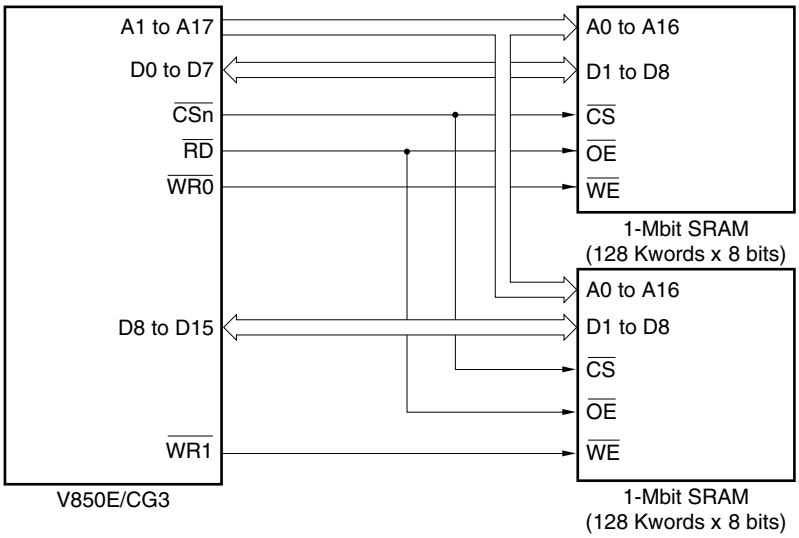
An example of connection to SRAM is shown below.

Figure 7-1: Example of Connection to SRAM

(a) When data bus width is 16 bits



(b) When data bus width is 8 bits

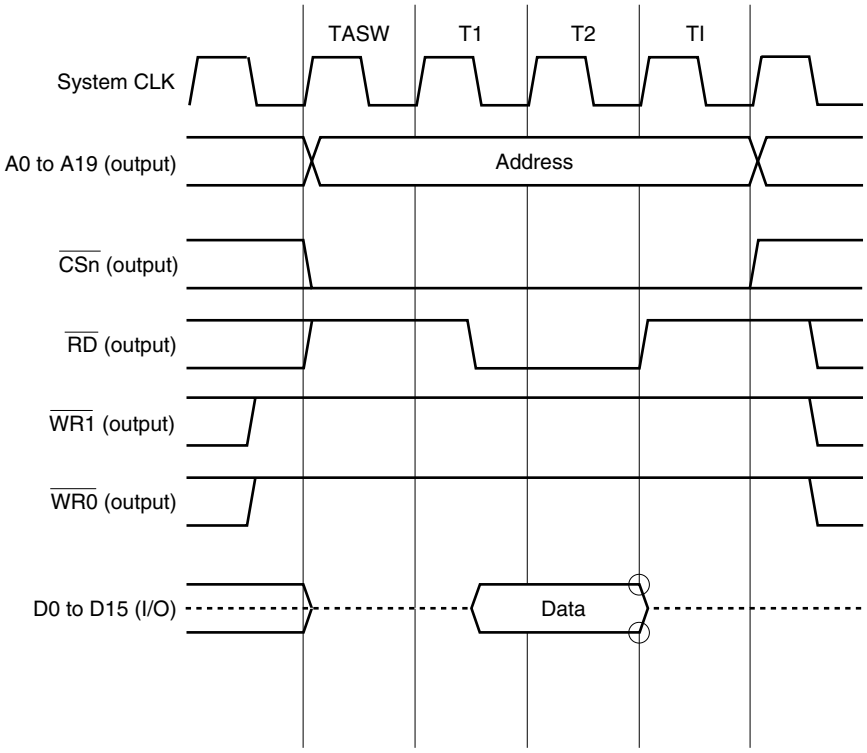


Remark: $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

7.1.3 SRAM, external ROM, external I/O access

Figure 7-2: SRAM, External ROM, External I/O Access Timing (1/4)

(a) During read (address setup wait, idle state insertion)



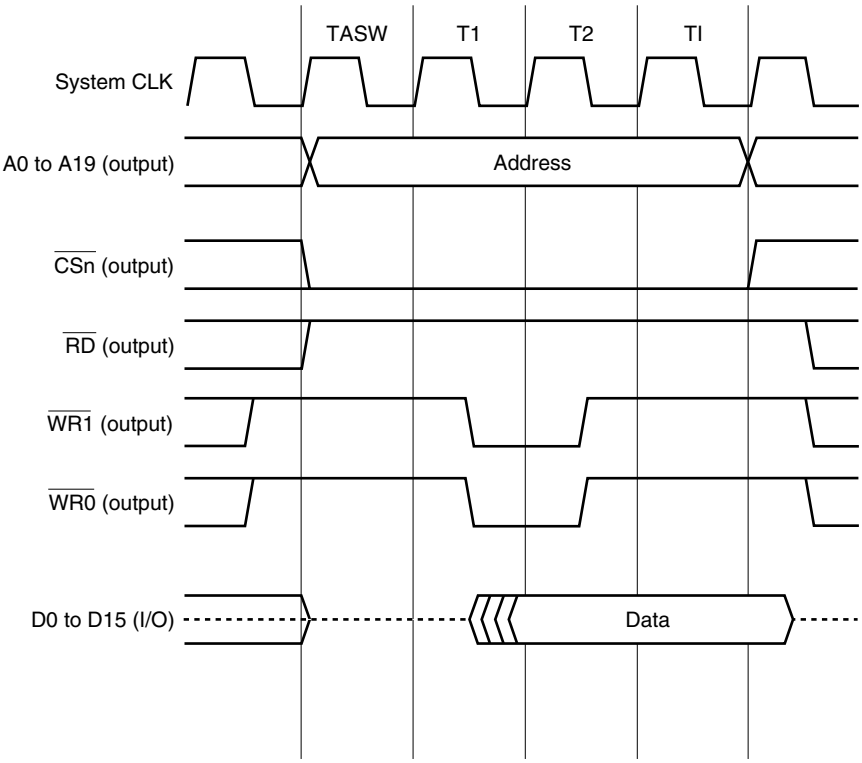
Remarks: **1.** The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-2: SRAM, External ROM, External I/O Access Timing (2/4)

(b) During write (address setup wait, idle state insertion)



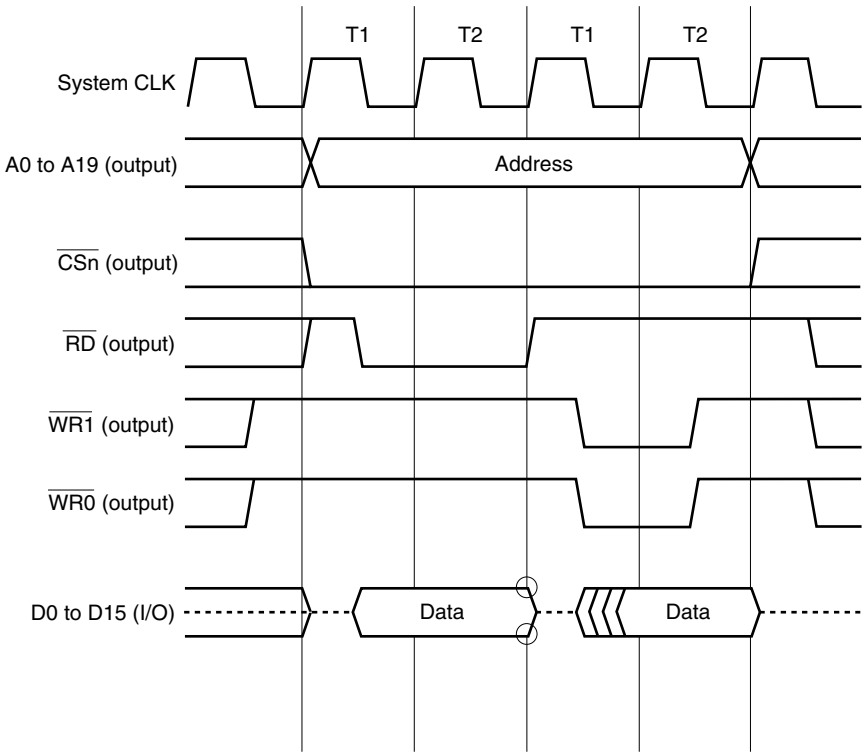
Remarks: 1. The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-2: SRAM, External ROM, External I/O Access Timing (3/4)

(c) When read → write operation



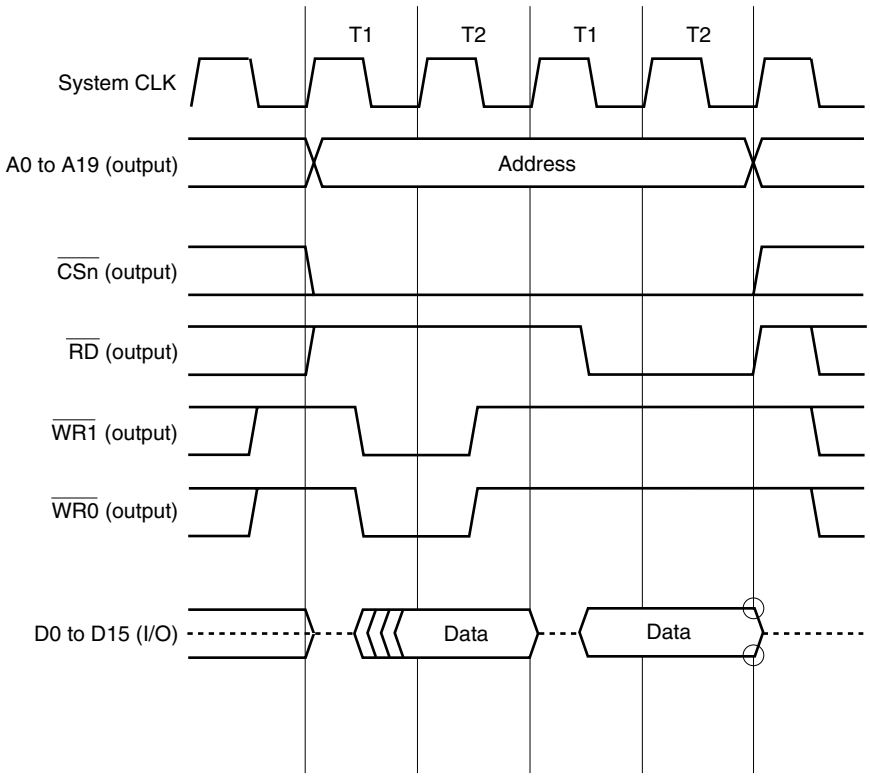
Remarks: **1.** The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-2: SRAM, External ROM, External I/O Access Timing (4/4)

(d) When write → read operation



Remarks: **1.** The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

7.2 Page ROM Controller (ROMC)

The page ROM controller (ROMC) is provided for access to ROM (page ROM) with the page access function.
Comparison of addresses with the immediately preceding bus cycle is carried out and wait control for normal access (off-page) and page access (on-page) is executed. This controller can handle page widths from 8 to 128 bytes.

7.2.1 Features

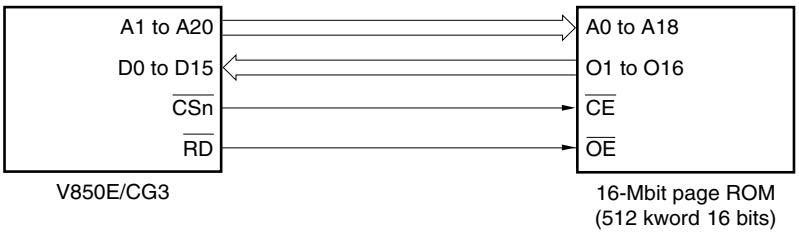
- Direct connection to 8-bit/16-bit page ROM supported
- In case of 16-bit bus width: 4/8/16/32/64 word page access supported
- In case of 8-bit bus width: 8/16/32/64/128 word page access supported
- Page ROM access a minimum of 2 states.
- On-page judgment function
- Addresses to be compared can be changed through setting of the PRC register.
- Up to 7 states of programmable data waits can be inserted during the on-page cycle through setting of the PRC register.
- Up to 7 states of programmable data wait can be inserted during the off-page cycle through setting of the DWC0 and DWC1 registers.
- Waits can be controlled with pin input.

7.2.2 Page ROM connections

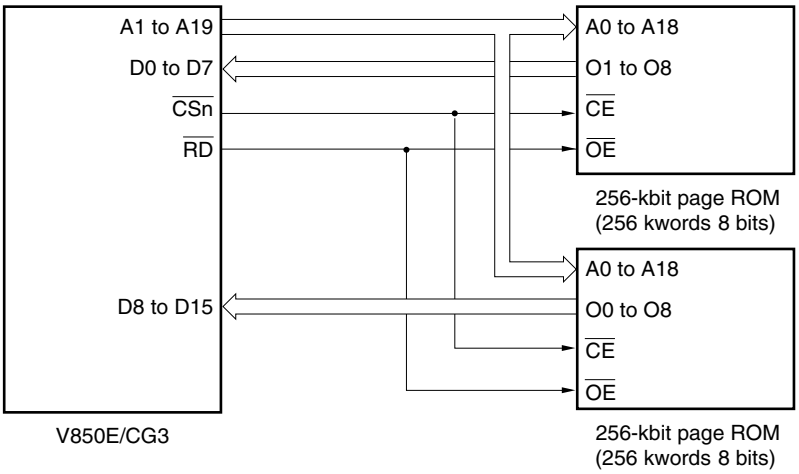
Examples of page ROM connections are shown below.

Figure 7-3: Example of Page ROM Connections

(a) In case of 16-bit data bus width



(b) In case of 8-bit data bus width



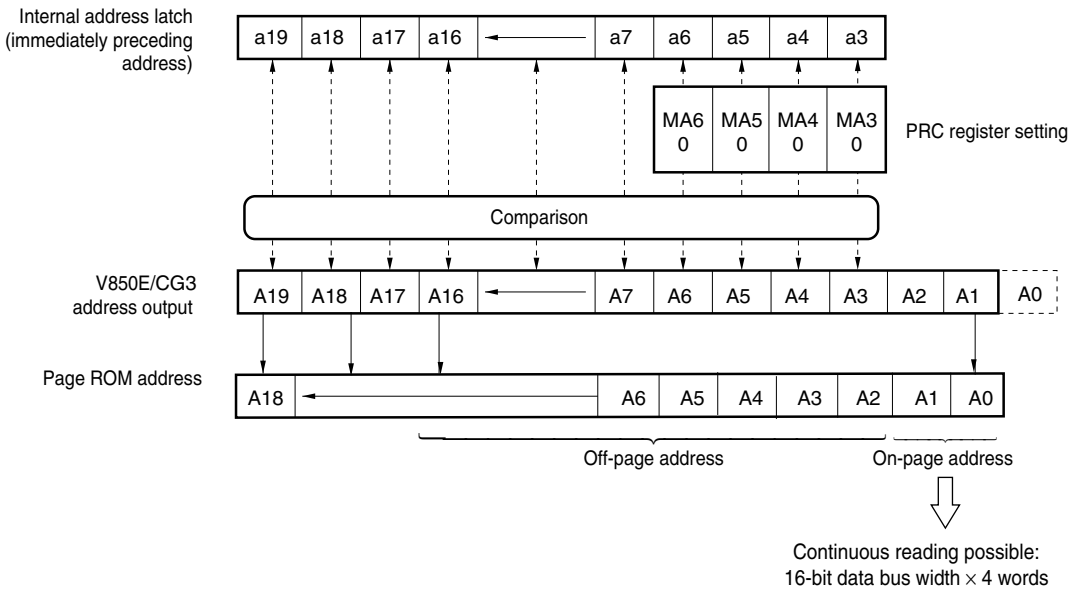
Remark: $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

7.2.3 On-page/off-page judgment

Whether a page ROM cycle is on-page or off-page is judged by latching the address of the previous cycle and comparing it with the address of the current cycle.
Through the page ROM configuration register (PRC), according to the configuration of the connected page ROM and the number of continuously readable bits, one of the addresses (A3 to A6) is set as the masking address (no comparison is made).

Figure 7-4: On-Page/Off-Page Judgment during Page ROM Connection (1/2)

(a) In case of 16-Mbit (1 M × 16 bits) page ROM (4-word page access)



(b) In case of 16-Mbit (1 M × 16 bits) page ROM (8-word page access)

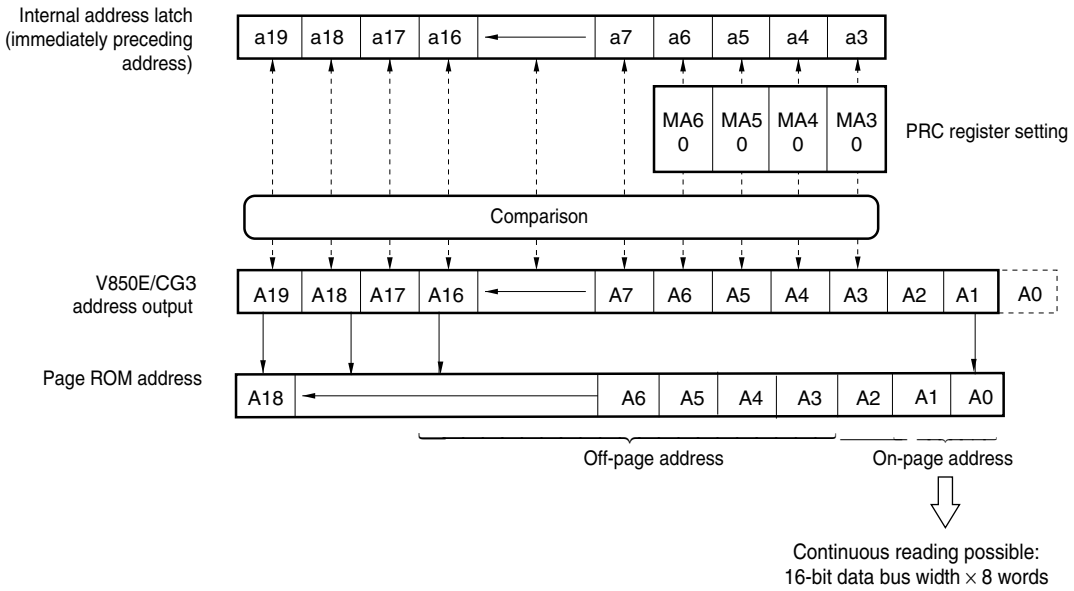
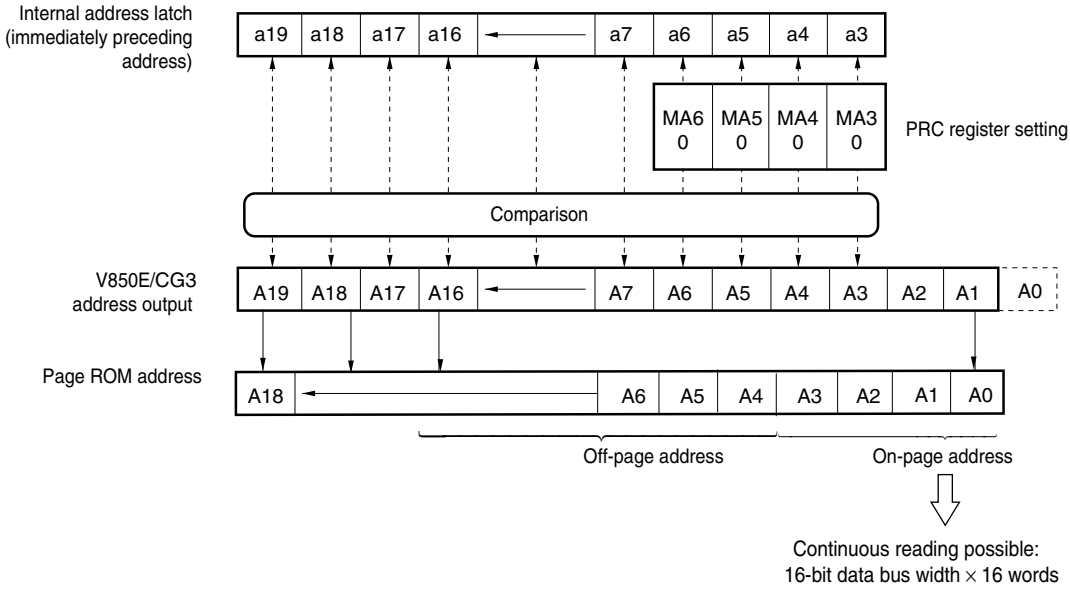


Figure 7-4: On-Page/Off-Page Judgment during Page ROM Connection (2/2)

(c) In case of 32-Mbit (2 M × 16 bits) page ROM (16-word page access)



7.2.4 Page ROM configuration register (PRC)

This register specifies whether page ROM on-page access is enabled or disabled. If on-page access is enabled, the masking address (no comparison is made) out of the addresses (A3 to A6) corresponding to the configuration of the page ROM being connected to and the number of bits that can be read continuously, as well as the number of waits corresponding to the internal system clock, are set.
This register can be read/written in 16-bit units.

Figure 7-5: Page ROM Configuration Register (PRC)

	15	14	13	12	11	10	9	8	Address	R/W	Initial value
TPRC		PRW2	PRW1	PRW0					FFFFF49AH	R/W	7000H
	7	6	5	4	3	2	1	0			
					MA6	MA5	MA4	MA3			

Bit Position	Bit Name	Function
14 to 12	PRW2 to PRW0	Page-ROM On-page Wait Control Sets the number of waits corresponding to the internal system clock. The number of waits set by this bit are inserted only when on-page. When off-page, the waits set by registers DWC0 and DWC1 are inserted.
3 to 0	MA6 to MA3	Mask Address Each respective address (A6 to A3) corresponding to MA6 to MA3 is masked (masked by 1). The masked address is not subject to comparison during on/off-page judgment. It is set according to the number of continuously readable bits.

PRW2	PRW1	PRW0	Number of Inserted Wait Cycles
0	0	0	0
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

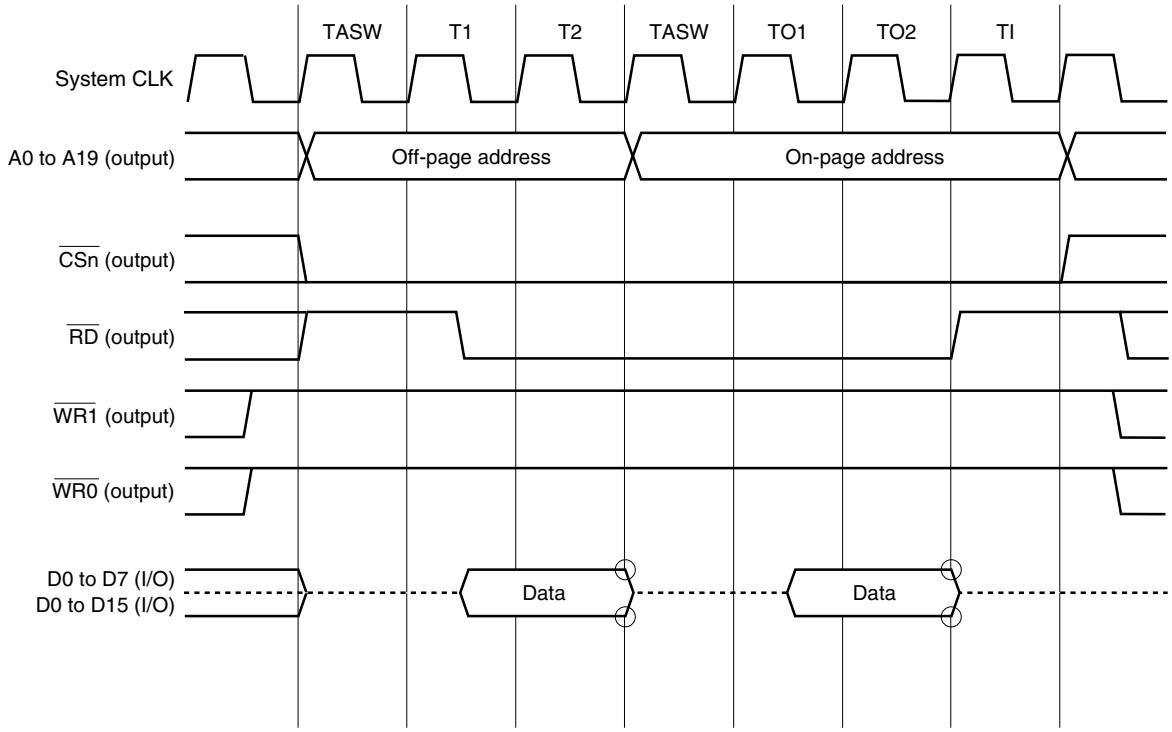
MA6	MA5	MA4	MA3	Number of Continuously Readable Bits
0	0	0	0	4 words × 16 bits (8 words × 8 bits)
0	0	0	1	8 words × 16 bits (16 words × 8 bits)
0	0	1	1	16 words × 16 bits (32 words × 8 bits)
0	1	1	1	32 words × 16 bits (64 words × 8 bits)
1	1	1	1	64 words × 16 bits (128 words × 8 bits)

Caution: Write to the PRC register after reset, and then do not change the set value. Also, do not access an external memory area other than that for this initialization routine until initial setting of the PRC register is finished. However, it is possible to access external memory areas whose initialization has been finished.

7.2.5 Page ROM access

Figure 7-6: Page ROM Access Timing (1/4)

(a) During read (when half word/word access with 8-bit bus width or when word access with 16-bit bus width)



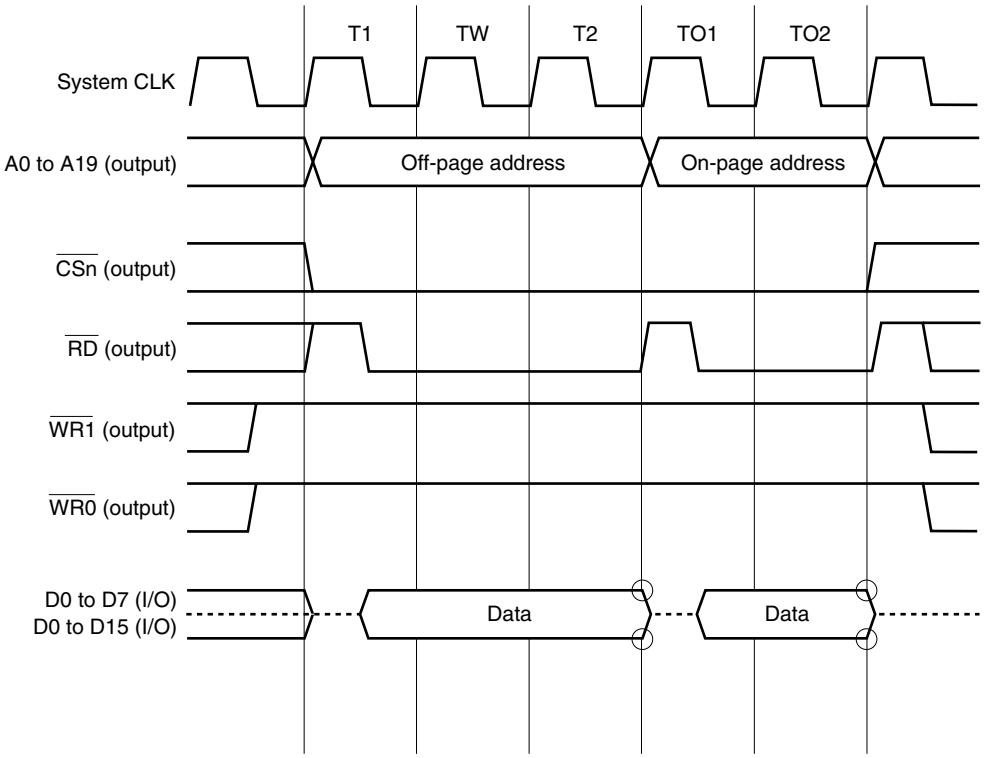
Remarks: **1.** The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-6: Page ROM Access Timing (2/4)

(b) During read (when byte access with 8-bit bus width or when byte/half word access with 16-bit bus width)



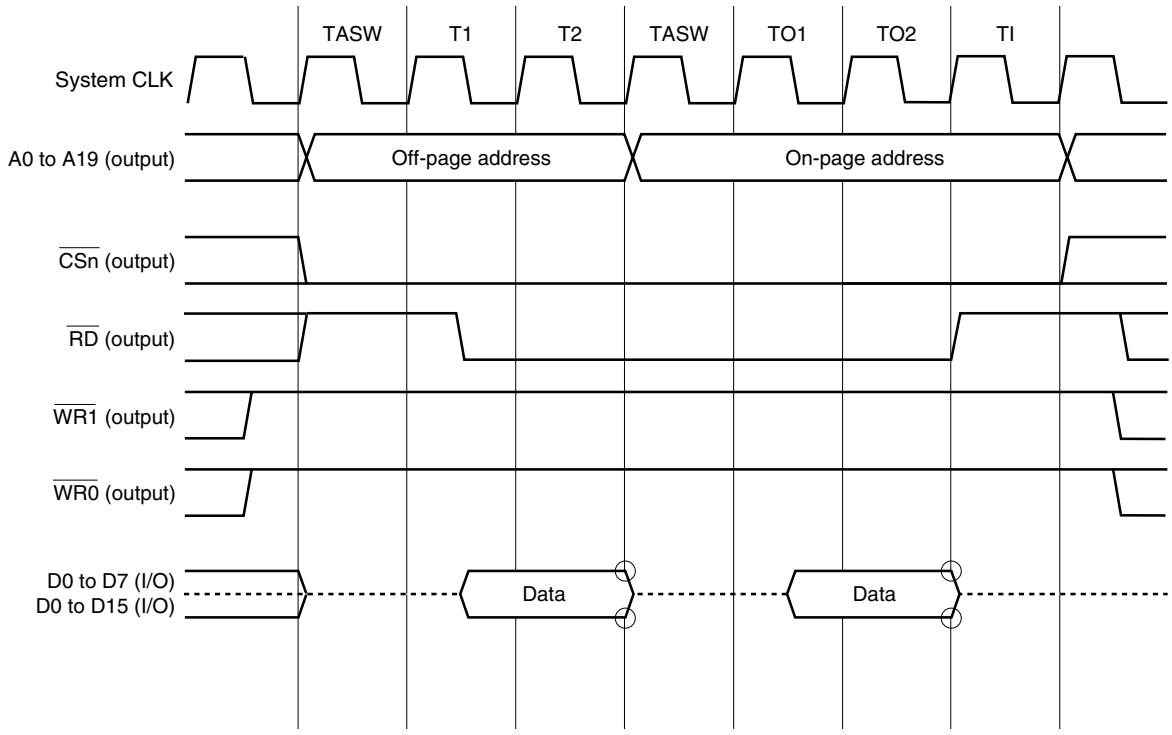
Remarks: 1. The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-6: Page ROM Access Timing (3/4)

(c) During read (address setup wait, idle state insertion)
(when half word/word access with 8-bit bus width or when word access with 16-bit bus width)



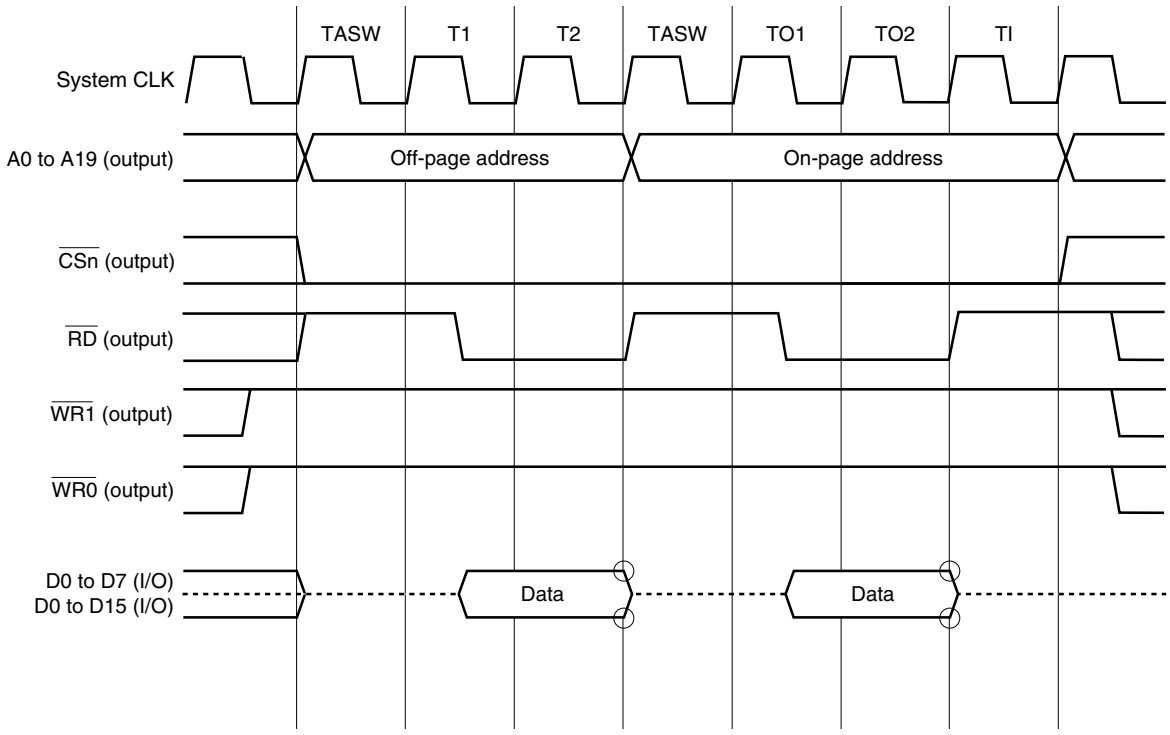
Remarks: 1. The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

Figure 7-6: Page ROM Access Timing (4/4)

(d) During read (address setup wait, idle state insertion)
(when byte access with 8-bit bus width or when byte/half word access with 16-bit bus width)



Remarks: **1.** The circles m indicate the sampling timing.

2. The broken line indicates the high-impedance state.

3. $\overline{CSn}$ = $\overline{CS0}$, $\overline{CS3}$ and $\overline{CS4}$

[MEMO]

Chapter 8 Interrupt/Exception Processing Function

The CarGate+ is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 64 maskable and one non-maskable 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. Generally, an exception takes precedence over an interrupt.
The can process interrupt requests 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).
Eight levels of software-programmable priorities can be specified for each interrupt request. Interrupt servicing starts after no fewer than 11 system clocks (344 ns (@ 32 MHz)) following the generation of an interrupt request.

8.1 Features

- Interrupts
  - Non-maskable interrupts: 1 sources
  - Maskable interrupts: 63 sources
  - 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 traps: 1 source (illegal opcode exception)

Interrupt/exception sources are listed in Table 8-1, “Interrupt Vector Table,” on page 294.

Table 8-1: Interrupt Vector Table (1/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Generating Source	Generating Unit				
Reset	Interrupt	RESET	–	$\overline{\text{RESET}}$ input	Pin	–	0000H	00H	undef.
Non-maskable	Interrupt	NMI0	–	P60 NMI Input	Port Module	–	0010H	10H	nextPC
SW exception	Exception	TRAP0n	–	TRAP0 instruction(n=0...F)	–	–	004nH	40H	nextPC
	Exception	TRAP1n	–	TRAP1 instruction(n=0...F)	–	–	005nH	50H	nextPC
Exception	Exception	ILGOP	–	Illegal opcode	–	–	0060H	60H	nextPC
Maskable	Interrupt	INTP0	P0IC	P01	Port Module	0	80H	80H	Next PC
	Interrupt	INTP1	P1IC	P30	Port Module	1	90H	90H	Next PC
	Interrupt	INTP2	P2IC	P32	Port Module	2	A0H	A0H	Next PC
	Interrupt	INTP3	P3IC	P20	Port Module	3	B0H	B0H	Next PC
	Interrupt	INTP4	P4IC	P21	Port Module	4	C0H	C0H	Next PC
	Interrupt	INTP5	P5IC	P22	Port Module	5	D0H	D0H	Next PC
	Interrupt	INTP6	P6IC	P23	Port Module	6	E0H	E0H	Next PC
	Interrupt	INTP7	P7IC	P24	Port Module	7	F0H	F0H	Next PC
	Interrupt	INTP8	P8IC	P25	Port Module	8	100H	100H	Next PC
	Interrupt	INTP9	P9IC	P26	Port Module	9	110H	110H	Next PC
	Interrupt	INTP10	P10IC	P27	Port Module	10	120H	120H	Next PC
	Interrupt	INTTP0OV	TP0OVIC	TMP0 overflow	TMP0	11	130H	130H	Next PC
	Interrupt	INTTP0CC0	TP0CC0IC	TMP0 capture compare channel 0	TMP0	12	140H	140H	Next PC
	Interrupt	INTTP0CC1	TP0CC1IC	TMP0 capture compare channel 1	TMP0	13	150H	150H	Next PC
	Interrupt	INTTP1OV	TP1OVIC	TMP1 overflow	TMP1	14	160H	160H	Next PC
	Interrupt	INTTP1CC0	TP1CC0IC	TMP1 capture compare channel 0	TMP1	15	170H	170H	Next PC
	Interrupt	INTTP1CC1	TP1CC1IC	TMP1 capture compare channel 1	TMP1	16	180H	180H	Next PC
	Interrupt	INTTP2OV	TP2OVIC	TMP2 overflow	TMP2	17	190H	190H	Next PC
	Interrupt	INTTP2CC0	TP2CC0IC	TMP2 capture compare channel 0	TMP2	18	1A0H	1A0H	Next PC
	Interrupt	INTTP2CC1	TP2CC1IC	TMP2 capture compare channel 1	TMP2	19	1B0H	1B0H	Next PC
	Interrupt	INTBRG0	BRG0IC	BRG0 overflow	BRG0	20	1C0H	1C0H	Next PC
	Interrupt	INTAD	ADIC	A/D conversion end	A/D	21	1D0H	1D0H	Next PC
	Interrupt	INTC0ERR	C0ERRIC	CAN0 error	CAN0	22	1E0H	1E0H	Next PC
	Interrupt	INTC0WUP	C0WUPIC	CAN0 wake up	CAN0	23	1F0H	1F0H	Next PC
	Interrupt	INTC0REC	C0RECIC	CAN0 receive	CAN0	24	200H	200H	Next PC
	Interrupt	INTC0TRX	C0TRXIC	CAN0 transmit	CAN0	25	210H	210H	Next PC
	Interrupt	INTC1ERR	C1ERRIC	CAN1 error	CAN1	26	220H	220H	Next PC
	Interrupt	INTC1WUP	C1WUPIC	CAN1 wake up	CAN1	27	230H	230H	Next PC
	Interrupt	INTC1REC	C1RECIC	CAN1 receive	CAN1	28	240H	240H	Next PC
	Interrupt	INTC1TRX	C1TRXIC	CAN1 transmit	CAN1	29	250H	250H	Next PC
	Interrupt	INTC2ERR	C2ERRIC	CAN2 error	CAN2	30	260H	260H	Next PC
	Interrupt	INTC2WUP	C2WUPIC	CAN2 wake up	CAN2	31	270H	270H	Next PC
	Interrupt	INTC2REC	C2RECIC	CAN2 receive	CAN2	32	280H	280H	Next PC
	Interrupt	INTC2TRX	C2TRXIC	CAN2 transmit	CAN2	33	290H	290H	Next PC
	Interrupt	INTC3ERR	C3ERRIC	CAN3 error	CAN3	34	2A0H	2A0H	Next PC
	Interrupt	INTC3WUP	C3WUPIC	CAN3 wake up	CAN3	35	2B0H	2B0H	Next PC

Table 8-1: Interrupt Vector Table (2/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Generating Source	Generating Unit				
Maskable	Interrupt	INTC3REC	C3RECIC	CAN3 receive	CAN3	36	2C0H	2C0H	Next PC
	Interrupt	INTC3TRX	C3TRXIC	CAN3 transmit	CAN3	37	2D0H	2D0H	Next PC
	Interrupt	INTC4ERR	C4ERRIC	CAN4 error	CAN4	38	2E0H	2E0H	Next PC
	Interrupt	INTC4WUP	C4WUPIC	CAN4 wake up	CAN4	39	2F0H	2F0H	Next PC
	Interrupt	INTC4REC	C4RECIC	CAN4 receive	CAN4	40	300H	300H	Next PC
	Interrupt	INTC4TRX	C4TRXIC	CAN4 transmit	CAN4	41	310H	310H	Next PC
	Interrupt	INTCB0T	CB0TIC	CSIB0 transmit	CSIB0	42	320H	320H	Next PC
	Interrupt	INTCB0RD	CB0RDIC	CSIB0 receive	CSIB0	43	330H	330H	Next PC
	Interrupt	INTCB0RE	CB0REIC	CSIB0 receive error	CSIB0	44	340H	340H	Next PC
	Interrupt	INTCB1T	CB1TIC	CSIB1 transmit	CSIB1	45	350H	350H	Next PC
	Interrupt	INTCB1RD	CB1RDIC	CSIB1 receive	CSIB1	46	360H	360H	Next PC
	Interrupt	INTCB1RE	CB1REIC	CSIB1 receive error	CSIB1	47	370H	370H	Next PC
	Interrupt	INTCB2T	CB2TIC	CSIB2 transmit	CSIB2	48	380H	380H	Next PC
	Interrupt	INTCB2RD	CB2RDIC	CSIB2 receive	CSIB2	49	390H	390H	Next PC
	Interrupt	INTCB2RE	CB2REIC	CSIB2 receive error	CSIB2	50	3A0H	3A0H	Next PC
	Interrupt	INTUA0RE	UA0REIC	UARTA0 receive error	UARTA0	51	3B0H	3B0H	Next PC
	Interrupt	INTUA0RD	UA0RDIC	UARTA0 receive	UARTA0	52	3C0H	3C0H	Next PC
	Interrupt	INTUA0T	UA0TIC	UARTA0 transmit	UARTA0	53	3D0H	3D0H	Next PC
	Interrupt	INTUA1RE	UA1REIC	UARTA1 receive error	UARTA1	54	3E0H	3E0H	Next PC
	Interrupt	INTUA1RD	UA1RDIC	UARTA1 receive	UARTA1	55	3F0H	3F0H	Next PC
	Interrupt	INTUA1T	UA1TIC	UARTA1 transmit	UARTA1	56	400H	400H	Next PC
	Interrupt	INTIIC0	IIC0IC	IIC0 interrupt	IIC0	57	410H	410H	Next PC
	Interrupt	INTCB0R	CB0IC	CSIB0 receive & error interrupt	CSIB0	59	430H	430H	Next PC
	Interrupt	INTCB1R	CB1IC	CSIB1 receive & error interrupt	CSIB1	60	440H	440H	Next PC
	Interrupt	INTCB2R	CB2IC	CSIB2 receive & error interrupt	CSIB2	61	450H	450H	Next PC
	Interrupt	INTUA0R	UA0IC	UARTA0 receive & error interrupt	UARTA0	62	460H	460H	Next PC
	Interrupt	INTUA1R	UA1IC	UARTA1 receive & error interrupt	UARTA1	63	470H	470H	Next PC

Remarks: **1.** Default priority: The priority order when two or more maskable interrupt requests are generated at the same time. The highest priority is 0.

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

3. nextPC: The PC value that starts the processing following interrupt/exception processing.

4. The execution address of the illegal instruction when an illegal opcode exception occurs is calculated by (Restored PC – 4).

8.2 Non-Maskable Interrupts

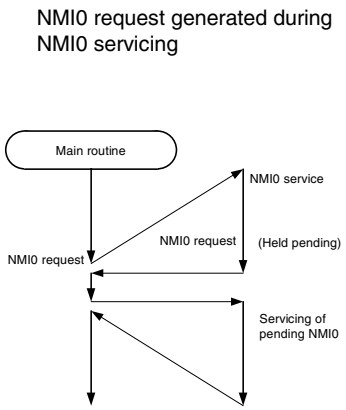
A non-maskable interrupt request is acknowledged unconditionally, even when interrupts are in the interrupt disabled (DI) status.
Non-maskable interrupt of μPD70F3178 is available for the following request:

- NMI pin input

When the valid edge specified by the ESN0 bit of the Interrupt mode register 3 (INTM3) is detected on the NMI pin, the interrupt occurs.

Caution: If PSW.NP is cleared to 0 by the LDSR instruction during non-maskable interrupt servicing, a NMI0 interrupt afterwards cannot be acknowledged correctly.

Figure 8-1: NMI0 request during NMI0 serving



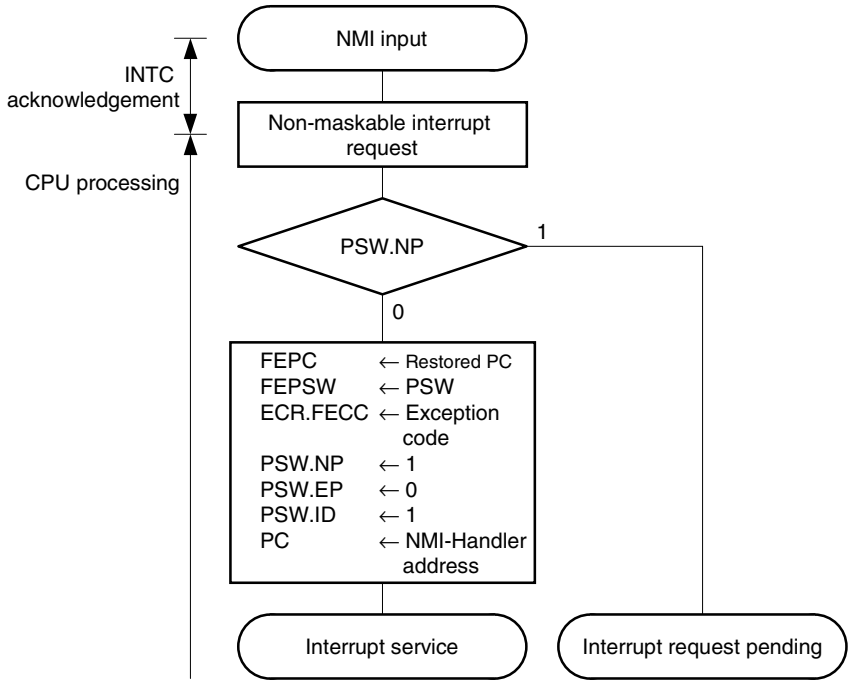
8.2.1 Operation

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

<6> Saves the restored PC to FEPC.
<7> Saves the current PSW to FEPSW.
<8> Writes exception code 0010H to the higher halfword (FECC) of ECR.
<9> Sets the NP and ID bits of the PSW and clears the EP bit.
<10> Sets the handler address corresponding to the non-maskable interrupt to the PC, and transfers control.

The processing configuration of a non-maskable interrupt is shown in Figure 8-2.

Figure 8-2: Processing Configuration of Non-Maskable Interrupt



8.2.2 Restore

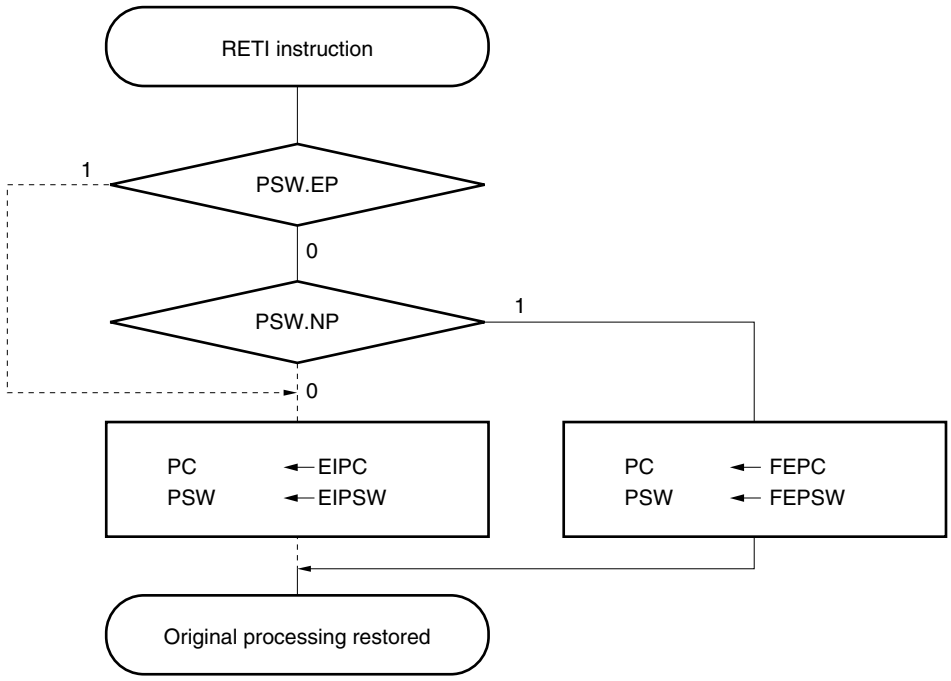
(1) NMI0

Execution is restored from the non-maskable interrupt (NMI0) processing 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> Restores the values of the PC and the 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 8-3 illustrates how the RETI instruction is processed.

Figure 8-3: RETI Instruction Processing



Caution: When the PSW.EP bit and PSW.NP bit are changed by the LDSR instruction during non-maskable interrupt 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 0 and PSW.NP back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark: The solid line indicates the CPU processing flow.

(2) NMIWDT

Restoring by RETI instruction is not possible. Perform a system reset after interrupt servicing.

8.2.3 Non-maskable interrupt status flag (NP)

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

Figure 8-4: Non-maskable Interrupt Status Flag (NP)

	31																							8	7	6	5	4	3	2	1	0	Initial value
PSW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NP	EP	ID	SAT	CY	OV	S	Z	00000020H

Bit Position	Bit Name	Function
7	NP	Indicates whether NMI interrupt processing is in progress. 0: No NMI interrupt processing 1: NMI interrupt currently being processed

8.2.4 Edge detection function

The behaviour of the non-maskable interrupt (NMI0) can be specified by the interrupt mode register 3.
The valid edge of the external NMI pin input can be specified by the ESN0 bit.
The register can be read/written in 8-bit or 1-bit units.

Figure 8-5: Interrupt Mode Register 3 (INTM3)

	7	6	5	4	3	2	1	0	Address	Initial value
INTM3	0	0	0	0	0	0	0	ESN0	FFFFF886H	00H

Bit Position	Bit Name	Function
0	ESN0	Specifies the NMI pin's valid edge 0: Falling edge 1: Rising edge

Notes: 1. This register can be written only once. ESN0 is cleared to "0" by Reset.

2. This register should always be programmed even if the user needs to use the reset value. This will prevent unintended write to this register afterwards.

3. NMI functionality is masked by PMC60. Selection of valid edge for NMI must be performed while PMC60 is "0".

8.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The μPD70F3178 has 58 maskable interrupt sources.
If two or more maskable interrupt requests are generated at the same time, they are acknowledged according to the default priority. In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers (programmable priority control).
When an interrupt request has been acknowledged, the acknowledgement of other maskable interrupt requests is disabled and the interrupt disabled (DI) status is set.
When the EI instruction is executed in an interrupt processing routine, the interrupt enabled (EI) status is set, which enables servicing of interrupts having a higher priority than the interrupt request 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.
However, if multiple interrupts are executed, the following processing is necessary.

<1> Save EIPC and EIPSW in memory or a general-purpose register before executing the EI instruction.
<2> Execute the DI instruction before executing the RETI instruction, then reset EIPC and EIPSW with the values saved in (1).

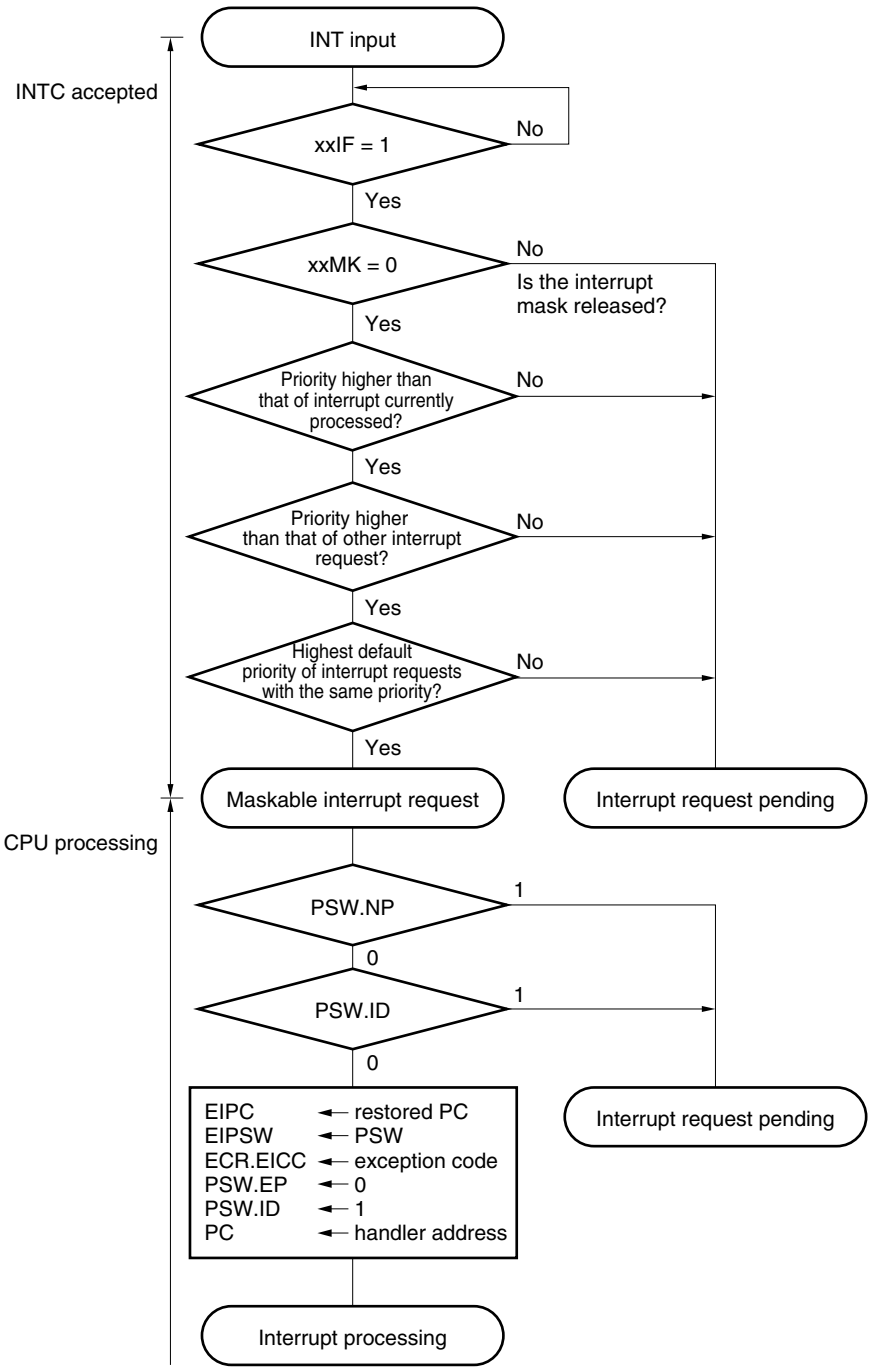
8.3.1 Operation

If a maskable interrupt occurs by INT input, 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 processing configuration of a maskable interrupt is shown in Figure 8-6, “Maskable Interrupt Processing,” on page 301.

Figure 8-6: Maskable Interrupt Processing



Note: For the ISPR register, see 8.3.6 ”In-service priority register (ISPR)” on page 311.

An INT input masked by the interrupt controllers and an INT input that occurs while another interrupt is being processed (when PSW.NP = 1 or PSW.ID = 1) are held pending internally by the interrupt controller. In such case, if the interrupts are unmasked, or when PSW.NP = 0 and PSW.ID = 0 as set by the RETI and LDSR instructions, input of the pending INT starts the new maskable interrupt processing.

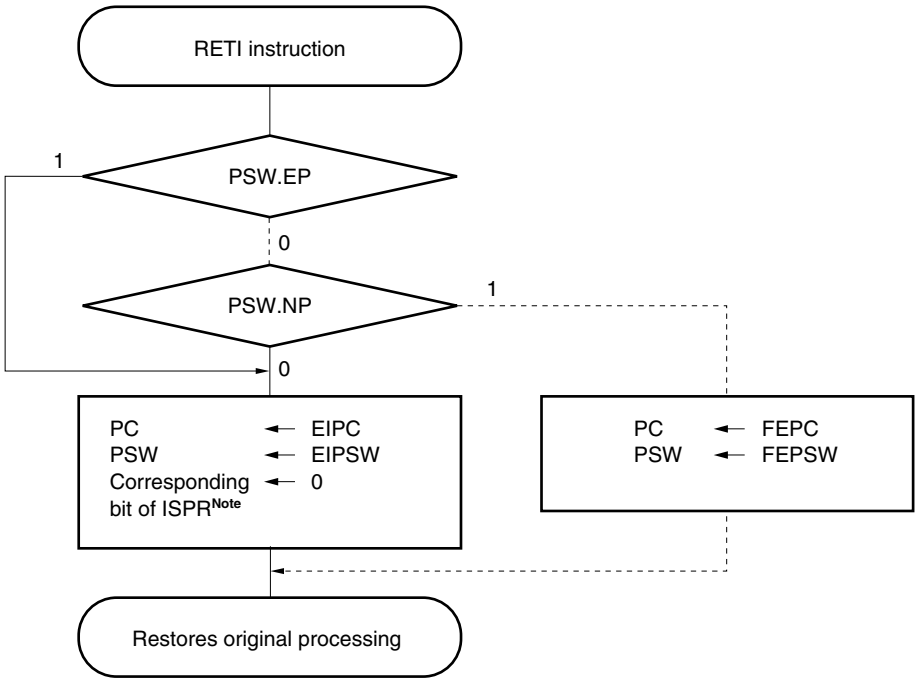
8.3.2 Restore

Recovery from maskable interrupt processing 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> Restores the values of the PC and the 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 8-7 illustrates the processing of the RETI instruction.

Figure 8-7: RETI Instruction Processing



Note: For the ISPR register, see 8.3.6 ”In-service priority register (ISPR)” on page 311.

Caution: **When the PSW.EP bit and the PSW.NP bit are changed by the LDSR instruction during maskable interrupt 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 0 and PSW.NP back to 0 using the LDSR instruction immediately before the RETI instruction.**

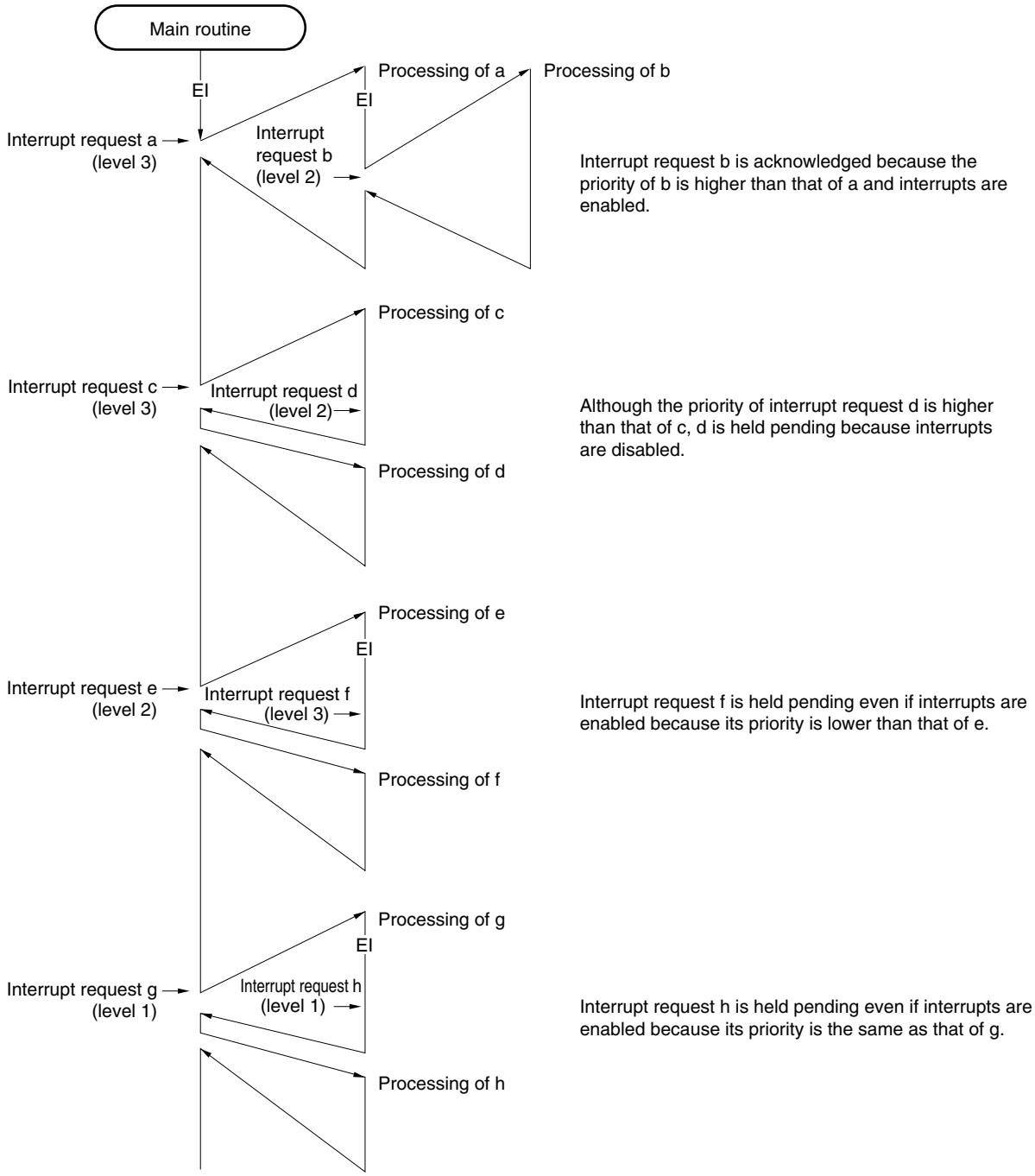
Remark: The solid lines show the CPU processing flow.

8.3.3 Priorities of maskable interrupts

The μPD70F3178 provides multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.
There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels 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, interrupts are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, refer to Table 8-1, “Interrupt Vector Table,” on page 294. The programmable priority control customizes interrupt requests into eight levels by setting the priority level specification flag.

Note that when an interrupt request 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.

Figure 8-8: Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed (1/2)

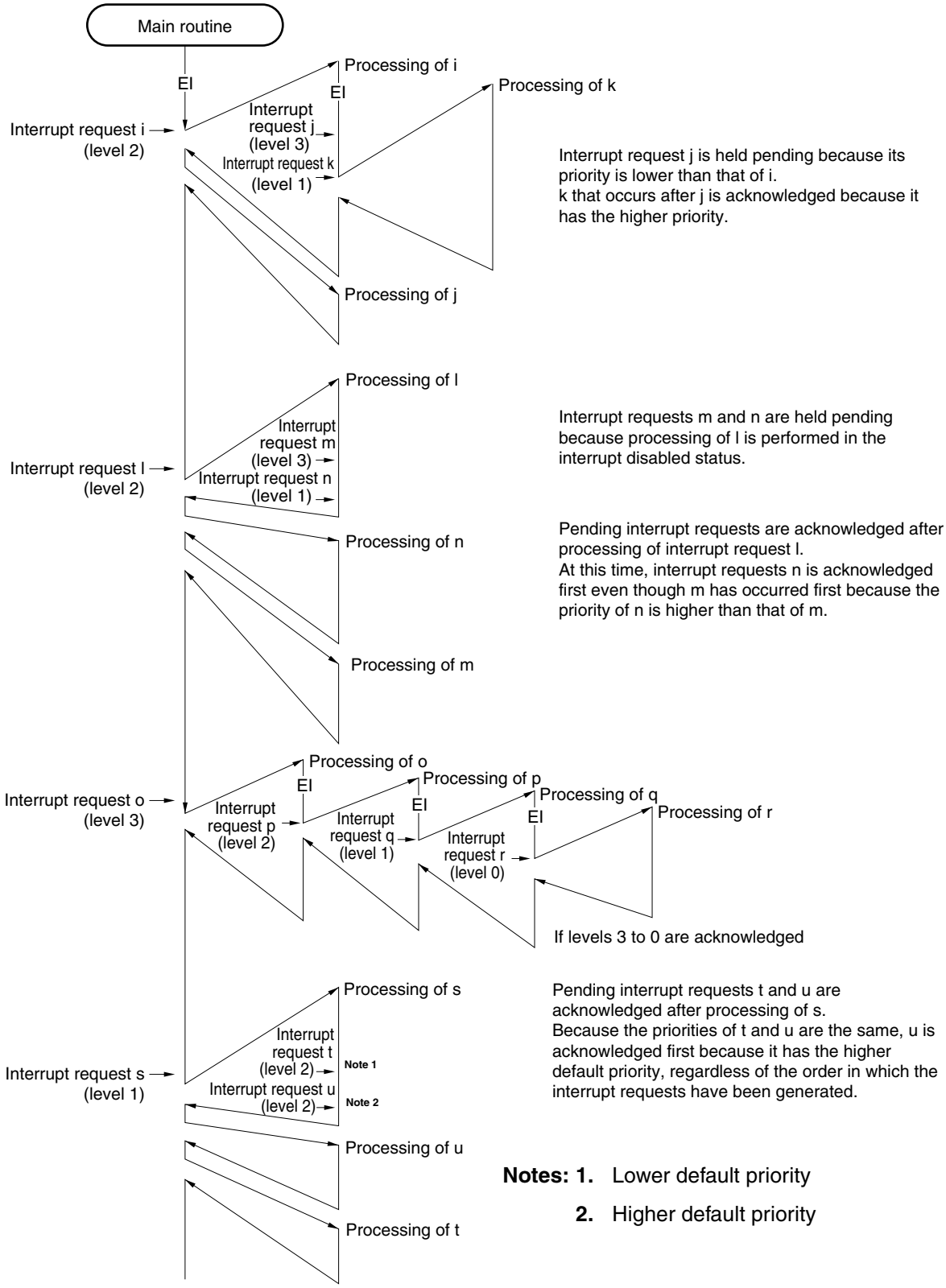


Caution: The values of the EIPC and EIPSW registers must be saved before executing multiple interrupts. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remarks: 1. <a> to <u> in the figure are the temporary names of interrupt requests shown for the sake of explanation.

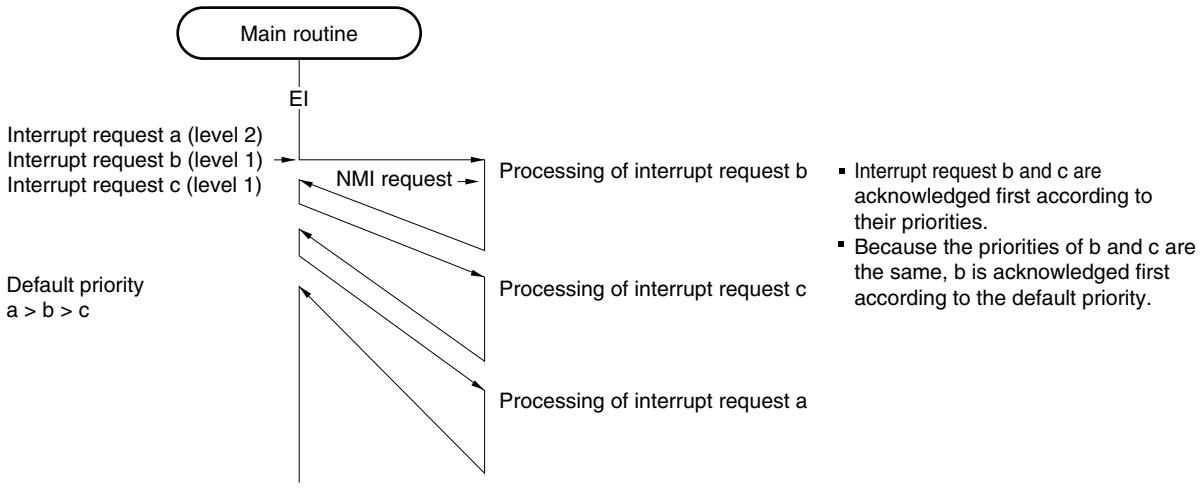
2. The default priority in the figure indicates the relative priority between two interrupt requests.

Figure 8-8: Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed (2/2)



Caution: The values of the EIPC and EIPSW registers must be saved before executing multiple interrupts. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Figure 8-9: Example of Processing Interrupt Requests Simultaneously Generated



Caution: The values of the EIPC and EIPSW registers must be saved before executing multiple interrupts. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remark: <a> to <c> in the figure are the temporary names of interrupt requests shown for the sake of explanation.

8.3.4 Interrupt control register (xxIC)

An interrupt control register is assigned to each interrupt request (maskable interrupt) and sets the control conditions for each maskable interrupt request.
This register can be read/written in 8-bit or 1-bit units.

Figure 8-10: Interrupt Control Register (xxIC)

	7	6	5	4	3	2	1	0	Address	Initial value
xxIC	xxIF	xxMK	0	0	0	xxPR2	xxPR1	xxPR0	FFFFF110H to FFFF182H	47H

Bit Position	Bit Name	Function
7	xxIF	This is an interrupt request flag. 0: Interrupt request not issued 1: Interrupt request issued The flag xxIFn is reset automatically by the hardware if an interrupt request is acknowledged.
6	xxMK	This is an interrupt mask flag. 0: Enables interrupt processing 1: Disables interrupt processing (pending)
2 to 0	xxPR2 to xxPR0	8 levels of priority order are specified for each interrupt.

xxPR2	xxPR1	xxPR0	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)

Remark: xx: Identification name of each peripheral unit (WT, TMD, P, TMG, GCC, AD, MAC, FC, CSI, UART, DMA)

The address and bit of each interrupt control register are shown in the following Table 8-2, “Interrupt Control Register Overview,” on page 308.

Table 8-2: Interrupt Control Register Overview (1/2)

Address	Register	7	6	5	4	3	2	1	0
FFFFF110H	P0IC	P0IF	P0MK	0	0	0	P0PR2	P0PR1	P0PR0
FFFFF112H	P1IC	P1IF	P1MK	0	0	0	P1PR2	P1PR1	P1PR0
FFFFF114H	P2IC	P2IF	P2MK	0	0	0	P2PR2	P2PR1	P2PR0
FFFFF116H	P3IC	P3IF	P3MK	0	0	0	P3PR2	P3PR1	P3PR0
FFFFF118H	P4IC	P4IF	P4MK	0	0	0	P4PR2	P4PR1	P4PR0
FFFFF11AH	P5IC	P5IF	P5MK	0	0	0	P5PR2	P5PR1	P5PR0
FFFFF11CH	P6IC	P6IF	P6MK	0	0	0	P6PR2	P6PR1	P6PR0
FFFFF11EH	P7IC	P7IF	P7MK	0	0	0	P7PR2	P7PR1	P7PR0
FFFFF120H	P8IC	P8IF	P8MK	0	0	0	P8PR2	P8PR1	P8PR0
FFFFF122H	P9IC	P9IF	P9MK	0	0	0	P9PR2	P9PR1	P9PR0
FFFFF124H	P10IC	P10IF	P10MK	0	0	0	P10PR2	P10PR1	P10PR0
FFFFF126H	TP0OVIC	TP0OVIF	TP0OVMK	0	0	0	TP0OVPR2	TP0OVPR1	TP0OVPR0
FFFFF128H	TP0CC0IC	TP0CC0IF	TP0CC0MK	0	0	0	TP0CC0PR2	TP0CC0PR1	TP0CC0PR0
FFFFF12AH	TP0CC1IC	TP0CC1IF	TP0CC1MK	0	0	0	TP0CC1PR2	TP0CC1PR1	TP0CC1PR0
FFFFF12CH	TP1OVIC	TP1OVIF	TP1OVMK	0	0	0	TP1OVPR2	TP1OVPR1	TP1OVPR0
FFFFF12EH	TP1CC0IC	TP1CC0IF	TP1CC0MK	0	0	0	TP1CC0PR2	TP1CC0PR1	TP1CC0PR0
FFFFF130H	TP1CC1IC	TP1CC1IF	TP1CC1MK	0	0	0	TP1CC1PR2	TP1CC1PR1	TP1CC1PR0
FFFFF132H	TP2OVIC	TP2OVIF	TP2OVMK	0	0	0	TP2OVPR2	TP2OVPR1	TP2OVPR0
FFFFF134H	TP2CC0IC	TP2CC0IF	TP2CC0MK	0	0	0	TP2CC0PR2	TP2CC0PR1	TP2CC0PR0
FFFFF136H	TP2CC1IC	TP2CC1IF	TP2CC1MK	0	0	0	TP2CC1PR2	TP2CC1PR1	TP2CC1PR0
FFFFF138H	BRG0IC	BRG0IF	BRG0MK	0	0	0	BRG0PR2	BRG0PR1	BRG0PR0
FFFFF13AH	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
FFFFF13CH	C0ERRIC	C0ERRIF	C0ERRMK	0	0	0	C0ERRPR2	C0ERRPR1	C0ERRPR0
FFFFF13EH	C0WUPIC	C0WUPIF	C0WUPMK	0	0	0	C0WUPPR2	C0WUPPR1	C0WUPPR0
FFFFF140H	C0RECIC	C0RECIF	C0RECMK	0	0	0	C0RECPR2	C0RECPR1	C0RECPR0
FFFFF142H	C0TRXIC	C0TRXIF	C0TRXMK	0	0	0	C0TRXPR2	C0TRXPR1	C0TRXPR0
FFFFF144H	C1ERRIC	C1ERRIF	C1ERRMK	0	0	0	C1ERRPR2	C1ERRPR1	C1ERRPR0
FFFFF146H	C1WUPIC	C1WUPIF	C1WUPMK	0	0	0	C1WUPPR2	C1WUPPR1	C1WUPPR0
FFFFF148H	C1RECIC	C1RECIF	C1RECMK	0	0	0	C1RECPR2	C1RECPR1	C1RECPR0
FFFFF14AH	C1TRXIC	C1TRXIF	C1TRXMK	0	0	0	C1TRXPR2	C1TRXPR1	C1TRXPR0
FFFFF14CH	C2ERRIC	C2ERRIF	C2ERRMK	0	0	0	C2ERRPR2	C2ERRPR1	C2ERRPR0
FFFFF14EH	C2WUPIC	C2WUPIF	C2WUPMK	0	0	0	C2WUPPR2	C2WUPPR1	C2WUPPR0
FFFFF150H	C2RECIC	C2RECIF	C2RECMK	0	0	0	C2RECPR2	C2RECPR1	C2RECPR0
FFFFF152H	C2TRXIC	C2TRXIF	C2TRXMK	0	0	0	C2TRXPR2	C2TRXPR1	C2TRXPR0
FFFFF154H	C3ERRIC	C3ERRIF	C3ERRMK	0	0	0	C3ERRPR2	C3ERRPR1	C3ERRPR0
FFFFF156H	C3WUPIC	C3WUPIF	C3WUPMK	0	0	0	C3WUPPR2	C3WUPPR1	C3WUPPR0
FFFFF158H	C3RECIC	C3RECIF	C3RECMK	0	0	0	C3RECPR2	C3RECPR1	C3RECPR0
FFFFF15AH	C3TRXIC	C3TRXIF	C3TRXMK	0	0	0	C3TRXPR2	C3TRXPR1	C3TRXPR0
FFFFF15CH	C4ERRIC	C4ERRIF	C4ERRMK	0	0	0	C4ERRPR2	C4ERRPR1	C4ERRPR0
FFFFF15EH	C4WUPIC	C4WUPIF	C4WUPMK	0	0	0	C4WUPPR2	C4WUPPR1	C4WUPPR0

Table 8-2: Interrupt Control Register Overview (2/2)

Address	Register	7	6	5	4	3	2	1	0
FFFFF160H	C4RECIC	C4RECIF	C4RECMK	0	0	0	C4RECPR2	C4RECPR1	C4RECPR0
FFFFF162H	C4TRXIC	C4TRXIF	C4TRXMK	0	0	0	C4TRXPR2	C4TRXPR1	C4TRXPR0
FFFFF164H	CB0TIC	CB0TIF	CB0TMK	0	0	0	CB0TPR2	CB0TPR1	CB0TPR0
FFFFF166H	CB0RDIC	CB0RDIF	CB0RDMK	0	0	0	CB0RDPR2	CB0RDPR1	CB0RDPR0
FFFFF168H	CB0REIC	CB0REIF	CB0REMK	0	0	0	CB0REPR2	CB0REPR1	CB0REPR0
FFFFF16AH	CB1TIC	CB1TIF	CB1TMK	0	0	0	CB1TPR2	CB1TPR1	CB1TPR0
FFFFF16CH	CB1RDIC	CB1RDIF	CB1RDMK	0	0	0	CB1RDPR2	CB1RDPR1	CB1RDPR0
FFFFF16EH	CB1REIC	CB1REIF	CB1REMK	0	0	0	CB1REPR2	CB1REPR1	CB1REPR0
FFFFF170H	CB2TIC	CB2TIF	CB2TMK	0	0	0	CB2TPR2	CB2TPR1	CB2TPR0
FFFFF172H	CB2RDIC	CB2RDIF	CB2RDMK	0	0	0	CB2RDPR2	CB2RDPR1	CB2RDPR0
FFFFF174H	CB2REIC	CB2REIF	CB2REMK	0	0	0	CB2REPR2	CB2REPR1	CB2REPR0
FFFFF176H	UA0REIC	UA0REIF	UA0REMK	0	0	0	UA0REPR2	UA0REPR1	UA0REPR0
FFFFF178H	UA0RDIC	UA0RDIF	UA0RDMK	0	0	0	UA0RDPR2	UA0RDPR1	UA0RDPR0
FFFFF17AH	UA0TIC	UA0TIF	UA0TMK	0	0	0	UA0TPR2	UA0TPR1	UA0TPR0
FFFFF17CH	UA1REIC	UA1REIF	UA1REMK	0	0	0	UA1REPR2	UA1REPR1	UA1REPR0
FFFFF17EH	UA1RDIC	UA1RDIF	UA1RDMK	0	0	0	UA1RDPR2	UA1RDPR1	UA1RDPR0
FFFFF180H	UA1TIC	UA1TIF	UA1TMK	0	0	0	UA1TPR2	UA1TPR1	UA1TPR0
FFFFF182H	IIC0IC	IIC0IF	IIC0MK	0	0	0	IIC0PR2	IIC0PR1	IIC0PR0
reserved for internal use									
FFFFF186H	CB0RIC	CB0RIF	CB0RMK	0	0	0	CB0RPR2	CB0RPR1	CB0RPR0
FFFFF188H	CB1RIC	CB1RIF	CB1RMK	0	0	0	CB1RPR2	CB1RPR1	CB1RPR0
FFFFF18AH	CB2RIC	CB2RIF	CB2RMK	0	0	0	CB2RPR2	CB2RPR1	CB2RPR0
FFFFF18CH	UA0RIC	UA0RIF	UA0RMK	0	0	0	UA0RPR2	UA0RPR1	UA0RPR0
FFFFF18EH	UA1RIC	UA1RIF	UA1RMK	0	0	0	UA1RPR2	UA1RPR1	UA1RPR0

Remark: For the interrupt source to the respective controlling registers xxICn refer to “Interrupt Vector Table” on page 294.

8.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)

These registers set the interrupt mask state for the maskable interrupts.
The xxMK bit of the IMR0 to IMR3 registers is equivalent to the xxMK bit of the xxIC register.
IMRm registers can be read/written in 16-bit units (m = 0 to 3).
When the IMRm register is divided into two registers: higher 8 bits (IMRmH register) and lower 8 bits (IMRmL register), these registers can be read/written in 8-bit or 1-bit units (m = 0 to 3).
The address of the lower 8-bit register IMRmL is equal to that of the 16-bit IMRm register, and the higher 8-bit register IMRmH can be accessed on the following address (address (IMRm) + 1).

Figure 8-11: Interrupt Mask Registers 0 to 3 (IMR0 to IMR3)

	15	14	13	12	11	10	9	8	Address	Initial value
IMR0	TP1CC0MK	TP1OVMK	TP0CC1MK	TP0CC0MK	TP0OVMK	P10MK	P9MK	P8MK	FFFFF100H	FFFFH
	7	6	5	4	3	2	1	0		
	P7MK	P6MK	P5MK	P4MK	P3MK	P2MK	P1MK	P0MK		
	15	14	13	12	11	10	9	8	Address	Initial value
IMR1	C2WUPMK	C2ERRMK	C1TRXMK	C1RECMK	C1WUPMK	C1ERRMK	C0TRXMK	C0RECMK	FFFFF102H	FFFFH
	7	6	5	4	3	2	1	0		
	C0WUPMK	C0ERRMK	ADMK	BRG0MK	TP2CC1MK	TP2CC0MK	TP2OVMK	TP1CC1MK		
	15	14	13	12	11	10	9	8	Address	Initial value
IMR2	CB1REMK	CB1RDMK	CB1TMK	CB0REMK	CB0RDMK	CB0TMK	C4TRXMK	C4RECMK	FFFFF104H	FFFFH
	7	6	5	4	3	2	1	0		
	C4WUPMK	C4ERRMK	C3TRXMK	C3RECMK	C3WUPMK	C3ERRMK	C2TRXMK	C2RECMK		
	15	14	13	12	11	10	9	8	Address	Initial value
IMR3	UA1RMK	UA0RMK	CB2RMK	CB1RMK	CB0RMK	1	IIC0MK	UA1TMK	FFFFF106H	FFFFH
	7	6	5	4	3	2	1	0		
	UA1RDMK	UA1REMK	UA0TMK	UA0RDMK	UA0REMK	CB2REMK	CB2RDMK	CB2TMK		

Bit Position	Bit Name	Function
15 to 0	xxMK	Interrupt mask flag. 0: Interrupt servicing enabled 1: Interrupt servicing disabled (pending)

Remark: xx: Identification name of each peripheral unit (TMP, P, AD, FC, CSI, UART,IIC)

8.3.6 In-service priority register (ISPR)

This register holds the priority level of the maskable interrupt currently acknowledged. When an interrupt request is acknowledged, the bit of this register corresponding to the priority level of that interrupt request 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 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.

Figure 8-12: In-Service Priority Register (ISPR)

	7	6	5	4	3	2	1	0	Address	Initial value
ISPR	ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0	FFFFF1FAH	00H

Bit Position	Bit Name	Function
7 to 0	ISPR7 to ISPR0	Indicates priority of interrupt currently acknowledged 0: Interrupt request with priority n not acknowledged 1: Interrupt request with priority n acknowledged

Remark: n = 0 to 7 (priority level)

8.3.7 Maskable interrupt status flag (ID)

The ID flag is bit 5 of the PSW and this controls the maskable interrupt's operating state, and stores control information regarding enabling or disabling of interrupt requests.

Figure 8-13: Maskable Interrupt Status Flag (ID)

	31																								8	7	6	5	4	3	2	1	0	Initial value
PSW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NP	EP	ID	SAT	CY	OV	S	Z		00000020H

Bit Position	Bit Name	Function
5	ID	Indicates whether maskable interrupt processing is enabled or disabled. 0: Maskable interrupt request acknowledgement enabled 1: Maskable interrupt request acknowledgement disabled (pending) 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 to PSW. Non-maskable interrupt requests and exceptions are acknowledged regardless of this flag. when a maskable interrupt is acknowledged, the ID flag is automatically set to 1 by hardware. The interrupt request generated during the acknowledgement disabled period (ID = 1) is acknowledged when the PIFn bit of PICn register is set to 1, and the ID flag is reset to 0.

8.4 Noise Elimination Circuit

Figure 8-14: Port Interrupt Input Circuit (P01, P27)

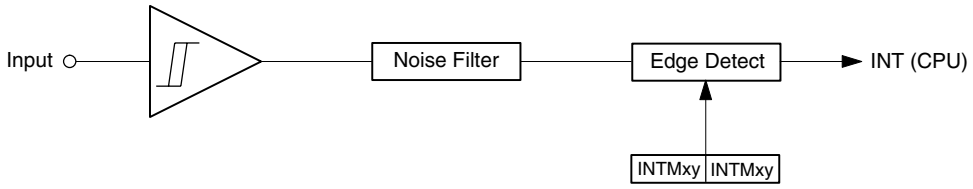


Figure 8-15: TimerP0 - 2 Input Circuit (P20-P25)

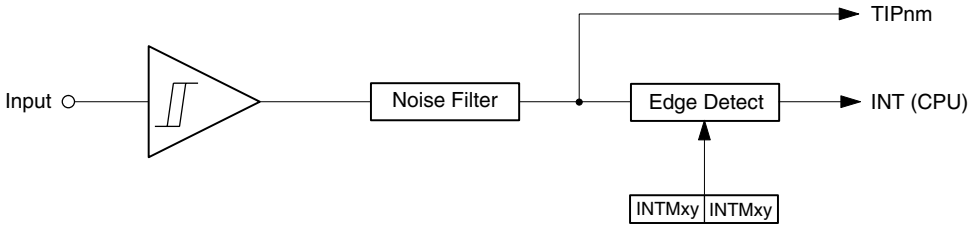


Figure 8-16: TimerP2 TTRGP2 Input Circuit (P26)

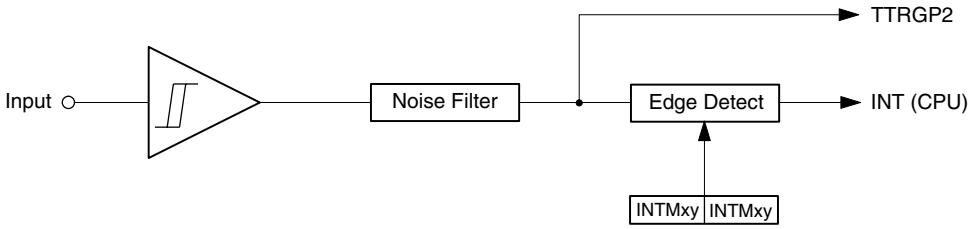


Figure 8-17: UARTA0-1 Input Circuit (P30, P32)

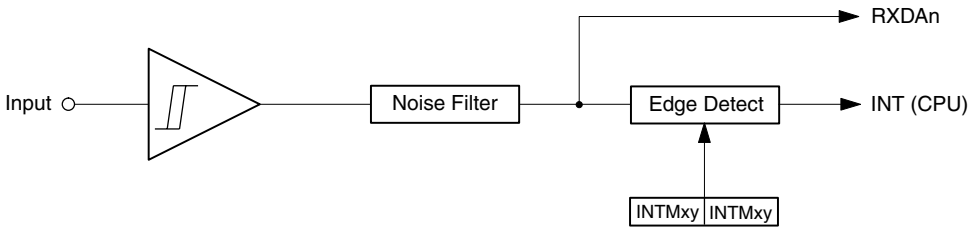
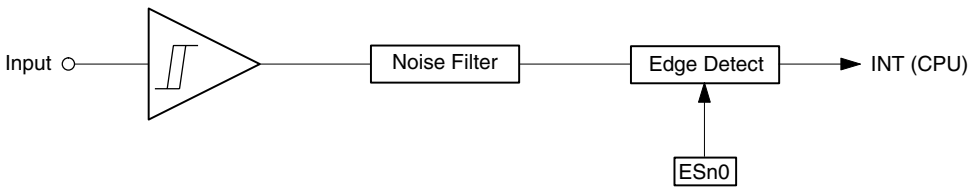


Figure 8-18: NMI Input Circuit (P00)



8.4.1 Analog filter

The analog filter consists of a comparator stage, which compares the input pin level against a delayed input pin level. The filter output follows the filter input, if this compare operation matches.

8.4.2 Interrupt trigger mode selection

The valid edge of the INTP pins can be selected by the program. The edge that can be selected as the valid edge is one of the following.

- Rising edge
- Falling edge
- Both the rising and the falling edges

8.4.3 Interrupt edge detection control registers

Valid interrupt edges can be selected by INTM0 to INTM3 registers. Masking of interrupts is done inside the concerning interrupt control registers xxIC.

(1) Interrupt mode register 0 (INTM0)

Figure 8-19: Interrupt Mode Register 0 (INTM0)

	7	6	5	4	3	2	1	0	Address	Initial value
INTM0	INTM07	INTM06	INTM05	INTM04	INTM03	INTM02	INTM01	INTM00	FFFFF880H	00H

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>INTM07, INTM06</td><td>Edge selection for INTP3 to interrupt controller.
Selects active edge for interrupt generation.
<table>
<tr><th>INTM07</th><th>INTM06</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>5, 4</td><td>INTM05, INTM04</td><td>Edge selection for INTP2 to interrupt controller.
<table>
<tr><th>INTM05</th><th>INTM04</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>3, 2</td><td>INTM03, INTM02</td><td>Edge selection for INTP1 to interrupt controller.
<table>
<tr><th>INTM03</th><th>INTM02</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>1, 0</td><td>INTM01, INTM00</td><td>Edge selection for INTP0 to interrupt controller.
<table>
<tr><th>INTM01</th><th>INTM00</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
</table>

Note: Programming edge detection or port mode register can trigger unintended interrupt requests. Therefore be sure to mask the respective interrupt requests.

(2) Interrupt mode register 1 (INTM1)

Figure 8-20: Interrupt Mode Register 1 (INTM1)

	7	6	5	4	3	2	1	0	Address	Initial value
INTM1	INTM17	INTM16	INTM15	INTM14	INTM13	INTM12	INTM11	INTM10	FFFFF882H	00H

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>INTM17,
INTM16</td><td>Edge selection for INTP7 to interrupt controller.
Selects active edge for interrupt generation.
<table>
<tr><th>INTM17</th><th>INTM16</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>5, 4</td><td>INTM15,
INTM14</td><td>Edge selection for INTP6 to interrupt controller.
<table>
<tr><th>INTM15</th><th>INTM14</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>3, 2</td><td>INTM13,
INTM12</td><td>Edge selection for INTP5 to interrupt controller.
<table>
<tr><th>INTM13</th><th>INTM12</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>1, 0</td><td>INTM11,
INTM10</td><td>Edge selection for INTP4 to interrupt controller.
<table>
<tr><th>INTM11</th><th>INTM10</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
</table>

Caution: Programming edge detection or port mode register can trigger unintended interrupt requests. Therefore be sure to mask the respective interrupt requests.

(3) Interrupt mode register 2 (INTM2)

Figure 8-21: Interrupt Mode Register 2 (INTM2)

	7	6	5	4	3	2	1	0	Address	Initial value
INTM2	0	0	INTM25	INTM24	INTM23	INTM22	INTM21	INTM20	FFFFF884H	00H

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>5, 4</td><td>INTM25,
INTM24</td><td>Edge selection for INTP10 to interrupt controller.
<table>
<tr><th>INTM25</th><th>INTM24</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>3, 2</td><td>INTM23,
INTM22</td><td>Edge selection for INTP9 to interrupt controller.
<table>
<tr><th>INTM23</th><th>INTM22</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
<tr><td>1, 0</td><td>INTM21,
INTM20</td><td>Edge selection for INTP8 to interrupt controller.
<table>
<tr><th>INTM21</th><th>INTM20</th><th>Edge selection</th></tr>
<tr><td>0</td><td>0</td><td>Falling edge</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Both edges</td></tr>
</table></td></tr>
</table>

Caution: Programming edge detection or port mode register can trigger unintended interrupt requests. Therefore be sure to mask the respective interrupt requests.

(4) Interrupt mode register 3 (INTM3)

Figure 8-22: Interrupt Mode Register 3 (INTM3)

	7	6	5	4	3	2	1	0	Address	Initial value
INTM3	0	0	0	0	0	0	INTM31	INTM30	FFFFF886H	00H

Bit Position	Bit Name	Function
1, 0	INTM31, INTM30	Edge selection for NMI to interrupt controller.

INTM21	INTM20	Edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Reserved
1	1	Both edges

Cautions: 1. NMI functionality is masked by PMC00. Selection of valid edge for NMI must be performed while PMC00 is “0”.

2. Install appropriate interrupt handler for NMI before reprogramming edge detection or port function.

8.5 Software Exception

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

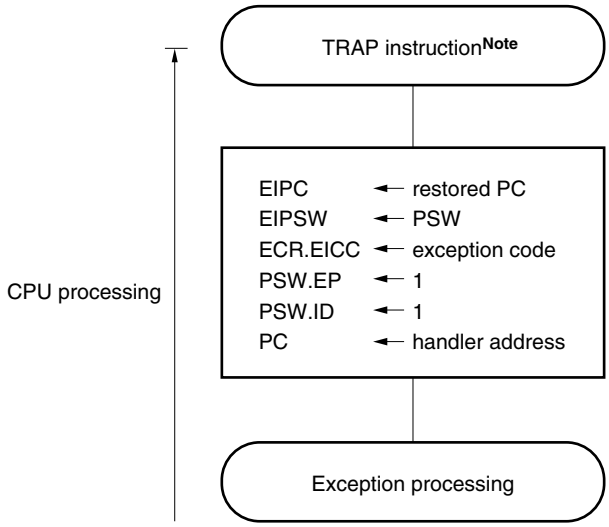
8.5.1 Operation

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 8-23 illustrates the processing of a software exception.

Figure 8-23: 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.

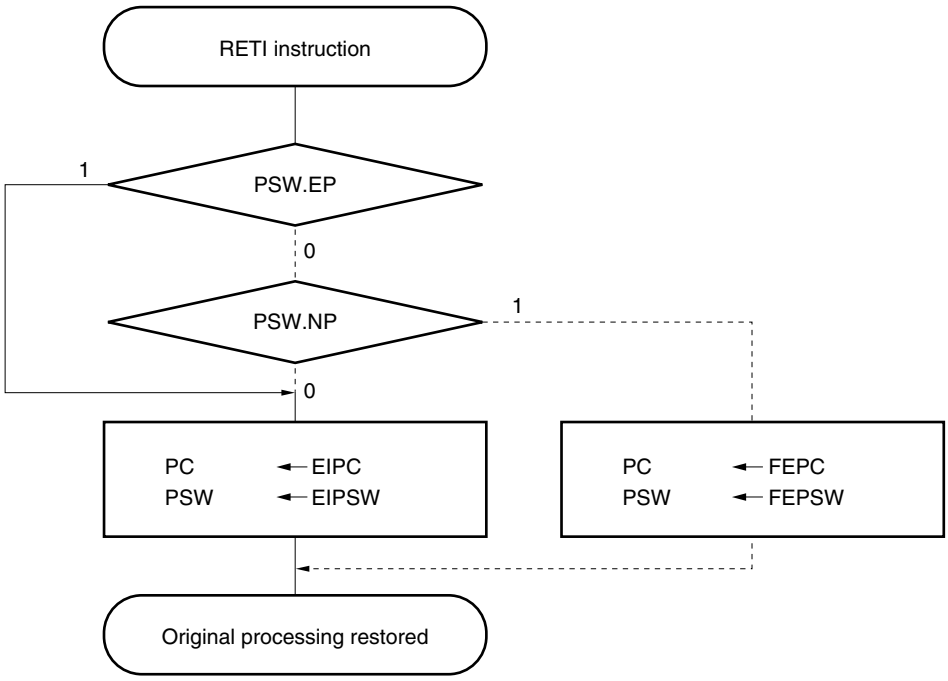
8.5.2 Restore

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 8-24 illustrates the processing of the RETI instruction.

Figure 8-24: 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 lines show the CPU processing flow.

8.5.3 Exception status flag (EP)

The EP flag is bit 6 of PSW, and is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs.

Figure 8-25: Exception Status Flag (EP)

	31																							8	7	6	5	4	3	2	1	0	Initial value
PSW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NP	EP	ID	SAT	CY	OV	S	Z	00000020H

Bit Position	Bit Name	Function
6	EP	Shows that exception processing is in progress. 0: Exception processing not in progress. 1: Exception processing in progress.

8.6 Exception Trap

An exception trap is an interrupt that is requested when an illegal execution of an instruction takes place. In the μPD70F3178, an illegal opcode exception (ILGOP: Illegal Opcode Trap) is considered as an exception trap.

8.6.1 Illegal opcode definition

The illegal instruction has an opcode (bits 10 to 5) of 111111B, a sub-opcode (bits 23 to 26) 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.

15 11	10 5	4 0	31 27	26 23	22 16
× × × × ×	1 1 1 1 1 1	× × × × ×	× × × × ×	0 1 1 1 to 1 1 1 1	× × × × × × 0

Remark: ×: Arbitrary

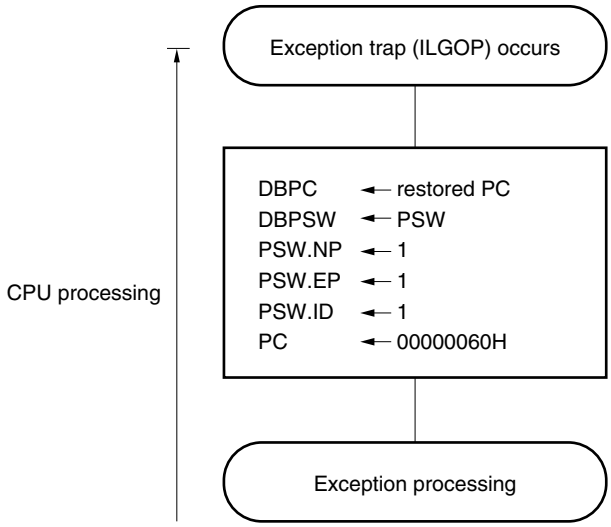
(1) Operation

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 8-26 illustrates the processing of the exception trap.

Figure 8-26: Exception Trap Processing



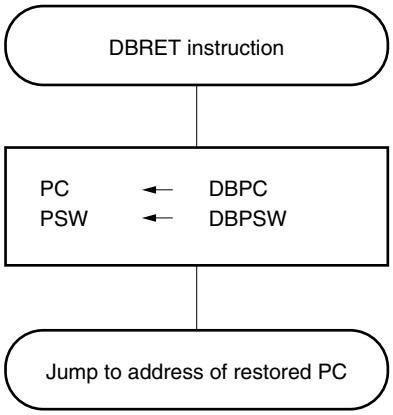
(2) Restore

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 8-27 illustrates the restore processing from an exception trap.

Figure 8-27: Restore Processing from Exception Trap



8.7 Multiple Interrupt Processing Control

Multiple interrupt processing control is a process by which an interrupt request that is currently being processed can be interrupted during processing if there is an interrupt request with a higher priority level, and the higher priority interrupt request is received and processed first.
If there is an interrupt request with a lower priority level than the interrupt request currently being processed, that interrupt request is held pending.
Maskable interrupt multiple processing control is executed when an interrupt has an enable status (ID = 0). Thus, if multiple interrupts are executed, it is necessary to have an interrupt enable status (ID = 0) even for an interrupt processing routine.
If a maskable interrupt enable or a software exception is generated in a maskable interrupt or software exception service program, it is necessary to save EIPC and EIPSW.
This is accomplished by the following procedure.

(1) Acknowledgment of maskable interrupts in service program

Service program of maskable interrupt or exception

```
...
...
• EIPC saved to memory or register
• EIPSW saved to memory or register
• EI instruction (interrupt acknowledgment enabled)
...
...                                        ¨ Maskable interrupt acknowledgment
...
...
• DI instruction (interrupt acknowledgment disabled)
• Saved value restored to EIPSW
• Saved value restored to EIPC
• RETI instruction
```

(2) Generation of exception in service program

Service program of maskable interrupt or exception

... ... • EIPC saved to memory or register • EIPSW saved to memory or register ... • TRAP instruction ... • Saved value restored to EIPSW • Saved value restored to EIPC • RETI instruction	¨ Exception such as TRAP instruction acknowledged.

The priority order for multiple interrupt processing control has 8 levels, from 0 to 7 for each maskable interrupt request (0 is the highest priority), but it can be set as desired via software. Setting of the priority order level is done using the PPRn0 to PPRn2 bits of the interrupt control request register (PICn), which is provided for each maskable interrupt request. After system reset, an interrupt request is masked by the PMKn bit and the priority order is set to level 7 by the PPRn0 to PPRn2 bits.

The priority order of maskable interrupts is as follows.

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

Interrupt processing that has been suspended as a result of multiple processing control is resumed after the processing of the higher priority interrupt has been completed and the RETI instruction has been executed.
A pending interrupt request is acknowledged after the current interrupt processing has been completed and the RETI instruction has been executed.

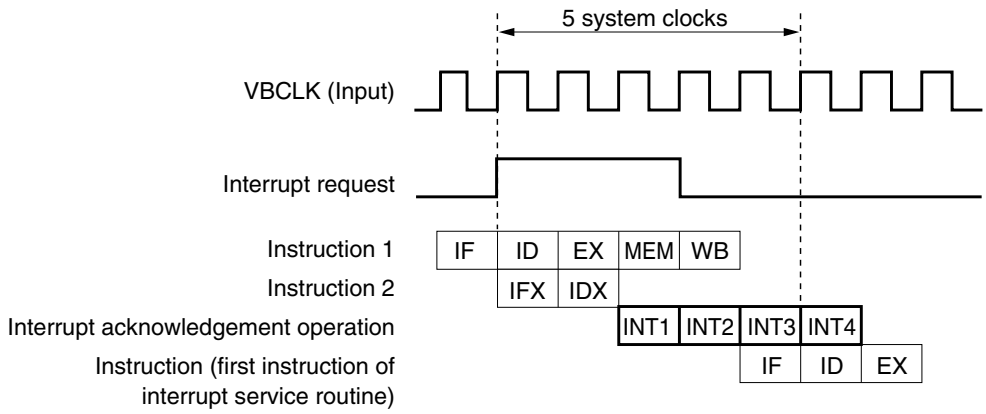
Caution: In a non-maskable interrupt processing routine (time until the RETI instruction is executed), maskable interrupts are suspended and not acknowledged.

8.8 Interrupt Response Time

The following table describes the μPD70F3178 interrupt response time (from interrupt generation to start of interrupt processing).
Except in the following cases, the interrupt response time is a minimum of 5 clocks. To input interrupt requests continuously, leave a space of at least 5 clocks between interrupt request inputs.

- During software or hardware STOP mode
- When an external bus is accessed
- When there are two or more successive interrupt request non-sampling instructions (see 8.9 "Periods in Which Interrupts Are Not Acknowledged" on page 327).
- When the interrupt control register is accessed

Figure 8-28: Pipeline Operation at Interrupt Request Acknowledgment (Outline)



Remark: INT1 to INT4: Interrupt acknowledgement processing
IFx: Invalid instruction fetch
IDx: Invalid instruction decode

Table 8-3: Interrupt Response Time

Interrupt Response Time (Internal System Clocks)			Condition
	Internal Interrupt	External interrupt	
Minimum	5	5 + analog delay time	The following cases are exceptions: • In IDLE/software STOP mode • External bit access • Two or more interrupt request non-sample instructions are executed • Access to interrupt control register
Maximum	11	11 + analog delay time	

8.9 Periods in Which Interrupts Are Not Acknowledged

An interrupt is acknowledged while an instruction is being executed. However, no interrupt will be acknowledged between an interrupt non-sample instruction and the next instruction.
The interrupt request non-sampling instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- The store instruction for the interrupt control register (PICn), in-service priority register (ISPR), and command register (PRCMD).

[MEMO]

Chapter 9 A/D Converter

9.1 Functions

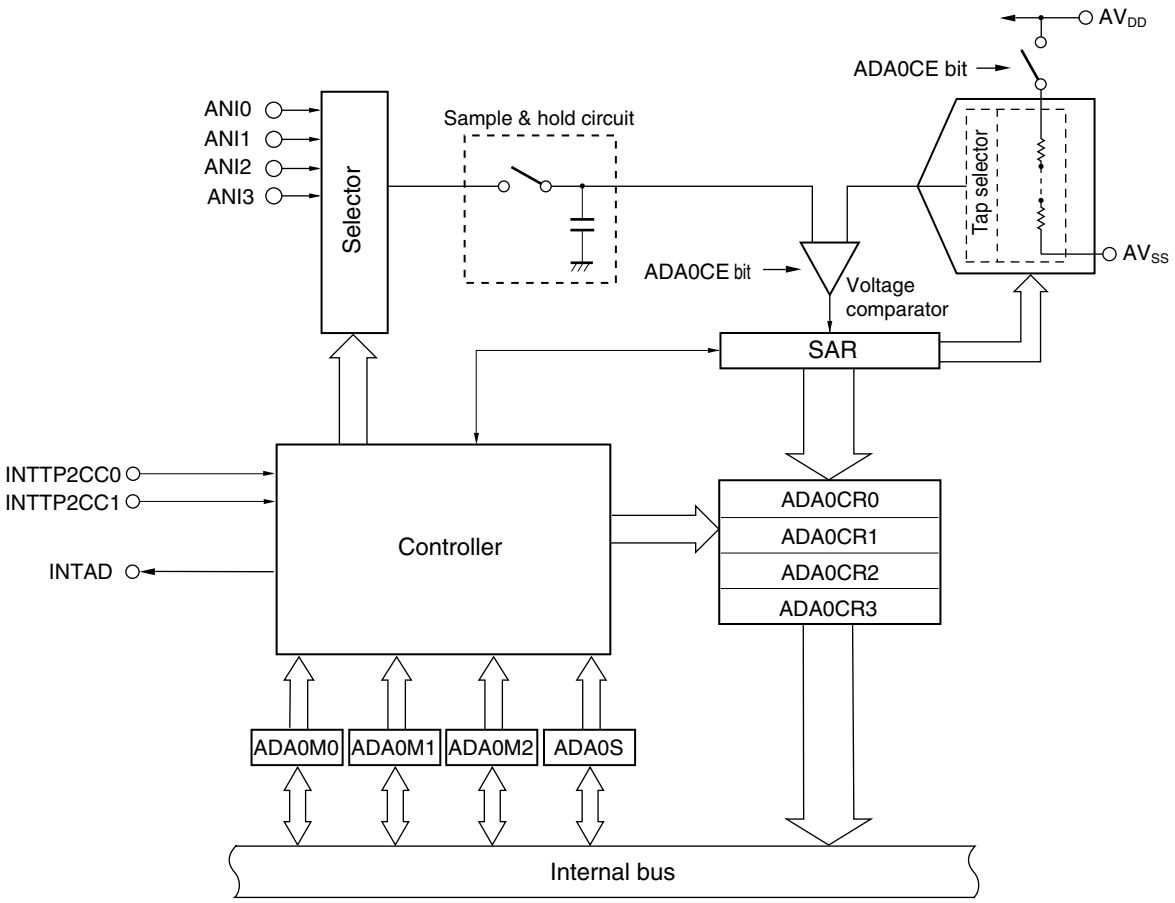
The A/D converter converts analog input signals into digital values, has a resolution of 10 bits, and can handle up to 4 analog input signal channels (ANI0 to ANI3).
The A/D converter has the following features.

Caution: It is mandatory that the V850E/CG3 (CarCate+) ADC must be enabled after the occurrence of any Reset-Source within a time-period of max. 1 s between the conditions "Reset-Release" and "start of the first ADC conversion". With the execution of the first ADC conversion, the ADC circuit is initialized.
The execution of the ADC's enable sequence and its execution of the first conversion is mandatory regardless if ADC operation respectively function is required from the user application or not.

- 10-bit resolution
- 4 channels
- Successive approximation method
- Operating voltage: AV_{DD} = 4.0 to 5.5 V
- Analog input voltage: 0 V to AV_{DD}
- The following functions are provided as operation modes.
    - Continuous select mode
    - Continuous scan mode
- The following functions are provided as trigger modes.
    - Software trigger mode
    - Timer trigger mode
- Power-fail monitor function (conversion result compare function)

The block diagram of the A/D converter is shown below.

Figure 9-1: Block Diagram of A/D Converter



9.2 Configuration

The A/D converter includes the following hardware.

Table 9-1: Configuration of A/D Converter (V850E/CG3)

Item	Configuration
Analog inputs	4 channels (ANI0 to ANI3 pins)
Registers	Successive approximation register (SAR) A/D conversion result registers 0 to 3 (ADA0CR0 to ADA0CR3) A/D conversion result registers 0H to 3H (ADCR0H to ADCR3H): Only higher 8 bits can be read
Control registers	A/D converter mode registers 0 to 2 (ADA0M0 to ADA0M2) 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 is complete), the contents of the SAR register are transferred to the ADA0CRn register.

Remark: n = 0 to 3

(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 4 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.
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 writing 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 3

(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 of the A/D conversion result register (ADA0CRnH).
This register can be read or written in 8-bit or 1-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_{DD} and AV_{SS} and generates a voltage for comparison with the analog input signal.

(7) ANI0 to ANI3 pins

These are analog input pins for the 4 A/D converter channels and are used to input analog signals to be converted into digital signals. Pins other than the one selected as the analog input by the ADA0S register can be used as input port pins.

Cautions: 1. Make sure that the voltages input to the ANI0 to ANI3 pins do not exceed the rated values. In particular if a voltage of AV_{DD} or higher 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.

2. The analog input pins (ANI0 to ANI3) function alternately as input port pins (P70 to P73). If any of ANI0 to ANI3 is selected and A/D converted, do not execute an input instruction to ports 7 and 12 during conversion. If executed, the conversion resolution may be degraded.

(8) AV_{DD} pin

This is the pin used to input the reference voltage of the A/D converter. The signals input to the ANI0 to ANI3 pins are converted to digital signals based on the voltage applied between the AV_{DD} 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.

9.3 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 operations.
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.

Figure 9-2: A/D converter mode register 0 (ADA0M0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0M0	ADA0CE	0	ADA0MD1	ADA0MD0	ADA0ETS1	ADA0ETS0	ADA0TMD	ADA0EF	FFFFF200H	R/W	00H

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
Other than above		Setting prohibited

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	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, 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 the ADA0CE bit to 0 to reduce the current consumption.

4. Be sure to clear bit 6.

(2) A/D converter mode register 1 (ADA0M1)

The ADA0M1 register is an 8-bit register that controls the conversion time specification.
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this bit to 00H.

Figure 9-3: A/D converter mode register 1 (ADA0M1) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0M1	ADA0HS1	ADA0HS0	0	0	ADA0FR3	ADA0FR2	ADA0FR1	ADA0FR0	FFFFF201H	R/W	00H

Caution: Be sure to clear bits 5 and 4.

Remark: For A/D conversion time setting examples, see **Table 9-2**.

Table 9-2: Conversion Mode Setting Example

ADA0HS		ADA0FR3 to ADA0FR0				A/D Conversion Time	f_{XX} = 32 MHz	f_{XX} = 24 MHz	f_{XX} = 4 MHz	A/D Stabilization Time[Note]
1	0	3	2	1	0					
0	0	0	0	0	0	$31/f_{XX}$	Setting prohibited		19.50 μs	$16/f_{XX}$
		0	0	0	1	$62/f_{XX}$	4.84 μs	6.46 μs	38.75 μs	$31/f_{XX}$
		0	0	1	0	$93/f_{XX}$	7.28 μs	9.71 μs	Setting prohibited	$47/f_{XX}$
		0	0	1	1	$124/f_{XX}$	9.31 μs	12.42 μs	Setting prohibited	$50/f_{XX}$
		0	1	0	0	$155/f_{XX}$	11.25 μs	15.00 μs	Setting prohibited	$50/f_{XX}$
		0	1	0	1	$186/f_{XX}$	13.19 μs	17.58 μs	Setting prohibited	$50/f_{XX}$
		0	1	1	0	$217/f_{XX}$	15.13 μs	20.17μs	Setting prohibited	$50/f_{XX}$
		0	1	1	1	$248/f_{XX}$	17.06 μs	22.75 μs	Setting prohibited	$50/f_{XX}$
		1	0	0	0	$279/f_{XX}$	19.00 μs	25.33 μs	Setting prohibited	$50/f_{XX}$
		1	0	0	1	$310/f_{XX}$	20.94 μs	27.92 μs	Setting prohibited	$50/f_{XX}$
		1	0	1	0	$341/f_{XX}$	22.88 μs	30.50 μs	Setting prohibited	$50/f_{XX}$
		1	0	1	1	$372/f_{XX}$	24.81 μs	33.08 μs	Setting prohibited	$50/f_{XX}$
		1	1	0	0	$403/f_{XX}$	26.75 μs	35.670 μs	Setting prohibited	$50/f_{XX}$
		1	1	0	1	$434/f_{XX}$	28.69 μs	Setting prohibited	Setting prohibited	$50/f_{XX}$
		1	1	1	0	$465/f_{XX}$	30.63 μs	Setting prohibited	Setting prohibited	$50/f_{XX}$
		1	1	1	1	$496/f_{XX}$	32.56 μs	Setting prohibited	Setting prohibited	$50/f_{XX}$
0	1	x	x	x	x	Setting prohibited				
1	x	x	x	x	x					

Note: When the ADA0CE bit of the ADA0M0 register is changed from 0 to 1 to secure the A/D converter stabilization time, the first 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.

Figure 9-4: A/D converter mode register 2 (ADA0M2) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0M2	0	0	0	0	0	0	ADA0TMD1	ADA0TMD0	FFFFF203H	R/W	00H

ADA0TMD1	ADA0TMD0	Specification of hardware trigger mode
0	0	Setting prohibited
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.

(4) A/D converter channel specification register 0 (ADA0S)

The ADA0S register specifies the pin that inputs 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.

Figure 9-5: A/D converter channel specification register 0 (ADA0S) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0S	0	0	0	0	0	0	ADA0S1	ADA0S0	FFFFF202H	R/W	00H

ADA0S1	ADA0S0	Select mode	Scan mode
0	0	ANI0	ANI0
0	1	ANI1	ANI0, ANI1
1	0	ANI2	ANI0 to ANI2
1	1	ANI3	ANI0 to ANI3

Caution: Be sure to clear bit 7 to 2.

(5) A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH)

The ADA0CRn and ADA0CRnH registers store the 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 are read from the higher 10 bits of the ADA0CRn register, and 0 is read from the lower 6 bits. The higher 8 bits of the conversion result are read from the ADA0CRnH register.

Figure 9-6: A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH) Format

Address: ADACR0 FFFFF210H, ADA0CR1 FFFFF212H_C
ADACR2 FFFFF214H, ADA0CR3 FFFFF216H

	15	14	13	12	11	10	9	8	R/W	After reset
ADA0CRn	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	R	00H
(n=0 to 3)	7	6	5	4	3	2	1	0		
	AD1	AD0								

Address: ADACR0H FFFFF211H, ADA0CR1H FFFFF213H
ADACR2H FFFFF215H, ADA0CR3H FFFFF217H

	7	6	5	4	3	2	1	0	R/W	After reset
ADA0CRnH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	R	00H
(n=0 to 3)										

Remark: n = 0 to 3

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI3) and the A/D conversion result (of A/D conversion result register n (ADA0CRn)) is as follows.

$$ADA0CR = INT\left(\frac{V_{IN}}{AV_{REF0}} \times 1{,}024 + 0.5\right)$$

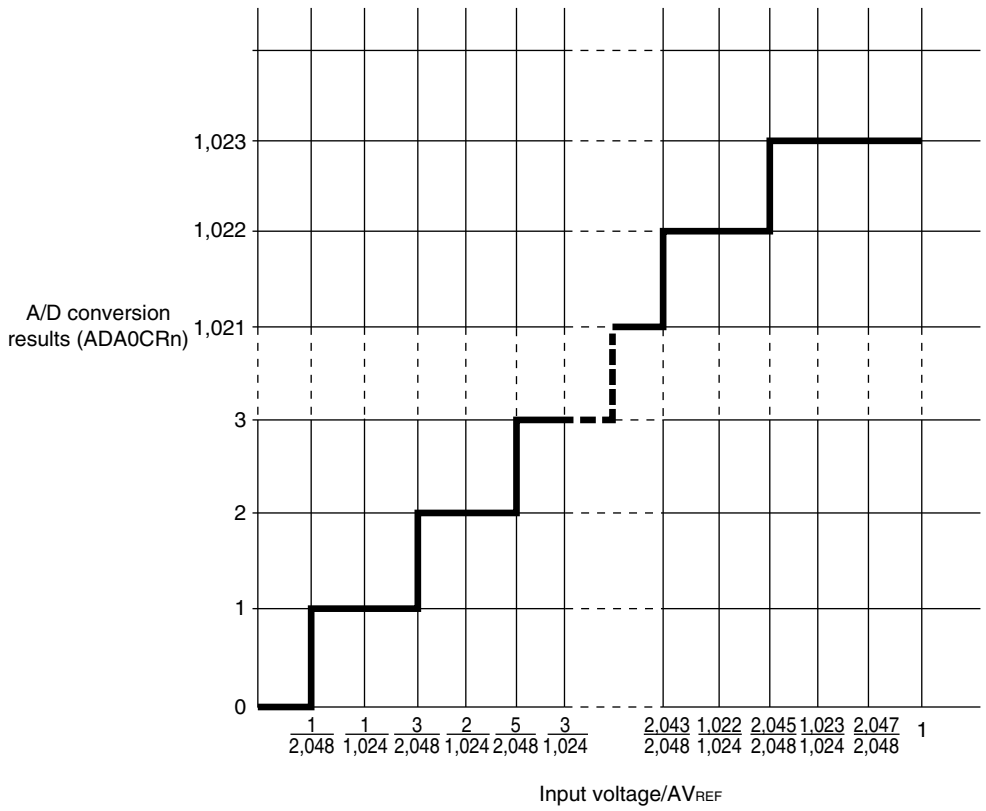
Or,

$$(ADA0CR - 0.5) \times \frac{AV_{REF0}}{1{,}024} \leq V_{IN} < (ADA0CR + 0.5) \times \frac{AV_{REF0}}{1{,}024}$$

INT( ): Function that returns the integer of the value in ( )
V_{IN}: Analog input voltage
AV_{DD}: AV_{DD} pin voltage
ADA0CR: Value of A/D conversion result register n (ADA0CRn)

Figure 9-7 shows the relationship between the analog input voltage and the A/D conversion results.

Figure 9-7: Relationship Between Analog Input Voltage and A/D Conversion Results



Remark: n = 0 to 3

(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.

Figure 9-8: Power-fail compare mode register (ADA0PFM) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0PFM	ADA0PFE	ADA0PFC	0	0	0	0	0	0	FFFFF204H	R/W	00H

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.

3. Be sure to clear bit 5 to 0.

(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.

Figure 9-9: Power-fail compare threshold value register (ADA0PFT) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
ADA0PFT									FFFFF205H	R/W	00H

9.4 Operation

9.4.1 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 or 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 A/D conversion is complete.

<4> Set bit 9 of the successive approximation register (SAR). The tap selector selects (1/2) AV_{DD} 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_{DD}, the MSB of the SAR register remains set. If it is lower than (1/2) AV_{DD}, 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.

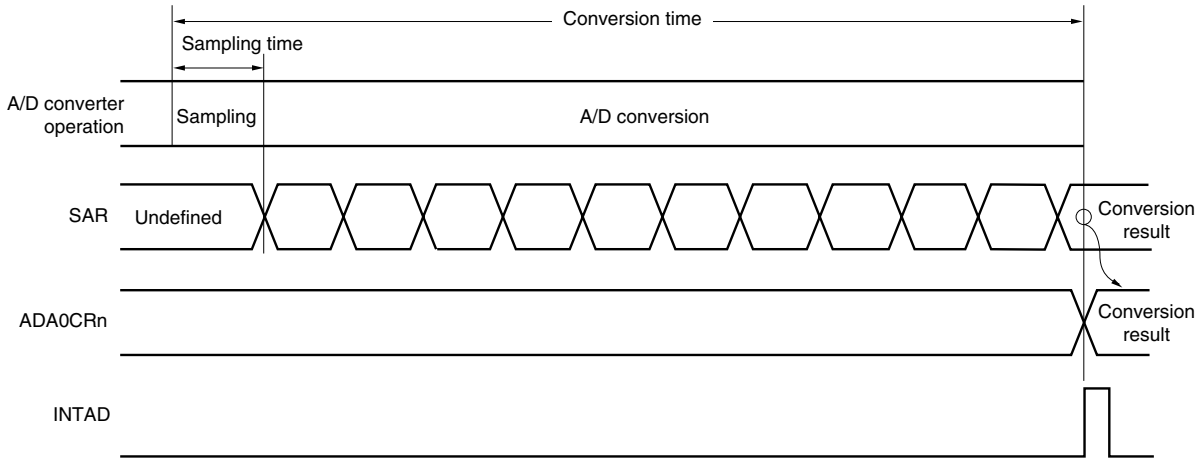
- Bit 9 = 1: (3/4) AV_{DD}
- Bit 9 = 0: (1/4) AV_{DD}

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 is complete, 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 9-10: A/D Converter Basic Operation



9.4.2 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 ANI3 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.

When conversion is started, the ADA0EF bit is set to 1 (indicating that conversion 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) Timer trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI3) 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. The timer compare match interrupt request signal (INTTP2CC0 or INTTP2CC1) 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 conversion is in progress). While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that conversion 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.

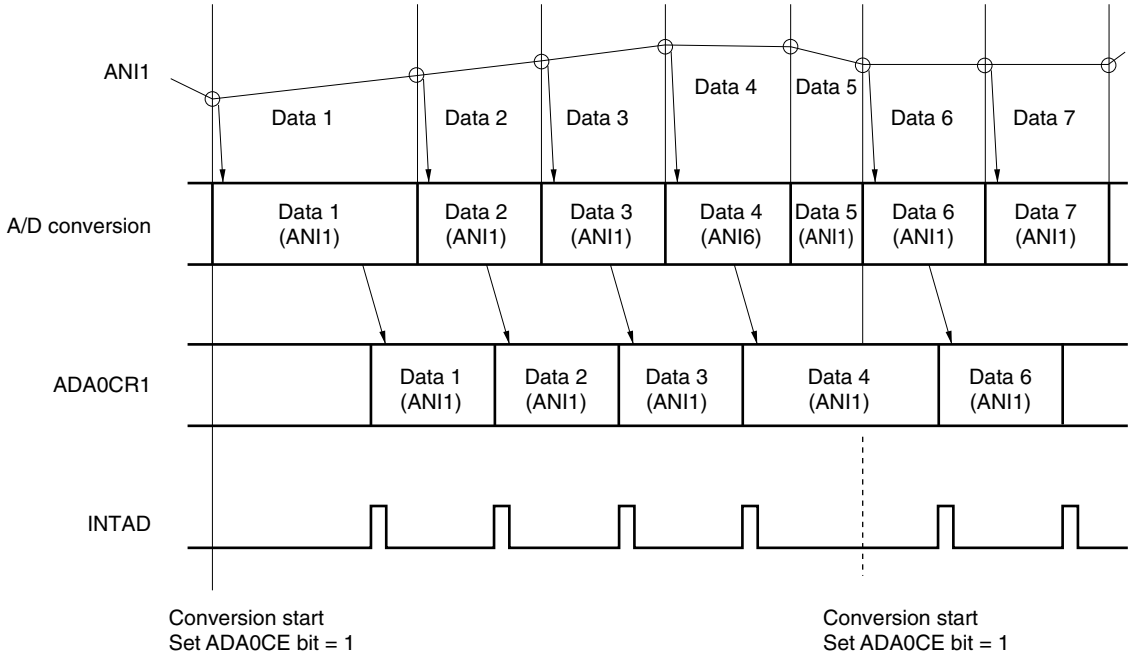
9.4.3 Operation mode

Two operation modes are available as the modes in which to set the ANI0 to ANI3 pins: continuous select mode and continuous 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 an one-to-one basis. Each time A/D conversion is 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 3).

Figure 9-11: Timing Example of Continuous Select Mode Operation (ADA0S = 01H)

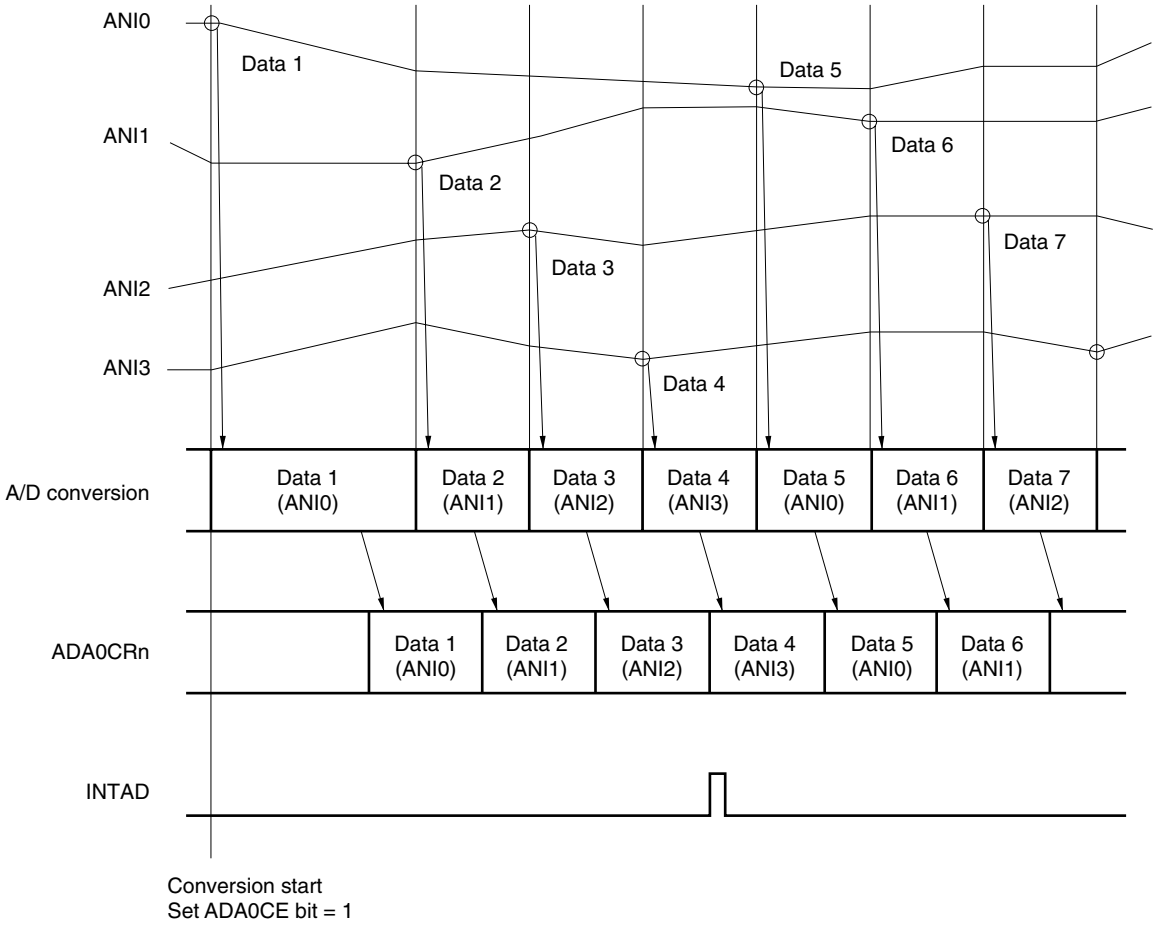


(2) Continuous scan mode

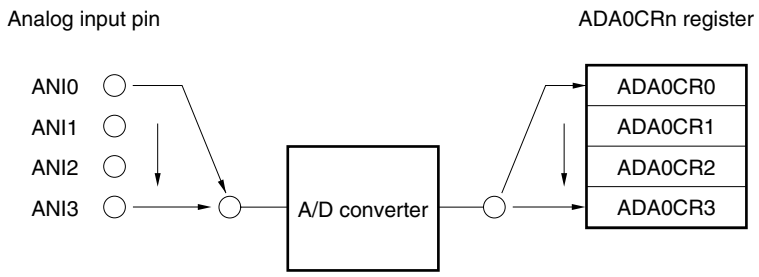
In this mode, analog input pins are sequentially selected, from the ANI0 pin to the pin specified by the ADA0S register, and their values are converted into digital values.
The result of each conversion is stored in the ADA0CRn register corresponding to the analog input pin. When conversion of the analog input pin specified 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 3).

Figure 9-12: Timing Example of Continuous Scan Mode Operation (ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



9.4.4 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 is 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.

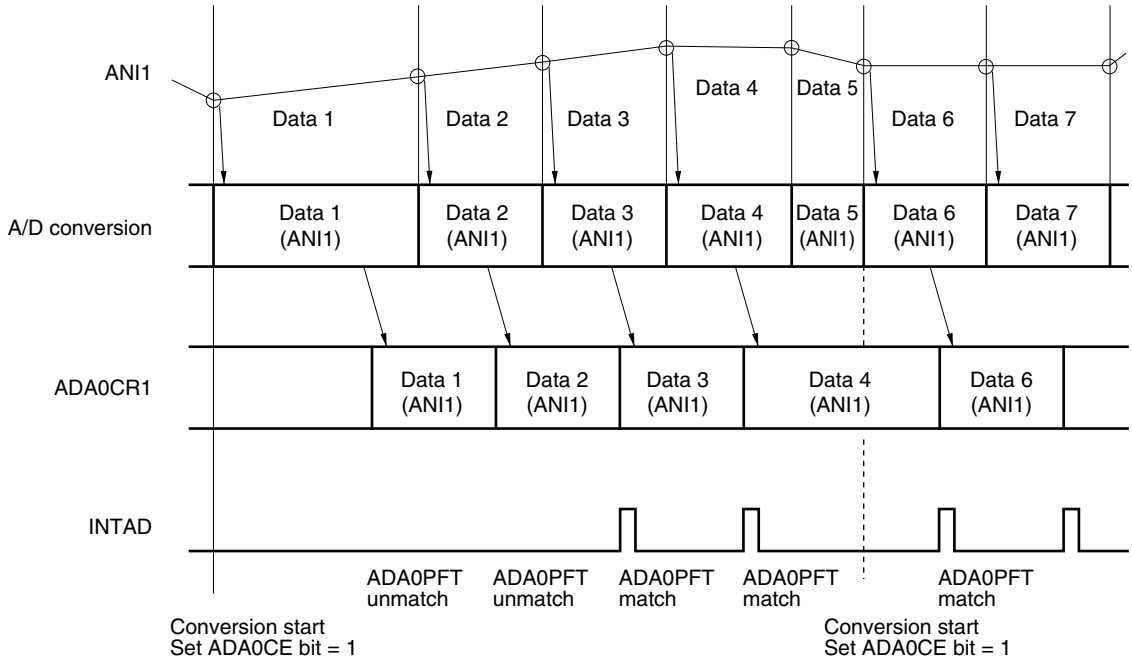
Remark: n = 0 to 3

In the power-fail compare mode, two modes are available as modes in which to set the ANI0 to ANI3 pins: continuous select mode and continuous scan mode.

(1) Continuous select mode

In this mode, the result of converting the voltage of the analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail comparison 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 3).

Figure 9-13: Timing example of continuous select mode operation (when power-fail comparison is made: ADA0S Register = 01H)



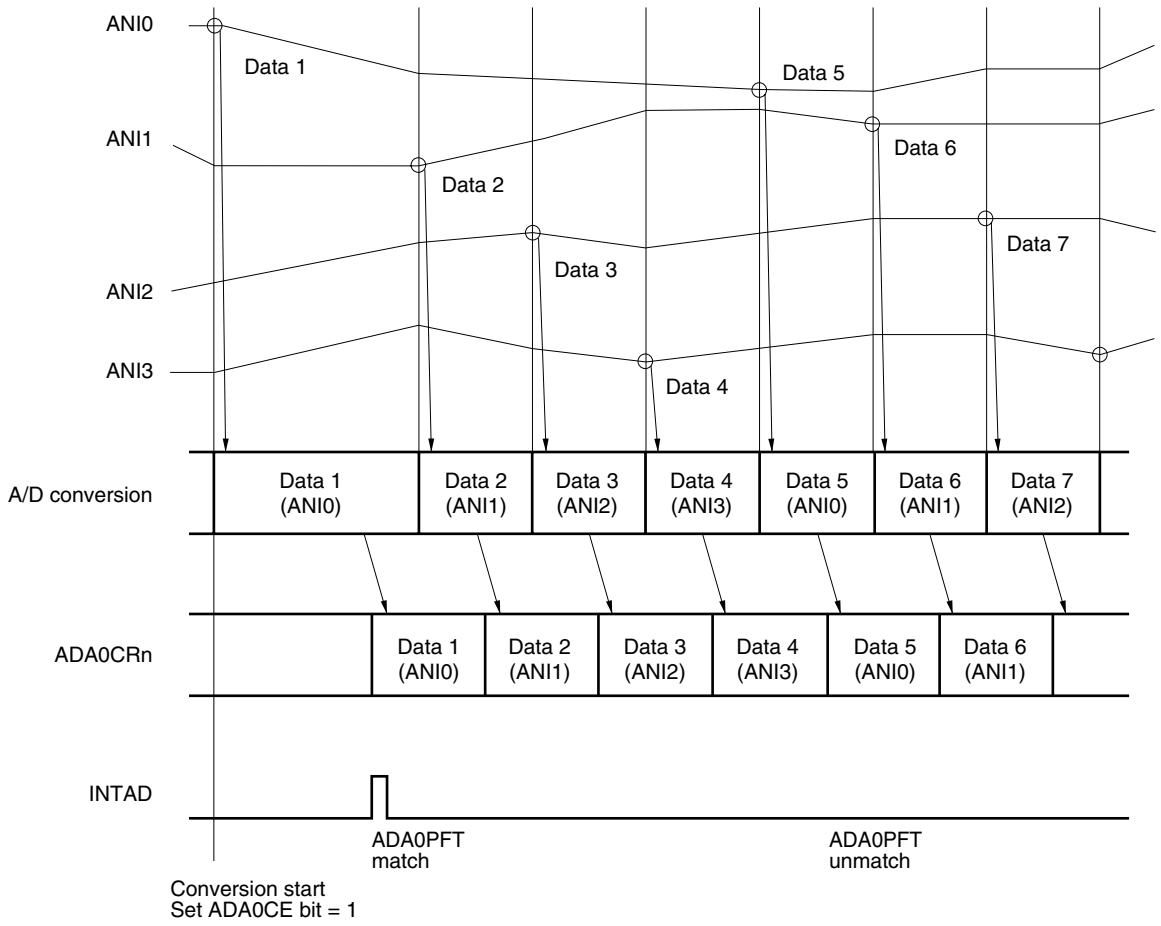
(2) Continuous scan mode

In this mode, the results of converting the voltages of the analog input pins sequentially selected from the ANI0 pin to the pin specified by the ADA0S register are stored, 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 comparison 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.

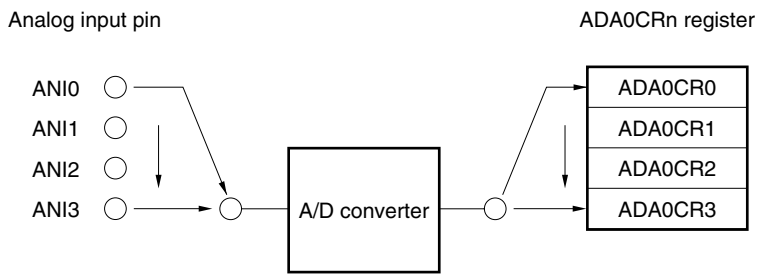
After the result of the first conversion has been stored in the ADA0CR0 register, the results of sequentially converting the voltages on the analog input pins up to the pin specified by the ADA0S register are continuously stored. After completion of conversion, the next conversion is started from the ANI0 pin again, unless the ADA0CE bit of the ADA0M0 register is cleared to 0.

Figure 9-14: Timing example of continuous scan mode operation (when power-fail comparison is made: ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



9.5 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.

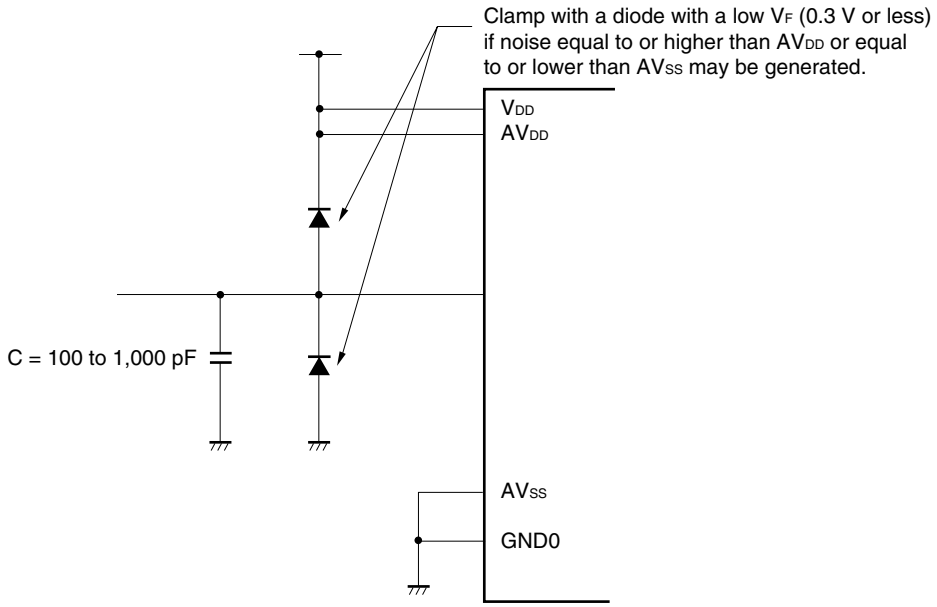
(2) Input range of ANI0 to ANI3 pins

Input the voltage within the specified range to the ANI0 to ANI3 pins. If a voltage equal to or higher than AV_{DD} 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) Counter measures against noise

To maintain the 10-bit resolution, the ANI0 to ANI3 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 9-15 is recommended.

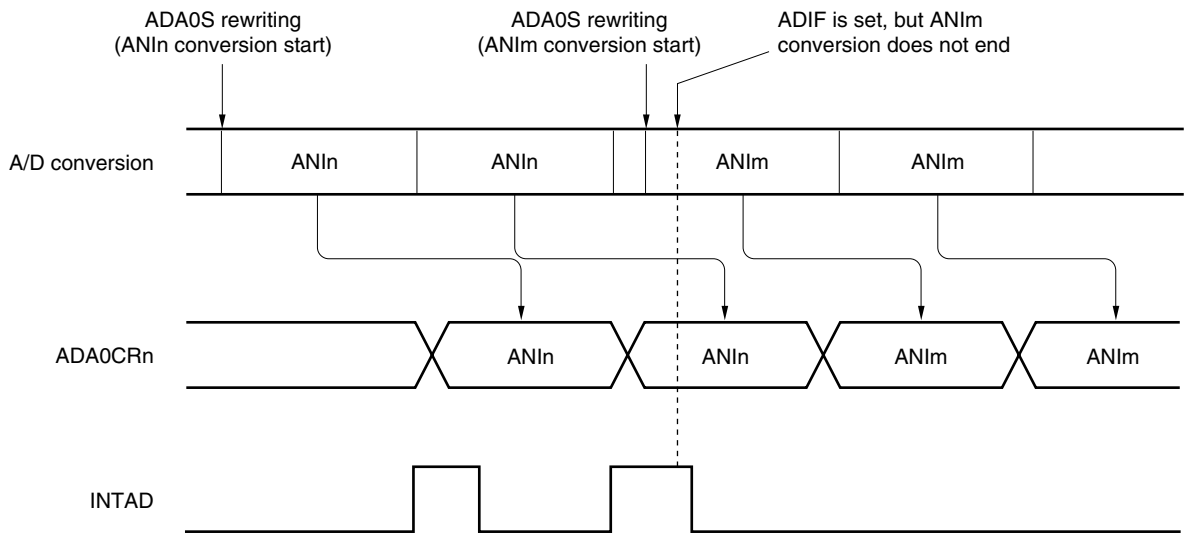
Figure 9-15: Processing of Analog Input Pin



(4) 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 is stopped, clear the ADIF flag before resuming conversion.

Figure 9-16: Generation Timing of A/D Conversion End Interrupt Request



Remark: n = 0 to 3
m = 0 to 3

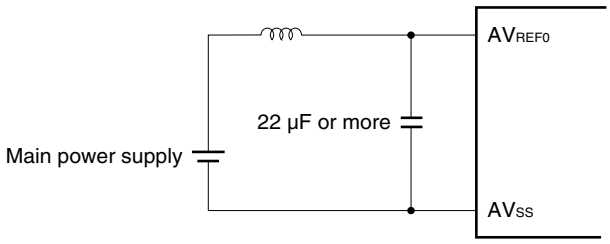
(5) AV_{DD} pin

(a) The AV_{DD} pin is used as the power supply pin of the A/D converter and also supplies power to the alternate-function ports. In an application where a backup power supply is used, be sure to supply the same voltage as V_{DD} to the AV_{DD} pin as shown in Figure 9-15, "Processing of Analog Input Pin," on page 351.

(b) The AV_{DD} pin is also used as the reference voltage pin of the A/D converter. If the source supplying power to the AV_{DD} 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_{DD} and AV_{SS} pins to suppress the reference voltage fluctuation as shown in Figure 9-17.

(c) If the source supplying power to the AV_{DD} 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 9-17: AV_{DD} Pin Processing Example



(6) 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 to the ADA0M0 to ADA0M2 and ADA0S registers. The correct conversion result may not be read at a timing different from the above.

9.6 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

$= (AV_{DD} - 0)/100$

$= AV_{DD}/100$

When the resolution is 10 bits, 1 LSB is as follows:

1 LSB $= 1/2^{10} = 1/1{,}024$

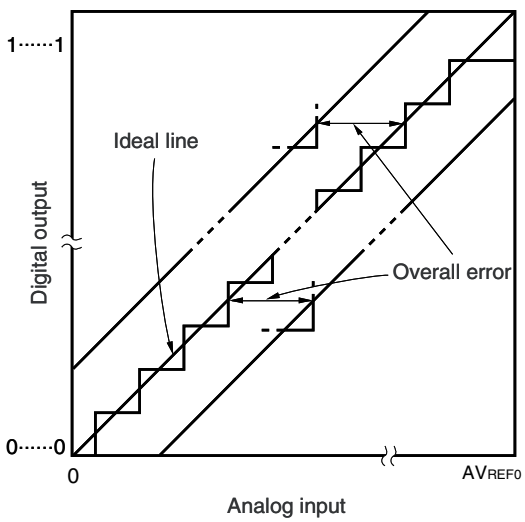
$= 0.098\%FSR$

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.

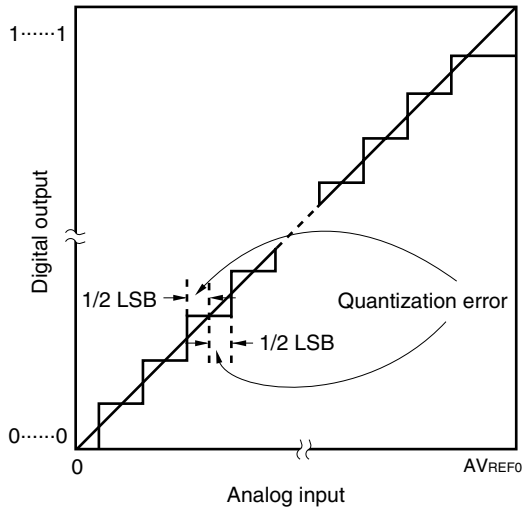
Figure 9-18: Overall Error



(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.

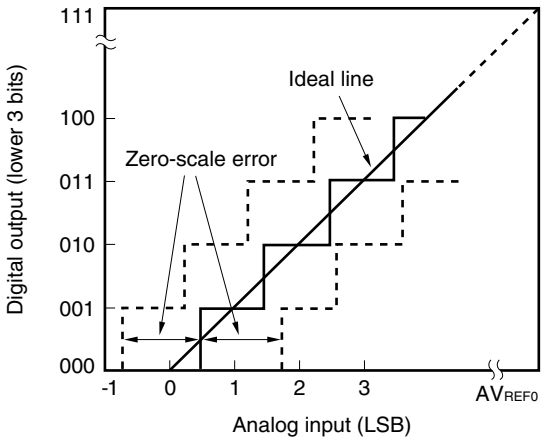
Figure 9-19: Quantization Error



(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).

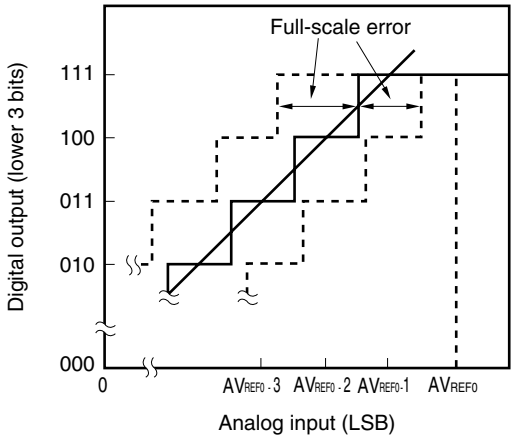
Figure 9-20: Zero-Scale Error



(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).

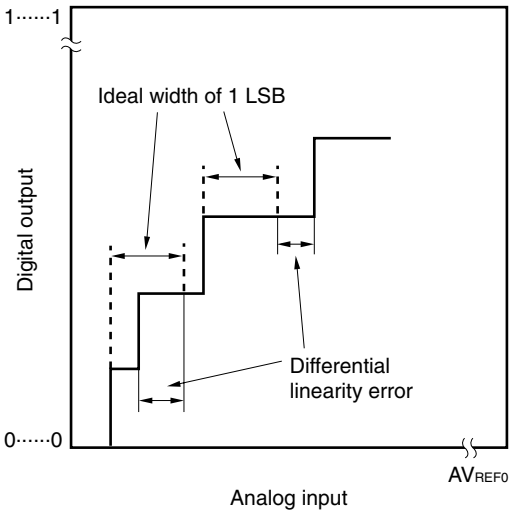
Figure 9-21: Full-Scale Error



(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.

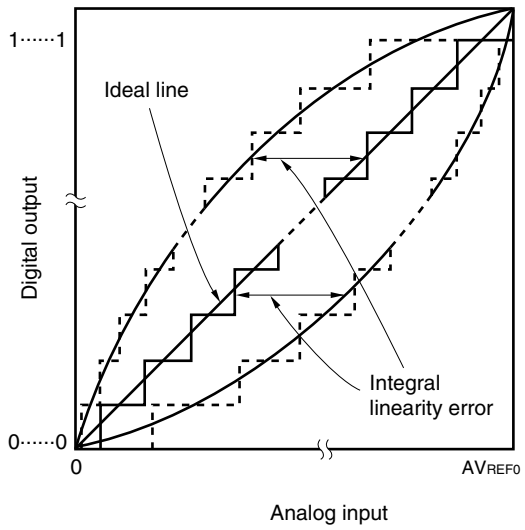
Figure 9-22: Differential Linearity Error



(7) Integral linearity error

This error indicates the extent to which the conversion characteristics differ from the ideal linear relationship. 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 9-23: Integral Linearity Error



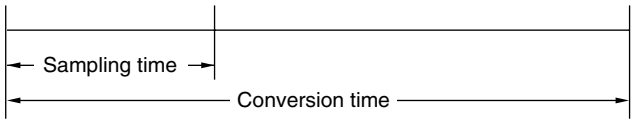
(8) Conversion time

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 9-24: Sampling Time



[MEMO]

Chapter 10 Asynchronous Serial Interface A (UARTA)

The V850E/CG3 includes 2 asynchronous serial interface A (UARTA).

10.1 Features

- Transfer rate: 300 bps to 312.5 kbps (using internal system clock of 20 MHz and dedicated baud rate generator)
- Full-duplex communication: UARTA receive data register n (UAnRX)
- UARTA transmit data register n (UAnTX)
- 2-pin configuration: TXDAn: Output pin of transmit data
- RXDAn: Input pin of receive data
- Reception error detection function
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources types:
  - Reception complete interrupt (INTUAnR):

    An interrupt is generated in the reception enabled status by ORing three types of reception errors. It is also generated when receive data is transferred from the shift register to receive buffer register n after completion of serial transfer.
  - Reception complete interrupt (INTUAnRD):

    An interrupt is generated when receive data is transferred from the shift register to receive buffer register n after completion of serial transfer.
  - Receive error interrupt (INTUAnRE):

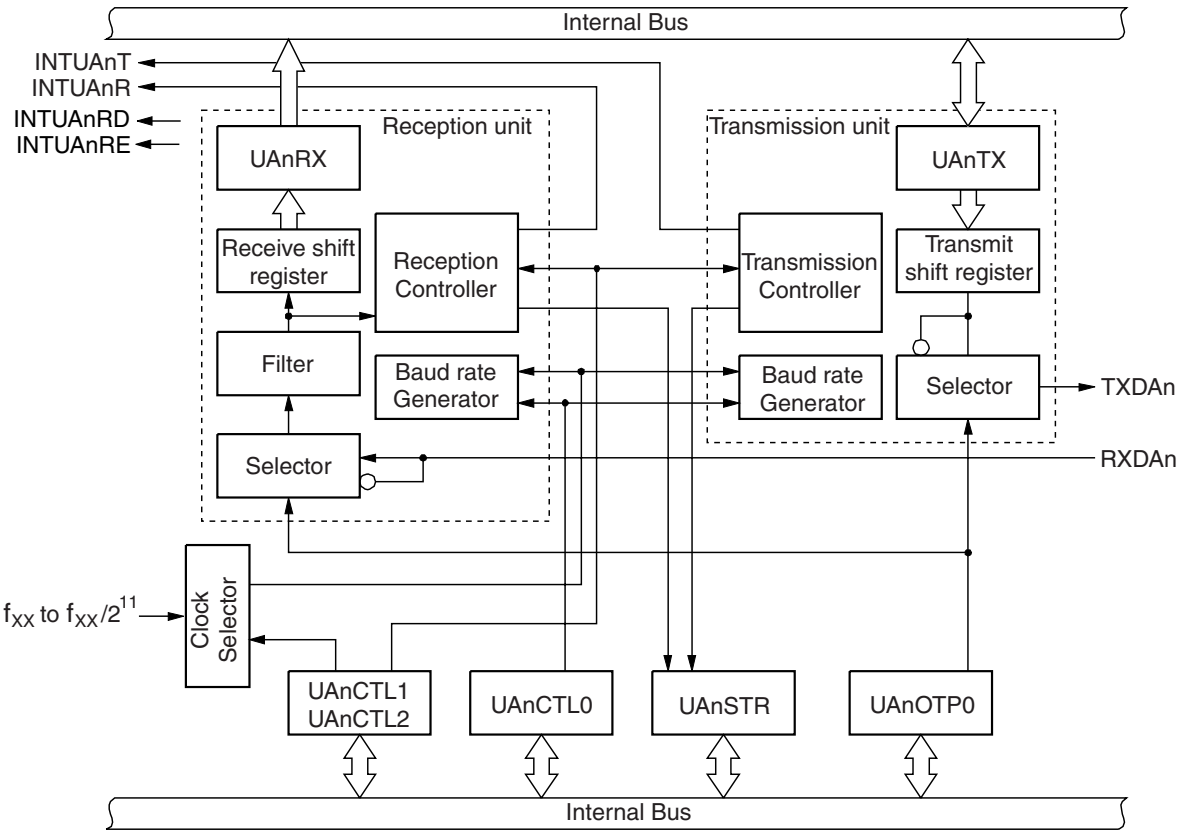
    An interrupt is generated in the reception enabled status by ORing three types of reception errors.
  - Transmission enable interrupt (INTUAnT):

    Generated when transmit data is transferred from the transmit buffer register to the shift register in the transmission enabled status.
- Character length of transmit/receive data is specified by the UAnCTL0 register.
- Character length: 7 or 8 bits
- Parity function: odd, even, 0, none
- Transmission stop bit: 1 or 2 bits
- Dedicated baud rate generator
- MSB/LSB first transfer selectable

- Transmit/receive data reversible
- 13 to 20 bits selectable for SBF (Sync Break Field) transmission in LIN (Local Interconnect Network) communication format
- 11 or more bits recognizable for SBF reception in LIN communication format
- SBF reception flag

10.2 Configuration

Figure 10-1: Block Diagram of Asynchronous Serial Interface A



Remark: n = 0 to 1

10.2.1 I/O pins

The pins of asynchronous serial interface A (UARTA) function alternately as port pins. For how to select the alternate functions, refer to the description of registers in the Chapter 15 ”Port Functions” on page 749.

Table 10-1: List of Pins of Asynchronous Serial Interface A

Pin Name	Alternate Function Pin	I/O	Function
RXDA0	P30	Input	Serial receive data input (UARTA0)
RXDA1	P32		Serial receive data input (UARTA1)
TXDA0	P31	Output	Serial transmit data output (UARTA0)
TXDA1	P93		Serial transmit data output (UARTA1)

10.2.2 Control registers

(1) UARTAn control register 0 (UAnCTL0)
The UAnCTL0 register is an 8-bit register that specifies the operation of the asynchronous serial interface.

(2) UARTAn control register 1 (UAnCTL1)
The UAnCTL1 register is an 8-bit register that selects the input clock of the asynchronous serial interface.

(3) UARTAn control register 2 (UAnCTL2)
The UAnCTL2 register is an 8-bit register that controls the baud rate of the asynchronous serial interface.

(4) UARTAn option control register 0 (UAnOPT0)
The UAnOPT0 register is an 8-bit register that controls serial transfer by the asynchronous serial interface.

(5) UARTAn status register (UAnSTR)
The UAnSTR register is a collection of flags that indicate the contents of the error when a reception error occurs. The corresponding reception error flag is set to 1 when a reception error occurs, and is reset to 0 when the UAnSTR register is read.

(6) UARTAn receive shift register
This shift register converts the serial data input to the RXDAn pin into parallel data. When data of 1 byte is received and then a stop bit is detected, the receive data is transferred to the UAnRX register.
This register cannot be directly manipulated.

(7) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit buffer register that holds receive data. When seven characters are received, 0 is stored in the higher bit (in LSB-first reception).

While reception is enabled, receive data is transferred from the UARTAn receive shift register to the UAnRX register in synchronization with completion of shift-in processing of one frame.

When the data has been transferred to the UAnRX register, a reception complete interrupt request signal (INTUAnR) is generated.

(8) UARTAn transmit shift register

The transmit shift register converts the parallel data transferred from the UAnTX register into serial data. When data of 1 byte is transferred from the UAnTX register, the data of the shift register is output from the TXDAn pin.

This register cannot be directly manipulated.

(9) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit buffer for transmit data. By writing transmit data to the UAnTX register, a transmission operation is started. When data can be written to the UAnTX register (when data of one frame is transferred from the UAnTX register to the UARTAn transmit shift register), a transmission enable interrupt request signal (INTUAnT) is generated.

10.3 Control Registers

(1) UARTAn control register 0 (UAnCTL0)

The UAnCTL0 register is an 8-bit register that controls the serial transfer operation of UARTAn.
This register can be read or written in 8-bit or 1-bit units.
Reset input sets this register to 10H

Figure 10-2: UARTAn control register 0 (UAnCTL0) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnCTL0 (n=0 to 1)	UAnPWR	UAnTXE	UAnRXE	UAnDIR	UAnPS1	UAnPS0	UAnCL	UAnSL	FFFFFA00H, FFFFFA10H	R/W	10H

UAnPWR	Control of operation of UARTAn
0	Disable clock operation (asynchronously reset UARTAn)
1	Enable clock operation.
The UAnPWR bit controls the operating clock and asynchronously resets UARTAn. When this bit is cleared to 0, the output of the TXDAn pin is fixed to the high level	

UAnTXE	Transmission operation enable
0	Stop transmission operation
1	Enable transmission operation
When the UAnTXE bit is cleared to 0, the output of the TXDAn pin is fixed to the high level. This bit is synchronized with the operating clock. When the transmission unit is initialized, therefore, set the UAnTXE bit from 0 to 1. The transmission operation will be enabled two clocks later. A value written to the UAnTXE bit is ignored when the UAnPWR bit = 0	

UAnRXE	Reception operation enable
0	Stop reception operation
1	Enable reception operation
When the UAnRXE bit is cleared to 0, the reception operation is stopped. Consequently, even if specified data is transferred, the reception complete interrupt is not output, and the UAnRX register is not updated. The UAnRXE bit is synchronized with the operating clock. When the reception unit is initialized, therefore, set the UAnRXE bit from 0 to 1. The reception operation will be enabled two clocks later. A value written to the UAnRXE bit is ignored when the UAnPWR bit = 0	

UAnDIR	Selection of transfer direction mode (MSB/LSB)
0	MSB first
1	LSB first
This bit can be rewritten only when the UAnPWR bit = 0 or when UAnTXE bit = UAnRXE bit = 0	

Figure 10-2: UARTAn control register 0 (UAnCTL0) Format (2/2)

UAnPS1	UAnPS0	Selection of parity for transmission	Selection of parity for reception
0	0	No parity output	Reception without parity
0	1	Output 0 parity	Reception with 0 parity
1	0	Output odd parity	Identified as odd parity
1	1	Output even parity	Identified as even parity
• This bit can be rewritten only when the UAnPWR bit = 0 or when the UAnTXE bit = UAnRXE bit = 0. • If “Reception with 0 parity” is selected for reception, the parity is not identified. Consequently, the UAnPE bit of the UAnSTR register is not set, and an error interrupt is not generated even if a parity error occurs. • Clear the UAnPS1 and UAnPS0 bits to “00” to execute transmission/reception in LIN format.			

UAnCL	Specification of data character length of one frame of transmit/receive data
0	7 bits
1	8 bits
This bit can be rewritten only when the UAnPWR bit = 0 or when the UAnTXE bit = UAnRXE bit = 0	

UAnSL	Specification of stop bit length of transmit data
0	1 bit
1	2 bits
This bit can be rewritten only when the UAnPWR bit = 0 or when the UAnTXE bit = UAnRXE bit = 0	

Remark: For details of the parity, refer to 10.5.9 ”Types and operation of parity” on page 384.

Caution: Set the UAnPWR bit to 1 and UAnRXE bit to 1 while a high level is being input to the RXDAn pin (when the UAnRDL bit of the UAnOP0 register is 0). If the UAnPWR and UAnRXE bits are set to 1 while a low level is being input to the RXDAn pin, reception is started.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is an 8-bit register that selects the clock of UARTAn.
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this register to 00H.

Figure 10-3: UARTAn control register 1 (UAnCTL1) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnCTL1 (n=0 to 2)	0	0	0	0	UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0	FFFFFA01H, FFFFFA11H	R/W	00H

UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0	Selection of base clock (f_{XCLK})
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	0	$f_{XX}/64$
0	1	1	1	$f_{XX}/128$
1	0	0	0	$f_{XX}/256$
1	0	0	1	$f_{XX}/512$
1	0	1	0	$f_{XX}/1024$
1	0	1	1	$f_{XX}/2048$
Other than above				Setting prohibited

Note: The ASCKA0 pin can be used only when UARTA0 (n = 0) is used. Setting this bit is prohibited when UARTA1 to UARTA3 are used.

Caution: This register can be rewritten only when the UAnPWR bit of the UAnCTL0 register = 0.

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is used to select the baud rate (serial transfer rate) clock of UARTAn.
This register can be read or written in 8-bit units.
Reset input sets this register to FFH.

Figure 10-4: UARTAn control register 2 (UAnCTL2) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnCTL2 (n=0 to 2)	UAnBRS7	UAnBRS6	UAnBRS5	UAnBRS4	UAnBRS3	UAnBRS2	UAnBRS1	UAnBRS0	FFFFFA02H, FFFFFA12H	R/W	FFH

UAnBRS7	UAnBRS6	UAnBRS5	UAnBRS4	UAnBRS3	UAnBRS2	UAnBRS1	UAnBRS0	Rated value (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	0	1	1	1	1	0	254	$f_{XCLK}/254$
1	1	1	1	1	1	1	1	255	$f_{XCLK}/255$

Remark: f_{XCLK} is the frequency of the base clock selected by the UAnCTL1 register.

Cautions: 1. This register can be rewritten only when the UAnPWR bit of the UAnCTL0 register = 0 or when the UAnTXE bit = UAnRXE bit = 0.

2. The baud rate is the serial clock divided by two.

(4) UARTAn option control register 0 (UAnOPT0)

The UAnOPT0 register is an 8-bit register that controls the serial transfer operation of UARTAn.
This register can be read or written in 8-bit or 1-bit units.
Reset input sets this register to 14H.

Figure 10-5: UARTAn option control register 0 (UAnOPT0) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnOPT0 (n=0 to 1)	UAnSFR	UAnSRT	UAnSTT	UAnSLS2	UAnSLS1	UAnSLS0	UAnTDL	UAnRDL	FFFFFA03H, FFFFFA13H	R/W	14H

UAnSFR	SBF reception flag
0	When UAnCTL0 register's UAnPWR bit = UAnRXE bit = 0. Or, on normal completion of SBF reception
1	SBF reception in progress
• This bit indicates that SBF (Sync Brake Field) is received in LIN communication. • In case of an SBF reception error, the UAnSRF bit holds the high level, and then SBF reception is started again. • The UAnSFR bit is a read-only bit.	

UAnSRT	SBF reception trigger
0	-
1 **Note**	SBF reception trigger
• This is the reception trigger bit of SBF in LIN communication. It is always 0 when read. To receive SBF, set the UAnSRT bit to 1 to enable SBF reception. • Set the UAnPWR bit and UAnRXE bit of the UAnCTL0 register to 1 and then set the UAnSRT bit.	

UAnSTT	SBF transmission trigger
0	-
1 **Note**	SBF transmission trigger
• This is the transmission trigger bit of SBF in LIN communication. It is always 0 when read. • Set the UAnPWR bit and UAnTXE bit of the UAnCTL0 register to 1 and then set the UAnSTT bit.	

Note: While the Trigger-Bit is set once, it is not allowed to reset it by application-software afterwards. The Trigger-Bits will be reset by hardware automatically.

Figure 10-5: UARTAn option control register 0 (UAnOPT0) Format (2/2)

UAnSLS2	UAnSLS1	UAnSLS0	SBF length selection
1	0	1	Outputs 13 bits (reset value)
1	1	0	Outputs 14 bits
1	1	1	Outputs 15 bits
0	0	0	Outputs 16 bits
0	0	1	Outputs 17 bits
0	1	0	Outputs 18 bits
0	1	1	Outputs 19 bits
1	0	0	Outputs 20 bits
This bit can be set when the UAnPWR bit of the UAnCTL0 register = 0 or when the UAnTXE 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 bit can be inverted by the UAnTDL bit. • This bit 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 bit can be inverted by the UAnRDL bit. • This bit can be set when the UAnPWR bit of the UAnCTL0 register = 0 or when the UAnRXE bit of the UAnCTL0 register = 0	

Remark: For details of the parity, refer to **13.5.9 Types and operation of parity**

(5) UARTAn status register (UAnSTR)

The UAnSTR register is an 8-bit register that indicates the transfer status of UARTAn and the contents of a reception error.
This bit can be read or written in 8-bit or 1-bit units, but the UAnTSF bit can only be read. The UAnPE, UAnFE, and UAnOVE bits can be read or written, but they can only be cleared by writing 0 to them, and cannot be set by writing 1 (if 1 is written to these bits, they hold the current status).
The following table shows the initialization conditions of these bits.

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	• Writing of 0 • UAnRXE bit of UAnCTL0 register = 0

Figure 10-6: UARTAn status register (UAnSTR) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnSTR (n=0 to 1)	UAnTSF	0	0	0	0	UAnPE	UAnFE	UAnOVE	FFFFFA04H, FFFFFA14H	R/W	00H

UAnTSF	Transfer status flag
0	• When UAnPWR bit of UAnCTL0 register = 0 or when UAnTXE bit of UAnCTL0 register = 0 • If next transfer data is not in UAnTX after completion of transfer
1	Writing to UAnTX register
The UAnTSF bit is always 1 when transmission is executed continuously. Before initializing the transmission unit, check that the UAnTSF bit = 0. If the transmission unit is initialized while the UAnTSF bit = 1, the transmit data cannot be guaranteed	

UAnPE	Parity error flag
0	• When UAnPWR bit of UAnCTL0 register = 0 or when UAnRXE bit of UAnCTL0 register = 0 • When 0 is written to this bit
1	When the parity of the received data does not match the parity bit
• The operation of the UAnPE bit differs depending on how the UAnPS1 and UAnPS0 bits of the UAnCTL0 register are set. • Although the UAnPE bit can be read or written, it can only be cleared by writing 0, and cannot be set by writing 1. It holds the current status when 1 is written	

Figure 10-6: UARTAn status register (UAnSTR) Format (2/2)

UAnFE	Framing error flag
0	• When UAnPWR bit of UAnCTL0 register = 0 or when UAnRXE bit of UAnCTL0 register = 0 • When 0 is written
1	When a stop bit is not detected on reception
• Only the first bit of the receive data is checked as a stop bit, regardless of the value of the UAnSL bit of the UAnCTL0 register. • Although the UAnFE bit can be read or written, it can only be cleared by writing 0, and cannot be set by writing 1. It holds the current status when 1 is written.	

UAnOVE	Overrun error flag
0	• When UAnPWR bit of UAnCTL0 register = 0 or when UAnRXE bit of UAnCTL0 register = 0 • When 0 is written
1	When receive data is set to the UAnRX register and the next reception operation is completed before that data is read
• If an overrun error occurs, the next receive data is not written to the receive buffer but discarded. • Although the UAnOVE bit can be read or written, it can only be cleared by writing 0, and cannot be set by writing 1. It holds the current status when 1 is written	

(6) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit buffer register that stores the parallel data converted by the receive shift register.
On completion of reception of 1 byte of data, the data stored in the receive shift register is transferred to the UAnRX register.
If the data length is specified to be 7 bits and when data is received with the LSB first, the receive data is transferred to bits 6 to 0 of the UAnRX register, and the MSB is always 0. If data is received with the MSB first, the receive data is transferred to bits 7 to 1 of the UAnRX register, and the LSB is always 0.
If an overrun error (UAnOVE) occurs, the receive data at that time is not transferred to the UAnRX register.
The UAnRX register is read-only, in 8-bit units.
Reset input and setting the UAnPWR bit of the UAnCTL0 register to 0 set this register to FFH.

Figure 10-7: UARTAn receive data register (UAnRX) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnRX (n=0 to 1)									FFFFFA06H, FFFFFA16H	R	FFH

(7) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit register that sets transmit data.
This register can be read or written in 8-bit or 1-bit units.
Reset input sets this register to FFH.

Figure 10-8: UARTAn transmit data register (UAnTX) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
UAnTX (n=0 to 1)									FFFFFA07H, FFFFFA17H	R/W	FFH

10.4 Interrupt Request Signals

UARTAn generates the following types of interrupt request signals:

- Reception complete & Error interrupt request signal (INTUAnR)
- Reception complete interrupt request signal (INTUAnRD)
- Transmission enable interrupt request signal (INTUAnT)
- Receive error interrupt request signal (INTUAnRE)

Table 10-2: Interrupts and their Default Priority

Interrupt	Priority
Reception complete	High
Transmission enable	Low

(1) Reception complete & error interrupt request signal (INTUAnR)

When data is shifted in to the receive shift register with reception enabled, and transferred to the UAnRX register, the reception complete interrupt request signal is generated.
This interrupt request signal can also be generated if a reception error occurs, instead of a reception error interrupt.
When the reception complete interrupt request signal is acknowledged and the data is read, read the UAnSTR register to check that the result of reception is not an error.
Reception complete interrupt request signals are not generated while reception is disabled.

(2) Reception complete interrupt request signal (INTUAnRD)

When data is shifted in to the receive shift register with reception enabled, and transferred to the UAnRX register, the reception complete interrupt request signal is generated.
Reception complete interrupt request signals are not generated while reception is disabled.

(3) Receive error interrupt request signal (INTUAnRE)

This interrupt request signal will be generated if a reception error occurs.

(4) Transmission enable interrupt request signal (INTUAnT)

The transmission enable interrupt request signal is generated when transmit data is transferred from the UAnTX register to the UARTAn transmit shift register with transmission enabled.

10.5 Operation

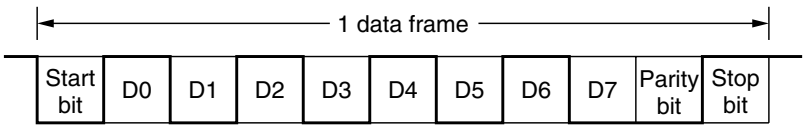
10.5.1 Data format

Full-duplex serial data is transmitted or received.
The transmit/receive data is in the format shown in Figure 10-9, consisting of a start bit, character bits, a parity bit, and 1 or 2 stop bits.
The character bit length in one data frame, parity, stop bit length, and whether data is transferred with the MSB or LSB first, are specified by the UAnCTL0 register.
The UAnTDL bit of the UAnOPT0 register is used to specify whether the signal output from the TXDAn pin is inverted or not.

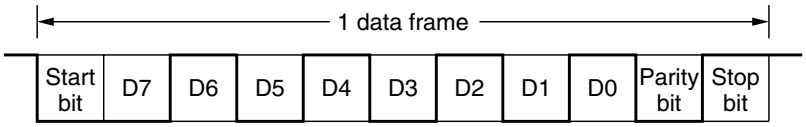
- Start bit… 1 bit
- Character bit… 7 or 8 bits
- Parity bit… Even parity, odd parity, 0 parity, or no parity
- Stop bit… 1 or 2 bits

Figure 10-9: Format of Transmit/Receive Data of UARTA (1/2)

(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 inverted

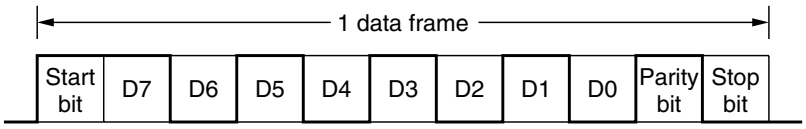
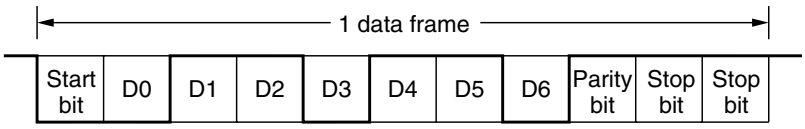
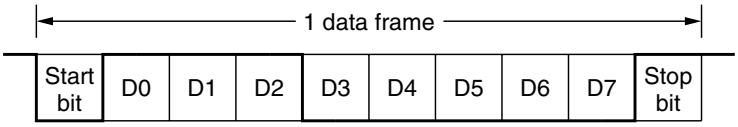


Figure 10-9: Format of Transmit/Receive Data of UARTA (2/2)

(d) 7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36H



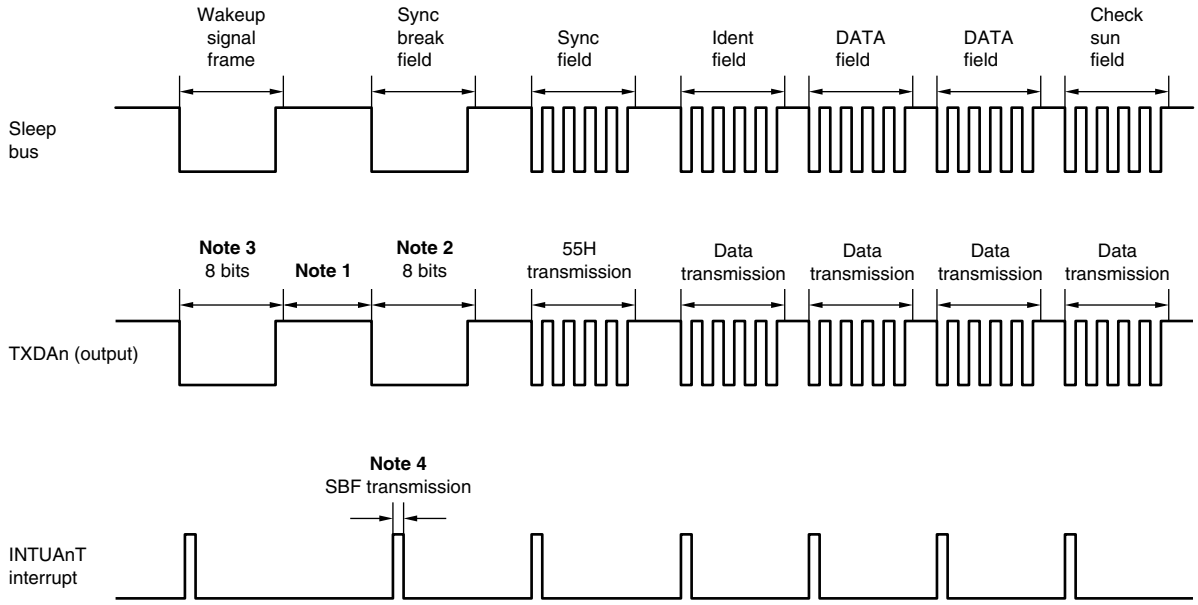
(e) 8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87H



10.5.2 SBF transmission/reception format

The V850E/CG3 has an SBF (Sync Break Field) transmission/reception control function as a LIN (Local Interconnect Network) function.

Figure 10-10: Outline of Transmission Operation of LIN



Notes: 1. The interval between each field is controlled by software.

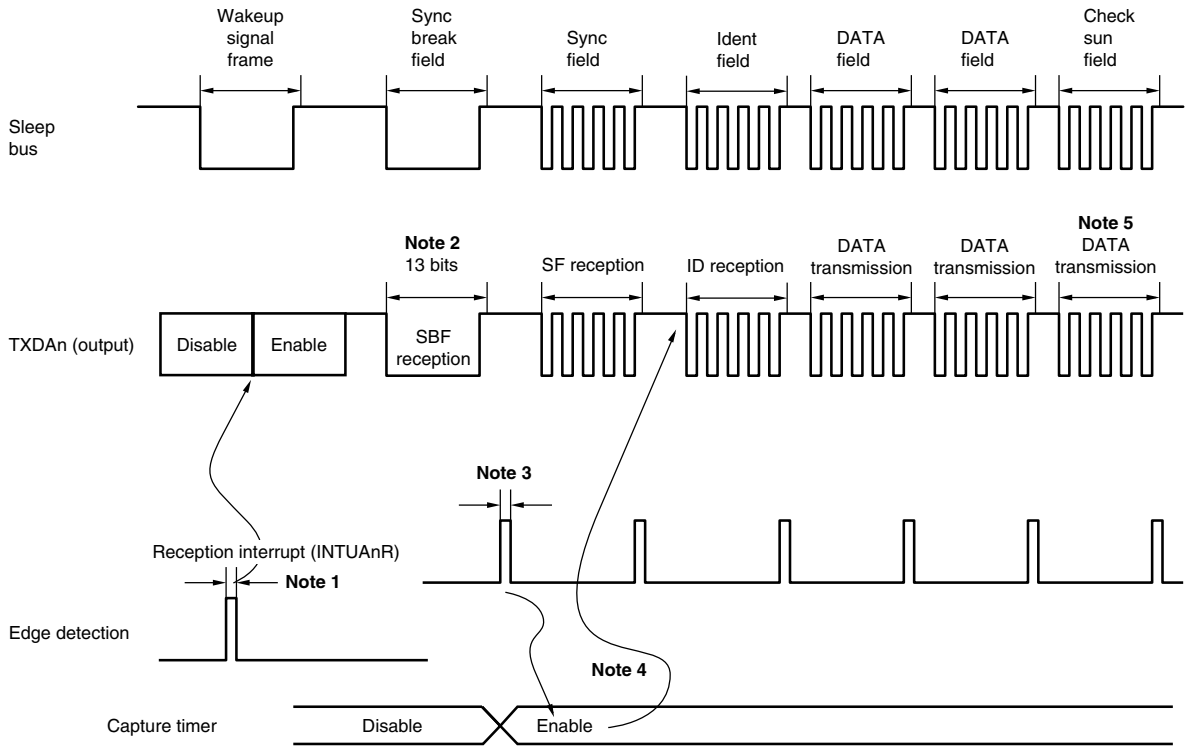
2. SBF is output by hardware. The output width is the bit length specified by the UAnSBL2 to UAnSBL0 bits of the UAnOPT0 register. If the output width must be adjusted more finely, the UAnBRST7 to UAnBRS0 bits of the UAnCTLn register can be used.

3. The wake-up signal frame is substituted by 80H transfer in the 8-bit mode.

4. A transmission enable interrupt request signal (INTUAnT) is output each time transmission is started. The INTUAnT signal is also output when SBF transmission is started.

Remark: n = 0 to 3

Figure 10-11: Outline of Reception Operation of LIN



Notes: 1. The wake-up signal is detected by the edge detector of the pin, and enables UARTA and places it in the SBF reception mode.

2. Reception is performed until the STOP bit is detected. When SBF reception of 11 bits or more is detected, it is assumed that normal SBF reception has been completed, and the interrupt signal is output. If SBF reception of less than 11 bits is detected, it is assumed that an SBF reception error has occurred. No interrupt signal is output and UARTA returns to the SBF reception mode.

3. When SBF reception is completed normally, the interrupt signal is output. The SBF reception complete interrupt enables a timer. Error detection by the UAnOVE, UAnPE, and UAnFE bits of the UAnSTR register is suppressed. Consequently, neither error detection processing of UART communication nor data transfer from the UARTAn receive shift register to UAnRX register is executed. The UARTAn receive shift register holds the default value FFH.

4. The RXDAn pin is connected to TI (capture input) of the timer, and the transfer rate and baud rate error are calculated. After SF reception, UARTA is no longer enabled. The value with the baud rate error corrected is set to the UAnCTL2 register to enable reception.

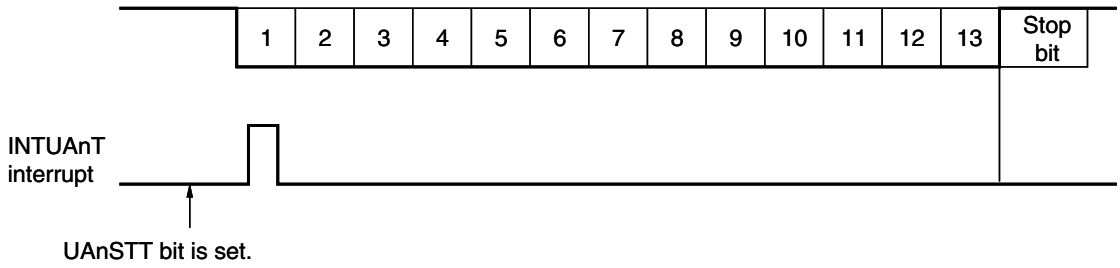
5. The checksum field is distinguished by software. After CSF reception, UARTA is initialized and the SBF mode is set again by software.

Remark: n = 0 to 1

10.5.3 SBF transmission

Transmission is enabled when the UAnPWR bit and UAnTXE bit of the UAnCTL0 register are set to 1, and SBF transmission is started by setting the SBF transmission trigger (UAnSTT bit of the UAnOPT0 register) to 1.
After that, a 13-bit to 20-bit low level, as specified by the UAnSLS2 to UAnSLS0 bits of the UAnOPT0 register, is output. A transmission enable interrupt request signal (INTUAnT) is generated when SBF transmission is started. After SBF transmission is completed, the UAnSTT bit is automatically cleared, and the UART transmission mode is restored.
The transmission operation is stopped until the data to be transmitted next is written to the UAnTX register or the SBF transmission trigger (UAnSTT bit) is set.

Figure 10-12: SBF Transmission

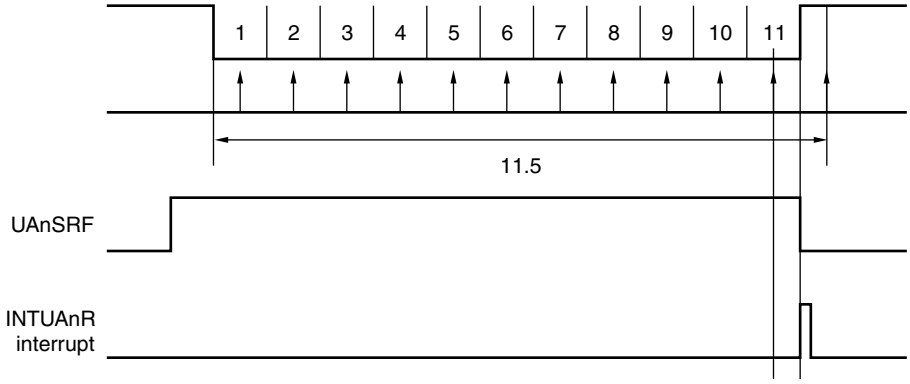


10.5.4 SBF reception

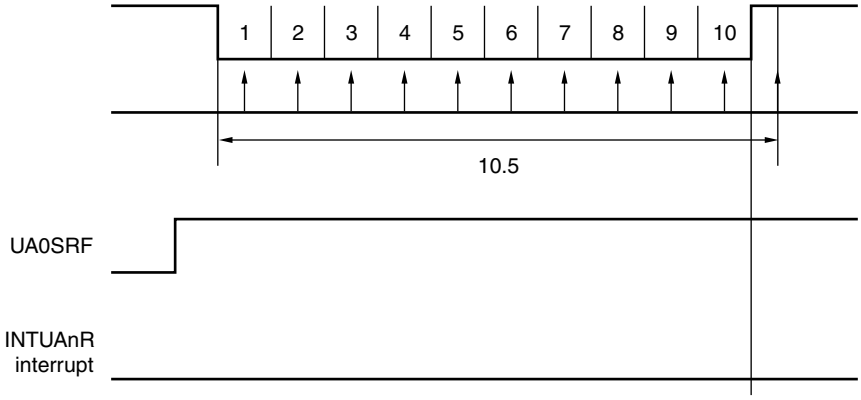
When the UAnPWR bit of the UAnCTL0 register is set to 1 and then the UAnRX bit of the UAnCTL0 register is set to 1, UARTA waits for reception.
When the SBF reception trigger (UAnSRT bit of the UAnOPT0 register) is set to 1, UARTA waits for SBF reception.
In the SBF reception waiting status, the RXDAn pin is monitored and the start bit is detected, in the same manner as in the reception wait status of UART.
When the start bit is detected, reception is started, and the internal counter counts up at the selected baud rate.
When the stop bit is received, a reception complete interrupt request signal (INTUAnR) is generated as normal processing, if the width of SBF is 11 bits or longer. The UAnSRF bit of the UAnOPT0 register is automatically cleared, and SBF reception is completed. Error detection by the UAnOVE, UAnPE, and UAnFE bits of the UAnSTR register is suppressed, and error detection processing of UART communication is not performed. Moreover, data is not transferred from the UARTAn receive shift register to the UAnRX register, and the UAnRX register holds the default value FFH. If the width of SBF is 10 bits or less, the interrupt does not occur, reception is completed, and the SBF reception mode is restored again, as error processing. At this time, the UAnSRF bit is not cleared.

Figure 10-13: SBF Reception

(a) Normal SBF reception (STOP bit is detected when SBF width exceeds 10.5 bits)



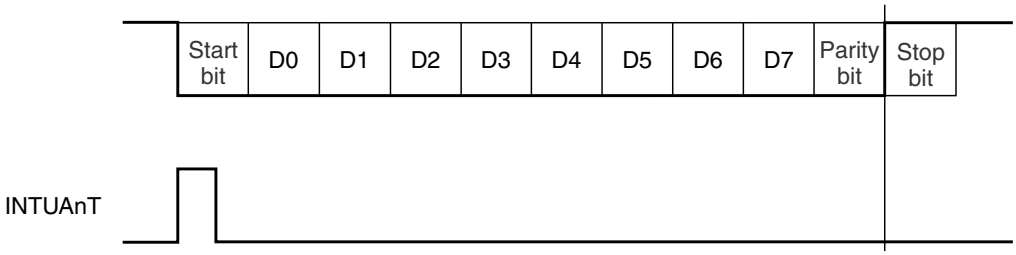
(b) SBF reception error (STOP bit is detected when SBF width is 10.5 bits or less)



10.5.5 UART transmission

When the UAnPWR bit of the UAnCTL0 register is set to 1, the TXDAn pin outputs a high level.
If the UAnTXE bit of the UAnCTL0 register is subsequently set to 1, transmission is enabled. Transmission is started by writing transmit data to the UAnTX register. A start bit, parity bit, and stop bit are automatically appended to the transmit data.
When transmission is started, the data in the UAnTX register is transferred to the UARTAn transmit shift register.
As soon as the data of the UAnTX register has been transferred to the UARTAn transmit shift register, a transmission enable interrupt request signal (INTUAnT) is generated. Then the UARTAn transmit shift register sequentially outputs the data to the TXDAn pin, starting from the LSB. When the INTUAnT signal is generated, writing the next transfer data to the UAnTX register is enabled.
By writing the data to be transmitted next to the UAnTX register during transfer, transmission can be continuously executed.

Figure 10-14: UART Transmission



10.5.6 Procedure of continuous transmission

With UARTA, the next transmit data can be written to the UAnTX register as soon as the UARTAn transmit shift register has started its shift operation. The timing at which data is transferred to the UARTAn transmit shift register can be identified by the transmission enable interrupt request signal (INTUAnT). The INTUAnT signal enables continuous transmission even while an interrupt is being serviced after transmission of 1 data frame, so that an efficient communication rate can be realized.
During continuous transmission, do not write the next transmit data to the UAnTX register before a transmit request interrupt signal (INTUAnT) is generated after transmit data is written to the UAnTX register and transferred to the UARTAn transmit shift register. If a value is written to the UAnTX register before a transmit request interrupt signal is generated, the previously set transmit data is overwritten by the latest transmit data.

Caution: While continuous transmission is being executed, execute initialization after checking that the UAnTSF bit is 0. If initialization is executed while the UAnTSF bit is 1, the transmit data cannot be guaranteed.

Figure 10-15: Processing Flow of Continuous Transfer

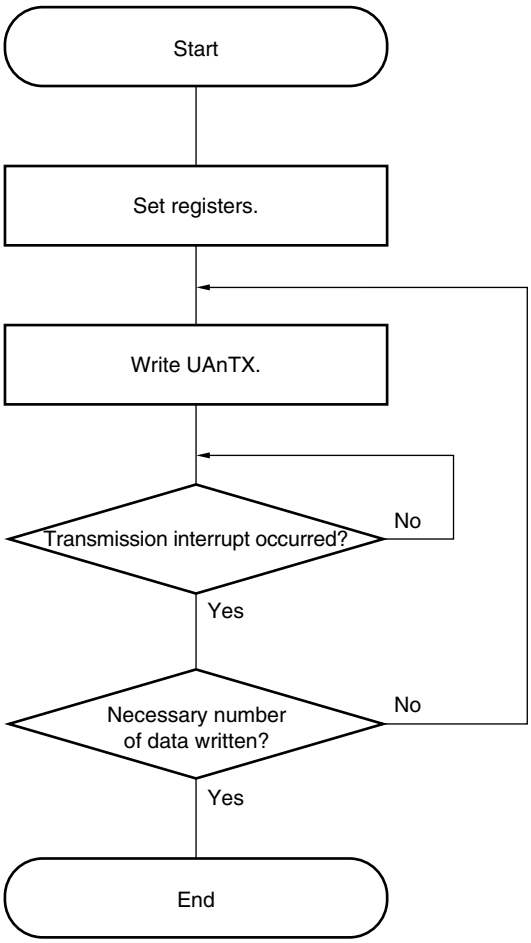
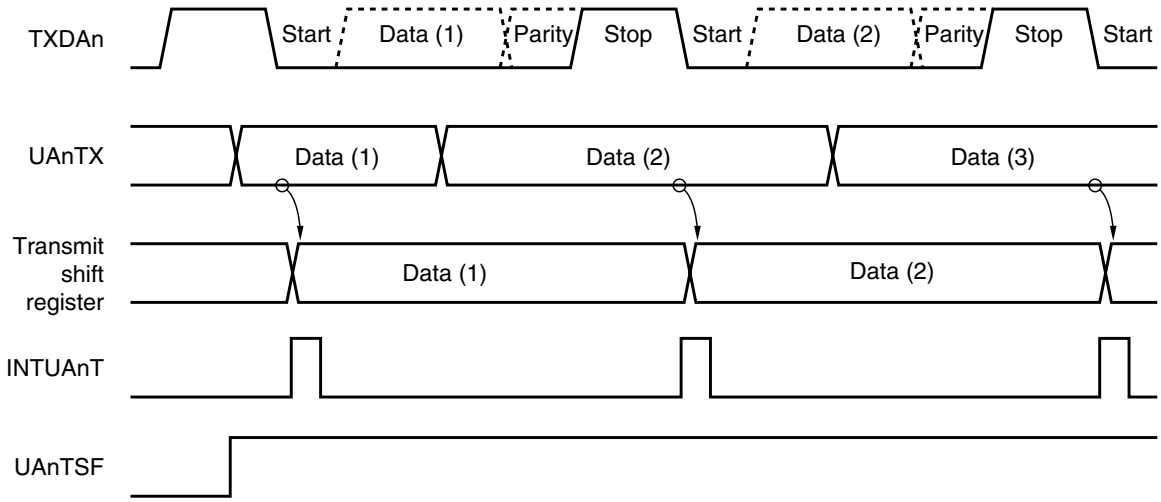
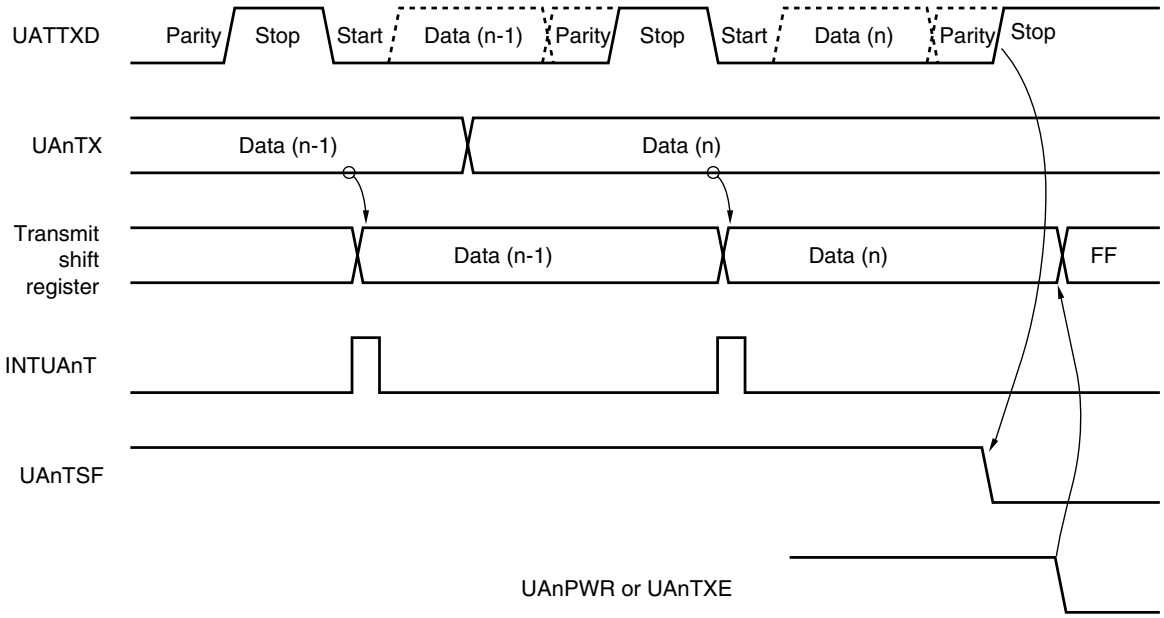


Figure 10-16: Timing of Continuous Transmission Operation

(a) Start of transfer



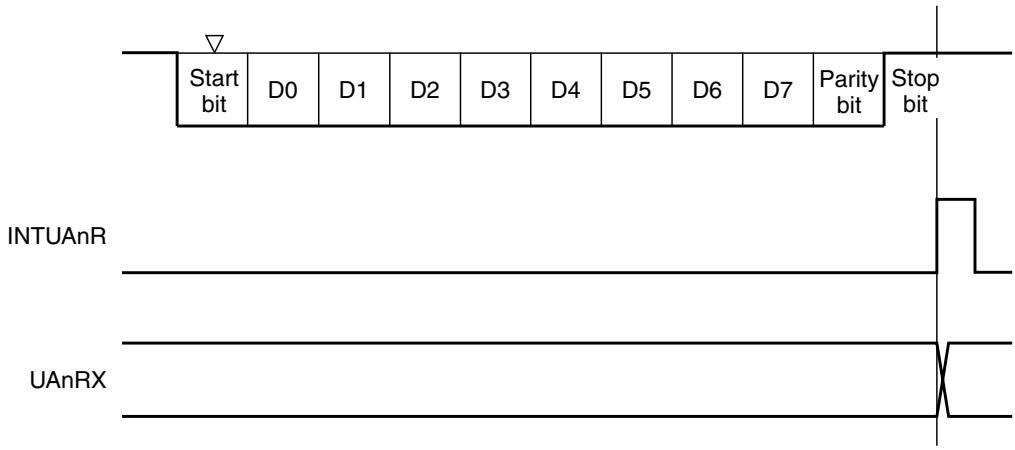
(b) End of transfer



10.5.7 UART reception

When the UAnPWR bit of the UAnCTL0 register is set to 1 and then the UAnRX bit of the UAnCTL0 register is set to 1, UARTA waits for reception. In the reception wait status, the RXDAn pin is monitored and the start bit is detected.
To recognize the start bit, a two-stage detection routine is used.
When the falling of the RXDAn pin is detected, an 8-bit counter starts counting. When the 8-bit counter has counted the set value of the UAnCTL2 register, the level of the RXDAn pin is monitored again (indicated by ▽ in Figure 10-17). If the RXDAn pin is low at this time, the start bit is recognized. When the start bit is recognized, reception is started, and serial data is sequentially stored in the UARTAn receive shift register at the selected baud rate.
When the stop bit is received, a reception complete interrupt request signal (INTUAnR) is generated and, at the same time, the data of the UARTAn receive shift register is written to the UAnRX register. If an overrun error occurs (indicated by the UAnOVE bit of the UAnSTR register), the receive data is not written to the UAnRX register.
Even if a parity error (indicated by the UAnPE bit of the UAnSTR register) or framing error (indicated by the UAnFE bit of the UAnSTR register) occurs in the middle of reception, reception continues to the reception position of the stop bit. The INTUAnR signal is generated when reception is completed.

Figure 10-17: UART Reception



Cautions: 1. Be sure to read the UAnRX register even when a reception error occurs. Unless the UAnRX register is read, an overrun error occurs when the next data is received, and the reception error status persists.

2. It is always assumed that the number of stop bits is 1 during reception. A second stop bit is ignored.

10.5.8 Reception errors

Reception errors are classified into three types: parity errors, framing errors, and overrun errors. As a result of receiving data, an error flag is set in the UAnSTR register, and a reception complete interrupt request signal (INTUAnR) is generated.
By reading the contents of the UAnSTR register in the reception error interrupt servicing, which error has occurred during reception can be checked.
The reception error flag is cleared by writing 0 to it.

Table 10-3: Reception Error Causes

Error Flag	Reception Error	Cause
UAnPE	Parity error	Received parity bit does not match setting.
UAnFE	Framing error	Stop bit is not detected.
UAnOVE	Overrun error	Next data reception is completed before data is read from receive buffer.

10.5.9 Types and operation of parity

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 a bit error in communication data. Usually, the same type of parity bit is used on both the transmission side and reception side.
Even parity and odd parity can be used to detect a “1” bit error (odd number). With zero parity and no parity, no errors are detected.

(1) Even parity

(a) During transmission

The number of bits that are “1” in the transmit data, including the parity bit, is controlled to be even. The value of the parity bit is as follows.

- Number of bits that are “1” in transmit data is odd: 1
- Number of bits that are “1” in transmit data is even: 0

(b) During reception

The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is odd, a parity error occurs.

(2) Odd parity

(a) During transmission

Opposite to even parity, the number of bits that are “1” in the transmit data, including the parity bit, is controlled to be odd. The value of the parity bit is as follows.

- Number of bits that are “1” in transmit data is odd: 0
- Number of bits that are “1” in transmit data is even: 1

(b) During reception

The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is even, a parity error occurs.

(3) 0 parity

The parity bit is cleared to 0 during transmission, regardless of the transmit data.
The parity bit is not checked during reception. Therefore, a parity error does not occur regardless of whether the parity bit is 0 or 1.

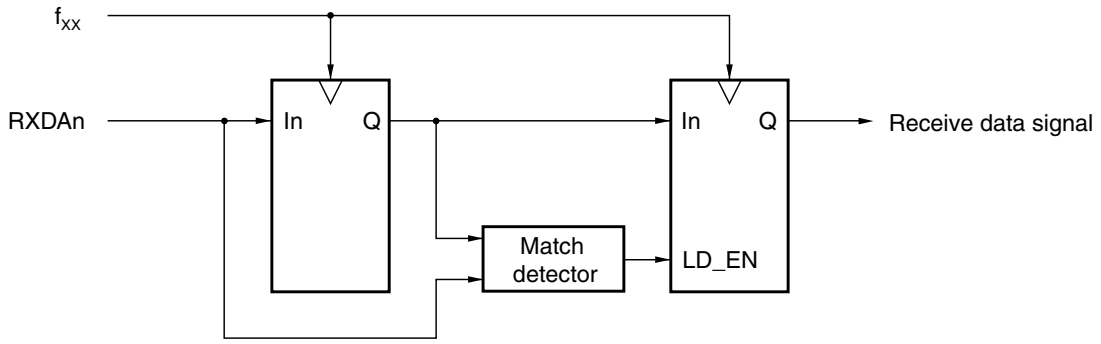
(4) No parity

No parity bit is appended to the transmit data.
Reception is performed assuming that there is no parity bit. Because no parity bit is used, a parity error does not occur.

10.5.10 Noise filter of receive data

The RXDAn pin is sampled using the internal system clock (f_{XX}).
If the sampled value is the same twice in a row, the output of the match detector changes, and the signal on the RXDAn pin is sampled as input data.
Because the circuit configuration of the noise filter is as shown in Figure 10-18, internal processing of a reception operation is delayed two clocks from the external signal status.

Figure 10-18: Noise Filter Circuit

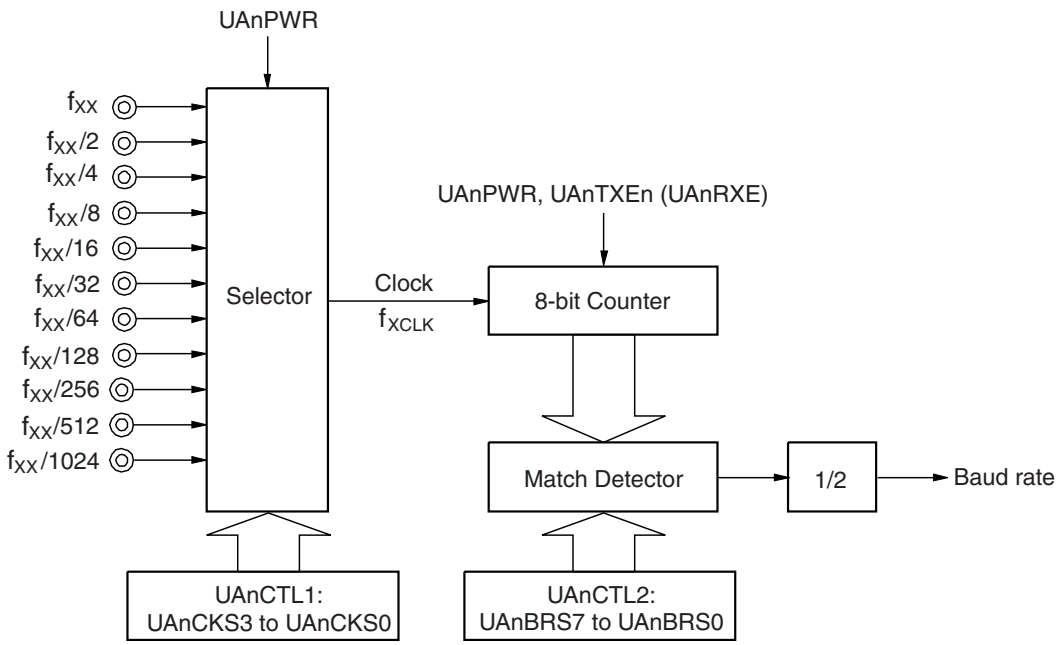


10.5.11 Dedicated Baud Rate Generator

The dedicated baud rate generator consists of a source clock selector and 8-bit programmable counters, and generates a serial clock for transmission/reception by UARTAn. The output of the dedicated baud rate generator can be selected as the serial clock on a channel by channel basis.
8-bit counters are provided separately for transmission and reception.

(1) Configuration of baud rate generator

Figure 10-19: Configuration of Baud Rate Generator



Remarks: 1. n = 0 to 1

2. f_{XX}: Internal system clock

(a) Base clock

The clock selected by the UAnCKS3 to UAnCKS0 bits of the UAnCTL1 register is supplied to the 8-bit counter when the UAnPWR bit of the UAnCTL0 register is 1. This clock is called the base clock, and its frequency is called f_{XCLK}. When the UAnPWR bit is 0, the base clock is fixed to the low level.

(b) Generation of serial clock

A serial clock can be generated in accordance with the setting of the UAnCTL1 and UAnCTL2 registers (n = 0 to 3).
The base clock is selected by using the UAnCKS3 to UAnCKS0 bits of the UAnCTL1 register.
The division ratio of the 8-bit counter can be selected by using the UAnBRS7 to UAnBRS0 bits of the UAnCTL2 register.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is used to select the clock for UARTAn.
For details, refer to **10.3 (2)"UARTAn control register 1 (UAnCTL1)" on page 365**.

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is used to select the baud rate (serial transfer rate) clock for UARTAn.
For details, refer to **10.3 (3)"UARTAn control register 2 (UAnCTL2)" on page 366**.

(4) Baud rate

The baud rate can be calculated by the following expression.

$$\text{Baud rate} = f_{CLK} / (2 \times k) \text{ [bps]}$$

Remark: f_{XCLK} = Frequency of base clock selected by UAnCKS3 to UAnCKS0 bits of UAnCTL1 register
k = Value set by UAnBRS7 to UAnBRS0 bits of UAnCTL2 register (k = 4, 5, 6, …, 255)

(5) Error of baud rate

The baud rate error is calculated by the following expression.

$$\text{Error (\%)} = \{(\text{Actual baud rate}) / (\text{desired baud rate})\} \times 100 \text{ [\%]}$$

Cautions: 1. Cautions1.Keep the baud rate error on the transmission side to within the permissible error on the reception side.

2. Keep the baud rate error on the reception side to within the range described in (7) Permissible baud rate range for reception.

Example:

Frequency of base clock = 20 MHz = 20,000,000 Hz
Set value of UAnBRS7 to UAnBRS0 bits of UAnCTL2 register = 01000001B (k = 65)
Target baud rate = 153,600 bps

Baud rate = 20,000,000/(2 × 65)
= 153,846 [bps]

Error = (153,846/153,600 – 1) × 100
= 0.160 [%]

(6) Example of baud rate setting

Table 10-4: Baud Rate Generator Set Data

Baud Rate [bps]	f_{xx} = 32 MHz			f_{xx} = 24 MHz		
	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)
300	0x0B	0x1A	0.16	0x0A	0x27	0.16
600	0x0A	0x1A	0.16	0x09	0x27	0.16
1200	0x09	0x1A	0.16	0x08	0x27	0.16
2400	0x08	0x1A	0.16	0x07	0x27	0.16
4800	0x07	0x1A	0.16	0x06	0x27	0.16
9600	0x06	0x1A	0.16	0x05	0x27	0.16
19200	0x05	0x1A	0.16	0x04	0x27	0.16
31250	0x04	0x20	0.00	0x03	0x30	0.00
38400	0x04	0x1A	0.16	0x03	0x27	0.16
76800	0x03	0x1A	0.16	0x02	0x27	0.16
153600	0x02	0x1A	0.16	0x01	0x27	0.16
312500	0x00	0x33	0.39	0x00	0x26	1.05

Remarks: 1. f_{XX} : Internal System Clock

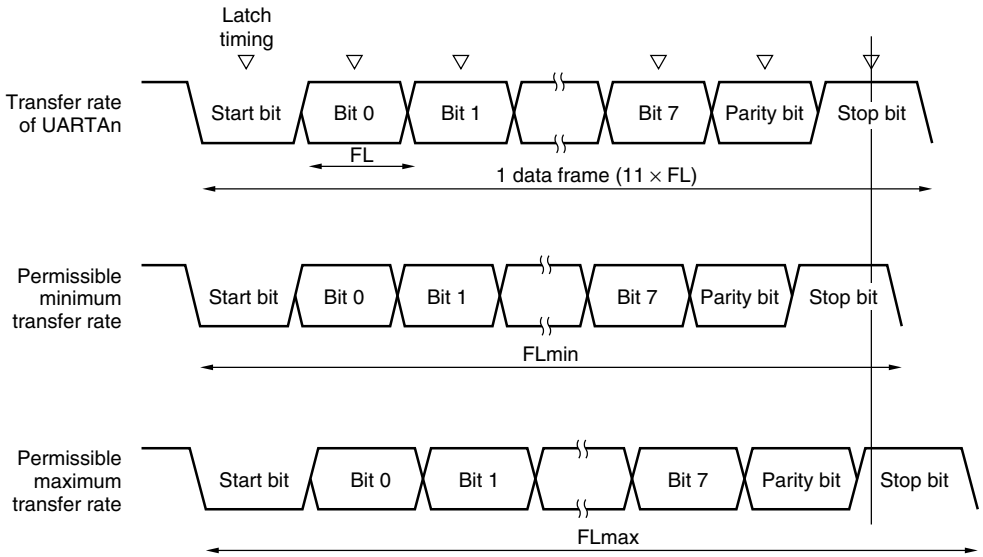
2. ERR : Baud Rate Error [%]

(7) Permissible baud rate range for reception

The permissible baud rate error during reception is shown below.

Caution: Be sure to set the baud rate error for reception to within the permissible error range, by using the expressions shown below.

Figure 10-20: Permissible Baud Rate Range for Reception



After the start bit is detected, the counter set by the UAnCTL2 register determines the latch timing of the receive data, as shown in Figure 10-20. If the last data (stop bit) is received at this latch timing, the data can be correctly received.
Assuming 11 bits of data are to be received, the theoretical baud rate is as follows.

$$FL = (Brate) - 1$$

Brate: Baud rate of UARTAn (n = 0 to 1)
k: Set value of UAnCTL2 (n = 0 to 1)
FL: 1-bit data length

Margin of latch timing: 2 clocks

Permissible minimum transfer rate:

$$Flmin = 11 \times FL - (k-2)/2k \times FL = ((21k + 2) / 2k)\ FL$$

Therefore, the maximum receivable baud rate on the transmission side is as follows.

$$BRmax = (FLmin/11)^{-1} = (22k / (21k+2))Brate$$

Similarly, the permissible maximum transfer rate can be calculated as follows:

$$(10/11) \times FLmax = 11 \times FL - ((k+2) / (2k)) \times FL = ((21k-2)/2k) \times FL$$

$$FLlmax = ((21k-2) /20k)\ FL \times 11$$

The minimum receivable baud rate on the transmission side is as follows.

$$BRmin = (Flmax/11)^{-1} = (20k/(21k+2))\ Brate$$

The permissible error of the baud rate from the transmission side of UARTAn can be calculated by the above expressions of the minimum/maximum baud rate. The result is as shown in Table 10-5.

Table 10-5: Permissible Maximum/Minimum Baud Rate Error

Division Ratio (k)	Permissible Maximum Baud Rate Error	Permissible Minimum 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%

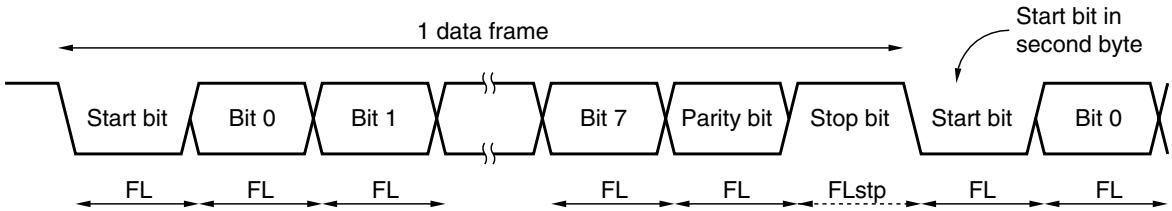
Remarks: 1. The reception accuracy is dependent upon the number of bits in 1 frame, input clock frequency, and division ratio (k). The higher the input clock frequency and the higher the division ratio (k), the higher the accuracy.

2. k: Set value of UAnCTL2 (n = 0 to 3)

(8) Transfer rate for continuous transmission

The transfer rate from the stop bit to the start bit of the next data is extended two clocks when continuous transmission is executed. However, the timing on the reception side is initialized when the start bit is detected, and therefore, the transfer result is not affected.

Figure 10-21: Transfer Rate for Continuous Transmission



Where 1 bit data length is FL, stop bit length is FLstp, and base clock frequency is f_{XCLK}, the stop bit length can be calculated by the following expression.

$$FLstp = FL + 2/f_{XCLK}$$

Therefore, the transfer rate for continuous transmission is as follows.

$$\text{Transfer rate} = 11 \times FL + 2/f_{XCLK}$$

[MEMO]

Chapter 11 3-Wire Serial Interface (CSIB)

The V850E/CG3 includes 3x 3-wire serial interface (CSIB).

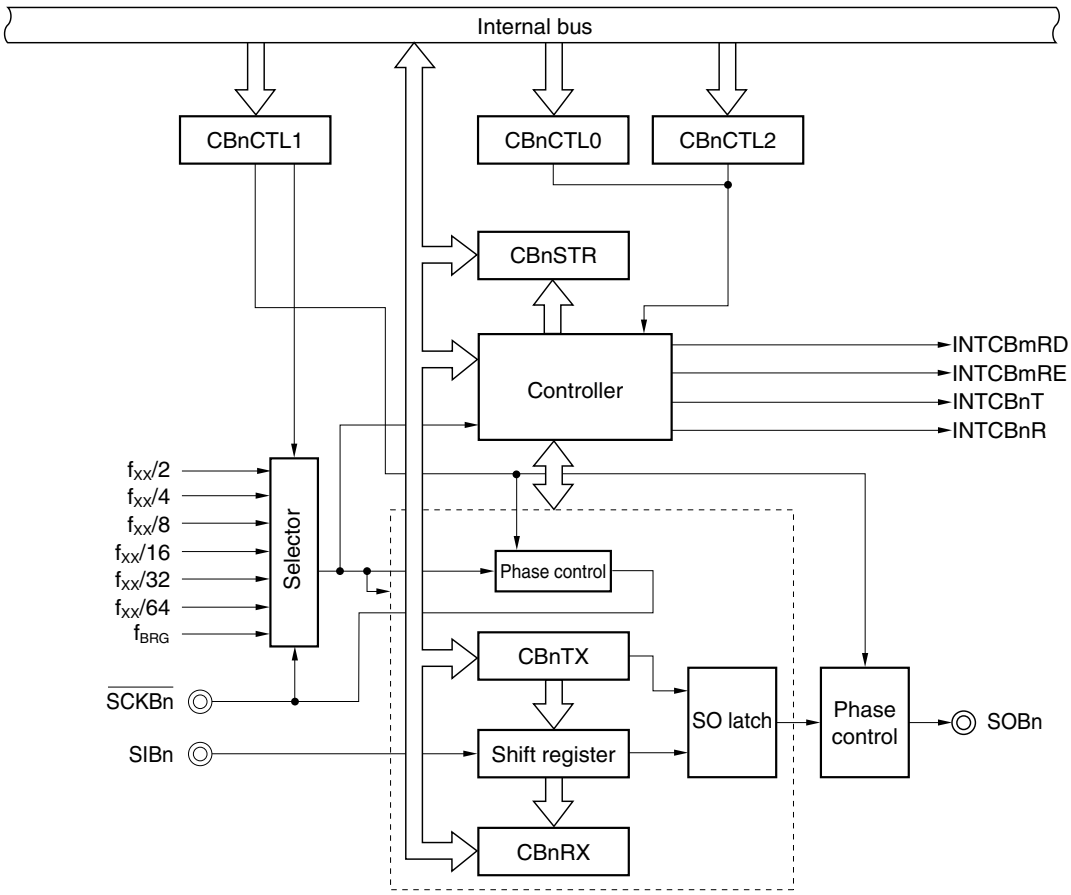
11.1 Features

- Master mode and slave mode selectable
- 3-wire serial interface for 8-bit to 16-bit transfer
- Interrupt request signals (INTCBnT, INTCBnR, INTCBnRD, INTCBnRE)
- Serial clock and data phase selectable
- Transfer data length selectable from 8 to 16 bits in 1-bit units
- Data transfer with MSB- or LSB-first selectable
- 3-wire
  - SOBn: Serial data output
  - SIBn: Serial data input
  - SCKBn: Serial clock I/O
- Transmission mode, reception mode, and transmission/reception mode selectable

Remark: n = 0 to 2

11.2 Configuration

Figure 11-1: Block Diagram of 3-Wire Serial Interface



Remark: n = 0 to 2

11.3 I/O Pins

The pins of the 3-wire serial interface (CSIB) function alternately as port pins. For how to select the alternate function, refer to the descriptions of the registers in **Chapter 15 ”Port Functions” on page 749**.

Table 11-1: List of 3-Wire Serial Interface Pins

Pin Name	Alternate-Function Pin	I/O	Function
SIB0	P40	Input	Serial receive data input (CSIB0)
SIB1	P43		Serial receive data input (CSIB1)
SIB2	P34		Serial receive data input (CSIB2)
SOB0	P41	Output	Serial transmit data input (CSIB0)
SOB1	P44		Serial transmit data input (CSIB1)
SOB2	P35 / SDA		Serial transmit data input (CSIB2)
$\overline{\text{SCKB0}}$	P42	I/O	Serial clock I/O (CSIB0)
$\overline{\text{SCKB1}}$	P45		Serial clock I/O (CSIB1)
$\overline{\text{SCKB2}}$	P36 / $\overline{\text{SCL}}$		Serial clock I/O (CSIB2)

Remark: The number of channels differs depending on the product.

11.3.1 Control registers

(1) CSIBn control register 0 (CBnCTL0)
The CBnCTL0 register is an 8-bit register that specifies the operation of the 3-wire serial interface.

(2) CSIBn control register 1 (CBnCTL1)
The CBnCTL1 register is an 8-bit register that selects the transmission/reception timing and input clock of the 3-wire serial interface.

(3) CSIBn control register 2 (CBnCTL2)
This is an 8-bit register that controls the number of serial transfer bits of CSIB.

(4) CSIBn status register (CBnSTR)
The CBnSTR register is a collection of flags that indicate the nature of a reception error that has occurred. Each error flag is set to 1 if the corresponding reception error occurs.

(5) CSIBn receive data register (CBnRX)
The CBnRX register is a 16-bit buffer register that holds receive data.
When data is transferred to the CBnRX register, a reception complete interrupt request signal (INTCBnR) is generated.

(6) CSIBn transmit data register (CBnTX)
The CBnTX register is a 16-bit buffer register that holds transmit data.
When data is transferred to the CBnTX register, a transmission enable interrupt request signal (INTCBnT) is generated.

11.4 Control Registers

(1) CSIBn control register 0 (CBnCTL0)

This register controls the serial transfer operation of CSIB.
This register can be read or written in 8-bit or 1-bit units.
Reset input sets this register to 01H.

Figure 11-2: CSIBn control register 0 (CBnCTL0) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnCTL0 (n=0 to 2)	CBnPWR	CBnTXE **Note**	CBnRXE **Note**	CBnDIR **Note**	0	0	CBnTMS **Note**	CBnSCE	FFFFFD00H, FFFFFD10H, FFFFFD20H	R/W	01H

CBnPWR	Specification of CSIB operation stop or enable
0	Stop clock operation (asynchronously reset CSIBn)
1	Enable clock operation.
The CBnPWR bit controls the operating clock of CSIB and resets the internal circuit	

CBnTXE**Note**	Specification of transmission operation stop or enable
0	Stop transmission
1	Enable transmission
When the CBxTXE bit is cleared to 0, the serial output pin SOBn is fixed to the low level and communication is stopped	

CBnRXE**Note**	Reception operation enable
0	Stop reception
1	Enable reception
When the CBnRXE bit is cleared to 0, because reception is stopped, the reception complete interrupt is not output and the receive data in the CBnRX register is not updated even if the specified data is transferred.	

CBnDIR**Note**	Specification of transfer direction mode (MSB/LSB)
0	MSB first
1	LSB first

Note: This register can be rewritten only when the CBnPWR bit = 0. However, the CBnPWR bit can also be set to 1 at the same time as rewriting these bits.

Figure 11-2: CSIBn control register 0 (CBnCTL0) Format (2/2)

CBnTMS[Note]	Specification of transfer mode
0	Single transfer mode
1	Continuous transfer mode
When the CBnTMS bit = 0, the single transfer mode is set in which continuous transmission/reception is not supported. Even when only transmission is executed, an interrupt is output on completion of reception transfer.	

CBnSCE	Specification of start transfer disable or enable
0	Stop clock output
1	Enable clock output
The CBnSCE bit controls starting of the transfer operation in the master mode. If only reception is enabled in the single transfer mode (CBnRXE bit = 1, CBnTXE bit = 0), the reception operation is started when the CBnRX register is read. To read the last receive data, clear the CBnSCE bit to 0, read the last receive data, and then disable starting the next reception operation. Similarly, if only reception is enabled in the continuous transfer mode (CBnTMS bit = 1), starting the reception operation can be disabled after completion of reception of the last receive data, by clearing the CBnSCE bit to 0 one clock before reception of the last receive data is completed. After the last data is read, reception is enabled by setting the CBnSCE bit to 1 again and reading the CBnRX register. In the slave reception mode, the CBnSCE bit also enables the internal operating clock, and therefore, set the CBnSCE bit to 1.	

Note: This register can be rewritten only when the CBnPWR bit = 0. However, the CBnPWR bit can also be set to 1 at the same time as rewriting these bits.

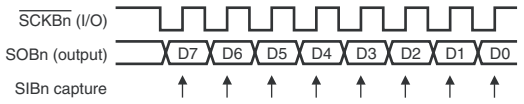
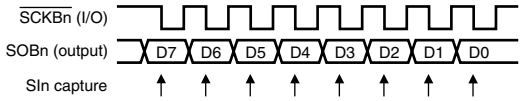
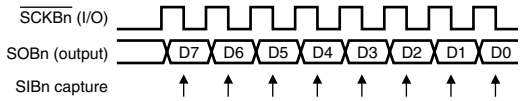
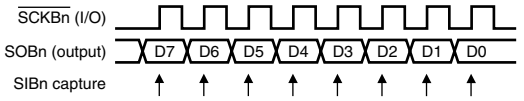
(2) CSIBn control register 1 (CBnCTL1)

This is an 8-bit register that selects the transmission/reception timing and input clock of CSIBn.
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 only when the CBnPWR bit of the CBnCTL0 register is 0 or when the CBnTXE and CBnRXE bits are 0.

Figure 11-3: CSIBn control register 1 (CBnCTL1) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnCTL1 (n=0 to 2)	0	0	0	CBnCKP	CBnDAP	CBnCKS2	CBnCKS1	CBnCKS0	FFFFFD01H, FFFFFD11H, FFFFFD21H	R/W	00H

CBnCKP	CBnDAP	Specification of transmission/reception timing of data of 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	BRG	Master mode
1	1	1	External clock (SCKBn)	Slave mode

Note: f_{BRG}: Output clock frequency of prescaler 3

For details of the prescaler, refer to 11.7 ”Prescaler 3” on page 420.

(3) CSIBn control register 2 (CBnCTL2)

This is an 8-bit register that controls the number of serial transfer bits of CSIB.
It 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 when the CBnPWR bit of the CBnCTL0 register = 0 or when the CB0TXE and CB0RXE bits = 0.

Figure 11-4: CSIBn control register 2 (CBnCTL2) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnCTL2 (n=0 to 2)	0	0	0	0	CBnCL3	CBnCL2	CBnCL1	CBnCL0	FFFFFD02H, FFFFFD12H, FFFFFD22H	R/W	00H

CBnCL3	CBnCL2	CBnCL1	CBnCL0	Bit length of serial register
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	x	x	x	16 bits

Caution: If the number of transfer bits is not 8 or 16, prepare data, justifying it to the least significant bit of the CBnTX or CBnRX register

(4) CSIBn status register (CBnSTR)

This is an 8-bit register that indicates the status of CSIB.
Although this register can be read or written in 8-bit or 1-bit units, the CBnSTF flag is read-only.
Reset input clears this register to 00H.
Clearing the CBnPWR bit of the CBnCTL0 register to 0 also initializes this register.

Figure 11-5: CSIBn status register (CBnSTR) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnSTR (n=0 to 2)	CBnTSF	0	0	0	0	0	0	CBnOVE	FFFFFD03H, FFFFFD13H, FFFFFD23H	R/W	00H

CBnTSF	Transfer operation status flag
0	Idle status
1	Operating status
This bit is set when data is prepared in the CBnTX register for transmission. It is set when dummy data is read from the CBnRX register for reception. It is cleared when the edge of the last clock is completed **Caution: Do not start next transfer request while CBnTSF flag is set, otherwise transfer is not started and communication is closed.**	

CBnOVE	Overrun error flag
0	No overrun
1	Overrun
• An overrun error occurs if the next reception is started without the CPU reading the value of the receive buffer during reception or on completion of a reception operation. The CBnOVE flag indicates occurrence of this overrun error. • The OBnOVE flag is cleared when 0 is written to it. It is not set when 1 is written to it.	

(5) 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.
If reception is enabled, a reception operation is started when the CBnRX register is read.
If the transfer data length is 8 bits, the lower 8 bits of the CBnRX register are read-only in 8-bit units as the CBnRXL register.
Reset input clears this register to 0000H.
Clearing the CBnPWR bit of the CBnCTL0 register to 0 also initializes this register.

Figure 11-6: CSIBn receive data register (CBnRX) Format

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnRX (n=0 to 2)																	FFFFFD04H, FFFFFD14H, FFFFFD24H	R/W	0000H

(6) CSIBn transmit data register (CBnTX)

The CBnTX register is a 16-bit buffer register to which transfer data of CSIB is written.
This register can be read or written in 16-bit units.
If transmission is enabled, a transmission operation is started when the CBnTX register is read.
If the transfer data length is 8 bits, the lower 8 bits of the CBnTX register can be read or written in 8-bit units as the CBnTXL register.
Reset input clears this register to 0000H.
Clearing the CBnPWR bit of the CBnCTL0 register to 0 also initializes this register.

Figure 11-7: CSIBn transmit data register (CBnTX) Format

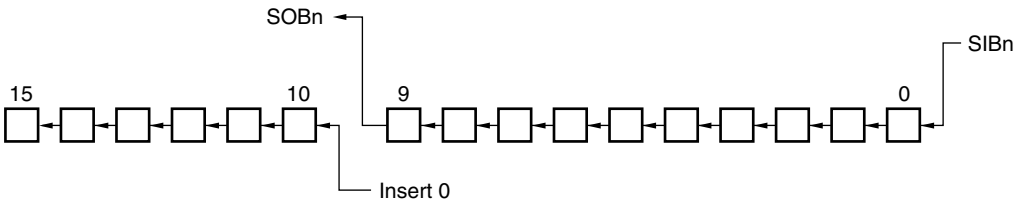
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	R/W	After reset
CBnTX (n=0 to 2)																	FFFFFD06H, FFFFFD16H, FFFFFD26H	R/W	0000H

11.4.1 Transfer data length change function

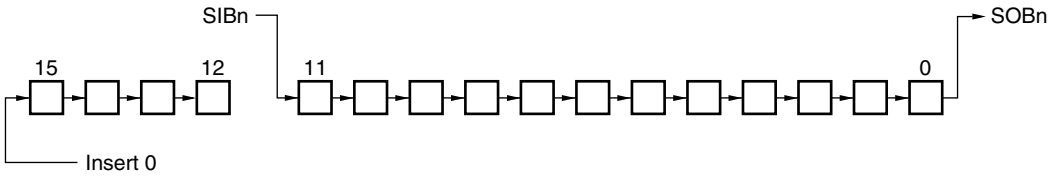
The transfer data length of CSIB can be changed from 8 to 16 bits in 1-bit units by using the CBnCL3 to CBnCL0 bits of the CBnCTL2 register.
If a transfer data length of other than 16 bits is specified, set data in the CBnTX or CBnRX register, justifying to the least significant bit, regardless of whether the first transfer bit is the MSB or LSB. Any data can be set to the higher bits that are not used, but the receive data is 0 after serial transfer.

Figure 11-8: Changing Transfer Data Length

(a) When transfer bit length = 10 bits, MSB first



(b) When transfer bit length = 12 bits, LSB first



11.4.2 Interrupt request signals

CSIBn can generate the following 4 types of interrupt request signals.

- Reception complete & error interrupt request signal (INTCBnR)
- Reception complete interrupt request signal (INTCBnRD)
- Receive error interrupt request signal (INTCBnRE)
- Transmission enable interrupt request signal (INTCBnT)

Of these two interrupt request signals, the reception complete interrupt request signal has the higher priority by default, and the priority of the transmission enable interrupt request signal is lower.

Table 11-2: Interrupts and their Default Priority

Interrupt	Priority
Reception complete	High
Transmission enable	Low

(1) Reception complete & error interrupt request signal (INTCBnR)

When receive data is transferred to the CBnRX register while reception is enabled, the reception complete interrupt request signal is generated.
This interrupt request signal can also be generated if a reception error occurs, instead of a reception error interrupt.
When the reception complete interrupt request signal is acknowledged and the data is read, read the CBnSTR register to check that the result of reception is not an error.
The reception complete interrupt request signal is not generated while reception is disabled.

(2) Reception complete interrupt request signal (INTCBnRD)

When receive data is transferred to the CBnRX register while reception is enabled, the reception complete interrupt request signal is generated.

(3) Receive error interrupt request signal (INTCBnRE)

This interrupt request signal is generated when a reception error occurs.

(4) Transmission enable interrupt request signal (INTCBnT)

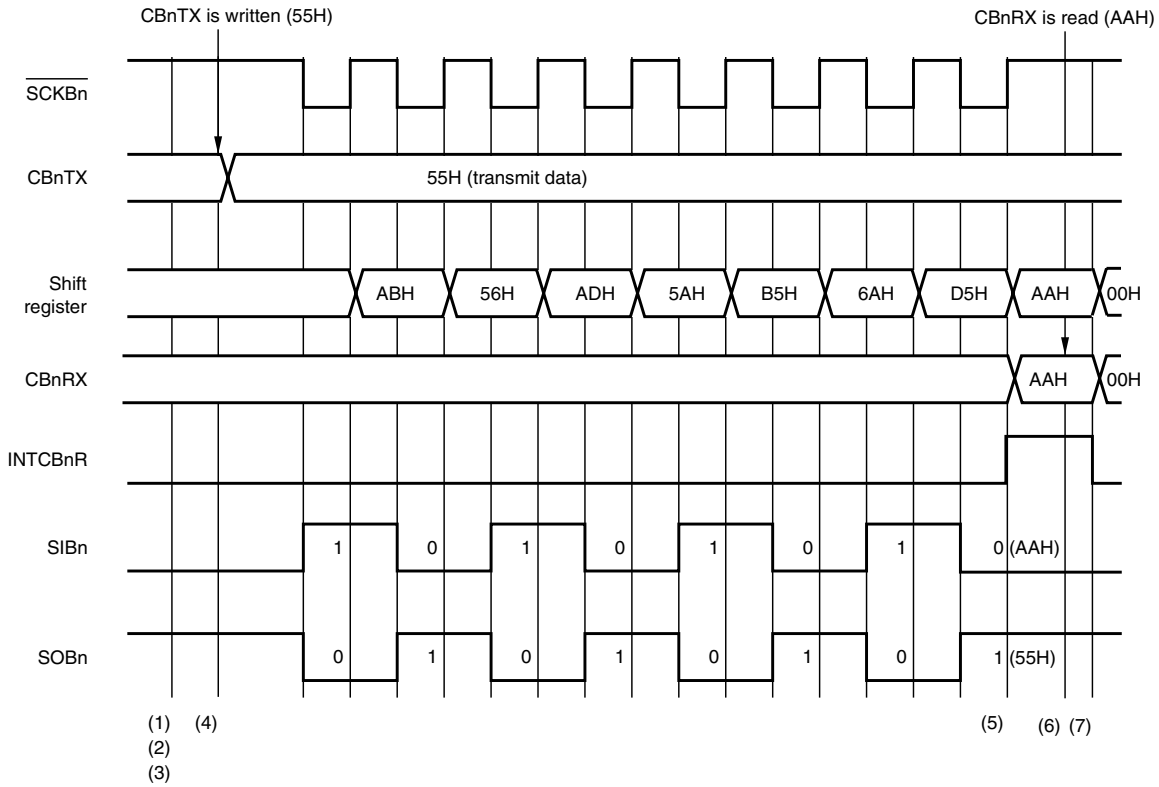
The transmission enable interrupt request signal is generated when transmit data is transferred from the CBnTX register while transmission enabled.

11.5 Operation

11.5.1 Single transfer (master mode, transmission/reception mode)

Figure 11-9 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL register = 0, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-9: Single Transfer Timing (Master Mode, Transmission/Reception Mode)



<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 register.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable transmission/reception by setting the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Write transfer data to the CBnTX register (start transmission).
<5> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX (CBnRXL) register can be read.
<6> Read the CBnRX register before clearing the CBnPWR bit to 0.
<7> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0 (end of transmission/reception).

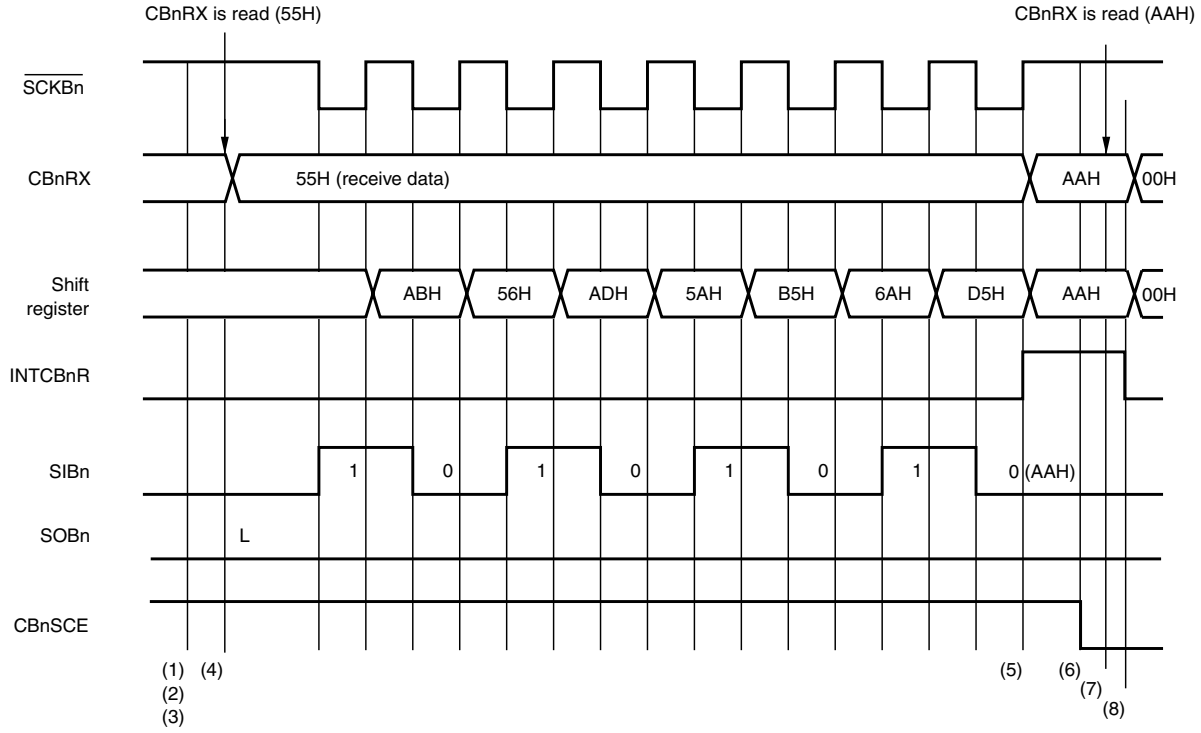
To transfer more data, repeat (4) to (6) before (7).

Remark: The processing in (2) and (3) can be set simultaneously.

11.5.2 Single transfer mode (master mode, reception mode)

Figure 11-10 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 0, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-10: Single Transfer Timing (Master Mode, Reception Mode)



<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable reception by setting the CBnRXE bit of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Read dummy data from the CBnRX register (reception start trigger).
<5> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX (CBnRXL) register can be read.
<6> Get the last receive data ready by clearing the CBnSCE bit of the CBnCTL0 register to 0.
<7> Read the CBnRX register before clearing the CBnPWR bit to 0.
<8> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0 (end of reception).

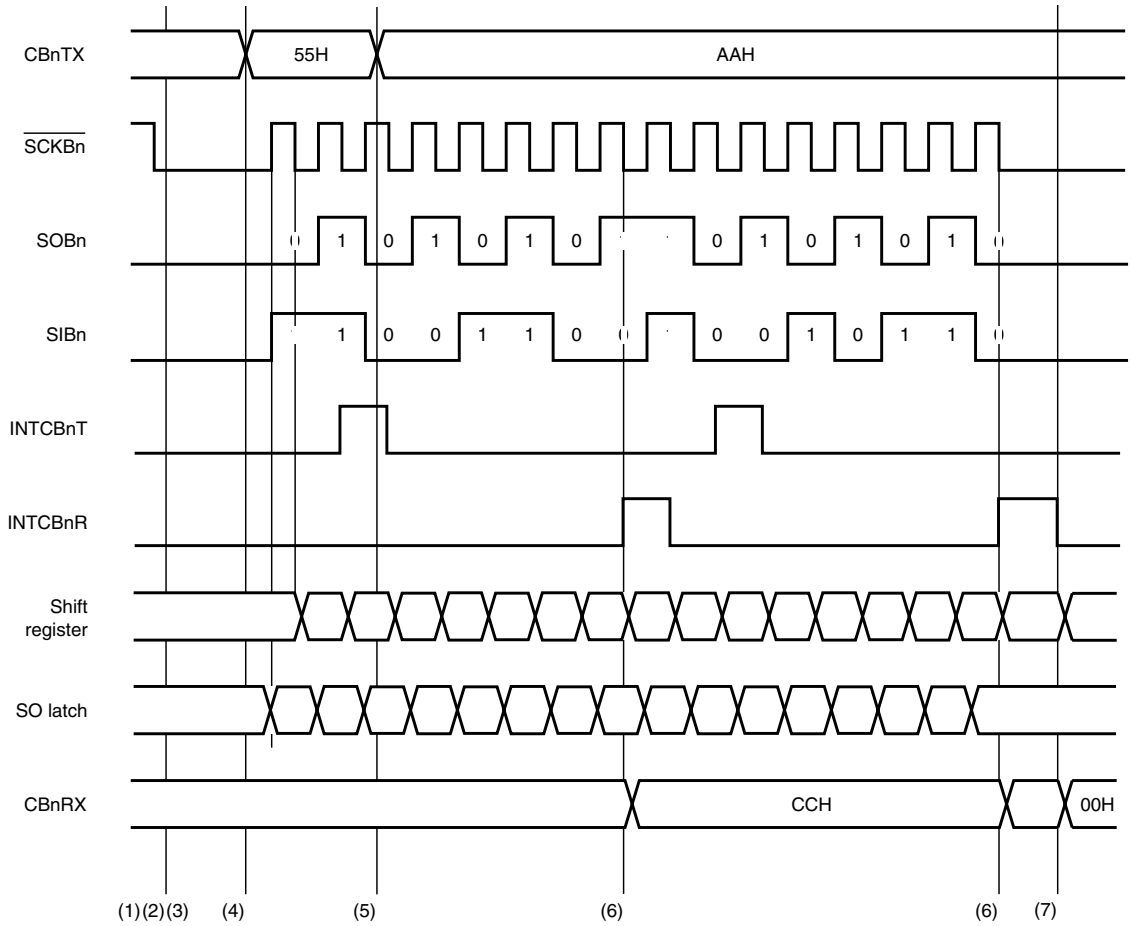
To transfer more data, repeat (4) and (5) before (6) (at this time, read receive data that is also used as a reception trigger, instead of dummy data).

Remark: The processing in (2) and (3) can be set simultaneously.

11.5.3 Continuous mode (master mode, transmission/reception mode)

Figure 11-11 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 0, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-11: Continuous Transfer Timing (Master Mode, Transmission/Reception Mode)



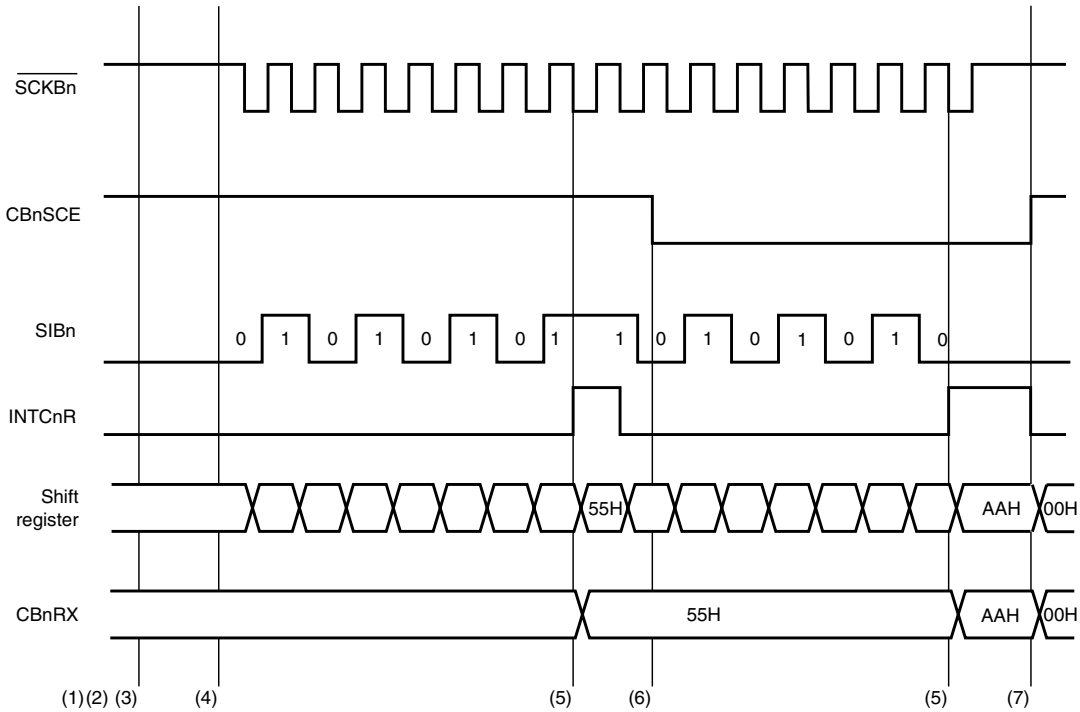
<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable transmission/reception by setting the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Write transfer data to the CBnTX register (start transmission).
<5> Write the transfer data to the CBnTX register when the transmission enable interrupt request signal (INTCBnT) is generated.
<6> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX (CBnRXL) register can be read. Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.
<7> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0.

To transfer more data, repeat (4) to (6) before (7).

11.5.4 Continuous mode (master mode, reception mode)

Figure 11-12 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 1, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-12: Continuous Transfer Timing (Master Mode, Reception Mode)



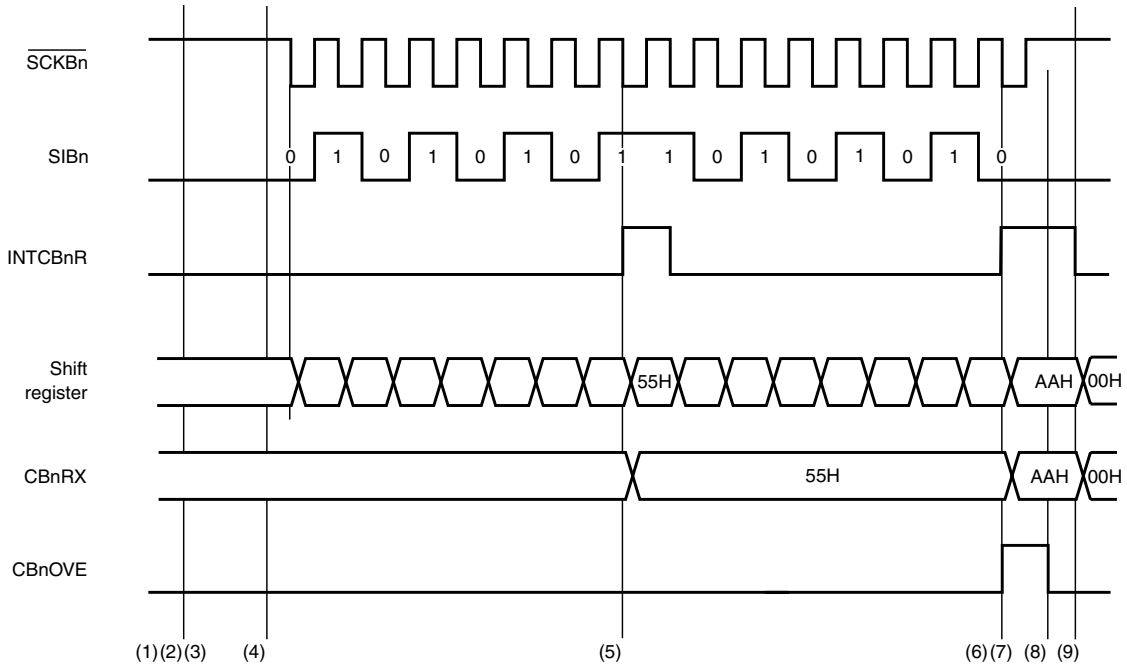
<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable reception by setting the CBnRXE bit of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Read dummy data from the CBnRX register (reception start trigger).
<5> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX (CBnRXL) register can be read. Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.
<6> Prepare the last receive data by clearing the CBnSCE bit of the CBnCTL0 register to 0.
<7> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0 (end of reception).

To transfer more data, repeat (4) and (5) before (6).

11.5.5 Continuous reception mode (error)

Figure 11-13 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 1, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-13: Continuous Transfer Timing (Error)

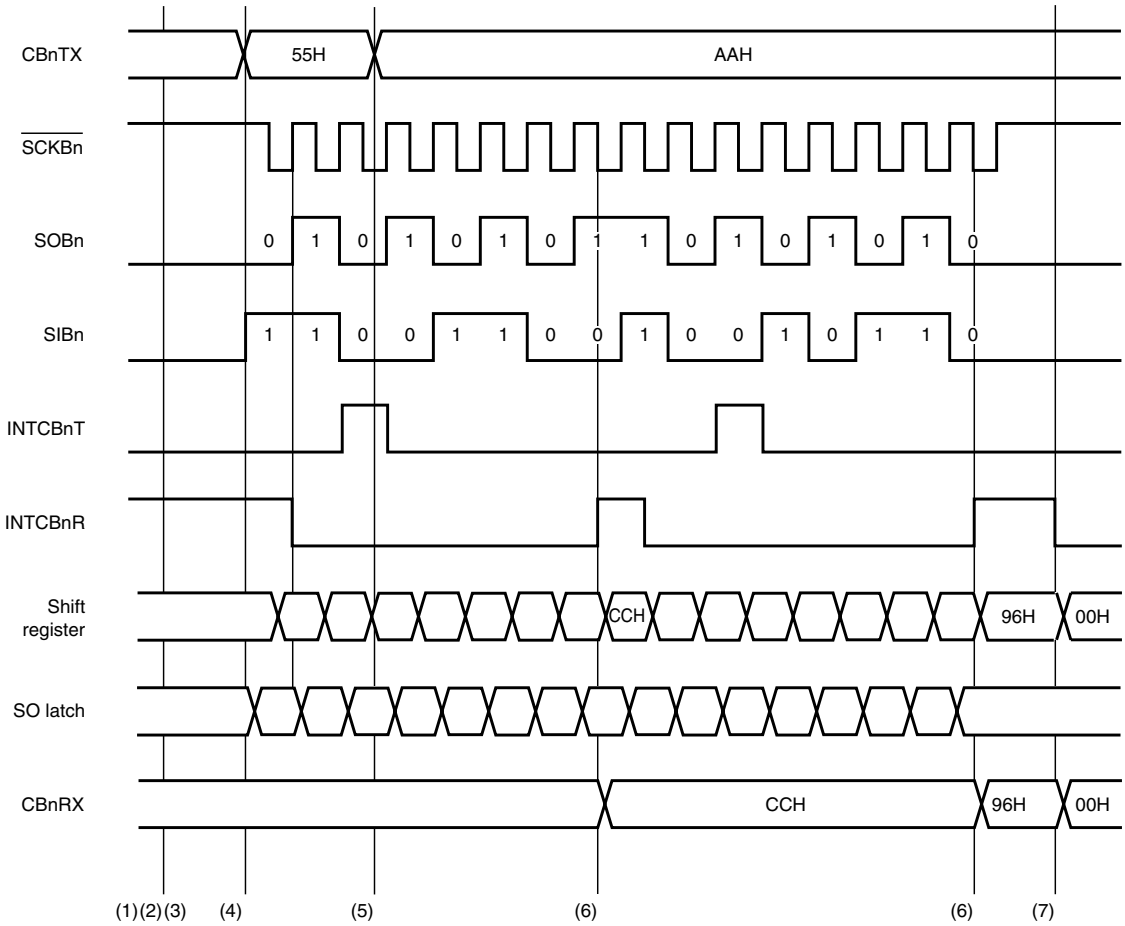


<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable reception by setting the CBnRXE bit of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Read dummy data from the CBnRX register (reception start trigger).
<5> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX (CBnRXL) register can be read.
<6> If the data cannot be read before the next transfer is completed, the CBnOVE flag of the CBnSTR register is set on completion of reception, and the reception complete interrupt request signal (INTCBnR) is generated. Read the CBnRX register before the next receive data arrives.
<7> Confirm that the CBnOVE bit = 1, in the INTCBnR interrupt processing, and perform overrun error processing.
<8> Clear the CBnOVE bit to 0.
<9> Confirm that the CBnTSF bit of the CBnSTR register = 0. Clear the CBnPWR bit to 0 and stop clock supply to CSIB.

11.5.6 Continuous mode (slave mode, transmission/reception mode)

Figure 11-14 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 1, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-14: Continuous Transfer Timing (Slave Mode, Transmission/Reception Mode)



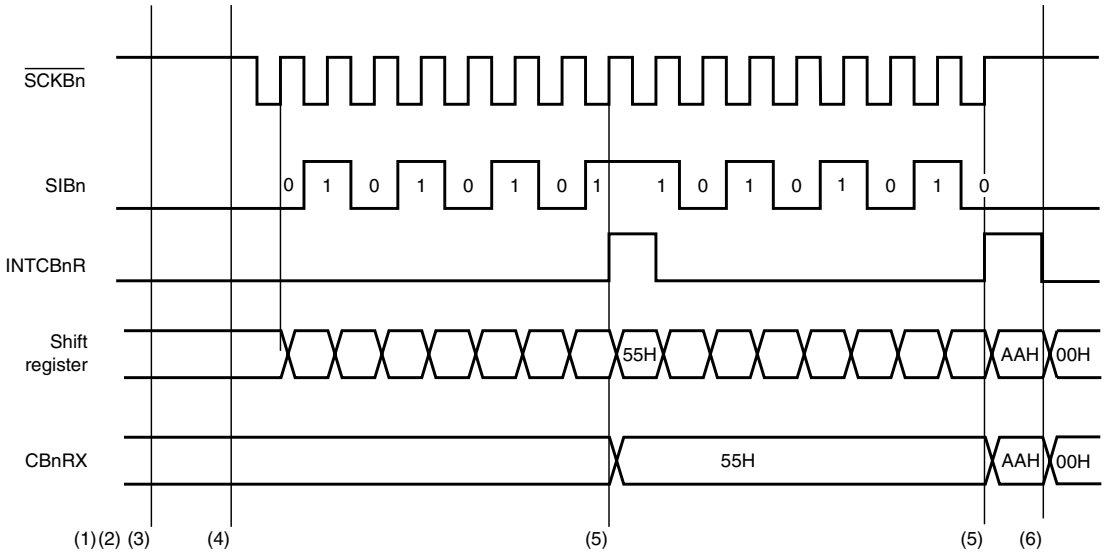
<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable transmission/reception by setting the CBnTXE and CBnRXE bits of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Write transfer data to the CBnTX.
<5> Write the transfer data to the CBnTX register when the transmission enable interrupt request signal (INTCBnT) is generated.
<6> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX register can be read. Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.
<7> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0 (end of transmission/reception).

To transfer more data, repeat (4) to (6) before (7).

11.5.7 Continuous mode (slave mode, reception mode)

Figure 11-15 shows the transfer timing when data is transferred with the MSB first (CBnDIR bit of CBnCTL0 register = 0), when the CBnCKP bit of the CBnCTL1 register = 0, when the CBnDAP bit of the CBnCTL1 register = 0, and when the transfer data length is 8 bits (CBnCL3 to CBnCL0 bits of the CBnCTL2 register = 0, 0, 0, 0).

Figure 11-15: Continuous Transfer Timing (Slave Mode, Reception Mode)



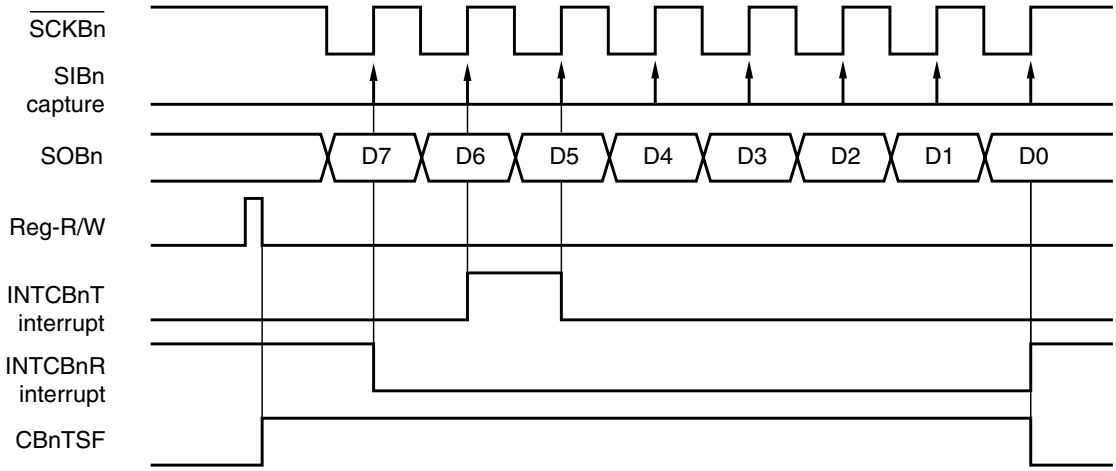
<1> Specify the transfer mode by setting the CBnCTL1 and CBnCTL2 registers.
<2> Specify the transfer mode by using the CBnDIR bit of the CBnCTL0 register and, at the same time, enable reception by setting the CBnRXE bit of the CBnCTL0 register to 1.
<3> Enable CSIB operating clock supply by setting the CBnPWR bit of the CBnCTL0 register to 1.
<4> Read dummy data from the CBnRX register (reception start trigger).
<5> The reception complete interrupt request signal (INTCBnR) is generated to inform the CPU that the CBnRX register can be read. Read the CBnRX register before the next receive data arrives or before the CBnPWR bit is cleared to 0.
<6> Confirm that the CBnTSF bit of the CBnSTR register = 0, and stop clock supply to CSIB by clearing the CBnPWR bit to 0 (end of reception).

To transfer more data, repeat (4) and (5) before (6).

11.5.8 Clock timing

Figure 11-16: Clock Timing (1/2)

(a) CBnCKP = 0, CBnDAP = 0



(b) CBnCKP = 1, CBnDAP = 0

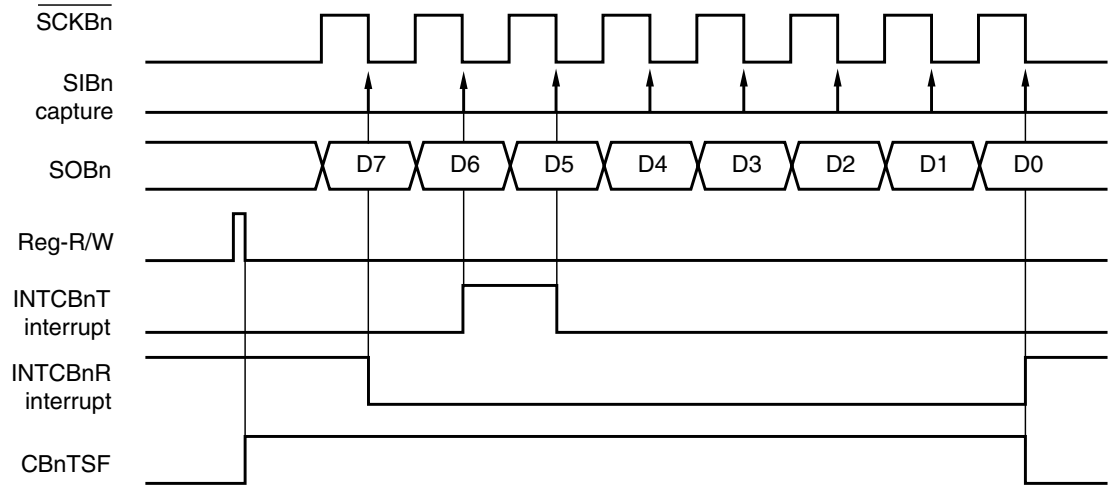
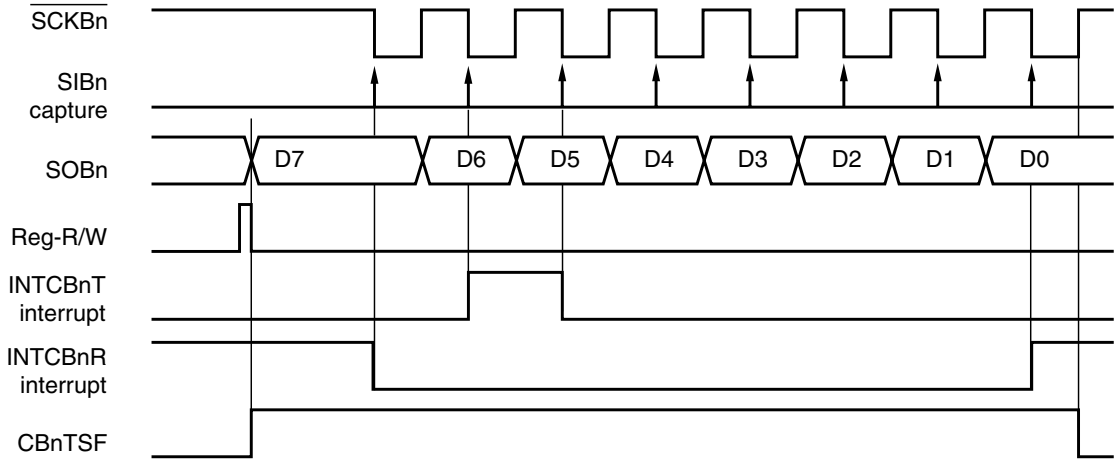
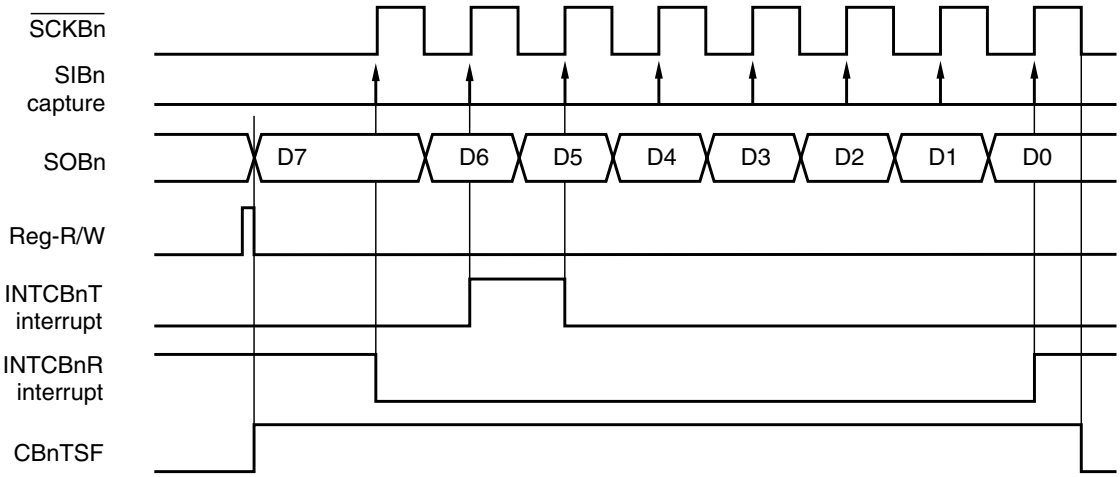


Figure 11-16: Clock Timing (2/2)

(c) CBnCKP = 0, CBnDAP = 1



(d) CBnCKP = 1, CBnDAP = 1



11.5.9 Output pin status with operation disabled

(1) $\overline{\text{SCKBn}}$ pin

The output status of the SCKBn pin is as follows when CSIBn operation is disabled (when the CBnPWR bit of the CBnCTL0 register = 0).

CBnCKP	$\overline{\text{SCKBn}}$ Pin Output
0	Fixed to high level
1	Fixed to low level

Remarks: 1. The output of the SCKBn pin changes if the CBnCKP bit of the CBnCTL1 register is rewritten.

2. n = 0 to 2

(2) SOBn pin

The output status of the SOBn pin is as follows when CSIBn operation is disabled (CBnPWR bit = 0).

CBnTXE	CBnDAP	CBnDIR	SOBn Pin Output
0	×	×	Fixed to low level
1	0	×	Value of SOBn latch (low level)
1	1	0	Value of CBnTXn (MSB)
1	1	1	Value of CBnTXn (LSB)

Remarks: 1. The output of the SOBn pin changes if any of the CBnTXE, CBnDAP, and CBnDIR bits of the CBnCTL1 register is rewritten.

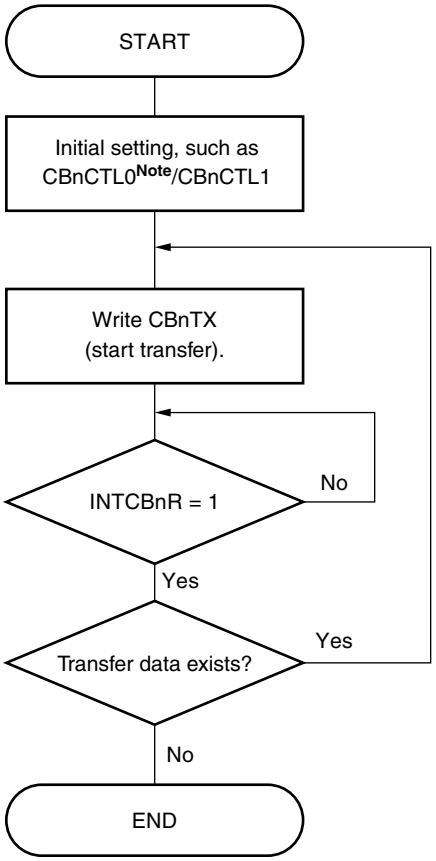
2. n = 0 to 2

3. ×: don't care

11.6 Operation Flow

(1) Single transmission

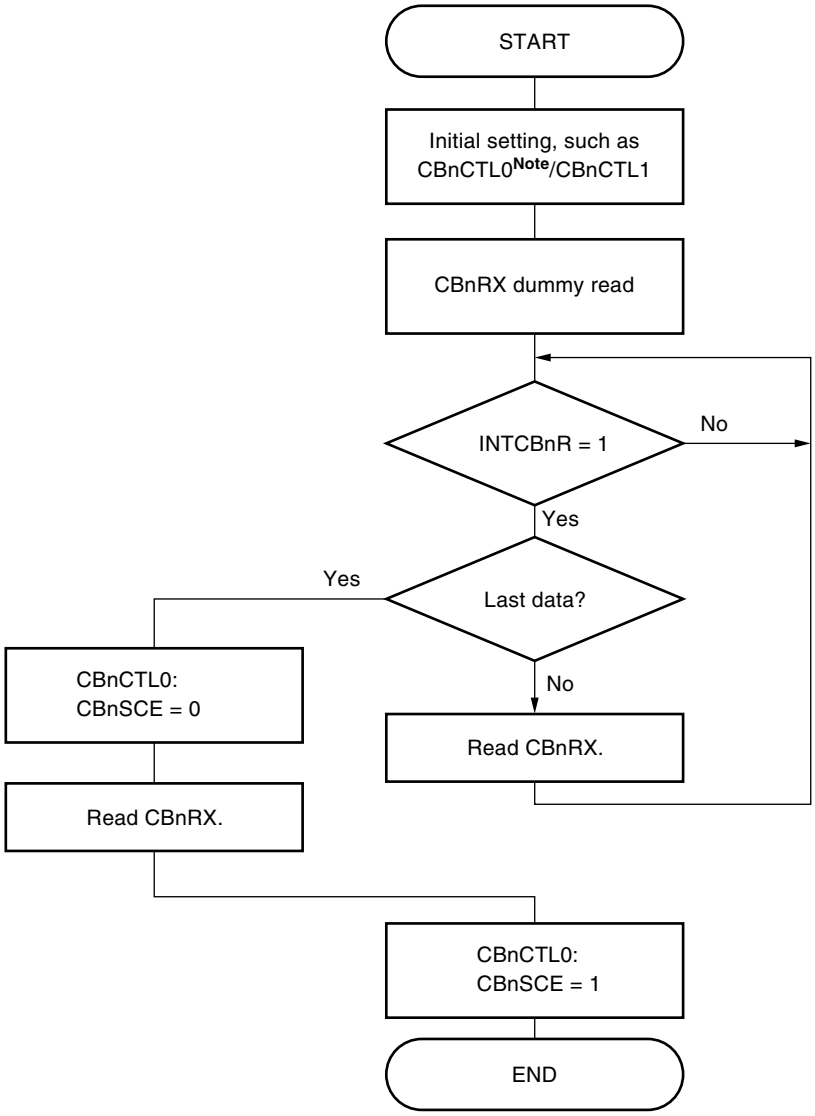
Figure 11-17: Single Transmission Flow



Note: Set the CBnSCE bit to 1 as the initial setting

(2) Single reception (Master)

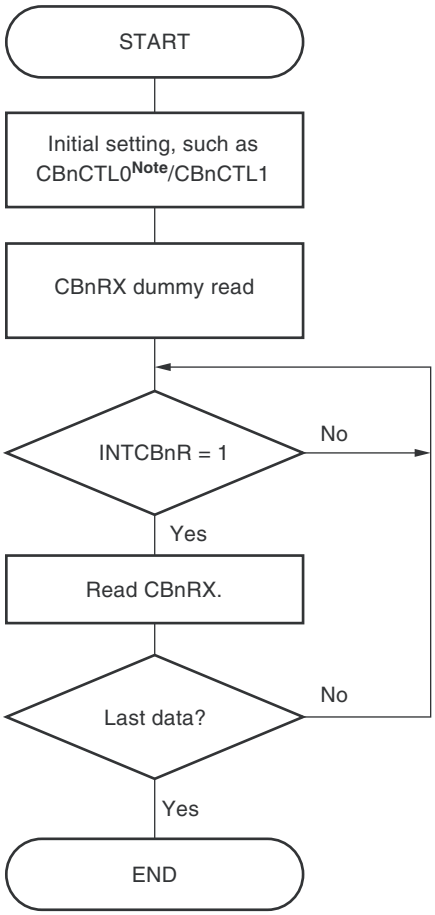
Figure 11-18: Single Reception Flow (Master)



Note: Set the CBnSCE bit to 1 as the initial setting.

(3) Single reception (Slave)

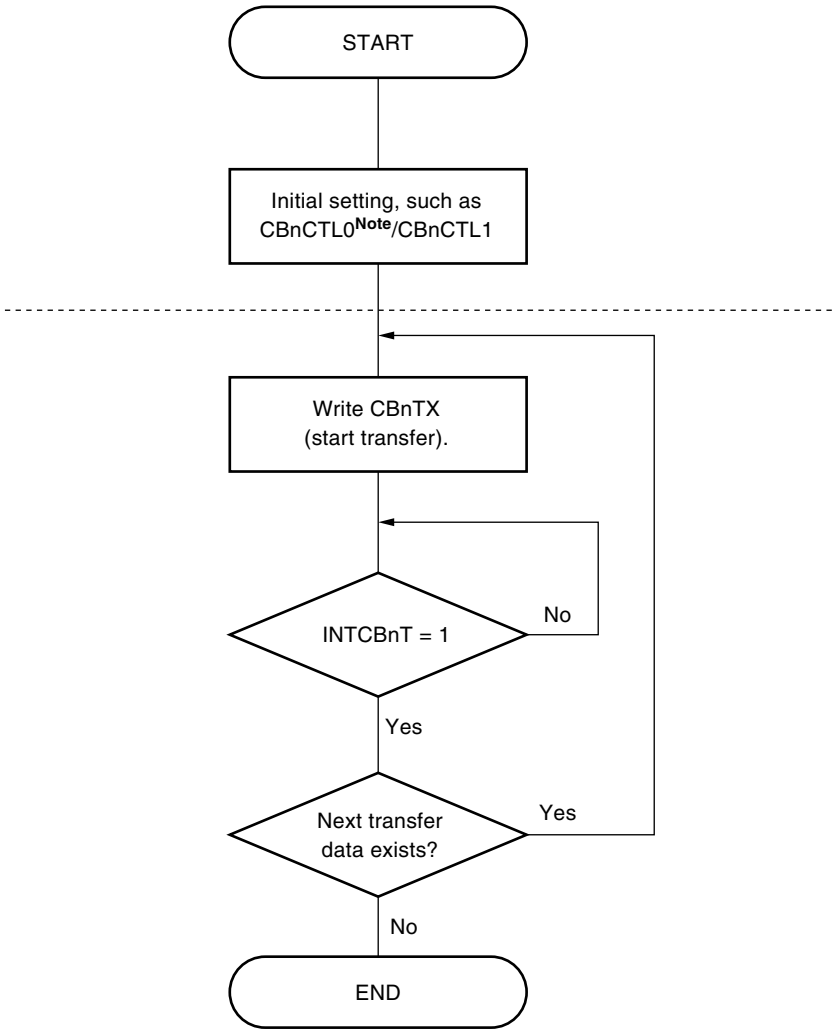
Figure 11-19: Single Reception Flow (Slave)



Note: Set the CBnSCE bit to 1 as the initial setting.

(4) Continuous transmission

Figure 11-20: Continuous Transmission Flow

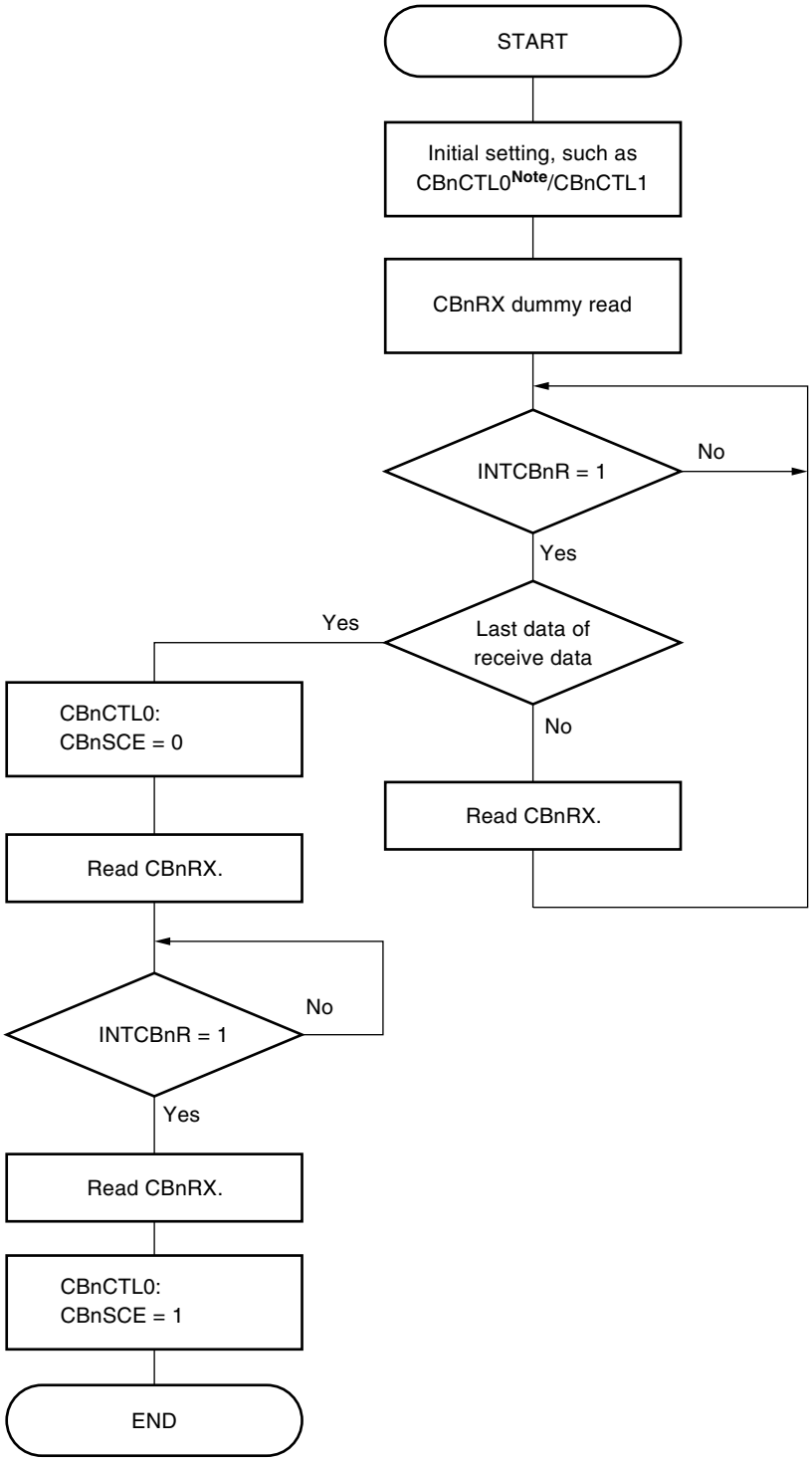


Note: Set the CBnSCE bit to 1 as the initial setting.

Remark: The flow shown below the broken lines is the flow of transmission. Execute this flow to start transmission a second and subsequent time.

(5) Continuous reception (Master)

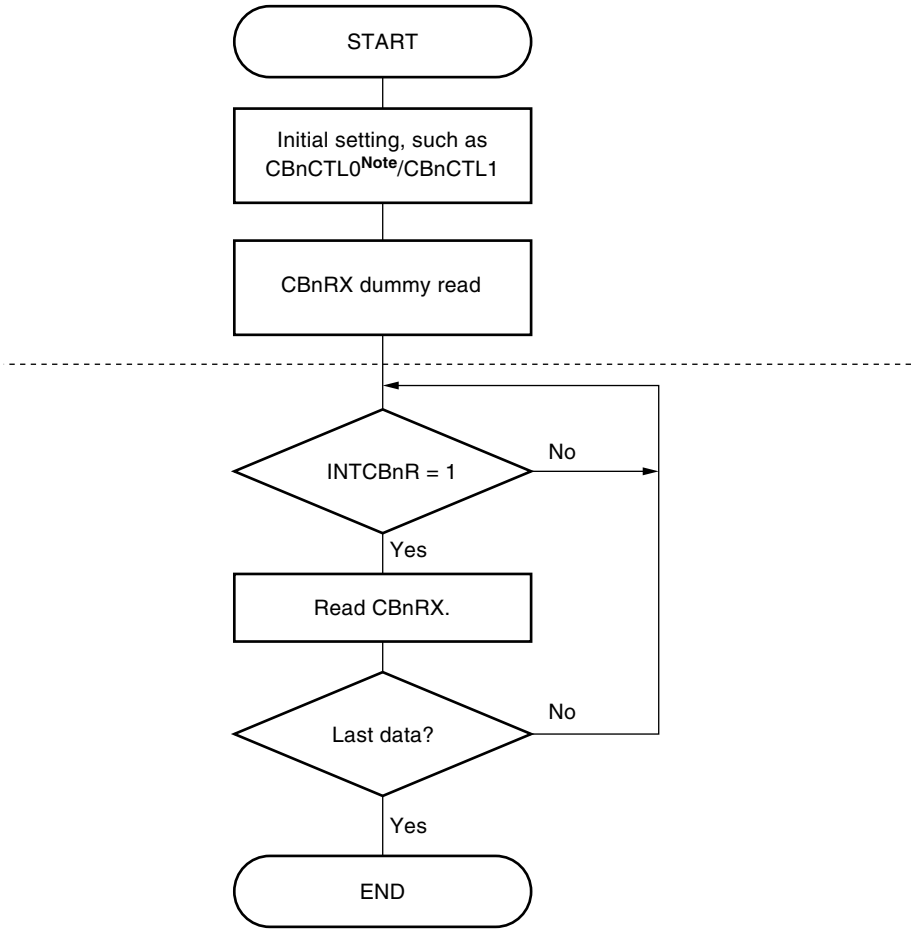
Figure 11-21: Continuous Reception Flow (Master)



Note: Set the CBnSCE bit to 1 as the initial setting.

(6) Continuous reception (Slave)

Figure 11-22: Continuous Reception Flow (Slave)



Note: Set the CBnSCE bit to 1 as the initial setting.

Remark: The flow shown below the broken lines is the flow of transmission. Execute this flow to start transmission a second and subsequent time.

11.7 Prescaler 3

Prescaler 3 has the following function:

- Generation of clock CSIB

11.7.1 Control registers of prescaler 3

(1) Prescaler mode register 0 (PRSM0)

The PRSM0 register is used to control generation of the count clock for the watch timer and CSIB0.
This register can be read or written in 8-bit or 1-bit units.
Reset input clears this register to 00H.

Cautions: 1. Do not change the values of the BGCS01 and BGCS00 bits while the watch timer is operating.

2. Set the PRSM0 register before setting the BGCE0 bit to 1.

Figure 11-23: Prescaler mode register 0 (PRSM0) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
PRSM0	0	0	0	BGCE0	0	0	BGCS01	BGCS00	FFFFF8B0H	R/W	00H

BGCE0	Prescaler output
0	Disabled (fixed to 0
1	Enabled

BGCS01	BGCS00	Selection of count clock (f_{BRG})		
			24 MHz	32 MHz
0	0	f_X	41.67 ns	31.25 ns
0	1	$f_X/2$	83.33 ns	62.5 ns
1	0	$f_X/4$	166.66 ns	125 ns
1	1	$f_X/8$	333.33 ns	250 ns

(2) Prescaler compare register 0 (PRSCM0)

This is an 8-bit compare register.
It can be read or written in 8-bit units.
Reset input clears this register to 00H.

Cautions: 1. Do not rewrite the PRSCM0 register while the watch timer is operating.

2. Set the PRSCM0 register before setting the BGCE0 bit of the PRSM0 register to 1.

Figure 11-24: Prescaler compare register 0 (PRSCM0) Format

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
PRSCM0	PRSCM07	PRSCM06	PRSCM05	PRSCM04	PRSCM03	PRSCM02	PRSCM01	PRSCM00	FFFFF8B1H	R/W	00H

[MEMO]

Chapter 12 I²C Bus

Caution: To use the I²C bus function, set the P36/SCL0 and P35/SDA0 to N-ch open-drain output (alternate function).

12.1 Mode Switching of I²C Bus and Other Serial Interfaces

12.1.1 CSIB2 and I²C0 mode switching

In the V850E/CG3, CSIB2 and I²C0 are alternate functions of the same pin and therefore cannot be used simultaneously. CSIB2 and I²C0 switching must be set in advance using the PMC3 and PFC3 registers.

Caution: The transmit/receive operation of CSIB2 and I²C0 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 12-1: CSIB2 and I²C00 Mode Switch Settings

	7	6	5	4	3	2	1	0	Address	R/W	After reset
PMC3	0	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	0xFFFFF446	RW	0x00

	7	6	5	4	3	2	1	0	Address	R/W	After reset
PFC3	0	PFC36	PFC35	0	0	0	0	0	0xFFFFF466	R/W	0x00

PMC36	PFC36	PM36	Operation mode
0	×	×	Port I/O mode
1	0	0	CSIB2 mode (SCKB0 output)
1	0	1	CSIB2 mode (SCKB0 input)
1	1	0	setting prohibited
1	1	1	I²C0 mode (SCL0 input)

PMC35	PFC35	PM35	Operation mode
0	×	×	Port I/O mode
1	0	0	CSIB2 mode (SOB0)
1	0	1	Port I/O mode
1	1	0	setting prohibited
1	1	1	I²C0 mode (SDA0)

Remark: × = don't care

12.2 Features

I²C0 have the following two modes.

- Operation stopped mode
- I²C (Inter IC) bus mode (multi-masters supported)

(1) Operation stopped mode

In this mode, serial transfers are not performed, thus enabling a reduction in power consumption.

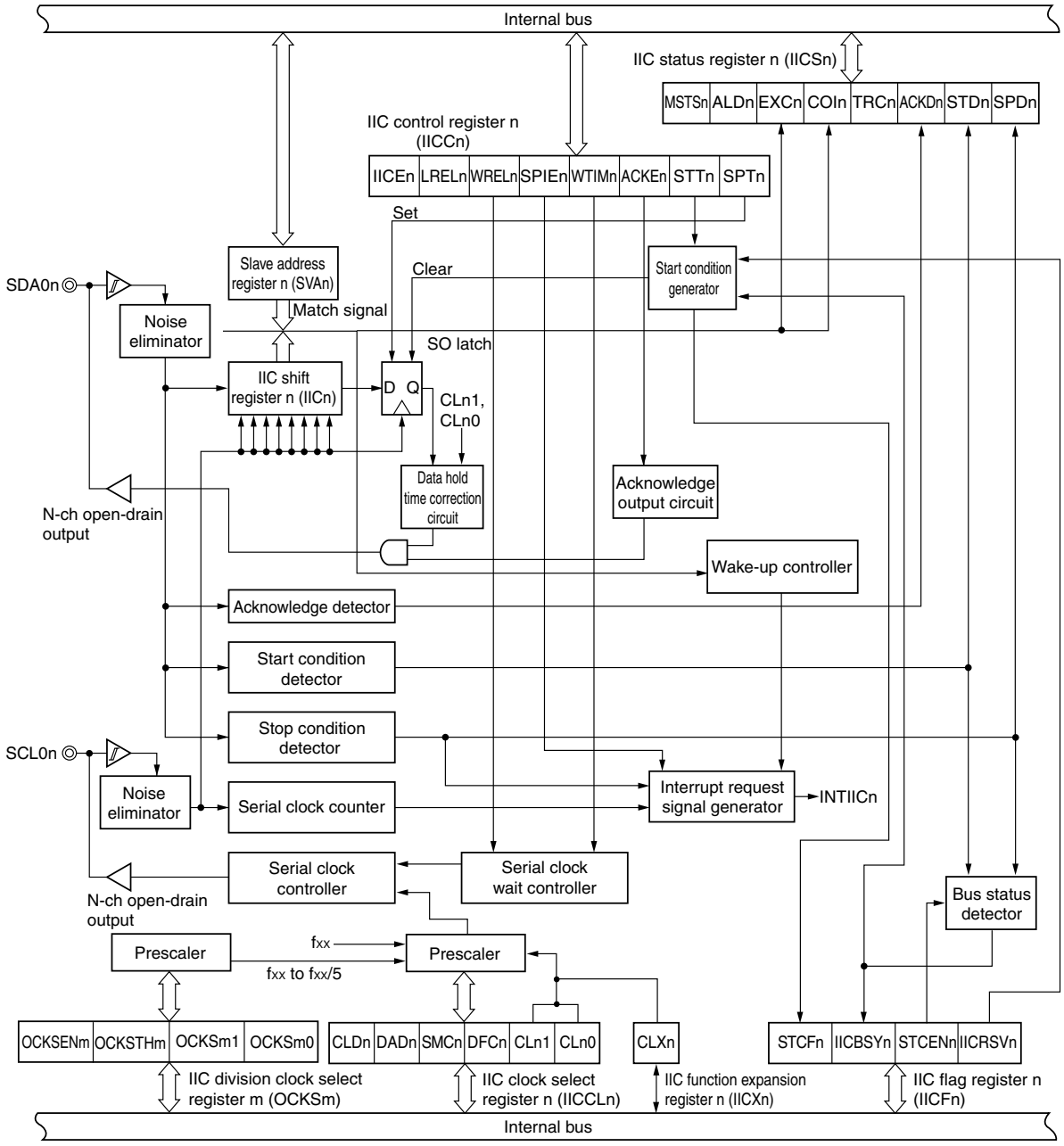
(2) I²C bus mode (multi-master support)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock pin (SCL0) and a serial data bus pin (SDA0).
This mode complies with the I²C bus format and the master device can output “start condition”, “data”, and “stop condition” data to the slave device via the serial data bus. The slave device automatically detects the received data by hardware. This function can simplify the part of an application program that controls the I²C bus.
Since SCL0 and SDA0 pins are used for N-ch open-drain outputs, I²C0 requires pull-up resistors for the serial clock line and the serial data bus line.

12.3 Configuration

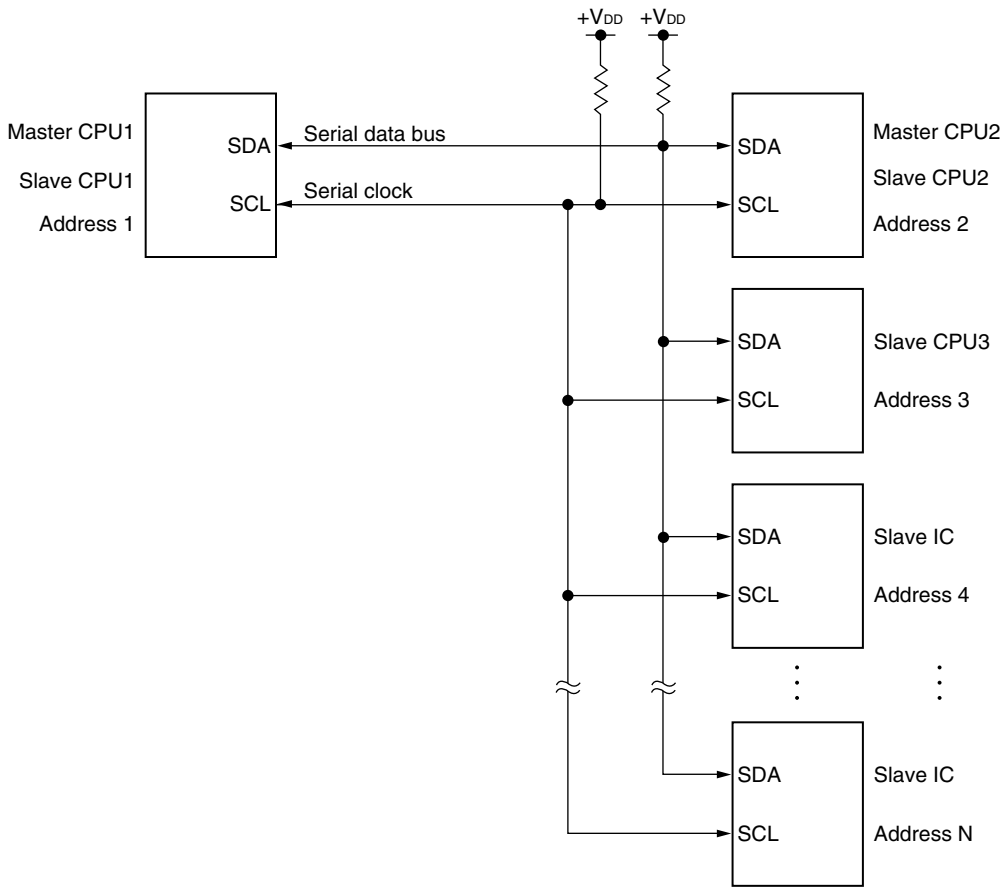
The block diagram of the I²C0 is shown below.

Figure 12-2: Block Diagram of I²C0



A serial bus configuration example is shown below.

Figure 12-3: Serial Bus Configuration Example Using I²C Bus



I²C0 includes the following hardware.

Table 12-1: Configuration of I²C0

Item	Configuration
Registers	IIC shift register (IIC) Slave address register (SVA0)
Control registers	IIC control register (IICC0) IIC status register (IICS0) IIC flag register (IICF00) IIC clock select register (IICCL0) IIC function expansion register (IICX0) IIC division clock select registers 0, 1 (OCKS0, OCKS1)

(1) IIC shift register (IIC0)

The IIC register converts 8-bit serial data into 8-bit parallel data and vice versa, and can be used for both transmission and reception.
Write and read operations to the IIC0 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 (SVA0)

The SVA0 register sets local addresses when in slave mode.
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 SDA0 pin.

(4) Wake-up controller

This circuit generates an interrupt request when the address received by this register matches the address value set to the SVA0 register or when an extension code is received.

(5) Prescaler

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 (INTIIC0).
An I²C interrupt is generated following either of two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by IICCn0.WTIM0 bit)
- Interrupt occurrence due to stop condition detection (set by IICC0.SPIE0 bit)

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0 pin from the sampling clock.

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) ACK output circuit, stop condition detector, start condition detector, and ACK detector

These circuits are used to output and detect various control signals.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the SCL0 pin.

(12) Start condition generator

A start condition is issued when the IICC0.STT0 bit is set.
However, in the communication reservation disabled status (IICF0.IICRSV0 bit = 1), this request is ignored and the IICFn.STCFn bit is set if the bus is not released (IICF0.IICBSY0 bit = 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 bus status detector to the initial status by using the IICF0.STCEN bit.

12.4 Control Registers

I²C0 are controlled by the following registers.

- IIC control registers 0 (IICC0)
- IIC status registers 0 (IICS0)
- IIC flag registers 0 (IICF0)
- IIC clock select registers 0 (IICCL0)
- IIC function expansion registers 0 (IICX0)
- IIC division clock select registers 0 (OCKS0)

The following registers are also used.

- IIC shift registers 0 (IIC0)
- Slave address registers 0 (SVA0)

(1) IIC control registers 0 (IICC0)

The IICC0 registers enable/stop I²C0n operations, set the wait timing, and set other I²C operations. These registers can be read or written in 8-bit or 1-bit units.
Reset input clears these registers to 00H.

Figure 12-4: IIC control registers 0 (IICC0) Format (1/4)

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	R/W	After reset
IICC0	IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FFFFFD82H,	R/W	00H

IICE0	Specification of I²C0 operation enable/disable
0	Operation stopped. IICSn register reset[Note 1]. Internal operation stopped.
1	Operation enabled.
Condition for clearing (IICE0 bit = 0)	Condition for setting (IICE0 bit = 1)
• Cleared by instruction • After reset	• Set by instruction

LREL0	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 SCL0 and SDA0 lines are set to high impedance. The STT0 and SPT0 bits and the MSTS0, EXC0, COI0, TRC0, ACKD0, and STD0 bits of the IICS0 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 occurs or an extension code is received after the start condition.	
Condition for clearing (LREL0 bit = 0)[Note 2]	Condition for setting (LREL0 bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

WREL0	Wait cancellation control
0	Wait not cancelled
1	Wait cancelled. This setting is automatically cleared after wait is cancelled.
Condition for clearing (WREL0 bit = 0)[Note 2]	Condition for setting (WREL0 bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

Notes: 1. The IICS register, IICF0.STCF0 and IICF0.IICBSY0 bits, and IICCL0.CLD0 and IICCL0.DAD0 bits are reset.

2. This flag's signal is invalid when the IICE0 bit = 0.

Figure 12-4: IIC control registers 0 (IICC0 to IICC2) Format (2/4)

SPIE0	Enable/disable generation of interrupt request when stop condition is detected
0	Disabled
1	Enabled

Condition for clearing (SPIE0 bit = 0)[Note]	Condition for setting (SPIE0 bit = 1)
• Cleared by instruction • After reset	• Set by instruction

WTIM0	Control of wait and interrupt request generation
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and wait is set. Slave mode: After input of eight clocks, the clock is set to low level and wait is set for the 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 the master device.
During address transfer, an interrupt occurs at the falling edge of the ninth clock regardless of this bit setting. This bit setting becomes valid when the address transfer is completed. In master mode, a wait is inserted at the falling edge of the ninth clock during address transfer. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an ACK signal is issued. When the slave device has received an extension code, however, a wait is inserted at the falling edge of the eighth clock.	

Condition for clearing (WTIM0 bit = 0)[Note]	Condition for setting (WTIM0 bit = 1)
• Cleared by instruction • After reset	• Set by instruction

ACKE0	Acknowledgement control
0	Acknowledgment disabled.
1	Acknowledgment enabled. During the ninth clock period, the SDA0n line is set to low level. However, ACK is invalid in other than extension mode during address transfers.

Condition for clearing (ACKE0 bit = 0)[Note]	Condition for setting (ACKE0 bit = 1)[Note]
• Cleared by instruction • After reset	• Set by instruction

Note: This flag's signal is invalid when the IICE0 bit = 0.

Figure 12-4: IIC control registers 0 (IICC0) Format (3/4)

STT0	Start condition trigger
0	Start condition is not generated.
1	When bus is released (in STOP mode): A start condition is generated (for starting as master). The SDA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, the SCL0 line is changed to low level. During communication with a third party If the communication reservation function is enabled (IICF0.IICRSV0 bit = 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 (IICRSV0 = 1) • The IICF0.STCF0 bit is set. This trigger does not generate a start condition. In the wait state (when master device) A restart condition is generated after the wait is released.

Cautions concerning set timing:

- For master reception: Cannot be set during transfer. Can be set only when the ACKE0 bit has been set to 0 and the 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 (IICRSV0 bit = 1), the communication reservation status is entered.
- Cannot be set at the same time as the SPT0 bit

Condition for clearing (STT0 bit = 0)[Note]	Condition for setting (STT0 bit = 1)
• Cleared by loss in arbitration • Cleared after start condition is generated by master device • When the LREL0 bit = 1 (communication save) • When the IICE0 bit = 0 (operation stop) • After reset	• Set by instruction

Note: Clearing the IICE0 bit to 0 invalidates the signals of this flag.

Remark: The STT0 bit is 0 if it is read immediately after data setting.

Figure 12-4: IIC control registers 0 (IICC0) Format (4/4)

SPT0	Stop condition trigger
0	Stop condition is not generated.
1	Stop condition is generated (termination of master device's transfer). After the SDA0 line goes to low level, either set the SCL0 line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDA0 line is changed from low level to high level and a stop condition is generated.

Cautions concerning set timing

- For master reception: Cannot be set during transfer.

Can be set only when the ACKE0 bit has been set to 0 and during the wait period after the 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 STT0 bit.
- The SPT0 bit can be set only when in master mode[Note 1].
- When the WTIM0 bit has been set to 0, if the SPT0 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 WTIM0 bit should be changed from 0 to 1 during the wait period following output of eight clocks, and the SPT0 bit should be set during the wait period that follows output of the ninth clock.

Condition for clearing (SPT0 bit = 0)[Note 2]	Condition for setting (SPT0 bit = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When the LREL0 bit = 1 (communication save) • When the IICE0 bit = 0 (operation stop) • After reset	• Set by instruction

Notes: 1. Set the SPT0 bit only in master mode. However, when the IICRSV0 bit is 0, the SPT0 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 section 12.15 "Cautions" on page 481.

2. Clearing the IICE0 bit to 0 invalidates the signals of this flag.

Caution: When the TRC0 bit = 1, the WREL0 bit is set during the ninth clock and wait is canceled, after which the TRC0 bit is cleared and the SDA0 line is set to high impedance.

Remark: The SPT0 bit is 0 if it is read immediately after data setting

(2) IIC status registers 0 (IICS0)

The IICS0 registers indicate the status of the I²C0 bus.
These registers are read-only, in 8-bit or 1-bit units.
Reset input clears these registers to 00H.

Figure 12-5: IIC status registers 0 (IICS0 to IICS2) Format (1/3)

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	R/W	After reset
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	FFFFFD86H,	R	00H

MSTS0	Master device status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTS0 bit = 0)	Condition for setting (MSTS0 bit = 1)
• When a stop condition is detected • When the ALD0 bit = 1 (arbitration loss) • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset	• When a start condition is generated

ALD0	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 MSTS0 bit is cleared.

Condition for clearing (ALD0 bit = 0)	Condition for setting (ALD0 bit = 1)
• Automatically cleared after the IICS0 register is read **Note** • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset	• When the arbitration result is a “loss”.

Note: This register is also cleared when a bit manipulation instruction is executed for bits other than the IICS0 register.

Figure 12-5: IIC status registers 0 (IICS0) Format (2/3)

EXC0	Detection of extension code reception	
0	Extension code was not received.	
1	Extension code was received.	
Condition for clearing (EXC0 bit = 0)		Condition for setting (EXC0 bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset		• When the higher four bits of the received address data are either “0000” or “1111” (set at the rising edge of the eighth clock).

COI0	Matching address detection	
0	Addresses do not match.	
1	Addresses match.	
Condition for clearing (COI0 bit = 0)		Condition for setting (COI0 bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset		• When the received address matches the local address (SVA0 register) (set at the rising edge of the eighth clock).

TRC0	Transmit/receive status detection	
0	Receive status (other than transmit status). The SDA0 line is set to high impedance.	
1	Transmit status. The value in the SO latch is enabled for output to the SDA0 line (valid starting at the falling edge of the first byte’s ninth clock).	
Condition for clearing (TRC0 bit = 0)		Condition for setting (TRC0 bit = 1)
• When a stop condition is detected • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • Cleared by WREL0 bit = 1[Note] • When the ALD0 bit changes from 0 to 1 (arbitration loss) • 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)

Note: The TRC0 bit is cleared and SDA0 line becomes high impedance when the WREL0 bit is set and the wait state is canceled at the ninth clock by TRC0 bit = 1.

Figure 12-5: IIC status registers 0 (IICS0) Format (3/3)

ACKD0	ACK detection
0	ACK was not detected.
1	ACK was detected.

Condition for clearing (ACKD0 bit = 0)	Condition for setting (ACKD0 bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset	• After the SDA0 bit is set to low level at the rising edge of the SCL0 pin's ninth clock

STD0	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 (STD0 bit = 0)	Condition for setting (STD0 bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LREL0 bit = 1 (communication save) • When the IICE0 bit changes from 1 to 0 (operation stop) • After reset	When a start condition is detected

SPD0	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 (SPD0 bit = 0)	Condition for setting (SPD0 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 IICE0 bit changes from 1 to 0 (operation stop) • After reset	When a stop condition is detected

(3) IIC flag registers 0 (IICF0)

The IICF0 registers set the I²C0 operation mode and indicate the I²C bus status.
These registers can be read or written in 8-bit or 1-bit units. However, the STCF0 and IICBSY0 bits are read-only.
IICRSV0 enables/disables the communication reservation function (see section 12.14 "Communication Reservation" on page 476).
The initial value of the IICBSY0 bit is set by using the STCEN0 bit (see section 12.15 "Cautions" on page 481).
The IICRSV0 and STCEN0 bits can be written only when operation of I²C0 is disabled (IICC0.IICE0 bit = 0). After operation is enabled, IICF0 can be read.
Reset input clears these registers to 00H.

Figure 12-6: IIC flag registers 0 (IICF0) Format (1/2)

	<7>	<6>	5	4	3	2	<1>	<0>	Address	R/W	After reset
IICF0	STCF0	IICBSY0	0	0	0	0	STCEN0	IICRSV0	FFFFFD8AH,	R/W **Note**	00H

Note: Bits 6 and 7 are read-only bits.

STCF0	STT0 bit clear
0	Start condition issued
1	Start condition cannot be issued, STT0 bit cleared

Condition for clearing (STCF0 bit = 0)	Condition for setting (STCF0 bit = 1)
• Cleared by IICC0.STT0 bit = 1 • After reset	• When start condition is not issued and STT0 flag is cleared during communication reservation is disabled (IICRSV0 bit = 1).

IICBSY0	I²C0 bus status
0	Bus released status
1	Bus communication status

Condition for clearing (IICBSY0 bit = 0)	Condition for setting (IICBSY0 bit = 1)
• When stop condition is detected • After reset	• When start condition is detected • By setting the IICC0.IICE0 bit when the STCEN0 bit = 0

STCEN0	Initial start enable trigger
0	Start conditions cannot be generated until a stop condition is detected following operation enable (IICE0 bit = 1).
1	Start conditions can be generated even if a stop condition is not detected following operation enable (IICE0 bit = 1).

Condition for clearing (STCEN0 bit = 0)	Condition for setting (STCEN0 bit = 1)
• When start condition is detected • After reset	• Setting by instruction

Figure 12-6: IIC flag registers 0 (IICF0) Format (2/2)

IICRSV0	Communication reservation function disable bit
0	Communication reservation enabled
1	Communication reservation disabled

Condition for clearing (IICRSV0 bit = 0)	Condition for setting (IICRSV0 bit = 1)
• Clearing by instruction • After reset	• Setting by instruction

Cautions: 1. Write the STCEN0 bit only when operation is stopped (IICE0 bit = 0).

2. When the STCEN0 bit = 1, the bus released status (IICBSY0 bit = 0) is recognized regardless of the actual bus status immediately after the I²C0 bus operation is enabled. Therefore, to issue the first start condition (STT0 bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

3. Write the IICRSVn bit only when operation is stopped (IICE0 bit = 0).

(4) IIC clock select registers 0 (IICCL0)

The IICCL0 registers set the transfer clock for the I²C0 bus.
These registers can be read or written in 8-bit or 1-bit units. However, the CLD0 and DAD0 bits are read-only. The SMC0, CL01, and CL00 bits are set by the combination of the IICX0.CLX0 bit and the OCKSTH, OCKS1, and OCKS0 bits of the OCKSm register (see section 12.4 (6)"I²C0 transfer clock setting method" on page 441). Reset input clears these registers to 00H.

Figure 12-7: IIC clock select registers 0 (IICCL0) Format (1/2)

	7	6	<5>	<4>	3	2	1	0	Address	R/W	After reset
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00	FFFFFD84H	R/W **Note**	00H

CLD0	Detection of SCL0 pin level (valid only when IICC0.IICE0 bit = 1)
0	The SCL0 pin was detected at low level.
1	The SCL0 pin was detected at high level.
Condition for clearing (CLD0 bit = 0)	Condition for setting (CLD0 bit = 1)
• When the SCL0 pin is at low level • When the IICE0 bit = 0 (operation stop) • After reset	• When the SCL0 pin is at high level

DAD0	Detection of SDA0 pin level (valid only when IICE0 bit = 1)
0	The SDA0 pin was detected at low level.
1	The SDA0 pin was detected at high level.
Condition for clearing (DAD0 bit = 0)	Condition for setting (DAD0 bit = 1)
• When the SDA0 pin is at low level • When the IICE0 bit = 0 (operation stop) • After reset	• When the SDA0 pin is at high level

SMC0	Operation mode switching
0	Operation in standard mode.
1	Operation in high-speed mode.

DFC0	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
The digital filter can be used only in high-speed mode. In high-speed mode, the transfer clock does not vary regardless of the DFC0 bit setting (on/off). The digital filter is used to eliminate noise in high-speed mode.	

Figure 12-7: IIC clock select registers 0 (IICCL0) Format (2/2)

CL01	CL00	Operation mode switching	
		Standard mode	High-speed mode
0	0	2.00 MHz < Fxx < 4.19 MHz	4.00 MHz < Fxx < 8.38 MHz
0	1	4.19 MHz < Fxx < 8.38 MHz	4.00 MHz < Fxx < 8.38 MHz
1	0	4.19 MHz < Fxx < 8.38 MHz	4.00 MHz < Fxx < 8.38 MHz
1	1	setting prohibited	

Note: Bits 4 and 5 of IICCL0 are read-only bits.

Caution: Be sure to clear bits 7 and 6 of IICCL0.

(5) IIC function expansion registers 0 (IICX0)

The IICX0 to IICS2 registers set I²C0 function expansion (valid only in the high-speed mode). These registers can be read or written in 8-bit or 1-bit units.
Setting of the CLX0 bit is performed in combination with the SMC0, CL01, and CL00 bits of the IICCL0 register and the OCKSTH0, OCKS01, and OCKSm0 bits of the OCKS0 register (see section 12.4 (6)"I²C0 transfer clock setting method" on page 441).
Reset input clears these registers to 00H.

Figure 12-8: IIC function expansion registers 0 (IICX0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
IICX0	0	0	0	0	0	0	0	CLX0	FFFFFD85H,	R/W	00H

(6) I²C0 transfer clock setting method

The I²C0 transfer clock frequency (f_{SCL}) is calculated using the following expression.

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

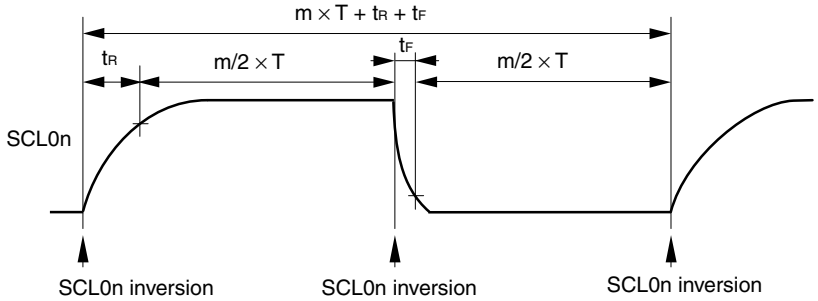
m =12, 18, 24, 36, 44, 48, 54, 60, 66, 72, 86, 88, 96, 132, 172, 176, 198, 220, 258, 344 (see Table 12-10, “Clock Settings,” on page 442).

T: $1/f_{XX}$
t_R: SCL0 pin rise time
t_F: SCL0 pin fall time

For example, the I²C0 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.

$$f_{SCL} = 1/(198 \times 52\ ns + 200\ ns + 50\ ns) \cong 94.7\ kHz$$

Figure 12-9: I²C0 transfer clock timing



The clock to be selected can be set by the combination of the SMC0, CL01, and CL00 bits of the IICCL0 register, the CLX0 bit of the IICX0 register, and the OCKSTH0, OCKS01, and OCKS00 bits of the OCKS0 register.

Table 12-10: Clock Settings

IICX0	IICCL0			Selection Clock	Transfer Clock	Settable Main Clock Frequency (f_{XX}) Range	Operating Mode
Bit 0	Bit 3	Bit 1	Bit 0				
CLX0	SMC0	CL01	CL00				
0	0	0	0	f_{XX} (when OCKS0=18H set)	$f_{XX}/44$	2.00 MHz ≤ f_{XX} ≤ 4.19 MHz	Standard mode (SMC0 = 0)
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/88$	4.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/132$	6.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKS0=12H set)	$f_{XX}/176$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/5$ (when OCKS0=13H set)	$f_{XX}/220$	10.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	0	0	1	f_{XX} (when OCKS0=18H set)	$f_{XX}/86$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/172$	8.38 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/258$	12.57 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKS0=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 OCKS0=18H set)	$f_{XX}/66$	6.4 MHz	
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/132$	12.8 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/198$	19.2 MHz	
0	1	0	×	f_{XX} (when OCKS0=18H set)	$f_{XX}/24$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	High-speed mode (SMC0 = 1)
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/48$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/72$	12.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKS0=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 OCKS0=18H set)	$f_{XX}/18$	6.4 MHz	
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/36$	12.8 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/54$	19.2 MHz	
1	1	0	×	f_{XX} (when OCKS0=18H set)	$f_{XX}/24$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/2$ (when OCKS0=10H set)	$f_{XX}/24$	8.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKS0=11H set)	$f_{XX}/36$	12.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKS0=12H set)	$f_{XX}/48$	16.00 MHz ≤ f_{XX} ≤ 16.67 MHz	
				$f_{XX}/5$ (when OCKS0=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 OCKS0 register, set the OCKS0 register to 00H (I²C division clock stopped status).

Remark: ×: don't care

(7) IIC division clock select registers 0 (OCKS0)

The OCKS0 registers control the I²C0 division clock.
These registers can be read or written in 8-bit units.
Reset input sets these registers to 01H.

Figure 12-11: IIC division clock select registers 0 (OCKS0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
OCKS0	0	0	0	OCKSEN0	OCKSTH0	0	OCKS01	OCKS00	FFFFF340H	R/W	01H

OCKSEN0	Operation setting of I²C division clock
0	Disable I²C division clock operation
1	Enable I²C division clock operation

OCKSTH0	OCKS01	OCKS00	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 (IIC0)

The IIC0 register is 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 IIC0 register during a data transfer.
A wait state is released by writing the IIC0 register during the wait period, and data transfer is started.
Reset input clears these registers to 00H.

Figure 12-12: IIC shift registers 0 (IIC0 to IIC2) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
IIC0									FFFFFD80H	R/W	00H

(9) Slave address registers 0 (SVA0)

The SVA0 registers hold the I²C bus's slave address.
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.

Figure 12-13: Slave address registers 0 (SVA0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
SVA0									FFFFFD83H	R/W	00H

12.5 I²C Bus Mode Functions

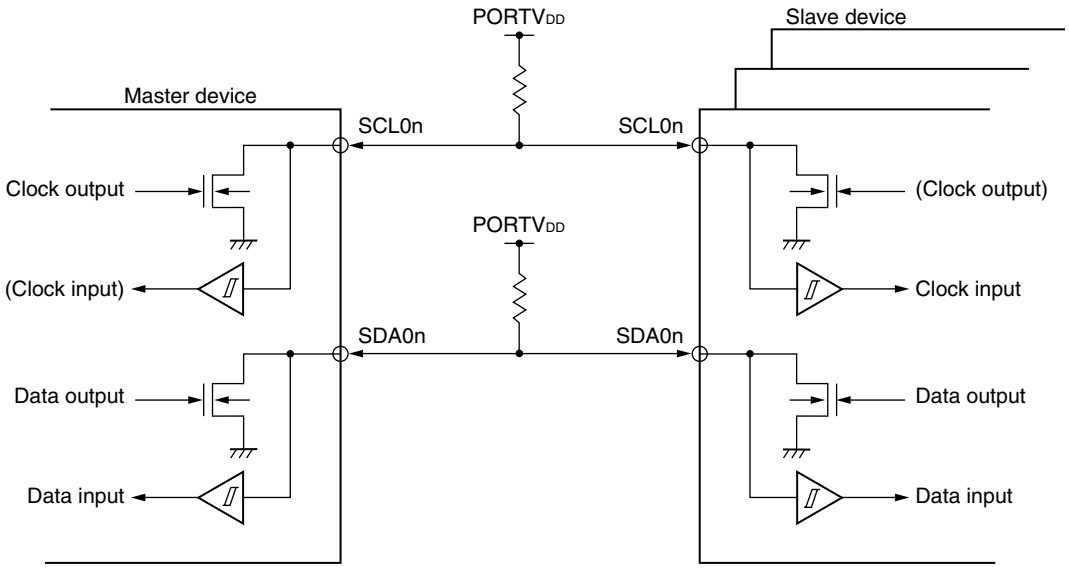
12.5.1 Pin configuration

The serial clock pin (SCL0) and serial data bus pin (SDA0) are configured as follows.

- SCL0 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.
- SDA0 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.

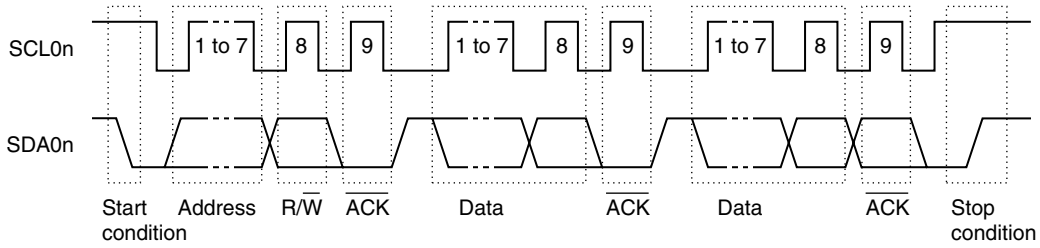
Figure 12-14: Pin Configuration Diagram



12.6 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 12-15: I²C Bus Serial Data Transfer Timing

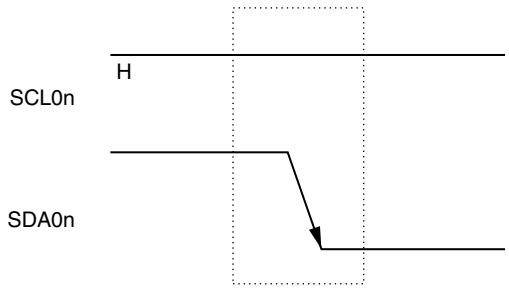


The master device outputs the start condition, slave address, and stop condition.
The acknowledge signal (ACK) can be output by either the master or slave device (normally, it is output by the device that receives 8-bit data).
The serial clock (SCL0) is continuously output by the master device. However, in the slave device, the SCL0 pin's low-level period can be extended and a wait can be inserted.

12.6.1 Start condition

A start condition is met when the SCL0 pin is high level and the SDA0 pin changes from high level to low level. The start condition for the SCL0 and SDA0 pins is a signal that the master device outputs to the slave device when starting a serial transfer. The slave device can defect the start condition.

Figure 12-16: Start Condition

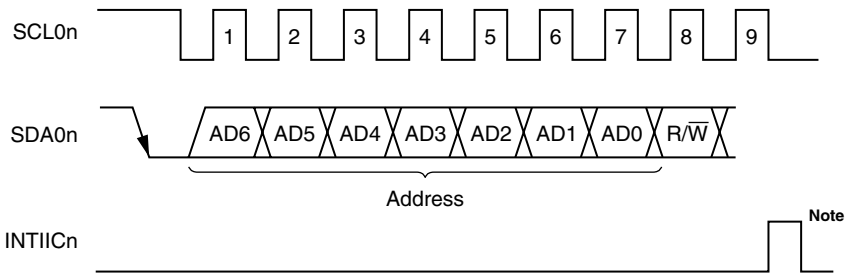


A start condition is output when the IICC0.STT0 bit is set (1) after a stop condition has been detected (IICS0.SPD0 bit = 1). When a start condition is detected, the IICS0.STD0 bit is set (1).

12.6.2 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 the bus lines. Therefore, each slave device connected via the bus lines must have an 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 SVA0 register. If the address data matches the values of the SVA0 register, the slave device is selected and communicates with the master device until the master device transmits a start condition or stop condition.

Figure 12-17: Address



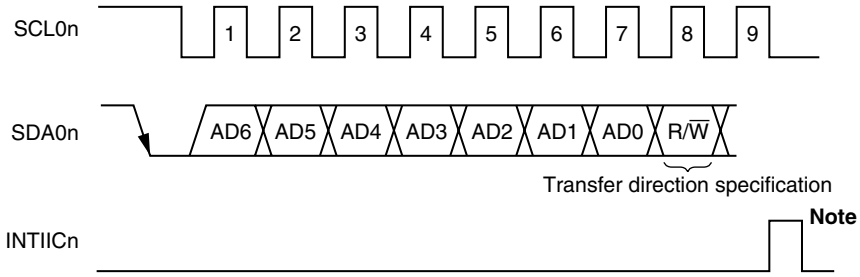
Note: The interrupt request signal (INTIIC0) is generated if a local address or extension code is received during slave device operation.

The slave address and the eighth bit, which specifies the transfer direction as described in section 12.6.3 ”Transfer direction specification” on page 448, are written together to IIC shift register 0 (IIC0) and then output. Received addresses are written to the IIC0 register.
The slave address is assigned to the higher 7 bits of the IIC0 register.

12.6.3 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 12-18: Transfer Direction Specification



Note: The INTIIC0 signal is generated if a local address or extension code is received during slave device operation.

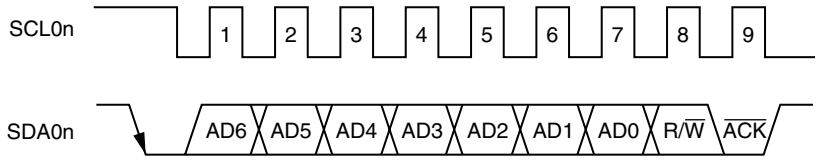
12.6.4 Acknowledge signal ($\overline{ACK}$)

The acknowledge signal ($\overline{ACK}$) is used by the transmitting and receiving devices to confirm serial data reception.
The receiving device returns one $\overline{ACK}$ signal for each 8 bits of data it receives. The transmitting device normally receives an $\overline{ACK}$ signal after transmitting 8 bits of data. However, when the master device is the receiving device, it does not output an $\overline{ACK}$ signal after receiving the final data to be transmitted. The transmitting device detects whether or not an $\overline{ACK}$ signal is returned after it transmits 8 bits of data. When an $\overline{ACK}$ signal is returned, the reception is judged as normal and processing continues. If the slave device does not return an $\overline{ACK}$ signal, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return an $\overline{ACK}$ signal may be caused by the following two factors.

(a) Reception was not performed normally.
(b) The final data was received.

When the receiving device sets the SDA0 line to low level during the ninth clock, the $\overline{ACK}$ signal becomes active (normal receive response).
When the IICC0.ACKE0 bit is set to 1, automatic $\overline{ACK}$ signal generation is enabled.
Transmission of the eighth bit following the 7 address data bits causes the IICS0.TRC0 bit to be set. When this TRC0 bit's value is 0, it indicates receive mode. Therefore, the ACKE0 bit should be set to 1. When the slave device is receiving (when TRC0 bit = 0), if the slave device does not need to receive any more data after receiving several bytes, clearing the ACKE0 bit to 0 will prevent the master device from starting transmission of the subsequent data.
Similarly, when the master device is receiving (when TRC0 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 $\overline{ACK}$ signal from being returned. This prevents the MSB from being output via the SDA0 line (i.e., stops transmission) during transmission from the slave device.

Figure 12-19: $\overline{ACK}$ Signal



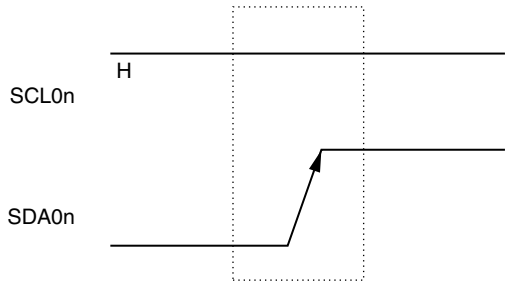
When the local address is received, an $\overline{ACK}$ signal is automatically output in synchronization with the falling edge of the SCL0n pin's eighth clock regardless of the value of the ACKEn bit. No $\overline{ACK}$ signal is output if the received address is not a local address.
The $\overline{ACK}$ signal output method during data reception is based on the wait timing setting, as described below.

When 8-clock wait is selected:	The $\overline{ACK}$ signal is output at the falling edge of the SCL0 pin's eighth clock if the
(IICC0.WTIM0 bit = 0)	ACKE0 bit is set to 1 before wait cancellation.
When 9-clock wait is selected:	The $\overline{ACK}$ signal is automatically output at the falling edge of the SCL0 pin's
(IICC0.WTIM0 bit = 1)	eighth clock if the ACKE0 bit has already been set to 1.

12.6.5 Stop condition

When the SCL0 pin is high level, changing the SDA0 pin from low level to high level generates a stop condition.
A stop condition is a signal that the master device outputs to the slave device when serial transfer has been completed. When used as the slave device, the start condition can be detected.

Figure 12-20: Stop Condition



A stop condition is generated when the IICC0.SPT0 bit is set to 1. When the stop condition is detected, the IICS0.SPD0 bit is set to 1 and the INTIIC0 signal is generated when the IICC0.SPIE0 bit is set to 1.

12.6.6 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 SCL0 pin to low level notifies the communication partner of the wait status. When the wait status has been cancelled for both the master and slave devices, the next data transfer can begin.

Figure 12-21: Wait Signal (1/2)

(a) When master device has a nine-clock wait and slave device has an eight-clock wait (master: transmission, slave: reception, and IICC0.ACKE0 bit = 1)

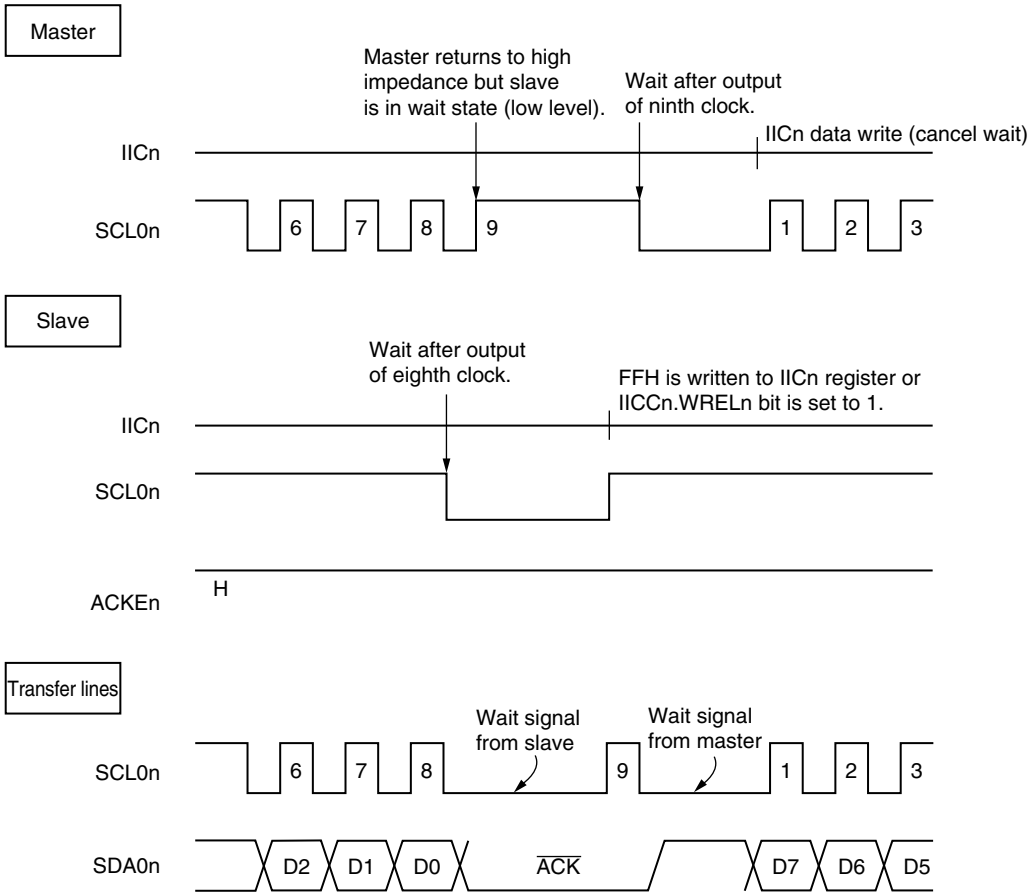
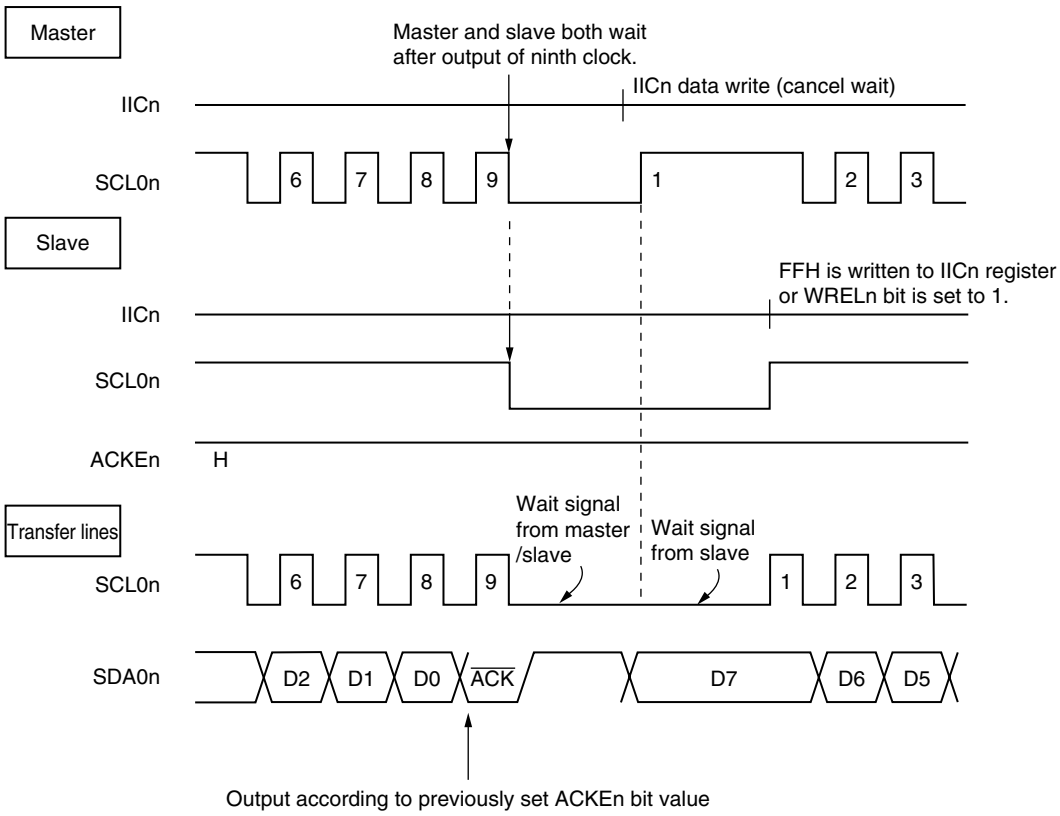


Figure 12-21: Wait Signal (2/2)

(b) When master and slave devices both have a nine-clock wait (master: transmission, slave: reception, and ACKE0 bit = 1)



A wait may be automatically generated depending on the setting of the IICC0.WTIM0 bit.
Normally, when the IICC0.WREL0 bit is set to 1 or when FFH is written to the IIC0 register on the receiving side, the wait status is cancelled and the transmitting side writes data to the IIC0 register to cancel the wait status.
The master device can also cancel the wait status via either of the following methods.

- By setting the IICC0.STT0 bit to 1
- By setting the IICC0.SPT0 bit to 1

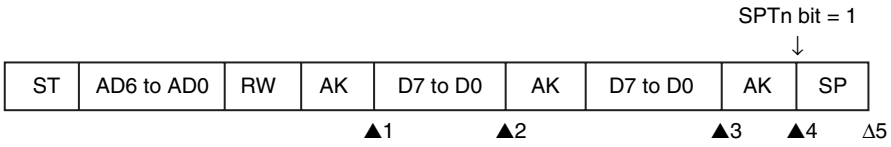
12.7 I²C Interrupt Request Signals (INTIIC0)

The following shows the value of the IICS0 register at the INTIIC0 interrupt request signal generation timing and at the INTIIC0 signal timing.

12.7.1 Master device operation

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

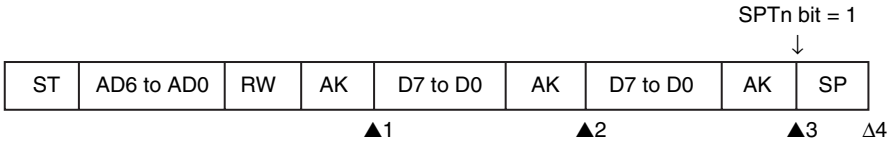
<1> When WTIM0 bit = 0



p1: IICS0 register = 10XXX110B
p2: IICS0 register = 10XXX000B
p3: IICS0 register = 10XXX000B (WTIM0 bit = 1)
p4: IICS0 register = 10XXXX00B
Δ5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1

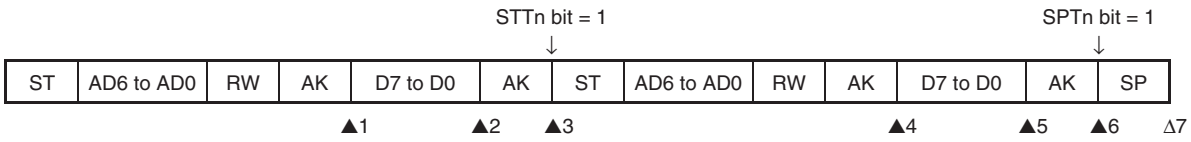


p1: IICS0 register = 10XXX110B
p2: IICS0 register = 10XXX100B
p3: IICS0 register = 10XXXX00B
Δ4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

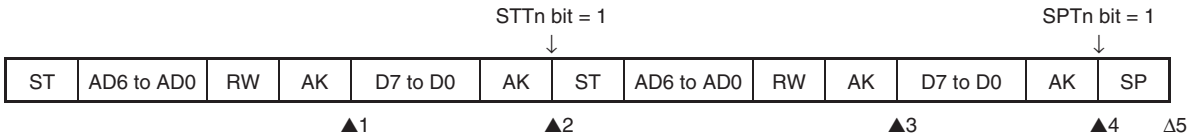
<1> When WTIM0 bit = 0



p1: IICS0 register = 10XXX110B
p2: IICS0 register = 10XXX000B (WTIM0 bit = 1)
p3: IICS0 register = 10XXXX00B (WTIM0 bit = 0)
p4: IICS0 register = 10XXX110B (WTIM0 bit = 0)
p5: IICS0 register = 10XXX000B (WTIM0 bit = 1)
p6: IICS0 register = 10XXXX00B
Δ 7: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1

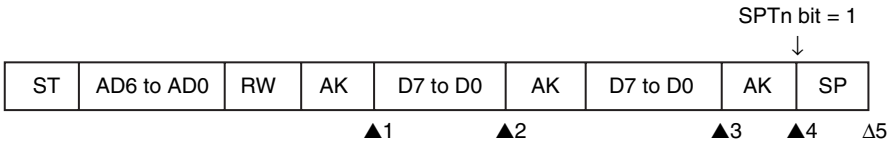


p1: IICS0 register = 10XXX110B
p2: IICS0 register = 10XXXX00B
p3: IICS0 register = 10XXX110B
p4: IICS0 register = 10XXXX00B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

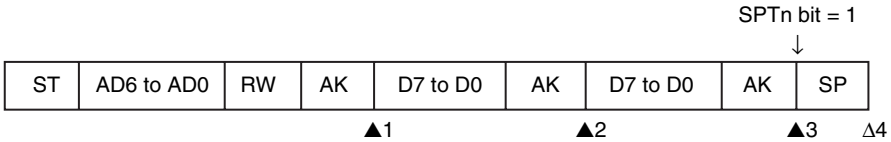
<1> When WTIM0 bit = 0



p1: IICS0 register = 1010X110B
p2: IICS0 register = 1010X000B
p3: IICS0 register = 1010X000B (WTIM0 bit = 1)
p4: IICS0 register = 1010XX00B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1



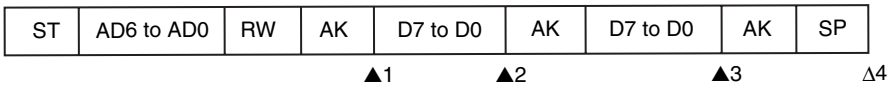
p1: IICS0 register = 1010X110B
p2: IICS0 register = 1010X100B
p3: IICS0 register = 1010XX00B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

12.7.2 Slave device operation (when receiving slave address data (address match))

(1) Start ~ Address ~ Data ~ Data ~ Stop

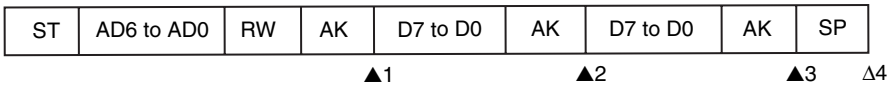
<1> When WTIM0 bit = 0



p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001X000B
p3: IICS0 register = 0001X000B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1



p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001X100B
p3: IICS0 register = 0001XX00B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address match)

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

p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001X000B
p3: IICS0 register = 0001X110B
p4: IICS0 register = 0001X000B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1 (after restart, address match)

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

p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001XX00B
p3: IICS0 register = 0001X110B
p4: IICS0 register = 0001XX00B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 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

p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001X000B
p3: IICS0 register = 0010X010B
p4: IICS0 register = 0010X000B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 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

p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001XX00B
p3: IICS0 register = 0010X010B
p4: IICS0 register = 0010X110B
p5: IICS0 register = 0010XX00B
Δ 6: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address mismatch (= 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

p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001X000B
p3: IICS0 register = 00000X10B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1 (after restart, address mismatch (= 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

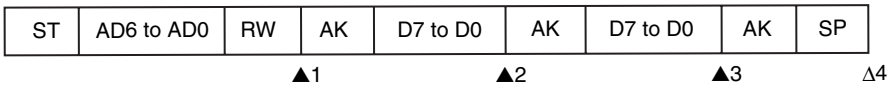
p1: IICS0 register = 0001X110B
p2: IICS0 register = 0001XX00B
p3: IICS0 register = 00000X10B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

12.7.3 Slave device operation (when receiving extension code)

(1) Start ~ Code ~ Data ~ Data ~ Stop

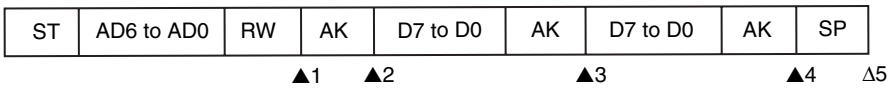
<1> When WTIM0 bit = 0



p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X000B
p3: IICS0 register = 0010X000B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1



p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X110B
p3: IICS0 register = 0010X100B
p4: IICS0 register = 0010XX00B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address match)

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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X000B
p3: IICS0 register = 0001X110B
p4: IICS0 register = 0001X000B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1 (after restart, address match)

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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X110B
p3: IICS0 register = 0010XX00B
p4: IICS0 register = 0001X110B
p5: IICS0 register = 0001XX00B
Δ 6: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X000B
p3: IICS0 register = 0010X010B
p4: IICS0 register = 0010X000B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X110B
p3: IICS0 register = 0010XX00B
p4: IICS0 register = 0010X010B
p5: IICS0 register = 0010X110B
p6: IICS0 register = 0010XX00B
Δ 7: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address mismatch (= 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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X000B
p3: IICS0 register = 00000X10B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1 (after restart, address mismatch (= 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

p1: IICS0 register = 0010X010B
p2: IICS0 register = 0010X110B
p3: IICS0 register = 0010XX00B
p4: IICS0 register = 00000X10B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

12.7.4 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

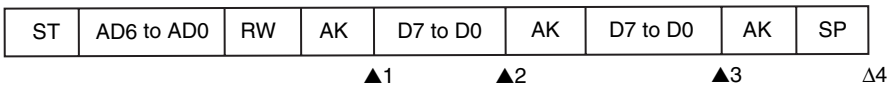
Δ 1: IICS0 register = 00000001B

Remark: Δ: Generated only when SPIE0 bit = 1

12.7.5 Arbitration loss operation (operation as slave after arbitration loss)

(1) When arbitration loss occurs during transmission of slave address data

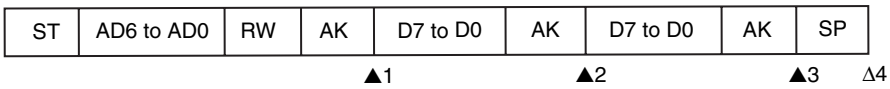
<1> When WTIM0 bit = 0



p1: IICS0 register = 0101X110B (Example: When ALD0 bit is read during interrupt servicing)
p2: IICS0 register = 0001X000B
p3: IICS0 register = 0001X000B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1

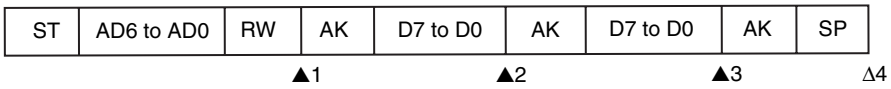


p1: IICS0 register = 0101X110B (Example: When ALD0 bit is read during interrupt servicing)
p2: IICS0 register = 0001X100B
p3: IICS0 register = 0001XX00B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(2) When arbitration loss occurs during transmission of extension code

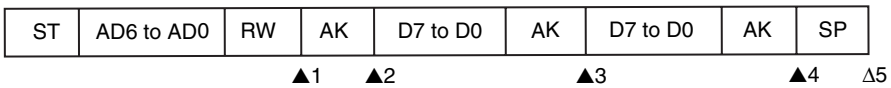
<1> When WTIM0 bit = 0



p1: IICS0 register = 0110X010B (Example: When ALD0 bit is read during interrupt servicing)
p2: IICS0 register = 0010X000B
p3: IICS0 register = 0010X000B
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

<2> When WTIM0 bit = 1

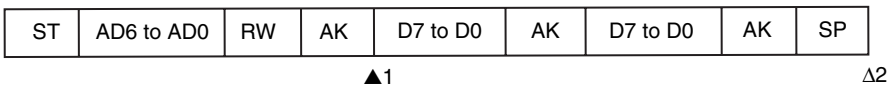


p1: IICS0 register = 0110X010B (Example: When ALD0 bit is read during interrupt servicing)
p2: IICS0 register = 0010X110B
p3: IICS0 register = 0010X100B
p4: IICS0 register = 0010XX00B
Δ 5: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

12.7.6 Operation when arbitration loss occurs (no communication after arbitration loss)

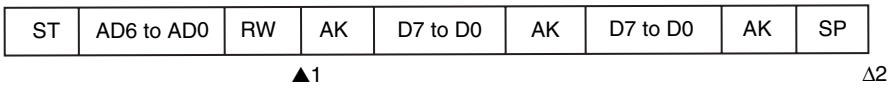
(1) When arbitration loss occurs during transmission of slave address data



p1: IICS0 register = 01000110B (Example: When ALD0 bit is read during interrupt servicing)
Δ 2: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1

(2) When arbitration loss occurs during transmission of extension code

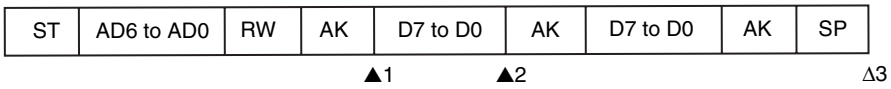


p1: IICS0 register = 0110X010B (Example: When ALD0 bit is read during interrupt servicing)
IICC0.LREL0 bit is set to 1 by software
Δ 2: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(3) When arbitration loss occurs during data transfer

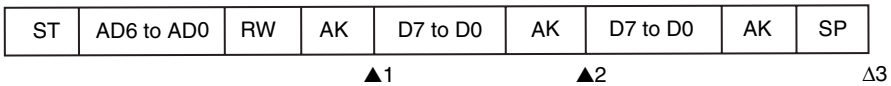
<1> When WTIM0 bit = 0



p1: IICS0 register = 10001110B
p2: IICS0 register = 01000000B (Example: When ALD0 bit is read during interrupt servicing)
Δ 3: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1

<2> When WTIM0 bit = 1



p1: IICS0 register = 10001110B
p2: IICS0 register = 01000100B (Example: When ALD0 bit is read during interrupt servicing)
Δ 3: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1

(4) When arbitration loss occurs due to restart condition during data transfer

<1> Not extension code (Example: Address mismatch)

ST	AD6 to AD0	RW	AK	D7 to Dn	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1					▲2		Δ3

p1: IICS0 register = 1000X110B
p2: IICS0 register = 01000110B (Example: When ALD0 bit is read during interrupt servicing)
Δ 3: IICS0 register = 00000001B

Remarks: 1. p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

2. D0 = D6 to D0

<2> Extension code

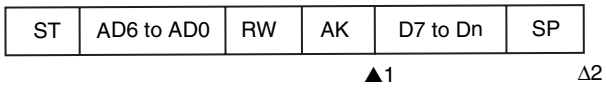
ST	AD6 to AD0	RW	AK	D7 to Dn	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
				▲1					▲2		Δ3

p1: IICS0 register = 1000X110B
p2: IICS0 register = 0110X010B (Example: When ALD0 bit is read during interrupt servicing)
IICC0.LREL0 bit is set to 1 by software
Δ 3: IICS0 register = 00000001B

Remarks: 1. p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

2. D0 = D6 to D0

(5) When arbitration loss occurs due to stop condition during data transfer



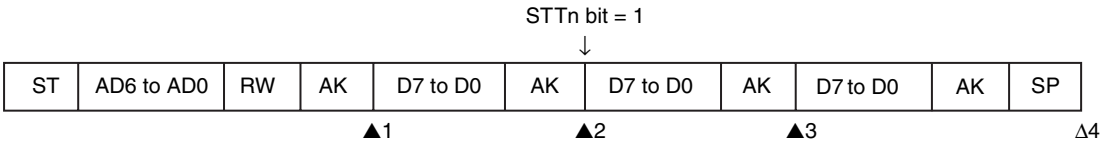
p1: IICS0 register = 1000X110B
Δ 2: IICS0 register = 01000001B

Remarks: 1. p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

2. D0 = D6 to D0

(6) When arbitration loss occurs due to low level of SDA0 pin when attempting to generate a restart condition

When WTIM0 bit = 1

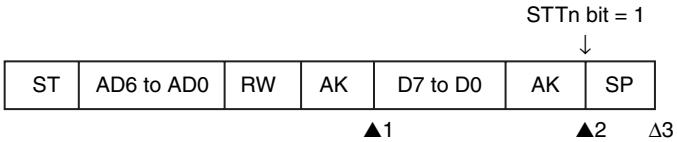


p1: IICS0 register = 1000X110B
p2: IICS0 register = 1000XX00B
p3: IICS0 register = 01000100B (Example: When ALD0 bit is read during interrupt servicing)
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(7) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

When WTIM0 bit = 1

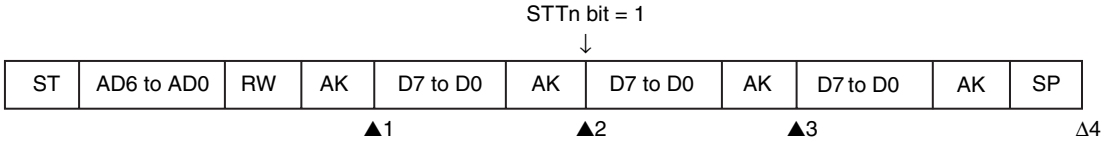


p1: IICS0 register = 1000X110B
p2: IICS0 register = 1000XX00B
Δ 3: IICS0 register = 01000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

(8) When arbitration loss occurs due to low level of SDA0 pin when attempting to generate a stop condition

When WTIM0 bit = 1



p1: IICS0 register = 1000X110B
p2: IICS0 register = 1000XX00B
p3: IICS0 register = 01000000B (Example: When ALD0 bit is read during interrupt servicing)
Δ 4: IICS0 register = 00000001B

Remark: p: Always generated
Δ: Generated only when SPIE0 bit = 1
×: don't care

12.8 Interrupt Request Signal (INTIIC0) Generation Timing and Wait Control

The setting of the IICC0.WTIM0 bit determines the timing by which the INTIIC0 register is generated and the corresponding wait control, as shown below.

Table 12-1: INTIIC0 Generation Timing and Wait Control

WTIM0 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 INTIIC0 signal and wait period occur at the falling edge of the ninth clock only when there is a match with the address set to the SVA0 register.
At this point, the ACK signal is output regardless of the value set to the IICC0.ACKE0 bit.
For a slave device that has received an extension code, the INTIIC0 signal occurs at the falling edge of the eighth clock.
When the address does not match after restart, the INTIIC0 signal is generated at the falling edge of the ninth clock, but no wait occurs.

2. If the received address does not match the contents of the SVA0 register and an extension code is not received, neither the INTIIC0 signal nor a wait occurs.

Remark: The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

(1) During address transmission/reception

- Slave device operation: Interrupt and wait timing are determined regardless of the WTIM0 bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing is determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing is determined according to the WTIM0 bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By setting the IICC0.WREL0 bit to 1
- By writing to the IIC0 register
- By start condition setting (IICC0.STT0 bit = 1)[Note]
- By stop condition setting (IICC0.SPT0 bit = 1)[Note]

Note: Master only

When an 8-clock wait has been selected (WTIM0 bit = 0), the output level of the ACK signal must be determined prior to wait cancellation.

(5) Stop condition detection

The INTIIC0 signal is generated when a stop condition is detected.

12.9 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 INTIIC0 signal occurs when a local address has been set to the SVA0 register and when the address set to the SVA0 register matches the slave address sent by the master device, or when an extension code has been received.

12.10 Error Detection

In I²C bus mode, the status of the serial data bus pin (SDA0) during data transmission is captured by the IIC0 register of the transmitting device, so the data of the IIC0 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.

12.11 Extension Code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (IICS0.EXC0 bit) is set for extension code reception and an interrupt request signal (INTIIC0) is issued at the falling edge of the eighth clock.
The local address stored in the SVA0 register is not affected.

(2) If 11110xx0 is set to the SVA0 register by a 10-bit address transfer and 11110xx0 is transferred from the master device, the results are as follows. Note that the INTIIC0 signal occurs at the falling edge of the eighth clock

- Higher four bits of data match: EXC0 bit = 1
- Seven bits of data match: IICS0.COI0 bit = 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 IICC0.LREL0 bit to 1 and the CPU will enter the next communication wait state.

Table 12-2: Extension Code Bit Definitions

Slave Address	R/W Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address that is reserved for different bus format
1111 0xx	×	10-bit slave address specification

12.12 Arbitration

When several master devices simultaneously output a start condition (when the IICC0.STT0 bit is set to 1 before the IICS0.STD0 bit is set to 1), communication between the master devices is performed while the number of clocks is adjusted until the data differs. This kind of operation is called arbitration.
When one of the master devices loses in arbitration, an arbitration loss flag (IICS0.ALD0 bit) is set to 1 via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 lines are both set to high impedance, which releases the bus.
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 setting of the ALD0 bit to 1, which is made by software.
For details of interrupt request timing, see 12.7 ”I²C Interrupt Request Signals (INTIIC0)” on page 453.

Figure 12-22: Arbitration Timing Example

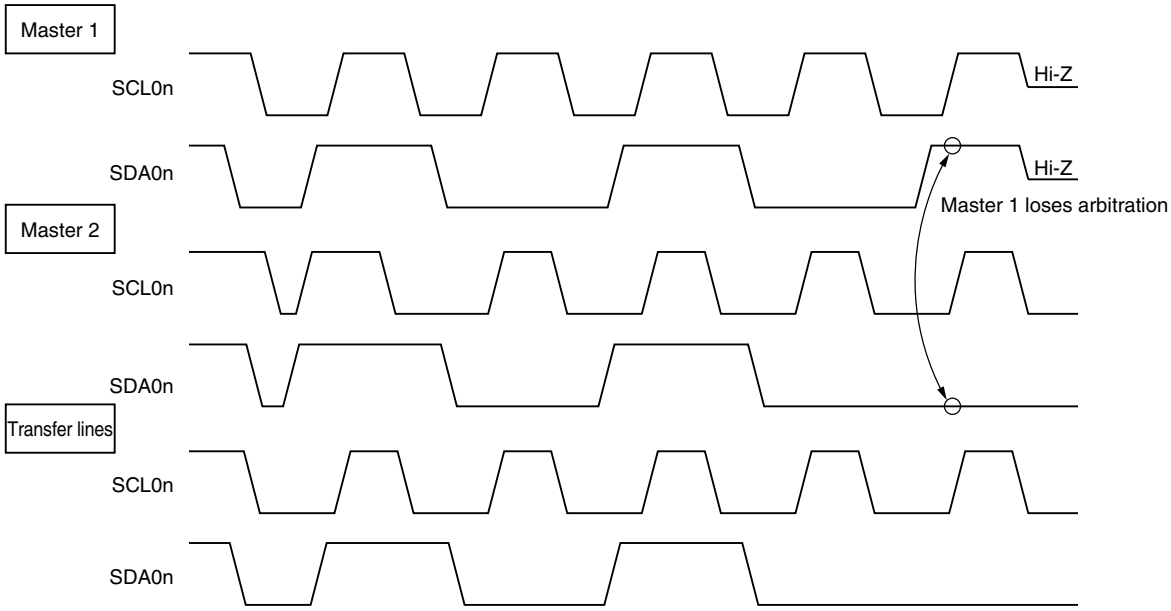


Table 12-3: Status During Arbitration and Interrupt Request Signal Generation Timing

Status During Arbitration	Interrupt Request Generation Timing
Transmitting address transmission	At falling edge of eighth or ninth clock following byte transfer[Note 1]
Read/write data after address transmission	
Transmitting extension code	
Read/write data after extension code transmission	
Transmitting data	
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 IICC0.SPIE0 bit = 1)[Note 2]
When SDA0 pin is low level while attempting to output restart condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When stop condition is detected while attempting to output restart condition	When stop condition is output (when IICC0.SPIE0 bit = 1)[Note 2]
When DSA00 pin is low level while attempting to output stop condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When SCL0 pin is low level while attempting to output restart condition	

Notes: 1. When the IICC0.WTIM0 bit = 1, an interrupt request signal occurs at the falling edge of the ninth clock. When the WTIM0 bit = 0 and the extension code's slave address is received, an interrupt request signal occurs at the falling edge of the eighth clock.

2. When there is a possibility that arbitration will occur, set the SPIE0 bit to 1 for master device operation.

12.13 Wake-up Function

The I²C bus slave function is a function that generates an interrupt request signal (INTIIC0) 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 IICC0.SPIE0 bit is set regardless of the wake-up function, and this determines whether interrupt request signals are enabled or disabled.

12.14 Communication Reservation

12.14.1 When communication reservation function is enabled (IICF0.IICRSV0 bit = 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 in 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 (ACK signal is not returned and the bus was released when the IICC0.LREL0 bit was set to 1).

If the IICC0.STT0 bit is set to 1 while the bus is not used, a start condition is automatically generated and a 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 IIC0 register causes master address transfer to start. At this point, the IICC0.SPIE0 bit should be set to 1.
When STT0 has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- If the bus has been released — A start condition is generated
- If the bus has not been released (standby mode) — Communication reservation

To detect which operation mode has been determined for the STT0 bit, set the STT0 bit to 1, wait for the wait period, then check the IICS0.MSTS0 bit.
The wait periods, which should be set via software, are listed in Table 17-6. These wait periods can be set by the SMC0, CL01, and CL00 bits of the IICCL0 register and the IICX0.CLX0 bit.

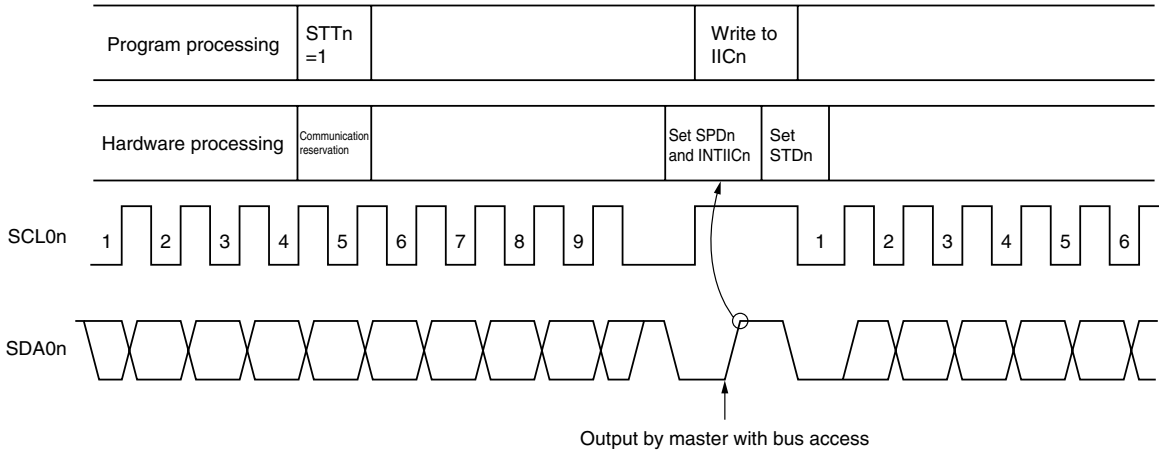
Table 12-4: Wait Periods

Clock Selection	CLX0	SMC0	CL01	CL00	Wait Period
f_{XX} (when OCKS0 = 18H set)	0	0	0	0	26 clocks
$f_{XX}/2$ (when OCKS0 = 10H set)	0	0	0	0	52 clocks
$f_{XX}/3$ (when OCKS0 = 11H set)	0	0	0	0	78 clocks
$f_{XX}/4$ (when OCKS0 = 12H set)	0	0	0	0	104 clocks
$f_{XX}/5$ (when OCKS0 = 13H set)	0	0	0	0	130 clocks
f_{XX} (when OCKS0 = 18H set)	0	0	0	1	47 clocks
$f_{XX}/2$ (when OCKS0 = 10H set)	0	0	0	1	94 clocks
$f_{XX}/3$ (when OCKS0 = 11H set)	0	0	0	1	141 clocks
$f_{XX}/4$ (when OCKS0 = 12H set)	0	0	0	1	188 clocks
f_{XX}	0	0	1	0	47 clocks
f_{XX} (when OCKS0 = 18H set)	0	1	0	×	16 clocks
$f_{XX}/2$ (when OCKS0 = 10H set)	0	1	0	×	32 clocks
$f_{XX}/3$ (when OCKS0 = 11H set)	0	1	0	×	48 clocks
$f_{XX}/4$ (when OCKS0 = 12H set)	0	1	0	×	64 clocks
f_{XX}	0	1	1	0	16 clocks
f_{XX} (when OCKS0 = 18H set)	1	1	0	×	10 clocks
$f_{XX}/2$ (when OCKS0 = 10H set)	1	1	0	×	20 clocks
$f_{XX}/3$ (when OCKS0 = 11H set)	1	1	0	×	30 clocks
$f_{XX}/4$ (when OCKS0 = 12H set)	1	1	0	×	40 clocks
f_{XX}	1	1	1	0	10 clocks

Remark: × = don't care

The communication reservation timing is shown below.

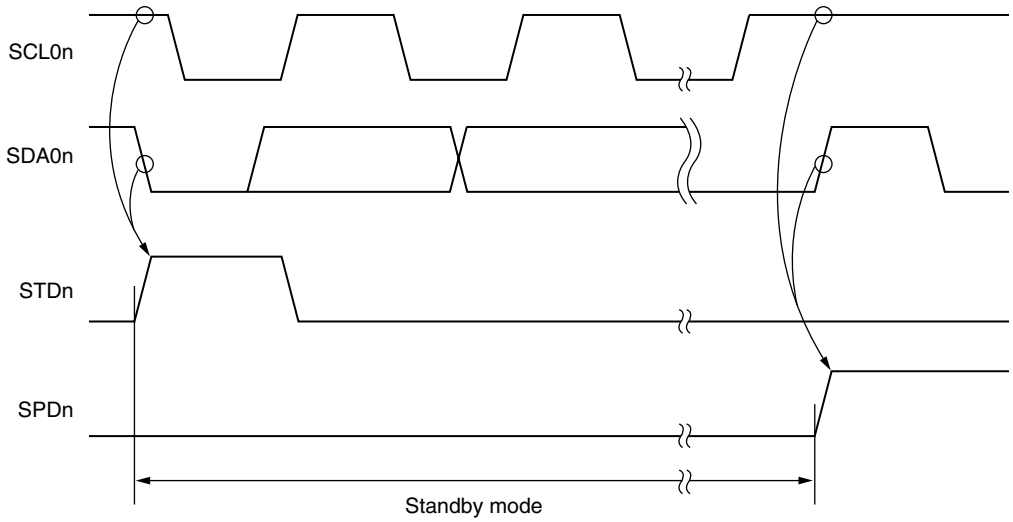
Figure 12-23: Communication Reservation Timing



Remark: STT0: Bit of IICC0 register
STD0: Bit of IICS0 register
SPD0: Bit of IICS0 register

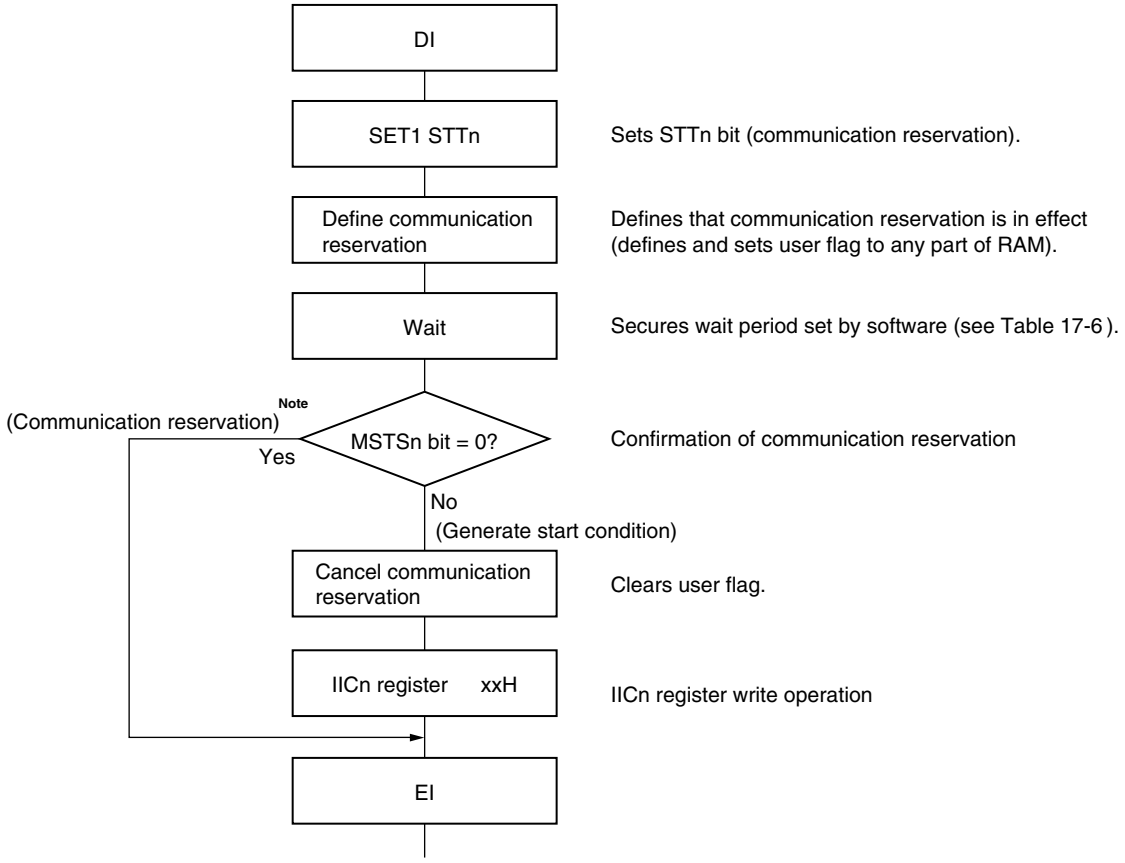
Communication reservations are accepted via the following timing. After the STD0 bit of the IICS0 register is set to 1, a communication reservation can be made by setting the STT0 bit of the IICC0 register to 1 before a stop condition is detected.

Figure 12-24: Timing for Accepting Communication Reservations



The communication reservation flowchart is illustrated below.

Figure 12-25: Communication Reservation Flowchart



Note: The communication reservation operation executes a write to the IIC0 register when a stop condition interrupt request occurs.

12.14.2 When communication reservation function is disabled (IICF0.IICRSV0 bit = 1)

When the IICC0.STT0 bit 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. There are two modes in 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 (ACK signal is not returned and the bus was released when the IICC0.LREL0 bit was set to 1).

To confirm whether the start condition was generated or request was rejected, check the IICF0.STCF0 flag. The time shown in Table 12-5 is required until the STCF0 flag is set after setting the STT0 bit to 1. Therefore, secure the time by software.

Table 12-5: Wait Periods

OCKSEN0	OCKS01	OCKS00	CL01	CL00	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

Remark: ×: don't care

12.15 Cautions

(1) When IICF0.STCEN0 bit = 0

Immediately after the I²C0 operation is enabled, the bus communication status (IICF0.IICBSY0 bit = 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 IICCL0 register.
<2> Set the IICC0.IICE0 bit.
<3> Set the IICC0.SPT0 bit.

(2) When IICF0.STCEN0 bit = 1

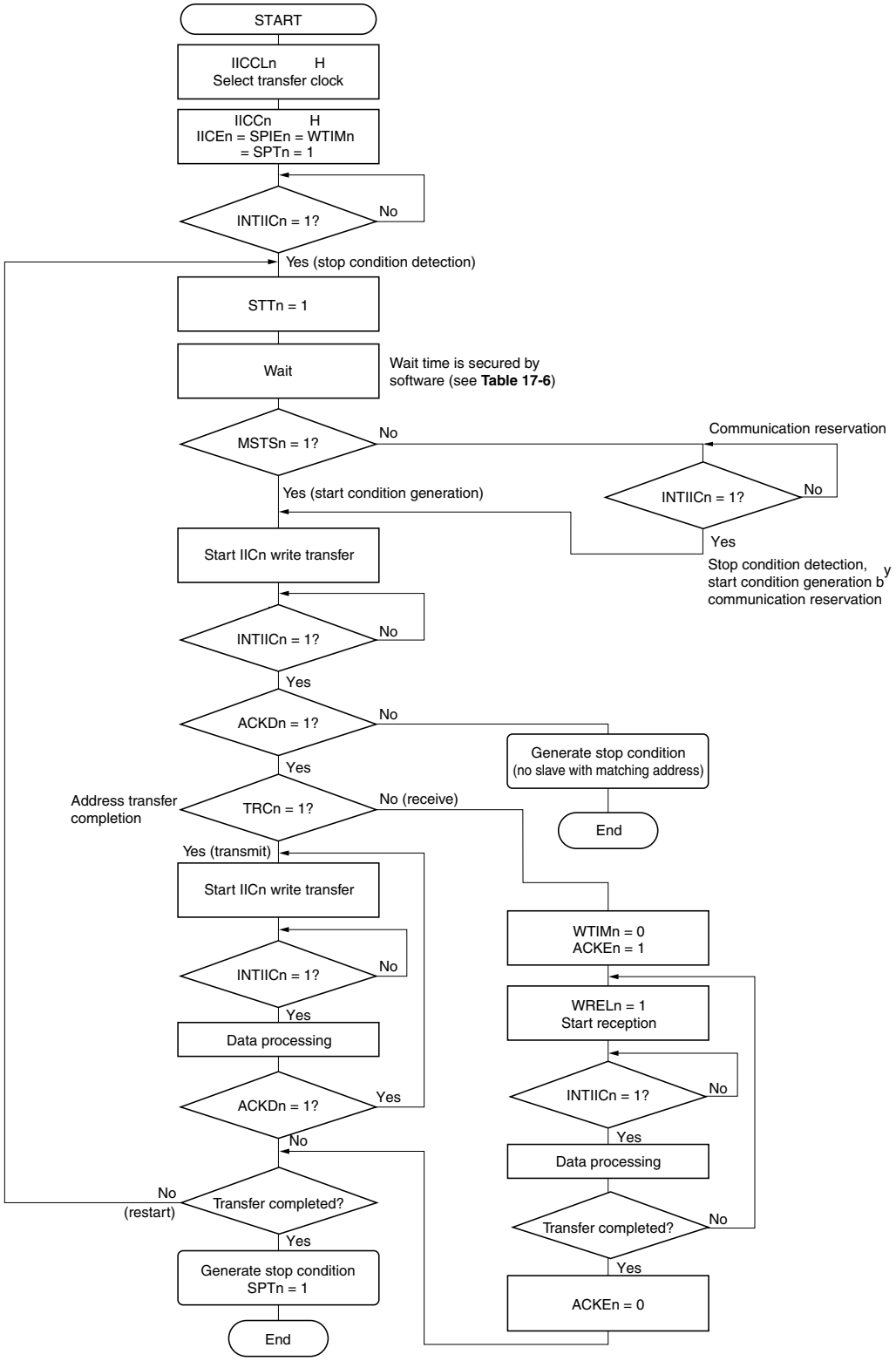
Immediately after I²C0 operation is enabled, the bus released status (IICBSY0 bit = 0) is recognized regardless of the actual bus status. To issue the first start condition (IICC0.STT0 bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

12.16 Communication Operations

12.16.1 Master operation 1

The following shows the flowchart for master communication when the communication reservation function is enabled (IICF0.IICRSV0 bit = 0) and the master operation is started after a stop condition is detected (IICF0.STCEN0 bit = 0).

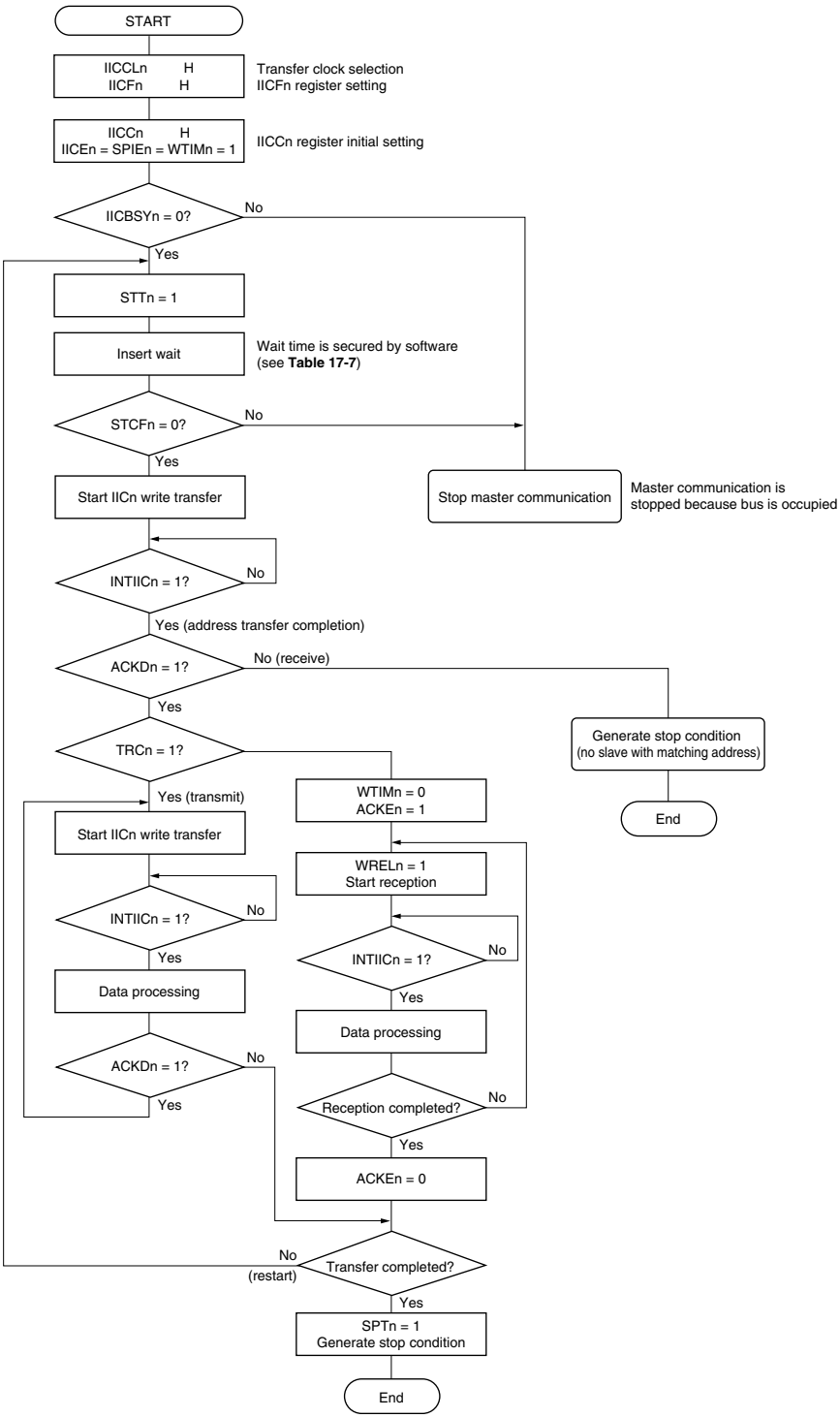
Figure 12-26: Master Operation Flowchart (1/2)



12.16.2 Master operation 2

The following shows the flowchart for master communication when the communication reservation function is disabled (IICRSV0 bit = 1) and the master operation is started without detecting a stop condition (STCEN0 bit = 1).

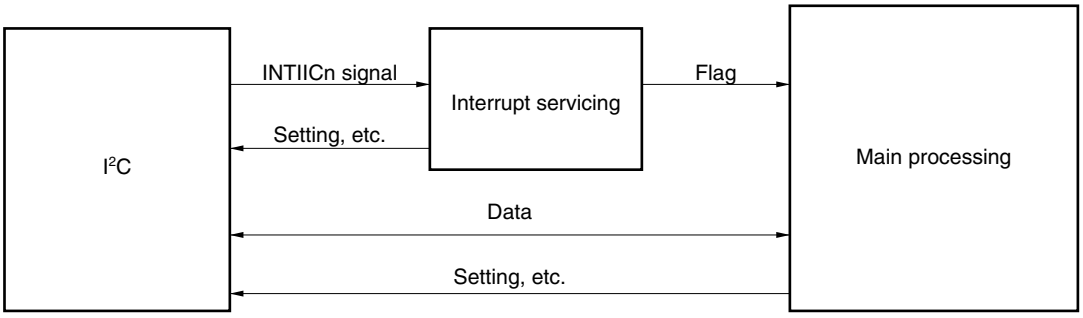
Figure 12-26: Master Operation Flowchart (2/2)



12.16.3 Slave operation

The following shows the processing procedure of the slave operation.
Basically, the operation of the slave device is event-driven. Therefore, processing by an INTIIC0 interrupt (processing requiring a significant change of the operation status, such as stop condition detection during communication) is necessary.
The following description assumes that data communication does not support extension codes. Also, it is assumed that the INTIIC0 interrupt servicing performs only status change processing and that the actual data communication is performed during the main processing.

Figure 12-27: Software Outline During Slave Operation



Therefore, the following three flags are prepared so that the data transfer processing can be performed by transmitting these flags to the main processing instead of INTIIC0 signal.

(1) Communication mode flag

This flag indicates the following communication statuses.

- Clear mode: Data communication not in progress
- Communication mode: Data communication in progress (valid address detection stop condition detection, ACK signal from master not detected, address mismatch)

(2) Ready flag

This flag indicates that data communication is enabled. This is the same status as an INTIICn interrupt during normal data transfer. This flag is set in the interrupt processing block and cleared in the main processing block. The ready flag for the first data for transmission is not set in the interrupt processing block, so the first data is transmitted without clear processing (the address match is regarded as a request for the next data).

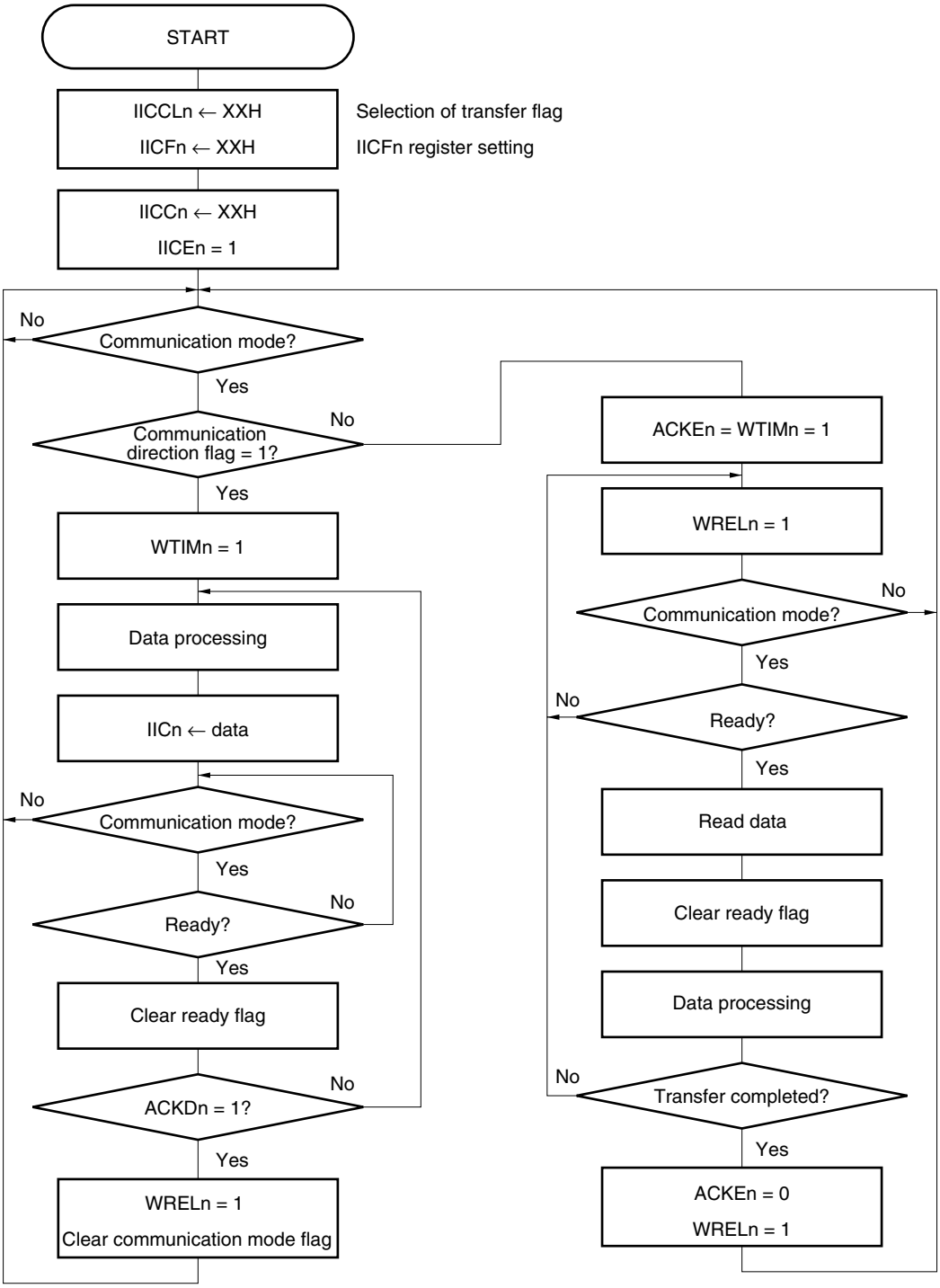
(3) Communication direction flag

This flag indicates the direction of communication and is the same as the value of IICS0.TRC0 bit.

The following shows the operation of the main processing block during slave operation.
Start I²C0 and wait for the communication enabled status. When communication is enabled, perform transfer using the communication mode flag and ready flag (the processing of the stop condition and start condition is performed by interrupts, conditions are confirmed by flags).
For transmission, repeat the transmission operation until the master device stops returning ACK signal. When the master device stops returning ACK signal, transfer is complete.

For reception, receive the required number of data and do not return ACK signal for the next data immediately after transfer is complete. After that, the master device generates the stop condition or restart condition. This causes exit from communications.

Figure 12-28: Slave Operation Flowchart (1/2)

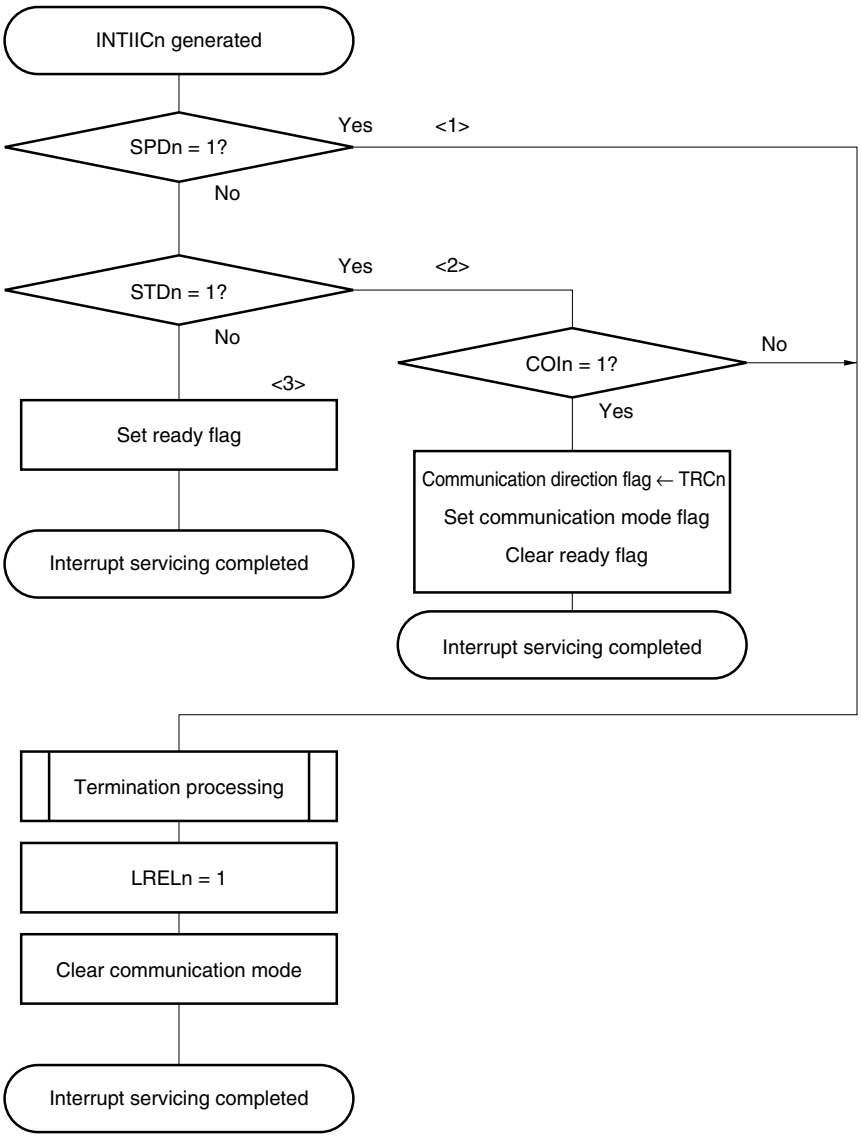


The following shows an example of the processing of the slave device by an INTIIC0 interrupt (it is assumed that no extension codes are used here). During an INTIIC0 interrupt, the status is confirmed and the following steps are executed.

<1> When a stop condition is detected, communication is terminated.
<2> When a start condition is detected, the address is confirmed. If the address does not match, communication is terminated. If the address matches, the communication mode is set and wait is released, and operation returns from the interrupt (the ready flag is cleared).
<3> For data transmission/reception, when the ready flag is set, operation returns from the interrupt while the IIC0 bus remains in the wait status.

Remark: <1> to <3> in the above correspond to <1> to <3> in **Figure 12-28: Slave Operation Flowchart (2/2)**.

Figure 12-28: Slave Operation Flowchart (2/2)

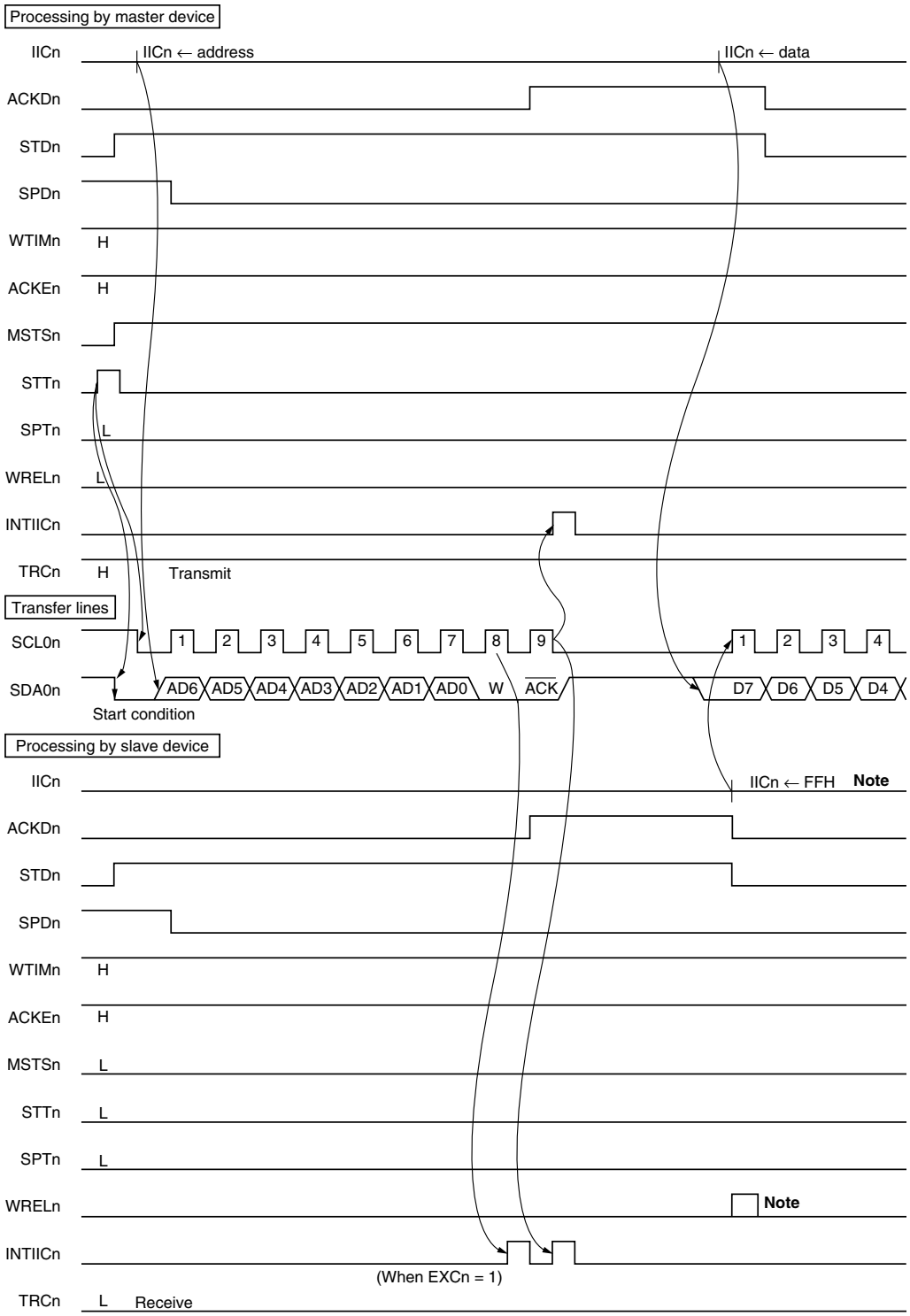


12.17 Timing of Data Communication

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 IICS0.TRC0 bit, which specifies the data transfer direction, and then starts serial communication with the slave device.
The shift operation of the IIC0 register is synchronized with the falling edge of the serial clock pin (SCL0). The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0 pin.
Data input via the SDA0 pin is captured by the IIC0 register at the rising edge of the SCL0 pin.
The data communication timing is shown below.

Figure 12-29: Example of Master to Slave Communication when 9-Clock wait Is selected for both Master and Slave) (1/3)

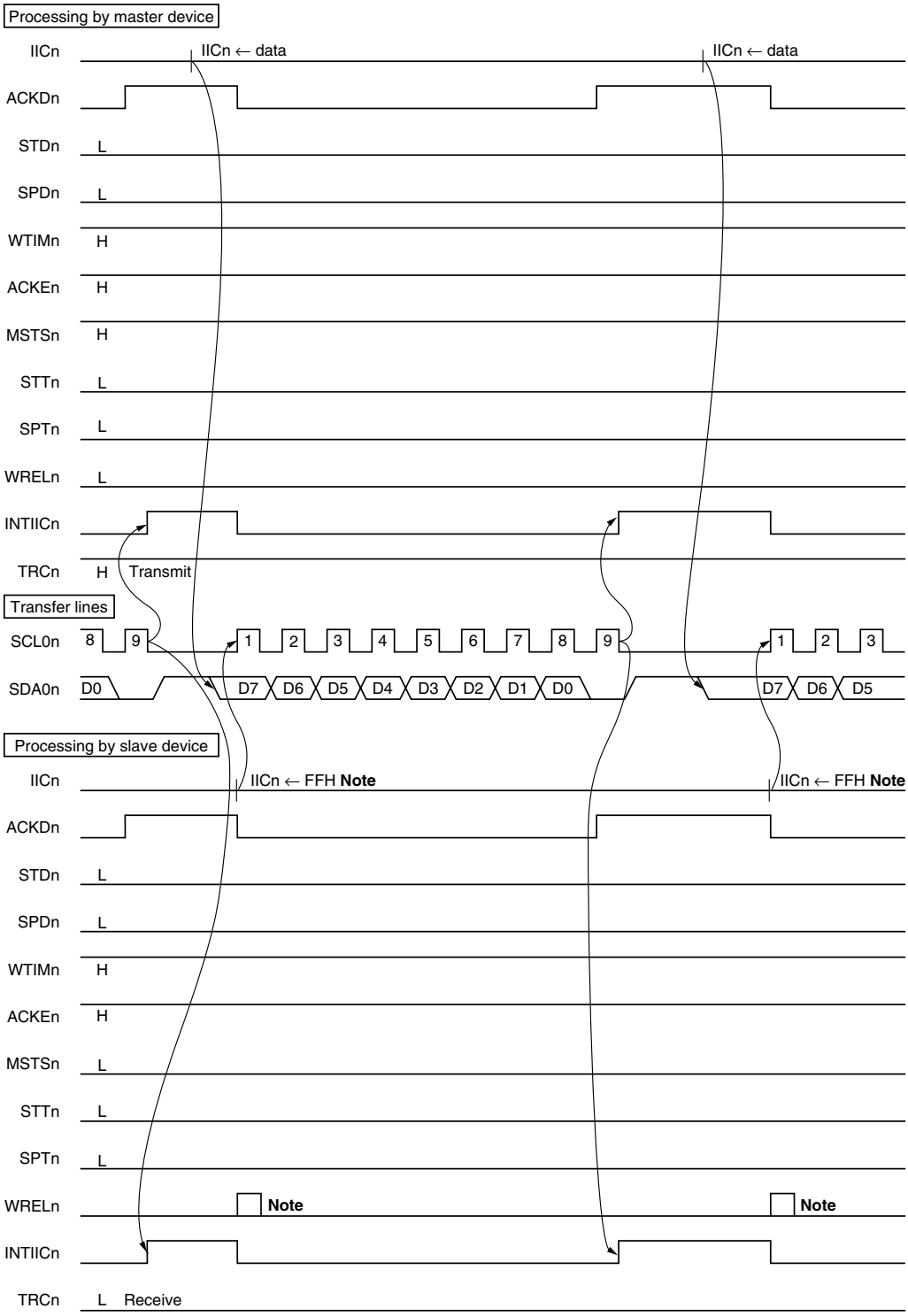
(a) Start condition ~ Address



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

Figure 12-29: Example of Master to Slave Communication when 9-Clock wait Is selected for both Master and Slave) (2/3)

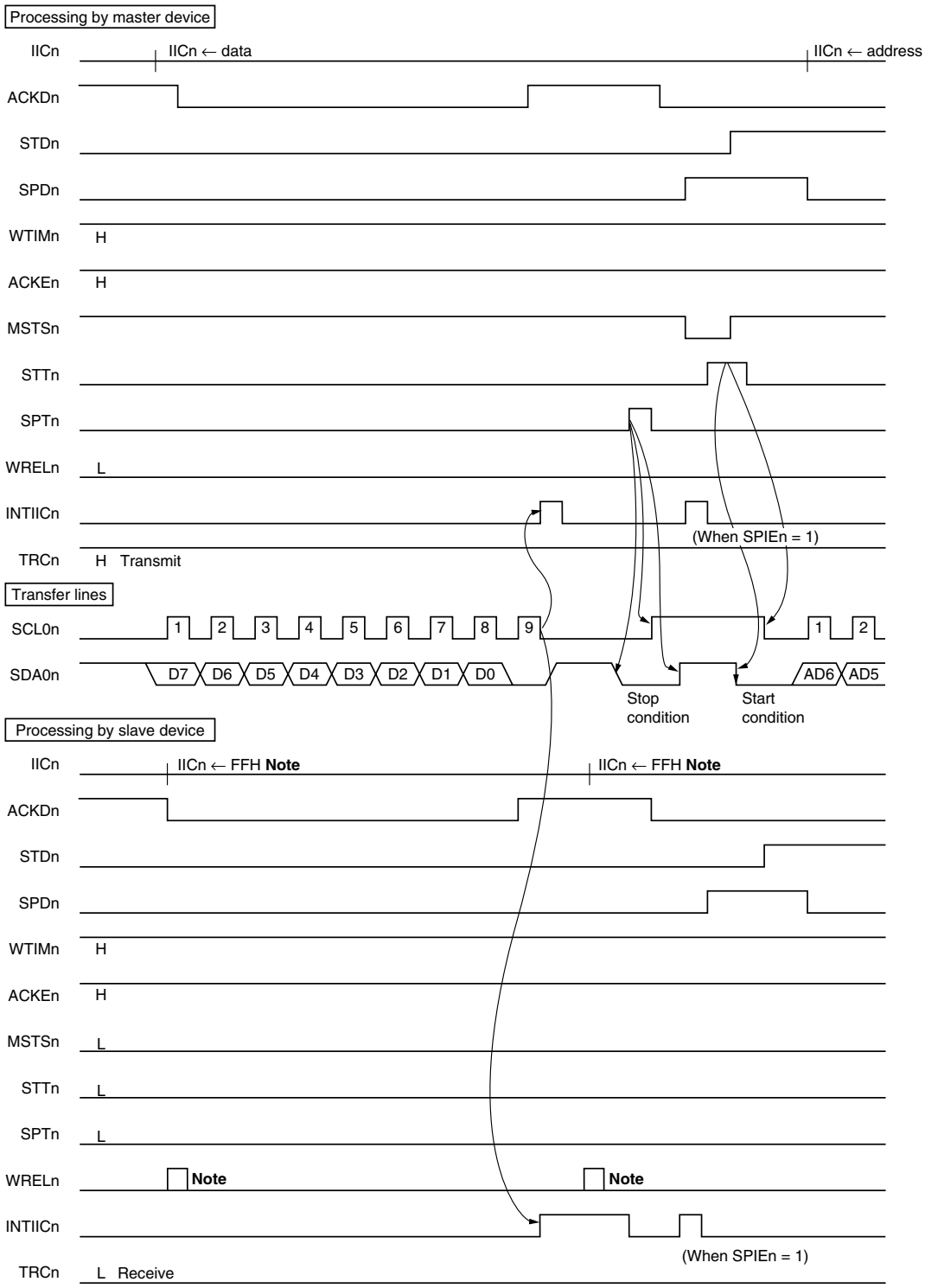
(b) Data



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

Figure 12-29: Example of Master to Slave Communication when 9-Clock wait Is selected for both Master and Slave) (3/3)

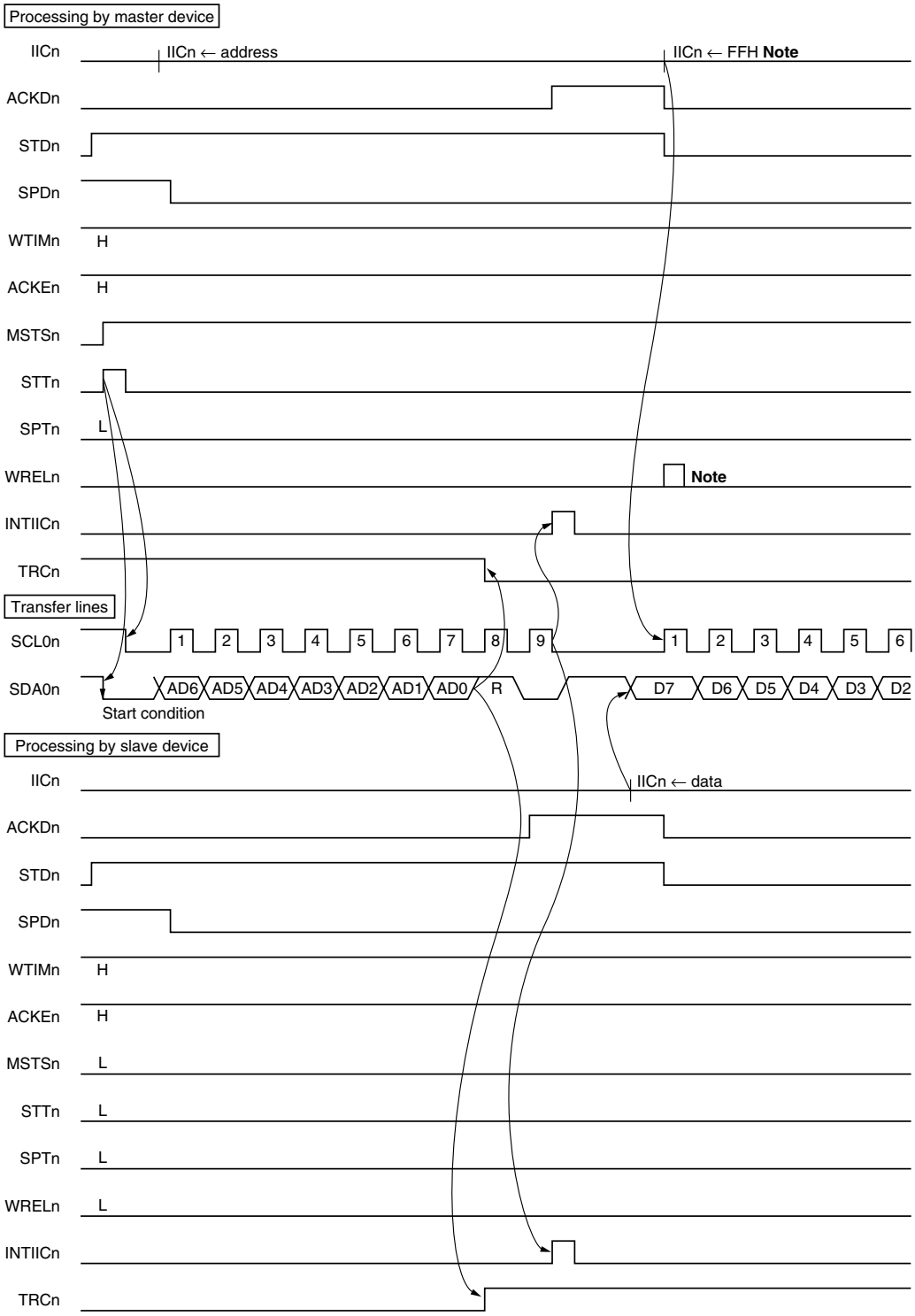
(c) Stop condition



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

Figure 12-30: Example of Slave to Master Communication (when 9-Clock wait Is selected for both Master and Slave) (1/3)

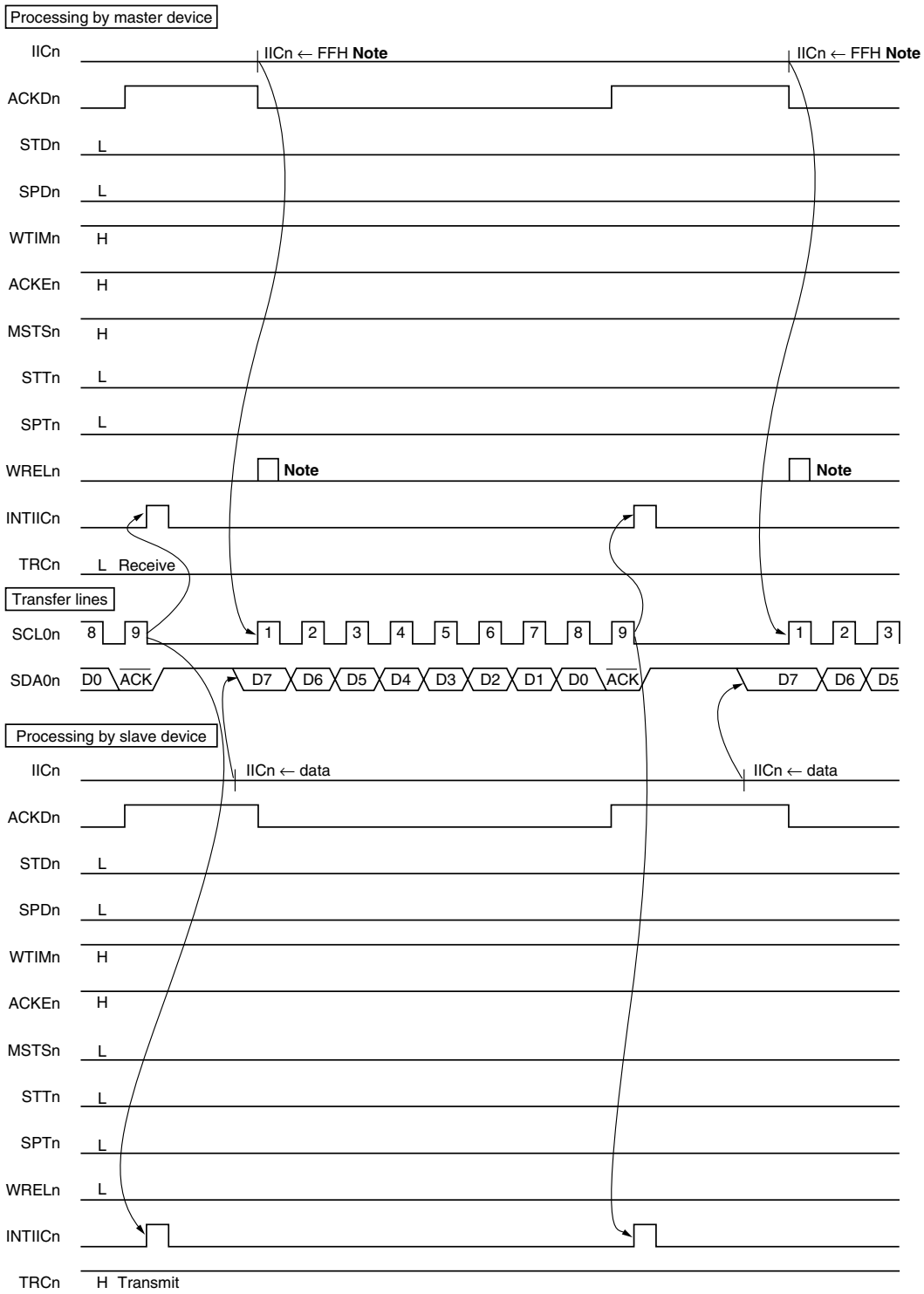
(a) Start condition ~ Address



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

Figure 12-30: Example of Slave to Master Communication (when 9-Clock wait Is selected for both Master and Slave) (2/3)

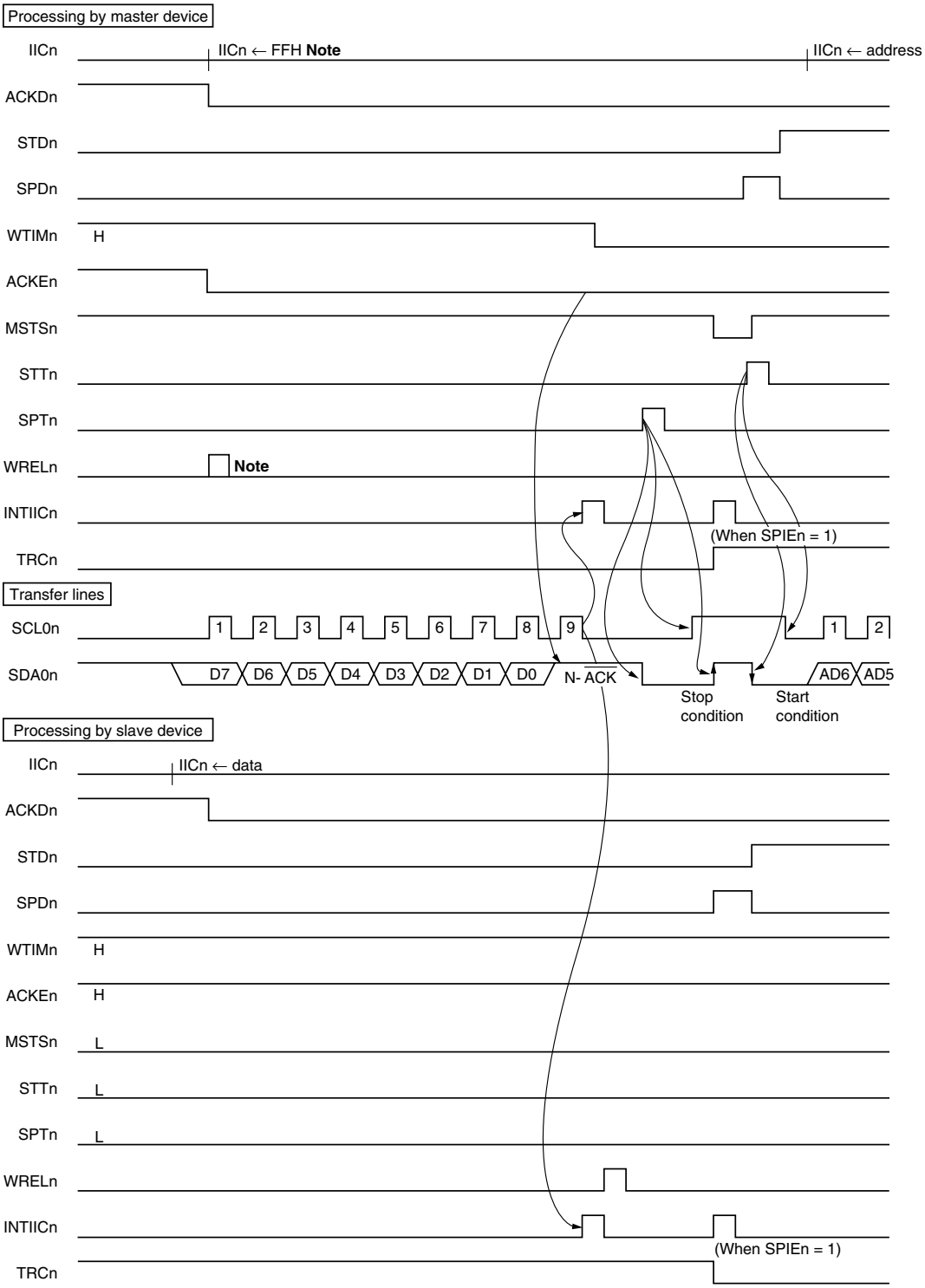
(b) Data



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

Figure 12-30: Example of Slave to Master Communication (when 9-Clock wait Is selected for both Master and Slave) (3/3)

(c) Stop condition



Note: To cancel slave wait, write FFH to IIC0 or set WREL0.

[MEMO]

Chapter 13 AFCAN Controller

13.1 Outline Description

This product features an on-chip 5-channel CAN (Controller Area Network) controller that complies with CAN protocol as standardized in ISO 11898.

13.1.1 Features

- Compliant with ISO 11898 and tested according to ISO/DIS 16845 (CAN conformance test)
- Standard frame and extended frame transmission/reception enabled
- Transfer rate: 1 Mbps max. (CAN clock input ≥ 16 MHz)
- 48 message buffers/5 channels
- Receive/transmit history list function
- Automatic block transmission function
- Multi-buffer receive block function
- Mask setting of four patterns is possible for each channel

13.1.2 Implementation

Implemented in CarGate+ are 4 AFCAN and 1 DAFCAN modules. The actual mapping of the SFR depends on the content of the BPC register (please refer to 3.7.1 ”Programmable peripheral I/O registers” on page 185). For a complete list please refer to 3.7 ”PPA related SFR Register” on page 79.

Each module has it’s own Channel Offset to this Base Address. These offsets are as follows:

Table 13-1: CAN Channel Offsets

CAN Channel	Channel Offset	Type
CAN0	0x0000	AFCAN
CAN1	0x0800	
CAN2	0x1000	
CAN3	0x1800	
CAN4	0x2000	DAFCAN

13.1.3 Overview of functions

Table 13-2 presents an overview of the CAN controller functions.

Table 13-2: Overview of Functions

Function	Details
Protocol	CAN protocol ISO 11898 (standard and extended frame transmission/reception)
Baud rate	Maximum 1 Mbps (CAN clock input ≥ 16 MHz)
Data storage	Storing messages in the CAN RAM
Number of messages	- 48 message buffers/5 channels - 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 to each message buffer. - Mask setting of four patterns is possible for each channel. - A receive completion interrupt is generated each time a message is received and stored in a message buffer. - Two or more receive message buffers can be used as a FIFO receive buffer (multi-buffer receive block function). - Receive history list function
Message transmission	- Unique ID can be set to each message buffer. - Transmit completion interrupt for each message buffer - Message buffer number 0 to 7 specified as the transmit message buffer can be used for automatic block transfer. Message transmission interval is programmable (automatic block transmission function (hereafter referred to as “ABT”)). - Transmission history list function
Remote frame processing	Remote frame processing by transmit message buffer
Time stamp function	- The time stamp function can be set for a message reception when a 16-bit timer is used in combination. Time stamp capture trigger can be selected (SOF or EOF in a CAN message frame can be detected.). - The time stamp function can be set for a transmit message. - A specific byte in a data field can be replaced by a capture time stamp[Note]
Diagnostic function	- 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
Forced release from bus-off state	- Default mode can be set while bus is off, so that bus can be forcibly released from the bus-off state.
Power save mode	- CAN Sleep mode (can be woken up by CAN bus) - CAN Stop mode (cannot be woken up by CAN bus)

Note: This function is valid only with a macro having an advanced time stamp function.

13.1.4 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 means of transmitting and receiving signals between the CAN module and the host CPU.

(2) MCM (Message Control Module)

This functional block controls access to the CAN protocol layer and to the CAN RAM within the CAN module.

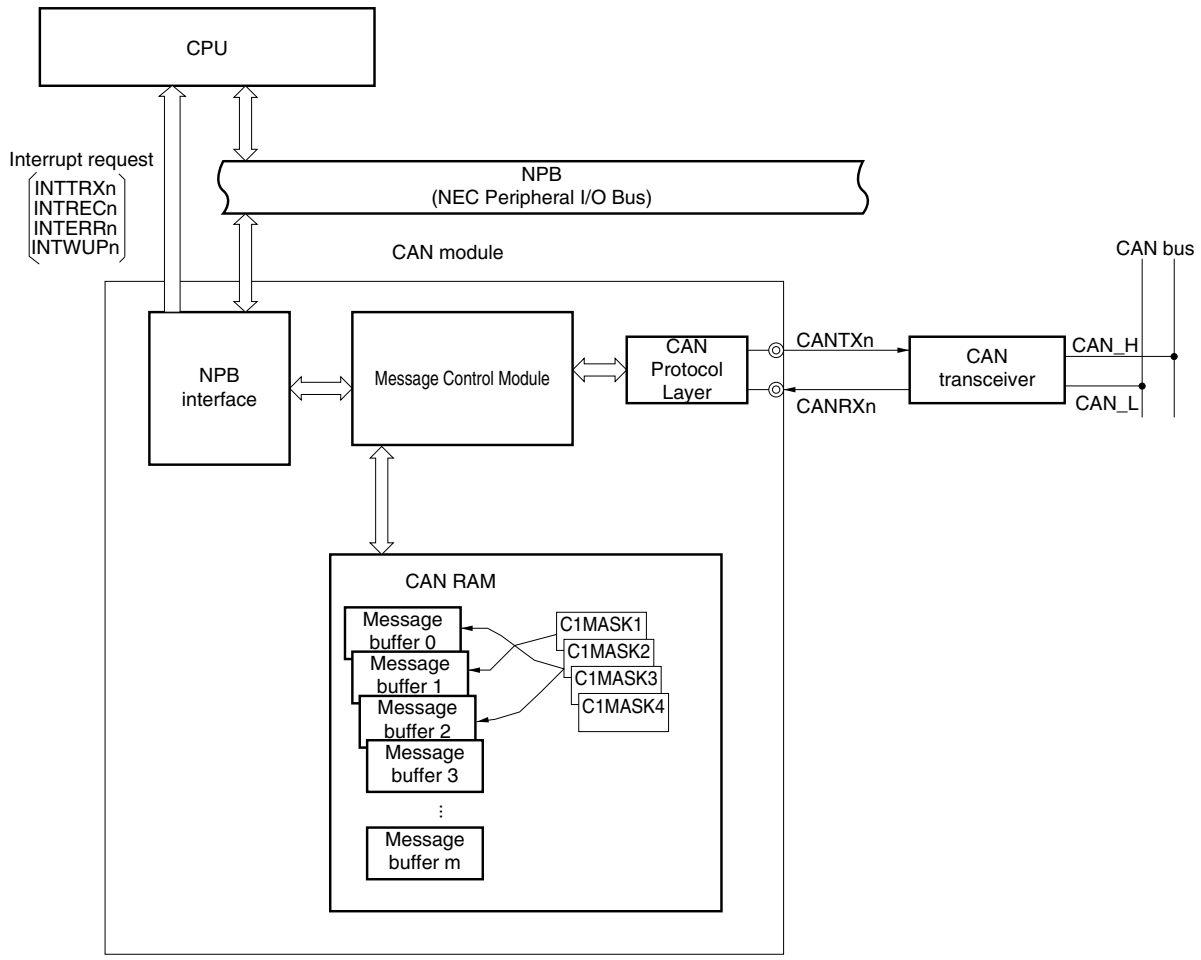
(3) CAN protocol layer

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 13-1: Block Diagram of CAN Module



Remark: m = 48 (m differs depending on the product.).

13.2 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 and medium access control. The composition of these layers is illustrated below.

Figure 13-2: Composition of Layers

Higher	Data link layer[Note]	• Logical link control (LLC)	• Acceptance filtering • Overload report • Recovery management
		• Medium access control (MAC)	• Data capsuled/not capsuled • Frame coding (stuffing/not stuffing) • Medium access management • Error detection • Error report • Acknowledgement • Seriated/not seriated
Lower	Physical layer		Prescription of signal level and bit description

Note: CAN controller specification

13.2.1 Frame format

(1) Standard format frame

- The standard format frame uses 11-bit identifiers, which means that it can handle up to 2048 messages.

(2) Extended format frame

- 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.

13.2.2 Frame types

The following four types of frames are used in the CAN protocol.

Table 13-3: 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.

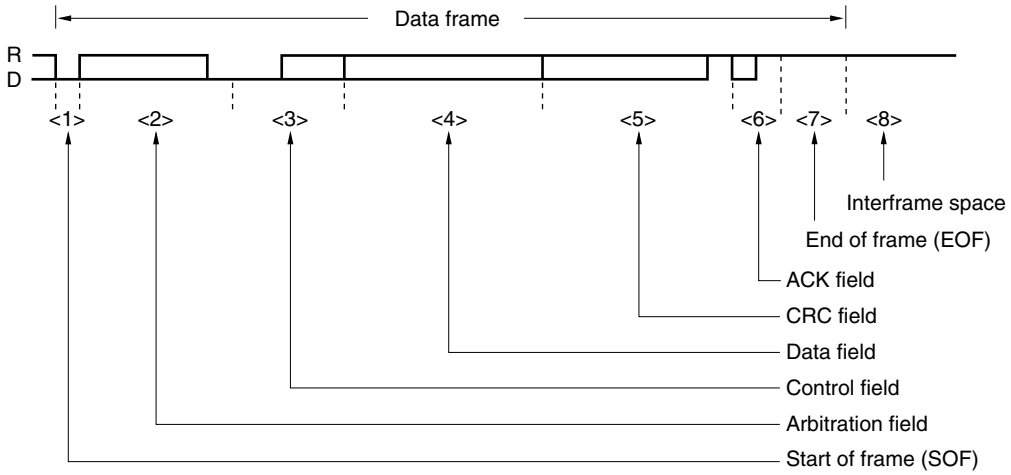
When a dominant level and a recessive level are transmitted simultaneously, the bus value becomes dominant level.

13.2.3 Data frame and remote frame

(1) Data frame

A data frame is composed of seven fields.

Figure 13-3: Data Frame

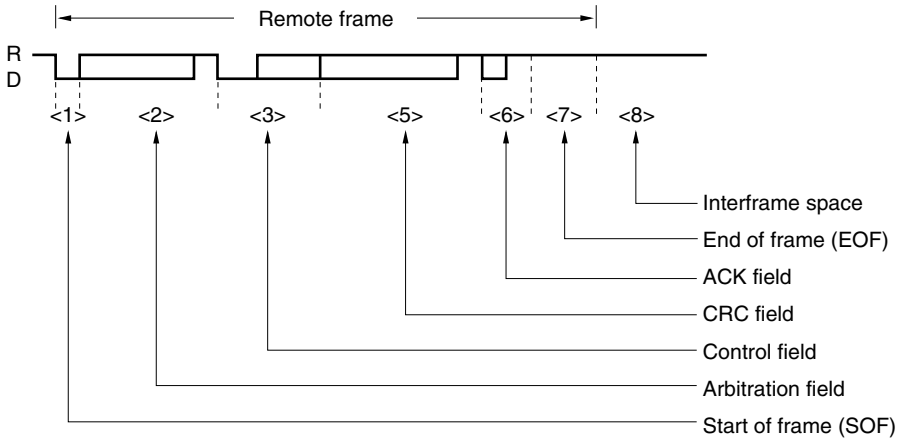


Remark: D: Dominant = 0
R: Recessive = 1

(2) Remote frame

A remote frame is composed of six fields.

Figure 13-4: Remote Frame



Remarks: 1. The data field is not transferred even if the control field's data length code is not "0000B".

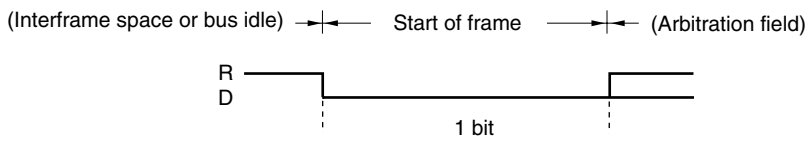
2. D: Dominant = 0
R: Recessive = 1

(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 13-5: Start of Frame (SOF)



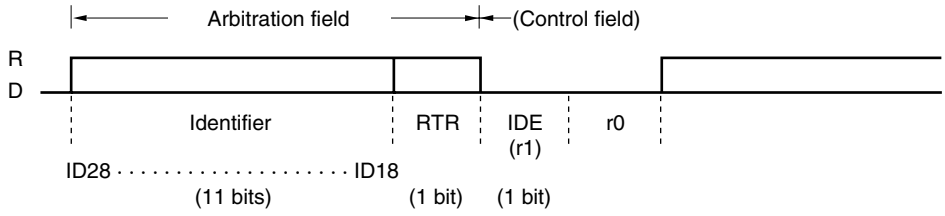
Remark: D: Dominant = 0
R: Recessive = 1

- If dominant level is detected in the bus idle state, a hard-synchronization is performed (the current TQ is assigned to be the SYNC segment).
- If dominant level is sampled at the sample point following such a hard-synchronization, the bit is assigned to be a SOF. If recessive level is detected, the protocol layer returns to the bus idle state and regards the preceding dominant pulse as a disturbance only. No error frame is generated in such case.

<2> Arbitration field

The arbitration field is used to set the priority, data frame/remote frame, and frame format.

Figure 13-6: Arbitration Field (in Standard Format Mode)

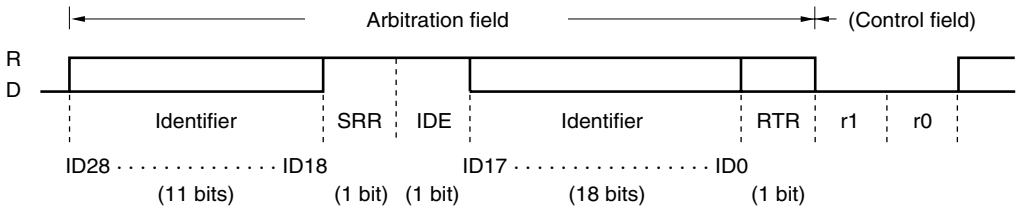


Cautions: 1. ID28 to ID18 are identifiers.

2. An identifier is transmitted MSB first.

Remark: D: Dominant = 0
R: Recessive = 1

Figure 13-7: Arbitration Field (in Extended Format Mode)



Cautions: 1. ID28 to ID18 are identifiers.

2. An identifier is transmitted MSB first.

Remark: D: Dominant = 0
R: Recessive = 1

Table 13-4: RTR Frame Settings

Frame Type	RTR Bit
Data frame	0 (D)
Remote frame	1 (R)

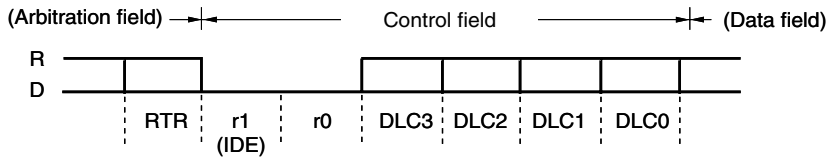
Table 13-5: Frame Format Setting (IDE Bit) and Number of Identifier (ID) Bits

Frame Format	SRR Bit	IDE Bit	Number. 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 13-8: 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 13-6: Data Length Setting

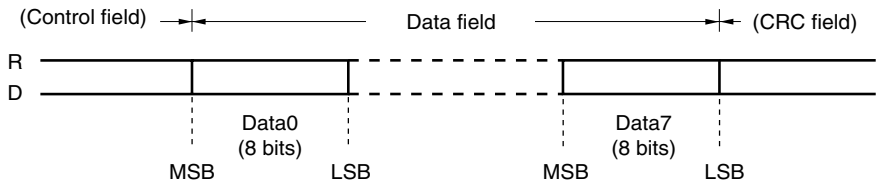
Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other than above				8 bytes regardless of the value of DLC3 to DLC0

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 13-9: Data Field

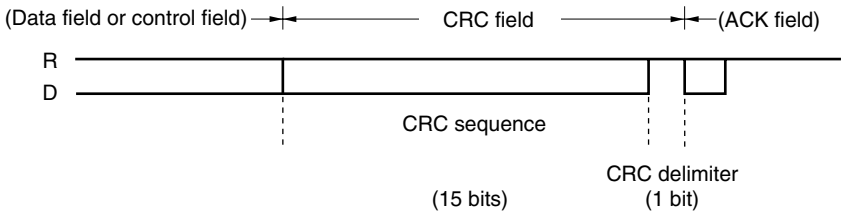


Remark: D: Dominant = 0
R: Recessive = 1

<5> CRC field

The CRC field is a 16-bit field that is used to check for errors in transmit data.

Figure 13-10: 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.

  $P(X) = X15 + X14 + X10 + X8 + X7 + X4 + X3 + 1$

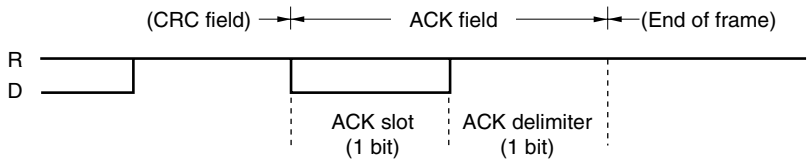
- Transmitting node: Transmits the CRC sequence calculated from the data (before bit stuffing) in the start of frame, arbitration field, control field, and data field.

- Receiving node: Compares the CRC sequence calculated using data bits that exclude the stuffing bits in the receive data 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 acknowledge normal reception.

Figure 13-11: ACK Field



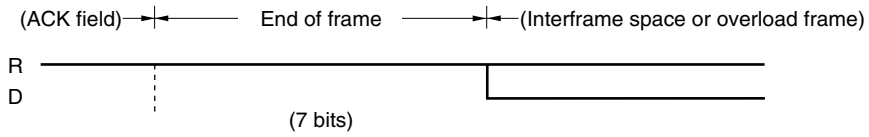
Remark: D: Dominant = 0
R: Recessive = 1

- 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 13-12: End of Frame (EOF)



Remark: D: Dominant = 0
R: Recessive = 1

<8> Interframe space

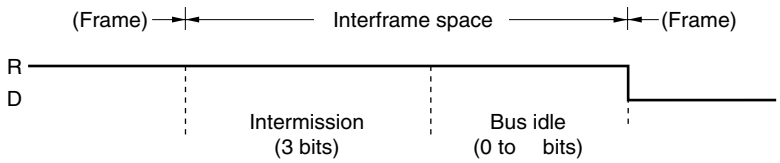
The interframe space is inserted after a data frame, remote frame, error frame, or overload frame to separate one frame from the next.

- The bus state differs depending on the error status.

(a) Error active node

The interframe space consists of a 3-bit intermission field and a bus idle field.

Figure 13-13: Interframe Space (Error Active Node)



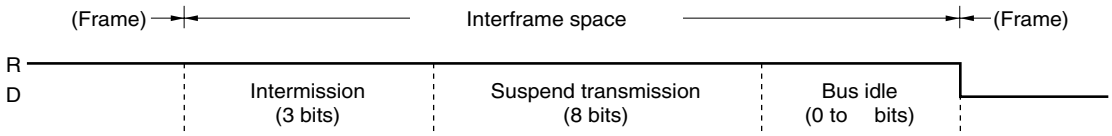
Remarks: 1. Bus idle: State in which the bus is not used by any node.

2. D: Dominant = 0
R: Recessive = 1

(b) Error passive node

The interframe space consists of an intermission field, a suspend transmission field, and a bus idle field.

Figure 13-14: 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-level bits transmitted from the node in the error passive status.

2. D: Dominant = 0
R: Recessive = 1

Usually, the intermission field is 3 bits. If the transmitting node detects a dominant level at the third bit of the intermission field, however, it executes transmission.

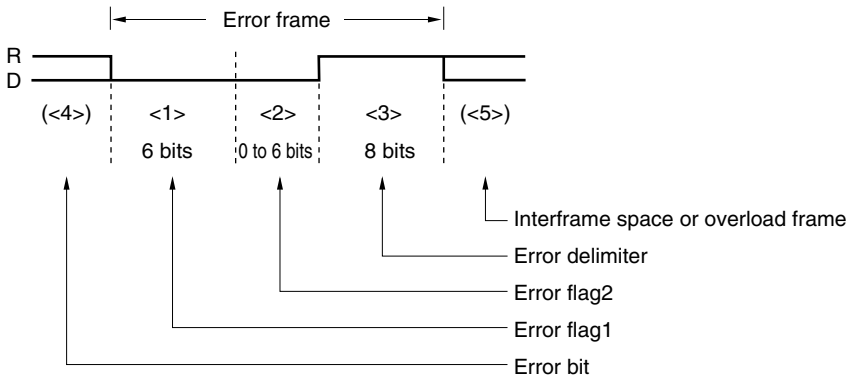
Table 13-7: Operation in Error Status

Error Status	Operation
Error active	A node in this status can transmit immediately after a 3-bit intermission.
Error passive	A node in this status can transmit 8 bits after the intermission.

13.2.4 Error frame

An error frame is output by a node that has detected an error.

Figure 13-15: Error Frame



Remark: D: Dominant = 0
R: Recessive = 1

Table 13-8: Definition Error Frame Fields

No.	Name	Bit Count	Definition
<1>	Error flag1	6	Error active node: Outputs 6 dominant-level bits consecutively. Error passive node: Outputs 6 recessive-level bits consecutively. If another node outputs a dominant level while one node is outputting a passive error flag, the passive error flag is not cleared until the same level is detected 6 bits in a row.
<2>	Error flag2	0 to 6	Nodes receiving error flag 1 detect bit stuff errors and issues this error flag.
<3>	Error delimiter	8	Outputs 8 recessive-level bits consecutively. If a dominant level is detected at the 8th bit, an overload frame is transmitted from the next bit.
<4>	Error bit	–	The bit at which the error was detected. The error flag is output from the bit next to the error bit. In the case of a CRC error, this bit is output following the ACK delimiter.
<5>	Interframe space/ overload frame	–	An interframe space or overload frame starts from here.

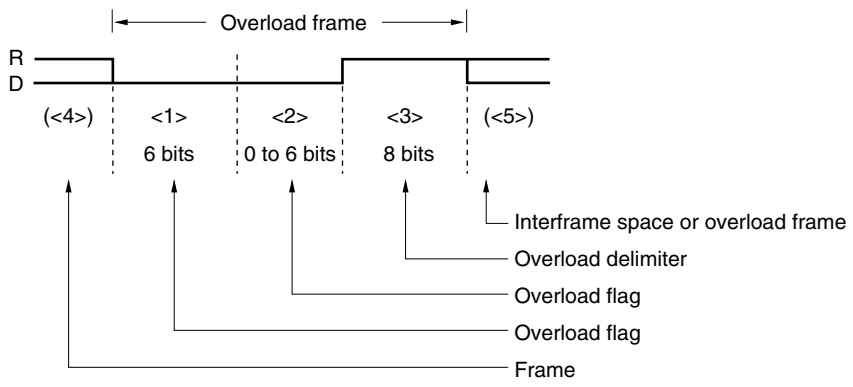
13.2.5 Overload frame

An overload frame is transmitted under the following conditions.

- When the receiving node has not completed the reception operation[Note]
- If a dominant level is detected at the first two bits during intermission
- If a dominant level is detected at the last bit (7th bit) of the end of frame or at the last bit (8th bit) of the error
- delimiter/overload delimiter

Note: The CAN is internally fast enough to process all received frames not generating overload frames.

Figure 13-16: Overload Frame



Remark: D: Dominant = 0
R: Recessive = 1

Table 13-9: Definition of Overload Frame Fields

No	Name	Bit Count	Definition
<1>	Overload flag	6	Outputs 6 dominant-level bits consecutively.
<2>	Overload flag from other node	0 to 6	The node that received an overload flag in the interframe space outputs an overload flag.
<3>	Overload delimiter	8	Outputs 8 recessive-level bits consecutively. If a dominant level is detected at the 8th bit, an overload frame is transmitted from the next bit.
<4>	Frame	–	Output following an end of frame, error delimiter, or overload delimiter.
<5>	Interframe space/ overload frame	–	An interframe space or overload frame starts from here.

13.3 Functions

13.3.1 Determining bus priority

(1) When a node starts transmission:

- During bus idle, the node that output data first transmits the data.

(2) When more than one node starts transmission:

- The node that outputs the dominant level for the longest consecutively from the first bit of the arbitration field
acquires the bus priority (if a dominant level and a recessive level are simultaneously transmitted, the dominant level is taken as the bus value).

- The transmitting node compares its output arbitration field and the data level on the bus.

Table 13-10: Determining Bus Priority

Level match	Continuous transmission
Level mismatch	Continuous transmission

(3) Priority of data frame and remote frame

- When a data frame and a remote frame are on the bus, the data frame has priority because its RTR bit, the last bit in the arbitration field, carries a dominant level.

Remark: If the extended-format data frame and the standard-format remote frame conflict on the bus (if ID28 to ID18 of both of them are the same), the standard-format remote frames takes priority.

13.3.2 Bit stuffing

Bit stuffing is used to establish synchronization by appending 1-bit inverted data if the same level continues for 5 bits, in order to prevent a burst error.

Table 13-11: Bit Stuffing

Transmission	During the transmission of a data frame or remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, 1 inverted-level bit of data is inserted before the following bit.
Reception	During the reception of a data frame or remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, reception is continued after deleting the next bit.

13.3.3 Multi masters

As the bus priority (a node acquiring transmit functions) is determined by the identifier, any node can be the bus master.

13.3.4 Multi cast

Although there is one transmitting node, two or more nodes can receive the same data at the same time because the same identifier can be set to two or more nodes.

13.3.5 CAN SLEEP mode/CAN STOP mode function

The CAN Sleep mode/CAN Stop mode function puts the CAN controller in waiting mode to achieve low power consumption.
The controller is woken up from the CAN Sleep mode by bus operation but it is not woken up from the CAN Stop mode by bus operation (the CAN Stop mode is controlled by CPU access).

13.3.6 Error control function

(1) Error types

Table 13-12: 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	Transmitting/ receiving node	Bit that outputting data on the bus at the start of frame to end of frame, error frame and overload frame.
Stuff error	Check the receive data at the stuff bit	6 consecutive bits of the same output level	Receiving node	Start of frame to CRC sequence
CRC error	Comparison of the CRC sequence generated from the receive data and the received CRC sequence	Mismatch of CRC	Receiving node	CRC field
Form error	Field/frame check of the fixed format	Detection of fixed format violation	Receiving node	- CRC delimiter - ACK field - End of frame - Error frame - Overload frame
ACK error	Check of the ACK slot by the transmitting node	Detection of reces-sive level in ACK slot	Transmitting node	ACK slot

(2) Output timing of error frame

Table 13-13: Output Timing of Error Frame

Type	Output Timing
Bit error, stuff error, form error, ACK error	Error frame output is started at the timing of the bit following the detected error.
CRC error	Error frame output is started at the timing of the bit following the ACK delimiter.

(3) Processing in case of error

The transmission node re-transmits the data frame or remote frame after the error frame (however, it does not re-transmit the frame in the single-shot mode.).

(4) Error state

(a) Types of error states

The following three types of error states are defined by the CAN specification.
- Error active
- Error passive
- Bus-off

These types of error states are classified by the values of the TEC7 to TEC0 bits (transmission error counter bits) and the REC6 to REC0 bits (reception error counter bits) of the CAN error counter register as shown in Table 13-14, “Types of Error States,” on page 513.
The present error state is indicated by the CAN module information register (CnINFO).
When each error counter value becomes equal to or greater than the error warning level (96), the TECS0 or RECS0 bit of the CnINFO register is set to 1. In this case, the bus state must be tested because it is considered that the bus has a serious fault. An error counter value of 128 or more indicates an error passive state and the TECS1 or RECS1 bit of the CnINFO register is set to 1.

- If the value of the transmission error counter is greater than or equal to 256 (actually, the transmission error counter does not indicate a value greater than or equal to 256), the bus-off state is reached and the BOFF bit of the CnINFO register is set to 1.

- If only one node is active on the bus at startup (i.e., a particular case such as when the bus is connected only to the local station), ACK is not returned even if data is transmitted. Consequently, re-transmission of the error frame and data is repeated. In the error passive state, however, the transmission error counter is not incremented and the bus-off state is not reached.

Table 13-14: Types of Error States

Type	Operation	Value of Error Counter	Indication of CnINFO Register	Operation specific to Given Error State
Error active	Transmission	0-95	TECS1, 0 = 00	• Outputs an active error flag (6 consecutive dominant-level bits) on detection of the error.
	Reception	0-95	RECS1, 0 = 00	
	Transmission	96-127	TECS1, 0 = 01	
	Reception	96-127	RECS1,0 = 01	
Error passive	Transmission	128-255	TECS1,0 = 11	• Outputs a passive error flag (6 consecutive recessive-level bits) on detection of the error. • Transmits 8 recessive-level bits, in between transmissions, following an intermission (suspend transmission)
	Reception	128 or more	RECS1,0 = 11	
Bus-off	Transmission	256 or more (not indicated) **Note**	BOFF = 1, TECS = 11	• Communication is not possible. Messages are not stored when receiving frames, however, the following operations of <1>, <2>, and <3> are done. <1> TSOUT toggles. <2> REC is incremented/decremented. <3> VALID bit is set. • If the initialization mode is set and then 11 recessive-level bits are generated 128 times in a row in an operation mode other than the initialization mode, the error counter is reset to 0 and the error active state can be restored.

Note: The value of the transmission error counter (TEC) is invalid when the BOFF bit is set to 1. If an error that increments the value of the transmission error counter by +8 while the counter value is in a range of 248 to 255, the counter is not incremented and the bus-off state is assumed.

(b) Error counter

The error counter counts up when an error has occurred, and counts down upon successful transmission and reception. The error counter is updated during the first bit of the error delimiter.

Table 13-15: Error Counter

State	Transmission Error Counter (TEC7 to TEC0)	Reception Error Counter (REC6 to REC0)
Receiving node detects an error (except bit error in the active error flag or overload flag).	No change	+1 (when REPS = 0)
Receiving node detects dominant level following error flag of error frame.	No change	+8 (when REPS = 0)
Transmitting node transmits an error flag. [As exceptions, the error counter does not change in the following cases.] <1> ACK error is detected in error passive state and dominant level is not detected while the passive error flag is being output. <2> A stuff error is detected in an arbitration field that transmitted a recessive level as a stuff bit, but a dominant level is detected.	+8	No change
Bit error detection while active error flag or overload flag is being output (error-active transmitting node)	+8	No change
Bit error detection while active error flag or overload flag is being output (error-active receiving node)	No change	+8(when REPS = 0)
When the node detects 14 consecutive dominant-level bits from the beginning of the active error flag or overload flag, and then subsequently detects 8 consecutive dominant-level bits. When the node detects 8 consecutive dominant levels after a passive error flag	+8 (during transmission)	+8 (during reception, when REPS = 0)
When the transmitting node has completed transmission without error (±0 if error counter = 0)	–1	No change
When the receiving node has completed reception without error	No change	–1 (1 ≤ REC6 to REC0 ≤ 127, when REPS = 0) ±0 (REC6 to REC0 = 0, when REPS = 0) Value of 119 to 127 is set (when REPS = 1)

(c) Occurrence of bit error in intermission

An overload frame is generated.

Caution: If an error occurs, it is controlled according to the contents of the transmission error counter and reception error counter before the error occurred. The value of the error counter is incremented after the error flag has been output.

(5) Recovery from bus-off state

When the CAN module is in the bus-off state, the transmission pins (CTXDn) cut off from the CAN bus always output the recessive level.
The CAN module recovers from the bus-off state in the following bus-off recovery sequence.

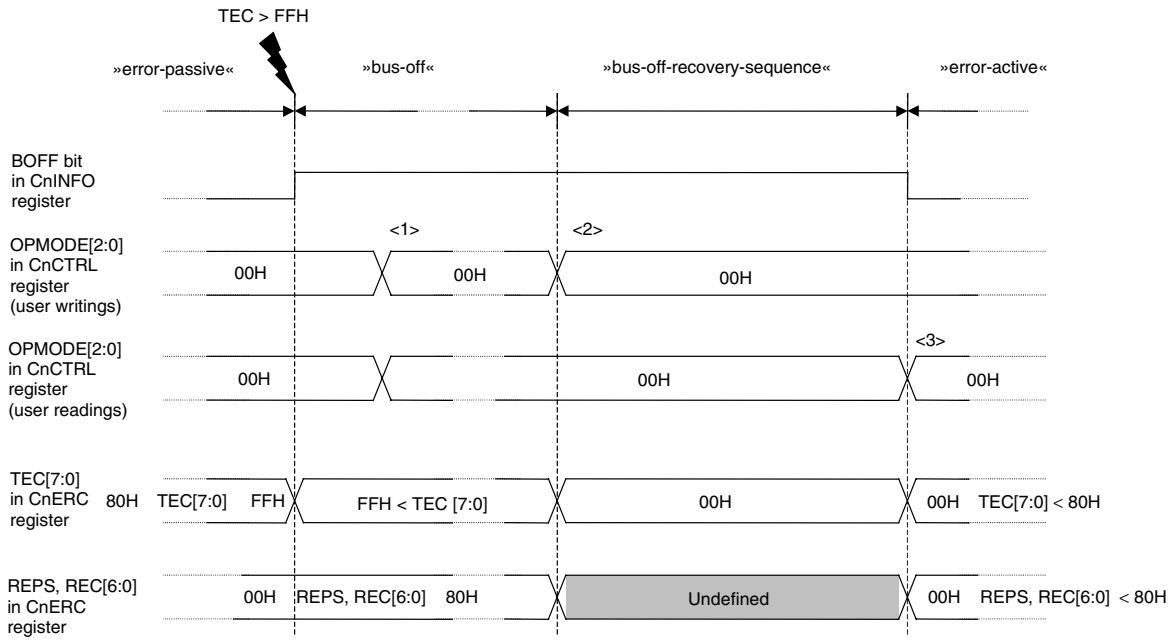
<1> A request to enter the CAN initialization mode
<2> A request to enter a CAN operation mode
(a) Recovery operation through normal recovery sequence
(b) Forced recovery operation that skips recovery sequence

(a) Recovery operation from bus-off state through normal recovery sequence

The CAN module first issues a request to enter the initialization mode (refer to timing <1> in Figure 13-17, "Recovery Operation from Bus-off State through Normal Recovery Sequence," on page 516). This request will be immediately acknowledged, and the OPMODE bits of the CnCTRL register are cleared to 000B. Processing such as analyzing the fault that has caused the bus-off state, re-defining the CAN module and message buffer using application software, or stopping the operation of the CAN module can be performed by clearing the GOM bit to 0.
Next, the user requests to change the mode from the initialization mode to an operation mode (refer to timing <2> in Figure 13-17, "Recovery Operation from Bus-off State through Normal Recovery Sequence," on page 516). This starts an operation to recover the CAN module from the bus-off state. The conditions under which the module can recover from the bus-off state are defined by the CAN protocol ISO 11898, and it is necessary to detect 11 consecutive recessive-level bits 128 times. At this time, the request to change the mode to an operation mode is held pending until the recovery conditions are satisfied. When the recovery conditions are satisfied (refer to timing <3> in Figure 13-17, "Recovery Operation from Bus-off State through Normal Recovery Sequence," on page 516), the CAN module can enter the operation mode it has requested. Until the CAN module enters this operation mode, it stays in the initialization mode. Whether the CAN module has completely entered the operation mode can be confirmed by reading the OPMODE bits of the CnCTRL register.
During the bus-off period and bus-off recovery sequence, the BOFF bit of the CnINFO register stays set (to 1). In the bus-off recovery sequence, the reception error counter (REC[6:0]) counts the number of times 11 consecutive recessive-level bits have been detected on the bus. Therefore, the recovery state can be checked by reading REC[6:0].

Caution: In the bus-off recovery sequence, REC[6:0] counts up (+1) each time 11 consecutive recessive-level bits have been detected. Even during the bus-off period, the CAN module can enter the CAN Sleep mode or CAN Stop mode. In this case, the bus-off recovery sequence starts not entering a CAN initialization mode at the same time when the CAN Sleep mode is released. Note that the bus-off recovery sequence will start when the CAN module will be woken up by the detection of the dominant edge other than clearing PSMODE by software.

Figure 13-17: Recovery Operation from Bus-off State through Normal Recovery Sequence



(b) Forced recovery operation that skips bus-off recovery sequence

The CAN module can be forcibly released from the bus-off state, regardless of the bus state, by skipping the bus-off recovery sequence. Here is the procedure.
First, the CAN module requests to enter the initialization mode. For the operation and points to be noted at this time, refer to **13.3.6 (5) "Recovery operation from bus-off state through normal recovery sequence" on page 515**.
Next, the module requests to enter an operation mode. At the same time, the CCERC bit of the CnCTRL register must be set to 1.
As a result, the bus-off recovery sequence defined by the CAN protocol ISO 11898 is skipped, and the module immediately enters the operation mode. In this case, the module is connected to the CAN bus after it has monitored 11 consecutive recessive-level bits. For details, refer to the processing in **Figure 13-78, "Setting CAN Sleep Mode/Stop Mode," on page 621**.

Caution: This function is not defined by the CAN protocol ISO 11898. When using this function, thoroughly evaluate its effect on the network system.

(6) Initializing CAN module error counter register (CnERC) in initialization mode

If it is necessary to initialize the CAN module error counter register (CnERC) and CAN module information register (CnINFO) for debugging or evaluating a program, they can be initialized to the default value by setting the CCERC bit of the CnCTRL register in the initialization mode. When initialization has been completed, the CCERC bit is automatically cleared to 0.

Cautions: 1. This function is enabled only in the initialization mode. Even if the CCERC bit is set to 1 in a CAN operation mode, the CnERC and CnINFO registers are not initialized.

2. The CCERC bit can be set at the same time as the request to enter a CAN operation mode.

13.3.7 Baud rate control function

(1) Prescaler

The CAN controller has a prescaler that divides the clock (f_{CAN}) supplied to CAN. This prescaler generates a CAN protocol layer basic clock (f_{TQ}) that is the CAN module system clock (f_{CANMOD}) divided by 1 to 256 (refer to **13.6.13 and Figure 13-38, "CAN Module Bit Rate Prescaler Register (CnBRP)," on page 550**).

(2) Data bit time (8-25 time quanta)

One data bit time is defined as shown in Figure 13-18.
The CAN controller sets time segment 1, time segment 2, and re-Synchronization Jump Width (SJW) as the parameter of data bit time, as shown in Figure 13-18. Time segment 1 is equivalent to the total of the propagation (prop) segment and phase segment 1 that are defined by the CAN protocol specification. Time segment 2 is equivalent to phase segment 2.

Figure 13-18: Segment Setting

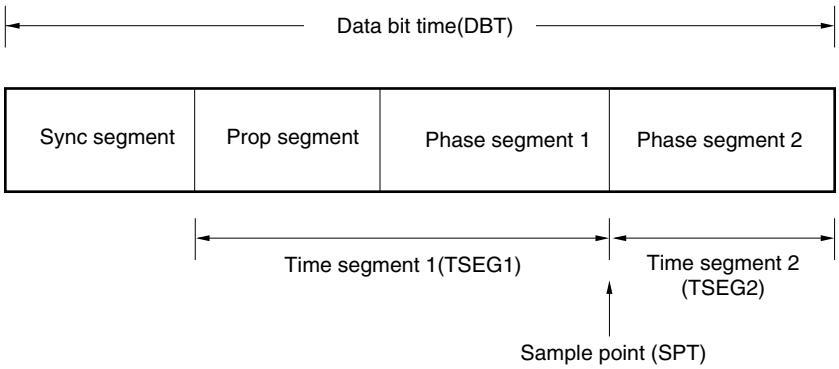


Table 13-16: Segment Setting

Segment Name	Settable Range	Notes on Setting to Confirm to CAN Specification
Time Segment 1 (TSEG1)	2TQ-16TQ	—
Time Segment 2 (TSEG2)	1TQ-8TQ	IPT[Note] of the CAN controller is 0TQ. To conform to the CAN protocol specification, therefore, a length equal to phase segment 1 must be set here. This means that the length of time segment 1 minus 1TQ is the settable upper limit of time segment 2.
Re-synchronization jump width (SJW)	1TQ-4TQ	The length of time segment 1 minus 1TQ or 4 TQ, whichever is smaller.

Note: IPT: Information Processing Time

Remark: Reference: The CAN standard ISO 11898 specification defines the segments constituting the data bit time as shown in Figure 13-19 on the next page.

Figure 13-19: Configuration of Data Bit Time Defined by CAN Specification

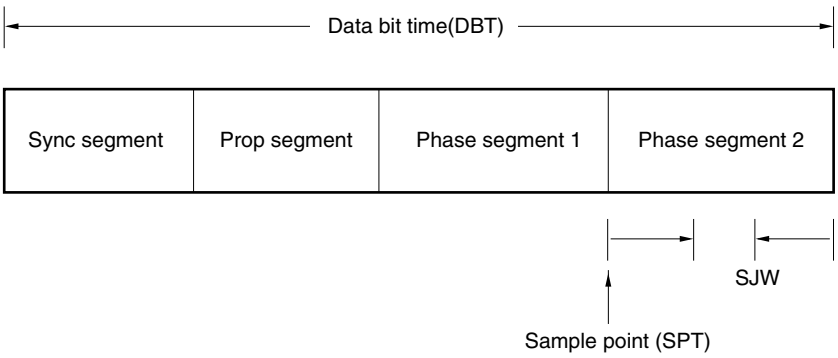


Table 13-17: Configuration of Data Bit Time Defined by CAN Specification

Segment Name	Segment Length	Description
Sync Segment (Synchronization Segment)	1	This segment starts at the edge where the level changes from recessive to dominant when hard-synchronization is established.
Prop Segment	Programmable to 1 to 8 or more	This segment absorbs the delay of the output buffer, CAN bus, and input buffer. The length of this segment is set so that ACK is returned before the start of phase segment 1. Time of prop segment ≥ (Delay of output buffer) + 2 × (Delay of CAN bus) + (Delay of input buffer) This segment compensates for an error of data bit time. The longer this segment, the wider the permissible range but the slower the communication speed.
Phase Segment 1	Programmable to 1 to 8	
Phase Segment 2	Phase Segment 1 or IPT**Note**, whichever greater	
SJW	Programmable from 1TQ to length of segment 1 or 4TQ, whichever is smaller	This width sets the upper limit of expansion or contraction of the phase segment during re-synchronization

Note: IPT: Information Processing Time

(3) Synchronizing data bit

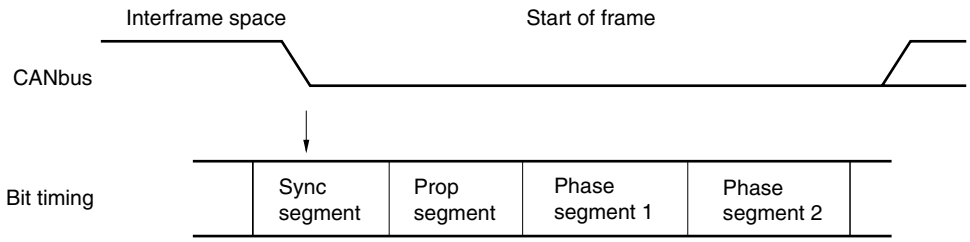
- The receiving node establishes synchronization by a level change on the bus because it does not have a sync signal.
- The transmitting node transmits data in synchronization with the bit timing of the transmitting node.

(a) Hard-synchronization

This synchronization is established when the receiving node detects the start of frame in the interframe space.

- When a falling edge is detected on the bus, that TQ means the sync segment and the next segment is the prop segment. In this case, synchronization is established regardless of SJW.

Figure 13-20: Hard-synchronization at Recognition of Dominant Level during Bus Idle



(b) Re-synchronization

Synchronization is established again if a level change is detected on the bus during reception (only if a recessive level was sampled previously).

- The phase error of the edge is given by the relative position of the detected edge and sync segment.
-

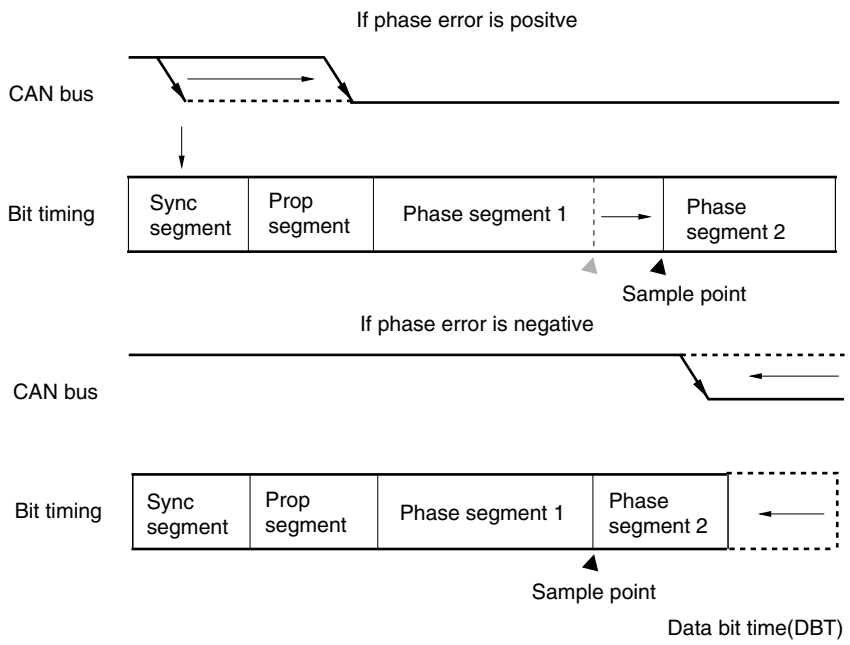
<Sign of phase error>

0: If the edge is within the sync segment
Positive: If the edge is before the sample point (phase error)
Negative: If the edge is after the sample point (phase error)

If phase error is positive: Phase segment 1 is longer by specified SJW.
If phase error is negative: Phase segment 2 is shorter by specified SJW.

- The sample point of the data of the receiving node moves relatively due to the “discrepancy” in baud rate between the transmitting node and receiving node.

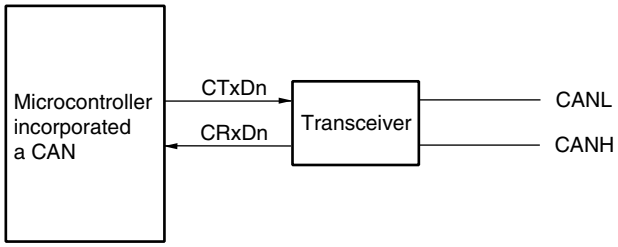
Figure 13-21: Re-synchronization



13.4 Connection With Target System

The microcontroller incorporated a CAN has to be connected to the CAN bus using an external transceiver.

Figure 13-22: Connection to CAN Bus



Remark: n: 0 - 4 = number of channel

13.5 Internal Registers of CAN Controller

13.5.1 CAN Controller configuration

Table 13-18: List of CAN Controller Registers (1/2)

Item	Register Name
CAN global registers	CAN global control register (CnGMCTRL)
	CAN global clock selection register (CnGMCS)
	CAN global automatic block transmission control register (CnGMABT)
	CAN global configuration register (CnGMCONF)
	CAN global automatic block transmission delay register (CnGMABTD)
CAN module registers	CAN module mask 1 register (CnMASK1L, CnMASK1H)
	CAN module mask 2 register (CnMASK2L, CnMASK2H)
	CAN module mask3 register (CnMASK3L, CnMASK3H)
	CAN module mask 4 registers (CnMASK4L, CnMASK4H)
	CAN module control register (CnCTRL)
	CAN module last error code register (CnLEC)
	CAN module information register (CnINFO)
	CAN module error counter register (CnERC)
	CAN module interrupt enable register (CnIE)
	CAN module interrupt status register (CnINTS)
	CAN module bit rate prescaler register (CnBRP)
	CAN module bit rate register (CnBTR)
	CAN module last in-pointer register (CnLIPT)
	CAN module receive history list register (CnRGPT)
	CAN module last out-pointer register (CnLOPT)
	CAN module transmit history list register (CnTGPT)
	CAN module time stamp register (CnTS)
Message buffer registers	CAN message data byte 01 register m (CnMDATA01m)
	CAN message data byte 0 register m (CnMDATA0m)
	CAN message data byte 1 register m (CnMDATA1m)
	CAN message data byte 23 register m (CnMDATA23m)
	CAN message data byte 2 register m (CnMDATA2m)
	CAN message data byte 3 Register m (CnMDATA3m)
	CAN message data byte 45 Register m (CnMDATA45m)
	CAN message data byte 4 Register m (CnMDATA4m)
	CAN message data byte 5 Register m (CnMDATA5m)
	CAN message data byte 67 Register m (CnMDATA67m)
	CAN message data byte 6 register m (CnMDATA6m)
	CAN message data byte 7 register m (CnMDATA7m)
	CAN message data length register m (CnMDLCm)

Table 13-18: List of CAN Controller Registers (2/2)

Item	Register Name
Message buffer registers	CAN message register m (CnMCONFm)
	CAN message ID register m (CnMIDLm, CnMIDHm)
	CAN message control register m (CnMCTRLm)

Remarks: 1. CAN global registers are identified by CnGM <register function>.
CAN module registers are identified by Cn <register function>.
Message buffer registers are identified by CnM <register function>.

2. n: 0-4 = Number of channel, m: 0-47: Number of buffer

13.5.2 Register access type

Table 13-19: CAN Global Register Access Types

Offset Address Note 2	Register Name	Symbol	R/W	Bit Manipulation Units			Default Value
				1	8	16	
00H	CAN global control register	CnGMCTRL	R/W	–	–	×	0000H
02H	CAN global clock selection register	CnGMCS	R/W	–	×	–	0FH
04H	CAN global configuration register	CnGMCONF	R	–	×	×	19H Note 1
06H	CAN global automatic block transmission control register	CnGMABT	R/W	–	–	×	0000H
08H	CAN global automatic block transmission delay register	CnGMABTD	R/W	–	×	–	00H

Notes: 1. This value differs for DAFCAN module (n=4), please refer to 14.11.39 ”Message Configuration Register (MCONFm)” on page 744

2. Offset of Global Register Area is 0x000.
The actual register address is calculated as follows:

Register Address = Peripheral Programmable Area Address (BPC)
+ Channel Offset
+ Global Register Area Offset
+ Offset Address as listed in table above

Remark: n: 0 - 4 = Number of channel

Table 13-20: CAN Module Register Access Types

Offset Address **Note**	Register Name	Symbol	R/W	Bit Manipulation Units			Default Value
				1	8	16	
00H	CAN module mask 1 register	CnMASK1L	R/W	–	–	×	undefined
02H		CnMASK1H					
04H	CAN module mask 2 register	CnMASK2L	R/W	–	–	×	undefined
06H		CnMASK2H					
08H	CAN module mask 3 register	CnMASK3L	R/W	–	–	×	undefined
0AH		CnMASK3H					
0CH	CAN module mask 4 register	CnMASK4L	R/W	–	–	×	undefined
0EH		CnMASK4H					
10H	CAN module control register	CnCTRL	R/W	–	–	×	0000H
12H	CAN module last error code register	CnLEC	R/W	–	×	–	00H
13H	CAN module information register	CnINFO	R	–	×	–	00H
14H	CAN module error counter register	CnERC	R	–	–	×	0000H
16H	CAN module interrupt enable register	CnIE	R/W	–	–	×	0000H
18H	CAN module interrupt status register	CnINTS	R/W	–	–	×	0000H
1AH	CAN module bit rate prescaler register	CnBRP	R/W	–	×	–	FFH
1CH	CAN module bit rate register	CnBTR	R/W	–	–	×	370FH
1EH	CAN module last in-pointer register	CnLIPT	R	–	×	–	undefined
20H	CAN module receive history list register	CnRGPT	R/W	–	–	×	xx02H
22H	CAN module last out-pointer register	CnLOPT	R	–	×	–	undefined
24H	CAN module transmit history list register	CnTGPT	R/W	–	–	×	xx02H
26H	CAN module time stamp register	CnTS	R/W	–	–	×	0000H

Remark: n: 0-4 = Number of channel

Note: Offset of CAN Module Register Area is 0x040.
The actual register address is calculated as follows:

Register Address = Peripheral Programmable Area Address (BPC)
+ Channel Offset
+ CAN Module Register Area Offset
+ Offset Address as listed in table above

Table 13-21: Message Buffer Access Types

Offset Address Note	Register Name	Symbol	R/W	Bit Manipulation Units			Default value
				1	8	16	
00H	CAN message data byte 01 register m	CnMDATA01m	R/W	–	–	×	undefined
00H	CAN message data byte 0 register m	CnMDATA0m	R/W	–	×	–	undefined
01H	CAN message data byte 1 register m	CnMDATA1m	R/W	–	×	–	undefined
02H	CAN message data byte 23 register m	CnMDATA23m	R/W	–	–	×	undefined
02H	CAN message data byte 2 register m	CnMDATA2m	R/W	–	×	–	undefined
03H	CAN message data byte 3 register m	CnMDATA3m	R/W	–	×	–	undefined
04H	CAN message data byte 45 register m	CnMDATA45m	R/W	–	–	×	undefined
04H	CAN message data byte 4 register m	CnMDATA4m	R/W	–	×	–	undefined
05H	CAN message data byte 5 register m	CnMDATA5m	R/W	–	×	–	undefined
06H	CAN message data byte 67 register m	CnMDATA67m	R/W	–	–	×	undefined
06H	CAN message data byte 6 register m	CnMDATA6m	R/W	–	×	–	undefined
07H	CAN message data byte 7 register m	CnMDATA7m	R/W	–	×	–	undefined
08H	CAN message data length register m	CnMDLCm	R/W	–	×	–	0000xxxxB
09H	CAN message configuration register m	CnMCONFm	R/W	–	×	–	undefined
0AH	CAN message ID register m	CnMIDLm	R/W	–	–	×	undefined
0CH		CnMIDHm					
0EH	CAN message control register m	CnMCTRLm	R/W	–	–	×	00x00000 000xx000B

Remark: n: 0-4 = Number of channel
m: 0-47: Number of buffer

Note: Offset of Message Buffer Area is 0x100.
The actual register address is calculated as follows:

Register Address = Peripheral Programmable Area Address (BPC)
+ Channel Offset
+ Message Buffer Area Offset
+ Offset Address as listed in table above

13.5.3 Register bit configuration

Table 13-22: Bit Configuration of CAN Global Registers

Offset 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
00H	CnGMCTRL (W)	0	0	0	0	0	0	0	Clear GOM
01H		0	0	0	0	0	0	Set EFSD	Set GOM
00H	CnGMCTRL (R)	0	0	0	0	0	0	EFSD	GOM
01H		MBON	0	0	0	0	0	0	0
02H	CnGMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0
04H	CnGMCONF	Undefined (reserved for future use)		GCONF5	GCONF4	GCONF3	GCONF2	GCONF1	GCONF0
06H	CnGMABT (W)	0	0	0	0	0	0	0	Clear ABTTRG
07H		0	0	0	0	0	0	Set ABTCLR	Set ABTTRG
06H	CnGMABT (R)	0	0	0	0	0	0	ABTCLR	ABTTRG
07H		0	0	0	0	0	0	0	0
08H	CnGMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0

Remark: n: 0-4 = Number of channel

Table 13-23: Bit Configuration of CAN Module Registers (1/2)

Offset 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
00H	CnMASK1L	CM1ID [7:0]							
01H		CM1ID [15:8]							
02H	CnMASK1H	CM1ID [23:16]							
03H		0	0	0	CM1ID [28:24]				
04H	CnMASK2L	CM2ID [7:0]							
05H		CM2ID [15:8]							
06H	CnMASK2H	CM2ID [23:16]							
07H		0	0	0	CM2ID [28:24]				
08H	CnMASK3L	CM3ID [7:0]							
09H		CM3ID [15:8]							
0AH	CnMASK3H	CM3ID [23:16]							
0BH		0	0	0	CM3ID [28:24]				
0CH	CnMASK4L	CM4ID [7:0]							
0DH		CM4ID [15:8]							
0EH	CnMASK4H	CM4ID [23:16]							
0FH		0	0	0	CM4ID [23:16]				
10H	CnCTRL (W)	0	Clear AL	Clear VALID	Clear PSMODE1	Clear PSMODE0	Clear OPMODE2	Clear OPMODE1	Clear OPMODE0
11H		Set CCERC	Set AL	0	Set PSMODE1	Set PSMODE0	Set OPMODE2	Set OPMODE1	Set OPMODE0
10H	CnCTRL (R)	CCERC	AL	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0
11H		0	0	0	0	0	0	RSTAT	TSTAT
12H	CnLEC (W)	0	0	0	0	0	0	0	0
12H	CnLEC (R)	0	0	0	0	0	LEC2	LEC1	LEC0
13H	CnINFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0
14H	CnERC	TEC[7:0]							
15H		REPS	REC[6:0]						
16H	CnIE (W)	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0
17H		0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0
16H	CnIE (R)	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0
17H		0	0	0	0	0	0	0	0
18H	CnINTS (W)	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0
19H		0	0	0	0	0	0	0	0
18H	CnINTS (R)	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0
19H		0	0	0	0	0	0	0	0
1AH	CnBRP	TQPRS[7:0]							
1BH	–	Access prohibited (reserved for future use)							
1CH	CnBTR	0	0	0	0	TSEG1[3:0]			
1DH		0	0	SJW[1:0]		0	TSEG2[2:0]		

Table 13-23: Bit Configuration of CAN Module Registers (2/2)

Offset 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
1EH	CnLIPT	LIPT[7:0]							
1FH	–	Access prohibited (reserved for future use)							
20H	CnRGPT (W)	0	0	0	0	0	0	0	Clear ROVF
21H		0	0	0	0	0	0	0	0
20H	CnRGPT (R)	0	0	0	0	0	0	RHPM	ROVF
21H		RGPT[7:0]							
22H	CnLOPT	LOPT[7:0]							
23H	–	Access prohibited (reserved for future use)							
24H	CnTGPT (W)	0	0	0	0	0	0	0	Clear TOVF
25H		0	0	0	0	0	0	0	0
24H	CnTGPT (R)	0	0	0	0	0	0	THPM	TOVF
25H		TGPT[7:0]							
26H	CnTS (W)	0	0	0	0	0	Clear TSLOCK	Clear TSSEL	Clear TSEN
27H		0	0	0	0	0	Set TSLOCK	Set TSSEL	Set TSEN
26H	CnTS (R)	0	0	0	0	0	TSLOCK	TSSEL	TSEN
27H		0	0	0	0	0	0	0	0
28H-3FH	–	Access prohibited (reserved for future use)							

Remark: n: 0-4 = Number of channel

Table 13-24: Bit Configuration of Message Buffer Registers

Offset 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
00H	CnMDATA01m	Message data (byte0)							
01H		Message data (byte1)							
00H	CnMDATA0m	Message data (byte 0)							
01H	CnMDATA1m	Message data (byte 1)							
02H	CnMDATA23m	Message data (byte 2)							
03H		Message data (byte 3)							
02H	CnMDATA2m	Message data (byte 2)							
03H	CnMDATA3m	Message data (byte 3)							
04H	CnMDATA45m	Message data (byte 4)							
05H		Message data (byte 5)							
04H	CnMDATA4m	Message data (byte 4)							
05H	CnMDATA5m	Message data (byte 5)							
06H	CnMDATA67m	Message data (byte 6)							
07H		Message data (byte 7)							
06H	CnMDATA6m	Message data (byte 6)							
07H	CnMDATA7m	Message data (byte 7)							
08H	CnMDLCm	0				MDLC3	MDLC2	MDLC1	MDLC0
09H	CnMCONFm	OWS	RTR	MT2	MT1	MT0	0	0	MA0
0AH	CnMIDLm	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
0BH		ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
0CH	CnMIDHm	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
0DH		IDE	0	0	ID28	ID27	ID26	ID25	ID24
0EH	CnMCTRLm (W)	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY
0FH		0	0	0	0	Set IE	0	Set TRQ	Set RDY
0EH	CnMCTRLm (R)	0	0	0	MOW	IE	DN	TRQ	RDY
0FH		0	0	MUC	0	0	0	0	0
10-1FH	–	Access prohibited (reserved for future)							

Remark: n: 0-4 = Number of channel
m: 0-47: Number of buffer

13.6 Control Registers

13.6.1 CAN Global Control Register (CnGMCTRL)

The CnGMCTRL register is used to control the operation of the CAN module.

Figure 13-23: CAN Global Control Register (CnGMCTRL) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMCTRL	MBON	0	0	0	0	0	0	0	01H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	EFSD	GOM	00H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMCTRL	0	0	0	0	0	0	Set EFSD	Set GOM	01H	
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	Clear GOM	00H	

(a) Read

MBON	Bit Enabling Access to Message Buffer Register, Transmit/Receive History List Registers
0	Write access and read access to the message buffer register and the transmit/receive history list registers is disabled.
1	Write access and read access to the message buffer register and the transmit/receive history list registers is enabled.

Cautions: 1. While the MBON bit is cleared (to 0), software access to the message buffers (CnMDATA0m, CnMDATA1m, CnMDATA01m, CnMDATA2m, CnMDATA3m, CnMDATA23m, CnMDATA4m, CnMDATA5m, CnMDATA45m, CnMDATA6m, CnMDATA7m, CnMDATA67m, CnMDLCm, CnMCONFm, CnMIDLm, CnMIDHm, and CnMCTRLm), or registers related to transmit history or receive history (CnLOPT, CnTGPT, CnLIPT, and CnRGPT) is disabled.

2. This bit is read-only. Even if 1 is written to MBON while it is 0, the value of MBON does not change, and access to the message buffer registers, or registers related to transmit history or receive history remains disabled.

Remark: MBON bit is cleared (to 0) when the CAN module enters CAN Sleep mode/CAN Stop mode or GOM bit is cleared (to 0).
MBON bit is set (to 1) when the CAN Sleep mode/the CAN Stop mode is released or GOM bit is set (to 1).

Figure 13-23: CAN Global Control Register (CnGMCTRL) Format(2/2)

EFSD	Bit Enabling Forced Shut Down
0	Forced shut down by GOM = 0 disabled.
1	Forced shut down by GOM = 0 enabled.

Caution: To request forced shut down, the GOM bit must be cleared to 0 immediately after the EFSD bit has been set to 1. If access to another register (including reading the CnGMCTRL register) is executed without clearing the GOM bit immediately after the EFSD bit has been set to 1, the EFSD bit is forcibly cleared to 0, and the forced shut down request is invalid.

GOM	Global Operation Mode Bit
0	CAN module is disabled from operating.
1	CAN module is enabled to operate.

Caution: The GOM bit can be cleared only in the initialization mode or immediately after EFSD bit is set (to 1).

(b) Write

Set EFSD	EFSD Bit Setting
0	No change in ESFD bit.
1	EFSD bit set to 1.

Set GOM	Clear GOM	GOM Bit Setting
0	1	GOM bit cleared to 0
1	0	GOM bit set to 1
Other than above		No change in GOM bit.

13.6.2 CAN Global Clock Selection Register (CnGMCS)

The CnGMCS register is used to select the CAN module system clock.

Figure 13-24: CAN Global Clock Selection Register (CnGMCS) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnGMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0	002H	0FH
R/W										

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	$f_{CAN}/16$ (default value)

Remark: f_{CAN} = Clock supply to CAN

13.6.3 CAN Global Automatic Block Transmission Control Register (CnGMABT)

The CnGMABT register is used to control the automatic block transmission (ABT) operation.

Figure 13-25: CAN global automatic block transmission control register (CnGMABT) (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMABT	0	0	0	0	0	0	0	0	007H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	ABTCLR	ABTTRG	006H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMABT	0	0	0	0	0	0	Set ABTCLR	Set ABTTRG	007H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	Clear ABTTRG	006H	00H

Caution: Before changing the normal operation mode with ABT to the initialization mode, be sure to set the CnGMABT register to the default value (00H).

(a) Read

ABTCLR	Automatic Block Transmission Engine Clear Status Bit
0	Clearing the automatic transmission engine is completed
1	The automatic transmission engine is being cleared.

Remarks: 1. Set the ABTCLR bit to 1 while the ABTTRG bit is cleared (0).
The operation is not guaranteed if the ABTCLR bit is set to 1 while the ABTTRG bit is set to 1.

2. When the automatic block transmission engine is cleared by setting the ABTCLR bit to 1, the ABTCLR bit is automatically cleared to 0 as soon as the requested clearing processing is complete.

ABTTRG	Automatic Block Transmission Status Bit
0	Automatic block transmission is stopped.
1	Automatic block transmission is under execution.

Caution: Do not set the ABTTRG bit (ABTTRG = 1) in the initialization mode. If the ABTTRG bit is set in the initialization mode, the operation is not guaranteed after the CAN module has entered the normal operation mode with ABT.

Figure 13-25: CAN global automatic block transmission control register (CnGMABT) (2/2)

(b) Write

Set ABTCLR	Automatic Block Transmission Engine Clear Request Bit
0	The automatic block transmission engine is in idle state or under operation.
1	Request to clear the automatic block transmission engine. After the automatic block transmission engine has been cleared, automatic block transmission is started from message buffer 0 by setting the ABTTRG bit to 1.

Set ABTTRG	Clear ABTTRG	Automatic Block Transmission Start Bit
0	1	Request to stop automatic block transmission.
1	0	Request to start automatic block transmission.
Other than above		No change in ABTTRG bit.

13.6.4 CAN Global Configuration Register (CnGMCONF)

The CnGMCONF register provides the configuration information of the CAN module.

Figure 13-26: CAN Global Configuration Register (CnGMCONF) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMCONF (Read only)	Note							GCONF8	05H	0x00
	7	6	5	4	3	2	1	0		
	Note		GCONF5	GCONF4	GCONF3	GCONF2	GCONF1	GCONF0	0x04	0x19

Note: Undefined (reserved for future use)

GCONF2	GCONF1	GCONF0	Number of CAN interface channels[Note]
0	0	0	reserved
0	0	1	1 CAN interface channels
0	1	0	reserved
0	1	1	reserved
1	0	0	reserved
1	0	1	reserved
1	1	0	reserved
1	1	1	reserved

Note: This is not the number of the channels of the device.

GCONF5	GCONF4	GCONF3	Number of message buffers
0	0	0	Reserved
0	0	1	16 message buffers
0	1	0	32 message buffers
0	1	1	48 message buffers
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

GCONF8	Mirror Function for DIAG macro
0	no Mirror function is implemented
1	Mirror function is implemented

13.6.5 CAN Global Automatic Block Transmission Delay Register (CnGMABTD)

The CnGMABTD register is used to set the interval at which the data of the message buffer assigned to ABT is to be transmitted in the normal operation mode with ABT.

Figure 13-27: CAN Global Automatic Block Transmission Delay Register (CnGMABTD) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnGMABT	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0	008H	00H
R/W										

ABTD3	ABTD2	ABTD1	ABTD0	Data frame interval during automatic block transmission (unit: Data bit time (DBT))
0	0	0	0	0 DBT (default value)
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

Cautions: 1. Do not change the contents of the CnGMABTD register while the ABTTRG bit is set to 1.

2. The timing at which the ABT message is actually transmitted onto the CAN bus differs depending on the status of transmission from the other station or how a request to transmit a message other than an ABT message (message buffers 8 to m_{MAX}–1[Note]) is made.

Note: $m_{MAX} = 48$

13.6.6 CAN Module Mask Control Register (CnMASKaL, CnMASKaH) (a = 1, 2, 3, or 4)

The CnMASKaL and CnMASKaH registers are used to extend the number of receivable messages into the same message buffer by masking part of the ID comparison of a message and invalidating the ID of the masked part.

Figure 13-28: CANn Module Mask1 Registers (CnMASK1L, CnMASK1H) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK1L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	01H	undefined
R/W										
	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	00H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK1H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	03H	undefined
R/W										
	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	02H	undefined

Figure 13-29: CANn Module Mask2 Registers (CnMASK2L, CnMASK2H) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK2L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	05H	undefined
R/W										
	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	04H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK2H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	07H	undefined
R/W										
	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	06H	undefined

Figure 13-30: CANn Module Mask3 Registers (CnMASK3L, CnMASK3H) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK3L R/W	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	09H	undefined
	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	08H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK3H R/W	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	0BH	undefined
	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	0AH	undefined

Figure 13-31: CANn Module Mask4 Registers (CnMASK4L, CnMASK4H) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK4L R/W	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	0DH	undefined
	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	0CH	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMASK4H R/W	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	0FH	undefined
	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	0EH	undefined

CMID28 to CMID0	Sets Mask Pattern of ID Bit.
0	The ID bits of the message buffer set by the CMID28 to CMID0 bits are compared with the ID bits of the received message frame
1	The ID bits of the message buffer set by the CMID28 to CMID0 bits are not compared with the ID bits of the received message frame (they are masked).

Remark: Masking is always defined by an ID length of 29 bits. If a mask is assigned to a message with a standard ID, CMID17 to CMID0 are ignored. Therefore, only CMID28 to CMID18 of the received ID are masked. The same mask can be used for both the standard and extended IDs.

13.6.7 CAN Module Control Register (CnCTRL)

The CnCTRL register is used to control the operation mode of the CAN module.

Figure 13-32: CAN Module Control Register (CnCTRL) Format (1/4)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnCTRL	0	0	0	0	0	0	RSTAT	TSTAT	11H	00H
R/W										
	7	6	5	4	3	2	1	0		
	CCERC	AL	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0	10H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnCTRL	Set CCERC	Set AL	0	Set PSMODE1	Set PSMODE0	Set OPMODE2	Set OPMODE1	Set OPMODE0	11H	
R/W										
	7	6	5	4	3	2	1	0		
	0	Clear AL	Clear VALID	Clear PSMODE1	Clear PSMODE0	Clear OPMODE2	Clear OPMODE1	Clear OPMODE0	10H	

(a) Read

RSTAT	Reception Status Bit
0	Reception is stopped
1	Reception is in progress

Remark: The RSTAT bit is set to 1 under the following conditions (timing):
- The SOF bit of a receive frame is detected
- On occurrence of arbitration loss during a transmit frame
- The RSTAT bit is cleared to 0 under the following conditions (timing)
- When a recessive level is detected at the second bit of the interframe space
- On transition to the initialization mode at the first bit of the interframe space

Figure 13-32: CANn Module Control Register (CnCTRL) Format (2/4)

TSTAT	Transmission Status Bit
0	Transmission is stopped.
1	Transmission is in progress

Remark: The TSTAT bit is set to 1 under the following conditions (timing):
- The SOF bit of a transmit frame is detected
- The first bit of an error flag is detected during a transmit frame
- The TSTAT bit is cleared to 0 under the following conditions (timing)
- During transition to bus-off state
- On occurrence of arbitration loss in transmit frame
- On detection of recessive level at the second bit of the interframe space
- On transition to the initialization mode at the first bit of the interframe space

CCERC	Error Counter Clear Bit
0	The CnERC and CnINFO registers are not cleared in the initialization mode.
1	The CnERC and CnINFO registers are cleared in the initialization mode

Remarks:
1. The CCERC bit is used to clear the CnERC and CnINFO registers for re-initialization or forced recovery from the bus-off state. This bit can be set to 1 only in the initialization mode.
2. When the CnERC and CnINFO registers have been cleared, the CCERC bit is also cleared to 0 automatically.
3. The CCERC bit can be set to 1 at the same time as a request to change the initialization mode to an operation mode is made.4. The CCERC bit is read-only in the CAN Sleep mode or CAN Stop mode.
4. The receive data may be broken in case of setting the CCERC bit to (1) immediately after entering the INIT mode.

AL	Bit to Set Operation in Case of Arbitration Loss
0	Re-transmission is not executed in case of an arbitration loss in the single-shot mode
1	Re-transmission is executed in case of an arbitration loss in the single-shot mode

Remark: The AL bit is valid only in the single-shot mode

VALID	Valid Receive Message Frame Detection Bit
0	A valid message frame has not been received since the VALID bit was last cleared to 0
1	A valid message frame has been received since the VALID bit was last cleared to 0

Remarks:
1. Detection of a valid receive message frame is not dependent upon storage in the receive message buffer (data frame) or transmit message buffer (remote frame).
2. Clear the VALID bit (0) before changing the initialization mode to an operation mode.
3. If only two CAN nodes are connected to the CAN bus with one transmitting a message frame in the normal mode and the other in the reception mode, the VALID bit is not set to 1 before the transmitting node enters the error passive state.
4. In order to clear the VALID bit, set the Clear VALID bit to 1 first and confirm that the VALID bit is cleared. If it is not cleared, perform clearing processing again.

Figure 13-32: CANn Module Control Register (CnCTRL) Format (3/4)

PSMODE1	PSMODE0	Power Save Mode
0	0	No power save mode is selected
0	1	CAN Sleep mode
1	0	Setting prohibited
1	1	CAN Stop mode

Caution: Transition to and from the CAN Stop mode must be made via CAN Sleep mode. A request for direct transition to and from the CAN Stop mode is ignored.

OPMODE2	OPMODE1	OPMODE0	Operation Mode
0	0	0	No operation mode is selected (CAN module is in the initialization mode)
0	0	1	Normal operating mode
0	1	0	Normal operation mode with automatic block transmission function (normal operation mode with ABT)
0	1	1	Receive-only mode
1	0	0	Single-shot mode
1	0	1	Self-test mode
Other than above			Setting prohibited.

Remark: The OPMODE[2:0] bits are read-only in the CAN Sleep mode or CAN Stop mode

(b) Write

Set CCERC	Setting of CCERC Bit
1	CCERC bit is set to 1
Other than above	CCERC bit is not changed

Set AL	Clear AL	Setting of AL Bit
0	1	AL bit is cleared to 0
1	0	AL bit is set to 1
Other than above		AL bit is not changed

Clear VALID	Setting of VALID Bit
0	VALID bit is not changed
1	VALID bit is cleared to 0

Figure 13-32: CANn Module Control Register (CnCTRL) Format (4/4)

Set PSMODE0	Clear PSMODE0	Setting of PSMODE0 Bit
0	1	PSMODE0 bit is cleared to 0
1	0	PSMODE bit is set to 1
Other than above		PSMODE0 bit is not changed

Set PSMODE1	Clear PSMODE1	Setting of PSMODE1 Bit
0	1	PSMODE1 bit is cleared to 0
1	0	PSMODE1 bit is set to 1
Other than above		PSMODE1 bit is not changed

Set OPMODE0	Clear OPMODE0	Setting of OPMODE0 Bit
0	1	OPMODE0 bit is cleared to 0
1	0	OPMODE0 bit is set to 1
Other than above		OPMODE0 bit is not changed

Set OPMODE1	Clear OPMODE1	Setting of OPMODE1 Bit
0	1	OPMODE1 bit is cleared to 0
1	0	OPMODE1 bit is set to 1
Other than above		OPMODE1 bit is not changed

Set OPMODE2	Clear OPMODE2	Setting of OPMODE2 Bit
0	1	OPMODE2 bit is cleared to 0
1	0	OPMODE2 bit is set to 1
Other than above		OPMODE2 bit is not changed

13.6.8 CAN Module Last Error Code Register (CnLEC)

The CnLEC register provides the error information of the CAN protocol.

Figure 13-33: CAN Module Last Error Code Register (CnLEC) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnLEC	0	0	0	0	0	LEC2	LEC1	LEC0	12H	00H
R/W										

Remarks: 1. The contents of the CnLEC register are not cleared when the CAN module changes from an operation mode to the initialization mode.

2. If an attempt is made to write a value other than 00H to the CnLEC register by software, the access is ignored.

LEC2	LEC1	LEC0	Last CAN Protocol Error Information
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 transmit a recessive-level bit as part of a transmit message (except the arbitration field), but the value on the CAN bus is a dominant-level bit.)
1	0	1	Bit error (The CAN module tried to transmit a dominant-level bit as part of a transmit message, ACK bit, error frame, or overload frame, but the value on the CAN bus is a recessive-level bit.)
1	1	0	CRC error
1	1	1	Undefined

13.6.9 CAN Module Information Register (CnINFO)

The CnINFO register indicates the status of the CAN module.

Figure 13-34: CAN Module Information Register (CnINFO) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnINFO R/O	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0	13H	00H

BOFF	Bus-off State Bit
0	Not bus-off state (transmit error counter ≤ 255) (The value of the transmit counter is less than 256)
1	Bus-off state (transmit error counter > 255) (The value of the transmit counter is 256 or more)

TECS1	TECS0	Transmission Error Counter Status Bit
0	0	The value of the transmission error counter is less than that of the warning level (<96)
0	1	The value of the transmission error counter is in the range of the warning level (96 to 127)
1	0	Undefined
1	1	The value of the transmission error counter is in the range of the error passive or bus-off state (≥ 128

RECS1	RECS0	Reception Error Counter Status Bit
0	0	The value of the reception error counter is less than that of the warning level (<96)
0	1	The value of the reception error counter is in the range of the warning level (96 to 127)
1	0	Undefined
1	1	The value of the reception error counter is in the error passive range (≥ 128)

13.6.10 CAN Module Error Counter Register (CnERC)

The CnERC register indicates the count value of the transmission/reception error counter.

Figure 13-35: CAN Module Error Counter Register (CnERC) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnERC	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0	15H	00H
R/O										
	7	6	5	4	3	2	1	0		
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	14H	00H

REPS	Reception Error Passive Status Bit
0	The reception error counter is not in the error passive range (< 128)
1	The reception error counter is in the error passive range (≥ 128)

REC6-REC0	Reception Error Counter Bit
0-127	Number of reception errors. These bits reflect the status of the reception error counter. The number of errors is defined by the CAN protocol.

Remark: REC6 to REC0 of the reception error counter are invalid in the reception error passive state (RECS [1:0] = 11B)

TEC7-TEC0	Transmission Error Counter Bit
0-255	Number of transmission errors. These bits reflect the status of the transmission error counter. The number of errors is defined by the CAN protocol

Remark: TEC7 to TEC0 of the transmission error counter are invalid in the bus-off state (BOFF = 1)

13.6.11 CAN Module Interrupt Enable Register (CnIE)

The CnIE register is used to enable or disable the interrupts of the CAN module.

Figure 13-36: CAN Module Interrupt Enable Register (CnIE) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnIE	0	0	0	0	0	0	0	0	17H	00H
	7	6	5	4	3	2	1	0		
	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0	16H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnIE	0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0	17H	
	7	6	5	4	3	2	1	0		
	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0	16H	

(a) Read

CIE5 to CIE0	CAN Module Interrupt Enable Bit
0	Output of the interrupt corresponding to interrupt status register CINTSx is disabled
1	Output of the interrupt corresponding to interrupt status register CINTSx is enabled.

Figure 13-36: CANn Module Interrupt Enable Register (CnIE) Format (2/2)

(b) Write

Set CIE5	Clear CIE5	Setting of CIE5 Bit
0	1	CIE5 bit is cleared to 0
1	0	CIE5 bit is set to 1
Other than above		CIE5 bit is not changed

Set CIE4	Clear CIE4	Setting of CIE4 Bit
0	1	CIE4 bit is cleared to 0
1	0	CIE4 bit is set to 1
Other than above		CIE4 bit is not changed

Set CIE3	Clear CIE3	Setting of CIE3 Bit
0	1	CIE3 bit is cleared to 0
1	0	CIE3 bit is set to 1
Other than above		CIE3 bit is not changed

Set CIE2	Clear CIE2	Setting of CIE2 Bit
0	1	CIE2 bit is cleared to 0
1	0	CIE2 bit is set to 1
Other than above		CIE2 bit is not changed

Set CIE1	Clear CIE1	Setting of CIE1 Bit
0	1	CIE1 bit is cleared to 0
1	0	CIE1 bit is set to 1
Other than above		CIE1 bit is not changed

Set CIE0	Clear CIE0	Setting of CIE0 Bit
0	1	CIE0 bit is cleared to 0
1	0	CIE0 bit is set to 1
Other than above		CIE0 bit is not changed

13.6.12 CAN Module Interrupt Status Register (CnINTS)

The CnINTS register indicates the interrupt status of the CAN module.

Figure 13-37: CAN Module Interrupt Status Register (CnINTS) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnINTS	0	0	0	0	0	0	0	0	19H	00H
	7	6	5	4	3	2	1	0		
	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0	18H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnINTS R	0	0	0	0	0	0	0	0	19H	
	7	6	5	4	3	2	1	0		
	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0	18H	

(a) Read

CINTS5 - CINTS0	CAN interrupt Status Bit
0	No related interrupt source event is pending
1	A related interrupt source event is pending

Interrupt Status Bit	Related Interrupt Source Event
CINTS5	Wakeup interrupt from CAN Sleep mode[Note]
CINTS4	Arbitration loss interrupt
CINTS3	CAN protocol error interrupt
CINTS2	CAN error status interrupt
CINTS1	Interrupt on completion of reception of valid message frame to message buffer m
CINTS0	Interrupt on normal completion of transmission of message frame from message buffer m

Note: The CINTS5 bit is set only when the CAN module is woken up from the CAN Sleep mode by a CAN bus operation. The CINTS5 bit is not set when the CAN Sleep mode has been released by software.

Figure 13-37: CAN Module Interrupt Status Register (CnINTS) Format (2/2)

(b) Write

Clear CINTS5-CINTS0	Setting of CINTS5 to CINTS0 Bits
0	CINTS5 to CINTS0 bits are not changed
1	CINTS5 to CINTS0 bits are cleared to 0

13.6.13 CAN Module Bit Rate Prescaler Register (CnBRP)

The CnBRP register is used to select the CAN protocol layer basic clock (f_{TQ}). The communication baud rate is set to the CnBTR register.

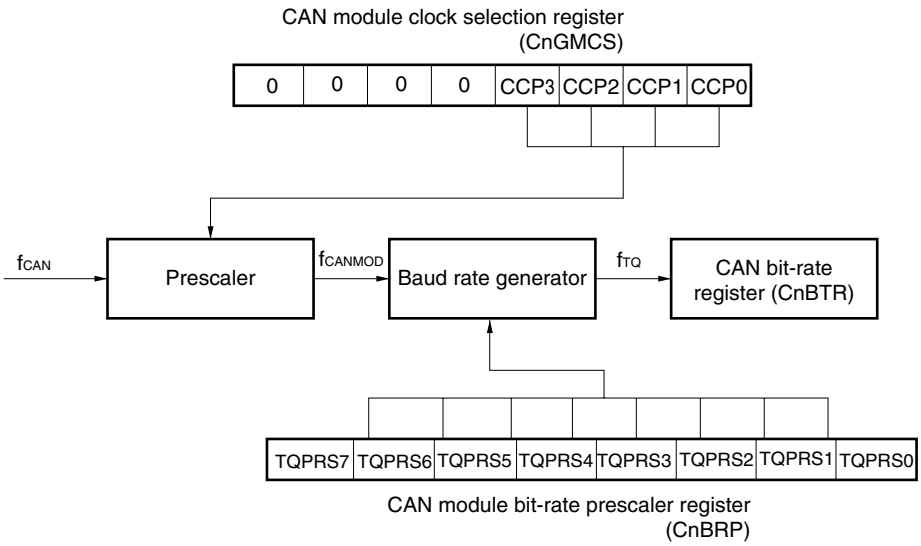
Figure 13-38: CAN Module Bit Rate Prescaler Register (CnBRP)

(a) CAN Module Bit Rate Prescaler Register (CnBRP) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnBRP	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0	1AH	FFH
R/W										

TQPRS7 - TQPRS0	CAN Protocol Layer Basic System Clock (f_{TQ})
0	$f_{CANMOD}/1$
1	$f_{CANMOD}/2$
n	$f_{CANMOD}/(n+1)$
.....	.....
255	$f_{CANMOD}/256$ (default value)

(b) CAN Module Bit Rate Prescaler Register (CnBRP) Clock



Remark: f_{CAN} : Clock supplied to CAN
f_{CANMOD}: CAN module system clock
f_{TQ} : CAN protocol layer basic system clock

Caution: The CnBRP register can be write-accessed only in the initialization mode.

13.6.14 CAN Module Bit Rate Register (CnBTR)

The CnBTR register is used to control the data bit time of the communication baud rate.

Figure 13-39: CAN Module Bit Rate Register (CnBTR) (1/2)

(a) CAN Module Bit Rate Register (CnBTR) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnBTR	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20	1DH	37H
R/W										
	7	6	5	4	3	2	1	0		
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10	1CH	0FH

(b) CAN Module Bit Rate Register (CnBTR) Data Bit Time

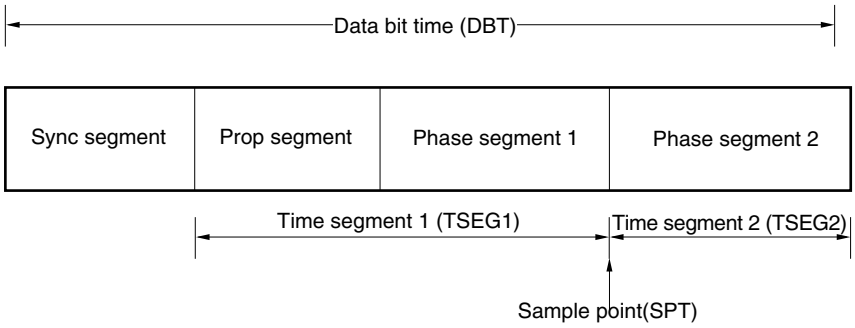


Figure 13-39: CAN Module Bit Rate Register (CnBTR) (2/2)

(c) CAN Module Bit Rate Register (CnBTR) Format

SJW1	SJW0	Length of Synchronization Jump Width
0	0	1TQ
0	1	2TQ
1	0	3TQ
1	1	4TQ (default value)

TSEG22	TSEG21	TSEG20	Length of Time Segment 2
0	0	0	1TQ
0	0	1	2TQ
0	1	0	3TQ
0	1	1	4TQ
1	0	0	5TQ
1	0	1	6TQ
1	1	0	7TQ
1	1	1	8TQ (default value)

TSEG13	TSEG12	TSEG11	TSEG10	Length of Time Segment 1
0	0	0	0	Setting prohibited.
0	0	0	1	2TQ**Note**
0	0	1	0	3TQ**Note**
0	0	1	1	4TQ
0	1	0	0	5TQ
0	1	0	1	6TQ
0	1	1	0	7TQ
0	1	1	1	8TQ
1	0	0	0	9TQ
1	0	0	1	10TQ
1	0	1	0	11TQ
1	0	1	1	12TQ
1	1	0	0	13TQ
1	1	0	1	14TQ
1	1	1	0	15TQ
1	1	1	1	16TQ (default value)

Note: This setting must not be made when the CnBRP register = 00H.

Remark: TQ = $1/f_{TQ}$ (f_{TQ}: CAN protocol layer basic system clock)

13.6.15 CAN Module Last In-Pointer Register (CnLIPT)

The CnLIPT register indicates the number of the message buffer in which a data frame or a remote frame was last stored.

Figure 13-40: CAN Module Last In-Pointer Register (CnLIPT) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnLIPT	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0	1EH	undefined
R/W										

LIPT7 - LIPT0	Last In-Pointer Register (CnLIPT)
0.....m (m_{MAX} –1)**Note**	When the CnLIPT register is read, the contents of the element indexed by the last in-pointer (LIPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame was last stored

Note: m_{MAX} = 48 (maximum number of message buffer)

Remark: The read value of the CnLIPT register is undefined if a data frame or a remote frame has never been stored in the message buffer. If the RHPM bit of the CnRGPT register is set to 1 after the CAN module has changed from the initialization mode to an operation mode, therefore, the read value of the CnLIPT register is undefined.

13.6.16 CAN Module Receive History List Register (CnRGPT)

The CnRGPT register is used to read the receive history list.

Figure 13-41: CAN Module Receive History List Register (CnRGPT) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnRGPT	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0	21H	undefined
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	RHPM	ROVF	20H	02H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnRGPT	0	0	0	0	0	0	0	0	21H	
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	Clear ROVF	20H	

(a) Read

RGPT7 - RGPT0	Receive History List Get Pointer
0.....m (m_{MAX} -1)**Note 1**	When the CnRGPT register is read, the contents of the element indexed by the receive history list get pointer (RGPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame has been stored

RHPM**Note 2**	Receive History List Pointer Match
0	The receive history list has at least one message buffer number that has not been read
1	The receive history list has no message buffer numbers that has not been read

Notes: 1. m_{MAX} = 48

2. The read value of RGPT0 to 7 is invalid when RHPM = 1.

Figure 13-41: CAN Module Receive History List Register (CnRGPT) Format (2/2)

(a) Read

ROVF	Receive History List Overflow Bit
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffer in which a new data frame or remote frame has been received and stored are recorded to the receive history list (the receive history list has a vacant element)
1	At least 23 entries have been stored since the host processor has serviced the RHL last time (i.e. read CnRGPT). Thus the sequence of receptions can not be recovered completely now. The first 22 entries are sequentially stored while the last entry can have been overwritten by newly received messages multiple times because all buffer numbers are stored at position LIPT-1 when ROVF bit is set[Note]

Note: The RHL will be updated, but the LIPT pointer will not be incremented. Always the position the LIPT pointer -1 is pointing to is overwritten.

(b) Write

Clear ROVF	Setting of ROVF Bit
0	ROVF bit is not changed
1	ROVF bit is cleared to 0

13.6.17 CAN Module Last Out-Pointer Register (CnLOPT)

The CnLOPT register indicates the number of the message buffer to which a data frame or a remote frame was transmitted last.

Figure 13-42: CAN Module Last Out-Pointer Register (CnLOPT) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnLOPT R/O	LOPT7	LOPT6	LOPT5	LOPT4	LOPT3	LOPT2	LOPT1	LOPT0	22H	undefined

LOPT7 - LOPT0	Last Out-Pointer of Transmit History List (LOPT)
0.....m (m_{MAX} -1)**Note**	When the CnLOPT register is read, the contents of the element indexed by the last out-pointer (LOPT) of the receive history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last

Note: m_{MAX} = 48

Remark: The value read from the CnLOPT register is undefined if a data frame or remote frame has never been transmitted from a message buffer. If the THPM bit is set to 1 after the CAN module has changed from the initialization mode to an operation mode, therefore, the read value of the CnLOPT register is undefined.

13.6.18 CAN Module Transmit History List Register (CnTGPT)

The CnTGPT register is used to read the transmit history list.

Figure 13-43: CAN Module Transmit History List Register (CnTGPT) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnTGPT	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0	25H	undefined
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	THPM	TOVF	24H	02H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnTGPT	0	0	0	0	0	0	0	0	25H	
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	Clear TOVF	24H	

(a) Read

TGPT7 - TGPT0	Transmit History List Read Pointer
0.....m (m_{MAX} -1)**Note 1**	When the CnTGPT register is read, the contents of the element indexed by the read pointer (TGPT) of the transmit history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last

THPM**Note 2**	Transmit History Pointer Match
0	The transmit history list has at least one message buffer number that has not been read
1	The transmit history list has no message buffer number that has not been read

Notes: 1. m_{MAX} = 48

2. The read value of TGPT0 to TGPT7 is invalid when THPM = 1.

Remark: Transmission from message buffer 0 to 7 is not recorded to the transmit history list in the normal operation mode with ABT.

Figure 13-43: CAN Module Transmit History List Register (CnTGPT) Format (2/2)

(a) Read

TOVF	Transmit History List Overflow Bit
0	All the message buffer numbers that have not been read are preserved.All the numbers of the message buffers to which a new data frame or remote frame has been transmitted are recorded to the transmit history list (the transmit history list has a vacant element)
1	At least 7 entries have been stored since the host processor has serviced the THL last time (i.e. read CnTGPT). Thus the sequence of transmissions can not be recovered completely now. The first 6 entries are sequentially stored while the last entry can have been overwritten by newly transmitted messages multiple times because all buffer numbers are stored at position LOPT-1 when TOVF bit is set. **Note**

Note: The THL will be updated, but the LOPT pointer will not be incremented. Always the position the LOPT pointer -1 is pointing to is overwritten.

(b) Write

Clear TOVF	Setting of TOVF Bit
0	TOVF bit is not changed
1	TOVF bit cleared to 0

13.6.19 CAN Module Time Stamp Register (CnTS)

The CnTS register is used to control the time stamp function.

Figure 13-44: CAN Module Time Stamp Register (CnTS) Format (1/2)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnTS	0	0	0	0	0	0	0	0	27H	00H
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	TSLOCK	TSSEL	TSEN	26H	00H

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnTS	0	0	0	0	0	Set TSLOCK	Set TSSEL	Set TSEN	27H	
	7	6	5	4	3	2	1	0		
	0	0	0	0	0	Clear TSLOCK	Clear TSSEL	Clear TSEN	26H	

Remarks: 1. The displayed CnTS register provides only the necessary bits for the Basic Time Stamp function.
The Advanced Time Stamp function requires a modified hardware.

2. The lock function of the time stamp function must not be used when the CAN module is in the normal operation mode with ABT

Figure 13-44: CAN Module Time Stamp Register (CnTS) Format (2/2)

(a) Read

TSLOCK	Time Stamp Lock Function Enable Bit
0	Time stamp lock function stopped. The TSOUT signal is toggled each time the selected time stamp capture event occurs.
1	Time stamp lock function enabled. The TSOUT signal is toggled each time the selected time stamp capture event occurs. However, the TSOUT output signal is locked when a data frame has been correctly received to message buffer 0[Note]

Note: The TSEN bit is automatically cleared to 0

TSSEL	Time Stamp Capture Event Selection Bit
0	The time capture event is SOF
1	The time stamp capture event is the last bit of EOF

TSEN	TSOUT Operation Setting Bit
0	Disable TSOUT toggle operation
1	Enable TSOUT toggle operation

(b) Write

Set TSLOCK	Clear TSLOCK	Setting of TSLOCK Bit
0	1	TSLOCK bit is cleared to 0
1	0	TSLOCK bit is set to 1
Other than above		TSLOCK bit is not changed

Set TSSEL	Clear TSSEL	Setting of TSSEL Bit
0	1	TSSEL bit is cleared to 0
1	0	TSSEL bit is set to 1
Other than above		TSSEL bit is not changed

Set TSEN	Clear TSEN	Setting of TSEN Bit
0	1	TSEN bit is cleared to 0
1	0	TSEN bit is set to 1
Other than above		TSEN bit is not changed

13.6.20 CAN Message Data Byte Register (CnMDATAxm) (x = 0 to 7), (CnMDATAzm) (z = 01, 23, 45, 67)

The CnMDATAxm, CnMDATAzm registers are used to store the data of a transmit/receive message. The CnMDATAzm registers can access the CnMDATAxm registers in 16-bit units.

Figure 13-45: CAN Message Data Byte Register (CnMDATA x m) (x = 0 to 7) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnMDATA0m	MDATA07	MDATA06	MDATA05	MDATA04	MDATA03	MDATA02	MDATA01	MDATA00	00H	undefined
CnMDATA1m	MDATA17	MDATA16	MDATA15	MDATA14	MDATA13	MDATA12	MDATA11	MDATA10	01H	undefined
CnMDATA2m	MDATA27	MDATA26	MDATA25	MDATA24	MDATA23	MDATA22	MDATA21	MDATA20	02H	undefined
CnMDATA3m	MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30	03H	undefined
CnMDATA4m	MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40	04H	undefined
CnMDATA5m	MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA51	MDATA50	05H	undefined
CnMDATA6m	MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60	06H	undefined
CnMDATA7m	MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70	07H	undefined

Figure 13-45: CAN Message Data Byte Register (CnMDATAzm) (z = 01, 23, 45, 67) Format (2/2)

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMDATA01m	MDATA0115	MDATA0114	MDATA0113	MDATA0112	MDATA0111	MDATA0110	MDATA019	MDATA018	01H	undefined
	7	6	5	4	3	2	1	0		
	MDATA017	MDATA016	MDATA015	MDATA014	MDATA013	MDATA012	MDATA011	MDATA010	00H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMDATA23m	MDATA2315	MDATA2314	MDATA2313	MDATA2312	MDATA2311	MDATA2310	MDATA239	MDATA238	03H	undefined
	7	6	5	4	3	2	1	0		
	MDATA237	MDATA236	MDATA235	MDATA234	MDATA233	MDATA232	MDATA231	MDATA230	02H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMDATA45m	MDATA4515	MDATA4514	MDATA4513	MDATA4512	MDATA4511	MDATA4510	MDATA459	MDATA458	05H	undefined
	7	6	5	4	3	2	1	0		
	MDATA457	MDATA456	MDATA455	MDATA454	MDATA453	MDATA452	MDATA451	MDATA450	04H	undefined

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMDATA67m	MDATA6715	MDATA6714	MDATA6713	MDATA6712	MDATA6711	MDATA6710	MDATA679	MDATA678	07H	undefined
	7	6	5	4	3	2	1	0		
	MDATA677	MDATA676	MDATA675	MDATA674	MDATA673	MDATA672	MDATA671	MDATA670	06H	undefined

13.6.21 CAN Message Data Length Register m (CnMDLCm)

The CnMDLCm register is used to set the number of bytes of the data field of a message buffer.

Figure 13-46: CAN Message Data Length Register m (CnMDLCm) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnMDLCm	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0	08H	0000xxxxB
R/W										

MDLC3	MDLC2	MDLC1	MDLC0	Data Length Of Transmit/Receive Message
0	0	0	0	0 byte
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	Setting prohibited (If these bits are set during transmission, 8-byte data is transmitted regardless of the set DLC value when a data frame is transmitted. However, the DLC actually transmitted to the CAN bus is the DLC value set to this register.)**Note**
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Note: The data and DLC value actually transmitted to CAN bus are as follows:

Type of Transmit Frame	Length of Transmit Data	DLC Transmitted
Data frame	Number of bytes specified by DLC (however, 8 bytes if DLC ≥ 8)	MDLC[3:0]
Remote frame	0 bytes	

Cautions: 1. Be sure to set bits 7 to 4 0000B.

2. Receive data is stored in as many CnMDATAx as the number of bytes (however, the upper limit is 8) corresponding to DLC. CnMDATAx in which no data is stored is undefined.

13.6.22 CAN Message Configuration Register (CnMCONFm)

The CnMCONFm register is used to specify the type of the message buffer and to set a mask.

Figure 13-47: CAN Message Configuration Register (CnMCONFm) Format

Symbol	7	6	5	4	3	2	1	0	Address	Default value
CnMCONFm	OWS	RTR	MT2	MT1	MT0	0	0	MA0	09H	undefined
R/W										

OWS	Overwrite Select Bit
0	The message buffer that has already received a data frame[Note] is not overwritten by a newly received data frame. The newly received data frame is discarded
1	The message buffer that has already received a data frame[Note] is overwritten by a newly received data frame

Note: The “message buffer that has already received a data frame” is a receive message buffer whose DN bit has been set to 1

Remark: A remote frame is received and stored, regardless of the setting of OWS and DN. A remote frame that satisfies the other conditions (ID matches, RTR = 0, TRQ = 0) is always received and stored in the corresponding message buffer (interrupt generated, DN flag set, MDLC[3:0] bits updated, and recorded to the receive history list)

RTR	Remote Frame Request Bit[Note]
0	Transmit a data frame
1	Transmit a remote frame

Note: The RTR bit specifies the type of message frame that is transmitted from a message buffer defined as a transmit message buffer.Even if a valid remote frame has been received, RTR of the transmit message buffer that has received the frame remains cleared to 0.Even if a remote frame whose ID matches has been received from the CAN bus with the RTR bit of the transmit message buffer set to 1 to transmit a remote frame, that remote frame is not received or stored (interrupt generated, DN flag set, MDLC[3:0] bits updated, and recorded to the receive history list).

MT2	MT1	MT0	Message Buffer Type Setting Bit
0	0	0	Transmit message buffer
0	0	1	Receive message buffer (no mask setting)
0	1	0	Receive message buffer (mask 1 set)
0	1	1	Receive message buffer (mask 2 set)
1	0	0	Receive message buffer (mask 3 set)
1	0	1	Receive message buffer (mask 4 set)
Other than above			Setting prohibited.

MA0	Message Buffer Assignment Bit
0	Message buffer not used
1	Message buffer used

Caution: Be sure to write 0 to bits 2 and 1.

13.6.23 CAN Message ID Register m (CnMIDLm, CnMIDHm)

The CnMIDLm and CnMIDHm registers are used to set an identifier (ID).

Figure 13-48: CAN Message ID Register m (CnMIDLm, CnMIDHm) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMIDLm	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	0BH	undefined
R/W										
	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	0AH	

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMIDHm	IDE	0		0	ID28	ID27	ID26	ID25	0DH	undefined
R/W										
	7	6	5	4	3	2	1	0		
	ID23	ID22		ID21	ID20	ID19	ID18	ID17	0CH	

IDE	Format Mode Specification Bit
0	Standard format mode (ID28 to ID18: 11 bits)[Note]
1	Extended format mode (ID28 to ID0: 29 bits)

Note: The ID17 to ID0 bits are not used.

ID28 to ID0	Message ID
ID28 to ID18	Standard ID value of 11 bits (when IDE = 0)
ID28 to ID0	Extended ID value of 29 bits (when IDE = 1)

Caution: Be sure to write 0 to bits 14 and 13 of the CnMIDHm register.

13.6.24 CAN Message Control Register m (CnMCTRLm)

The CnMCTRLm register is used to control the operation of the message buffer.

Figure 13-49: CAN Message Control Register m (CnMCTRLm) Format (1/3)

(a) Read

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMCTRLm	0	0	MUC	0	0	0	0	0	0FH	00x00000B
	7	6	5	4	3	2	1	0		
	0	0	0	MOW	IE	DN	TRQ	RDY	0EH	000xx000B

(b) Write

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnMCTRLm	0	0	0	0	Set IE	0	Set TRQ	Set RDY	0FH	
	7	6	5	4	3	2	1	0		
	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY	0EH	

(a) Read

MUC[Note]	Message Buffer Data Updating Bit
0	The CAN module is not updating the message buffer (reception and storage)
1	The CAN module is updating the message buffer (reception and storage)

Note: The MUC bit is undefined until the first reception and storage is performed

MOW	Message Buffer Overwrite Status Bit
0	The message buffer is not overwritten by a newly received data frame
1	The message buffer is overwritten by a newly received data frame

Remark: MOW is not set to 1 even if a remote frame is received and stored in the transmit message buffer with DN = 1

IE	Message Buffer Interrupt Request Enable Bit
0	Receive message buffer: Valid message reception completion interrupt disabled. Transmit message buffer: Normal message transmission completion interrupt disabled
1	Receive message buffer: Valid message reception completion interrupt enabled. Transmit message buffer: Normal message transmission completion interrupt enabled

Figure 13-49: CANn Message Control Register m (CnMCTRLm) Format (2/3)

DN	Message Buffer Data Updating Bit
0	A data frame or remote frame is not stored in the message buffer
1	A data frame or remote frame is stored in the message buffer

TRQ	Message Buffer Transmission Request Bit
0	No message frame transmitting request that is pending or being transmitted is in the message buffer
1	The message buffer is holding transmission of a message frame pending or is transmitting a message frame

RDY	Message Buffer Ready Bit
0	The message buffer can be written by software. The CAN module cannot write to the message buffer
1	Writing the message buffer by software is ignored (except a write access to the RDY, TRQ, DN, and MOW bits). The CAN module can write to the message buffer

Caution: Do not clear the RDY bit (0) during message transmission. Follow the transmission abort process about clearing the RDY bit (0) for redefinition of the message buffer.

(b) Write

Clear MOW	MOW Bit Setting
0	MOW bit is not changed
1	MOW bit is cleared to 0

Set IE	Clear IE	Setting of IE Bit
0	1	IE bit is cleared to 0
1	0	IE bit is set to 1
Other than above		IE bit is not changed

Clear DN	Setting of DN Bit
1	DN bit is cleared to 0
0	DN bit is not changed

Caution: Do not set the DN bit to 1 by software. Be sure to write 0 to bit 10

Figure 13-49: CANn Message Control Register m (CnMCTRLm) Format (3/3)

(b) Write

Set TRQ	Clear TRQ	Setting of TRQ Bit
0	1	TRQ bit is cleared to 0
1	0	TRQ bit is set to 1
Other than above		TRQ bit is not changed

Set RDY	Clear RDY	Setting of RDY Bit
0	1	RDY bit is cleared to 0
1	0	RDY bit is set to 1
Other than above		RDY bit is not changed

13.7 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.

- CAN global control register (CnGMCTRL)
- CAN global automatic block transmission control register (CnGMABT)
- CAN module control register (CnCTRL)
- CAN module interrupt enable register (CnIE)
- CAN module interrupt status register (CnINTS)
- CAN module receive history list register (CnRGPT)
- CAN module transmit history list register (CnTGPT)
- CAN module time stamp register (CnTS)
- CAN message control register (CnMCTRLm)

Remark: n: 0 - 4 = Number of channel
m: 0 - 47 = message buffer number

All the 16 bits in the above registers can be read via the usual method. Use the procedure described in Figure 13-50 below to set or clear the lower 8 bits in these registers.
Setting or clearing of lower 8 bits in the above registers is performed in combination with the higher 8 bits (refer to the 16-bit data after a write operation in Figure 13-51). Figure 13-50 shows how the values of set bits or clear bits relate to set/clear/no change operations in the corresponding register.

Figure 13-50: Example of Bit Setting/Clearing Operations

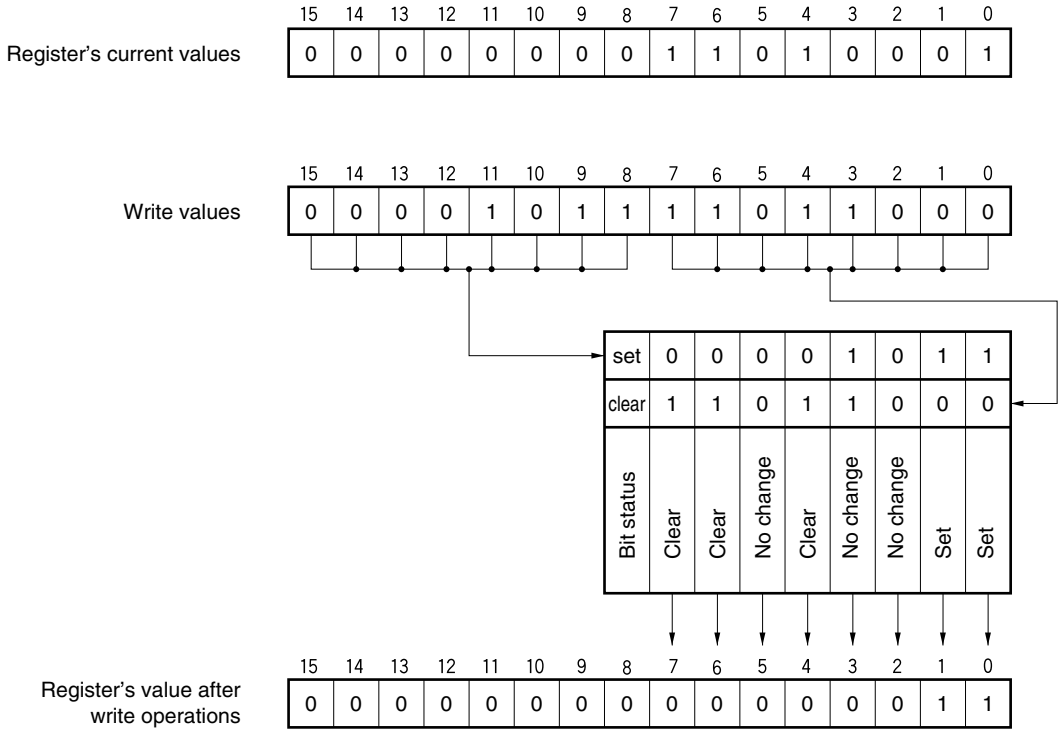


Figure 13-51: 16-Bit Data during Write Operation

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

Remark: n = 0 to 7

13.8 CAN Controller Initialization

13.8.1 Initialization of CAN module

Before the CAN module operation is enabled, the CAN module system clock needs to be determined by setting the CCP[3:0] bits of the CnGMCS 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 the GOM bit of the CnGMCTRL register.
For the procedure of initializing the CAN module, refer to **13.16 "Operation of CAN Controller" on page 604**.

13.8.2 Initialization of message buffer

After the CAN module is enabled, the message buffers contain undefined values. A minimum initialization for all the message buffers, even for those not used in the application, is necessary before switching the CAN module from the initialization mode to one of the operation modes.

- Clear the RDY, TRQ, and DN bits of the CnMCTRLm register to 0.
- Clear the MA0 bit of the CnMCONFm register to 0.

13.8.3 Redefinition of message buffer

Redefining a message buffer means changing the ID and control information of the message buffer while a message is being received or transmitted, without affecting other transmission/reception operations.

(1) To redefine message buffer in initialization mode

Place the CAN module in the initialization mode once and then change the ID and control information of the message buffer in the initialization mode. After changing the ID and control information, set the CAN module in an operation mode.

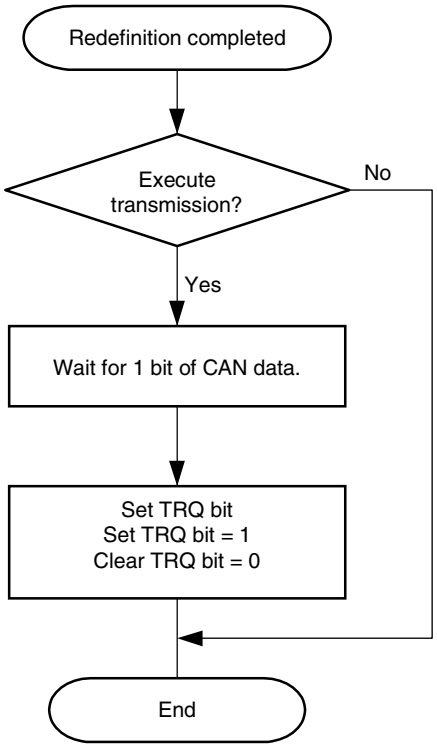
(2) To redefine message buffer during reception

Perform redefinition as shown in **Figure 13-64, "Message Buffer Redefinition," on page 607**.

(3) To redefine message buffer during transmission

To rewrite the contents of a transmit message buffer to which a transmission request has been set, perform transmission abort processing (refer to **13.10.4 (1)"Transmission abort in normal operation mode" on page 586** and **13.10.4 (3)"Transmission abort in normal operation mode with automatic block transmission (ABT)" on page 586**). Confirm that transmission has been aborted or completed, and then redefine the message buffer. After redefining the transmit message buffer, set a transmission request using the procedure described below. When setting a transmission request to a message buffer that has been redefined without aborting the transmission in progress, however, the 1-bit wait time is not necessary.

Figure 13-52: Setting Transmission Request (TRQ) to Transmit Message Buffer after Redefining



Cautions: 1. When a message is received, reception filtering is performed in accordance with the ID and mask set to each receive message buffer. If the procedure in Figure Figure 13-64, "Message Buffer Redefinition," on page 607 is not observed, the contents of the message buffer after it has been redefined may contradict the result of reception (result of reception filtering). If this happens, check that the ID and IDE received first and stored in the message buffer following redefinition are those stored after the message buffer has been redefined. If no ID and IDE are stored after redefinition, redefine the message buffer again.

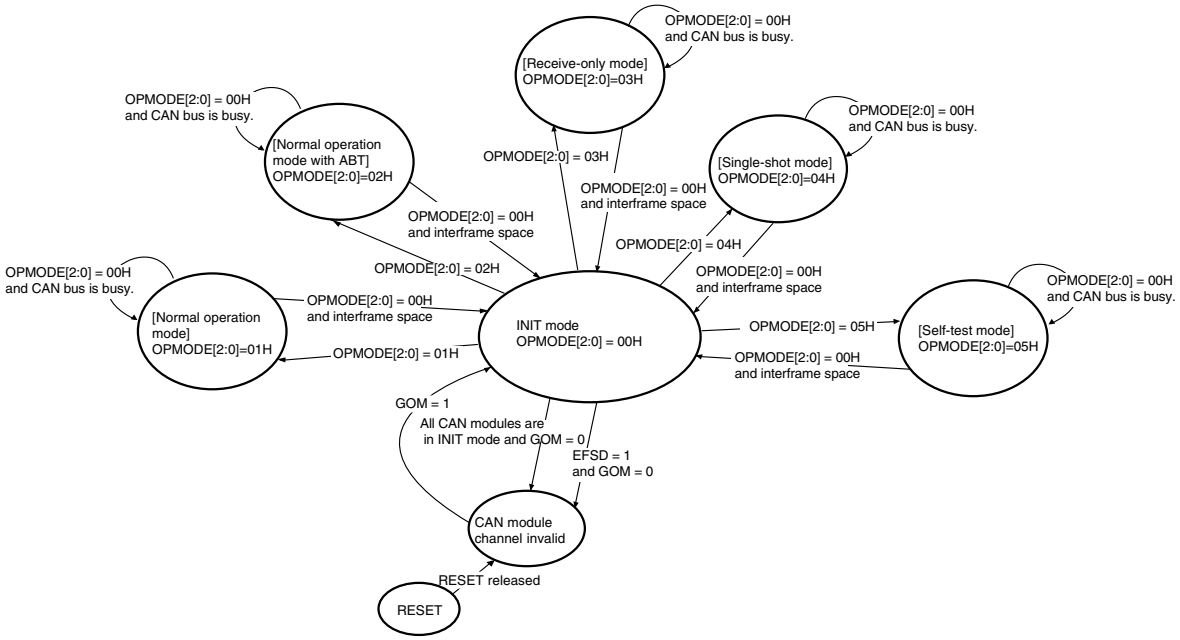
2. When a message is transmitted, the transmission priority is checked in accordance with the ID, IDE, and RTR bits set to each transmit message buffer to which a transmission request was set. The transmit message buffer having the highest priority is selected for transmission. If the procedure in Figure 13-52 is not observed, a message with an ID not having the highest priority may be transmitted after redefinition.

13.8.4 Transition from Initialization mode to Operation mode

The CAN module can be switched to the following operation modes:

- Normal operation mode
- Normal operation mode with ABT
- Receive-only mode
- Single-shot mode
- Self-test mode

Figure 13-53: Transition to Operation Modes



The transition from the initialization mode to an operation mode is controlled by the bit string OPMODE[2:0] in the CnCTRL register.
Changing from one operation mode into another requires shifting to the initialization mode in between. Do not change one operation mode to another directly; otherwise the operation will not be guaranteed.
Requests for transition from the operation mode to the initialization mode are held pending when the CAN bus is not in the interframe space (i.e., frame reception or transmission is in progress), and the CAN module enters the initialization mode at the first bit in the interframe space (the value of OPMODE[2:0] are changed to 00H). After issuing a request to change the mode to the initialization mode, read the OPMODE[2:0] bits until their value becomes 000B to confirm that the module has entered the initialization mode (refer to **Figure 13-62, "Re-initialization," on page 605**).

13.8.5 Resetting Error Counter CnERC of CAN module

If it is necessary to reset the CAN module error counter CnERC and the CAN module information register CnINFO when re-initialization or forced recovery from the bus-off state is made, set the CCERC bit of the CnCTRL register to 1 in the initialization mode. When this bit is set to 1, the CAN module error counter CnERC and the CAN module information register CnINFO are cleared to their default values.

13.9 Message Reception

13.9.1 Message reception

In all the operation modes, the complete message buffer area is analyzed to find a suitable buffer to store a newly received message. All message buffers satisfying the following conditions are included in that evaluation (RX-search process).

- Used as a message buffer
  (MA0 bit of CnMCONFm register set to 1B.)
- Set as a receive message buffer
  (MT[2:0] bits of CnMCONFm register set to 001B, 010B, 011B, 100B, or 101B.)
- Ready for reception
  (RDY bit of CnMCTRLm register set to 1.)

When two or more message buffers of the CAN module receive a message, the message is stored according to the priority explained below. The message is always stored in the message buffer with the highest priority, not in a message buffer with a low priority. For example, when an unmasked receive message buffer and a receive message buffer linked to mask 1 have the same ID, the received message is not stored in the message buffer linked to mask 1, even if that message buffer has not received a message and a message has already been received in the unmasked receive message buffer. In other words, when a condition has been set to store a message in two or more message buffers with different priorities, the message buffer with the highest priority always stores the message; the message is not stored in message buffers with a lower priority. This also applies when the message buffer with the highest priority is unable to receive and store a message (i.e., when DN = 1 indicating that a message has already been received, but rewriting is disabled because OWS = 0). In this case, the message is not actually received and stored in the candidate message buffer with the highest priority, but neither is it stored in a message buffer with a lower priority.

Table 13-25: Message Reception

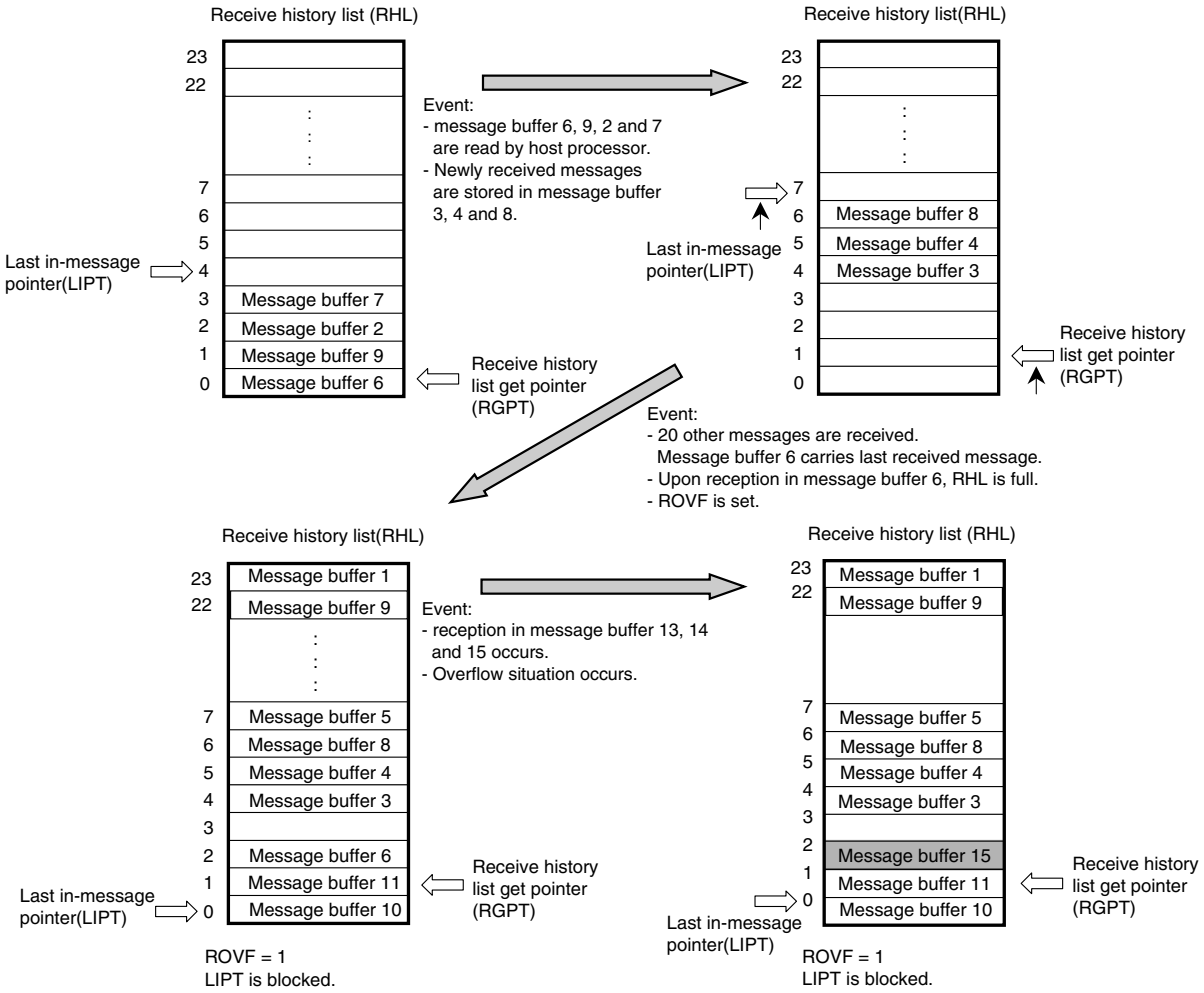
Priority	Storing Condition If Same ID is Set	
1 (high)	Unmasked message buffer	DN = 0
		DN = 1 and OWS = 1
2	Message buffer linked to mask 1	DN = 0
		DN = 1 and OWS = 1
3	Message buffer linked to mask 2	DN = 0
		DN = 1 and OWS = 1
4	Message buffer linked to mask 3	DN = 0
		DN = 1 and OWS = 1
5 (low)	Message buffer linked to mask 4	DN = 0
		DN = 1 and OWS = 1

13.9.2 Receive history list function

The receive history list (RHL) function records in the receive history list the number of the receive message buffer in which each data frame or remote frame was received and stored. The RHL consists of storage elements equivalent to up to 23 messages, the last in-message pointer (LIPT) with the corresponding CnLIPT register and the receive history list get pointer (RGPT) with the corresponding CnRGPT register.
The RHL is undefined immediately after the transition of the CAN module from the initialization mode to one of the operation modes.
The CnLIPT register holds the contents of the RHL element indicated by the value of the LIPT pointer minus 1. By reading the CnLIPT register, therefore, the number of the message buffer that received and stored a data frame or remote frame first can be checked. The LIPT pointer is utilized as a write pointer that indicates to what part of the RHL a message buffer number is recorded. Any time a data frame or remote frame is received and stored, the corresponding message buffer number is recorded to the RHL element indicated by the LIPT pointer. Each time recording to the RHL has been completed, the LIPT pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.
The RGPT pointer is utilized as a read pointer that reads a recorded message buffer number from the RHL. This pointer indicates the first RHL element that the CPU has not read yet. By reading the CnRGPT register by software, the number of a message buffer that has received and stored a data frame or remote frame can be read. Each time a message buffer number is read from the CnRGPT register, the RGPT pointer is automatically incremented.
If the value of the RGPT pointer matches the value of the LIPT pointer, the RHPM bit (receive history list pointer match) of the CnRGPT register is set to 1. This indicates that no message buffer number that has not been read remains in the RHL. If a new message buffer number is recorded, the LIPT pointer is incremented and because its value no longer matches the value of the RGPT pointer, the RHPM bit is cleared. In other words, the numbers of the unread message buffers exist in the RHL.
If the LIPT pointer is incremented and matches the value of the RGPT pointer minus 1, the ROVF bit (receive history list overflow) of the CnRGPT register is set to 1. This indicates that the RHL is full of numbers of message buffers that have not been read. When further message reception and storing occur, the last recorded message buffer number is overwritten by the number of the message buffer that received and stored the new message. In this case, after the ROVF bit has been set (1), the recorded message buffer numbers in the RHL do not completely reflect the chronological order.
However the non-recovered receptions in the RHL are still recoverable. Therefore the application needs to browse all RX-buffer and check the DN bits.

As long as the RHL contains 23 or less entries the sequence of occurrence is maintained. If more receptions occur without reading the RHL by the host processor, complete sequence of receptions can not be recovered.

Figure 13-54: Receive History List



ROVF = 1 defines that LIPT equals RGPT - 1 while message buffer number stored to element indicated by LIPT - 1.

13.9.3 Mask function

It can be defined whether masking of the identifier that is set to a message buffer is linked with another message buffer.
By using the mask function, the identifier of a message received from the CAN bus can be compared with the identifier set to a message buffer in advance. Regardless of whether the masked ID is set to "0" or "1", the received message can be stored in the defined message buffer.
While the mask function is in effect, an identifier bit that is defined to be "1" by a mask in the received message is not compared with the corresponding identifier bit 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 ID, in which bits ID27 to ID25 are "0" and bits ID24 and ID22 are "1", are to be stored in message buffer 14. The procedure for this example is shown below.

Figure 13-55: Mask Function Identifier Examples (1/2)

(a) <1> Identifier to be stored in message buffer

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x

Remark: x = don’t care

(b) <2> Identifier to be configured in message buffer 14 (example)
(using CANn message ID registers L14 and H14 (CnMIDL14 and CnMIDH14))

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				

ID with ID27 to ID25 cleared to "0" and ID24 and ID22 set to "1" is registered (initialized) to message buffer 14.

Remark: Message buffer 14 is set as a standard format identifier that is linked to mask 1 (MT[2:0] of CnMCONF14 register are set to 010B).

Figure 13-55: Mask Function Identifier Examples (2/2)

(c) <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				

Remark: 1: Not compared (masked)
0: Compared

The CMID27 to CMID24 and CMID22 bits are cleared to "0", and CMID28, CMID23, and CMID21 to CMID0 bits are set to "1".

13.9.4 Multi buffer receive block function

The multi buffer receive block (MBRB) function is used to store a block of data in two or more message buffers sequentially with no CPU interaction, by setting the same ID to two or more message buffers with the same message buffer type. These message buffers can be allocated anywhere in the message buffer memory, they do not even have to follow each other adjacently.
Suppose, for example, the same message buffer type is set to 10 message buffers, message buffers 10 to 19, and the same ID is set to each message buffer. If the first message whose ID matches the ID of the message buffers is received, it is stored in message buffer 10. At this point, the DN bit of message buffer 10 is set, prohibiting overwriting the message buffer when subsequent messages are received.
If the next message with a matching ID is received, it is received and stored in message buffer 11. Each time a message with a matching ID is received, it is sequentially (in the ascending order) stored in message buffers 12, 13, and so on. Even when a data block consisting of multiple messages is received, the messages can be stored and received without overwriting the previously received matching-ID data.
Whether a data block has been received and stored can be checked by setting the IE bit of the CnMCTRLm register of each message buffer. For example, if a data block consists of k messages, k message buffers are initialized for reception of the data block. The IE bit in message buffers 0 to (k-2) is cleared to 0 (interrupts disabled), and the IE bit in message buffer k-1 is set to 1 (interrupts enabled). In this case, a reception completion interrupt occurs when a message has been received and stored in message buffer k-1, indicating that MBRB has become full. Alternatively, by clearing the IE bit of message buffers 0 to (k-3) and setting the IE bit of message buffer k-2, a warning that MBRB is about to overflow can be issued.
The basic conditions of storing receive data in each message buffer for the MBRB are the same as the conditions of storing data in a single message buffer.

Cautions:

1. **MBRB can be configured for each of the same message buffer types. Therefore, even if a message buffer of another MBRB whose ID matches but whose message buffer type is different has a vacancy, the received message is not stored in that message buffer, but instead discarded.**
2. **MBRB does not have a ring buffer structure. Therefore, after a message is stored in the message buffer having the highest number in the MBRB configuration, a newly received message will not be stored in the message buffer having the lowest message buffer number.**
3. **MBRB operates based on the reception and storage conditions; there are no settings dedicated to MBRB, such as function enable bits. By setting the same message buffer type and ID to two or more message buffers, MBRB is automatically configured.**
4. **With MBRB, "matching ID" means "matching ID after mask". Even if the ID set to each message buffer is not the same, if the ID that is masked by the mask register matches, it is considered a matching ID and the buffer that has this ID is treated as the storage destination of a message.**
5. **The priority between MBRBs is mentioned in the Table 13-25, “Message Reception,” on page 574.**

13.9.5 Remote frame reception

In all the operation modes, when a remote frame is received, the message buffer that is to store the remote frame is searched from all the message buffers satisfying the following conditions.

- Used as a message buffer
  (MA0 bit of CnMCONFm register set to 1B.)
- Set as a transmit message buffer
  (MT[2:0] bits in CnMCONFm register set to 000B)
- Ready for reception
  (RDY bit of CnMCTRLm register set to 1.)
- Set to transmit message
  (RTR bit of CnMCONFm register is cleared to 0.)
- Transmission request is not set.
  (TRQ bit of CnMCTRLm register is cleared to 1.)

Upon acceptance of a remote frame, the following actions are executed if the ID of the received remote frame matches the ID of a message buffer that satisfies the above conditions.

- The DLC[3:0] bit string in the CnMDLCm register stores the received DLC value.
- CnMDATA0m to CnMDATA7m in the data area are not updated (data before reception is saved).
- The DN bit of the CnMCTRLm register is set to 1.
- The CINTS1 bit of the CnINTS register is set to 1 (if the IE bit in the CnMCTRLm register of the message buffer that receives and stores the frame is set to 1).
- The reception completion interrupt (INTRECn) is output (if the IE bit in the CnMCTRLm register of the message buffer that receives and stores the frame is set to 1 and if the CIE1 bit of the CnIE register is set to 1).
- The message buffer number is recorded to the receive history list.

Caution: When a message buffer is searched for receiving and storing a remote frame, overwrite control by the OWS bit of the CnMCONFm register of the message buffer and the DN bit of the CnMCTRLm register are not affected.
If more than one transmit message buffer has the same ID and the ID of the received remote frame matches that ID, the remote frame is stored in the transmit message buffer with the lowest message buffer number.

13.10 Message Transmission

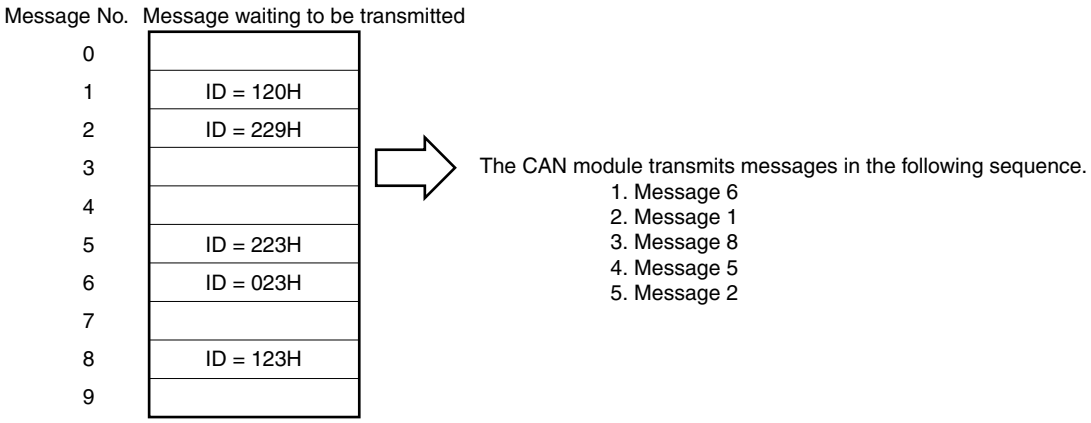
13.10.1 Message transmission

A message buffer with its TRQ bit set to 1 participates in the search for the most high-prioritized message when the following conditions are fulfilled. This behaviour is valid for all operational modes.

- Used as a message buffer
  (MA0 bit of CnMCONFm register set to 1B.)
- Set as a transmit message buffer
  (MT[2:0] bits of CnMCONFm register set to 000B.)
- Ready for transmission
  (RDY bit of CnMCTRLm register set to 1.)

The CAN system is a multi-master communication system. In a system like this, the priority of message transmission is determined based on message identifiers (IDs). To facilitate transmission processing by software when there are several messages awaiting transmission, the CAN module uses hardware to check the ID of the message with the highest priority and automatically identifies that message. This eliminates the need for software-based priority control.
Transmission priority is controlled by the identifier (ID).

Figure 13-56: Message Processing Example



After the transmit message search, the transmit message with the highest priority of the transmit message buffers that have a pending transmission request (message buffers with the TRQ bit set to 1 in advance) is transmitted.
If a new transmission request is set, the transmit message buffer with the new transmission request is compared with the transmit message buffer with a pending transmission request. If the new transmission request has a higher priority, it is transmitted, unless transmission of a message with a low priority has already started. If transmission of a message with a low priority has already started, however, the new transmission request is transmitted later. The highest priority is determined according to the following rules.

Table 13-26: Message Transmission

Priority	Conditions	Description
1 (high)	Value of first 11 bits of ID [ID28 to ID18]:	The message frame with the lowest value represented by the first 11 bits of the ID is transmitted first. If the value of an 11-bit standard ID is equal to or smaller than the first 11 bits of a 29-bit extended ID, the 11-bit standard ID has a higher priority than message frame with the 29-bit extended ID.
2	Frame type	A data frame with an 11-bit standard ID (RTR bit is cleared to 0) has a higher priority than a remote frame with a standard ID and a message frame with an extended ID.
3	ID type	A message frame with a standard ID (IDE bit is cleared to 0) has a higher priority than a message frame with an extended ID.
4	Value of lower 18 bits of ID [ID17 to ID0]:	If more than one transmission-pending extended ID message frame have equal values in the first 11 bits of the ID and the same frame type (equal RTR bit values), the message frame with the lowest value in the lower 18 bits of its extended ID is transmitted first.
5 (low)	Message buffer number	If two or more message buffers request transmission of message frames with the same ID, the message from the message buffer with the lowest message buffer number is transmitted first.

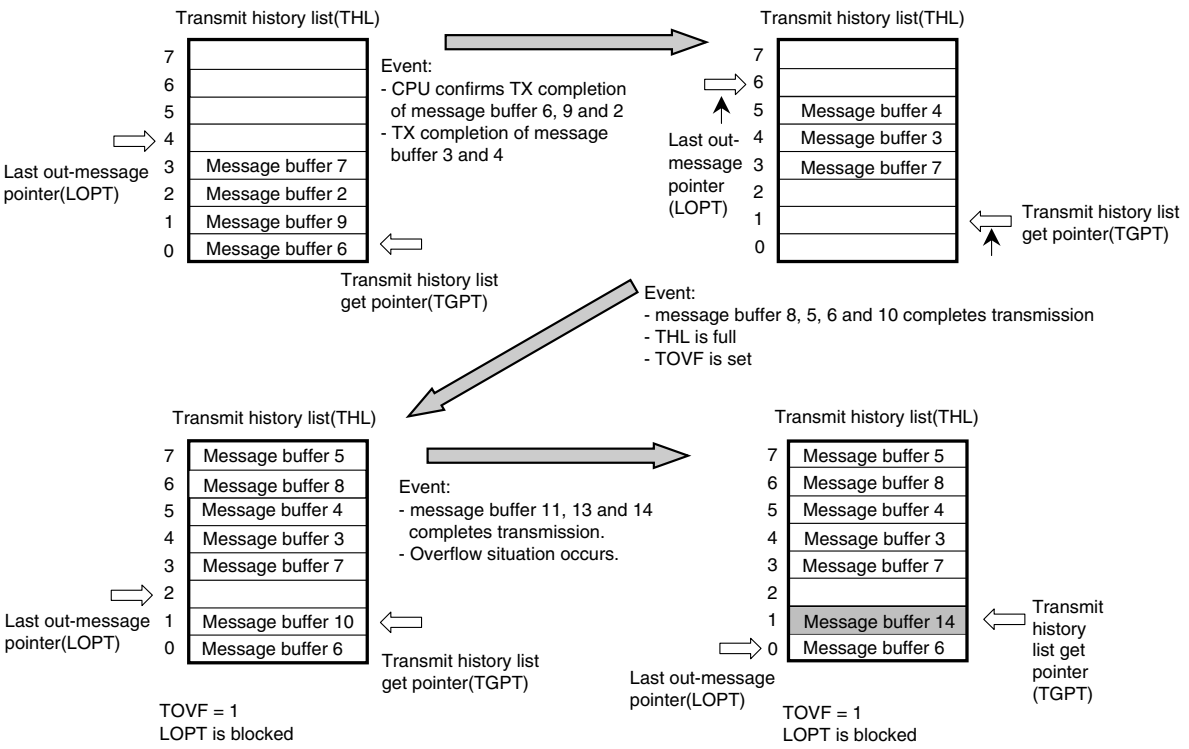
Remark: If automatic block transmission request bit ABTTRG is set to 1 in the normal operation mode with ABT, the TRQ bit is set to 1 only for one message buffer in the ABT message buffer group.
If the ABT mode was triggered by ABTTRG (1), one TRQ is set to 1 in the ABT area (buffer 0 through 7). Beyond this TRQ, the application can request transmissions (set TRQ to 1) for other TX-message buffers that do not belong to the ABT area. In that case an interval arbitration process (TX-search) evaluates all TX-message buffers with TRQ set to 1 and chooses the message buffer that contains the highest prioritized identifier for the next transmission. If there are 2 or more identifiers that have the highest priority (i.e. identical identifiers), the message located at the lowest message buffer number is transmitted at first.
Upon successful transmission of a message frame, the following operations are performed.

- The TRQ flag of the corresponding transmit message buffer is automatically cleared to 0.
- The transmission completion status bit CINTS0 of the CnINTS register is set to 1 (if the interrupt enable bit (IE) of the corresponding transmit message buffer is set to 1).
- An interrupt request signal INTRRX1 is output (if the CIE0 bit of the CnIE register is set to 1 and if the interrupt enable bit (IE) of the corresponding transmit message buffer is set to 1).

13.10.2 Transmit history list function

The transmit history list (THL) function records in the transmit history list the number of the transmit message buffer from which data or remote frames have been were sent. The THL consists of storage elements equivalent to up to seven messages, the last out-message pointer (LOPT) with the corresponding CnLOPT register, and the transmit history list get pointer (TGPT) with the corresponding CnTGPT register.
The THL is undefined immediately after the transition of the CAN module from the initialization mode to one of the operation modes.
The CnLOPT register holds the contents of the THL element indicated by the value of the LOPT pointer minus 1. By reading the CnLOPT register, therefore, the number of the message buffer that transmitted a data frame or remote frame first can be checked. The LOPT pointer is utilized as a write pointer that indicates to what part of the THL a message buffer number is recorded. Any time a data frame or remote frame is transmitted, the corresponding message buffer number is recorded to the THL element indicated by the LOPT pointer. Each time recording to the THL has been completed, the LOPT pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.
The TGPT pointer is utilized as a read pointer that reads a recorded message buffer number from the THL. This pointer indicates the first THL element that the CPU has not yet read. By reading the CnTGPT register by software, the number of a message buffer that has completed transmission can be read. Each time a message buffer number is read from the CnTGPT register, the TGPT pointer is automatically incremented.
If the value of the TGPT pointer matches the value of the LOPT pointer, the THPM bit (transmit history list pointer match) of the CnTGPT register is set to 1. This indicates that no message buffer numbers that have not been read remain in the THL. If a new message buffer number is recorded, the LOPT pointer is incremented and because its value no longer matches the value of the TGPT pointer, the THPM bit is cleared. In other words, the numbers of the unread message buffers exist in the THL.
If the LOPT pointer is incremented and matches the value of the TGPT pointer minus 1, the TOVF bit (transmit history list overflow) of the CnTGPT register is set to 1. This indicates that the THL is full of message buffer numbers that have not been read. If a new message is received and stored, the message buffer number recorded last is overwritten by the number of the message buffer that transmitted its message afterwards. After the TOVF bit has been set (1), therefore, the recorded message buffer numbers in the THL do not completely reflect the chronological order.
However the other transmitted messages can be found by a CPU search applied to all transmit message buffers unless the CPU has not overwritten a transmit object in one of these buffers beforehand. In total up to six transmission completions can occur without overflowing the THL.

Figure 13-57: Transmit History List



TOVF = 1 defines that LOPT equals TOPT - 1 while message buffer number stored to element indicated by LOPT - 1.

13.10.3 Automatic block transmission (ABT)

The automatic block transmission (ABT) function is used to transmit two or more data frames successively with no CPU interaction. The maximum number of transmit message buffers assigned to the ABT function is eight (message buffer numbers 0 to 7).
By setting OPMODE[2:0] of the CnCTRL register to 010B, "normal operation mode with automatic block transmission function" (hereafter referred to as ABT mode) can be selected.
To issue an ABT transmission request, define the message buffers by software first. Set the MA0 bit (1) in all the message buffers used for ABT, and define all the buffers as transmit message buffers by setting MT[2:0] bits to 000B. Be sure to set the same ID for each message buffer for ATB even when that ID is being used for all the message buffers. To use two or more IDs, set the ID of each message buffer by using the CnMIDLm and CnMIDHm registers. Set the CnMDLCm and CnMDATA0m to CnMDATA7m registers before issuing a transmission request for the ABT function.
After initialization of message buffers 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 the data for the ABT message buffers has been prepared, set the ABTTRG bit to 1. Automatic block transmission is then started. When ABT is started, the TRQ bit in the first message buffer (message buffer 0) is automatically set to 1. After transmission of the data of message buffer 0 has finished, TRQ of the next message buffer, message buffer 1, is set automatically. In this way, transmission is executed successively.
A delay time can be inserted by program in the interval in which the transmission request (TRQ) is automatically set while successive transmission is being executed. The delay time to be inserted is defined by the CnGMABTD register. The unit of the delay time is DBT (data bit time). DBT depends on the setting of the CnBRP and CnBTR registers.
Among transmit objects within the ABT-area, the priority of the transmission ID is not evaluated. The data of message buffers 0 to 7 are sequentially transmitted. When transmission of the data frame from message buffer 7 has been completed, the ABTTRG bit is automatically cleared to 0 and the ABT operation is finished.

If the RDY bit of an ABT message buffer is cleared during ABT, no data frame is transmitted from that buffer, ABT is stopped, and the ABTTRG bit is cleared. After that, transmission can be resumed from the message buffer where ABT stopped, by setting the RDY and ABTTRG bits to 1 by software. To not resume transmission from the message buffer where ABT stopped, the internal ABT engine can be reset by setting the ABTCLR bit to 1 while ABT mode is stopped and ABTTRG is cleared to 0. In this case, transmission is started from message buffer 0 if the ABTCLR bit is cleared to 0 and then the ABTTRG bit is set to 1.
An interrupt can be used to check if data frames have been transmitted from all the message buffers for ABT. To do so, the IE bit of the CnMCTRLm register of each message buffer except the last message buffer needs to be cleared (0). If a transmit message buffer other than those used by the ABT function (message buffer 8 to m_{MAX} - 1[Note]) is assigned to a transmit message buffer, the priority of the message to be transmitted is determined by the priority of the transmission message buffer ID of the ABT message buffer whose transmission is currently held pending and the transmission ID of the message buffers other than those used by the ABT function.
Transmission of a data frame from an ABT message buffer is not recorded in the transmit history list (THL).

Cautions:

1. **Set the ABTCLR bit to 1 while the ABTTRG bit is cleared to 0 in order to resume ABT operation at buffer No.0. If the ABTCLR bit is set to 1 while the ABTTRG bit is set to 1, the subsequent operation is not guaranteed.**
2. **If the automatic block transmission engine is cleared by setting the ABTCLR bit to 1, the ABTCLR bit is automatically cleared immediately after the processing of the clearing request is completed.**
3. **Do not set the ABTTRG bit in the initialization mode. If the ABTTRG bit is set in the initialization mode, the proper operation is not guaranteed after the mode is changed from the initialization mode to the ABT mode.**
4. **Do not set TRQ of the ABT message buffers to 1 by software in the normal operation mode with ABT. Otherwise, the operation is not guaranteed.**
5. **The CnGMABTD register is used to set the delay time that is inserted in the period from completion of the preceding ABT message to setting of the TRQ bit for the next ABT message when the transmission requests are set in the order of message numbers for each message for ABT that is successively transmitted in the ABT mode. The timing at which the messages are actually transmitted onto the CAN bus varies depending on the status of transmission from other stations and the status of the setting of the transmission request for messages other than the ABT messages (message buffer 8 to m_{MAX} - 1[Note]).**
6. **If a transmission request is made for a message other than an ABT message and if no delay time is inserted in the interval in which transmission requests for ABT are automatically set (CnGMABTD = 00H), messages other than ABT messages may be transmitted not depending on their priority compared to the priority of the ABT message.**
7. **Do not clear the RDY bit to 0 when ABTTRG = 1.**
8. **If a message is received from another node while normal operation mode with ABT is active, the TX-message from the ABT-area may be transmitted with delay of one frame although CnGMABTD register was set up with 00h.**

Note: m_{MAX} = 48

13.10.4 Transmission abort process

(1) Transmission abort in normal operation mode

The user can clear the TRQ bit of the CnMCTRLm register to 0 to abort a transmission request. The TRQ bit will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using the TSTAT bit of the CnCTRL register and the CnTGPT register, which indicate the transmission status on the CAN bus (for details, refer to the processing in **Figure 13-71, "Transmission Abort Processing (Except Normal Operation Mode with ABT)," on page 614**).

(2) Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)

The user can clear the ABTTRG bit of the CnGMABT register to 0 to abort a transmission request. After checking the ABTTRG bit of the CnGMABT register = 0, clear the TRQ bit of the CnMCTRLm register to 0. The TRQ bit will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using the TSTAT bit of the CnCTRL register and the CnTGPT register, which indicate the transmission status on the CAN bus (for details, refer to the processing).

(3) Transmission abort in normal operation mode with automatic block transmission (ABT)

To abort ABT that is already started, clear the ABTTRG bit of the CnGMABT register to 0. In this case, the ABTTRG bit remains 1 if an ABT message is currently being transmitted and until the transmission is completed (successfully or not), and is cleared to 0 as soon as transmission is finished. This aborts ABT.

If the last transmission (before ABT) was successful, the normal operation mode with ABT 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 in the last transmitted message buffer. If the TRQ bit is set to 1 when clearing the ABTTRG bit is requested, the internal ABT pointer points to the last transmitted message buffer (for details, refer to the process in **Figure 13-72, "Transmission Abort Processing Except for ABT Transmission (Normal Operation Mode with ABT)," on page 615**). If the TRQ bit is cleared to 0 when clearing the ABTTRG bit is requested, the internal ABT pointer is incremented (+1) and points to the next message buffer in the ABT area.

Caution: Be sure to abort ABT by clearing ABTTRG to 0. The operation is not guaranteed if aborting transmission is requested by clearing RDY.

When the normal operation mode with ABT is resumed after ABT has been aborted and ABTTRG is set to 1, the next ABT message buffer to be transmitted can be determined from the following table.

Table 13-27: Transmission Abort

Status of TRQ of ABT Message Buffer	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: The above resumption operation can be performed only if a message buffer ready for ABT exists in the ABT area. For example, an abort request that is issued while ABT of message buffer 7 is in progress is regarded as completion of ABT, rather than abort, if transmission of message buffer 7 has been successfully completed, even if ABTTRG is cleared to 0. If the RDY bit in the next message buffer in the ABT area is cleared to 0, the internal ABT pointer is retained, but the resumption operation is not performed even if ABTTRG is set to 1, and ABT ends immediately.

13.10.5 Remote frame transmission

Remote frames can be transmitted only from transmit message buffers. Set whether a data frame or remote frame is transmitted via the RTR bit of the CnMCONFm register. Setting (1) the RTR bit sets remote frame transmission.

13.11 Power Save Modes

13.11.1 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 module can enter the CAN Sleep mode from all operation modes. Release of the CAN Sleep mode returns the CAN module to exactly the same operation mode from which the CAN Sleep mode was entered.
In the CAN Sleep mode, the CAN module does not transmit messages, even when transmission requests are issued or pending.

(1) Entering CAN Sleep mode

The CPU issues a CAN Sleep mode transition request by writing 01B to the PSMODE[1:0] bits of the CnCTRL register.
This transition request is only acknowledged only under the following conditions.

- The CAN module is already in one of the following operation modes
  - Normal operation mode
  - Normal operation mode with ABT
  - Receive-only mode
  - Single-shot mode
  - Self-test mode
  - CAN Stop mode in all the above operation modes
- The CAN bus state is bus idle (the 4th bit in the interframe space is recessive)[Note]
- No transmission request is pending

Note: If the CAN bus is fixed to dominant, the request for transition to the CAN Sleep mode is held pending.

If any of the conditions mentioned above is not met, the CAN module will operate as follows:

- If the CAN Sleep mode is requested from the initialization mode, the CAN Sleep mode transition request is ignored and the CAN module remains in the initialization mode.
- If the CAN bus state is not bus idle (i.e., the CAN bus state is either transmitting or receiving) when the CAN Sleep mode is requested in one of the operation modes, immediate transition to the CAN Sleep mode is not possible. In this case, the CAN Sleep mode transition request is held pending until the CAN bus state becomes bus idle (the 4th bit in the interframe space is recessive). In the time from the CAN Sleep mode request to successful transition, the PSMODE[1:0] bits remain 00B. When the module has entered the CAN Sleep mode, PSMODE[1:0] are set to 01B.
- If a request for transition to the initialization mode and a request for transition to the CAN Sleep are made at the same time while the CAN module is in one of the operation modes, the request for the initialization mode is enabled. The CAN module enters the initialization mode at a predetermined timing. At this time, the CAN Sleep mode request is not held pending and is ignored.

- Even when initialization mode and sleep mode are requested simultaneously (i.e. the first request has not been granted while the second request is made), the request for initialization has priority over the sleep mode request. The sleep mode request is cancelled when the initialization mode is requested. When a pending request for initialization mode is present, a subsequent request for Sleep mode request is cancelled right at the point in time where it was submitted.

(2) Status in CAN Sleep mode

The CAN module is in one of the following states after it enters the CAN Sleep mode:

- The internal operating clock is stopped and the power consumption is minimized.
- The function to detect the falling edge of the CAN reception pin (CRXDn) remains in effect to wake up the CAN module from the CAN bus.
- To wake up the CAN module from the CPU, data can be written to PSMODE[1:0] of the CAN module control register (CnCTRL), but nothing can be written to other CAN module registers or bits.
- The CAN module registers can be read, except for CnLIPT, CnRGPT, CnLOPT, and CnTGPT.
- The CAN message buffer registers cannot be written or read.
- A request for transition to the initialization mode is not acknowledged and is ignored.

(3) Releasing CAN Sleep mode

The CAN Sleep mode is released by the following events:

- When the CPU writes 00B to the PSMODE[1:0] bits of the CnCTRL register
- A falling edge at the CAN reception pin (CRXDn) (i.e. the CAN bus level shifts from recessive to dominant)

Caution: Even if the falling edge belongs to the SOF of a receive message, this message will not be received and stored. If the CPU has turned off the clock to the CAN while the CAN was in Sleep mode, even subsequently the CAN Sleep mode will not be released and PSMODE [1:0] will continue to be 01B unless the clock to the CAN is supplied again. In addition to this, the receive message will not be received after that.

After releasing the Sleep mode, the CAN module returns to the operation mode from which the CAN Sleep mode was requested and the PSMODE[1:0] bits of the CnCTRL register are reset to 00B. If the CAN Sleep mode is released by a change in the CAN bus state, the CINTS5 bit of the CnINTS register is set to 1, regardless of the CIE bit of the CnIE register. After the CAN module is released from the CAN Sleep mode, it participates in the CAN bus again by automatically detecting 11 consecutive recessive-level bits on the CAN bus.
When a request for transition to the initialization mode is made while the CAN module is in the CAN Sleep mode, that request is ignored; the CPU has to be released from Sleep mode by software first before entering the initialization mode.

13.11.2 CAN STOP Mode

The CAN Stop mode can be used to set the CAN controller to standby mode to reduce power consumption. The CAN module can enter the CAN Stop mode only from the CAN Sleep mode. Release of the CAN Stop mode puts the CAN module in the CAN Sleep mode.
The CAN Stop mode can only be released (entering CAN Sleep mode) by writing 01B to the PSMODE[1:0] bits of the CnCTRL register and not by a change in the CAN bus state. No message is transmitted even when transmission requests are issued or pending.

(1) Entering CAN Stop mode

A CAN Stop mode transition request is issued by writing 11B to the PSMODE[1:0] bits of the CnCTRL register.
A CAN Stop mode request is only acknowledged when the CAN module is in the CAN Sleep mode. In all other modes, the request is ignored.

Caution: To set the CAN module to the CAN Stop mode, the module must be in the CAN Sleep mode. To confirm that the module is in the Sleep mode, check that PSMODE[1:0] = 01B, and then request the CAN Stop mode. If a bus change occurs at the CAN reception pin (CRXD) while this process is being performed, the CAN Sleep mode is automatically released. In this case, the CAN Stop mode transition request cannot be acknowledged.

(2) Status in CAN Stop mode

The CAN module is in one of the following states after it enters the CAN Stop mode:

- The internal operating clock is stopped and the power consumption is minimized.
- To wake up the CAN module from the CPU, data can be written to PSMODE[1:0] of the CAN module control register (CnCTRL), but nothing can be written to other CAN module registers or bits.
- The CAN module registers can be read, except for CnLIPT, CnRGPT, CnLOPT, and CnTGPT.
- The CAN message buffer registers cannot be written or read.
- An initialization mode transition request is not acknowledged and is ignored.

(3) Releasing CAN Stop mode

The CAN Stop mode can only be released by writing 01B to the PSMODE[1:0] bits of the CnCTRL register. After releasing the CAN Stop mode, the CAN module enters the CAN Sleep mode.
When the initialization mode is requested while the CAN module is in the CAN Stop mode, that request is ignored; the CPU has to release the Stop mode and subsequently CAN Sleep mode before entering the initialization mode. It is impossible to enter the other operation mode directly from the CAN Stop mode not entering the CAN Sleep mode, that request is ignored.

13.11.3 Example of using Power Saving modes

In some application systems, it may be necessary to place the CPU in a power saving mode to reduce the power consumption. By using the power saving mode specific to the CAN module and the power saving mode specific to the CPU in combination, the CPU can be woken up from the power saving status by the CAN bus.

Here is an example of using the power saving modes.

First, put the CAN module in the CAN Sleep mode (PSMODE = 01B). Next, put the CPU in the power saving mode. If an edge transition from recessive to dominant is detected at the CAN reception pin (CRXDn) in this status, the CINTS5 bit in the CAN module is set to 1. If the CIE5 bit of the CnCTRL register is set to 1, a wakeup interrupt (INTWUP) is generated. The CAN module is automatically released from the CAN Sleep mode (PSMODE = 00B) and returns to the normal operation mode. The CPU, in response to INTWUP, can release its own power saving mode and return to the normal operation mode.
To further reduce the power consumption of the CPU, the internal clocks, including that of the CAN module, may be stopped. In this case, the operating clock supplied to the CAN module is stopped after the CAN module is put in the CAN Sleep mode. Then the CPU enters a power saving mode in which the clock supplied to the CPU is stopped. If an edge transition from recessive to dominant is detected at the CAN reception pin (CRXDn) in this status, the CAN module can set the CINTS5 bit to 1 and generate the wakeup interrupt (INTWUP) even if it is not supplied with the clock. The other functions, however, do not operate because clock supply to the CAN module is stopped, and the module remains in the CAN Sleep mode. The CPU, in response to INTWUP, releases its power saving mode, resumes supply of the internal clocks, including the clock to the CAN module, after the oscillation stabilization time has elapsed, and starts instruction execution. The CAN module is immediately released from the CAN Sleep mode when clock supply is resumed, and returns to the normal operation mode (PSMODE = 00B).

13.12 Interrupt Function

13.12.1 Interrupts generated by CAN module

The CAN module 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 interrupt sources. When an interrupt request signal that corresponds to two or more interrupt sources is generated, the interrupt sources can be identified by using an interrupt status register. After an interrupt source has occurred, the corresponding interrupt status bit must be cleared to 0 by software.

Table 13-28: 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[Note 1]	CnINTS	CIE0[Note 1]	CnIE	INTTRXn	Message frame successfully transmitted from message buffer m
2	CINTS1[Note 1]	CnINTS	CIE1[Note 1]	CnIE	INTRECn	Valid message frame reception in message buffer m
3	CINTS2	CnINTS	CIE2	CnIE	INTERRn	CAN module error state interrupt[Note 2]
4	CINTS3	CnINTS	CIE3	CnIE		CAN module protocol error interrupt[Note 3]
5	CINTS4	CnINTS	CIE4	CnIE		CAN module arbitration loss interrupt
6	CINTS5	CnINTS	CIE5	CnIE	INTWUPn	CAN module wakeup interrupt from CAN Sleep mode[Note 4]

Notes: 1. The IE bit (message buffer interrupt enable bit) in the CnMCTRL register of the corresponding message buffer has to be set to 1 for that message buffer to participate in the interrupt generation process.

2. This interrupt is generated when the transmission/reception error counter is at the warning level, or in the error passive or bus-off state.

3. This interrupt is generated when a stuff error, form error, ACK error, bit error, or CRC error occurs.

4. This interrupt is generated when the CAN module is woken up from the CAN Sleep mode because a falling edge is detected at the CAN reception pin (CAN bus transition from recessive to dominant).

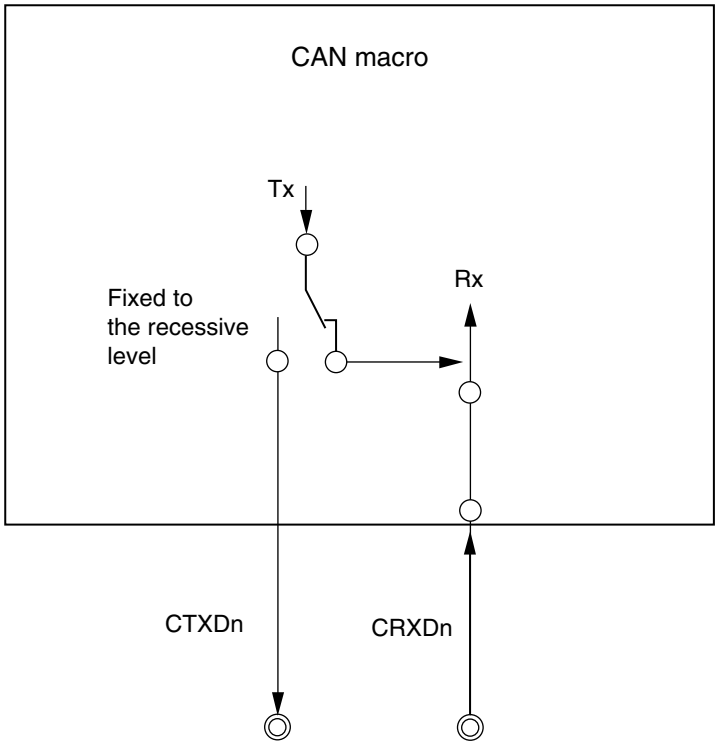
13.13 Diagnosis Functions and Special Operational Modes

The CAN module provides a receive-only mode, single-shot mode, and self-test mode to support CAN bus diagnosis functions or the operation of specific CAN communication methods.

13.13.1 Receive-Only mode

The receive-only mode is used to monitor receive messages without causing any interference on the CAN bus and can be used for CAN bus analysis nodes.
For example, this mode can be used for automatic baud-rate detection. The baud rate in the CAN module is changed until "valid reception" is detected, so that the baud rates in the module match ("valid reception" means a message frame has been received in the CAN protocol layer without occurrence of an error and with an appropriate ACK between nodes connected to the CAN bus). A valid reception does not require message frames to be stored in a receive message buffer (data frames) or transmit message buffer (remote frames). The event of valid reception is indicated by setting the VALID bit of the CnCTRL register (1).

Figure 13-58: CAN Module Terminal Connection in Receive-Only Mode



In the receive-only mode, no message frames can be transmitted from the CAN module to the CAN bus. Transmit requests issued for message buffers defined as transmit message buffers are held pending.
In the receive-only mode, the CAN transmission pin (CTXDn) in the CAN module is fixed to the recessive level. Therefore, no active error flag can be transmitted from the CAN module to the CAN bus even when a CAN bus error is detected while receiving a message frame. Since no transmission can be issued from the CAN module, the transmission error counter TEC is never updated. Therefore, a CAN module in the receive-only mode does not enter the bus-off state.

Furthermore, ACK is not returned to the CAN bus in this mode upon the valid reception of a message frame. Internally, the local node recognizes that it has transmitted ACK. An overload frame cannot be transmitted to the CAN bus.

Caution: If only two CAN nodes are connected to the CAN bus and one of them is operating in the receive-only mode, there is no ACK on the CAN bus. Due to the missing ACK, the transmitting node will transmit an active error flag, and repeat transmitting a message frame. The transmitting node becomes error passive after transmitting the message frame 16 times (assuming that the error counter was 0 in the beginning and no other errors have occurred). After the message frame for the 17th time is transmitted, the transmitting node generates a passive error flag. The receiving node in the receive-only mode detects the first valid message frame at this point, and the VALID bit is set to 1 for the first time.

13.13.2 Single-Shot mode

In the 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 control by software.) All other behaviour of single shot mode is identical to normal operation mode. Features of single shot mode can not be used in combination with normal mode with ABT.
The single-shot mode disables the re-transmission of an aborted message frame transmission according to the setting of the AL bit of the CnCTRL register. When the AL bit is cleared to 0, re-transmission upon arbitration loss and upon error occurrence is disabled. If the AL bit is set to 1, re-transmission upon error occurrence is disabled, but re-transmission upon arbitration loss is enabled. As a consequence, the TRQ bit in a message buffer defined as a transmit message buffer is cleared to 0 by the following events:

- Successful transmission of the message frame
- Arbitration loss while sending the message frame
- Error occurrence while sending the message frame

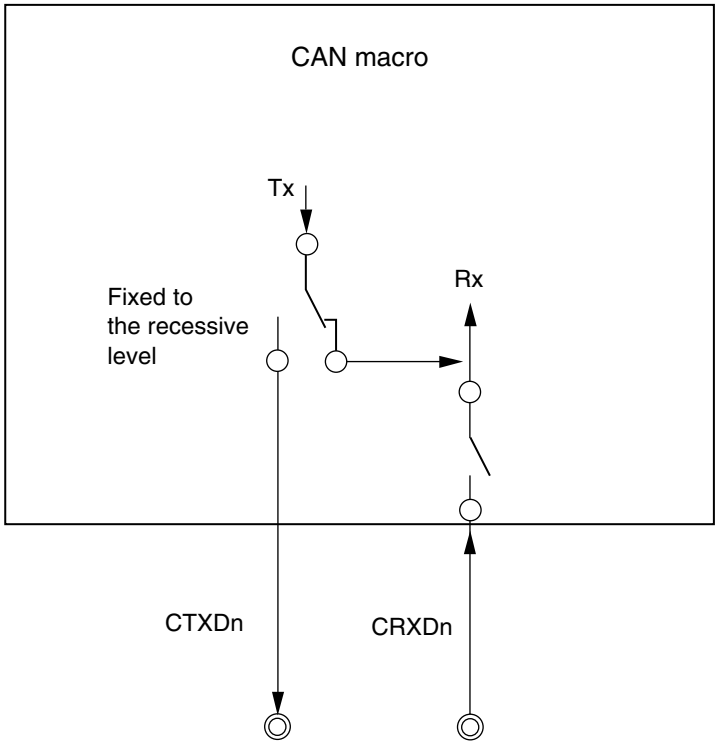
The events arbitration loss and error occurrence can be distinguished by checking the CINTS4 and CINTS3 bits of the CnINTS register respectively, and the type of the error can be identified by reading the LEC[2:0] bits of the CnLEC register.
Upon successful transmission of the message frame, the transmit completion interrupt bit CINTS0 of the CnINTS register is set to 1. If the CIE0 bit of the CnIE register is set to 1 at this time, an interrupt request signal is output.
The single-shot mode can be used when emulating time-triggered communication methods (e.g. TTCAN level 1).

Caution: The AL bit is only valid in Single-shot mode. It does not influence the operation of re-transmission upon arbitration loss in the other operation modes.

13.13.3 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 self-test mode, the CAN module is completely disconnected from the CAN bus, but transmission and reception are internally looped back. The CAN transmission pin (CTXDn) is fixed to the recessive level.
If the falling edge on the CAN reception pin (CRXDn) is detected after the CAN module has entered the CAN Sleep mode from the self-test mode, however, the module is released from the CAN Sleep mode in the same manner as the other operation modes. To keep the module in the CAN Sleep mode, use the CAN reception pin (CRXDn) as a port pin.

Figure 13-59: CAN Module Terminal Connection in Self-Test Mode



13.14 Time Stamp Function

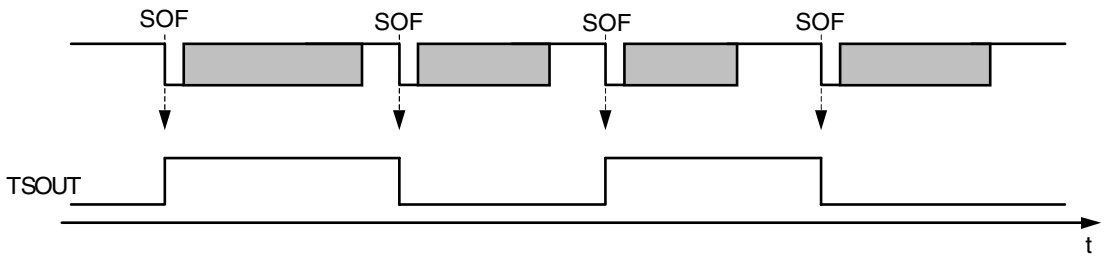
CAN is an asynchronous, serial protocol. All nodes connected to the CAN bus have a local, autonomous clock. As a consequence, the clocks of the nodes have no relation (i.e., the clocks are asynchronous and may have different frequencies).
In some applications, however, a common time base over the network (= global time base) is needed. In order to build up a global time base, a time stamp function is used. The essential mechanism of a time stamp function is the capture of timer values triggered by signals on the CAN bus.

The CAN controller supports the capturing of timer values triggered by a specific frame. An on-chip 16-bit capture timer unit in a microcontroller system is used in addition to the CAN controller. The 16-bit capture timer unit captures the timer value according to a trigger signal (TSOUT) for capturing that is output when a data frame is received from the CAN controller. The CPU can retrieve the time of occurrence of the capture event, i.e., the time stamp of the message received from the CAN bus, by reading the captured value. TSOUT can be selected from the following two event sources and is specified by the TSSEL bit of the CnTS register:

- SOF event (start of frame) (TSSEL = 0)
- EOF event (last bit of end of frame) (TSSEL = 1)

The TSOUT signal is enabled by setting the TSEN bit of the CnTS register to 1.

Figure 13-60: Timing Diagram of Capture Signal TSOUT



TSOUT toggles its level upon occurrence of the selected event during data frame reception (in the above timing diagram, the SOF is used as the trigger event source). To capture a timer value by using TSOUT, the capture timer unit must detect the capture signal at both the rising edge and falling edge.
This time stamp function is controlled by the TSLOCK bit of the CnTS register. When TSLOCK is cleared to 0, TSOUT toggles upon occurrence of the selected event. If TSLOCK is set to 1, TSOUT toggles upon occurrence of the selected event, but the toggle is stopped as the TSEN bit is automatically cleared to 0 when a data frame starts to be received and stored in message buffer 0. This suppresses the subsequent toggle occurrence by TSOUT, so that the time stamp value toggled last (= captured last) can be saved as the time stamp value of the time at which the data frame was received in message buffer 0.

Caution: The time stamp function using TSLOCK is to stop toggle of TSOUT by receiving a data frame in message buffer 0. Therefore, message buffer 0 must be set as a receive message buffer. Since a receive message buffer cannot receive a remote frame, toggle of TSOUT cannot be stopped by reception of a remote frame. Toggle of TSOUT does not stop when a data frame is received in a message buffer other than message buffer 0. For these reasons, a data frame cannot be received in message buffer 0 when the CAN module is in the normal operation mode with ABT, because message buffer 0 must be set as a transmit message buffer. In this operation mode, therefore, the function to stop toggle of TSOUT by TSLOCK cannot be used.

13.15 Baud Rate Settings

Make sure that the settings are within the range of limit values for ensuring correct operation of the CAN controller, as follows.

(a) 5TQ ≤ SPT (sampling point) ≤ 17 TQ
SPT = TSEG1 + 1

(b) 8 TQ ≤ DBT (data bit time) ≤ 25 TQ
DBT = TSEG1 + TSEG2 + 1TQ = TSEG2 + SPT

(c) 1 TQ ≤ 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 CANn bit rate register (CnBTR))
TSEG2[2:0] (Bits 10 to 8 of CANn bit rate register (CnBTR))

Table 13-29 shows the combinations of bit rates that satisfy the above conditions.

Table 13-29: Settable Bit Rate Combinations (1/3)

Valid Bit Rate Setting					CnBTR Register Setting Value		Sampling Point (Unit,%)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1[3:0]	TSEG2[2:0]	
25	1	8	8	8	1111	111	68.0
24	1	7	8	8	1110	111	66.7
24	1	9	7	7	1111	110	70.8
23	1	6	8	8	1101	111	65.2
23	1	8	7	7	1110	110	69.6
23	1	10	6	6	1111	101	73.9
22	1	5	8	8	1100	111	63.6
22	1	7	7	7	1101	110	68.2
22	1	9	6	6	1110	101	72.7
22	1	11	5	5	1111	100	77.3
21	1	4	8	8	1011	111	61.9
21	1	6	7	7	1100	110	66.7
21	1	8	6	6	1101	101	71.4
21	1	10	5	5	1110	100	76.2
21	1	12	4	4	1111	011	81.0
20	1	3	8	8	1010	111	60.0
20	1	5	7	7	1011	110	65.0
20	1	7	6	6	1100	101	70.0
20	1	9	5	5	1101	100	75.0
20	1	11	4	4	1110	011	80.0
20	1	13	3	3	1111	010	85.0
19	1	2	8	8	1001	111	57.9
19	1	4	7	7	1010	110	63.2
19	1	6	6	6	1011	101	68.4
19	1	8	5	5	1100	100	73.7
19	1	10	4	4	1101	011	78.9
19	1	12	3	3	1110	010	84.2
19	1	14	2	2	1111	001	89.5
18	1	1	8	8	1000	111	55.6
18	1	3	7	7	1001	110	61.1
18	1	5	6	6	1010	101	66.7
18	1	7	5	5	1011	100	72.2
18	1	9	4	4	1100	011	77.8
18	1	11	3	3	1101	010	83.3
18	1	13	2	2	1110	001	88.9
18	1	15	1	1	1111	000	94.4
17	1	2	7	7	1000	110	58.8

Table 13-29: Settable Bit Rate Combinations (2/3)

Valid Bit Rate Setting					CnBTR Register Setting Value		Sampling Point (Unit,%)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1[3:0]	TSEG2[2:0]	
17	1	4	6	6	1001	101	64.7
17	1	6	5	5	1010	100	70.6
17	1	8	4	4	1011	011	76.5
17	1	10	3	3	1100	010	82.4
17	1	12	2	2	1101	001	88.2
17	1	14	1	1	1110	000	94.1
16	1	1	7	7	0111	110	56.3
16	1	3	6	6	1000	101	62.5
16	1	5	5	5	1001	100	68.8
16	1	7	4	4	1010	011	75.0
16	1	9	3	3	1011	010	81.3
16	1	11	2	2	1100	001	87.5
16	1	13	1	1	1101	000	93.8
15	1	2	6	6	0111	101	60.0
15	1	4	5	5	1000	100	66.7
15	1	6	4	4	1001	011	73.3
15	1	8	3	3	1010	010	80.0
15	1	10	2	2	1011	001	86.7
15	1	12	1	1	1100	000	93.3
14	1	1	6	6	0110	101	57.1
14	1	3	5	5	0111	100	64.3
14	1	5	4	4	1000	011	71.4
14	1	7	3	3	1001	010	78.6
14	1	9	2	2	1010	001	85.7
14	1	11	1	1	1011	000	92.9
13	1	2	5	5	0110	100	61.5
13	1	4	4	4	0111	011	69.2
13	1	6	3	3	1000	010	76.9
13	1	8	2	2	1001	001	84.6
13	1	10	1	1	1010	000	92.3
12	1	1	5	5	0101	100	58.3
12	1	3	4	4	0110	011	66.7
12	1	5	3	3	0111	010	75.0
12	1	7	2	2	1000	001	83.3
12	1	9	1	1	1001	000	91.7
11	1	2	4	4	0101	011	63.6
11	1	4	3	3	0110	010	72.7
11	1	6	2	2	0111	001	81.8
11	1	8	1	1	1000	000	90.9

Table 13-29: Settable Bit Rate Combinations (3/3)

Valid Bit Rate Setting					CnBTR Register Setting Value		Sampling Point (Unit,%)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1[3:0]	TSEG2[2:0]	
10	1	1	4	4	0100	011	60.0
10	1	3	3	3	0101	010	70.0
10	1	5	2	2	0110	001	80.0
10	1	7	1	1	0111	000	90.0
9	1	2	3	3	0100	010	66.7
9	1	4	2	2	0101	001	77.8
9	1	6	1	1	0110	000	88.9
8	1	1	3	3	0011	010	62.5
8	1	3	2	2	0100	001	75.0
8	1	5	1	1	0101	000	87.5
7[Note]	1	2	2	2	0011	001	71.4
7[Note]	1	4	1	1	0100	000	85.7
6[Note]	1	1	2	2	0010	001	66.7
6[Note]	1	3	1	1	0011	000	83.3
5[Note]	1	2	1	1	0010	000	80.0
4[Note]	1	1	1	1	0001	000	75.0

Note: Setting with a DBT value of 7 or less is valid only when the value of the CnBRP register is other than 00H.

Caution: The values in Table 13-29 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

13.15.1 Representative examples of baud rate settings

Tables 13-30 and 13-31 show representative examples of baud rate setting.

Table 13-30: Representative Examples of Baud Rate Settings (f_{CANMOD} = 8 MHz) (1/2)

Set Baud Rate Value (Unit:kbps)	Division Ratio of CnBRP	CnBRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					CnBTR Register Setting Value		Sampling point (Unit:%)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1 [3:0]	TSEG2 [2:0]	
1000	1	00000000	8	1	1	3	3	0011	010	62.5
1000	1	00000000	8	1	3	2	2	0100	001	75.0
1000	1	00000000	8	1	5	1	1	0101	000	87.5
500	1	00000000	16	1	1	7	7	0111	110	56.3
500	1	00000000	16	1	3	6	6	1000	101	62.5
500	1	00000000	16	1	5	5	5	1001	100	68.8
500	1	00000000	16	1	7	4	4	1010	011	75.0
500	1	00000000	16	1	9	3	3	1011	010	81.3
500	1	00000000	16	1	11	2	2	1100	001	87.5
500	1	00000000	16	1	13	1	1	1101	000	93.8
500	2	00000001	8	1	1	3	3	0011	010	62.5
500	2	00000001	8	1	3	2	2	0100	001	75.0
500	2	00000001	8	1	5	1	1	0101	000	87.5
250	2	00000001	16	1	1	7	7	0111	110	56.3
250	2	00000001	16	1	3	6	6	1000	101	62.5
250	2	00000001	16	1	5	5	5	1001	100	68.8
250	2	00000001	16	1	7	4	4	1010	011	75.0
250	2	00000001	16	1	9	3	3	1011	010	81.3
250	2	00000001	16	1	11	2	2	1100	001	87.5
250	2	00000001	16	1	13	1	1	1101	000	93.8
250	4	00000011	8	1	3	2	2	0100	001	75.0
250	4	00000011	8	1	5	1	1	0101	000	87.5
125	4	00000011	16	1	1	7	7	0111	110	56.3
125	4	00000011	16	1	3	6	6	1000	101	62.5
125	4	00000011	16	1	5	5	5	1001	100	68.8
125	4	00000011	16	1	7	4	4	1010	011	75.0
125	4	00000011	16	1	9	3	3	1011	010	81.3
125	4	00000011	16	1	11	2	2	1100	001	87.5
125	4	00000011	16	1	13	1	1	1101	000	93.8
125	8	00000111	8	1	3	2	2	0100	001	75.0
125	8	00000111	8	1	5	1	1	0101	000	87.5
100	4	00000011	20	1	7	6	6	1100	101	70.0
100	4	00000011	20	1	9	5	5	1101	100	75.0
100	5	00000100	16	1	7	4	4	1010	011	75.0
100	5	00000100	16	1	9	3	3	1011	010	81.3

Table 13-30: Representative Examples of Baud Rate Settings (f_{CANMOD} = 8 MHz) (2/2)

Set Baud Rate Value (Unit:kbps)	Division Ratio of CnBRP	CnBRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					CnBTR Register Setting Value		Sampling point (Unit:%)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1 [3:0]	TSEG2 [2:0]	
100	8	00000111	10	1	3	3	3	0101	010	70.0
100	8	00000111	10	1	5	2	2	0110	001	80.0
100	10	00001001	8	1	3	2	2	0100	001	75.0
100	10	00001001	8	1	5	1	1	0101	000	87.5
83.3	4	00000011	24	1	7	8	8	1110	111	66.7
83.3	4	00000011	24	1	9	7	7	1111	110	70.8
83.3	6	00000101	16	1	5	5	5	1001	100	68.8
83.3	6	00000101	16	1	7	4	4	1010	011	75.0
83.3	6	00000101	16	1	9	3	3	1011	010	81.3
83.3	6	00000101	16	1	11	2	2	1100	001	87.5
83.3	8	00000111	12	1	5	3	3	0111	010	75.0
83.3	8	00000111	12	1	7	2	2	1000	001	83.3
83.3	12	00001011	8	1	3	2	2	0100	001	75.0
83.3	12	00001011	8	1	5	1	1	0101	000	87.5
33.3	10	00001001	24	1	7	8	8	1110	111	66.7
33.3	10	00001001	24	1	9	7	7	1111	110	70.8
33.3	12	00001011	20	1	7	6	6	1100	101	70.0
33.3	12	00001011	20	1	9	5	5	1101	100	75.0
33.3	15	00001110	16	1	7	4	4	1010	011	75.0
33.3	15	00001110	16	1	9	3	3	1011	010	81.3
33.3	16	00001111	15	1	6	4	4	1001	011	73.3
33.3	16	00001111	15	1	8	3	3	1010	010	80.0
33.3	20	00010011	12	1	5	3	3	0111	010	75.0
33.3	20	00010011	12	1	7	2	2	1000	001	83.3
33.3	24	00010111	10	1	3	3	3	0101	010	70.0
33.3	24	00010111	10	1	5	2	2	0110	001	80.0
33.3	30	00011101	8	1	3	2	2	0100	001	75.0
33.3	30	00011101	8	1	5	1	1	0101	000	87.5

Caution: The values in Table 13-30 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

Table 13-31: Representative Examples of Baud Rate Settings (f_{CANMOD} = 16 MHz) (1/2)

Set Baud Rate Value (Unit:kbps)	Division Ratio of CnBRP	CnBRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					CnBTR Register Setting Value		Sampling point (Unit:%)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1 [3:0]	TSEG2 [2:0]	
1000	1	00000000	16	1	1	7	7	0111	110	56.3
1000	1	00000000	16	1	3	6	6	1000	101	62.5
1000	1	00000000	16	1	5	5	5	1001	100	68.8
1000	1	00000000	16	1	7	4	4	1010	011	75.0
1000	1	00000000	16	1	9	3	3	1011	010	81.3
1000	1	00000000	16	1	11	2	2	1100	001	87.5
1000	1	00000000	16	1	13	1	1	1101	000	93.8
1000	2	00000001	8	1	3	2	2	0100	001	75.0
1000	2	00000001	8	1	5	1	1	0101	000	87.5
500	2	00000001	16	1	1	7	7	0111	110	56.3
500	2	00000001	16	1	3	6	6	1000	101	62.5
500	2	00000001	16	1	5	5	5	1001	100	68.8
500	2	00000001	16	1	7	4	4	1010	011	75.0
500	2	00000001	16	1	9	3	3	1011	010	81.3
500	2	00000001	16	1	11	2	2	1100	001	87.5
500	2	00000001	16	1	13	1	1	1101	000	93.8
500	4	00000011	8	1	3	2	2	0100	001	75.0
500	4	00000011	8	1	5	1	1	0101	000	87.5
250	4	00000011	16	1	3	6	6	1000	101	62.5
250	4	00000011	16	1	5	5	5	1001	100	68.8
250	4	00000011	16	1	7	4	4	1010	011	75.0
250	4	00000011	16	1	9	3	3	1011	010	81.3
250	4	00000011	16	1	11	2	2	1100	001	87.5
250	8	00000111	8	1	3	2	2	0100	001	75.0
250	8	00000111	8	1	5	1	1	0101	000	87.5
125	8	00000111	16	1	3	6	6	1000	101	62.5
125	8	00000111	16	1	7	4	4	1010	011	75.0
125	8	00000111	16	1	9	3	3	1011	010	81.3
125	8	00000111	16	1	11	2	2	1100	001	87.5
125	16	00001111	8	1	3	2	2	0100	001	75.0
125	16	00001111	8	1	5	1	1	0101	000	87.5
100	8	00000111	20	1	9	5	5	1101	100	75.0
100	8	00000111	20	1	11	4	4	1110	011	80.0
100	10	00001001	16	1	7	4	4	1010	011	75.0
100	10	00001001	16	1	9	3	3	1011	010	81.3
100	16	00001111	10	1	3	3	3	0101	010	70.0
100	16	00001111	10	1	5	2	2	0110	001	80.0
100	20	00010011	8	1	3	2	2	0100	001	75.0

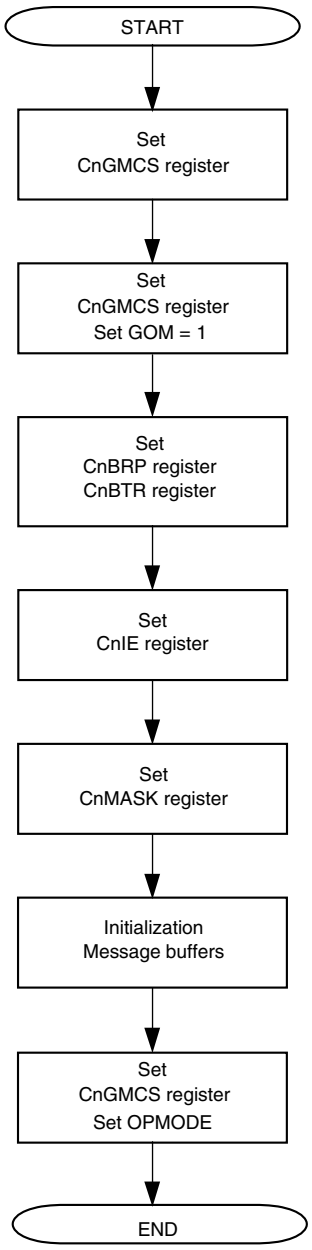
Table 13-31: Representative Examples of Baud Rate Settings (f_{CANMOD} = 16 MHz) (2/2)

Set Baud Rate Value (Unit:kbps)	Division Ratio of CnBRP	CnBRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					CnBTR Register Setting Value		Sampling point (Unit:%)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT 1	PHASE SEGMENT 2	TSEG1 [3:0]	TSEG2 [2:0]	
83.3	8	00000111	24	1	7	8	8	1110	111	66.7
83.3	8	00000111	24	1	9	7	7	1111	110	70.8
83.3	12	00001011	16	1	7	4	4	1010	011	75.0
83.3	12	00001011	16	1	9	3	3	1011	010	81.3
83.3	12	00001011	16	1	11	2	2	1100	001	87.5
83.3	16	00001111	12	1	5	3	3	0111	010	75.0
83.3	16	00001111	12	1	7	2	2	1000	001	83.3
83.3	24	00010111	8	1	3	2	2	0100	001	75.0
83.3	24	00010111	8	1	5	1	1	0101	000	87.5
33.3	30	00011101	24	1	7	8	8	1110	111	66.7
33.3	30	00011101	24	1	9	7	7	1111	110	70.8
33.3	24	00010111	20	1	9	5	5	1101	100	75.0
33.3	24	00010111	20	1	11	4	4	1110	011	80.0
33.3	30	00011101	16	1	7	4	4	1010	011	75.0
33.3	30	00011101	16	1	9	3	3	1011	010	81.3
33.3	32	00011111	15	1	8	3	3	1010	010	80.0
33.3	32	00011111	15	1	10	2	2	1011	001	86.7
33.3	37	00100100	13	1	6	3	3	1000	010	76.9
33.3	37	00100100	13	1	8	2	2	1001	001	84.6
33.3	40	00100111	12	1	5	3	3	0111	010	75.0
33.3	40	00100111	12	1	7	2	2	1000	001	83.3
33.3	48	00101111	10	1	3	3	3	0101	010	70.0
33.3	48	00101111	10	1	5	2	2	0110	001	80.0
33.3	60	00111011	8	1	3	2	2	0100	001	75.0
33.3	60	00111011	8	1	5	1	1	0101	000	87.5

Caution: The values in Table 13-31 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

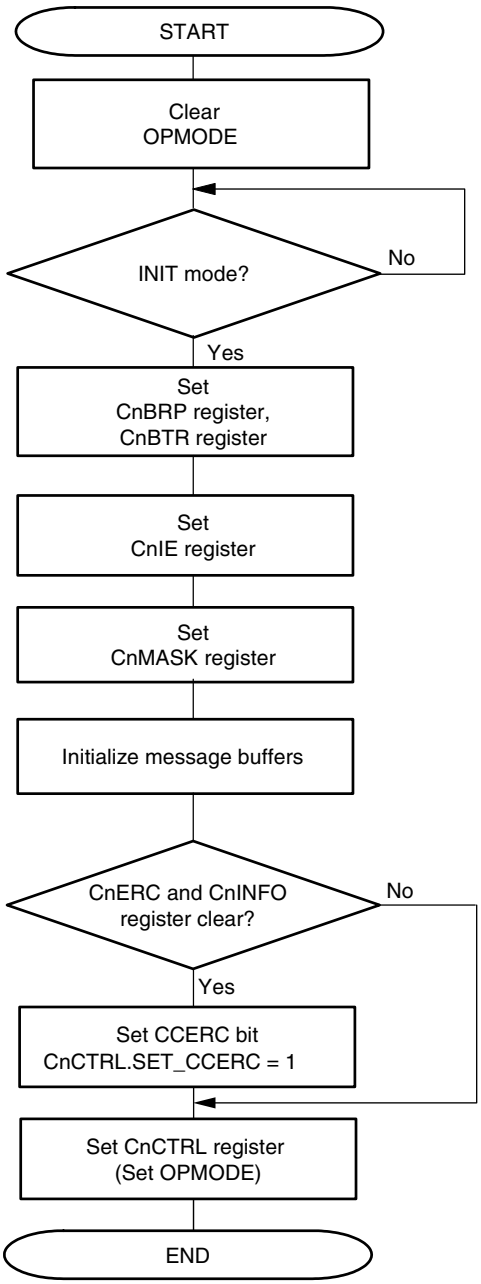
13.16 Operation of CAN Controller

Figure 13-61: Initialization



Remark: OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

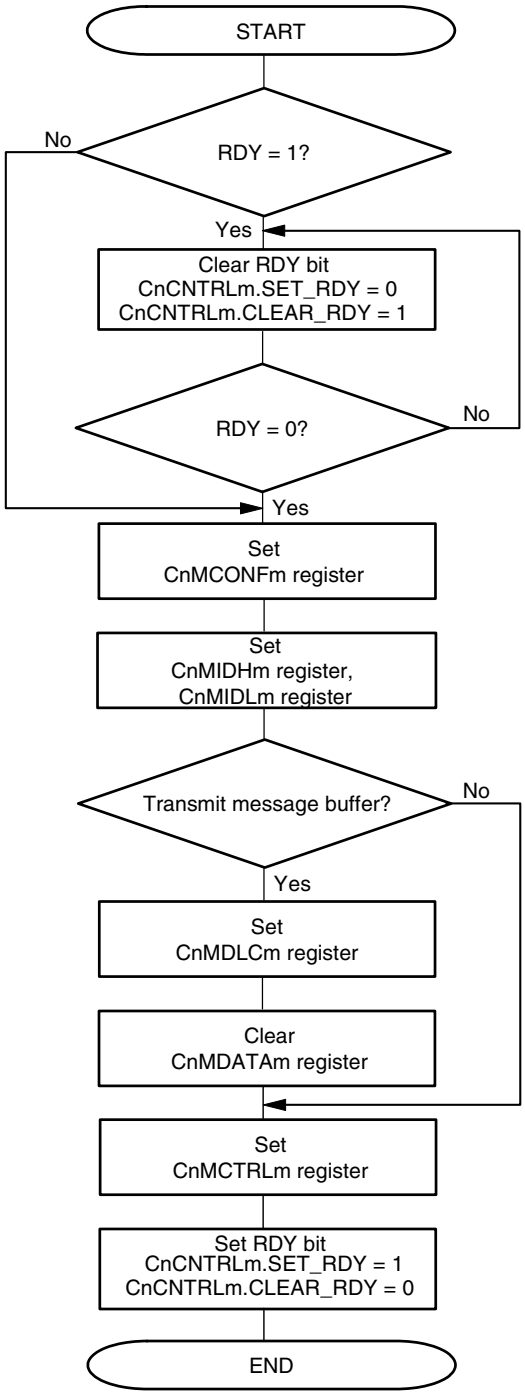
Figure 13-62: Re-initialization



Caution: After setting the CAN module to the initialization mode, avoid setting the module to another operation mode immediately after. If it is necessary to immediately set the module to another operation mode, be sure to access registers other than the CnCTRL and CnGMCTRL registers (e.g. set a message buffer).

Remark: OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

Figure 13-63: Message Buffer Initialization

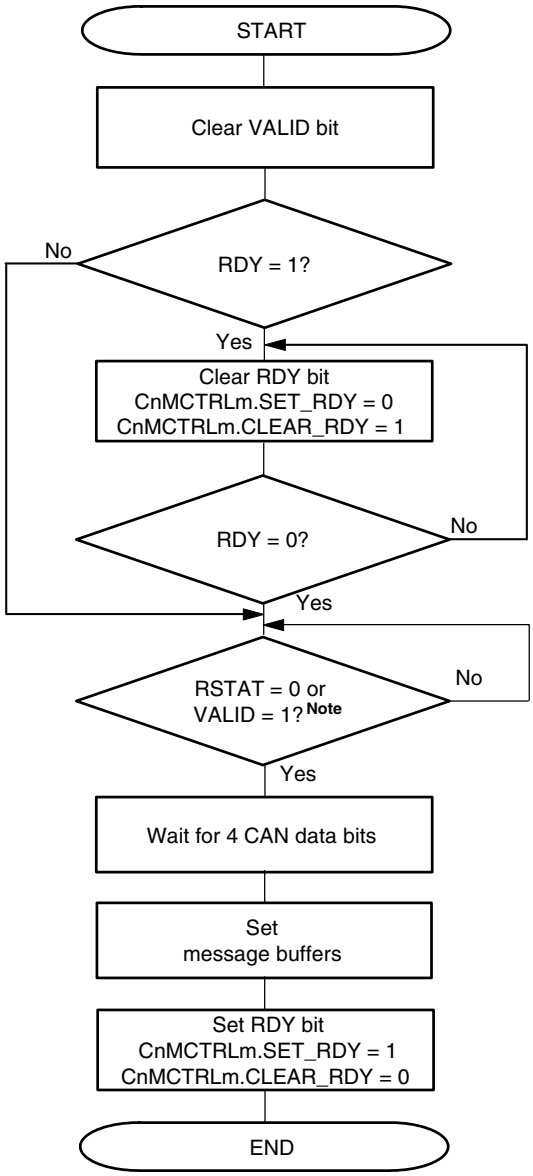


Cautions: 1. Before a message buffer is initialized, the RDY bit must be cleared.

2. Make the following settings for message buffers not used by the application:
- Clear the RDY, TRQ, and DN bits of the CnMCTRLm register to 0.
- Clear the MA0 bit of the CnMCONFm register to 0.

Figure 13-64 shows the processing for a receive message buffer (MT[2:0] bits of CnMCONFm register = 001B to 101B).

Figure 13-64: Message Buffer Redefinition



Note: Confirm that a message is being received because RDY bit must be set after a message is completely received.

Figure 13-65 shows the processing for a transmit message buffer during transmission (MT[2:0] bits of CnMCONFm register = 00B)

Figure 13-65: Message Buffer Redefinition during Transmission

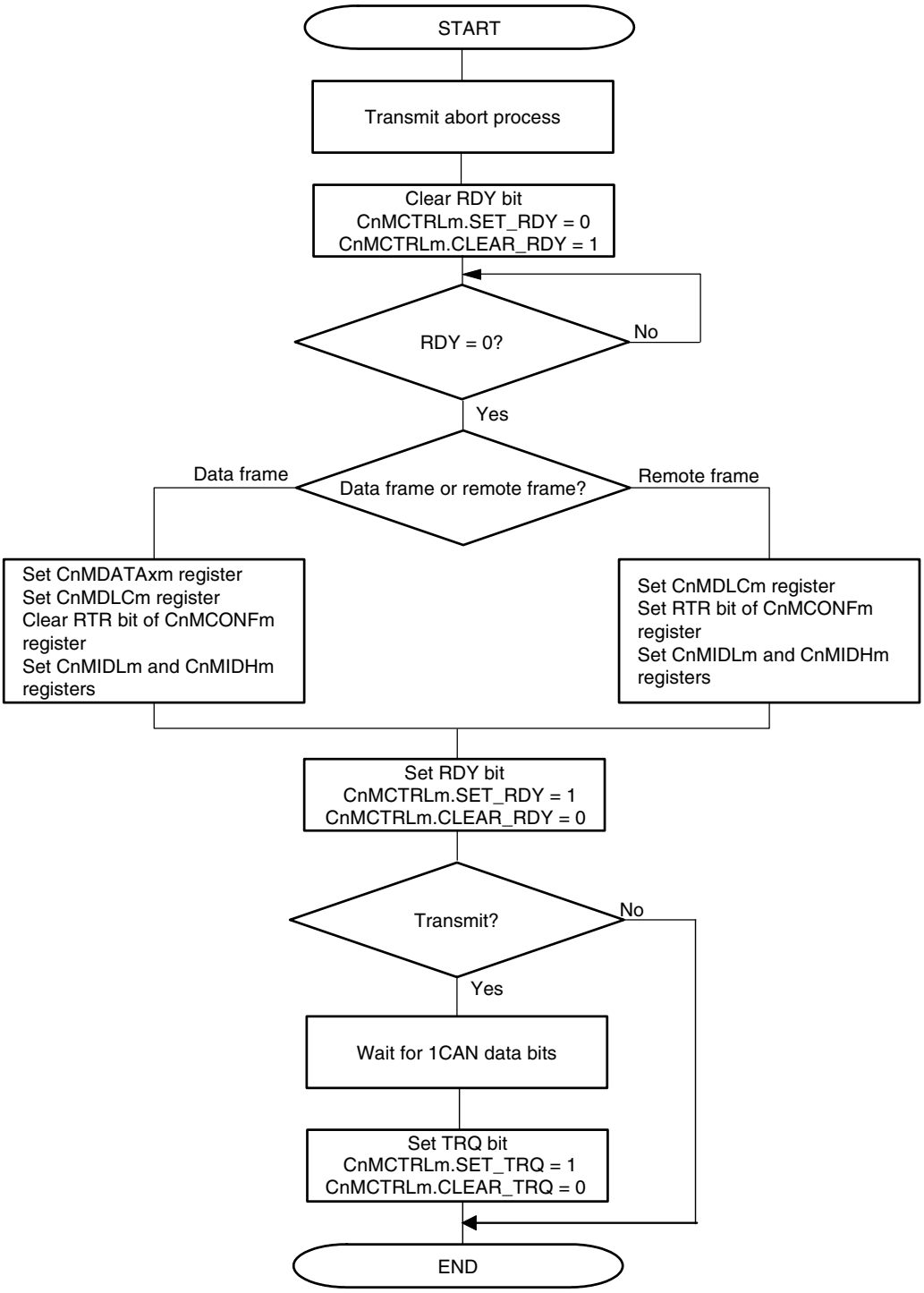
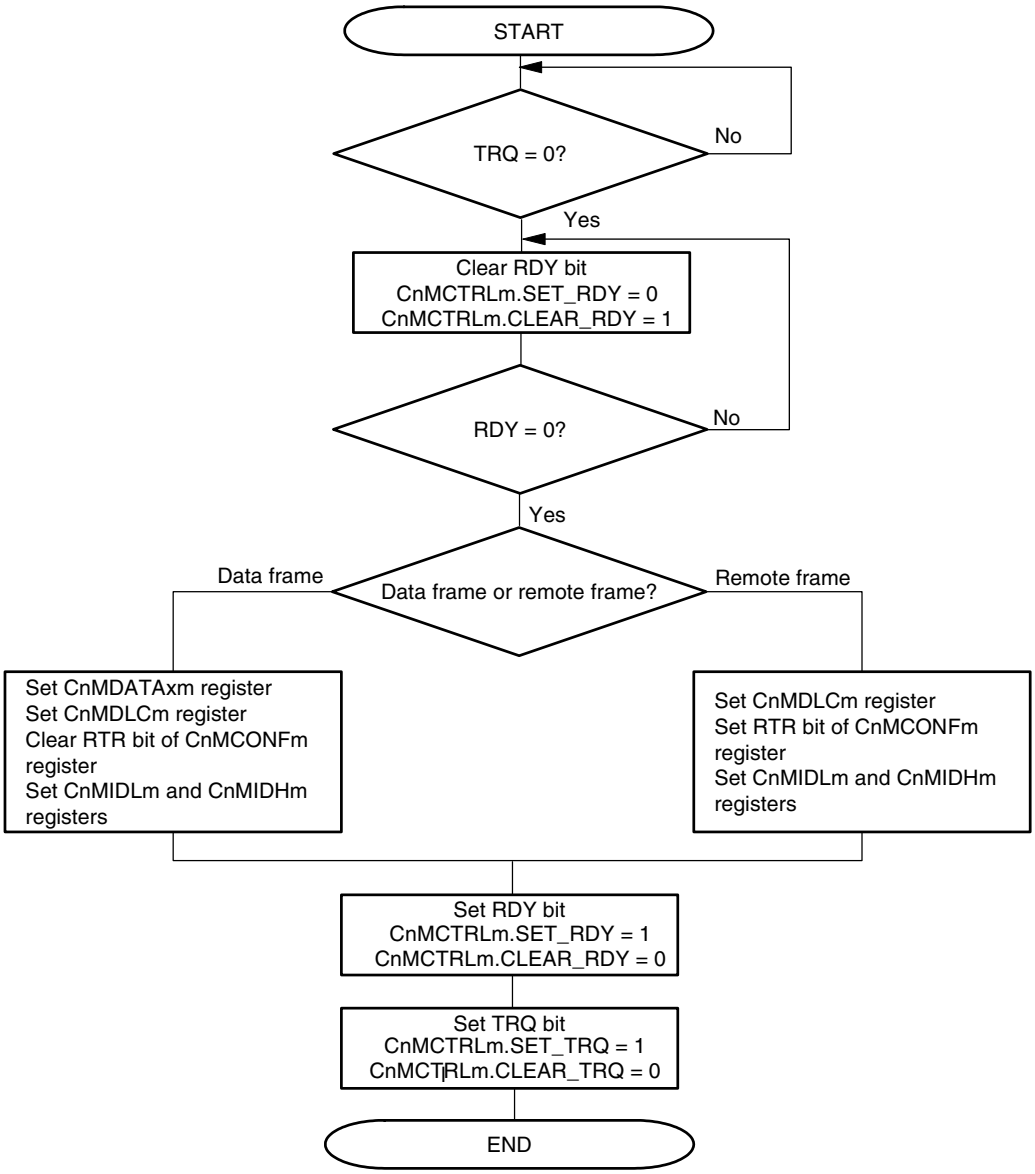


Figure 13-66 shows the processing for a transmit message buffer (MT[2:0] bits of CnMCONFm register = 000B).

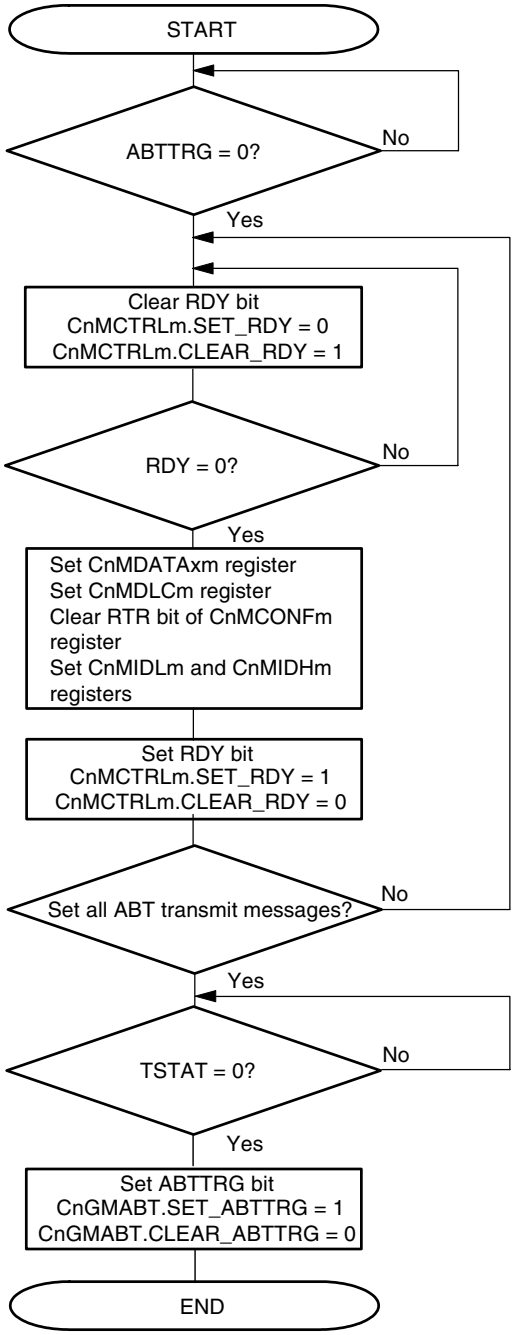
Figure 13-66: Message Transmit Processing



Caution: The TRQ bit should be set after the RDY bit is set.
The RDY bit and TRQ bit should not be set at the same time.

Figure 13-67 shows the processing for a transmit message buffer (MT[2:0] bits of CnMCONFm register = 000B).

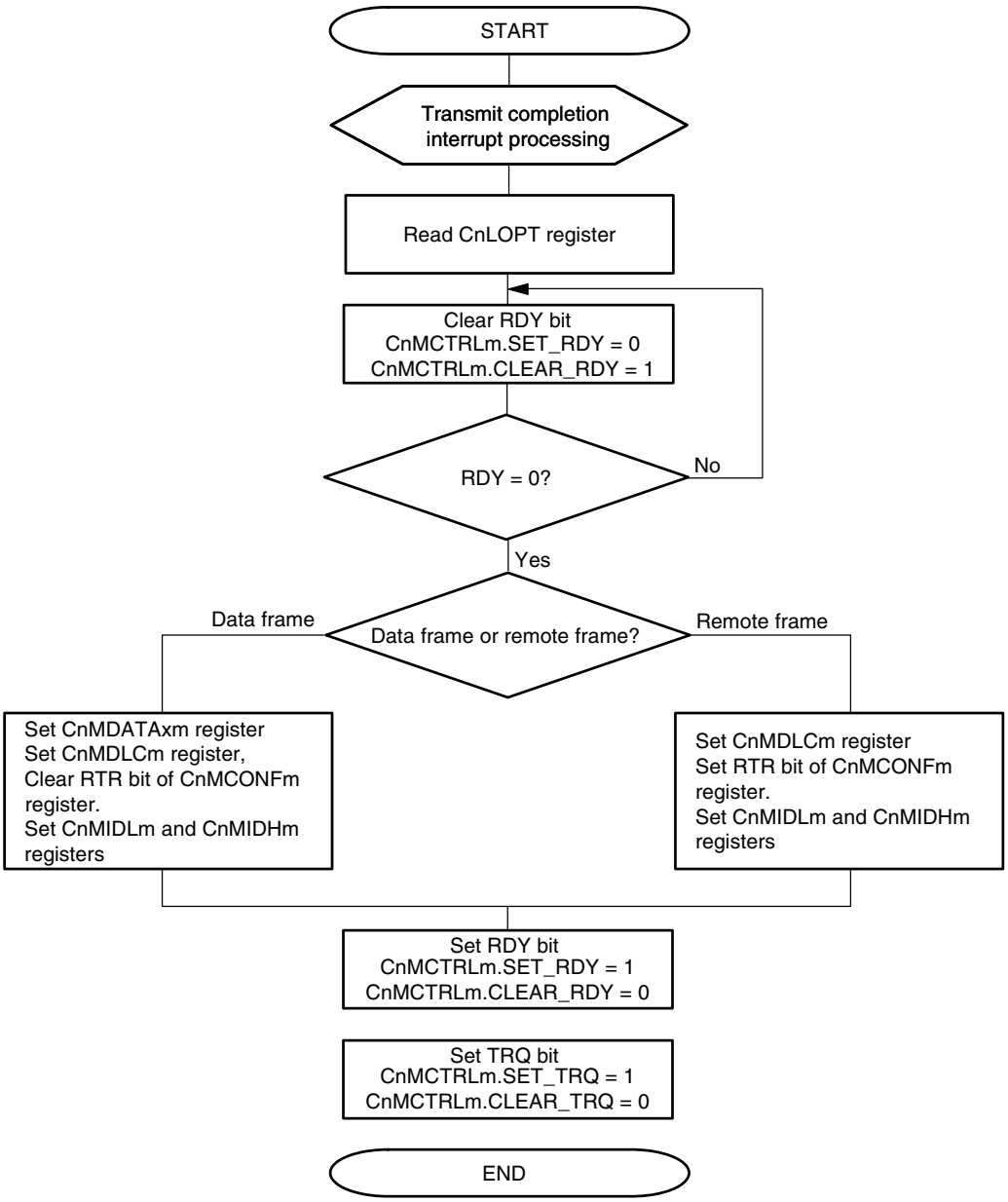
Figure 13-67: Message Transmit Processing (Normal Operation Mode with ABT)



Remark: This processing (normal operation mode with ABS) can only be applied to message buffers 0 to 7. For message buffers other than the ABT message buffers, refer to Figure 13-66.

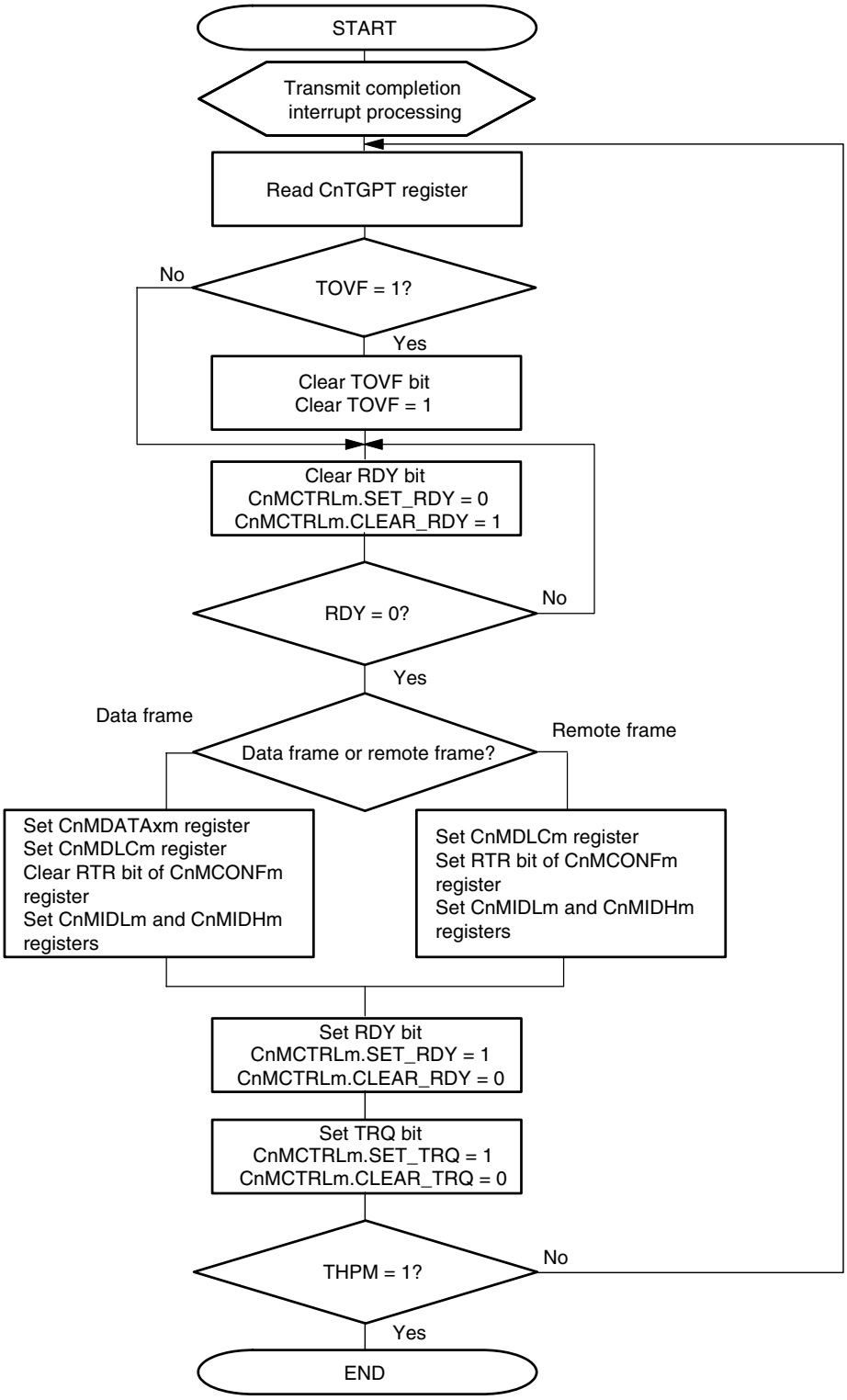
Caution: The ABTTRG bit should be set to 1 after the TSTAT bit is cleared to 0. Checking the TSTAT bit and setting the ABTTRG bit to 1 must be processed continuously.

Figure 13-68: Transmission via Interrupt (Using CnLOPT register)



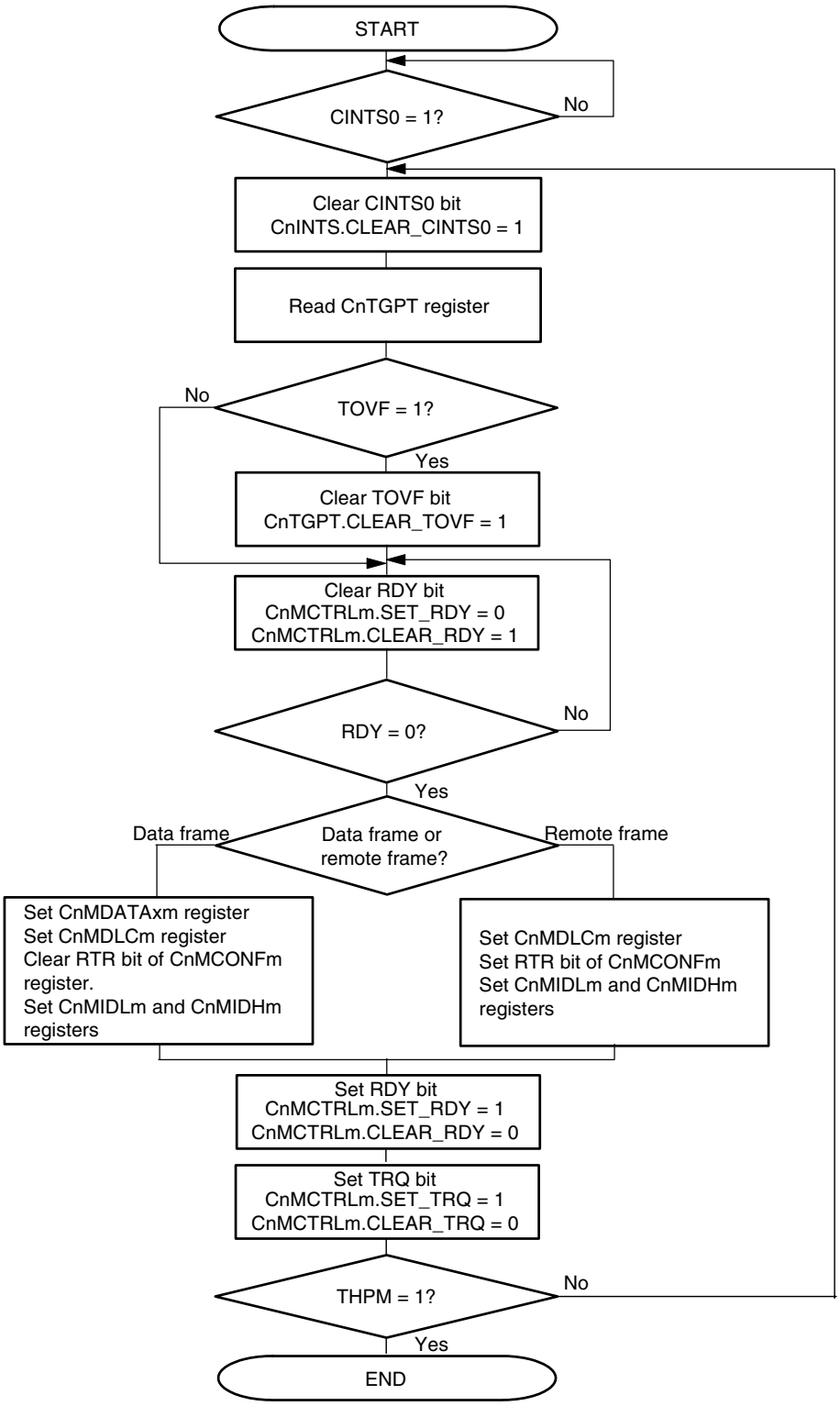
Caution: The TRQ bit should be set after the RDY bit is set.
The RDY bit and TRQ bit should not be set at the same time.

Figure 13-69: Transmit via Interrupt (Using CnTGPT register)



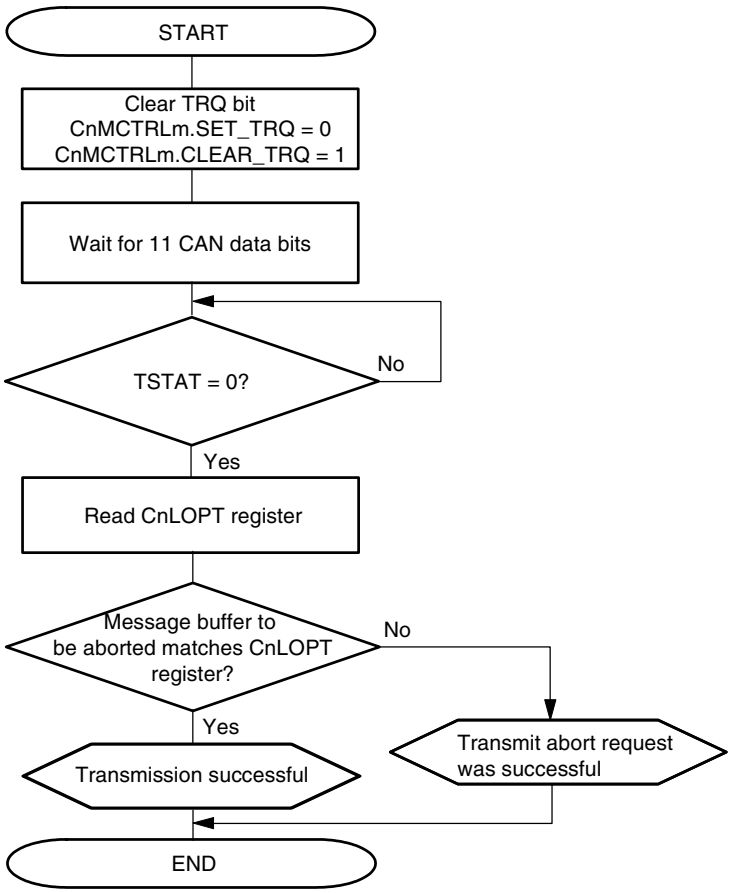
Caution: The TRQ bit should be set after the RDY bit is set.
The RDY bit and TRQ bit should not be set at the same time

Figure 13-70: Transmission via Software Polling



Caution: The TRQ bit should be set after the RDY bit is set.
The RDY bit and TRQ bit should not be set at the same time.

Figure 13-71: Transmission Abort Processing (Except Normal Operation Mode with ABT)



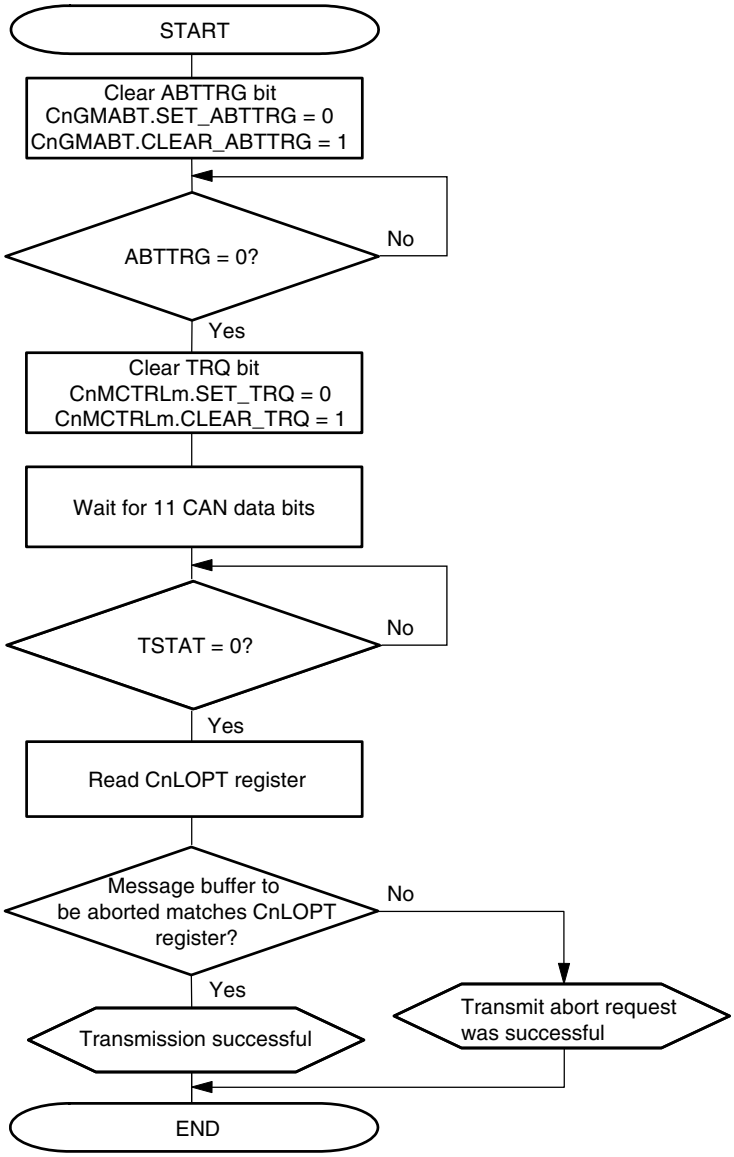
Cautions: 1. Execute transmission request abort processing by clearing the TRQ bit, not the RDY bit.

2. Before making a Sleep mode transition request, confirm that there is no transmission request left using this processing.

3. The TSTAT bit can be periodically checked by an user application or can be checked after the transmit completion interrupt.

4. Do not execute the new transmission request including in the other message buffers while transmission abort processing is in progress.

Figure 13-72: Transmission Abort Processing Except for ABT Transmission (Normal Operation Mode with ABT)



Cautions: 1. Execute transmission request abort processing by clearing the TRQ bit, not the RDY bit.

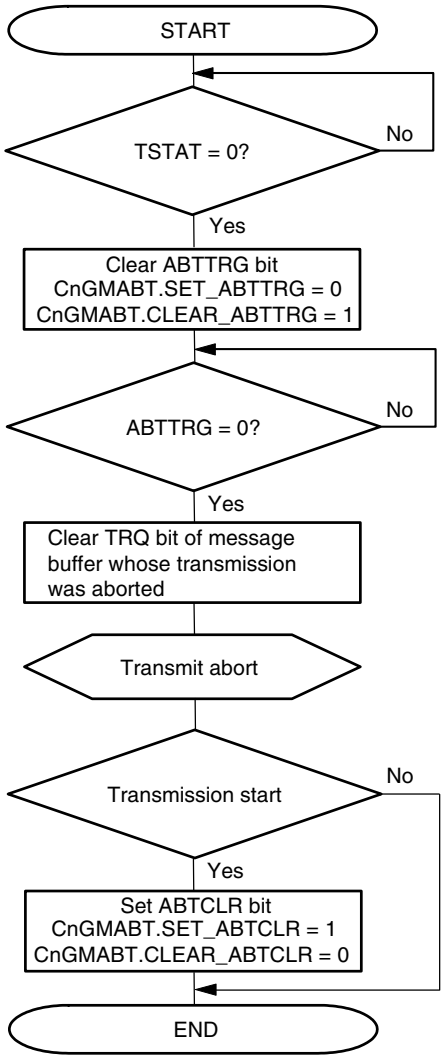
2. Before making a Sleep mode transition request, confirm that there is no transmission request left using this processing.

3. The TSTAT bit can be periodically checked by an user application or can be checked after the transmit completion interrupt.

4. Do not execute the new transmission request including in the other message buffers while transmission abort processing is in progress.

Figure 13-73 shows the processing not to skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.

Figure 13-73: ABT Transmission Abort Processing (Normal Operation Mode with ABT)

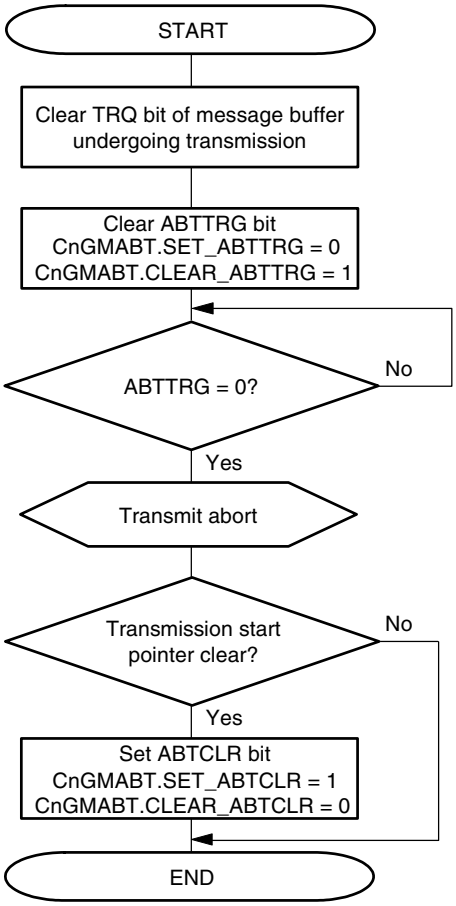


Cautions: 1. Do not set any transmission requests while ABT transmission abort processing is in progress.

2. Make a CAN Sleep mode/CAN Stop mode transition request after ABTTRG is cleared (after ABT mode is aborted) following the procedure shown in Figure 13-73 or 13-74. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in Figure 13-71.

Figure 13-74 shows the processing to skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.

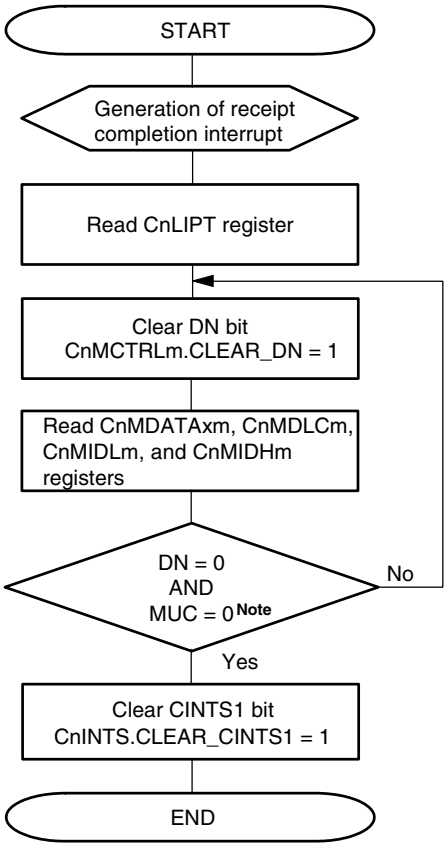
Figure 13-74: ABT Transmission Request Abort Processing (Normal Operation Mode with ABT)



Cautions: 1. Do not set any transmission requests while ABT transmission abort processing is in progress.

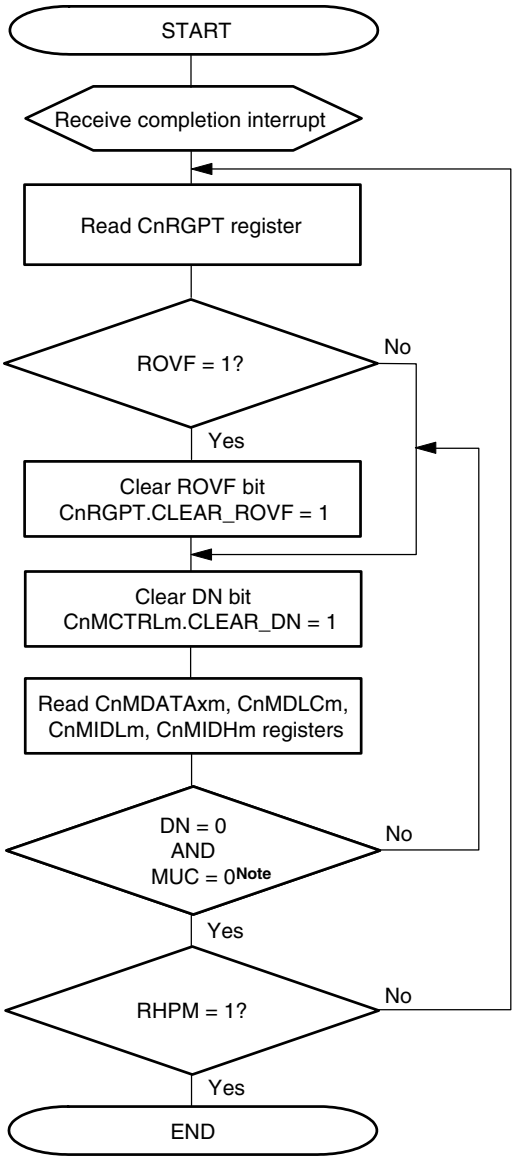
2. Make a CAN Sleep mode/CAN Stop mode request after ABTTRG is cleared (after ABT mode is stopped) following the procedure shown in Figures 13-73 or 13-74. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in Figure 13-71.

Figure 13-75: Reception via Interrupt (Using CnLIPT Register)



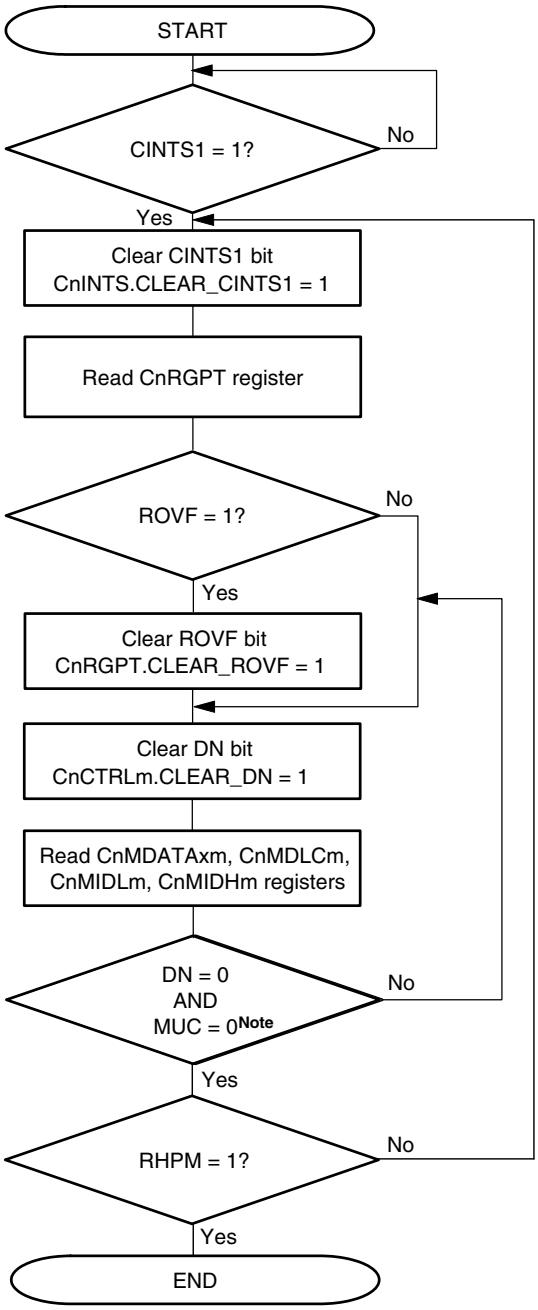
Note: Check the MUC and DN bits using one read access.

Figure 13-76: Reception via Interrupt (Using CnRGPT Register)



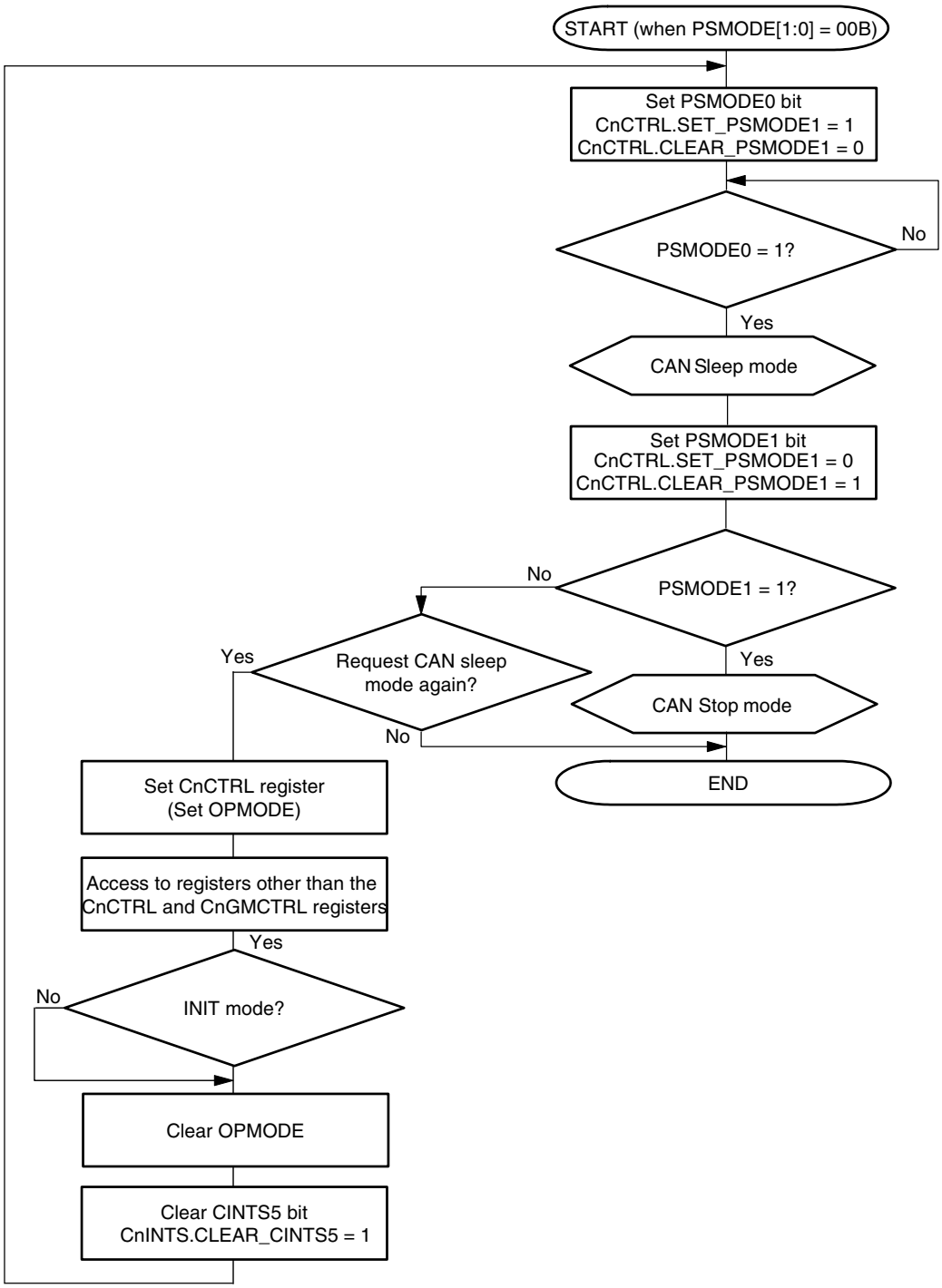
Note: Check the MUC and DN bits using one read access

Figure 13-77: Reception via Software Polling



Note: Check the MUC and DN bits using one read access

Figure 13-78: Setting CAN Sleep Mode/Stop Mode



Cautions: 1. To abort transmission before making a request for the CAN Sleep mode, perform processing according to Figures 13-71 and 13-72.

2. If the host CPU wants to enter a power save mode as well, the interrupt processing needs to be disabled before the CPU validates that Sleep mode has been entered. If the interrupt processing can not be disabled, the host CPU will never wakeup by CAN bus activity when the CAN Sleep mode is released between validation of the Sleep state and execution of the i.e. CPU HALT instruction.

Figure 13-79: Clear CAN Sleep/Stop Mode

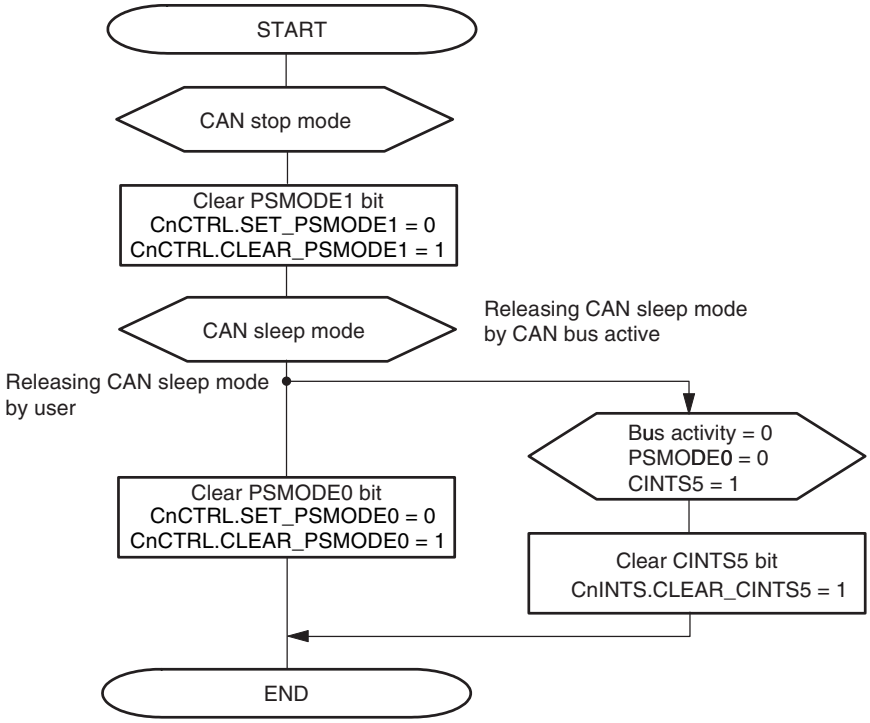


Figure 13-80: Bus-Off Recovery

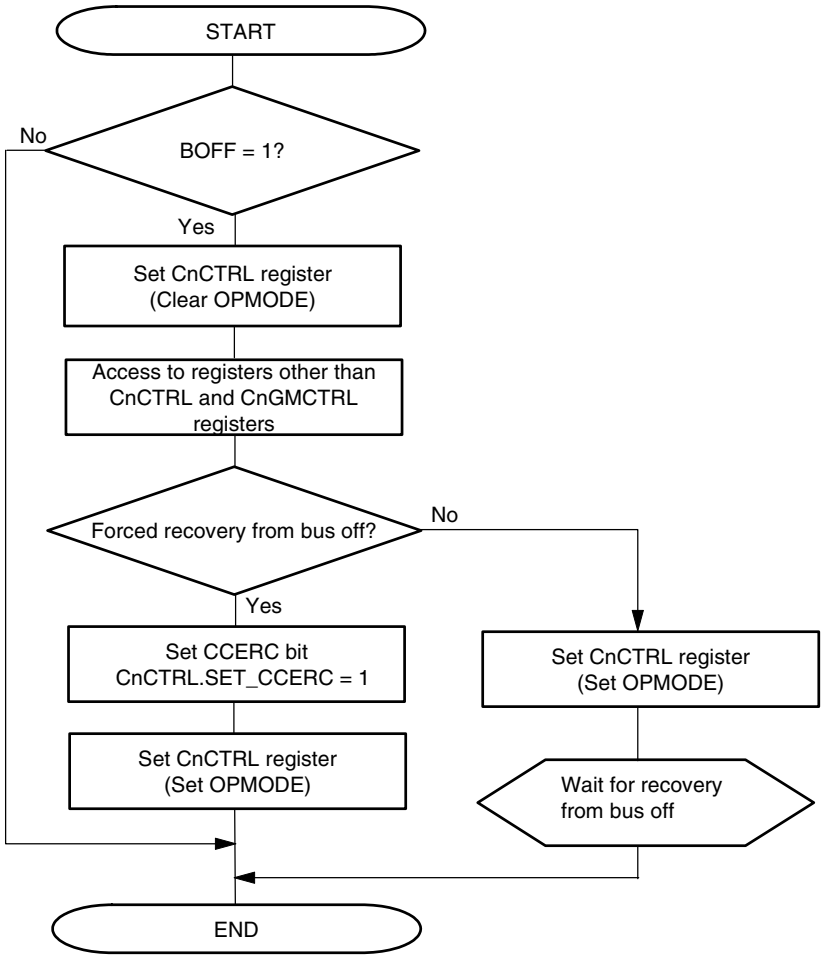


Figure 13-81: Normal Shutdown Process

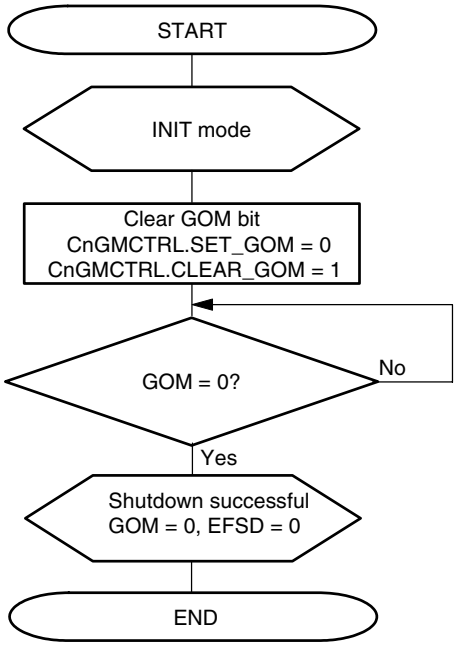
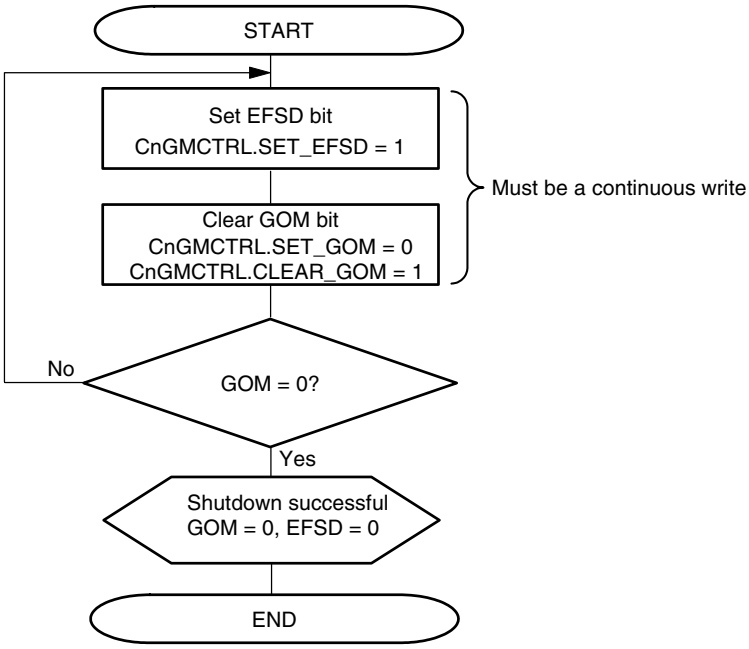


Figure 13-82: Forced Shutdown Process



Caution: Do not read- or write-access any registers by software between setting the EFSD bit and clearing the GOM bit.

Remark: OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

Figure 13-83: Error Handling

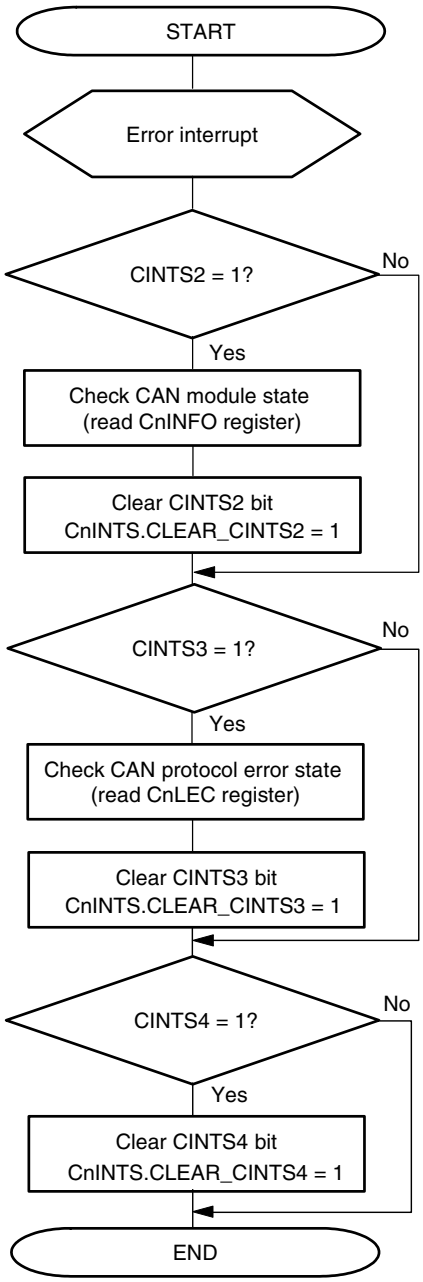


Figure 13-84: Setting CPU Standby (from CAN Sleep Mode)

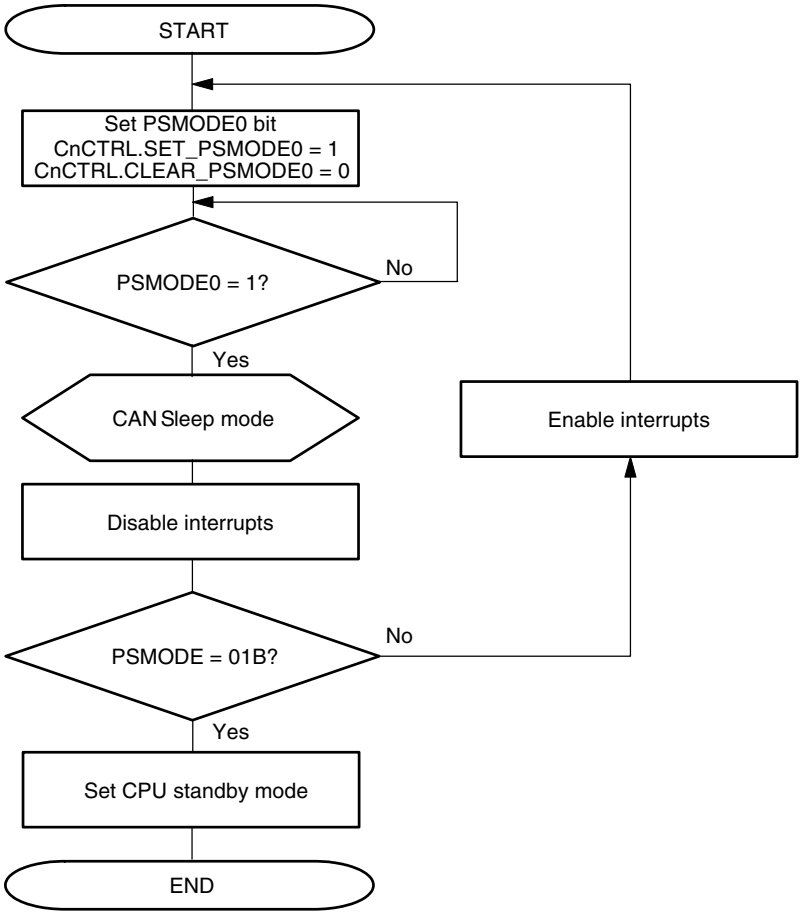
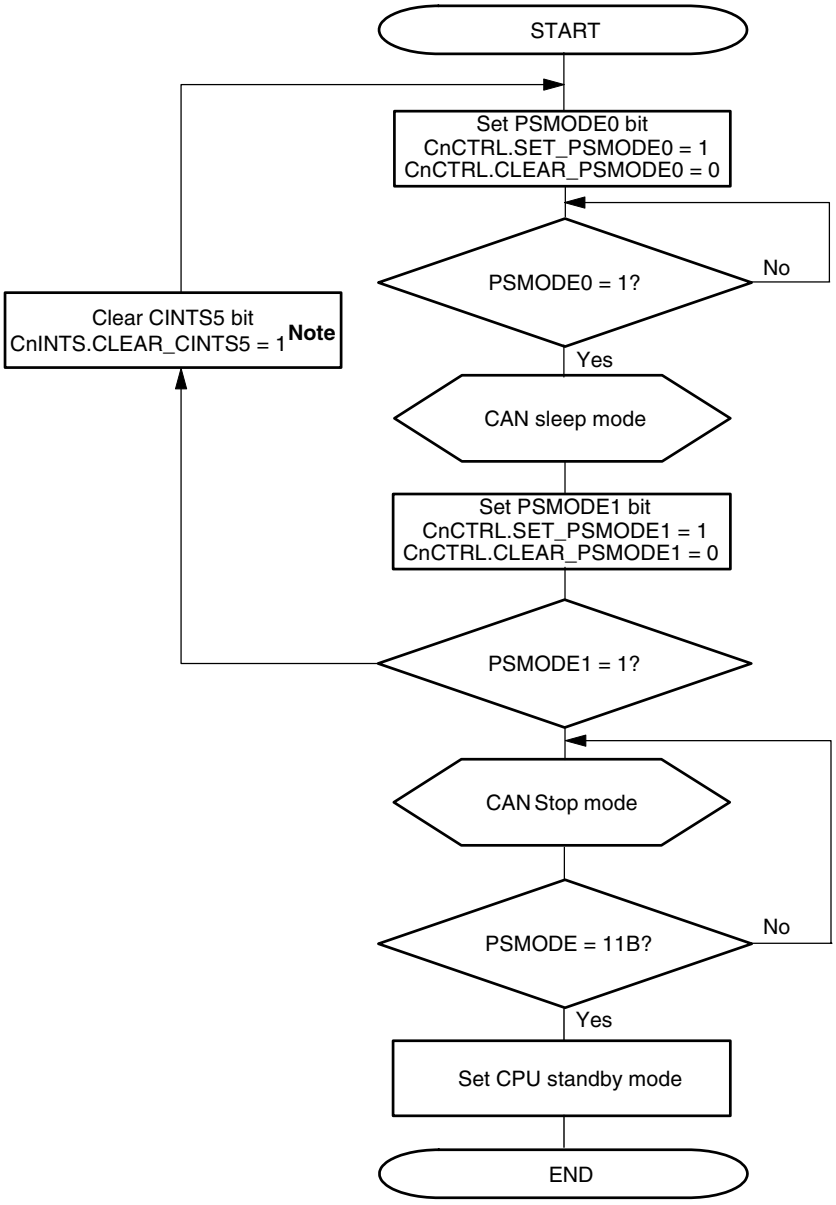


Figure 13-85: Setting CPU Standby (from CAN Stop Mode)



Note: During wakeup interrupts

Caution: The CAN Stop mode can only be released by writing 01B to the PSMODE[1:0] bit of the CnCTRL register and not by a change in the CAN bus state.

[MEMO]

Chapter 14 DAFCAN

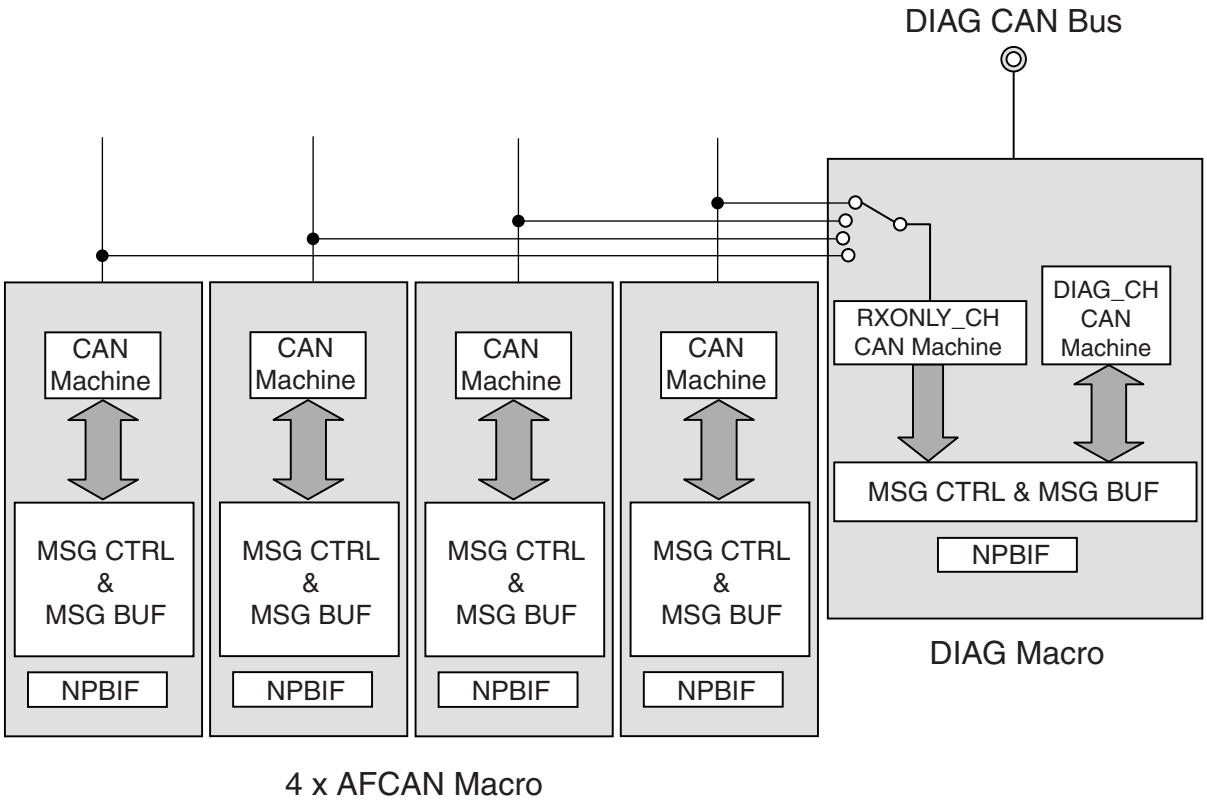
14.1 Introduction

This DAFCAN (Diagnostic AFCAN) macro is in first place an ordinary single channel AFCAN macro with 48 message buffers. In difference to the single channel AFCAN macro the DAFCAN macro features two CAN channels. The channel, which can be operated as an ordinary CAN interface, is called diagnostic channel (DIAG-CH).

For specific diagnostic purposes of the application a second CAN interface is available. This second channel has only limited operational modes because its main purpose is to listen only to the data traffic of one of the other single channel AFCAN macros. This channel is called listen-only channel (RXONLY-CH). It provides the mirror function, which transfers the messages from the source CAN bus (monitored channel) to the diagnostic channel from where these messages are automatically sent onto the CAN bus. Furthermore, it provides the mirror mode with the Transfer ID Filter function (w/ TIF). This mode can transmit particular ID only from the source CAN channel to destination channel automatically.

The mirror function as part of a diagnosis concept including other CAN channels is shown in Figure 14-1.

Figure 14-1: Diagnosis concept using DAFCAN and 4 other CAN channels



The RXONLY-CH and the DIAG-CH have independent control registers for power save (PSMODE) and operational (OPMODE) modes. The combination of both affects the message buffer allocation for the RXONLY-CH. When RXONLY-CH is in PSMODE == STOP or in OPMODE == INIT, the upper 16 message buffer are assigned to the DIAG-CH. This offers the opportunity to use the DIAG macro as an ordinary CAN channel with 48 message buffers when monitoring of other CAN channels is not necessary.

14.2 Overview of Functions

Table 14-1: Outline of DIAG Macro Functions

Feature	Details
Protocol	Active support of extended frame format (international standard ISO11898)
Channels	RXONLY-CH, DIAG-CH
Baud rate	Max. 1Mbit/s @ $f_{CAN} \geq 16$ MHz
Message Storage	RAM area with shared access (Accessing entities: CPU, RXONLY-CH CAN, DIAG-CH CAN) The DIAG macro features 48 message buffers. The upper 16 buffer are assigned to the RXONLY-CH depending on operational and power save mode.
Message Organisation	Each message buffer can be initialised either to be a transmit message buffer or a receive message buffer. A group of message buffers assigned as receive buffer can form a multi-buffer receive block. A group of message buffer assigned as transmit buffer can be used for automatic block transfer.
Masks (DIAG-CH only)	The DIAG-CH features 4 masks. Each mask can be assigned to each message buffer. There is no difference between global and local masks within one CAN module. Mirror mode does not provide any masks. All messages are copied to DIAG_CH. Mirror mode w/ TIF of the RXONLY-CH provides 8 reference transfer ID register (CnTIDRnL/H) and 1 mask transfer ID register (CnTIDML/H) for filtering. Only message matching the filter criteria are stored in the upper 16 message buffers. Only those messages stored here participate in the Mirror Mode.
Application Support for Message Handling	The DIAG-CH provides a fast mechanism to find receive message buffers with updated content (Reception History List). The DIAG-CH provides a fast mechanism to find the transmit message buffers, from which a message frame has been sent (Transmission History List). The RXONLY-CH does not provide these history lists.
Remote Frame Support	Remote frame handling by message buffer defined for transmit
Time Stamp Functions **Note 1** (DIAG-CH only)	Time stamp upon message reception Trigger for time stamp capture selectable (SOF or EOF detection in a CAN message frame) Append time stamp on transmission (specific bytes in the data field are replaced by the captured time stamp) **Note 2**
Diagnostics	The DIAG macro features a mirror function. Messages received by RXONLY-CH are automatically send on the DIAG-CH. The RXONLY-CH input can be switched to any of the other CAN channels of the device. Readable error counters Valid protocol activity flag for verification of bus connection “Receive-only Mode” CAN protocol error type decoding “Self-test Mode” (DIAG-CH only)
Power Save Modes	Sleep Mode: Wake up from CAN bus Stop Mode: No wake up from CAN bus The RXONLY-CH CAN I/F module and the DIAG-CH CAN I/F module can be configured independently to all these power save modes respectively. The whole macro consumes minimum power when both CAN I/F modules are in such modes simultaneously.

Notes: 1. The architecture of the DIAG macro is prepared for the integration of a TTCAN Module, but not all DIAG macro derivatives are equipped with a TTCAN Module.

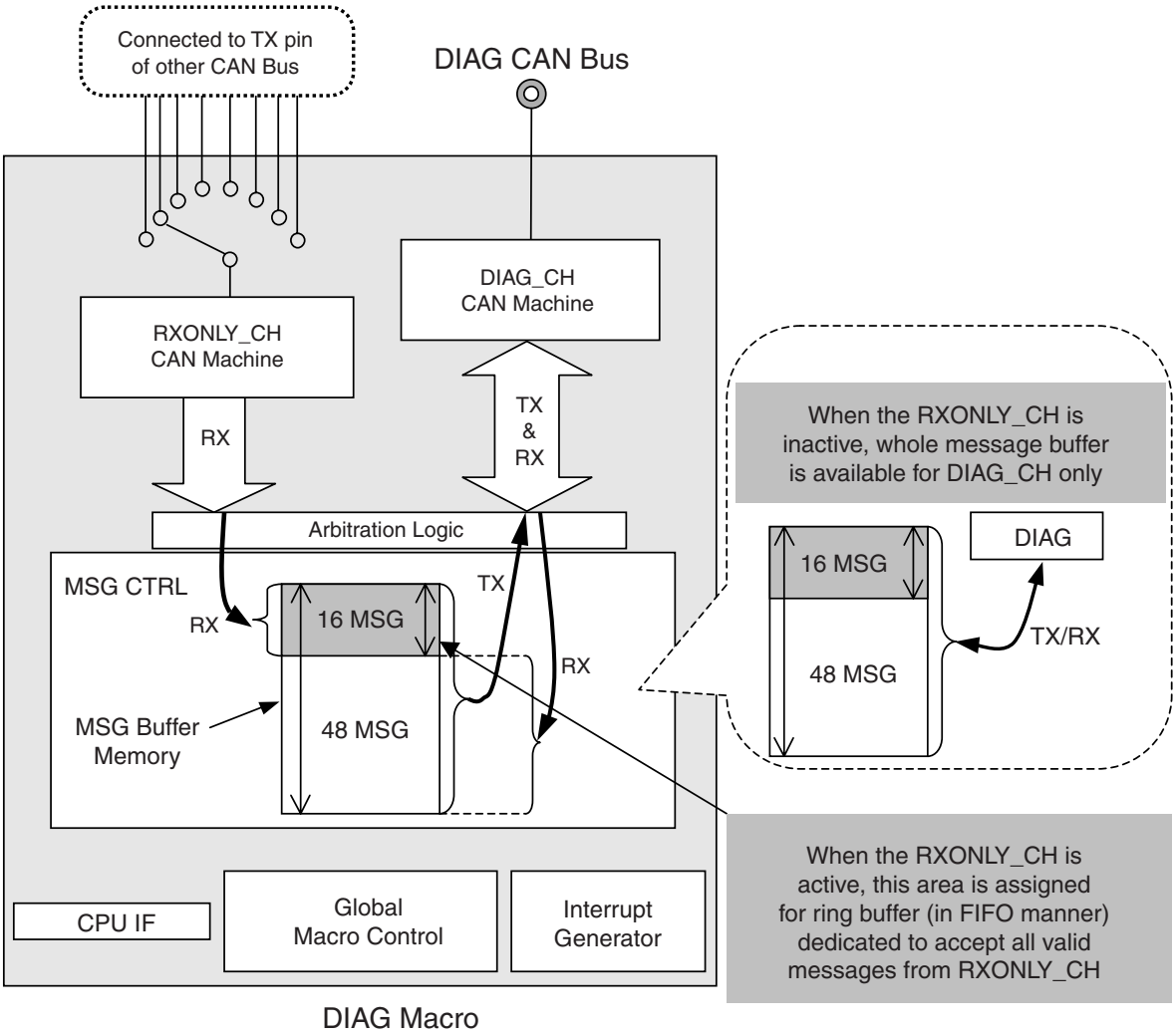
2. The 'Append time stamp on transmission' function is offered only in AFCAN/DIAG macro derivatives, which are equipped with the Advanced Time Stamp function.

14.3 Architecture

The DIAG macro is connected to a host CPU and provides two CAN bus interfaces. The DIAG-CH CAN I/F module is dedicated for a particular CAN bus, the diagnostic CAN bus. The RXONLY-CH CAN I/F module can be connected to several CAN buses via a programmable selector. The selector is located within the DIAG macro and provides connectivity for up to 8 sources.

The figure below shows an example of DIAG macro with 8 inputs for other CAN buses and 48 message buffers.

Figure 14-2: Architecture of DIAG Macro



The DIAG macro contains several sub-blocks that are described in the following chapters.

14.3.1 CPU I/F

The CPU I/F is the interface of the entire macro to the peripheral bus of the microcomputer system. The CPU I/F enables the CPU to access the control and status registers of all sub-blocks within the macro directly. The addresses of the control and status registers are mapped into the memory map of the microcomputer system.
The interrupt signals from the CAN Interrupt Generator to the Interrupt Controller of the microcomputer system are routed via the CPU I/F.
The macro clock f_{CAN} is connected to a peripheral macro clock of the clock generator of the microcomputer system.

14.3.2 Global Macro Control

The Global Macro Control sub-block contains the logic that enables or disables the entire macro. As well the macro clock f_{CANMOD} is set here.

14.3.3 CAN Interrupt Generator

In the CAN Interrupt Generator all interrupt request signals from the CAN I/F channels are collected. Appropriated signal shaping is applied to those interrupt request signals for a proper connection to the interrupt request signal inputs of the interrupt controller in the microcomputer system (i.e. pulse generation).

14.3.4 Message Control (MSG Ctrl)

The MSG Ctrl sub-block provides the memory for the message buffers and its control.
The upper 16 message buffers have no static assignment to RXONLY-CH or DIAG-CH. The assignment of the message buffers depends on the status of RXONLY-CH. The basic functions of these message buffers are:

When assigned to the RXONLY-CH:

- Receive all data frames from RXONLY-CH without any acceptance filtering when Mirror mode or RECONLY mode is selected. In case of the Mirror mode w/ TIF, data/remote frames are filtered by mask register and reference ID registers.
- Receive all remote frames from RXONLY-CH without any acceptance filtering.
- Behave as ring buffer in FIFO manner. Storage of received messages is calculated with help of LIPT of RXONLY_CH.
- Automatically send all messages to DIAG-CH in FIFO manner. The TX-search state machine of the DIAG-CH takes care to send these message on the diagnostic bus.
  Any TRQ by a FIFO candidate needs to start a TX-search, which always starts at message buffer #0. The TX-search always must be applied to both areas, the DIAG-CH message buffers and the FIFO.

When assigned to the DIAG-CH:

- Send data frames to DIAG-CH
- Receive data frames from DIAG-CH
- Receive remote frames from DIAG-CH
- Send remote frames to DIAG-CH
- Each message buffer contains its own identifier. Therefore a message buffer can be reserved to receive a data frame unambiguously or to transmit only a particular message frame.
- A mask can be assigned to a message buffer in case the reception of a group of data frames is required.
- Furthermore, message buffers can be grouped to build multi-buffer receive blocks (MBRB).

In other words, the complete message buffer memory (48 buffers) are fully assigned to DIAG-CH with all related features of an ordinary AFCAN channel.

The lower message buffers (#0 to #31) are always assigned to the DIAG-CH.

14.3.5 Arbitration Logic

The Arbitration Logic sub-block controls the access of the RXONLY-CH and the DIAG-CH to the MSG Ctrl sub-block.

14.3.6 RXONLY-CH CAN machine

The RXONLY-CH CAN machine sub-block contains logic of the CAN protocol transfer layer, which is dedicated to monitor a CAN BUS chosen by the selector. It never transmits any dominant bit on the monitored bus. The RXONLY-CH is only connected to the monitored CAN channel via the RX-input terminal.

14.3.7 DIAG-CH CAN machine

The DIAG-CH CAN machine sub-block contains all logic for the CAN Protocol transfer layer. All protocol activities required for the reception of data frames, the transmission of message frames and the CAN protocol error management are executed automatically.
For transmission the DIAG-CH CAN machine uses always all 48 message buffers. However the storage of received messages depends on the state of the RXONLY-CH CAN machine. If the RXONLY-CH CAN machine is ‘in power save mode’ (PSMODE = STOP) or in initialisation mode (OPMODE = INIT), all message buffer are assigned to the DIAG-CH. When RXONLY-CH CAN machine is online or in sleep mode while the operating mode ‘RXONLY’ or ‘Mirror Mode’ is selected, only the lower 32 message buffer (#0 - #31) are used to store received messages from the DIAG-CH CAN machine.

Note: When preparing for mirror mode while the upper 16 message buffers are still assigned to DIAG-CH, the application needs to clear all pending TRQ bits of these buffers at first. Then, all RDY of these buffers need to be cleared as well. Then all these buffer need to be configured as receive message buffers before the initialisation state for the RXONLY-CH is released; the buffers are assigned to RXONLY-CH.

The acceptance filtering process, which decides whether a received data frame has to be stored into one of the assigned message buffers, is executed in the CAN module. The acceptance filtering process manages the storage into message buffers with and without an assigned mask, into multi-buffer receive blocks, and it resolves ambiguous storage situations.
In case more than one of the assigned message buffers are defined as transmit message buffer, more than one transmit request may be pending when the CAN module is able to get access to the CAN bus. The CAN module automatically detects the pending transmit request with the highest priority, which is evaluated by specific rules, and processes the transmission.

14.3.8 Memory and register layout

Figure 14-3: Register Address Layout

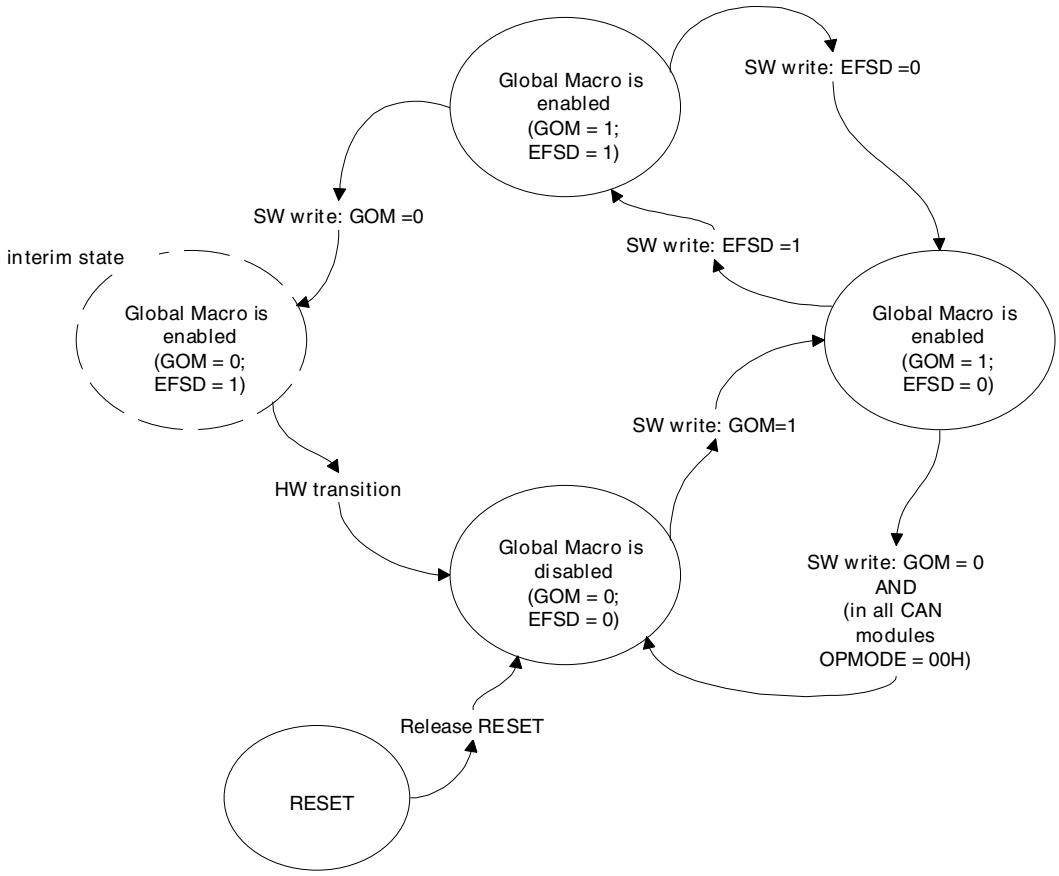
07FH 040H	CAN Module Register (for both RXONLY and DIAG)
03FH 000H	Global Macro Control Register

14.4 Macro Initialisation and Control

Macro initialisation is defined as the software processes, which are necessary to use the DIAG macro after a RESET (i.e. system reset generated with the $\overline{\text{RESET}}$ pin of the microcomputer system).
Global DIAG macro switch-off by software is performed by clearing the GOM bit (0).

After those 2 transitions the global macro is disabled. The CPU can check the status of the global macro by reading the GOM bit in the CGMCTRL register.

Figure 14-4: State transitions of Global Macro switching



14.4.1 Global Macro initialisation and control

(1) Global Macro Disabled (GOM=0)

The global macro is disabled after a RESET or a global DIAG macro switch-off by software. The CPU can check the status of the macro by reading the GOM bit in the CGMCTRL register.
Characteristic of global macro disabled (GOM=0):

- CAN I/F channel output signal CANnTX is fixed to recessive level (logical 1)
- Read and write access to the global macro registers CGMCTRL, CGMCS, CGMABT, CGMABTD is possible.
- No read access and no write access to the message buffers.
- CAN modules are disabled.
- No write access to the CAN module register (CnMASK1L, CnMASK1H, CnMASK2L, CnMASK2H, CnMASK3L, CnMASK3H, CnMASK4L, CnMASK4H, CnCTRL, CnLEC CnINFO, CnERC, CnIE, CnBRP, CnBTR, CnLIPT, CnRGPT, CnLOPT, CnTGPT, CnTS, CnCTRL_R, CnLEC_R, CnERC_R, CnIE_R, CnINTS_R, CnBRP_R, CnBTR_R, CnLIPT_R, CnBSEL_R). Accidental write accesses to those registers are ignored by the hardware.

(2) Initialisations before Enabling the Global Macro

The application software has to determine all settings for the macro clock f_{CANMOD} in the CGMCS register. The clock settings for the macro clock f_{CANMOD} must not be changed after the global macro is enabled.

(3) Global Macro Enabled

The global macro is switched on by setting GOM bit (1) in CGMCTRL register. Characteristics of global macro enabled (GOM=1):

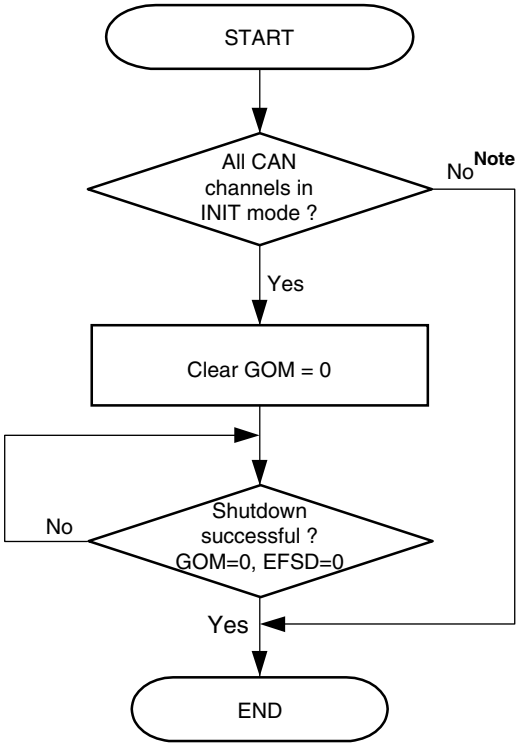
- CAN I/F channels are enabled
- Both CAN modules are in INIT mode
- The CAN module register CnCTRL, CnLEC, CnINFO, CnERC, CnIE, CnINTS, CnBRP, CnBTR, CnTS, CnCTRL_R, CnLEC_R, CnERC_R, CnIE_R, CnINTS_R, CnBRP_R, CnBTR_R, CnBSEL_R contain their initial values upon switching on the global macro. The CAN module register CnLIPT, CnRGPT, CnLOPT, CnTGPT, CnLIPT_R, CnLOPT_R contain undefined values.
- Write access to the global macro register CGMCS is ignored.
- Read and write access to all message buffers is possible.
- Read and write access to the CAN module registers (CnMASK1L, CnMASK1H, CnMASK2L, CnMASK2H, CnMASK3L, CnMASK3H, CnMASK4L, CnMASK4H, CnCTRL, CnLEC, CnERC, CnIE, CnINTS, CnBRP, CnBTR, CnTS, CnCTRL_R, CnLEC_R, CnIE_R, CnINTS_R, CnBRP_R, CnBTR_R, CnBSEL_R) is possible. Read access to the CAN module registers CnLIPT, CnRGPT, CnLOPT, CnTGPT, CnLIPT_R and is possible.
- The CAN module register CnTIDR0L/H, CnTIDR1L/H, CnTIDR2L/H, CnTIDR3L/H, CnTIDR4L/H, CnTIDR5L/H, CnTIDR6L/H, CnTIDR7L/H, CnTIDML/H contain their initial values upon switching on the global macro.

(4) Switch-Off the Global Macro

The global macro has not be switched off while one or more CAN modules are processing a message frame. A sudden interruption of a message frame transmission or reception will cause an error in other nodes connected to the same CAN bus.
Before switching off the global macro the application software has to switch all CAN modules in INIT mode. Clearing the GOM bit in CGMCTRL register will put a switch-off request to the global macro. The switch-off request is discarded in case one or more of the CAN modules are not properly set to INIT mode.

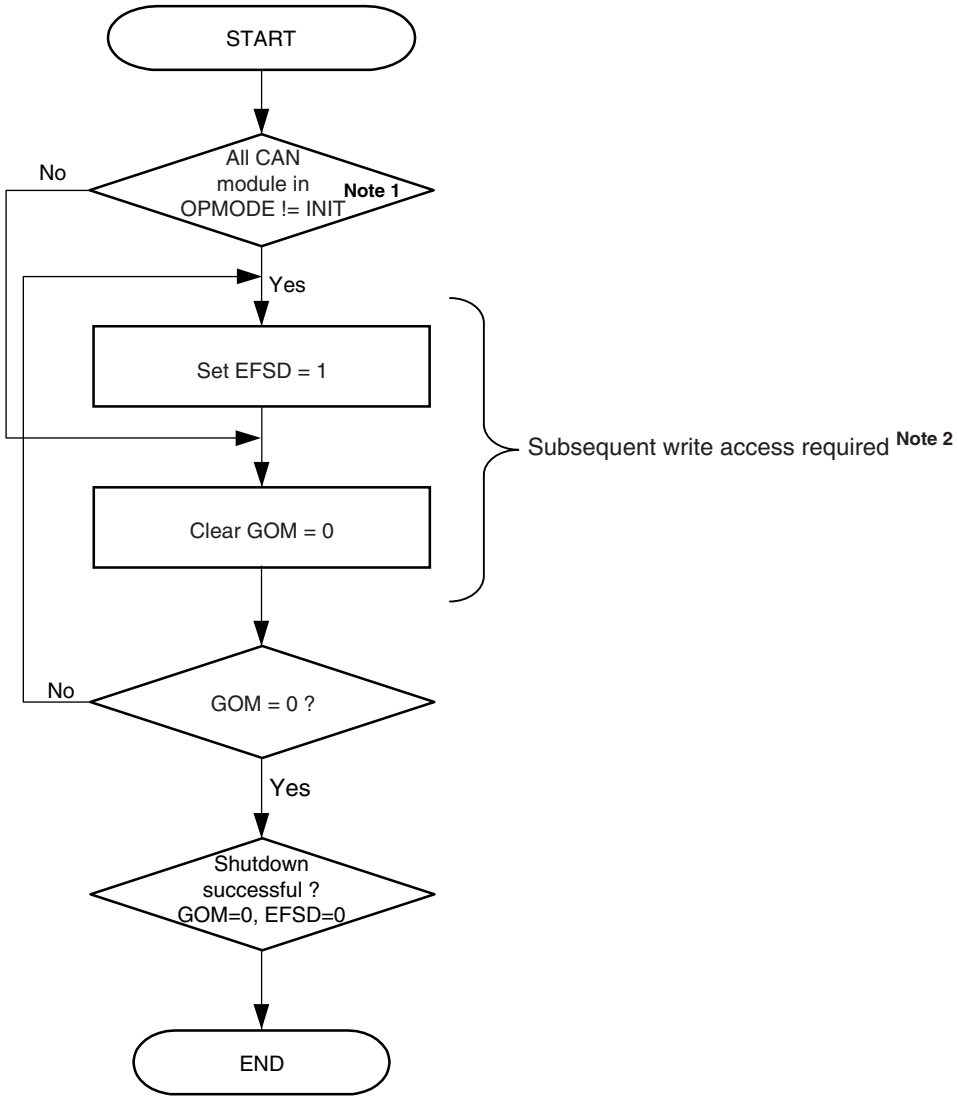
Some applications may require an immediate switch-off of the entire macro in an emergency, regardless of generating disturbance to the other nodes connected to the CAN bus. For such applications the EFSD bit in CGMCTRL register can be set (1) to allow an immediate global macro switch-off without switching the CAN modules to a defined state (i.e. INIT mode) beforehand.
When using the 'forced shut down' function by setting EFSD bit (1) the subsequent access to the macro has to be the instruction to clear GOM bit (0). In case setting EFSD bit (1) is not directly followed by clearing GOM bit (0), the EFSD bit is cleared (0) automatically (i.e. the 'forced shut down' function is disabled). Also a read access to the CGMCTRL register between setting of EFSD bit (1) and clearing GOM bit (0) will lead to an automatic clear (0) of EFSD bit.

Figure 14-5: Shutdown Process (Normal Shutdown)



Note: Consider to use forced shutdown procedure.

Figure 14-6: Shutdown Process (Forced Shutdown)



Notes: 1. OPMODE:
-Normal operation mode
-Normal operation mode with ABT
-Receive only mode
-Single shot mode
-Self test mode

2. Ensure that no CPU read or write access to any register happen between EFSD=1 and GOM=0.
To guarantee that also all interrupts have to be disabled.

(5) Flowchart of the Initialisation

The following flowcharts describe the basic initialisation and in addition the re-initialisation of the CAN module. Note that the configuration for the baud rate of RXONLY-CH should have the same configuration as the monitored CAN channel.

Figure 14-7: DIAG Macro Initialisation

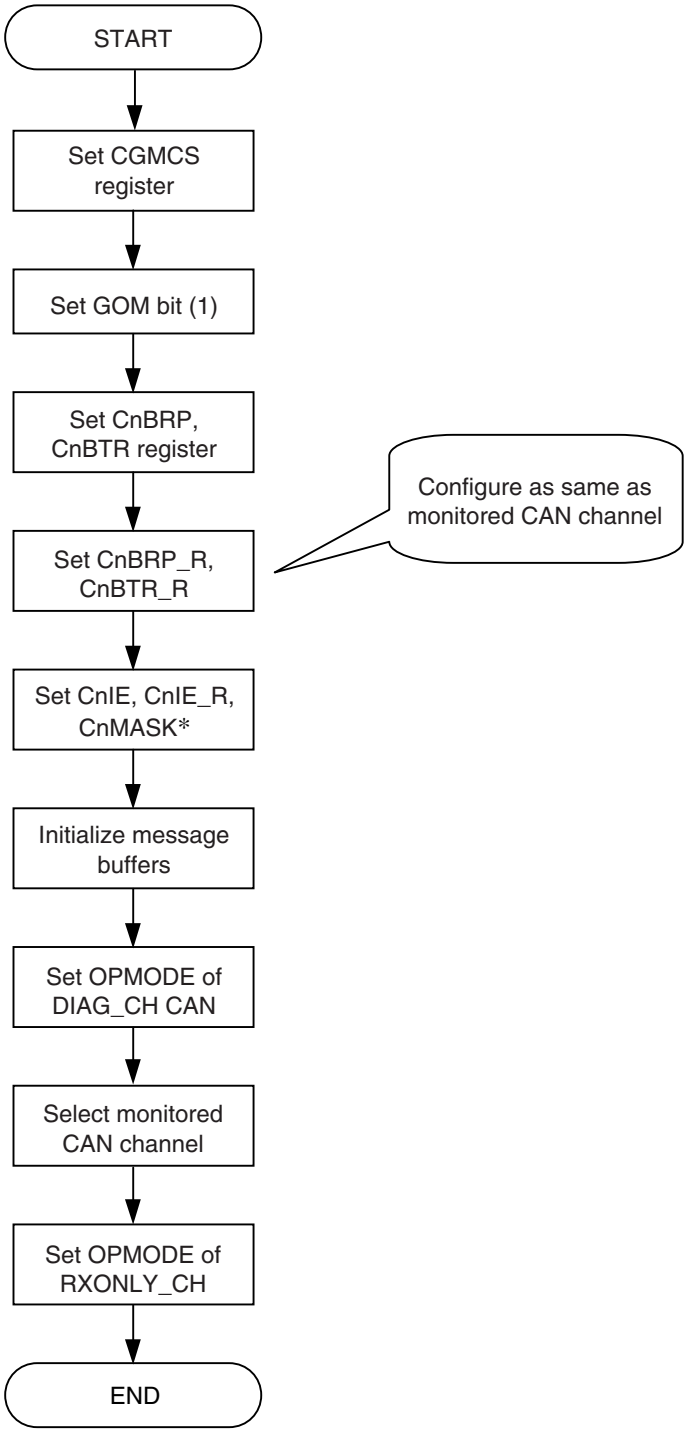
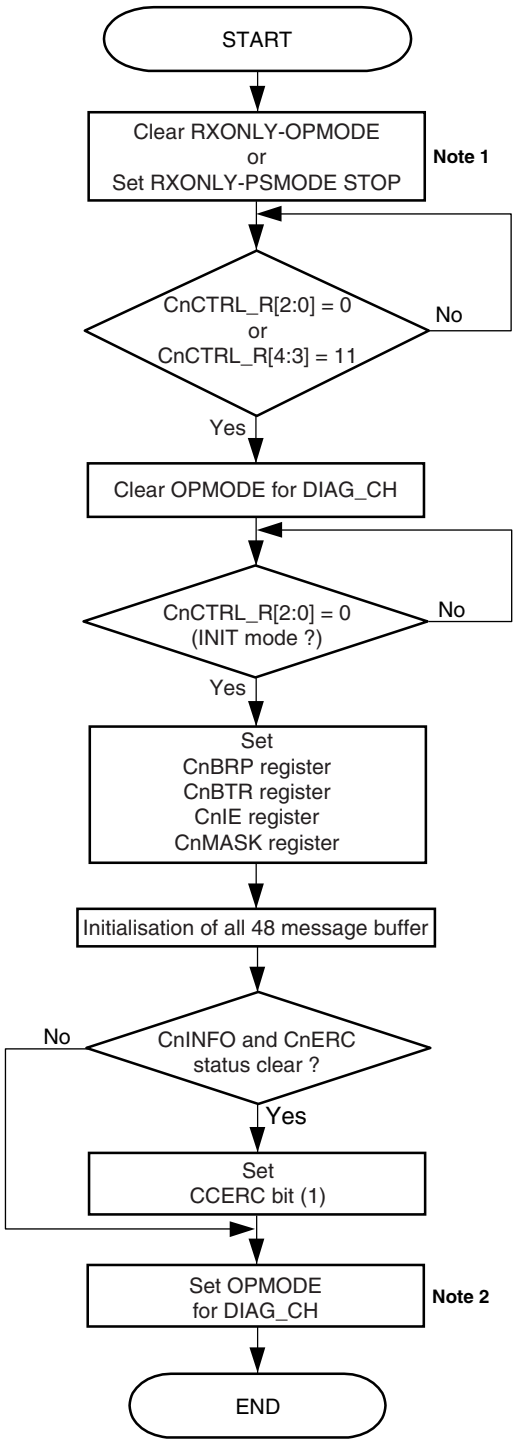


Figure 14-8: Re-initialisation of the DIAG-CH CAN Module using 48 buffers



Notes: 1. OPMODE for RXONLY_CH - Initialization mode
PSMODE for RXONLY_CH - STOP mode

2. OPMODE:
-Normal operation mode
-Normal operation mode with ABT
-Receive only mode
-Single shot mode
-Self test mode

Figure 14-9: Re-initialisation of the DIAG-CH CAN Module using TX ID Filter function

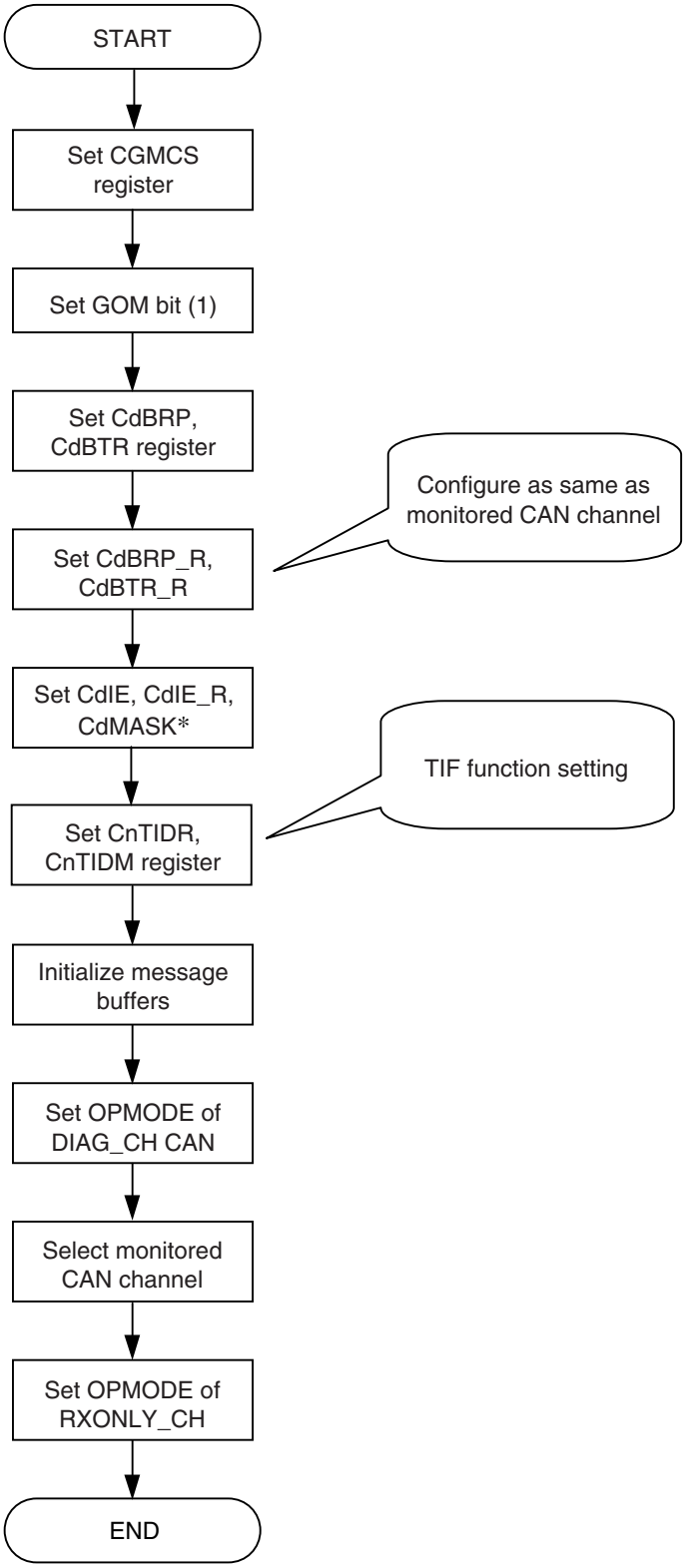
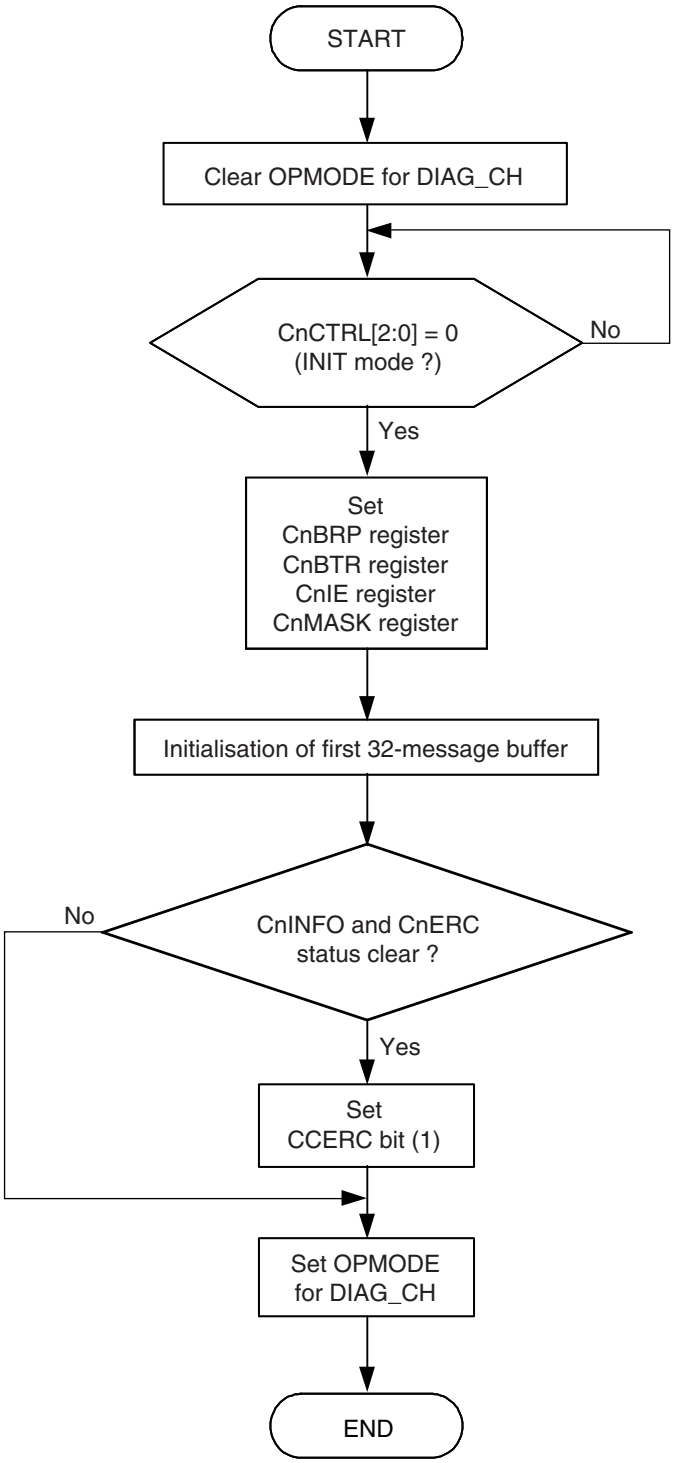


Figure 14-10: Re-initialisation of the DIAG-CH CAN Module using first 32 buffers only



Remark: OPMODE:
-Normal operation mode
-Normal operation mode with ABT
-Receive only mode
-Single shot mode
-Self test mode

14.4.2 Message buffer initialisation and configuration

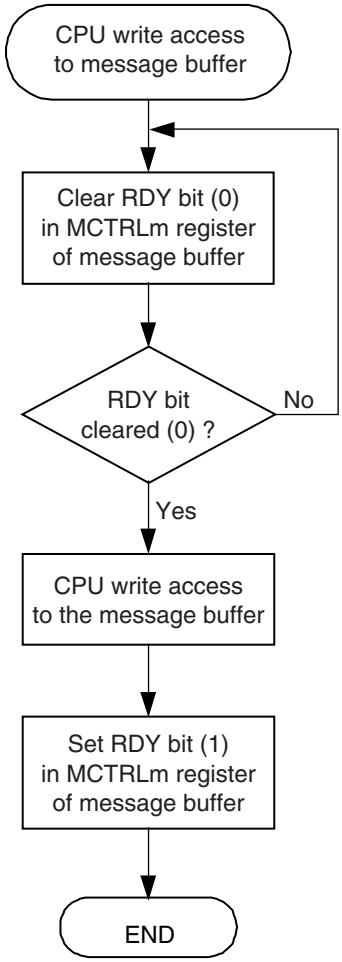
After a RESET or a global DIAG macro switch-off by software, the message buffers contain undefined values. A minimum initialisation for all message buffers, even for those not used in the application, is necessary before switching one or more CAN modules from INIT mode to one of the operational modes (refer to Figure 14-16, “Transitions for operational modes of the DIAG-CH CAN module,” on page 656 for details of the operational modes). The minimum initialisation has to include the proper configuration of the following bits and bit-strings in the message buffer registers MCTRLm and MCONFm to avoid unexpected behaviour of the macro in the operational modes.

Table 14-2: Minimum configuration of message buffer even when unused in the application

Bit	Register	Minimum initialisation value
RDY	MCTRLm	0B
TRQ	MCTRLm	0B
DN	MCTRLm	0B
MA0 to MA2	MCONFm	0x00B

The RDY bit in the MCTRLm register of each message buffer must be used as a semaphore to avoid access conflicts when writing to a message buffer.

Figure 14-11: CPU write access to a message buffer

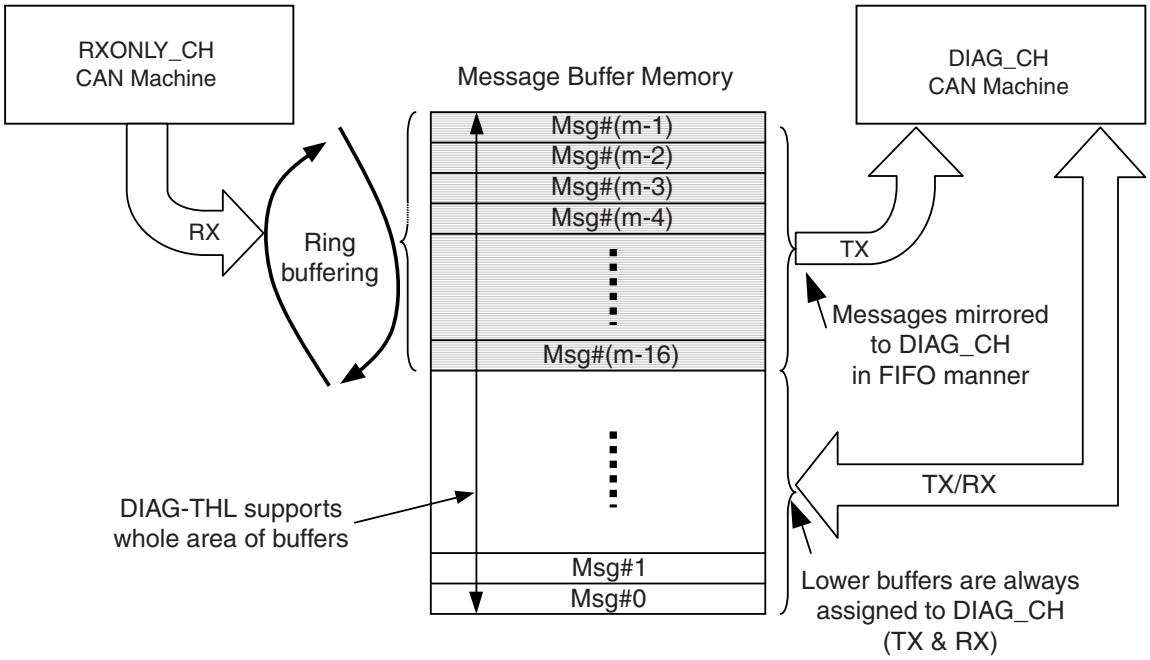


While RDY bit is cleared (0) the message buffer is not accessed by the assigned CAN I/F channel (CAN module and/or TTCAN module) and/or bridge module (RXONLY-CH). Thus no message frame can be received in the message buffer or transmitted from the message buffer. Whenever the CPU wants to write into the message buffer, the RDY bit must be cleared (0).
Once the CPU has finished the write access to the message buffer it must set the RDY bit (1) to allow CAN protocol processing to the particular message buffer by the assigned CAN I/F channel.
CPU write accesses to a message buffer with RDY bit set (1) are ignored.
For the RXONLY-CH the semaphore handling via RDY bit is identical to the rule mentioned above. However, the application is strongly advised to set all RDY bits (1) for the upper 16 message buffers in order to provide the maximum depth for the ring buffer in mirror mode. As well it is absolutely mandatory not to write to the RDY bit (clear or set the RDY bit) to any of the upper 16 message buffers while the RXONLY-CH is operated in mirror mode or mirror mode w/ TIF. This will terminate the mirror mode or mirror mode w/ TIF or cause an erroneous sequence of monitored messages on the diagnostic CAN bus.

14.4.3 Message buffer to CAN I/F channel assignment

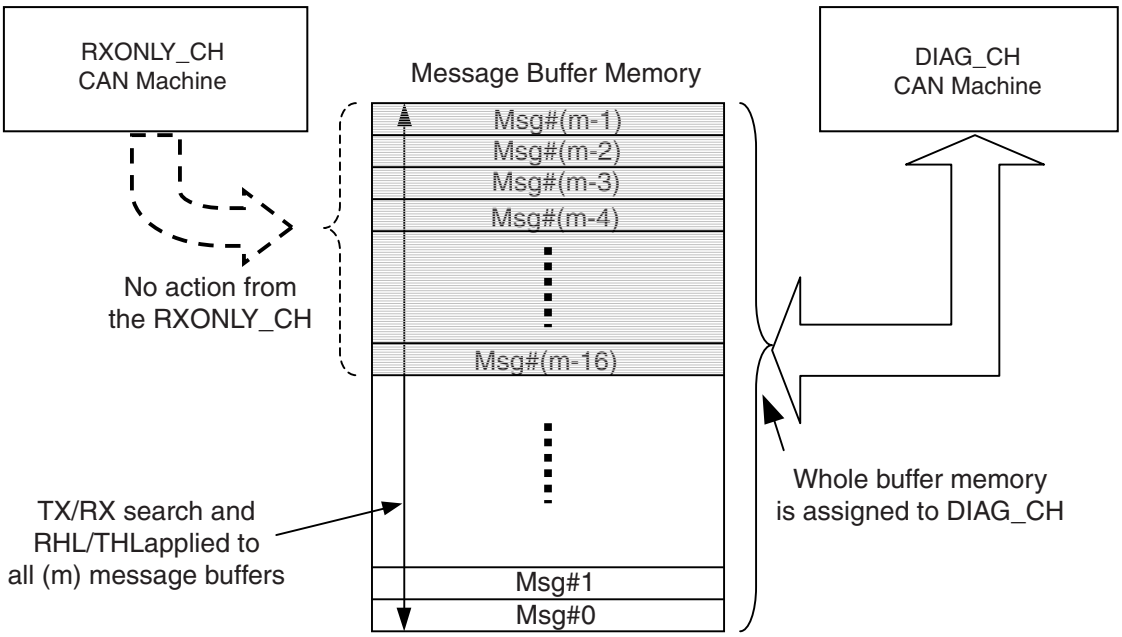
The area of upper 16 message buffers can be assigned to the DIAG-CH or RXONLY-CH while the rest of the message buffers is statically assigned to the DIAG-CH as shown in the figures below.

Figure 14-12: Operation in Mirror or Receive-only Mode of RXONLY-CH



The assignment of the upper 16 message buffers depends on the OPMODE and PSMODE of RXONLY-CH CAN machine.
Basically, these buffers are assigned to the RXONLY-CH while the RXONLY-CH is operational mode (i.e. mirror mode / rec-only mode). And additionally, it must be assigned to the RXONLY-CH while the RXONLY-CH is in SLEEP mode because the wake-up event on the RXONLY-CH is unpredictable. In STOP mode these buffers are assigned to the DIAG-CH in order to provide a mode of operation where the DIAG-CH uses all 48 buffers while the RXONLY-CH uses minimum power. The user needs to configure the upper16 message buffers before changing assignment (i.e. leaving initialisation or STOP mode of the RXONLY-CH).

Figure 14-13: Operation during INIT or STOP of RXONLY-CH



The assignment of message buffers versus operational and power-save modes is shown in the table below.

Table 14-3: Message buffer assignment versus PS/OPMODE

OPMODE/ PSMODE	MIRROR mode	MIRROR mode w/ TIF	REC-ONLY mode	INIT mode
NO PWR-SAVE	RXONLY	RXONLY	RXONLY	DIAG
SLEEP	RXONLY	RXONLY	RXONLY	
STOP	DIAG	DIAG	DIAG	

When the assignment of the upper 16 message buffers switches, the application should configure all buffers beforehand as follows.

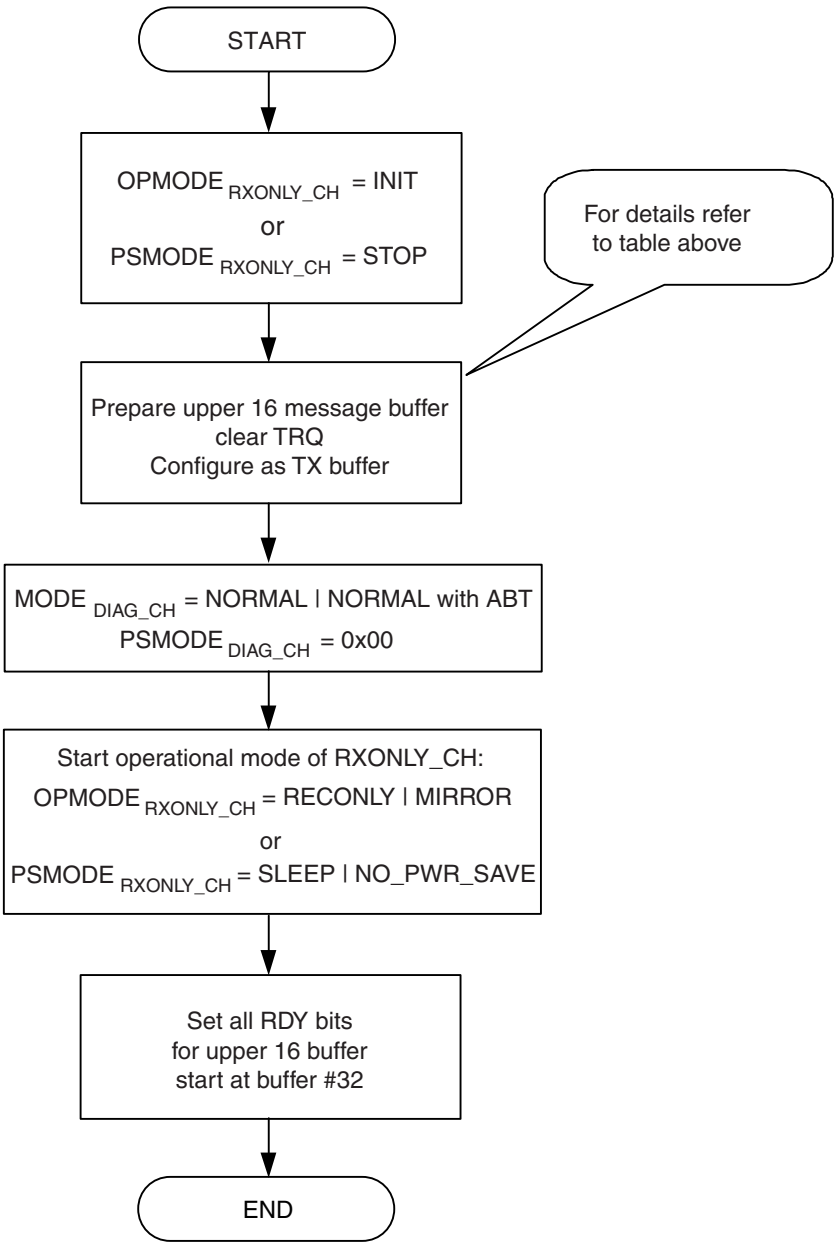
(1) Configuration before switching to Mirror Mode

For mirror mode operation all upper 16 message buffers shall be used as a ring buffer working in FIFO manner. Thus, the application has to prepare these buffers as TX-buffers beforehand as follows. Note that OWS setting is invalid (not necessary) because the non-overwriting of message buffers with DN = 1 is built in feature of operational modes of the RXONLY-CH.

Bit name	Configuration
MOW	Clear
IE	Depends on application
DN	Clear
TRQ	Clear
MA[2:0]	001b
MT[2:0]	000b
RTR	Don’t care
OWS	Don’t care
IDE	Don’t care
ID	Don’t care
DLC	Don’t care
MDATA	Don’t care

The flow of operations in order to change the assignment of message buffers to a CAN-channel is shown in the figure below. At first the application needs to apply RDY = 0 for the upper 16 message buffer or set the DIAG-CH into INIT mode before the new configuration can be written to the buffers.
Once the new configuration is set, the DIAG-CH followed by RXONLY-CH is put into its designated operational and power save mode. Finally the upper 16 message buffer are re-enabled by setting the RDY bit (1).

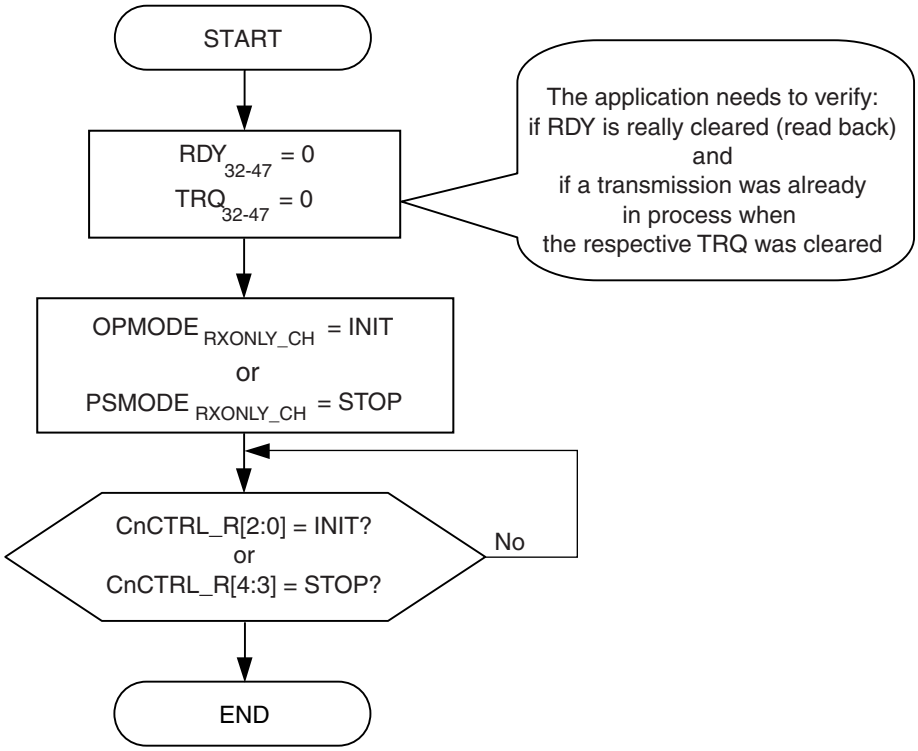
Figure 14-14: Set up of Mirror Mode, Mirror Mode w/ TIF or RECONLY Mode for RXONLY-CH



(2) Configuration before switching to DIAG side

All upper 16 message buffers can be assigned to the DIAG-CH in order to use them as normal TX/RX message buffers by the application. It is not necessary to apply a particular configuration for these buffers before the assignment is linked again to the DIAG-CH. However, it is recommended that the application clears RDY and TRQ bits before assigning the buffers to the DIAG-CH or puts the DIAG-CH into OPMODE = INIT or PSMODE = STOP before changing the assignment. Then, at least clear the RDY bits for the upper 16 message buffers before putting the DIAG-CH to normal mode. This will ensure proper message handling during the transition.

Figure 14-15: Cancellation of Mirror Mode or Receive-only Mode for RXONLY_CH



When clearing the TRQ in the upper message buffer a transmission from one of these buffers may still be in process. In order to ensure correct message handling (i.e. transmission history list), it is recommended to wait for the first transmission completion interrupt of the DIAG_CH before resuming the transition procedure.

14.4.4 Configuration of a transmit message buffer

Any message buffer out of the 48 available buffer can be used for transmission. In case of mirror mode, mirror mode w/ TIF or receive-only mode of RXONLY-CH, only the first 32 message buffers (#0 – 31) can be used by the application to transmit messages. Using a message buffer as a transmit message buffer the value 000B must be written to the message buffer type bit-string MT[2:0] in the MCONFm register.
Functions of a message buffer defined as a transmit message buffer are:

- Transmit of data frames
- Transmit of remote frames
- Reception of remote frames

The message buffer type should be defined in the INIT mode of a CAN module. A re-configuration of the message buffer type in the operational modes is possible but special care must be considered because the characteristic of the message buffer will change.
Writing an initial message frame identifier value in the identifier registers MIDLm and MIDHm registers can be done already in INIT mode of a CAN module, especially when using the transmit message buffer according the FullCAN-buffer principle (i.e. reserving a particular message buffer for only one message).
Using a transmit message buffer according the BasicCAN-buffer principle means to send various messages via that message buffer. As a consequence the MIDLm and MIDHm registers must be re-configured by CPU before each transmission process. Therefore re-configuration of the MIDLm and MIDHm registers is required during the operational modes of the CAN module.
MDLCm register and MDATA0m to MDATA7m registers can be initialised in INIT mode. Those registers must be defined before each transmission request of a data frame in the operational modes of the CAN module. Before launching the transmission request of a remote frame only the MDLC[3:0] bit-string must be defined and the RTR bit must be set (1).
After the initialisation of the message buffer has been finished, the RDY bit in the MCTRLm register needs to be set (1) again.

(1) Configuration of a transmit message buffer for automatic block transmission (ABT)

Automatic block transmission (ABT) is an operational mode of the CAN module in the CAN I/F channel, which allows consecutive transmission of large data blocks with a reduced CPU interaction (please refer to 14.7.3 ”Automatic block transmission (ABT)” on page 679). Automatic block transmission allows only the transmission of data frames.
The user must not transmit any remote frames from the message buffers belonging to the ABT message buffer group (i.e. message buffer 0 to message buffer 15) when the CAN module operates in the operational mode “Normal Operating Mode with Automatic Block Transmission”. Neither the user must not sent remote frames, which have got the same identifier as message objects located in the message buffers 0 to 15, by this or another node of the network.
The number of message buffers for automatic block transmission and their location in the message buffer memory is statically fixed according the following scheme:

Table 14-4: ABT message buffer allocation scheme

ABT message buffer type	Message buffer number
start-element:	0
end-element:	7

The automatic block transmission can simply be selected in any CAN I/F channel by switching it into the operational mode “Normal Operating Mode with Automatic Block Transmission”. The application software is responsible to set the bit-string MA[2:0] in all message buffer of ABT message buffer group corresponding to the CAN I/F channel number of the CAN module which has to operate in “Normal Operating Mode with Automatic Block Transmission”. Further, all message buffers must be defined as transmit message buffer by setting the value 00H to the message buffer type bit-string MT[2:0] in the MCONFm register.

The identifier of the data frames, which are processed by the ABT function, must be initialised according to the requirements of the application. In case all data of a data block has to be transported by using always the same data frame, the same identifier value must be written to all message buffers belonging to the ABT message buffers group. Writing the identifier values to the MIDLm and MIDHm registers of those message buffers should be done in the INIT mode of the CAN module.
Changing the MIDLm and MIDHm registers in the operational mode “Normal Operating Mode with Automatic Block Transmission” is also possible and should be used for applications where the data block transportation is realised by using data frames with different or alternating identifiers.
MDLCm register and MDATA0m to MDATA7m registers can be already initialised in INIT mode, but must be set before the transmission request for the ABT message buffer group is issued in the operational mode “Normal Operating Mode with Automatic Block Transmission”.
After the initialisation of a message buffer, which belongs to the ABT message buffer group, has been finished, the RDY bit in the MCTRLm register needs to be set (1). The TRQ bit in the MCTRLm register must not be set (1) by the user neither in INIT mode nor in “Normal Operating Mode with Automatic Block Transmission” mode. Issuing the transmit request for the ABT group is done by the ABTTRG in the CGMABT register.

(2) Configuration for a receive message buffer without link to a mask

Using a message buffer as a receive message buffer, in which only one explicit data frame is accepted (i.e. FullCAN-buffer principle), the value 001B must be written to the message buffer type bit-string MT[2:0] in the MCONFm register. The message buffer type can be defined in the INIT mode of a CAN module. A re-configuration of the message buffer type in the operational modes is possible.
An initial value must be written to the identifier registers MIDLm and MIDHm (standard identifier = ID[28:18] / extended identifier ID[28:0]) in the INIT mode of a CAN module. This value determines, which data frame received from the CAN bus is accepted by the message buffer. Re-configuration of the identifier during the operational modes is possible.
The data length code bits MDLCm register and the data field MDATA0m to MDATA7m registers do not need to be initialised during the INIT mode.
After the initialisation of the message buffer has been finished, the RDY bit in the MCTRLm register can be set (1).

(3) Configuration for a receive message buffer linked to a mask

Using a message buffer as a receive message buffer with a link to a mask in the assigned CAN I/F channel allows the acceptance of a range of data frames with different identifiers. The value of the ID[28:0] bit-string in the MIDLm and MIDHm registers and the value of the linked mask determine the range of acceptable data frames (refer chapter 3.5.3 for details).
The following values must be written to the message buffer type bit-string MT[2:0] in the MCONFm register to link a message buffer to one of the 4 masks in a CAN module of a CAN I/F channel.

Table 14-5: Message buffer configuration with mask link defined by MT[2:0]

MT[2:0] = 010B	to link message buffer to MASK1
MT[2:0] = 011B	to link message buffer to MASK2
MT[2:0] = 100B	to link message buffer to MASK3
MT[2:0] = 101BH	to link message buffer to MASK4

The message buffer type can be defined in the INIT mode of a CAN module. A re-configuration of the message buffer type in the operational modes is possible but special care must be considered because the characteristic of the message buffer will change.
An initial value must be written to the identifier MIDLm and MIDHm registers in the INIT mode of a CAN module. The value in the MIDLm and MIDHm registers and the value in the linked mask determine, which data frames received from the CAN bus are accepted by the message buffer (refer also 3.5.3). Re-configuration of the identifier during the operational modes is possible, but should be done with special care because the range of acceptable data frames will also change. The same guideline applies for the definition of the linked mask. In detail, masks should be initially defined during the INIT mode of the CAN module. In case a re-configuration of the mask is required during one of the operational modes, all message buffers with a link to that mask should be deactivated by clearing the RDY bit (0).
The data length code bits in MDLCm register and the data field in the MDATA0m to MDATA7m registers have not to be initialised during the INIT mode.
After the initialisation of the message buffer has been finished, the RDY bit in the MCTRLm register should be set (1).

(4) Initialisation of a Multi-buffer Receive Block (MBRB)

A Multi-buffer receive block (MBRB) is a group of receive message buffers assigned to the same CAN I/F channel (MA[2:0]) and with an equal type definition (MT[2:0]).
A MBRB can be used to collect several data frames from the CAN bus without an immediate CPU reaction upon the reception of each data frame (i.e. buffering).
Due to the different characteristic of receive message buffers without a link to a mask and with a link to a mask, MBRB of both types have also a different behaviour.

(a) MBRB of message buffers without a link to a mask

Criteria to built a MBRB not linked to a mask:
All message buffers must be assigned to the same CAN I/F channel (MA[2:0] criteria)
All message buffers must be configured as message buffer type 01H (MT[2:0] = 01H)
All message buffer must have the same identifier (ID[28:0] bit-string criteria).

As a consequence several MBRB without a link to a mask can be assigned to a CAN I/F channel.
Building a MBRB of message buffers with no link to a mask requires the same identifier in all message buffers. Hence the MIDLm and MIDHm registers (standard identifier = ID[28:18] / extended identifier = ID[28:0]) has to be initialised accordingly during INIT mode of the CAN module.
Changing the MIDLm and MIDHm registers in one of the message buffer belonging to a MBRB not linked to a mask in one of the operational modes of the CAN module does remove the message buffer from the MBRB. Therefore a MBRB should not be re-configured in the operational modes.
The data length code bits in MDLCm register and the data field in the MDATA0m to MDATA7m registers do not need to be initialised during the INIT mode.
After the initialisation of a message buffer has been finished, the RDY bit in the MCTRLm register can be set (1).

(b) MBRB of message buffers with a link to a mask

Criteria to built a MBRB linked to a mask:
All message buffers must be assigned to the same CAN I/F channel (MA[2:0] criteria)
All message buffers must be configured as message buffer linked to the same mask (MT[2:0] = 02H, or 03H, or 04H or 05H)

As a consequence only one MBRB with a link to a mask for each mask can be configured for one CAN I/F channel (in total 4 MBRB with link to a mask per CAN I/F channel).
The identifier values in the message buffers belonging to a MBRB of message buffers with link to a mask (i.e. MT[2:0] = 02H, 03H, 04H, 05H) do not have to be equal. But, because the combination of the content in the MIDLm and MIDHm registers and the actual content of the linked mask determine the range of acceptable data frames, the user must be aware that different ranges of data frames will be received in a MBRB linked to a mask with different content in the MIDLm and MIDHm registers.
Re-configuration of the MIDLm and MIDHm registers in the operational modes of the CAN module is possible, but should be done with special care, because changing the identifier will change the range of data frames, which can be accepted by the message buffer.
The data length code bits in the MDLCm register and the data field in the MDATA0m to MDATA7m registers do not need to be initialised during the INIT mode.
After the initialisation of a message buffer has been finished, the RDY bit in the MCTRLm register should be set (1).

14.4.5 DIAG Macro initialisation and control

After a RESET the registers CGMCTRL, CGMCS, CGMABT, CnMASK1L, CnMASK1H, CnMASK2L, CnMASK2H, CnMASK3L, CnMASK3H, CnMASK4L, CnMASK4H, CnCTRL, CnLEC, CnINFO, CnERC, CnIE, CnINTS, CnBRP, CnBTR, CnTS, CnCTRL_R, CnLEC_R, CnERC_R, CnIE_R, CnINTS_R, CnBRP_R, CnBTR_R, CnLIPT_R, CnBSEL_R of the CAN module in the CAN I/F channels contain their initial value.
Before switching the CAN module into an operational mode the registers (CnMASK1L, CnMASK1H, CnMASK2L, CnMASK2H, CnMASK3L, CnMASK3H, CnMASK4L, CnMASK4H, CnCTRL, CnIE, CnBRP, CnBTR, CnTS, CnCTRL_R, CnIE_R, CnBRP_R, CnBTR_R) must be initialised according the requirements of the application. Further, the CnLEC and CnLEC_R registers must be set to its initial value (00H) and the VALID bit in the CnCTRL and CnCTRL_R registers must be cleared (0) by the user.

14.4.6 CAN bit time programming

Any CAN module in a CAN I/F channel can be programmed separately to baud rates up to 1 Mbps.

- Requirements by ISO standard:
  Bit time “DBT”, programmable from 8 TQ to 25 TQ (TQ = time quanta)
  Information processing time “IPT” less or equal 2 TQ at all prescaler settings in CnBRP and CnBRP_R registers.

- Specific requirements:
  On AFCAN versions with 32 buffers baud rates of 1 Mbps are achieved for CAN module frequencies (f_{CANMOD}) of 8 MHz or more. For 48 Buffer versions at least 16 MHz need to be supplied. In order to insure proper operation during all real-time scenarios, the DIAG macro has to be operated at 16 MHz at least.
  The position of the sample point “SPT” can be set to more than 80% of the bit time independently of the length of the propagation segment “PROP_SEG”:

  Example:
  The “SPT” position of more than 80% of the bit time has to be possible even when “PHASE_SEG2” is small and the sum of the “PROP_SEG” and the phase segment 1 is smaller or equal 16 TQ and this sum subtracted by the length of “PHASE_SEG2” becomes greater than 8 TQ.
  The prescaler TQPRS[7:0] in the CnBRP register supports settings starting at 1 CAN module clock ($1/f_{CANMOD}$) per time quantum up to 256 CAN module clocks per time quantum.

For the comfortable programming of the CAN bit timing, the user interface shall only require the setting of the timing_segment1 (i.e. TSEG1[4:0] bit-string in CnBTR and CnBTR_R registers), the timing_segment2 (i.e. TSEG2[3:0] bit-string in CnBTR and CnBTR_R registers), the synchronisation jump width (i.e. SJW[1:0] bit-string in CnBTR and CnBTR_R registers), and the prescaler value (i.e. TQPRS[7:0] bit-string in CnBRP and CnBRP_R registers). For the CAN modules in the AFCAN macro the length of segments “SYNC_SEG”, “PROP_SEG”, “PHASE_SEG1” and “PHASE_SEG2” do not need to be defined explicitly. All necessary CAN bit timings are programmed by defining:

- the number of time quanta TQ per timing_segment1
  (timing_segment1 = “PROP_SEG” + “PHASE_SEG1”)
- the number of time quanta TQ per timing_segment2
  (timing_segment2 = “PHASE_SEG2”)

The CAN protocol segmentation is done by the CAN module automatically.

Due to re-synchronisation mechanisms the CAN module may lengthen “PHASE_SEG1” or shorten “PHASE_SEG2” by one or more TQ. The total number of TQ, for which the CAN module is permitted to lengthen or shorten the phase segments, is called synchronisation jump width “SJW”. The “SJW” value is programmable between 1 TQ and the minimum number of TQ of the “PHASE_SEG1” and 4 TQ.

The DIAG-CH provides the registers CnBRP and CnBTR and the RXONLY-CH provides CnBRP_R and CnBTR_R. The write access to these register has to be allowed only when the particular CAN module is in INIT mode. During an operational mode of the CAN module only reading of these registers has to be granted while the writing is blocked by hardware.

The general rules of valid baud rate settings according to ISO11898 are not validated by the hardware but the user needs to assure them.

After proper setting of the CAN bit time, the CAN module can be released from INIT mode. The CAN Protocol Transfer layer will run a start-up routine (refer ISO11898) to synchronise itself to the current bus activity. After finishing the start-up routine the CAN module becomes able to transmit and receive message frames on the CAN bus.

(1) Operating Considerations

Some rules to be observed for correct bit rate settings apply and have to be met by the application. Observing the following rules for the bit rate setting assures correct operation of a CAN module and compliance to the CAN protocol specification.

- Rule for sampling point “SPT” setting:
  The “SPT” position needs to be programmed between 3 TQ and 17 TQ.
  The “SPT” is determined by the sum of time quanta TQ programmed via TSEG1[3:0] bit-string plus 1 time quantum TQ for the “SYNC_SEG”.
  SPT = ([(binary value in TSEG1[3:0]) + 1] + 1) TQ
  3 TQ ≤ SPT ≤ 17 TQ

- Rule for data bit time “DBT” setting:
  The number of TQ per “DBT” is restricted to a range from 8 TQ to 25 TQ.
  The “DBT” time is determined by the sum of time quanta TQ programmed via TSEG1[3:0] bit-string plus time quanta TQ programmed via TSEG2[2:0] bit-string plus 1 time quantum TQ for the “SYNC_SEG”.
  “DBT” = ([(binary value in TSEG1[3:0]) + 1] + [(binary value in TSEG2[2:0]) +1] +1) TQ
  8 TQ ≤ DBT ≤ 25 TQ

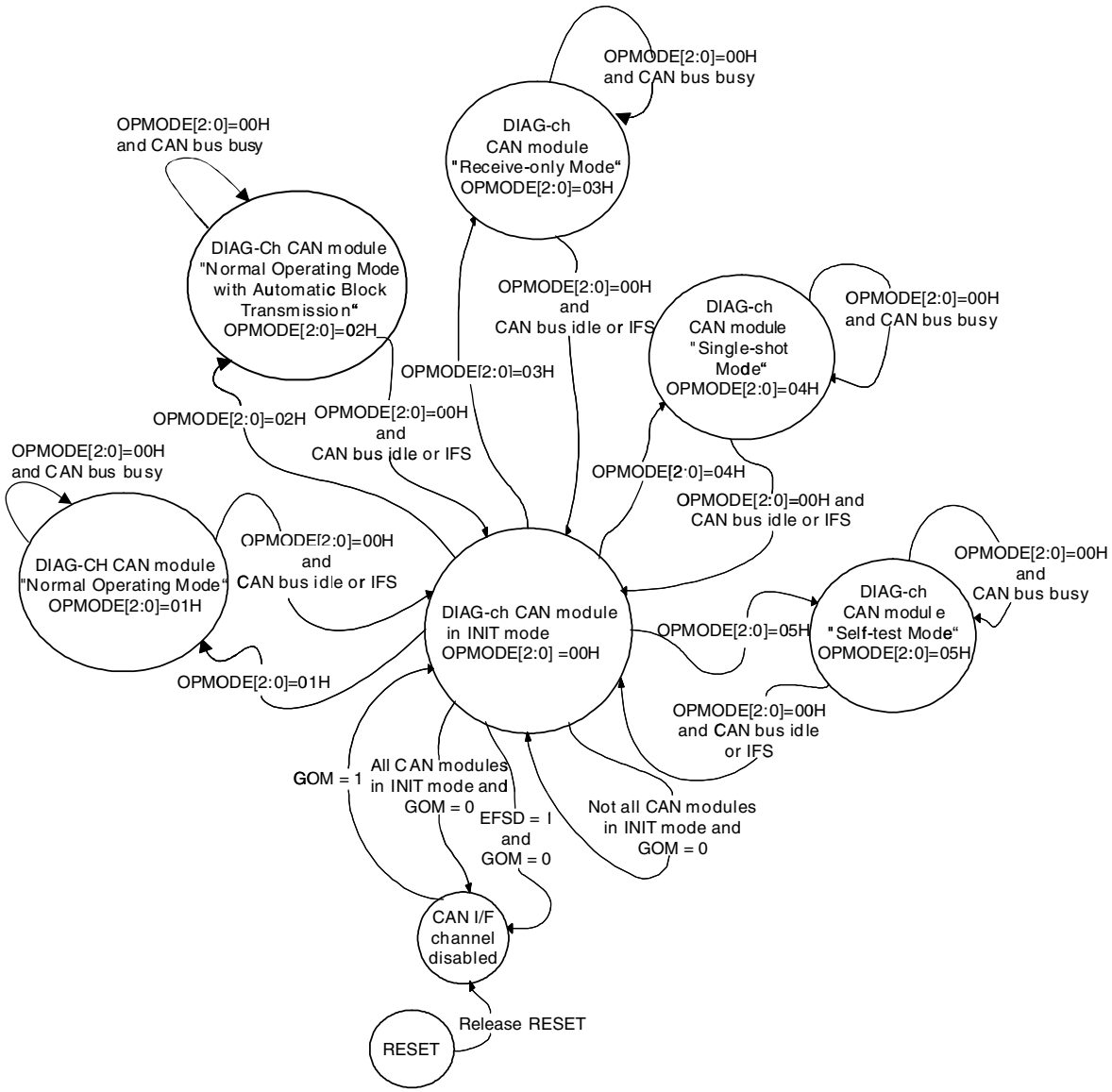
- Rule for synchronisation jump width “SJW” setting:
  The number of TQ allowed for “SJW” must not exceed the number of TQ for “PHASE_SEG2” and must not be more than 4 TQ. The length of “PHASE_SEG2” is given by the number of time quanta programmed in the TSEG2[2:0] bit-string.
  [(binary value in SJW[1:0]) + 1] ≤ [(binary value in TSEG2[2:0]) +1]
  1 TQ ≤ SJW ≤ 4 TQ

  The rules above represent requirements by CAN protocol according ISO 11898. Violating these rules may cause erroneous operation. After proper setting of the CAN bit time, the CAN module can be released from INIT mode into one of its operational modes.

(2) Transitions for Operational Modes of DIAG-CH

The DIAG-CH CAN machine can be switched the following operational modes:
“Normal Operating Mode”
“Normal Operating Mode with Automatic Block Transmission”
“Receive-only Mode”
“Single-shot Mode”

Figure 14-16: Transitions for operational modes of the DIAG-CH CAN module



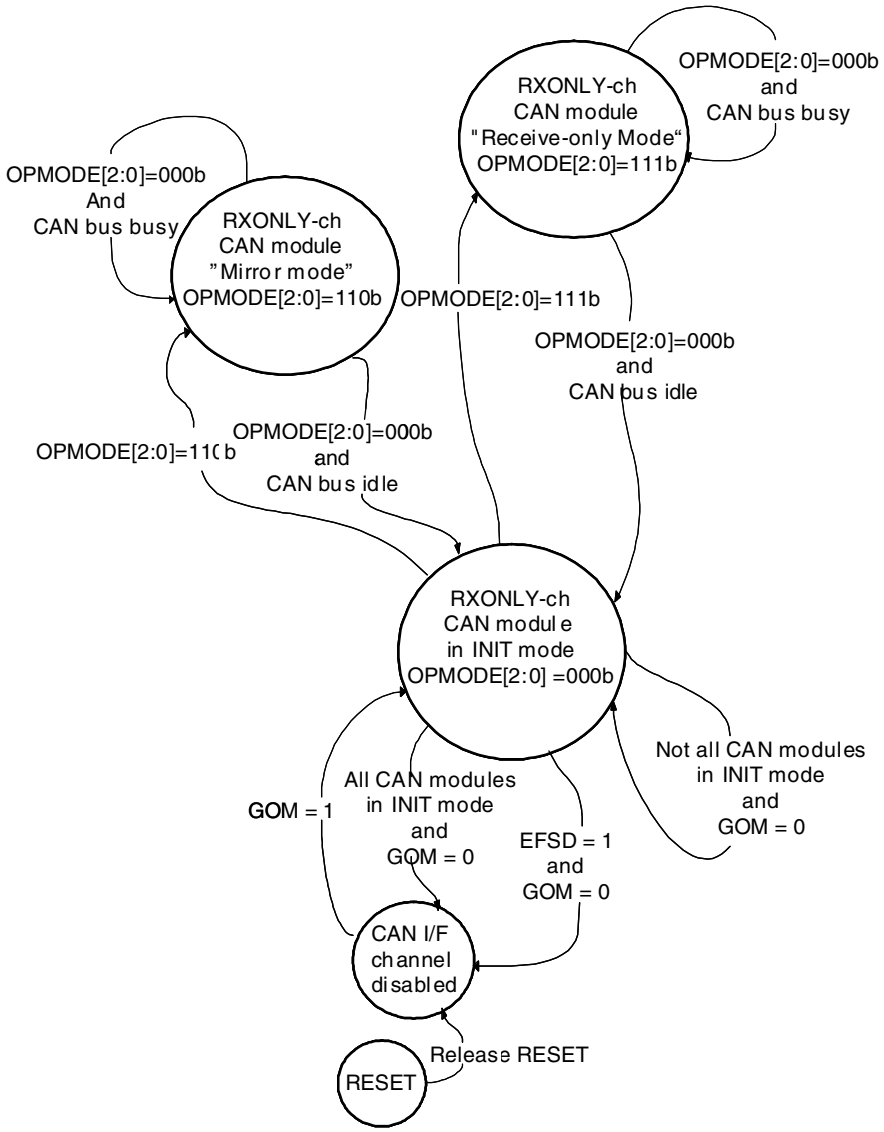
The transitions from INIT mode to the operational modes is controlled by the bit-string OPMODE[2:0] in the CnCTRL register.
Changing from one operational mode into another operational mode requires to transit into INIT mode intermittently. The CAN module refuses CPU attempts to change from one operational mode into another operational mode 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 has to be kept until when 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 bit-string OPMODE[2:0] value changes to 00H. The CPU has to confirm the proper transition into INIT mode by reading the OPMODE[2:0] bit-string until OPMODE[2:0] = 00H.

When a successful receive interrupt is detected for a specific CAN module that already entered INIT mode, this interrupt was generated by a reception process that coincided with the request of INIT mode by the CPU. The received message that caused this interrupt was still stored before INIT mode was becoming valid.

(3) Transition for Operational Modes of RXONLY-CH

The RXONLY-H CAN machine can be switched the following operational modes:
“Mirror Mode”
“Mirror Mode with Transfer ID Filter function (w/ TIF)”
“Receive-only Mode”

Figure 14-17: Transitions for operational modes of the RXONLY_CH CAN module



The transitions from INIT mode to the operational modes is controlled by the bit-string OPMODE[2:0] in the CnCTRL_R register.
The rules for changing from one to another operational mode are the same as for the DIAG-CH.

14.5 Macro Interrupts

The DIAG macro provides 12 different interrupt source events. The occurrence of those interrupt source events is stored in interrupt status registers. Four separate interrupt request signals are generated from the 12 source events. The signals are routed to the interrupt controller in the microcomputer system. In case the interrupt controller in the microcomputer system does not provide a sufficient number of interrupt request signal inputs, the 4 interrupt request signals of a CAN module can be grouped (i.e. logical OR). With help of the interrupt status register the user can determine the actual interrupt source event for a particular interrupt. After determination of the interrupt source event the user must clear the corresponding interrupt status bit.

Table 14-6: List of all macro interrupt sources

#	Interrupt Status Bit		Interrupt Enable Bit		Interrupt Request Signal	Interrupt Source Description
	Name	Register	Name	Register		
1	CINTS0	CnINTS	CIE0[Note]	CnIE	INTTRX	CAN module interrupt status bit for interrupt event 'Message frame successfully transmitted from message buffer m' by the DIAG-CH. The interrupt is also provided for mirrored messages.
2	CINTS1	CnINTS	CIE1[Note]	CnIE	INTREC	CAN module interrupt status bit for interrupt event 'Valid message frame reception in message buffer m' from the DIAG-CH.
3	CINTS1	CnINTS_R	CIE1[Note]	CnIE_R		CAN module interrupt status bit for interrupt event 'Valid message frame reception in one of the upper 16 message buffers from the RXONLY-CH.
4	CINTS2	CnINTS	CIE2	CnIE	INTERR	CAN module error state interrupt status of the DIAG_CH
5	CINTS2	CnINTS_R	CIE2	CnIE_R		CAN module error state interrupt status of the RXONLY_CH
6	CINTS3	CnINTS	CIE3	CnIE		CAN module protocol error interrupt status of the DIAG-CH
7	CINTS3	CnINTS_R	CIE3	CnIE_R		CAN module protocol error interrupt status of the RXONLY-CH
8	CINTS4	CnINTS	CIE4	CnIE		CAN module arbitration loss interrupt status of the DIAG-CH
9	CINTS5	CnINTS	CIE5	CnIE	INTWUP	DIAG-CH CAN machine wake-up interrupt status bit from SLEEP mode by CAN bus
10	CINTS5	CnINTS_R	CIE5	CnIE_R		RXONLY-CH CAN machine wake-up interrupt status bit from SLEEP mode by CAN bus
11	CINTS6	CnINTS_R	CIE6	CnIE_R	INTERR	The buffer overflow interrupt status bit, which is set 1 when the RXONLY-CH CAN machine fails to store a message because the upper 16 message buffer are all occupied (i.e. DN <CnLIPT_R +1> = 1).
12	CINTS7	CnINTS	CIE7	CnIE	INTTRX	Signals a completed transmission for upper 16 message buffer during Mirror Mode. The status bit and CIE7 are only meaningful when RXONLY_CH is operated in Mirror mode. CIE7 has no function when CIE0 in CnIE is cleared.

Note: The IE bit (message buffer interrupt enable bit) in the MCTRL register must set (1) for the message buffers, which shall participate in the interrupt generation process.

Figure 14-18: Schematic of the CAN macro interrupt generation (1/2)

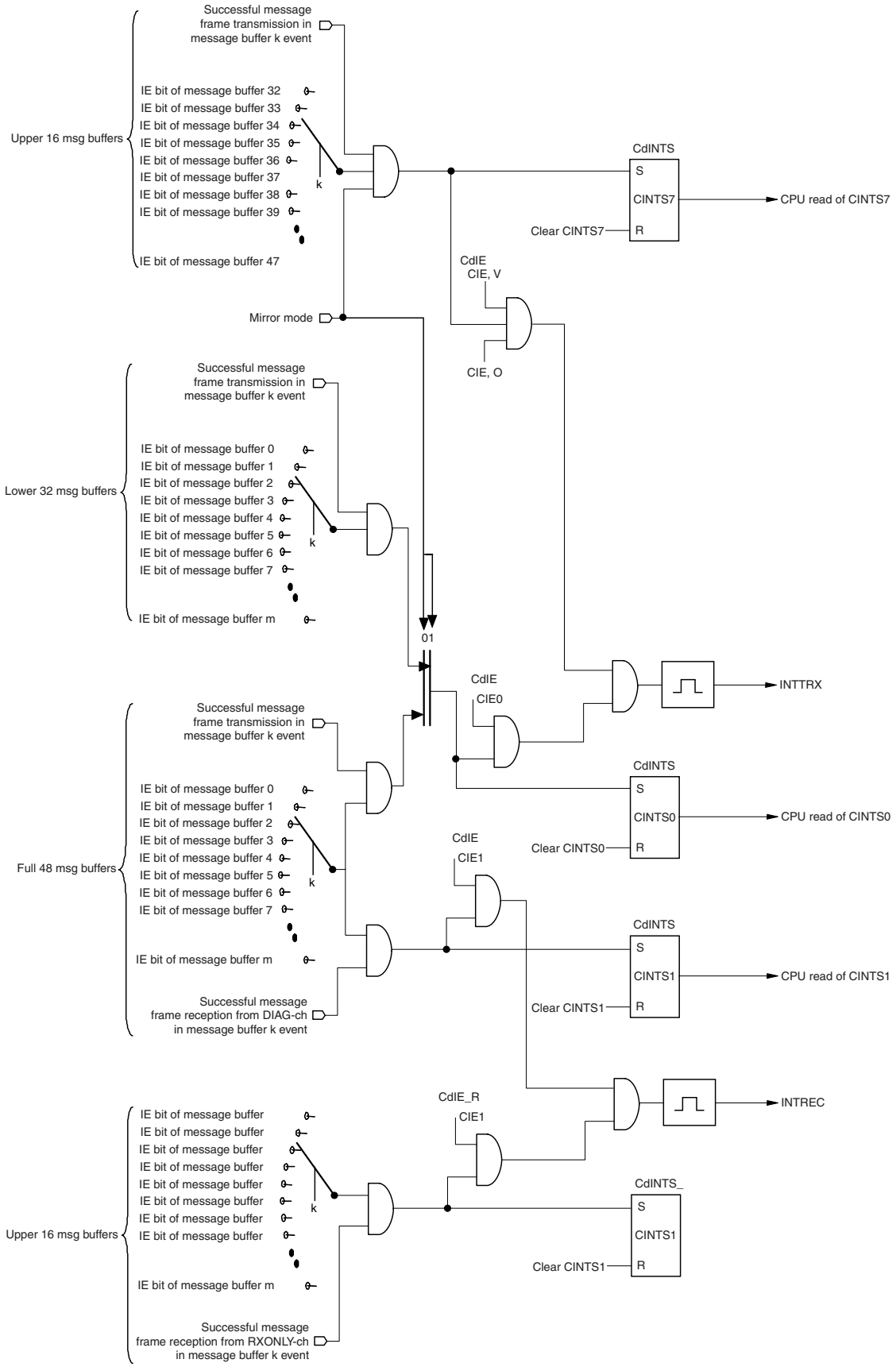
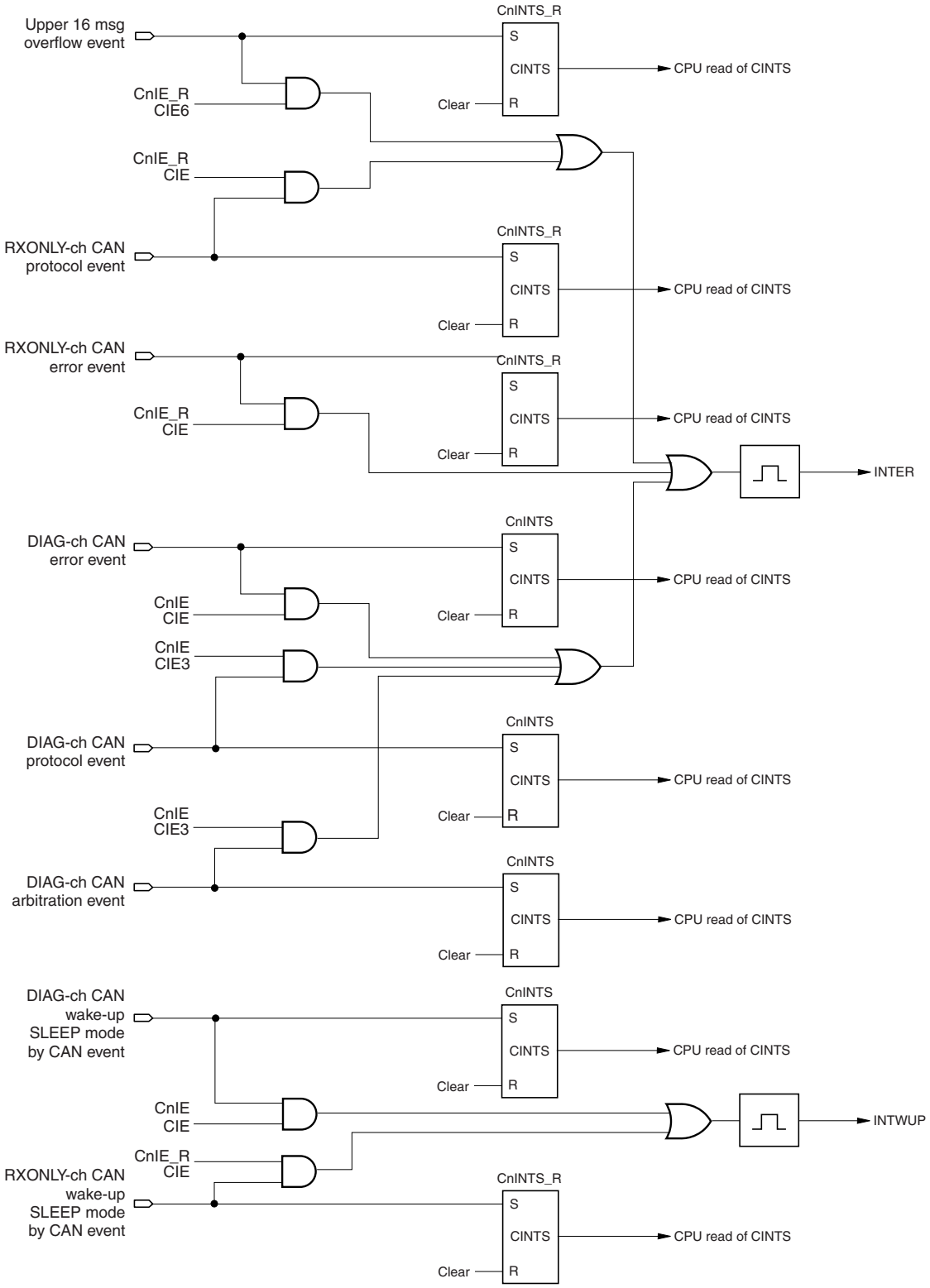


Figure 14-18: Schematic of the CAN macro interrupt generation (2/2)



14.6 Message Reception

14.6.1 Principal reception process

In general all operations linked to acceptance filtering are only available on the DIAG_CH. For the RXONLY_CH only simple receive operations apply. These are mentioned explicitly. If not mentioned, the descriptions in chapter 3.5 only apply to the DIAG-CH reception operations. Details of the RXONLY_CH reception process are found in chapter 14.8 ”Operational Modes of RXONLY_CH” on page 684.
In all operational modes of the CAN module every data frame, which is received from the CAN bus in the CAN module of a CAN I/F channel without an error (i.e. valid reception), is stored in the Temporary Receive Buffer.
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, which fulfil the below listed criterions, whether the received data frame has to be stored in one of the message buffers (i.e. data frame acceptance filtering):

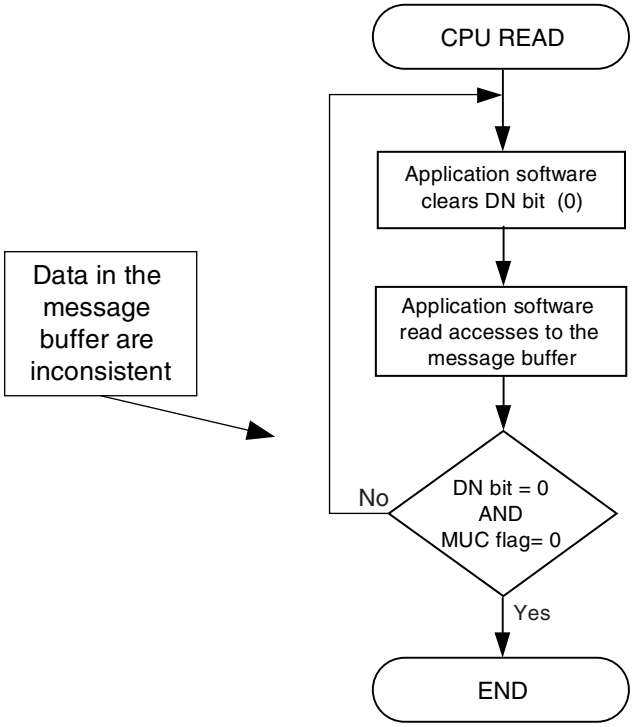
- message buffer has to be assigned to the CAN I/F channel, which received the data frame (MA[2:0] bit-string in MCONFm register)
- message buffer has to be configured as a receive message buffer (MT[2:0] bit-string in MCONFm register has to hold the values 01H, 02H, 03H, 04H or 05H)
- message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in MCTRLm register).

For message buffers configured as a receive message buffer without a link to a mask the identifier (standard identifier = ID[28:18] / extended identifier ID[28:0]) of the received data frame and the identifier MIDLm and MIDHm registers (standard identifier = ID[28:18] / extended identifier ID[28:0]) in the message buffer are compared. In case the identifiers are equal, the received data frame has to be stored in the particular message buffer.
For message buffers configured as a receive message buffer with a link to a mask the identifier (standard identifier = ID[28:18] / extended identifier ID[28:0]) of the received data frame is logical combined with the linked mask (bit-wise OR) and the identifier in the MIDLm and MIDHm registers of the message buffer (standard identifier = ID[28:18] / extended identifier ID[28:0]) itself is logical combined with the linked mask (bit-wise OR). The results of both logical combinations are compared (bit-wise Exclusive-NOR) to test whether the received data frame has to be stored in the particular message buffer.

Identifiers with extended format and identifier with standard format are not equal, even when the 11-bit of a standard format identifier do match with the 11 most significant bits of an extended format identifier. The IDE bit in the MIDHm register of each message buffer determines the identifier format (IDE cleared (0) identifier is standard format; IDE set (1) identifier is extended format).
Upon the acceptance of a received data frame in a receive message buffer of any type (MT[2:0] = 01H, 02H, 03H, 04H, 05H) the content of MIDLm and MIDHm registers, MDLCm register and MDATA0m to MDATA7m are updated according the values in the received data frame. If the received data frame contains less than 8 data bytes (i.e. data length code < 8), the data bytes in the message buffer, which are not updated, contain undefined values after the acceptance of the received data frame.
As a result of accepting a received data frame into a message buffer of any type the Data New bit (DN bit) is set (1) in case the message buffer has been marked as not-occupied (DN bit cleared (0) before message reception). In case the matching message buffer is marked occupied by DN bit set (1), it can be selected whether the received data frame overwrites the content of MIDLm and MIDHm registers, MDLCm register and MDATA0m to MDATA7m registers in the message buffer or the received data frame is discarded. The setting of OWS bit in MCONFm register determines whether a newly received data frame overwrites an occupied message buffer or the newly received data frame is discarded. A message buffer overwriting is indicated by automatically setting the MOW bit in the overwritten message buffer (1).
When overwriting is chosen for an occupied message buffer (i.e. OWS bit in MCONFm register set (1)) the DN bit in conjunction with the Message Buffer Under Change (MUC) bit can be used by the application software as semaphores to achieve data consistency. Every message buffer has a MUC bit. The

CAN module sets (1) it before updating a particular receive message buffer and resets (0) it after the update process finished. The flowchart below shows how application software has to read a message buffer in a proper way.

Figure 14-19: Message buffer reading with semaphore handling



Upon acceptance of a received data frame into a message buffer also the corresponding ‘valid data frame reception’ interrupt status bit CINTS1 in the CnINTS register is set (1). In addition an interrupt request signal will be released, when it has been enabled by setting the CIE1 enable bit in the CnIE register (1).

Under the following situations it becomes necessary to decide in which message buffer a received data frame has to be accepted:

- More than one message buffer with the same type, specified by the MT[2:0] bit-string, has been configured for a CAN module in a CAN I/F channel (i.e. multi-buffer receive block MBRB)
- The identifier of the received data frame matches with the identifier of a message buffer without a link to a mask and can be accepted into message buffers with a link to a mask.
- The received data frame can be accepted into message buffers with a link to different masks.

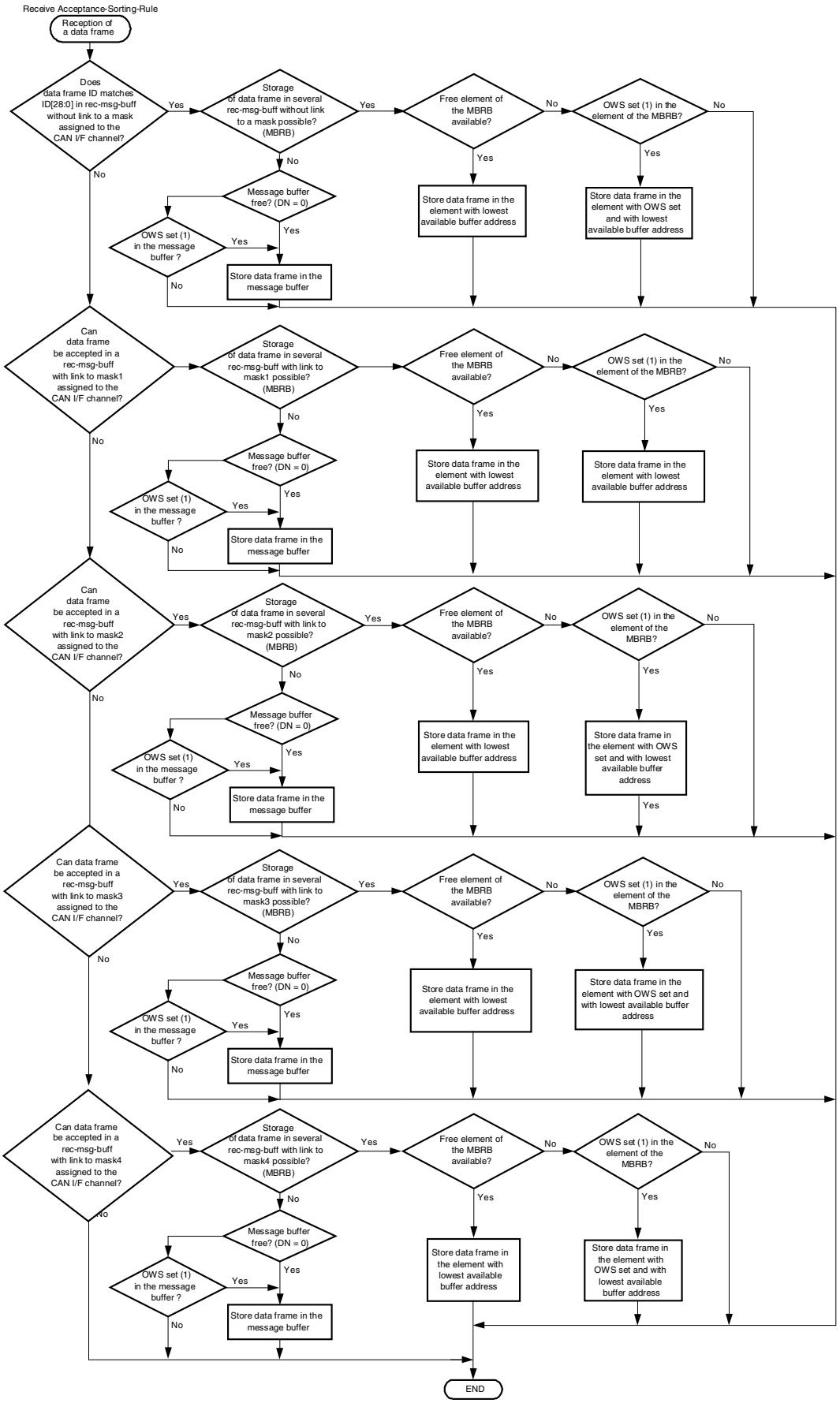
The Receive Message Acceptance Filter Machine uses the following rule for message acceptance and message sorting (priority for a particular message buffer) to avoid conflicts and to store a received message frame unambiguously.

Caution: The following flowchart does not show the scan algorithm in the Receive Message Acceptance Filtering machine, but it shows in which message buffer a received data frame is stored in case storage in several message buffers is possible

For the RXONLY-CH the acceptance filtering is obsolete. Any valid message will be stored in the upper 16 message buffer of the DIAG macro when Mirror Mode, mirror mode w/ TIF or RecOnly Mode is selected. In both modes the RXONLY-CH will store the message in the next higher buffer number where it finds a buffer with its DN bit cleared. At buffer #47 is wraps around to buffer #32.
In RecOnly Mode the application needs to read the messages and clear the DN bits in time to avoid the generation of the interrupt CINTS6 in CnINTS_R.

In Mirror Mode the reception process will also set the DN bit but no receive interrupt is generated. Rather another state machine is triggered to evaluate if the TRQ of this message buffer can be set, and thus launch the transmission of the message on the DIAG-CH. The state machine handling the setting of TRQ´s for the upper 16 message buffers is activated every time the DIAG-CH or the host CPU clears a TRQ bit. Even TRQ bits for the lower 32 message buffers need to be considered. When the state machine detects that the TRQ for the message buffer it flagged for transmission the last time was cleared, the TRQ for the next buffer is set unless the history list implies no pending entry.
As the DN bit is used by the RXONLY-CH to detect an empty (already transmitted) buffer, the DN bit is cleared automatically after the message was transmitted successfully on the DIAG-CH.

Figure 14-20: Message-Acceptance-Sorting-Rule



(1) Reception History

The CAN module in each CAN I/F channel provides a Reception History List (RHL) supporting the application software to find the message buffer, in which the last data frame or the last remote frame was stored. Further, with help of the RHL the application software can trace whether several message frames have been received since the application software has evaluated the Central Message Buffer Memory for received message frames the last time.

The RHL consists of 23 elements (23 bytes memory) and two pointers, the LIPT pointer (Last In-message Pointer) with the corresponding CnLIPT register and the RGPT pointer (Receive History List Get Pointer) with the corresponding CnRGPT register. For RXONLY-CH LIPT_R in CnLIPT_R register applies. A CnRGPT-R register is not provided because the sequence of stored messages is fixed by the ring buffer algorithm and thus known a priory.

The RHL-elements themselves contain the message buffer numbers, in which the CAN module stored data frames or remote frames. After the transition from INIT mode to one of the operational modes the RHL-elements contain undefined values.

Upon data frame or remote frame storage the corresponding message buffer number is written to the RHL-element, to which the LIPT pointer is pointing. After writing the message buffer number to the RHL-element the LIPT pointer is incremented by the CAN module. By default the LIPT pointer points to the RHL element with the lowest address (i.e. the first element). When the last RHL-element is filled with a message buffer number, the LIPT pointer is reset and points again to the RHL-element with the lowest address. Upon the next storage the content of the first RHL-element is overwritten by the number of the actual message buffer (i.e. continues ring behaviour).

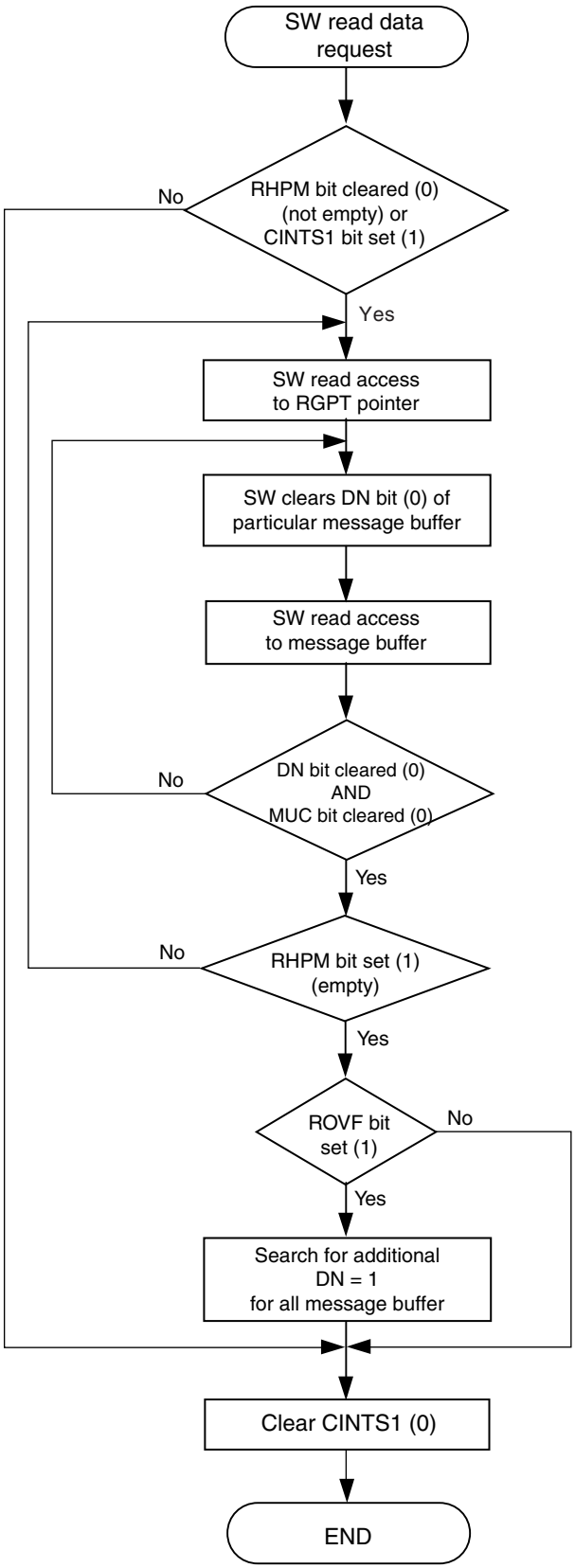
The RGPT pointer points to the RHL-element, which contains the message buffer number that has to be read by the CPU next. By default the RGPT pointer also points to the RHL-element with the lowest address. After CPU has read the CnRGPT register to get the message buffer number of the message buffer, which has to be read next, the RGPT pointer is automatically incremented to point to the next RHL-element. The automatic increment of the RGPT pointer is suspended when the RGPT pointer points to the same RHL-element as the LIPT pointer does.

The RHPM bit (Receive History List Pointers Match) in the CnRGPT register is set (1) whenever RGPT pointer will match the LIPT pointer. RHPM set (1) signals the application software that no more message frames has been received.

The ROVF bit (Receive History List Overflow) is set from the CAN module whenever the LIPT pointer points to the RGPT pointer -1.

Note that the LIPT_R value is maintained throughout the PSMODE = STOP for RXONLY_CH. This register is only cleared when entering OPMODE = INIT.

Figure 14-21: Message reception procedure using the Reception History List



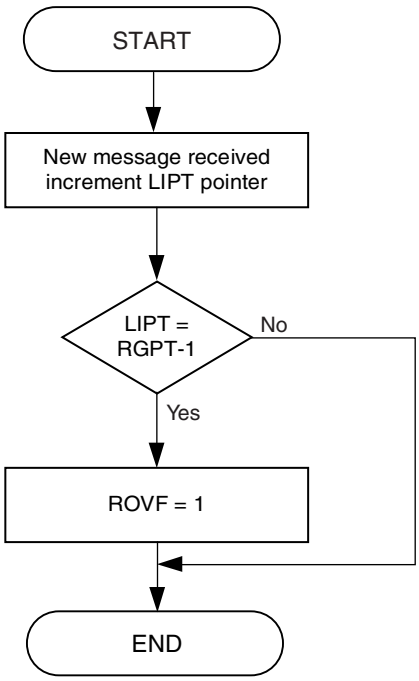
After the transition from INIT mode to one of the operational modes the LIPT pointer and the RGPT pointer are pointing both to the first RHL-element and RHPM bit is set (1).

The CPU has read-only access to the CnLIPT register. Reading the CnLIPT register delivers directly the content of the RHL-element, which contains the message buffer number of the receive message buffer in which the last data frame was stored or the number of the transmit message buffer to which the last remote frame was assigned. Internally the LIPT pointer is always pointing to the RHL-element to which the message buffer number of the next storage/assigned process has to be saved. Therefore, when the CPU reads the CnLIPT register, the RHL-element one before the element the LIPT pointer is pointing (temporary decrement of LIPT pointer) is accessed.
The CPU has read-only access to the CnRGPT register. Read access to the CnRGPT register delivers directly the content of the RHL-element, to which the RGPT pointer is pointing.

The RHL provides a mechanism to prevent the LIPT pointer overtaking the RGPT pointer. In case the RGPT pointer remains pointing to a certain RHL element (e.g. RHL element i), incoming message frames are logged into the RHL as long the LIPT pointer is not pointing itself to the same RHL element (i.e. RHL element i) as the RGPT pointer does. When the RHL element (i-2) two position before the element to which the RGPT pointer is pointing is written by the message buffer number involved in the last acceptance event, the LIPT pointer is automatically incremented and does now point to the i-1 RHL element as the RGPT pointer does. That event causes the following actions:

- The ROVF bit is set (1)
- The element where the LIPT-1 pointer is pointing to, is updated with every more received message. But the LIPT pointer is not incremented until the application software clears the ROVF bit (0).
- Full read access to the RGPT pointer

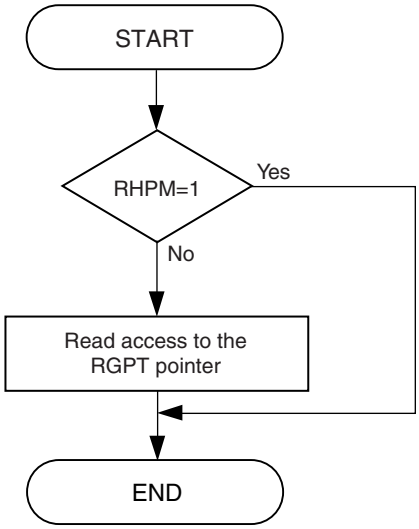
Figure 14-22: ROVF setting depending of the LIPT pointer and the RGPT pointer



Caution: In case the ROVF bit is set (1) the newly received messages will update the element the LIPT-1 pointer is pointing to, but the this pointer does not increment. Therefore in case that more messages are received after ROVF bit is set (1), there will be no chronological consistency.
The RGPT is not updated by the reception from the RXONLY-CH.

The RHPM bit indicates, if one or more messages are unread in the RHL. Therefore the user has to check before accessing the RGPT pointer the status of the RHPM bit.
The RHPM bit is set (1) whenever the RGPT pointer matches the LIPT pointer.

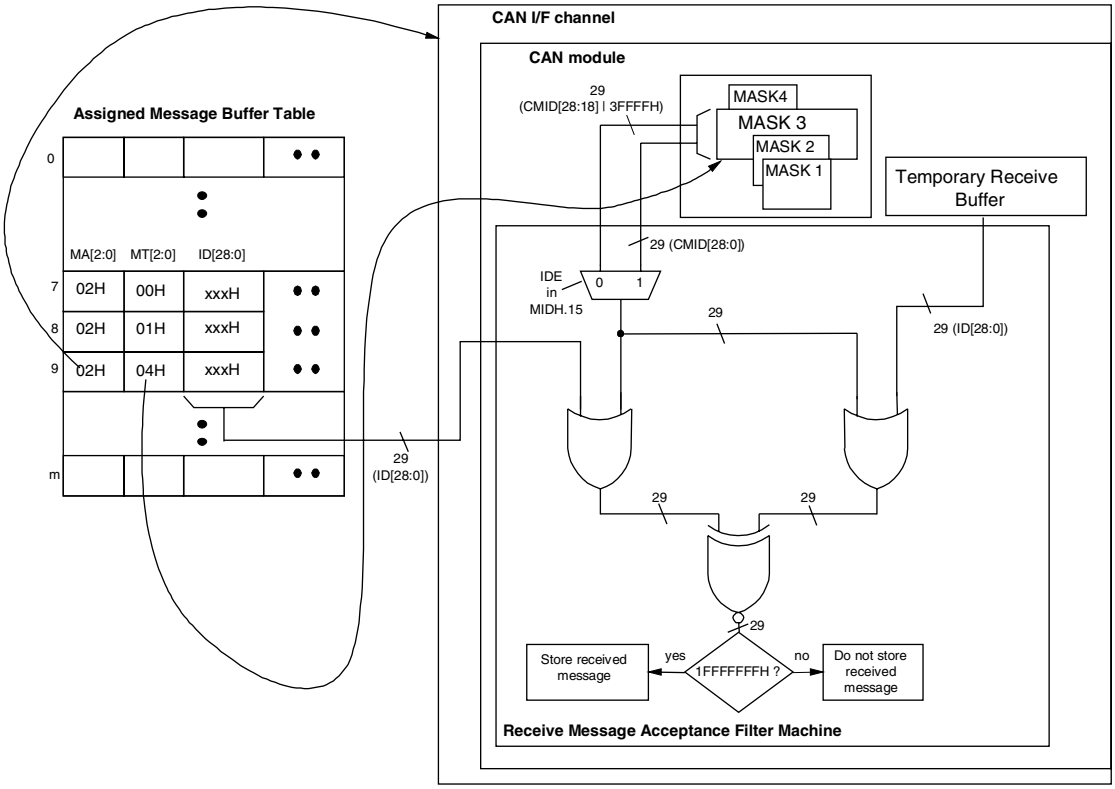
Figure 14-23: RGPT pointer handling with respect to RHPM bit



(2) Masked Receive Message Buffer

Every message buffer can be linked to one of the 4 masks in the assigned CAN module of the CAN I/F channel (bit-string MT[2:0]).
The identifier value programmed to the MIDLm and MIDHm registers of the message buffer and the value of the linked mask determine the data frames, which are accepted from the CAN bus into the particular message buffer.

Figure 14-24: Example on acceptance filtering for a message buffer with a link to a mask



Remark: The schematics in Figure 14-24 do not refer to the actual implementation rather they are chosen for explanation purposes. Especially the gates do not need to correspond to the gates of the macro development.

In the example shown in Figure 14-24 the message buffer 9 is assigned to the CAN I/F channel 2 (MA[2:0]=02H). The message buffer type of message buffer 9 is defined as receive message buffer with link to MASK 3 in the CAN module of the CAN I/F channel (MT[2:0]=04H).
Upon each evaluation, which is processed in the Receive Acceptance Filtering Machine, a received data frame will be accepted for message buffer 9 if the logical combination of Mask 3 and the identifier ID[28:0] in MIDL9 and MIDH9 compared to the identifier in the Temporary Receive Buffer in the CAN module of CAN I/F channel 2 delivers a match. The combination is a logically OR function applied bit-wise.
According to the principle:

$$X \vee 1 = 1$$

the bit-positions in the identifier which don't have to be evaluated for equivalence are set (1) in Mask3.

According to the principle:

$$X \vee 0 = X$$

the bit-positions in the identifier, which have to be evaluated for equivalence, are cleared (0). As a result only the bit-positions, which are not masked (i.e. the bit-positions in the Mask 3 cleared (0)), are evaluated in the final comparison by the logical bit-wise Exclusive-NOR function.
Since the CAN protocol allows two different types of identifier (i.e. 11-bit Standard identifier or 29-bit Extended identifier), the bit-width for the logical combinations in the receive message acceptance filtering has to match according the actual selected identifier type.

Caution: MA bits are not used for multi-channel assignment; rather MA has to be initialised for any AFCAN/DAFCAN module to 0x01.

14.6.2 Reception into a multi-buffer receive block

Receive message buffers, which are assigned to the same CAN I/F channel and are defined as an equivalent message buffer type, behaves as a Multi-buffer Receive Block (MBRB).
The particular reception behaviour for the message buffers in the MBRB is equal to the reception behaviour of a stand-alone receive message buffer.
By the configuration of the IE bit in the MCTRLm register of the message buffers the user can decide upon which data frame reception the CPU has to be informed that a data block has been received completely. In case a data block consists of k messages the user could initialise k message buffers for the reception of the data block. In the message buffers 0 to (k-1) the IE bits are cleared (0) (i.e. interrupt disabled) and in the message buffer k the IE bit is set (1) (i.e. interrupt is enabled).
In case the user wants to use the MBRB just as a back-up buffer other configurations of the IE bits are applicable (e.g.: using k message buffers in a MBRB, clear IE bit (0) in all message buffers except in message buffer (k-1) -> upon the data frame reception in message buffer (k-1) the user is warned that the MBRB is almost full and might overflow).
Setting of the OWS bit in the MCONFm register in each message buffer belonging to a MBRB depends on the requirement of the particular application.
The user has not to clear any RDY bit (0) within the MBRB block to ensure the full functionality of this feature. For example, if a message will be received in one of the buffers belonging to the MBRB block and exactly this buffer is disabled (RDY bit of this particular buffer cleared (0)) the message will neither be stored in this buffer nor in any other buffer of the MBRB block.

14.6.3 Reception of remote frames

The reception of remote frames applies to both, the DIAG-CH and the RXONLY-CH. However, for RXONLY-CH message storing follows different rules and acceptance filtering is not available.

In all operational modes of the CAN module every remote frame, which is received from the CAN bus in the CAN module of a CAN I/F channel without an error (i.e. valid reception), is stored in the Temporary Receive Buffer.
In the operational modes “Normal Operating”, “Normal Operating with ABT”, “Receive-only Mode”, “Single-Shot Mode” and “Self-test Mode” the Receive Message Acceptance Filtering Machine evaluates all message buffers, which fulfil the below listed criterions, whether the received remote frame has to be accepted.

- message buffer has to be assigned to the CAN I/F channel, which received the remote frame. (MA[2:0] bit-string in MCONFm register])
- message buffer has to be configured as a transmit message buffer (MT[2:0] bit-string in MCONFm register has to hold the value 00H)
- message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in MCTRLm register)
- RTR bit in the transmit message buffer has to be cleared (0)

Upon acceptance of a remote frame the following actions are executed in case the identifier of a received remote frame matches the identifier of a message buffer, which fulfils the above mentioned criterions:

- The MDLC[3:0] bit-string in the MDLCm register is overwritten by the DLC value of the received remote frame.
- The data bytes MDATAn are not updated - the previous data retains.
- The DN flag is set (1)
- The interrupt status bit CINTS1 in the CnINTS register is set (1)
- When CIE1 bit in the CnIE register is set (1), the interrupt request signal INTRECn in 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 MCONFm register has no meaning for the acceptance of a remote frame. It means the remote frame is accepted regardless the OWS bit is cleared (0) or set (1) and the DN flag has already been set (1).

In case 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.
A remote frame reception on the RXONLY-CH is handled as a normal data frame reception into the upper 16 message buffers without any acceptance filtering. The message will be stored according the ring buffer method.

14.7 Message Transmission

This chapter mainly refers to the DIAG_CH as the RXONLY_CH does only receive messages. However, specific transmit operations by the DIAG_CH that are linked to the mirror mode of the RXONLY_CH are explained here as well.

14.7.1 Principal transmission process

In the operational modes of the DIAG_CH “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 (1) in a message buffer, which fulfils the following criterions:

- The message buffer has to be assigned to the particular CAN I/F channel (MA[2:0] bit-string in MCONFm register])
- The message buffer has to be configured as a transmit message buffer (MT[2:0] bit-string in MCONFm register has to hold the value 00H)
- The message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in MCTRLm register)

The Transmit Message Search Machine compares the transmit message buffers with the new transmit request against transmit message buffers with pending transmit requests (i.e. message buffers, in which TRQ bit was set (1) beforehand).
In case the new transmit request gets the highest priority, the Transmit Message Search Machine tries to overwrite the Temporary Transmit Buffer with the message frame, which is stored in the corresponding message buffer. An overwriting is only possible as long as the transmit process has not been started for the message frame currently occupying the Temporary Transmit Buffer. If the overwriting of the Temporary Transmit Buffer is impossible, the new transmit request is served at a later point in time.
The highest priority is determined according the following rules:

- **Priority 1 (highest):**
  11 MSB Identifier Value rule [ID28:ID18]:
  The first 11 most significant bits of the identifier (i.e. ID28 to ID18) are the highest prior criteria 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 identifier have a higher priority than message frames with 29-bit extended identifier in case the value of the 11-bit standard identifier is equal or smaller than the 11 most significant bits (11 MSB) of the 29-bit extended identifier.

- **Priority 2:**
  Frame Type rule:
  When the Priority 1 rule does not provide an unambiguous result, because the 11 most significant bits of the identifier are equal, 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 are the next criteria to judge which message frame has to be sent first. In this case data frames with 11-bit standard identifier (i.e. RTR bit cleared (0)) have higher priority than remote frames with standard identifier and message frames with extended identifier.

- **Priority 3:**
  Identifier Type rule:
  If even Priority 2 rule can not 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 higher priority than a message frame with extended identifier.

- **Priority 4:**
  18 LSB Identifier Value rule [ID17:ID0]:
  The next criteria to find the message, which 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. Priority 4 rule does apply in case two extended identifier message frames are pending for transmission, which have equal values in the 11 most significant bits of the identifier and have the same frame type (RTR bit value is same).

- **Priority 5 (lowest):**
  Message Buffer Number
  The last criteria to find the message, which 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.

If the new transmit request does not have the highest priority, it will be served at a later point in time.

Remark: In case 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 (1) at a time. According the above mentioned rules this ABT-buffer competes with transmit message buffers, which do not belong to the ABT message buffers and have their TRQ bit set (1).
Within the ABT message buffers a fixed order determines which message buffer is sent next.

The Transmit Message Search Machine is triggered by the following events:

Application software trigger events:

- The transmission request was initiated by application software (TRQ set (1))
- The application software initiates an abort process for a previously submitted transmission request.

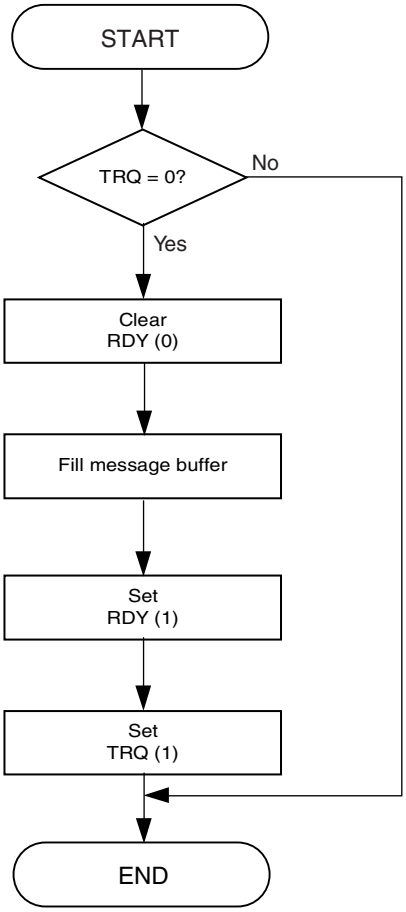
CAN module trigger events:

- The transmit process by the CAN Protocol Transfer Layer signals a successful finish of a message transmission.
- A transmission process is interrupted by a transmit error.

Upon a 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 CnINTS register is set (1). In addition an interrupt request signal will be released, when the CIE0 bit in the CnIE register was set (1) beforehand.

The transmission shall be invoked using the following flowchart:

Figure 14-25: Transmit preparation



Caution: RDY and TRQ of this particular message buffer have not to be set at the same time. First RDY has to be set and than after that TRQ. Otherwise the functionality of that buffer can not be guaranteed.

14.7.2 Transmit history

The DIAG-CH CAN module channel provides a Transmission History List (THL) supporting the application software to find the message buffer, from which the last data frame or the last remote frame was sent. Further, with help of the THL the application software can trace whether several message frames have been sent since the application software has evaluated the Central Message Buffer Memory for transmitted message frames the last time. It shows the history of all transmitted messages without any distinction between those sent by the mirror mode and those belonging anyway to the DIAG-CH (i.e. buffer #0 -#31).
The THL consists of 7 elements (7 bytes of memory) and two pointers, the LOPT pointer (Last Out-message Pointer) with the corresponding CnLOPT register, and the TGPT pointer (Transmit History List Get Pointer) with the corresponding CnTGPT register.
The THL-elements themselves contain the message buffer numbers, from which the CAN module has sent message frames. After the transition from INIT mode to one of the operational modes the THL-elements contain undefined values.
Upon successful transmission the corresponding message buffer number is written to the THL-element, to which the LOPT pointer is pointing. After writing the message buffer number to the THL-element the LOPT pointer is incremented by the CAN module. By default the LOPT pointer points to the THL-element with the lowest address (i.e. the first element). When the last THL-element is filled with a message buffer number, the LOPT pointer is reset and points again to the THL-element with the lowest address. Upon the next successful transmission the content of the first THL-element is overwritten by the number of the actual message buffer (i.e. continuous ring behaviour).
The TGPT pointer points to the THL-element, which contains the message buffer number that has to be read by the CPU next. The application software uses that information to manage finding a vacant transmit message buffer. By default the TGPT pointer also points to the THL-element with the lowest address.

After the CPU has read the TGPT pointer to get the message buffer number of the message buffer, which can be filled with new data, the TGPT pointer is automatically incremented to point to the next THL-element. The automatic increment of the TGPT pointer is suspended when the TGPT pointer points to the same THL-element as the LOPT pointer does.
The THPM bit (Transmit History List Pointers Match) in the CnTGPT register is set (1) whenever TGPT pointer and the LOPT pointer match. THPM set (1) signals the application software that no more message frames have been sent.
After the transition from INIT mode to one of the operational modes the LOPT pointer and the TGPT pointer pointing both to the first THL-element and THPM bit is set (1). Reading one of the pointers will deliver undefined values until THPM bit is cleared (0).

The CPU has read-only access to the CnLOPT register. Reading the CnLOPT register delivers directly the content of the THL-element, which contains the message buffer number of the transmit message buffer, from which the last message frame was sent. Internally the LOPT pointer is always pointing to the THL-element, to which the message buffer number of the next transmission process has to be saved. Therefore, when the CPU reads the CnLOPT register, the THL-element one before the element the LOPT pointer is pointing to (temporary decrement of the LOPT pointer) is accessed.
The CPU has read-only access to the CnTGPT register. Read access to the CnTGPT register delivers directly the content of the THL-element, to which TGPT pointer is pointing.

When the CAN module is operating in “Normal Operating Mode with Automatic Block Transmission” successful transmission from the message buffers assigned to the ABT function (ABT = Automatic Block Transmission) are not considered by the THL function. In short, any transmission sent from the message buffer 0 to message buffer 7 is not logged into the THL.

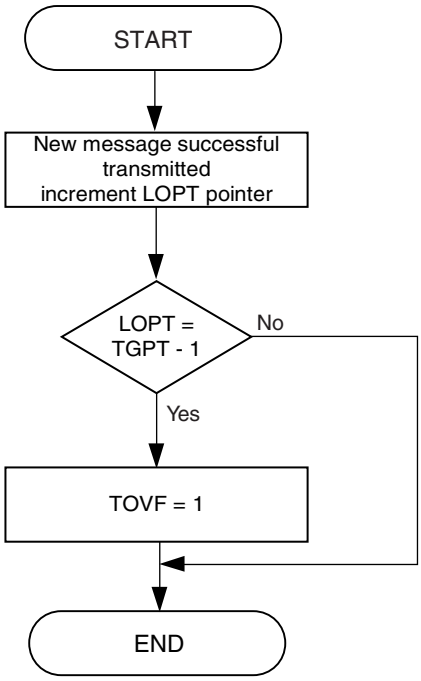
The THL provides a mechanism to prevent the LOPT pointer overtaking the TGPT pointer. In case the TGPT pointer remains pointing to a certain THL element (e.g. THL element i), successful transmissions of messages are logged into the THL as long the LOPT pointer is not pointing to the same THL element (i.e. THL element I again) as the TGPT pointer does. When the THL element (i-2) two position before the element to which the TGPT pointer is pointing is written by the message buffer number involved in

the next successful transmission event, the LOPT pointer is automatically incremented and does now point to the i-1 THL element as the TGPT pointer does.

That event causes the following actions:

- The TOVF bit is set (1)
- The element where the LOPT-1 pointer (LOPT temporary decrement) is pointing to, is updated with any successfully transmitted message from now on until the CPU clears the TOVF bit. Also the LOPT pointer is not incremented until the application software clears the TOVF bit (0).
- Full read access to the TGPT pointer.

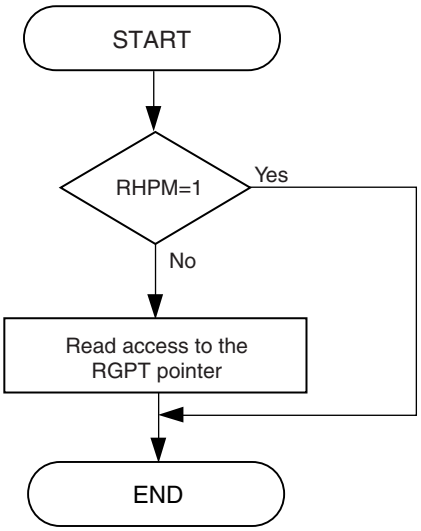
Figure 14-26: TOVF setting depending of the LOPT pointer and the TGPT pointer



Caution: In case the TOVF bit is set (1), new successfully transmitted messages will update the element the LOPT-1 pointer is pointing to, but they do not increment this pointer. Therefore in case that more messages are successfully transmitted after TOVF bit is set (1), there will be no chronological consistency.

The THPM bit indicates, if one or more messages are unread in the THL. Therefore the user has to check before accessing the TGPT pointer the status of the THPM bit.
The THPM bit is set (1) whenever the TGPT pointer matches the LOPT pointer.

Figure 14-27: TGTP pointer handling with respect to THPM bit



14.7.3 Automatic block transmission (ABT)

For the transfer of large data blocks by transmitting subsequently data frames, several transmit message buffers, which are assigned to a CAN I/F channel, can be grouped for automatic block transmission (ABT).
Automatic block transmission is an operational mode of the DIAG_CH. The required initialisation is described in chapter 14.4.4 ”Configuration of a transmit message buffer” on page 650 and the operational mode transition is described in chapter (2)”Transitions for Operational Modes of DIAG-CH” on page 656.
Once all message buffers of the ABT group are loaded by the CPU, the transmission process is triggered by setting the ABTTRG bit (1). The internal ABT-engine will start to set the TRQ bit (1) in the first message buffer of the ABT group (i.e. message buffer 0). After a single data frame has been finished successfully the internal ABT-engine launches the data frame transmission from the next message buffer in the ABT group. The user can program a delay time (CGMABTD register), which has to elapse before the internal ABT-engine sets the TRQ bit (1) in the subsequent message buffer. The unit for the time delay is “DBT” (bit time) where the absolute value depends on the actual bit time settings determined by the CnBRP and CnBTR registers of the DIAG_CH. This kind of operation is selected by the mode “Normal Operating Mode with Automatic Block Transmission”.
The internal ABT-engine controls the transmission of data frames beginning at message buffer 0 towards message buffer 7 in an ascending order. Once the data frame from message buffer 7 has been sent successfully, the ABTTRG bit is cleared (0) automatically.
For loading data into the message buffers belonging to the ABT group, the RDY bit in MCTRLm register of each message buffer has to be cleared (0). Once a particular message buffer is completely written the RDY bit needs to be set (1) to mark the message buffer as been loaded.
The CPU has not to set the TRQ bit in the message buffer, otherwise there is the possibility that some messages will not be transmitted. The TRQ bit in the message buffer belonging to the ABT group are automatically set (1) by the internal ABT-engine when the particular message buffer has to be transmitted. The internal ABT-engine does not set the TRQ bit (1) in case it encounters a message buffer with the RDY bit cleared (0). In that case the internal ABT-engine stops to operate and clears the ABTTRG bit (0). Once the user sets the particular RDY bit (1), he can start the ABT-engine resuming to transmit from the message buffer element, which caused the stop, by setting the ABTTRG bit (1). In case the user does not want to resume transmission from the message buffer at which the ABT-engine stopped, he can reset the ABT-engine by setting the ABTCLR bit (1). Clearing the ABT-engine will force the ABT-engine to start again from message buffer 0 as soon as the user sets the ABTTRG bit (1). Only in one message buffer belonging to the ABT group the TRQ flag is set (1) at a time.
The TRQ flags are handled as for stand-alone (other regular) transmit buffers. Once the data frame has been sent successfully, the TRQ flag is cleared (0).
Interrupt handling can be used to inform the application that from all message buffers belonging to the ABT group the data frame has been successfully sent. The IE bit in the MCTRLm register in all message buffers of the ABT group have to be cleared (0), except in the last message buffer (i.e. in the end-element message buffer 7) it is set (1).
Further, the interrupt handling can be used to start reloading the message buffers, from which a data frame has already been sent. As an example, the IE bit in message buffer 4 could be set (1) as well. Upon the ‘transmit successful’ interrupt caused by the transmission from message buffer 4, the CPU could start to reload the message buffers 0 to 4.

Within the message buffers belonging to the ABT group no evaluation for the data frame with the highest priority is executed even in case the identifiers are not equal.
In case further transmit message buffers - not belonging to the ABT group - are assigned to a CAN I/F channel, which is configured for the ABT function, priority evaluation is performed among the buffers outside the ABT buffer range including the buffer area for the mirror mode and the ABT message buffer, which is currently pending for transmit (i.e. TRQ bit set (1)).

Caution: When using the ABT mode with a delay of 0 (CGMABTD register set to 0), it might be possible that a frame with lower priority than the next ABT frame is transmitted next on the bus. This can be caused by the setting of the TRQ for the next ABT frame, which invokes a TX search among all buffers. As this is not always finished within the IFS, the message buffer found from the last TX search will be transmitted on the bus.

The function of the Transmit History List (THL) is not applied to the ABT function. It means, that after successful transmission of data frames from a message buffer belonging to the ABT group, the buffer number is not logged in to the THL.

Before switching the CAN module from “Normal Operating Mode with Automatic Block Transmission” to the INIT mode the user has to reset all bits the CGMABT register to their initial values.

(1) ABT mode in conjunction with SLEEP mode

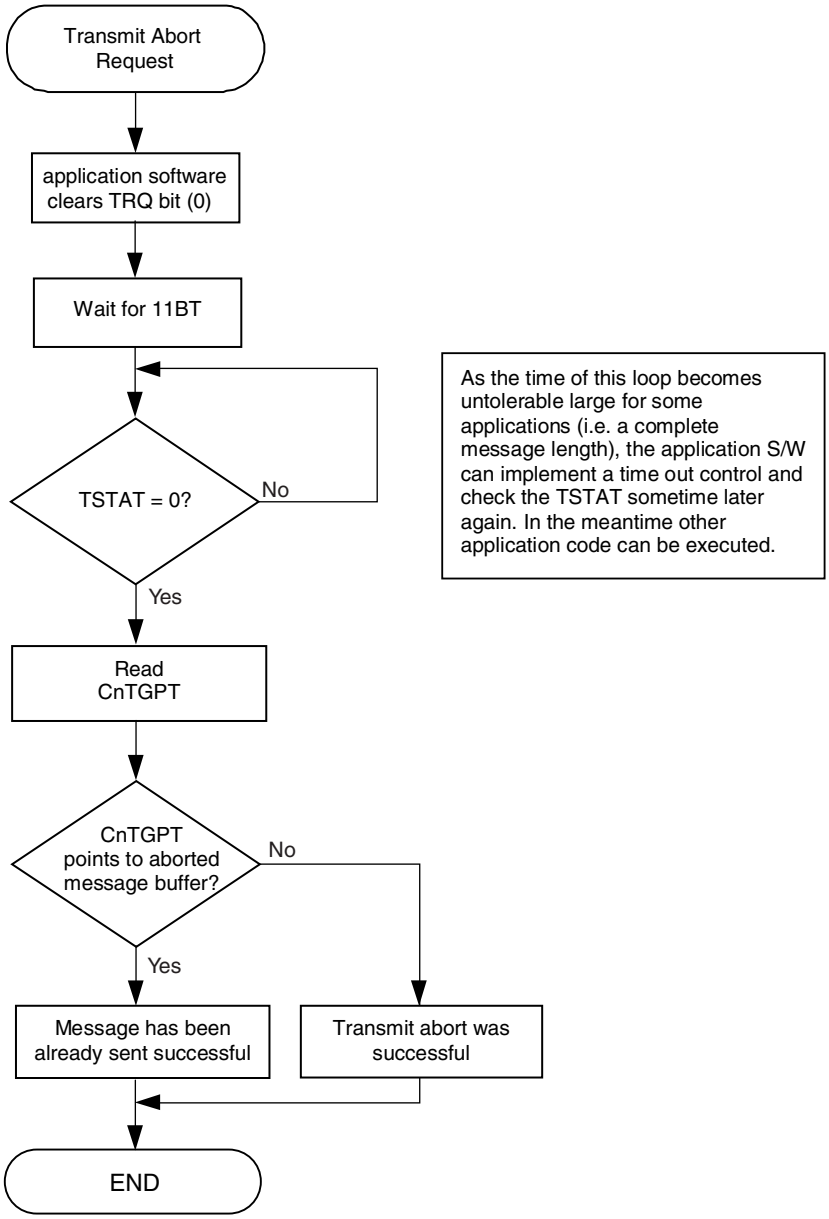
When the SLEEP mode of the CAN module is requested while the ABT mode is active (ABTTRG = 1), the CAN module transits to the SLEEP mode once bus idle is detected. In case of ABTD = 0 the CAN module transmits the ABT message without any delay between two consecutive frames. This results in a delayed SLEEP acceptance as the CAN module first transmits all ABT messages before the condition bus-idle is detected.
If ABTD is selected other value than 0 the SLEEP request will be accepted as soon as the CAN module detects the first bus-idle between two consecutive frames. In addition when entering the SLEEP mode of the CAN module the ABT transmission is automatically suspended by clearing ABTTRG to 0. After releasing the SLEEP mode of the CAN module it is possible to resume the ABT transmission by setting again ABTTRG to 1.

14.7.4 Transmission request abort process

(1) Transmission abort request in normal operation mode

Based on the needs in an application it may be necessary to abort a transmit request. The user can try to clear the TRQ bit in the MCTRLm register to abort a transmit request. The TRQ bit will be cleared immediately. If the abort was successful, i.e. the message is not transmitted, can be checked with the TSTAT bit in the CnCTRL register. This bit indicates the transmit activity of the AFCAN module. Refer to the flowchart below for the proper transmit abort request by application software.

Figure 14-28: Transmission Request Abort Process



Remark: To request a transmission abort for a particular buffer, the user shall not clear the RDY flag (0) but clear the TRQ flag of the same buffer (0).

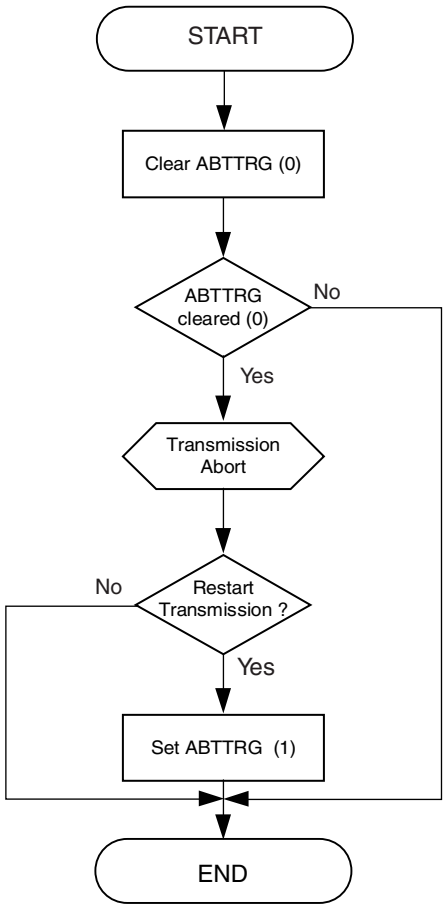
(2) Transmission abort request in normal operation mode with automatic block transmission

It may be 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 case of an erroneous transmission the internal ABT pointer is depending of 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 the ABTTRG (0) the internal ABT pointer points to the next ABT element.
If the TRQ bit remains its value at the clear of the ABTTRG (0) the internal ABT pointer points to the last transmitted ABT element.
In case of a restart of the ABT mode (ABTTRG is set (1) again) the next message to be transmitted can be determined from the following table.

TRQ	Abort after successful transmission	Abort after erroneous transmission
Untouched (1)	Next mailbox in the ABT area **Note**	Same mailbox in the ABT area **Note**
Cleared (0)	Next mailbox in the ABT area **Note**	Next mailbox in the ABT area

Note: Only if not the last message buffer (buffer 7) in the ABT area is reached or if all following buffers in the ABT area have their RDY bit cleared (0).

Figure 14-29: Transmission Request Abort Process with ABT mode



14.7.5 Transmission of remote frames

Remote frames are only sent from message buffers defined as transmit message buffer (MT[2:0]=00H). To distinguish a transmission request for a data frame and for a remote frame the RTR bit in MCONFm register has to be programmed accordingly. RTR bit set (1) determines a remote frame transmission request.
All methods described for data frames apply also for the transmission of a remote frame.

14.8 Operational Modes of RXONLY_CH

14.8.1 Receive–only mode

In Receive-only mode of the RXONLY-CH CAN module every data and every remote frame, which is received from the CAN bus in the RXONLY-CH CAN module without an error (i.e. valid reception), is stored in the upper 16 message buffers without any acceptance filtering.

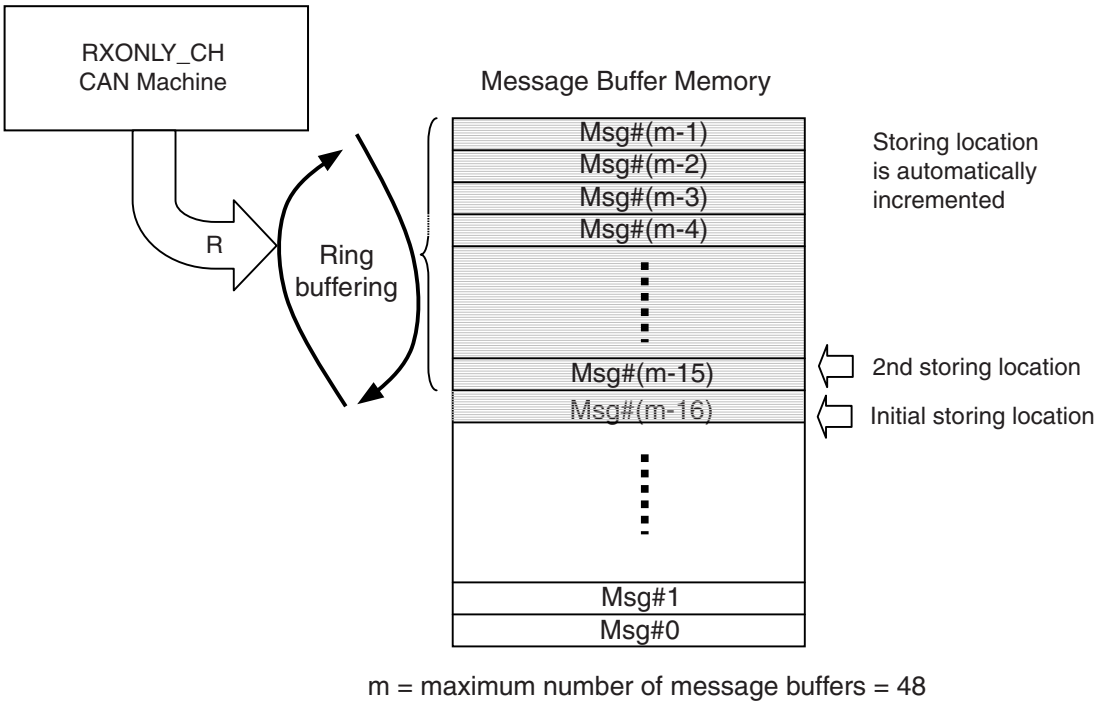
Storing received messages from RXONLY-CH requires the following conditions:

- The message buffer has to be assigned to the CAN I/F channel, which received the data frame (MA[2:0] bit-string in MCONF register has to hold the values 01H).
- The message buffer has to be marked ready for CAN protocol processing (RDY bit set (1) in MCTRL register)
- Additional condition necessary for Mirror Mode operation:
  The upper 16 message buffer need to be configured as transmit message buffer (MT[2:0] bit string in MCONF has to hold the value 0x00).

Caution: In difference to a regular AFCAN channel, any message is accepted by the buffer although it is configured as a transmit message buffer that normally would not except to receive data frames.
All transmit request bits of the upper 16 message buffer have to be cleared (TRQ bit reset (0) in MCTRL register) especially before Mirror Mode shall be invoked. This is necessary because the setting of TRQ for the upper 16 message buffer is handled by the mirror mode machine.

The upper 16 message buffer behave as ring buffer as shown in the figure below.

Figure 14-30: Reception Process of RXONLY



The storing pointer is initialised to (m-16) at the time when switching the assignment of upper 16 message buffer.

The reception process of the RXONLY_CH differs from the reception process of the regular AFCAN macro.

- There is no acceptance filtering; any data or remote frame will be received.
- No specific setting is required to receive standard and extended format frames in the upper 16 buffers during RECONLY and Mirror Mode. IDE bit will be written by the RX-store machine with respect to the received frame type.
- For received remote frames the RTR bit will be set by the RX-Store machine.

The table below compares the differences for each data type of the buffer.

Table 14-7: Differences in reception process between regular AFCAN and RXONLY_CH

Register	Bit String	DIAG /regular AFCAN	RXONLY_CH
MDATA0 –MDATA7	-	Written by RX-Store machine for remote frames value is kept	Written by RX-Store machine
MDLC	-	Written by RX-Store machine for remote frames value is kept	Written by RX-Store machine
MCONF	OWS, MT2-MT0, MA2-MA0	Set up by CPU during configuration	Set up by CPU during configuration
	RTR	Set up by CPU when preparing TX message	Written by RX-Store machine according received frame type
MIDH/MIDL	ID0-ID28	Written by RX-Store machine according rules of acceptance filtering	Written by RX-Store machine as received in message
	IDE	Set up by CPU during configuration	Set up at reception by RX-Store machine
MCTRL	MOW IE	Set up by CPU during configuration	Set up by CPU during configuration
	DN	Set by RX-Store machine and cleared by CPU	Set by RX-Store machine and cleared by CPU in RECONLY mode. Autonomously operated by RXONLY_CH in Mirror mode
	TRQ	Set by CPU (in ABT mode TRQ is set by ABT-engine autonomously)	Set by RXONLY_CH when operated in Mirror Mode
	RDY	Set/cleared by CPU	Cleared by CPU before RXONLY_CH is put to RECONLY or Mirror mode. Then set again by CPU in order to enable storing messages.

The transition into RECONLY mode is described in 14.4.6 ”CAN bit time programming” on page 654.

(1) Processing received frames in RECONLY mode of RXONLY_CH

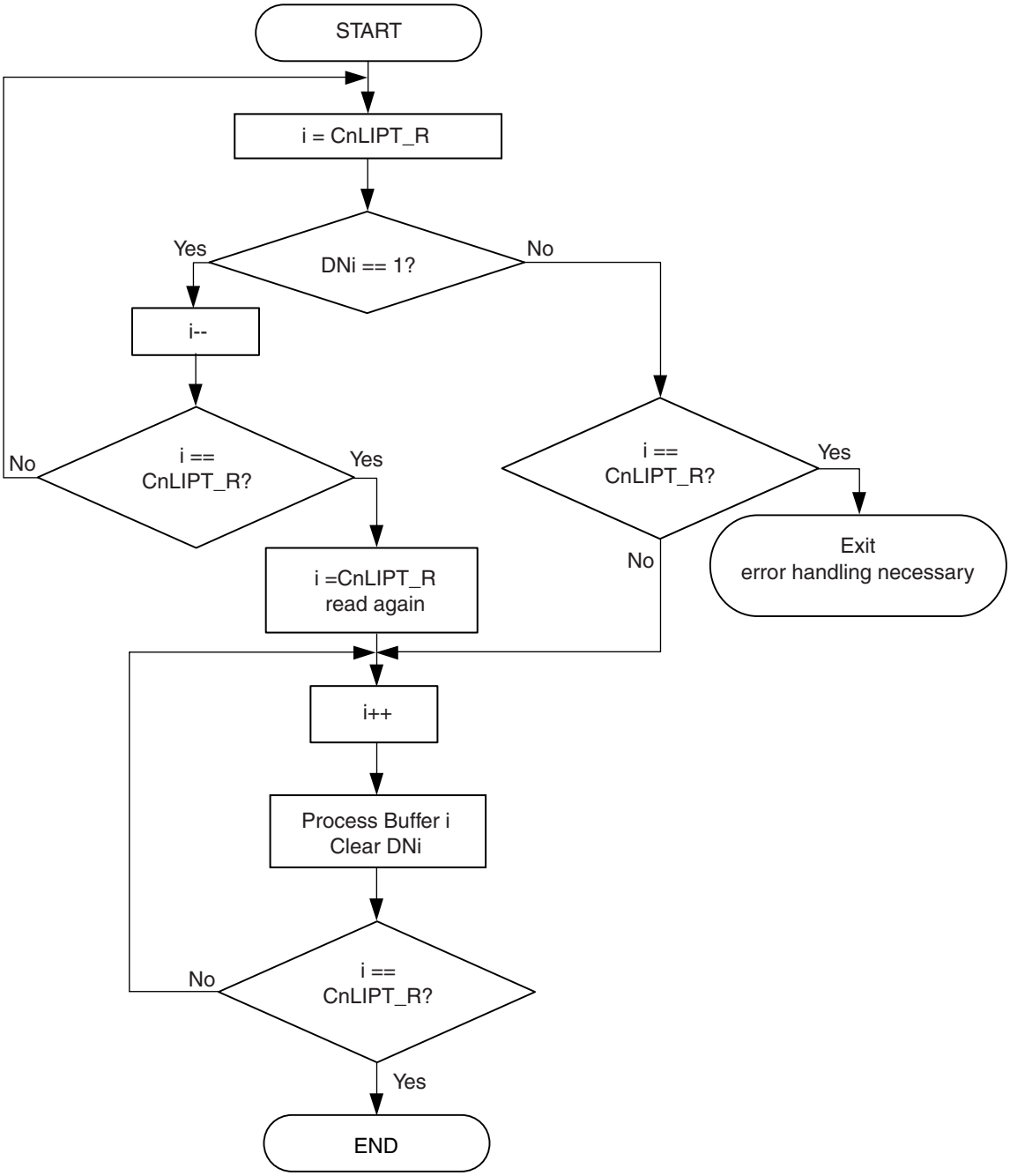
When the RXONLY_CH is operated in RECONLY mode, all valid messages are stored in the upper 16 message buffer. At initial start of that operating mode (i.e. when leaving INIT), the first message is stored in buffer 32. When resuming RECONLY mode leaving SLEEP mode, the storage will resume from the last position (message buffer) reached previously. If the user has set the option to get an interrupt, a receive interrupt is signalled to the host CPU.
The CPU can identify the most recently stored message with the help of the CnLIPT_R register. In case the CPU chooses to randomly poll the upper 16 message buffer for new messages and more than 1 message has its DN flag set, the sequence of their reception can be retrieved by a special algorithm.

In that case the CPU reads the CnLIPT_R register and decrements the message buffer number by one. Then CPU checks if the DN flag of that buffer is set. If not set, the buffer with the oldest message was one position ahead (in reverse direction of the CPU search).
If the DN flag is set the CPU repeats decrementing the buffer number with wrap around from buffer number 32 to 47 until it encounters a DN flag that is not set or until it reaches the buffer number that equals CnLIPT_R +1. In that case the CPU reads CnLIPT_R again in order to find out if newly received messages have been stored while the CPU was searching for the oldest message.
Then, this secondly read value incremented by one points to the oldest message.

The CPU needs to clear the DN flag after processing the received messages in order to avoid an overflow situation, which generates an error interrupt. In case this interrupt occurs, the host CPU can directly start processing messages at buffer CnLIPT_R+1, because the RXONLY_CH does not overwrite any buffer with DN=1. Once the CPU has cleared the DN flag of buffer CnLIPT_R+1, the RXONLY_CH will resume storing received messages. Clearing DN flags at other locations beforehand, will not let the reception process resume because the RXONLY_CH always stores messages in ascending order starting at CnLIPT_R+1.

The flow chart below illustrates the necessary algorithms for the software.

Figure 14-31: Receive Operation for RXONLY_CH in RECONLY Mode



The exit that requires error handling is normally not encountered. In that case the application would process a receive interrupt but did not find any newly received message. Typically, the application has failed to clear the interrupt pending bit in a previous interrupt routine.

14.8.2 Mirror mode

The DIAG macro features a message mirroring function. It automatically copies messages from the RXONLY-CH to the bus served by the DIAG-CH. When this function, the mirror mode, is enabled, any valid data frame and any valid remote frame received by the RXONLY-CH will be stored in the upper 16 message buffers in the same manner as in RECONLY mode. From that location these messages are automatically transmitted by the DIAG-CH in FIFO manner. No acceptance filtering is applied to the frame receptions.
The message mirroring is activated when the DIAG macro is configured as the follows:

- The OPMODE of RXONLY-CH CAN machine is configured as “Mirror mode”.
- The OPMODE of DIAG-CH CAN machine is configured as either “Normal operation mode” or “Normal operation mode with ABT”
- Neither of the CAN machines are in power save mode.

Other settings for the OPMODE for the DIAG_CH I.e., SSHT are not forbidden, but they my lead to unexpected behaviour i.e. in case of transient bus errors on the diagnosis bus the mirrored message will not be repeated and thus be not visible although it was successfully communicated on the bus that was monitored. The application is strongly recommended to use only the configuration as mentioned in the listing above.
The mirror mode engine sets one TRQ at a time sequentially in FIFO manner. So there is only one TRQ bit, which is set (1) in the upper 16 buffers when mirroring is enabled. The application must not set any TRQ bit in this upper 16 buffers when mirroring or RECONLY mode is enabled. Neither the application shall leave a TRQ bit set (1) when the mirror mode, mirror mode w/ TIF or RECONLY mode shall be entered next.
Even when the mirroring function is enabled, the TX-search of the DIAG_CH will be applied to the upper 16 message buffers although they are assigned to the RXONLY_CH. The search engine will conduct the TX-search on the lower buffers and the candidate from the upper 16 buffers. Thus, the mirrored messages can suffer a delay before they can be seen on the diagnosis CAN bus depending on the message priority loaded in the lower buffers.

(1) Operations of the Mirror Mode Engine

The following paragraphs describe the operation of the mirror mode engine (MME).
After RESET the LIPT_R points to the buffer number #32 where the first frame from the RXONLY-CH is stored. As soon as the DN flag is set (1), the MME starts to operate using current LIPT_R value. Note that LIPT_R value can be different as #32 when the user resumes Mirror mode from SLEEP mode of RXONYLY_CH.
If there is no other copy process from a previously handled message from the upper 16 buffers pending, the TRQ flag of the buffer with DN = 1 is set (1). The MME simultaneously memorises that a mirror mode operation is in progress (internal MMP flag is set) in order to queue additional requests (newly received messages from RXONLY_CH) for later execution. Up to 16 requests are stored by the MME. The MMP flag is cleared when all pending requests have been dealt with.

When the transmission by the DIAG-CH is finished, the MME clears the respective DN flag and it clears the current request for the MME (i.e. decrement the up down counter). The buffer number can be obtained from THL of DIAG_CH.
If no further requests are pending (MMP = 0) the MME turns idle until the RXONLY_CH receives a new message. If there are other requests pending, the MME walks to the next buffer (in RX-Store direction of RXONLY_CH), writes MMP = 1, and sets the TRQ for that message buffer.
The ‘transmission complete’ state can be easily detected by the TX-pending signal for the TX-complete interrupt. Thus, in Mirror mode the interrupt enable bit has to be set for all upper 16 message buffers. This configuration will also generate RX-interrupts from RXONLYX_CH. However the application can choose to suppress any interrupts generated by new receptions on the RXONLY_CH by masking the respective interrupt vector individually in the system interrupt controller or in the CxIE register of the AFCAN. The TX-complete interrupt can not be masked completely because the same interrupt vector is also used for the transmissions for the lower buffer numbers (#0 - #31). However the interrupt can be masked partially by the host CPU. For the trans-

missions from the upper message buffers the CPU can choose to mask the interrupt generation separately. This partial mask is only effective for transmit interrupt of the DIAG_CH in Mirror mode, i.e. when the upper 16 message buffer are assigned to the RXONLY_CH. Thus the CPU can configure both interrupts (RX for REXONLY_CH and sub-mask (#32 - #47) for TX-complete on DIAG_CH) such that it is able to follow up the Mirror mode operations (intrusive operation) or not (i.e. non-intrusive operation of Mirror mode). Any TX-complete signal generated by the upper 16 buffers, activates the MME. Then the MME clears the TX-interrupt pending signal before it clears the DN flag, and it decrements the MMP-counter as described above.
Note that the MMP flag is cleared when Mirror Mode is cancelled by a transition to INIT mode of RXONLY_CH. Any changes of the operational or power save modes of the DIAG_CH and switching the RXONLY_MODE to SLEEP and further to STOP mode have no influence on MMP flag. Thus the mirror mode can be resumed later on at the position where it suspended.
The following diagrams illustrates the phases of operation in Mirror mode.

Figure 14-32: Mirror Mode without Application Communication on DIAG_CH

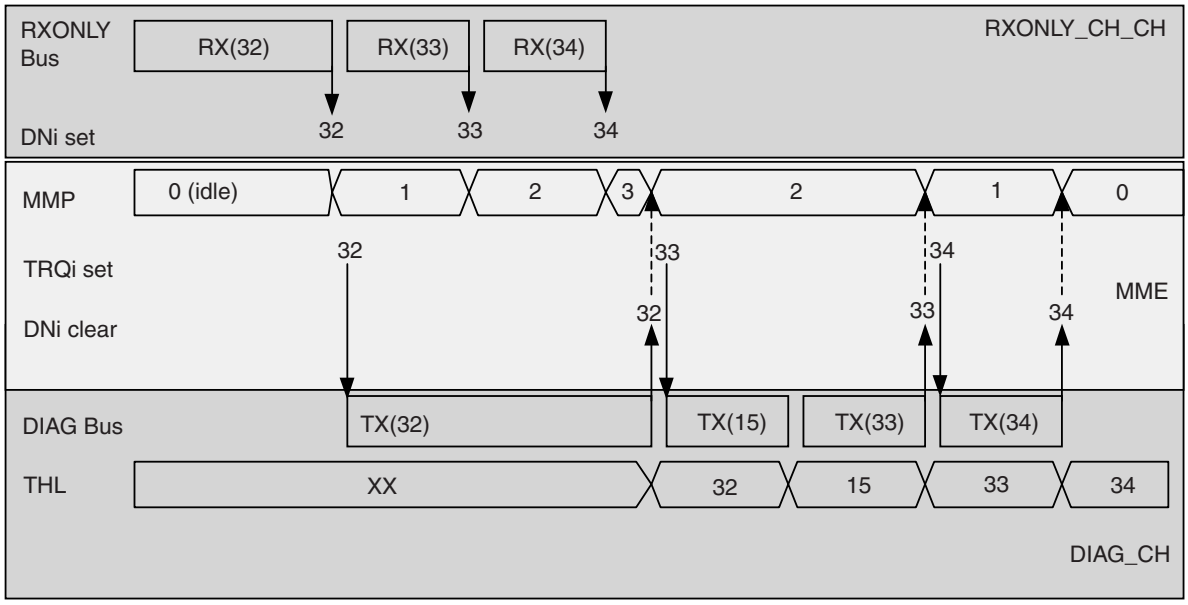
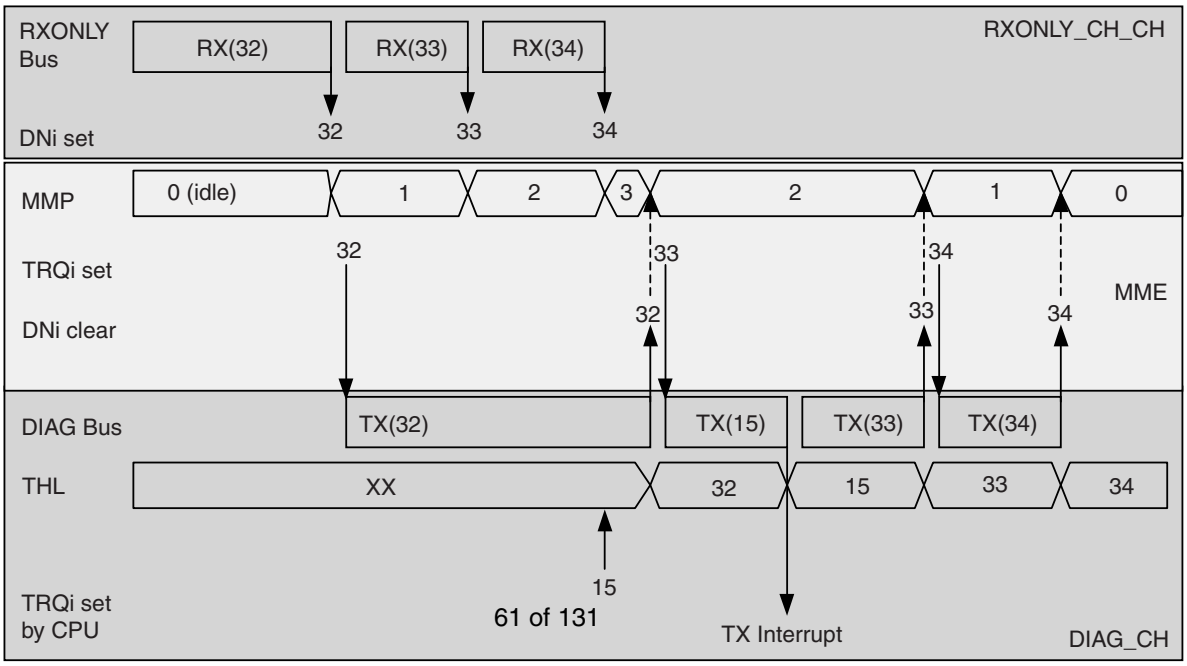


Figure 14-33: Mirror Mode with interleaved Application Communication on DIAG_CH



The RXONLY_CH can monitor up to eight other CAN channels. These sources can be selected via the BSEL[2:0] bits in the CnBSEL_R register during initialisation mode. Changing the selection during other operational modes is inhibited.

14.8.3 Mirror Mode with TIF

The mirror mode with TIF (Transfer Identifier filtering) applies acceptance filtering to the messages received on RXONLY-CH. All other operations are identical to Mirror Mode (OPMODE = 110b).
Up to 8 FullCAN identifiers can be specified in the transfer ID reference registers. Additionally one mask (Transfer ID Mask) register can be linked individually to each of the reference registers.
Any message matching one of the filter criteria will be stored in the upper 16 message buffer and thus participate in the mirror mode function. All messages not matching any filter criteria are not stored. This feature provides the application with a tool that minimizes the data throughput to a relevant subset of messages.

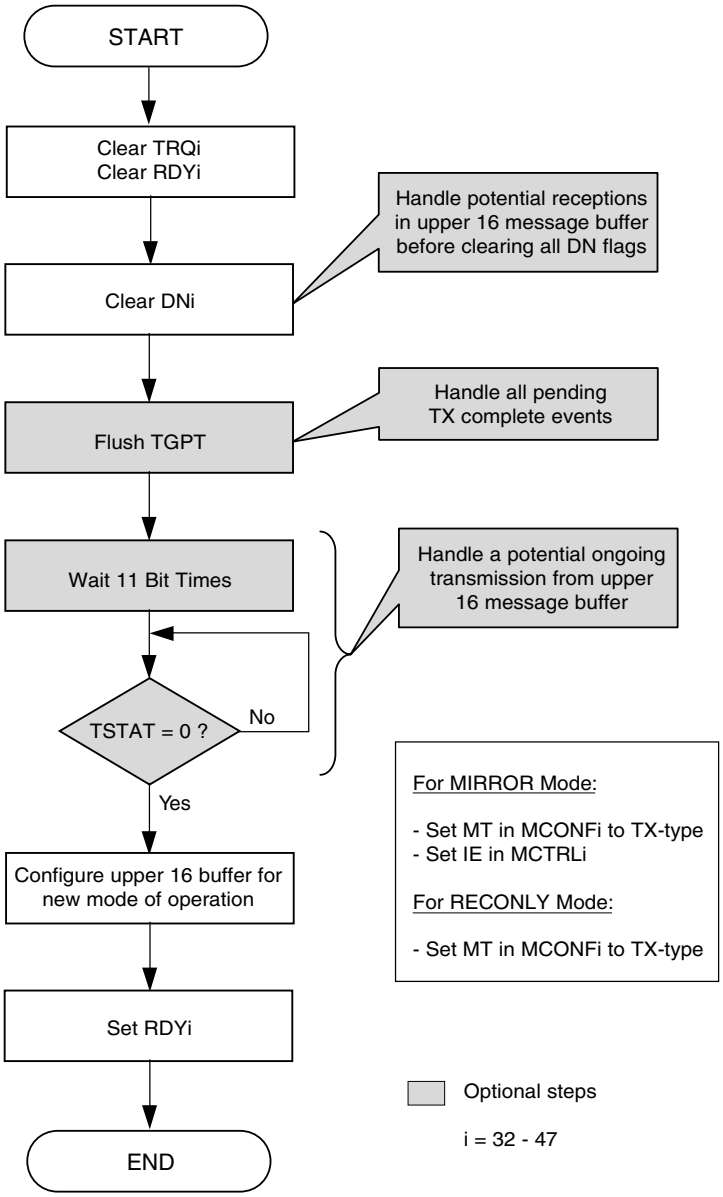
14.9 Transitions for Buffer Assignment

The DIAG macro provides for each channel several operational modes and several power save modes. When considering all possible states with respect to the independent settings for the DIAG_CH and the RXONLY_CH, a huge number of transitions are basically available. However, the majority of mode transitions are not possible because i.e. the direct transition from NORMAL to NORMAL with ABT is not allowed, as well as STOP mode can not be directly followed by NORMAL mode.

The Appendix B ”Transition Diagrams RXONLY Channel and DIAG Channel” on page 809 carries the complete list of all theoretically available transitions. The allowed transitions are highlighted.

This chapter describes the operations the host CPU has to perform in order to correctly switch between power save and operational modes of any of the two CAN channels of the DIAG macro. The flow chart below sketches the algorithm the host CPU needs to issue any time the buffer assignment changes.

Figure 14-34: Changing the Assignment of the upper 16 buffer



Whenever the assignment of the upper 16 buffers shall be changed, the host CPU needs to clear their RDY, TRQ, and DN bits at first. Before the DN flags are cleared, the application can optionally serve newly received messages.
In a second step the TGPT needs to be flushed, which forces the host CPU to handle all pending TX-complete events. If the application does not care on these events, this step can be skipped.

The third step is again optional and only applicable if the application used at least one of the upper 16 buffers for transmitting application messages. If the application likes to follow the sequence of these messages, it needs to wait 11 bit times and then check for bus idle. After this wait-algorithm any ongoing TX-messages from the DIAG_CH is completed and the transition flow can step ahead.

Then, the new buffer configuration is set up. When switching to RXONLY_CH assignment, all upper 16 buffers need to be configured as TX-type. If additionally Mirror mode shall be run, the IE bit of these buffers need to be set.
When switching the assignment to the DIAG_CH the application can setup its individual configuration.

In the second but last step, the new operational mode and the new power save mode for any of the channels can be invoked.

Finally, the RDY bits of all upper 16 message buffers need to be set again when the assignment is switched to the RXONLY_CH. In opposite direction (DIAG_CH) the RDY bits can be configured according the specific needs of the application.

14.10 Register Description

14.10.1 Registers of global macro control

Table 14-8: Global Macro Register (CPU read access) (1/2)

Register Name	Address-Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CGMCTRL	00H	0	0	0	0	0	0	EFSD	GOM
	01H	MBON	0	0	0	0	0	0	0
CGMCS	02H	0	0	0	0	CCP3	CCP2	CCP1	CCP0
	03H	access prohibited							
CGMCONF	04H	GCONF 7	GCONF 6	GCONF 5	GCONF 4	GCONF 3	GCONF 2	GCONF 1	GCONF 0
	05H	GCONF 15	GCONF 14	GCONF 13	GCONF 12	GCONF 11	GCONF 10	GCONF 9	GCONF 8
CGMABT	06H	0	0	0	0	0	0	ABTCLR	ABTTRG
	07H	0	0	0	0	0	0	0	0
CGMABTD	08H	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0
	09H - 3FH	access prohibited							
CnTIDR0L	0CH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	0DH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR0H	0EH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	0FH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR1L	10H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	11H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR1H	12H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	13H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR2L	14H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	15H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR2H	16H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	17H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR3L	18H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	19H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR3H	1AH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	1BH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR4L	1CH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	1DH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR4H	1EH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	1FH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR5L	20H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	21H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR5H	22H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	23H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24

Table 14-8: Global Macro Register (CPU read access) (2/2)

Register	Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CnTIDR6L	24H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	25H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR6H	26H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	27H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR7L	28H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	29H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR7H	2AH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	2BH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDML	2CH	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	2DH	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnTIDMH	2EH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID01 6
	2FH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	30H - 3FH	access prohibited							

Table 14-9: Global Macro Register (CPU write access) (1/2)

Register Name	Addr.-Off-set	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CGMCTRL	00H	0	0	0	0	0	0	0	clear GOM
	01H	0	0	0	0	0	0	set EFSD	set GOM
CGMCS	02H	0	0	0	0	CCP3	CCP2	CCP1	CCP0
	03H	access prohibited							
	04H	access prohibited							
	05H	access prohibited							
CGMABT	06H	0	0	0	0	0	0	0	clear ABTTRG
	07H	0	0	0	0	0	0	set ABTCLR	set ABTTRG
CGMABTD	08H	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0
	09H – 0BH	access prohibited							
CnTIDR0L	0CH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	0DH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR0H	0EH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	0FH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR1L	10H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	11H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR1H	12H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	13H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR2L	14H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	15H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR2H	16H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	17H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR3L	18H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	19H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR3H	1AH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	1BH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR4L	1CH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	1DH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR4H	1EH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	1FH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR5L	20H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	21H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR5H	22H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	23H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR6L	24H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	25H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8

Table 14-9: Global Macro Register (CPU write access) (2/2)

Register Name	Addr.-Off-set	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CnTIDR6H	26H	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	27H	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDR7L	28H	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	29H	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
CnTIDR7H	2AH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	2BH	IDE	IME	0	ID28	ID27	ID26	ID25	ID24
CnTIDML	2CH	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	2DH	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnTIDMH	2EH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	2FH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	30H - 3FH	access prohibited							

14.10.2 CAN Global Macro Control Register (CGMCTRL)

CGMCTRL (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	EFSD	GOM	00H	00H
Initial value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	MBON	0	0	0	0	0	0	0	01H	00H
Initial value	0	0	0	0	0	0	0	0		

CGMCTRL (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	0	clear GOM	00H	00H
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	set EFSD	set GOM	01H	00H

GOM	Global Macro Operation Mode bit
0	The Global Macro is switched off
1	The Global Macro is switched on

EFSD	Enable Forced Shut Down bit
0	Forced Shut Down is disabled. To switch off the entire macro by the GOM bit all CAN modules must be in the INIT mode.
1	Forced Shut Down is enabled. Switching off the entire macro by the GOM bit is possible regardless of the CAN modules' mode.

MBON	Message Buffer Access On
0	CPU write access and read access to the message buffers is impossible because all CAN IF channels are either in SLEEP and /or in STOP mode.
1	CPU write access and read access to the message buffers is possible.

Cautions: 1. User must not access MDATA0m, MDATA1m, MDATA2m, MDATA3m, MDATA4m, MDATA5m, MDATA6m, MDATA7m, MDLCm, MCONFm, MIDLm, MIDHm or MCTRLm registers while MBON bit is cleared (0).

2. When EFSD bit is set (1) the subsequent CPU access to the CAN module needs to clear the GOM bit (0). In case GOM bit is not cleared (0) in the subsequent access, the EFSD bit is cleared(0) automatically and the forced shut down is not executed.

14.10.3 CAN Global Macro Clock Selection Register (CGMCS)

CGMCS									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	CCP3	CCP2	CCP1	CCP0	002H	0FH
Initial Value	0	0	0	0	1	1	1	1		

CCP3	CCP2	CCP1	CCP1	CAN Macro Clock Prescaler for the macro Clock ϕ_{CANMOD}
0	0	0	0	$\phi_{CANMOD} = \phi_{CAN} / 1$
0	0	0	1	$\phi_{CANMOD} = \phi_{CAN} / 2$
0	0	1	0	$\phi_{CANMOD} = \phi_{CAN} / 3$
0	0	1	1	$\phi_{CANMOD} = \phi_{CAN} / 4$
0	1	0	0	$\phi_{CANMOD} = \phi_{CAN} / 5$
0	1	0	1	$\phi_{CANMOD} = \phi_{CAN} / 6$
0	1	1	0	$\phi_{CANMOD} = \phi_{CAN} / 7$
0	1	1	1	$\phi_{CANMOD} = \phi_{CAN} / 8$
1	0	0	0	$\phi_{CANMOD} = \phi_{CAN} / 9$
1	0	0	1	$\phi_{CANMOD} = \phi_{CAN} / 10$
1	0	1	0	$\phi_{CANMOD} = \phi_{CAN} / 11$
1	0	1	1	$\phi_{CANMOD} = \phi_{CAN} / 12$
1	1	0	0	$\phi_{CANMOD} = \phi_{CAN} / 13$
1	1	0	1	$\phi_{CANMOD} = \phi_{CAN} / 14$
1	1	1	0	$\phi_{CANMOD} = \phi_{CAN} / 15$
1	1	1	1	$\phi_{CANMOD} = \phi_{CAN} / 16$

14.10.4 CAN Global Macro Automatic Block Transmission Register (CGMABT)

CGMABT (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	ABTCLR	ABTTRG	006H	00H
Initial Value	0	0	0	0	0	0	0	0		

	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	007H	00H
Initial Value	0	0	0	0	0	0	0	0		

CGMABT (write)	7	6	5	4	3	2	1	0	Address Offset
	0	0	0	0	0	0	0	clear ABTTRG	006H

15	14	13	12	11	10	9	8	
0	0	0	0	0	0	set ABTCLR	set ABTTRG	007H

Remark: Before switching from “Normal Operating Mode with Automatic Block Transmission” to the INIT mode the user must clear the bits in the CGMABT register to their initial values.

ABTTRG	Automatic Block Transmission Trigger bit
0	Transmission from the ABT group message buffers not started
1	Transmission from the ABT group message buffers started

Remark: User must not set ABTTRG bit during INIT mode.
Otherwise when entering the “Normal Operating Mode with Automatic Block Transmission” after the INIT mode the correct operation can not be guaranteed.

ABTCLR	Automatic Block Transmission Clear bit
0	The ABT-engine is idle or under operation
1	Clear request to the ABT-engine. Upon re-start of the ABT-engine by setting ABTTRG (1), the ABT-engine starts transmission from the first ABT message buffer element (i.e. message buffer 0).

Cautions: **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 the user sets the ABTCLR bit (1) and that clear request has been accepted

14.10.5 CAN Global Configuration Register (CnGMCONF)

The CnGMCONF register provides the configuration information of the CAN module.

Figure 14-35: CAN Global Configuration Register (CnGMCONF) Format

Symbol	15	14	13	12	11	10	9	8	Address	Default value
CnGMCONF (Read only)	Note							GCONF8	05H	0x01
	7	6	5	4	3	2	1	0		
	Note		GCONF5	GCONF4	GCONF3	GCONF2	GCONF1	GCONF0	0x04	0x19

Note: Undefined (reserved for future use)

GCONF2	GCONF1	GCONF0	Number of CAN interface channels[Note]
0	0	0	reserved
0	0	1	1 CAN interface channel
0	1	0	reserved
0	1	1	reserved
1	0	0	reserved
1	0	1	reserved
1	1	0	reserved
1	1	1	reserved

Note: This is not the number of the channels of the device.

GCONF5	GCONF4	GCONF3	Number of message buffers
0	0	0	Reserved
0	0	1	16 message buffers
0	1	0	32 message buffers
0	1	1	48 message buffers
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

GCONF8	Mirror Function for DIAG macro
0	no Mirror function is implemented
1	Mirror function is implemented

14.10.6 CAN Global Macro Automatic Block Transmission Delay Register (CGMABTD)

CGMABTD									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0	008H	00H
Initial Value	0	0	0	0	0	0	0	0		

ABTD3	ABTD2	ABTD1	ABTD0	Delay between two consecutive ABT data frames. Unit is 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
1	0	0	1	reserved
1	0	1	0	reserved
1	0	1	1	reserved
1	1	0	0	reserved
1	1	0	1	reserved
1	1	1	0	reserved
1	1	1	1	reserved

Caution: The user must not change the content of the CGMABTD register while the ABTTRG bit is set (1).

14.10.7 CAN Transfer ID Reference Registers (CnTIDRmL, CnTIDRmH; m = 0 - 7)

The CnTIDRmL and CnTIDRmH registers are used to set the filter criteria for Mirror mode w/ TIF of RXONLY-CH. These registers become only effective for Mirror mode w/ TIF. The application needs to initialise all these registers before entering Mirror mode w/TIF.

CnTIDR0L									Address Offset	Initial Value
(read/write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	0CH	undefined
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	0DH	undefined
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR0H										
(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	0EH	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	0FH	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR1L									Address Offset	Initial Value
(read/write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	10H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	11H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR1H										
(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	12H	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	Undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	13H	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	Undef.	undef.		

CnTIDR2L

(read/write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	14H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	15H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR2H

(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	16H	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	17H	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR3L

(read/write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	18H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	19H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR3H

(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	1AH	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	1BH	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR4L

(read/write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	1CH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	1DH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR4H

(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	1EH	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	1FH	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR5L

(read/write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	20H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	21H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR5H

(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	22H	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	23H	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR6L									Address Offset	Initial Value
(read/write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	24H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	25H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR6H										
(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	26H	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	27H	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

CnTIDR7L									Address Offset	Initial Value
(read/write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	28H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	29H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDR7H										
(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	2AH	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	IME	0	ID28	ID27	ID26	ID25	ID24	2BH	xx0xxxxxB
Initial Value	undef.	undef.	0	undef.	undef.	undef.	undef.	undef.		

ID28 to ID0	Message ID Reference for Transfer Message
ID28 to ID18	range for the 11-bit Standard identifier values
ID28 to ID0	range for the 29-bit Extended identifier values

IDE	Identifier Extension Bit for Transfer Message
0	11-bit Standard identifier[Note]
1	29-bit Extended identifier.

Note: The bits ID17 to ID0 are not used and may contain undefined values.

IME	Mask Enable bit for Transfer Message
0	CnTIDRL/H register without a link to the CnTIDML/H
1	CnTIDRL/H register with a link to the CnTIDML/H

Caution: Be sure to write "0" to bit 13 of the CnTIDRHm registers.
When CPU attempts to write CnTIDRLm and CnTIDRHm in a mode other than INIT, it is simply ignored.

14.11 CAN Transfer ID Mask Registers (CnTIDML, CnTIDMH)

The CnTIDML and CnTIDMH registers are used to extend the number of receivable messages into the upper 16 message buffers by masking part of the ID of a message and invalidating the ID of the masked part.

CnTIDML (read / write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	2CH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	2DH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTIDMH (read / write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	2EH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	2FH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CMID28 to CMID0	Mask Identifier Pattern for Transfer message
0	Corresponding identifier bit in the received message frame and in the message buffer must match
1	Corresponding identifier bit in the received message frame and in the message do not care

Remark: When CPU attempts to write CnTIDML and CnTIDMH in a mode other than INIT, it is simply ignored.

14.11.1 Registers of the CAN Module

Table 14-10: CAN Module Register (CPU read access) (1/2)

Register Name	Addr.-Offset	Bit 7/ 15	Bit 6/ 14	Bit 5/ 13	Bit 4/ 12	Bit 3/ 11	Bit 2/ 10	Bit 1/ 9	Bit 0/ 8
CnMASK1L	00H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	01H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK1H	02H	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	03H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK2L	04H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	05H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK2H	06H	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	07H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK3L	08H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	09H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK3H	0AH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	0BH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK4L	0CH	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	0DH	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK4H	0EH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	0FH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnCTRL	10H	CCERC	AL	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0
	11H	0	0	0	0	0	0	RSTAT	TSTAT
CnLEC	12H	0	0	0	0	0	LEC2	LEC1	LEC0
CnINFO	13H	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0
CnERC	14H	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
	15H	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0
CnIE	16H	CIE7	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0
	17H	0	0	0	0	0	0	0	0
CnINTS	18H	CINTS7	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0
	19H	0	0	0	0	0	0	0	0
CnBRP	1AH	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0
	1BH	access prohibited							
CnBTR	1CH	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10
	1DH	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
CnLIPT	1EH	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0
	1FH	access prohibited							
CnRGPT	20H	0	0	0	0	0	0	RHPM	ROVF
	21H	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0
CnLOPT	22H	LOPT7	LOPT6	LOPT5	LOPT4	LOPT3	LOPT2	LOPT1	LOPT0
	23H	access prohibited							
CnTGPT	24H	0	0	0	0	0	0	THPM	TOVF
	25H	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0

Table 14-10: CAN Module Register (CPU read access) (2/2)

Register Name	Addr.-Offset	Bit 7/ 15	Bit 6/ 14	Bit 5/ 13	Bit 4/ 12	Bit 3/ 11	Bit 2/ 10	Bit 1/ 9	Bit 0/ 8
CnTS[Note]	26H	0	0	0	0	0	TSLOCK	TSSEL	TSEN
	27H	0	0	0	0	0	0	0	0
CnCTRL_R	28H	CCERC	0	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0
	29H	0	0	0	0	0	0	RSTAT	0
CnLEC_R	2AH	0	0	0	0	0	LEC2	LEC1	LEC0
	2BH	access prohibited							
CnERC_R	2CH	0	0	0	0	0	0	0	0
	2DH	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0
CnIE_R	2EH	0	CIE6	CIE5	0	CIE3	CIE2	CIE1	0
	2FH	0	0	0	0	0	0	0	0
CnINTS_R	30H	0	CINTS6	CINTS5	0	CINTS3	CINTS2	CINTS1	0
	31H	0	0	0	0	0	0	0	0
CnBRP_R	32H	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0
	33H	access prohibited							
CnBTR_R	34H	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10
	35H	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
CnLIPT_R	36H	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0
	37H	access prohibited							
CnTS_R [Note]	38H	0	0	0	0	0	TSLOCK	TSSEL	TSEN
	39H	0	0	0	0	0	0	0	0
CnBSEL_R	3AH	0	0	0	0	0	BSEL2	BSEL1	BSEL0
	3BH to 3FH	access prohibited							

Note: The displayed CnTS and CnTS_R registers provide only the necessary bits for the Basic Time Stamp function.
Modifications will have to be done for the Advanced Time Stamp function.

Table 14-11: CAN Module Register (CPU write access) (1/2)

Register Name	Addr.-Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CnMASK1L	00H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	01H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK1H	02H	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	03H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK2L	04H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	05H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK2H	06H	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	07H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK3L	08H	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	09H	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK3L	0AH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	0BH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnMASK4L	0CH	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	0DH	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
CnMASK4H	0EH	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID016
	0FH	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
CnCTRL	10H	0	clear AL	clear VALID	clear PSMODE1	clear PSMODE0	clear OPMODE2	clear OPMODE1	clear OPMODE0
	11H	Set CCERC	set AL	set VALID	set PSMODE1	set PSMODE0	set OPMODE2	set OPMODE1	set OPMODE0
CnLEC	12H	0	0	0	0	0	LEC2	LEC1	LEC0
CnINFO	13H	0	0	0	0	0	0	0	0
CnERC	14H	0	0	0	0	0	0	0	0
	15H	0	0	0	0	0	0	0	0
CnIE	16H	clear CIE7	0	clear CIE5	clear CIE4	clear CIE3	clear CIE2	clear CIE1	clear CIE0
	17H	set CIE7	0	set CIE5	set CIE4	set CIE3	set CIE2	set CIE1	set CIE0
CnINTS	18H	clear CINTS7	0	clear CINTS5	clear CINTS4	clear CINTS3	clear CINTS2	clear CINTS1	clear CINTS0
	19H	0	0	0	0	0	0	0	0
CnBRP	1AH	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0
	1BH	access prohibited							
CnBTR	1CH	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10
	1DH	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
CnLIPT	1EH	0	0	0	0	0	0	0	0
	1FH	access prohibited							
CnRGPT	20H	0	0	0	0	0	0	0	clear ROVF
	21H	0	0	0	0	0	0	0	0
CnLOPT	22H	0	0	0	0	0	0	0	0
	23H	access prohibited							

Table 14-11: CAN Module Register (CPU write access) (2/2)

Register Name	Addr.-Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
CnTGPT	24H	0	0	0	0	0	0	0	clear TOVF
	25H	0	0	0	0	0	0	0	0
CnTS[Note]	26H	0	0	0	0	0	clear TSLOCK	clear TSSEL	clear TSEN
	27H	0	0	0	0	0	set TSLOCK	set TSSEL	set TSEN
CnCTRL_R	28H	0	0	clear VALID	clear PSMODE1	clear PSMODE0	clear OPMODE2	clear OPMODE1	clear OPMODE0
	29H	Set CCERC	0	0	set PSMODE1	set PSMODE0	set OPMODE2	set OPMODE1	set OPMODE0
CnLEC_R	2AH	0	0	0	0	0	LEC2	LEC1	LEC0
	2BH	access prohibited							
CnERC_R	2CH	0	0	0	0	0	0	0	0
	2DH	0	0	0	0	0	0	0	0
CnIE_R	2EH	0	clear CIE6	clear CIE5	0	clear CIE3	clear CIE2	clear CIE1	0
	2FH	0	set CIE6	set CIE5	0	set CIE3	set CIE2	set CIE1	0
CnINTS_R	30H	0	clear CINTS6	clear CINTS5		clear CINTS3	clear CINTS2	clear CINTS1	0
	31H	0	0	0	0	0	0	0	0
CnBRP_R	32H	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0
	33H	access prohibited							
CnBTR_R	34H	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10
	35H	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
CnLIPT_R	36H	0	0	0	0	0	0	0	0
	37H	access prohibited							
CnTS_R Note	38H						clear TSLOCK	clear TSSEL	clear TSEN
	39H						set TSLOCK	set TSSEL	set TSEN
CnBSEL_R	3AH	0	0	0	0	0	BSEL2	BSEL1	BSEL0
	3BH to 3FH	access prohibited							

Note: The displayed CnTS and CnRS-R registers provide only the necessary bits for the Basic Time Stamp function.
Modifications will have to be done for the Advanced Time Stamp function.

14.11.2 DIAG_CH CAN Module Mask 1 Registers (CnMASK1L, CnMASK1H)

CnMASK1L									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	00H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	01H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnMASK1H										
(read / write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	02H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	03H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.3 DIAG_CH CAN Module Mask 2 Registers (CnMASK2L, CnMASK2H)

CnMASK2L									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	04H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	05H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnMASK2H										
(read / write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	06H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	07H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.4 DIAG_CH CAN Module Mask 3 Registers (CnMASK3L, CnMASK3H)

CnMASK3L									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	08H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	09H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnMASK3H										
(read / write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	0AH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	0BH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.5 DIAG_CH CAN Module Mask 4 Registers (CnMASK4L, CnMASK4H)

CnMASK4L									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0	0CH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	0DH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnMASK4H										
(read / write)	7	6	5	4	3	2	1	0		
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16	0EH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24	0FH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CMID28 to CMID0	Mask Identifier Pattern
0	Corresponding identifier bit in the received message frame and in the message buffer must match
1	Corresponding identifier bit in the received message frame and in the message do not care

14.11.6 DIAG_CH CAN Module Control Register (CnCTRL)

CnCTRL (read)

	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	CCERC	AL	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0	10H	00H
Initial Value	0	0	0	0	0	0	0	0		

	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	RSTAT	TSTAT	11H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnCTRL (write)

	7	6	5	4	3	2	1	0	Address Offset
	0	clear AL	clear VALID	clear PSMODE1	clear PSMODE0	clear OPMODE2	clear OPMODE1	clear OPMODE0	10H

	15	14	13	12	11	10	9	8	
	set CCERC	set AL	0	set PSMODE1	set PSMODE0	set OPMODE2	set OPMODE1	set OPMODE0	11H

OPMODE2	OPMODE1	OPMODE0	Operational Mode
0	0	0	No operational mode selected (DIAG-ch 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"
1	1	0	Prohibited by user / ignored by hardware
1	1	1	Prohibited by user / ignored by hardware

PSMODE1	PSMODE0	Power Save Mode
0	0	No power save mode selected (DIAG_CH CAN module is in INIT mode or in one of the operational modes)
0	1	SLEEP mode
1	0	Prohibited to use / ignored by hardware
1	1	STOP mode

VALID	Valid Reception bit
0	No valid message frame reception into the DIAG_CH CAN Protocol Transfer Layer since the VALID bit had been cleared (0) last time.
1	Valid message frame reception into the DIAG_CH CAN Protocol Transfer Layer since the VALID bit had been cleared (0) last time.

Remarks: **1.** User must not set the VALID bit (1) by using set VALID bit.

2. A valid reception does not require an acceptance of the message frame into a receive message buffer (data frame) or for a transmit message buffer (remote frame).

3. Before switching from INIT mode to any operational mode the user must clear the VALID bit (0).

4. A pending TRQ does not prevent the AFCAN to enter Sleep Mode if the respective message is not actually sent on the bus.

5. In case only two CAN nodes are connected to a CAN bus and one of the CAN nodes is in “Normal Operating mode” transmitting message frames while the other CAN node is in “Receive-only mode”, the VALID bit will be set (1) not before the transmitting node becomes error passive.

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 the arbitration was 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 the arbitration was lost.

Remarks: **1.** The AL bit is only applicable when the DIAG_CH CAN module operates in "Single-shot mode". In all other operational mode the AL bit needs to be cleared (0).

2. In case a DIAG_CH CAN module operates in the "Single-shot mode" and the AL bit is set (1), the interrupt CINTS4 is not generated upon ‘arbitration loss’.

CCERC	DIAG_CH CAN Clear Error Counter bit
0	While in INIT mode the DIAG_CH CAN module error counter CnERC and the DIAG_CH CAN module information register CnINFO will not be cleared.
1	While in INIT mode the DIAG_CH CAN module error counter CnERC and the DIAG_CH CAN module information register CnINFO will be cleared.

TSTAT	DIAG_CH CAN Transmission status
0	No transmission activity on the DIAG CAN bus
1	Transmission activity on the DIAG CAN bus

Remark: TSTAT is set under the following condition:
SOF of TX frame
First bit of an error flag during TX frame

TSTAT is cleared under the following condition:
In case of arbitration lost during TX frame
After detecting recessive bit at second bit of inter frame space
At transition to bus-off
Transition to INIT mode at 1st bit of 'interframe space'

RSTAT	DIAG_CH CAN Reception status
0	No receive activity on the DIAG CAN bus
1	Receive activity on the DIAG CAN bus

Remark: RSTAT is set under the following condition:
SOF of RX frame
Arbitration lost of an TX frame

RSTAT is cleared under the following condition:
After detecting recessive bit at second bit of inter frame space
Transition to INIT mode at 1st bit of 'interframe space'

14.11.7 DIAG_CH CAN Module Last Error Code Register (CnLEC)

CnLEC									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	0	LEC2	LEC1	LEC0	12H	00H
Initial Value	0	0	0	0	0	0	0	0		

Cautions: 1. The actual content of the CnLEC is not cleared by switching the DIAG_CH CAN module from an operational mode to the INIT mode.

2. When CPU attempts to write CnLEC with data other than 00h, it is simply ignored.

LEC2	LEC1	LEC0	Last Error Code of the 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 DIAG_CH CAN module tried to send a “recessive” ‘1’ bit as a part of a transmitted message (with the exception of the arbitration field), but the monitored DIAG_CH CAN bus value was “dominant” ‘0’}
1	0	1	Bit Error {the DIAG_CH CAN module tried to send a “dominant” ‘0’ bit as a part of a transmitted message or as an ACK bit, error or overload frame, but the monitored DIAG CAN bus value was “recessive” ‘1’}
1	1	0	CRC error
1	1	1	Unused

14.11.8 DIAG_CH CAN Module Information Register (CnINFO)

CnINFO									Address Offset	Initial Value
(read-only)	7	6	5	4	3	2	1	0		
	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0	13H	00H
Initial Value	0	0	0	0	0	0	0	0		

RECS1	RECS0	DIAG_CH Receive Error Counter Status
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)

TECS1	TECS0	DIAG_CH Transmit Error Counter Status
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 above warning level (≥ 128)

BOFF	Bus Off flag
0	DIAG_CH CAN is not bus-off (transmission error counter ≤ 255)
1	DIAG_CH CAN is bus-off (transmission error counter overflow)

14.11.9 DIAG_CH CAN Module Error Counter (CnERC)

CnERC (read-only)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	14H	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0	15H	00H
Initial Value	0	0	0	0	0	0	0	0		

14.11.10 DIAG_CH CAN Module Interrupt Enable Register (CnIE)

CnIE (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	CIE7	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0	16H	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	17H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnIE (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	clear CIE7	0	clear CIE5	clear CIE4	clear CIE3	clear CIE2	clear CIE1	clear CIE0	16H	
	15	14	13	12	11	10	9	8		
	set CIE7	0	set CIE5	set CIE4	set CIE3	set CIE2	set CIE1	set CIE0	17H	

CIE0 to CIE5, and CIE7	DIAG_CH CAN Interrupt Enable bits
0	Corresponding interrupt request signal INTxx is not generated when CINTSx is set by the interrupt source event.
1	Corresponding interrupt request signal INTxx is generated when CINTSx is set by the interrupt source event.

14.11.11 DIAG-CAN Module Interrupt Status Register (CnINTS)

CnINTS (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	CINTS7	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0	18H	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	19H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnINTS (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	clear CINTS7	0	clear CINTS5	clear CINTS4	clear CINTS3	clear CINTS2	clear CINTS1	clear CINTS0	18H	00H
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	19H	

CINTS0 to CINTS5, CINTS7	DIAG_CH CAN Interrupt Status bits
0	The related interrupt source event is not pending
1	The related interrupt source event is pending

Interrupt status bit	Related interrupt source event
CINTS0	DIAG_CH CAN module interrupt status bit for interrupt event 'Message frame successfully transmitted from message buffer m'
CINTS1	DIAG_CH CAN module interrupt status bit for interrupt event 'Valid message frame reception in message buffer m'
CINTS2	DIAG_CH CAN module error state interrupt status
CINTS3	DIAG_CH CAN module protocol error interrupt status
CINTS4	DIAG_CH CAN module arbitration loss interrupt status
CINTS5	DIAG_CH CAN module wake-up from SLEEP mode interrupt status bit **Note**
CINTS7	DIAG_CH CAN module completed transmission from upper 16 message buffer during Mirror mode on RXONLY_CH

Note: Only the wake-up from SLEEP mode by CAN bus activity generates the CINTS5 signal. The SLEEP mode release by CPU will not generate the CINTS5 signal.

14.11.12 DIAG_CH CAN Module Bit-Rate Prescaler Register (CnBRP)

CnBRP									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0	1AH	FFH
Initial Value	1	1	1	1	1	1	1	1		

TQPRS0 to TQPRS7	Time Quanta clock prescaler
0	$\phi_{TQ} = \phi_{CANMOD} / 1$
1	$\phi_{TQ} = \phi_{CANMOD} / 2$
n	$\phi_{TQ} = \phi_{CANMOD} / (n+1)$
...	...
255	$\phi_{TQ} = \phi_{CANMOD} / 256$

14.11.13 DIAG_CH CAN Bit Rate Register (CnBTR)

CnBTR									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10	1CH	0FH
Initial Value	0	0	0	0	1	1	1	1		
	15	14	13	12	11	10	9	8		
	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20	1DH	37H
Initial Value	0	0	1	1	0	1	1	1		

TSEG13	TSEG12	TSEG11	TSEG10	Length of Time Segment 1
0	0	0	0	prohibited
0	0	0	1	2 TQ
0	0	1	0	3 TQ
0	0	1	1	4 TQ
0	1	0	0	5 TQ
0	1	0	1	6 TQ
0	1	1	0	7 TQ
0	1	1	1	8 TQ
1	0	0	0	9 TQ
1	0	0	1	10 TQ
1	0	1	0	11TQ
1	0	1	1	12TQ
1	1	0	0	13TQ
1	1	0	1	14TQ
1	1	1	0	15TQ
1	1	1	1	16TQ

TSEG22	TSEG21	TSEG20	Length of Time Segment 2
0	0	0	1TQ
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

SJW1	SJW0	Length of Synchronisation Jump Width
0	0	1TQ
0	1	2 TQ
1	0	3 TQ
1	1	4 TQ

14.11.14 DIAG_CH CAN Module Last In-Pointer Register (CnLIPT)

CnLIPT									Address Offset	Initial Value
(read-only)	7	6	5	4	3	2	1	0		
	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0	1EH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

LIPT0 to LIPT7	Last In-Pointer of Receive History List for the DIAG_CH
0... message buffer $m\text{-}1_{max}$	Reading the CnLIPT register delivers the message buffer number in which the last data frame has been saved or the last remote frame has been stored into the message buffer area of DIAG-CH. The user can only evaluate CnLIPT when both of the following conditions are met: - the CAN channel is not in INIT mode - RHPM is cleared (0)

Remarks: 1. As long no data frame has been stored into a message buffer or no received remote frame has been assigned, an undefined value is read from CnLIPT register. The user must not read the CnLIPT register after switching the CAN module to one of the operational modes as long RHPM bit is set (1).

2. This register shows the reception history of the messages from DIAG-CH only.

14.11.15 DIAG_CH CAN Module Receive History List Get Pointer Register (CnRGPT)

CnRGPT (read)[Note 3]	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	RHPM	ROVF	20H	02H
Initial Value	0	0	0	0	0	0	1	0		
	15	14	13	12	11	10	9	8		
	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0	21H	undefined
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnRGPT (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	0	Clear ROVF	20H	
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	21H	

ROVF	Receive History List Overflow
0	Upon data frame storage or remote frame assignment, the corresponding message buffer number is logged into the DIAG-RHL.
1	At least 23 entries have been stored since the host processor has serviced the RHL last time (i.e. read CnRGPT). Thus the sequence of receptions can not be recovered completely now. The first 22 entries are sequentially stored while the last entry can have been overwritten by newly received messages multiple times because all buffer numbers are stored at position LIPT-1 when ROVF bit is set. [Note 1]

RHPM	Receive History List Pointers Match
0	There is at least one unread message in the DIAG-RHL
1	There is no unread message in the DIAG-RHL

RGPT0 to RGPT7	Receive History List Get Pointer
0... message buffer $m\text{-}1_{max.}$ [Note 2]	Reading the CnRGPT register delivers the message buffer number from which the user must read received data (receive message buffer) or to which a remote frame was received (transmit message buffer).

Notes: 1. The RHL will be updated, but the LIPT pointer will not be incremented. Always the position the LIPT pointer -1 is pointing to is overwritten.

2. The maximum number of message buffers ($m_{max.}$ = 48)

3. This register shows the reception history of the messages from DIAG-CH only.

14.11.16 DIAG_CH CAN Module Last Out-Pointer Register (CnLOPT)

CnLOPT									Address Offset	Initial Value
(read-only)	7	6	5	4	3	2	1	0		
	LOPT7	LOPT6	LOPT5	LOPT4	LOPT3	LOPT2	LOPT1	LOPT0	22H	undefined
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

LOPT0 to LOPT7	Last Out-Pointer of the Transmit History List for the DIAG_CH
0... 47 message buffer	Reading the CnLOPT register delivers the message buffer number from which the last message frame has been sent to DIAG CAN bus.

Remarks: **1.** As long no message frame has been sent, an undefined value is read from CnLOPT register as long THPM bit is set (1).

2. This register shows the transmit history of the messages from DIAG-CH only but the mirrored messages from RXONLY_CH are included.

14.11.17 CAN Module Transmit History List Get Pointer Register (CnTGPT)

CnTGPT (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	THPM	TOVF	24H	02H
Initial Value	0	0	0	0	0	0	1	0		
	15	14	13	12	11	10	9	8		
	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0	25H	undefined
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

CnTGPT (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	0	clear TOVF	24H	
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	25H	

TOVF	Transmit History List Overflow
0	Upon successful message frame transmission the corresponding message buffer number is logged into the THL.
1	At least 7 entries have been stored since the host processor has serviced the THL last time (i.e. read CnTGPT). Thus the sequence of transmissions can not be recovered completely now. The first 6 entries are sequentially stored while the last entry can have been overwritten by newly transmitted messages multiple times because all buffer numbers are stored at position LOPT-1 when TOVF bit is set. **Note**

THPM	Transmit History List Pointers Match
0	There is at least one unread message in the THL
1	There is no unread message in the THL

TGPT0 to TGPT7	Transmit History List Get Pointer
0... 48 message buffer	Reading the CnTGPT register delivers the message buffer number, which can be loaded with new data for the next transmission.

Note: The THL will be updated, but the LOPT pointer will not be incremented. Always the position the LOPT pointer -1 is pointing to is overwritten.

Remark: This register shows the transmit history of the messages from DIAG-CH only.

14.11.18 DIAG_CH CAN Module Time Stamp Register (CnTS)

CnTS (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	TSLOCK	TSSEL	TSEN	26H	00H
Initial Value	0	0	0	0	0	0	0	0		

	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	27H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnTS (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	clear TSLOCK	clear TSSEL	clear TSEN	26H	

	15	14	13	12	11	10	9	8		
	0	0	0	0	0	set TSLOCK	set TSSEL	set TSEN	27H	

Remarks: 1. The displayed CnTS register provides only the necessary bits for the Basic Time Stamp function.
The Advanced Time Stamp function requires a modified hardware.

2. The Basic Time Stamp function must not be used when the DIAG_CH CAN module operates in 'Normal Operating mode with Automatic Block Transmission'

TSEN	Time Stamp Signal Generation Enable bit
0	The time stamp signal TSOUT generation is disabled.
1	The time stamp signal TSOUT generation is enabled.

TSSEL	Time Stamp Capture Event Selection bit
0	The time stamp capture event is the SOF event (Start-of-Frame on the DIAG-CAN bus)
1	The time stamp capture event is the EOF event (the last bit of the End-of-Frame field. TSOUT signal is generated at the sample point of the last bit in the EOF field)

TSLOCK	Time Stamp Lock Function Enable bit
0	The time stamp lock function is disabled. Upon every selected time stamp capture event the TSOUT signal is toggled.
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.

14.11.19 RXONLY_CH CAN Module Control Register (CnCTRL_R)

CnCTRL_R

(read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	CCERC	0	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0	28H	00H
Initial Value	0	0	0	0	0	0	0	0		

	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	RSTAT	0	29H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnCTRL_R

(write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	clear VALID	clear PSMODE1	Clear PSMODE0	clear OPMODE2	clear OPMODE1	clear OPMODE0	28H	

	15	14	13	12	11	10	9	8	
	set CCERC	0	0	set PSMODE1	Set PSMODE0	Set OPMODE2	set OPMODE1	set OPMODE0	29H

OPMODE2	OPMODE1	OPMODE0	Operational Mode
0	0	0	No operational mode selected (RXONLY-CH CAN module is in INIT mode)
0	0	1	Prohibited to use / ignored by hardware
0	1	0	
0	1	1	"Receive-only mode"
1	0	0	Prohibited to use / ignored by hardware
1	0	1	
1	1	0	"Mirror mode"
1	1	1	“Mirror mode with Transfer ID Filter function (w/ TIF)”

PSMODE1	PSMODE0	Power Save Mode
0	0	No power save mode selected (RXONLY_CH CAN module is in INIT mode or in one of the operational modes)
0	1	SLEEP mode
1	0	Prohibited to use / ignored by hardware
1	1	STOP mode

VALID	Valid Reception bit
0	No valid message frame reception into the RXONLY_CH CAN Protocol Transfer Layer since the VALID bit had been cleared (0) last time.
1	Valid message frame reception into the RXONLY_CH CAN Protocol Transfer Layer since the VALID bit had been cleared (0) last time.

Caution: User must not set the VALID bit (1) by using set VALID bit.
A valid reception does not require an acceptance of the message frame into a receive message buffer (data or remote frame).
Before switching from INIT mode to any operational mode the user must clear the VALID bit (0).
In case only two CAN nodes are connected to a CAN bus and one of the CAN nodes is in "Normal Operating mode" transmitting message frames while the other CAN node is in "Receive-only mode", the VALID bit will be set (1) not before the transmitting node becomes error passive.

CCERC	RXONLY_CH CAN Clear Error Counter bit
0	While in INIT mode the RXONLY_CH CAN module error counter CnERC_R will not be cleared.
1	While in INIT mode the RXONLY_CH CAN module error counter CnERC_R will be cleared

RSTAT	RXONLY_CH CAN Reception status
0	No receive activity on the RXONLY CAN bus
1	Receive activity on the RXONLY CAN bus

Remarks: 1. RSTAT is set under the following condition:
SOF of RX frame

2. RSTAT is cleared under the following condition:
After detecting recessive bit at second bit of inter frame space
Transition to INIT mode at 1st bit of 'interframe space'

14.11.20 RXONLY_CH CAN Module Last Error Code Register (CnLEC_R)

CnLEC_R									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	0	LEC2	LEC1	LEC0	2AH	00H
Initial Value	0	0	0	0	0	0	0	0		

Remark: The actual content of the CnLEC_R is not cleared by switching the RXONLY_CH CAN module from an operational mode to the INIT mode.
When CPU attempts to write CnLEC_R with data other than 00h, it is simply ignored.

LEC2	LEC1	LEC0	Last Error Code of the Protocol Error Type
0	0	0	No error
0	0	1	Stuff error
0	1	0	Form error
1	1	0	CRC error
0	1	1	Unused
1	0	0	
1	0	1	
1	1	1	

14.11.21 RXONLY_CH CAN Module Error Counter (CnERC_R)

CnERC_R (read-only)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	0	0	0	2CH	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0	2DH	00H
Initial Value	0	0	0	0	0	0	0	0		

14.11.22 RXONLY_CH CAN Module Interrupt Enable Register (CnIE_R)

CnIE_R (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	CIE6	CIE5	0	CIE3	CIE2	CIE1	0	2EH	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	2FH	00H
Initial Value	0	0	0	0	0	0	0	0		

CnIE_R (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	clear CIE6	clear CIE5	0	clear CIE3	Clear CIE2	Clear CIE1	0	2EH	00H
	15	14	13	12	11	10	9	8		
	0	set CIE6	set CIE5	0	set CIE3	Set CIE2	set CIE1	0	2FH	

CIE1 to CIE6	RXONLY_CH CAN Interrupt Enable bits
0	Corresponding interrupt request signal INTxx is not generated when CINTSx is set by the interrupt source event.
1	Corresponding interrupt request signal INTxx is generated when CINTSx is set by the interrupt source event.

14.11.23 RXONLY-CAN Module Interrupt Status Register (CnINTS_R)

CnINTS_R (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	CINTS6	CINTS5	0	CINTS3	CINTS2	CINTS1	0	30H	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	31H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnINTS_R (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	clear CINTS6	clear CINTS5	0	clear CINTS3	clear CINTS2	clear CINTS1	0	30H	
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	31H	

CINTS1 to CINTS6	RXONLY_CH CAN Interrupt Status bits
0	The related interrupt source event is not pending
1	The related interrupt source event is pending

Interrupt status bit	Related interrupt source event
CINTS1	RXONLY_CH CAN module interrupt status bit for interrupt event 'Valid message frame reception in the FIFO area of message buffer
CINTS2	RXONLY_CH CAN module error state interrupt status
CINTS3	RXONLY_CH CAN module protocol error interrupt status
CINTS5	RXONLY_CH CAN module wake-up from SLEEP mode interrupt status bit **Note**
CINTS6	RXONLY_CH CAN module FIFO overflow interrupt status

Note: Only the wake-up from SLEEP mode by CAN bus activity generates the CINTS5 signal. The SLEEP mode release by CPU will not generate the CINTS5 signal.

14.11.24 RXONLY_CH CAN Module Bit-Rate Prescaler Register (CnBRP_R)

CnBRP_R									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0	32H	FFH
Initial Value	1	1	1	1	1	1	1	1		

TQPRS0 to TQPRS7	Time Quanta clock prescaler
0	$\phi_{TQ} = \phi_{CANMOD} / 1$
1	$\phi_{TQ} = \phi_{CANMOD} / 2$
N	$\phi_{TQ} = \phi_{CANMOD} / (n+1)$
...	...
255	$\phi_{TQ} = \phi_{CANMOD} / 256$

14.11.25 RXONLY_CH CAN Bit Rate Register (CnBTR_R)

CnBTR_R									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10	34H	0FH
Initial Value	0	0	0	0	1	1	1	1		
	15	14	13	12	11	10	9	8		
	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20	35H	37H
Initial Value	0	0	1	1	0	1	1	1		

TSEG13	TSEG12	TSEG11	TSEG10	Length of Time Segment 1
0	0	0	0	Prohibited
0	0	0	1	2 TQ
0	0	1	0	3 TQ
0	0	1	1	4 TQ
0	1	0	0	5 TQ
0	1	0	1	6 TQ
0	1	1	0	7 TQ
0	1	1	1	8 TQ
1	0	0	0	9 TQ
1	0	0	1	10 TQ
1	0	1	0	11TQ
1	0	1	1	12TQ
1	1	0	0	13TQ
1	1	0	1	14TQ
1	1	1	0	15TQ
1	1	1	1	16TQ

TSEG22	TSEG21	TSEG20	Length of Time Segment 2
0	0	0	1TQ
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

SJW1	SJW0	Length of Synchronisation Jump Width
0	0	1TQ
0	1	2 TQ
1	0	3 TQ
1	1	4 TQ

14.11.26 RXONLY-CH CAN Module Last In-Pointer Register (CnLIPT_R)

CnLIPT_R									Address Offset	Initial Value
(read-only)	7	6	5	4	3	2	1	0		
	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0	36H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

LIPT0 to LIPT7	Last In-Pointer of the Receive History List for the RXONLY-CH
0... message buffer47	Reading the CnLIPT_R register delivers the message buffer number in which the last data frame has been saved or the last remote frame has been stored into the message buffer area of RXONLY-CH. The user can only evaluate CnLIPT_R when both of the following conditions are met: - the CAN channel is in RECONLY or MIRROR mode. - A newly received message on RXONLY_CH was signalled by interrupt CINTS1 in CnINTS_R. As a pre-condition RXONLY-CH needs to have been put in INIT or STOP mode beforehand at least once. The usage of CnLIPT_R requires proper interrupt handling.

Remark: As long no data frame has been stored into a message buffer or no received remote frame has been assigned, an undefined value is read from CnLIPT_R register. The user must not read the CnLIPT_R register until first message acceptance to the buffer from RXONLY-CH after switching the GOM bit.

This register shows the reception history of the messages from RXONLY-CH only.

14.11.27 RXONLY_CH CAN Module Time Stamp Register (CnTS_R)

CnTS_R (read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	TSLOCK	TSSEL	TSEN	38H	00H
Initial Value	0	0	0	0	0	0	0	0		
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	0	0	0	39H	00H
Initial Value	0	0	0	0	0	0	0	0		

CnTS_R (write)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	0	clear TSLOCK	clear TSSEL	clear TSEN	38H	
	15	14	13	12	11	10	9	8		
	0	0	0	0	0	set TSLOCK	set TSSEL	set TSEN	39H	

Remarks: **1.** The displayed CnTS_R register provides only the necessary bits for the Basic Time Stamp function.
Advanced Time Stamp function requires a modified hardware.

2. The Basic Time Stamp function must not be used when the DIAG_CH CAN module operates in 'Normal Operating mode with Automatic Block Transmission.

TSEN	Time Stamp Signal Generation Enable bit
0	The time stamp signal TSOUT_R generation is disabled.
1	The time stamp signal TSOUT_R generation is enabled.

TSSEL	Time Stamp Capture Event Selection bit
0	The time stamp capture event is the SOF event (Start-of-Frame on the RXONLY-CH CAN-bus)
1	The time stamp capture event is the EOF event (the last bit of the End-of-Frame field. TSOUT_R signal is generated at the sample point of the last bit in the EOF field)

TSLOCK	Time Stamp Lock Function Enable bit
0	The time stamp lock function is disabled. Upon every selected time stamp capture event the TSOUT_R signal is toggled.
1	The time stamp lock function is enabled. The TSOUT_R signal generation is locked after a data frame was successfully received in message buffer 0.

Caution: The TSLOCK function on RXONLY_CH is only invoked when Mirror mode, mirror mode w/ TIF or RECONLY mode are active. At all other operational modes of the DAFCAN, the TSLOCK function via TSOUT_D is active for all message buffers.

14.11.28 RXONLY_CH CAN Module Bus Selector (CnBSEL_R)

CnBSEL_R									Address Offset	Initial Value
(read / write)	7	6	5	4	3	2	1	0		
	0	0	0	0	0	BSEL2	BSEL1	BSEL0	3AH	00H
Initial Value	0	0	0	0	0	0	0	0		

BSEL2 to BSEL0	Bus selector for selecting to be monitored CAN BUS
0... CAN bus 4	Selecting the connection of RXONLY-ch

BSEL2	BSEL1	BSEL0	Selected CAN BUS input source
0	0	0	DIAG_CH CAN-bus
0	0	1	CAN-bus input 1
0	1	0	CAN-bus input 2
0	1	1	CAN-bus input 3
1	0	0	CAN-bus input 4
1	0	1	CAN-bus input 5
1	1	0	CAN-bus input 6
1	1	1	CAN-bus input 7

Remarks: 1. When CPU attempts to write CnBESL_R in the mode other than INIT, it is simply ignored.

2. When CnBSEL_R is “000b”, the RXONLY-ch is connected to the DIAG-CH CAN BUS.

14.11.29 Registers of Message Buffers

Table 14-12: CAN Message Buffer Register (CPU read access)

Register Name	Addr.-Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/ 9	Bit 0/ 8
MDATA0	00H	MDATA07	MDATA06	MDATA05	MDATA04	MDATA03	MDATA02	MDATA01	MDATA00
MDATA1	01H	MDATA17	MDATA16	MDATA15	MDATA14	MDATA13	MDATA12	MDATA11	MDATA10
MDATA2	02H	MDATA27	MDATA26	MDATA25	MDATA24	MDATA23	MDATA22	MDATA21	MDATA20
MDATA3	03H	MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30
MDATA4	04H	MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40
MDATA5	05H	MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA51	MDATA50
MDATA6	06H	MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60
MDATA7	07H	MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70
MDLC	08H	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0
MCONF	09H	OWS	RTR	MT2	MT1	MT0	MA2	MA1	MA0
MIDL	0AH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	0BH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
MIDH	0CH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	0DH	IDE	0	0	ID28	ID27	ID26	ID25	ID24
MCTRL	0EH	0	0	0	MOW	IE	DN	TRQ	RDY
	0FH	0	0	MUC	0	0	0	0	0
	10H to 1FH	Access prohibited							

Table 14-13: CAN Message Buffer Register (CPU write access)

Register Name	Addr.-Offset	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
MDATA0	00H	MDATA07	MDATA06	MDATA05	MDATA04	MDATA03	MDATA02	MDATA01	MDATA00
MDATA1	01H	MDATA17	MDATA16	MDATA15	MDATA14	MDATA13	MDATA12	MDATA11	MDATA10
MDATA2	02H	MDATA27	MDATA26	MDATA25	MDATA24	MDATA23	MDATA22	MDATA21	MDATA20
MDATA3	03H	MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30
MDATA4	04H	MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40
MDATA5	05H	MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA51	MDATA50
MDATA6	06H	MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60
MDATA7	07H	MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70
MDLC	08H	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0
MCONF	09H	OWS	RTR	MT2	MT1	MT0	MA2	MA1	MA0
MIDL	0AH	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	0BH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
MIDH	0CH	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
	0DH	IDE	0	0	ID28	ID27	ID26	ID25	ID24
MCTRL	0EH	0	0	0	clear MOW	clear IE	clear DN	clear TRQ	clear RDY
	0FH	0	0	0		set IE	set DN	set TRQ	set RDY
	10H to 1FH	Access prohibited							

14.11.30 Message Data Byte 0 Register (MDATA0m)

MDATA0m	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	MDATA07	MDATA06	MDATA05	MDATA04	MDATA03	MDATA02	MDATA01	MDATA00	00H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.31 Message Data Byte 1 Register (MDATA1m)

MDATA1m	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	MDATA17	MDATA16	MDATA15	MDATA14	MDATA13	MDATA12	MDATA11	MDATA10	01H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.32 Message Data Byte 2 Register (MDATA2m)

MDATA2m	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	MDATA27	MDATA26	MDATA25	MDATA24	MDATA23	MDATA22	MDATA21	MDATA20	02H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.33 Message Data Byte 3 Register (MDATA3m)

MDATA3m	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30	03H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.34 Message Data Byte 4 Register (MDATA4m)

MDATA4m	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40	04H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.35 Message Data Byte 5 Register (MDATA5m)

MDATA5m									Address Offset	Initial Value
	7	6	5	4	3	2	1	0		
	MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA01	MDATA50	05H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.36 Message Data Byte 6 Register (MDATA6m)

MDATA6m									Address Offset	Initial Value
	7	6	5	4	3	2	1	0		
	MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60	06H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.37 Message Data Byte 7 Register (MDATA7m)

MDATA7m									Address Offset	Initial Value
	7	6	5	4	3	2	1	0		
	MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70	07H	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

14.11.38 Message Data Length Code Register (MDLCm)

MDLCm	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0	08H	0000xxxxB
Initial Value	0	0	0	0	undef.	undef.	undef.	undef.		

MDLC3	MDLC2	MDLC1	MDLC0	Message Data Length Code
0	0	0	0	Data frame contains 0 byte in the data field
0	0	0	1	Data frame contains 1 byte in the data field
0	0	1	0	Data frame contains 2 bytes in the data field
0	0	1	1	Data frame contains 3 bytes in the data field
0	1	0	0	Data frame contains 4 bytes in the data field
0	1	0	1	Data frame contains 5 bytes in the data field
0	1	1	0	Data frame contains 6 bytes in the data field
0	1	1	1	Data frame contains 7 bytes in the data field
1	0	0	0	Data frame contains 8 bytes in the data field
1	0	0	1	Data frame contains 8 bytes in the data field **Note**
1	0	1	0	Data frame contains 8 bytes in the data field **Note**
1	0	1	1	Data frame contains 8 bytes in the data field **Note**
1	1	0	0	Data frame contains 8 bytes in the data field **Note**
1	1	0	1	Data frame contains 8 bytes in the data field **Note**
1	1	1	0	Data frame contains 8 bytes in the data field **Note**
1	1	1	1	Data frame contains 8 bytes in the data field **Note**

Note: The valid value range for the bit-string MDLC[3:0] is 00H to 0FH. In case the value written to MDLC[3:0] is greater than 8, the DLC value in the message frame must be transferred as programmed but only 8 data bytes are transferred in the data field.

The user need to write always 0000B to the MDLCm register bits 7 to 4.

14.11.39 Message Configuration Register (MCONFm)

MCONFm	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	OWS	RTR	MT2	MT1	MT0	MA2	MA1	MA0	09H	xxxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

MA2	MA1	MA0	Message Buffer Assignment
0	0	0	Message buffer is not assigned to any CAN I/F channel
0	0	1	Message buffer is assigned to CAN I/F channel 1
0	1	0	N/A due to new architecture
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Remarks: **1.** In case the particular CAN I/F channel is not implemented into an AFCAN macro derivative, the message buffer is not assigned to any CAN I/F channel.

2. Due to the new concept for multi-channel AFCAN, the MA[2:0] bit string has become partly obsolete. All combinations referencing a channel number larger than 1, do not assign the message buffer to any CAN I/F channel. However MA[2:0] = 1 needs to be assigned to let the message buffer be serviced by its dedicated CAN I/F channel (RXONLY-CH or DIAG-CH).

3. The DIAG macro assigns RXONLY-CH and DIAG-CH to a particular message buffer automatically by the OPMODE/PSMODE of the RXONLY-CH.

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 without a link to a mask in the assigned CAN I/F channel
0	1	0	message buffer is a receive message buffer with a link to the mask1 in the assigned CAN I/F channel
0	1	1	message buffer is a receive message buffer with a link to the mask2 in the assigned CAN I/F channel
1	0	0	message buffer is a receive message buffer with a link to the mask3 in the assigned CAN I/F channel
1	0	1	message buffer is a receive message buffer with a link to the mask4 in the assigned CAN I/F channel
1	1	0	prohibited for user / ignored for hardware
1	1	1	prohibited for user / ignored for hardware

RTR	Remote Request [Note]
0	Message frame to be sent is a data frame
1	Message frame to be sent is a remote frame

Note: The RTR bit does only determine the message frame for the transmission process from a message buffer defined as transmit message buffer (i.e. MT[2:0] = 00H).
Upon a valid reception of a remote frame RTR remains cleared (0) in the corresponding transmit message buffer. However, if the buffer is assigned to RXONLY_CH and the Mirror or RECONLY mode is active, the RTR bit will be stored as received from the monitored bus.
In case a transmit message buffer is prepared to sent a remote frame by setting 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 (i.e. no interrupt generation, no update of the DN flag, no update of the MDLC[3:0] bit-string and no update of the Receive History List)

OWS	Overwrite Select
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 denotes a receive message buffer, which has already accepted a data frame (i.e. DN is set (1)), but the CPU has not yet read that message buffer.
OWS is not recognized by RXONLY_CH when operating in Mirror or RECONLY mode. Rather the “do not overwrite” mechanism is integrated into the RX-Store process as an intrinsic function.

14.11.40 Message Identifier Registers (MIDLm, MIDHm)

MIDLm									Address Offset	Initial Value
(read/write)	7	6	5	4	3	2	1	0		
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	0AH	undef.
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	0BH	
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		

MIDHm										
(read/write)	7	6	5	4	3	2	1	0		
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16	0CH	x00xxxxxx xxxxxxxB
Initial Value	undef.	undef.	undef.	undef.	undef.	undef.	undef.	undef.		
	15	14	13	12	11	10	9	8		
	IDE	0	0	ID28	ID27	ID26	ID25	ID24	0DH	
Initial Value	undef.	0.	0	undef.	undef.	undef.	undef.	undef.		

ID28 toID0	Message Identifier
ID[28:18] = 0... (2^{11} – 1)	range for the 11-bit Standard identifier values
ID[28:0] = 0... (2^{29} – 1)	range for the 29-bit Extended identifier values

IDE	Identifier Extension bit
0	11-bit Standard identifier. **Note**
1	29-bit Extended identifier.

Note: The bits ID17 to ID0 are not used and may contain undefined values.

When RXONLY_CH is operated in Mirror or RECONLY mode, the IDE bit will be overwritten with the actually received frame format (i.e. standard or extended).

14.11.41 Message Control Register (MCTRLm)

MCTRLm (Read)	7	6	5	4	3	2	1	0	Address Offset	Initial Value
	0	0	0	MOW	IE	DN	TRQ	RDY	0EH	00x000000 00xx000B
Initial Value	0	0	0	undef.	undef.	0	0	0		

	15	14	13	12	11	10	9	8	
	0	0	MUC	0	0	0	0	0	0FH
Initial Value	0	0	undef.	0	0	0	0	0	

MCTRLm (Write)	7	6	5	4	3	2	1	0	Address	Initial Value
	0	0	0	clear MOW	clear IE	clear DN	clear TRQ	clear RDY	0EH	

	15	14	13	12	11	10	9	8	
	0	0	0	0	set IE	set DN	set TRQ	set RDY	0FH

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).

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.

DN	Message Buffer Data New bit
0	No remote frame has been stored into the message buffer (message buffer is defined as transmit message buffer (MT[2:0] = 00H)) No data frame has been stored into the message buffer (message buffer is defined as receive message buffer (MT[2:0] > 00H)).
1	A remote frame has been stored into the message buffer (message buffer is defined as transmit message buffer (MT[2:0] = 00H)) A data frame has been stored into the message buffer (message buffer is defined as receive message buffer (MT[2:0] > 00H)).

Remark: The user must not set the DN flag (1) by using the set-DN instruction.
Special care applies to DN, TRQ, and RDY bit when Mirror Mode, mirror mode w/ TIF or RECONLY mode is activated. Please, refer to chapter 14.8 ”Operational Modes of RXONLY_CH” on page 684 for details.

IE	Interrupt Enable for message buffer interrupt event
0	Interrupt generation disabled for the following events: Interrupt events linked to CINTS0 interrupt status bit in CnINTS register {i.e. at MT[2:0] = 0 'Data frame successfully transmitted from message buffer m', 'Remote frame successfully transmitted from message buffer m'} Interrupt events linked to CINTS1 interrupt status bit in CnINTS register {i.e. for MT[2:0] = 1, 2, 3, 4 or 5 'Valid data frame reception in message buffer m' and for MT[2:0] = 0 'Valid remote frame reception in message buffer m'} Interrupt events linked to CINTS1 interrupt status bit in CnINTS_R register when operated in MIRROR or RECONLY mode {i.e. ''Valid data or remote frame reception in message buffer m' for m= 32 – 47}
1	Interrupt generation enabled for the following events: Interrupt events linked to CINTS0 interrupt status bit in CnINTS register {i.e. MT[2:0] = 0 'Data frame successfully transmitted from message buffer m', 'Remote frame successfully transmitted from message buffer m'} Interrupt events linked to CINTS1 interrupt status bit in CnINTS register {MT[2:0] = 1, 2, 3, 4 or 5 'Valid data frame reception in message buffer m' and for MT[2:0] = 0 'Valid remote frame reception in message buffer m' Interrupt events linked to CINTS1 interrupt status bit in CnINTS_R register when operated in MIRROR or RECONLY mode {i.e. ''Valid data or remote frame reception in message buffer m' for m= 32 – 47}

MOW	Message Buffer Overwritten flag
0	The message buffer has not been overwritten by a newly received data frame.
1	The message buffer has been overwritten by a newly received data frame.

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

Remark: The value of MUC bit is undefined until the first message has been received successfully.

Chapter 15 Port Functions

15.1 Features

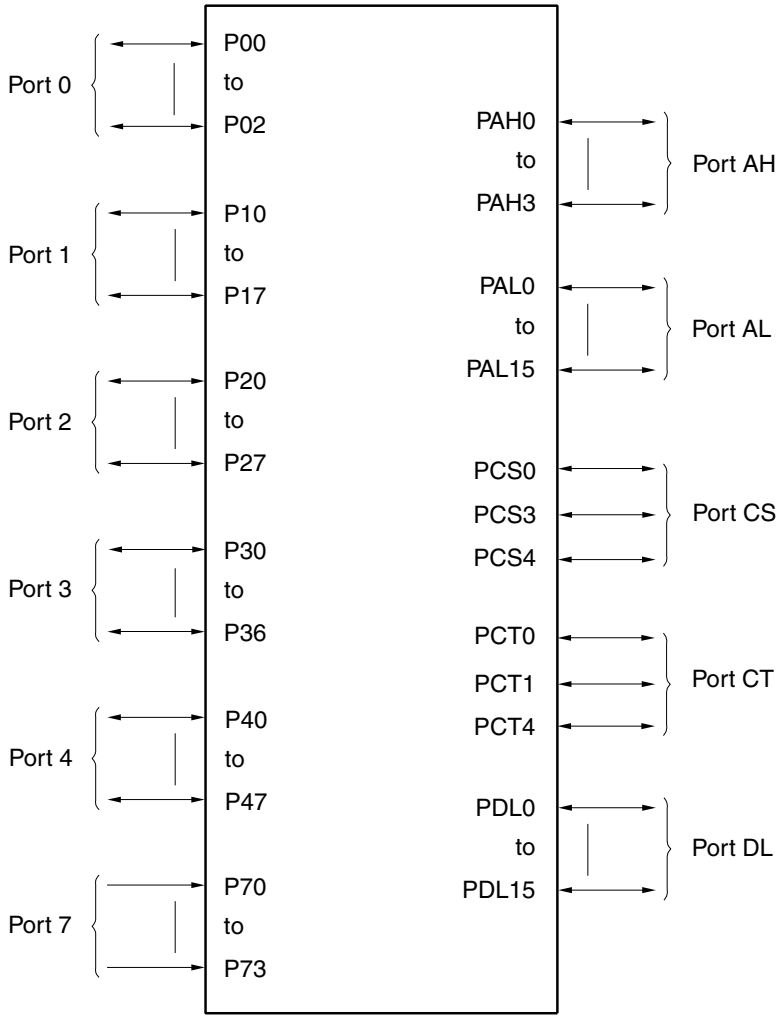
- Input/Output ports (5 V): 76
- Input ports (5 V): 4
- Ports alternate as input/output pins of other peripheral functions
- Input or output can be specified in bit units

15.2 Port Configuration

The μPD70F3178 incorporates a total of 80 input/output ports (4 ports are input only). The ports are named ports P0 through P4, P7, and PAH, PAL, PCS, PCT and PDL.

The configuration is shown below.

Figure 15-1: Port Configuration



(1) Functions of each port

The µPD70F3178 has the ports shown below.
Any port can operate in 8- or 1-bit units and can provide a variety of controls.
Moreover, besides its function as a port, some have functions as the input/output pins of on-chip peripheral I/O in control mode.
Refer to “(3) Port block diagram” for a block diagram of the block type of each port.

Table 15-1: Functions of each port

Port Name	Pin Name	Port Function	Function In Control Mode	Block Type
Port 0	P00 to P02	3-bit input/output	Non maskable interrupt (NMI), external interrupt request input (INTP0)	B,C,D
Port 1	P10 to P17	8-bit input/output	Serial interface input/output (AFCAN0 to AFCAN4)	B
Port 2	P20 to P27	8-bit input/output	Real-time pulse unit (RPU) input/output, external interrupt request input, serial interface input(CRXD5)	C,D
Port 3	P30 to P36	7-bit input/output	Serial interface input/output (UART, CSI, I²C), external interrupt request input	B,C,D,L
Port 4	P40 to P47	8-bit input/output	Serial interface input/output (CAN, CSI), external interrupt input	B
Port 7	P70 to P73	4-bit input (digital input of ANI0 to ANI3)	-	-
Port AH	PAH0 to PAH3	4-bit input/output	External address bus (A16 to A19)	A
Port AL	PAL0 to PAL15	16-bit input/output	External address bus (A0 to A15)	A
Port CS	PCS0, PCS3, PCS4	3-bit input/output	External bus interface control signal output	A
Port CT	PCT0, PCT1, PCT4	3-bit input/output	External bus interface control signal output	A
Port DL	PDL0 to PDL15	16-bit input/output	External bus interface data signal input/output	I

(2) Functions of each port pin on reset and registers that set port or control mode

Table 15-2: Port Pin Functions (1/2)

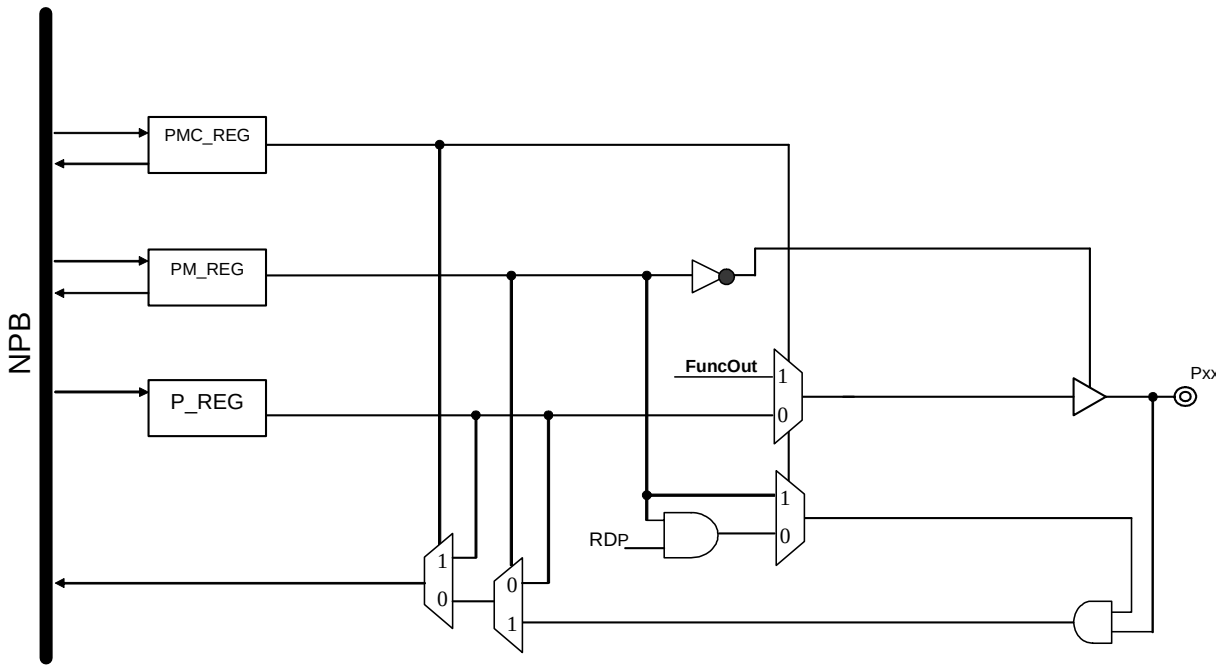
Port Name	Pin Name	Pin Function after Reset	Mode Setting Register
Port 0	P00/NMI	P00 (Input mode)	PMC0
	P01/INTP0	P01 (Input mode)	
	P02	P02 (Input mode)	
Port 1	P10/CRXD0	P10 (Input mode)	PMC1
	P11/CTXD0	P11 (Input mode)	
	P12/CRXD1	P12 (Input mode)	
	P13/CTXD1	P13 (Input mode)	
	P14/CRXD2	P14 (Input mode)	
	P15/CTXD2	P15 (Input mode)	
	P16/CRXD3	P16 (Input mode)	
	P17/CTXD3	P17 (Input mode)	
Port 2	P20/TXP00/INTP3	P20 (Input mode)	PMC2
	P21/TXP01/INTP4	P21 (Input mode)	
	P22/TXP10/INTP5	P22 (Input mode)	
	P23/TXP11/INTP6	P23 (Input mode)	
	P24/TXP20/INTP7	P24 (Input mode)	
	P25/TXP21/INTP8	P25 (Input mode)	
	P26/TTRGP2/INTP9	P26 (Input mode)	
	P27/CRXD5/INTP10	P27 (Input mode)	
Port 3	P30/RXDA0/INTP1	P30 (Input mode)	PMC3
	P31/TXDA0	P31 (Input mode)	
	P32/RXDA1/INTP2	P32 (Input mode)	
	P33/TXDA1	P33 (Input mode)	
	P34/SIB2	P34 (Input mode)	
	P35/SOB2/SDA0	P35 (Input mode)	
	P36/$\overline{\text{SCKB2}}$/$\overline{\text{SCL}}$	P36 (Input mode)	
Port 4	P40/SIB0	P40 (Input mode)	PMC4
	P41/SOB0	P41 (Input mode)	
	P42/$\overline{\text{SCKB0}}$	P42 (Input mode)	
	P43/SIB1	P43 (Input mode)	
	P44/SOB1	P44 (Input mode)	
	P45/$\overline{\text{SCKB1}}$	P45 (Input mode)	
	P46/CRXD4	P46 (Input mode)	
	P47/CTXD4	P47 (Input mode)	
Port 7	P70/ ANI0	P70 (Input)/ ANI0	-
	P71/ ANI1	P71 (Input)/ ANI1	
	P72/ ANI2	P72 (Input)/ ANI2	
	P73/ ANI3	P73 (Input)/ ANI3	

Table 15-2: Port Pin Functions (2/2)

Port Name	Pin Name	Pin Function after Reset	Mode Setting Register
Port AH	PAH0/A16	PAH0 (Input mode)	PMCAH
	PAH1/A17	PAH1 (Input mode)	
	PAH2/A18	PAH2 (Input mode)	
	PAH3/A19	PAH3 (Input mode)	
Port AL	PAL0/A0	PAL0 (Input mode)	PMCAL
	PAL1/A1	PAL1 (Input mode)	
	PAL2/A2	PAL2 (Input mode)	
	PAL3/A3	PAL3 (Input mode)	
	PAL4/A4	PAL4 (Input mode)	
	PAL5/A5	PAL5 (Input mode)	
	PAL6/A6	PAL6 (Input mode)	
	PAL7/A7	PAL7 (Input mode)	
	PAL8/A8	PAL8 (Input mode)	
	PAL9/A9	PAL9 (Input mode)	
	PAL10/A10	PAL10 (Input mode)	
	PAL11/A11	PAL11 (Input mode)	
	PAL12/A12	PAL12 (Input mode)	
	PAL13/A13	PAL13 (Input mode)	
	PAL14/A14	PAL14 (Input mode)	
	PAL15/A15	PAL15 (Input mode)	
Port CS	PCS0/$\overline{\text{CS0}}$	$\overline{\text{CS0}}$ (Chip select output mode)	PMCCS
	PCS3/$\overline{\text{CS3}}$	PCS3 (Input mode)	
	PCS4/$\overline{\text{CS4}}$	PCS4 (Input mode)	
Port CT	PCT0/$\overline{\text{WR0}}$	PCT0 (Input mode)	PMCCT
	PCT1/$\overline{\text{WR1}}$	PCT1 (Input mode)	
	PCT4/$\overline{\text{RD}}$	$\overline{\text{RD}}$ (Read strobe signal output mode)	
Port DL	PDL0/D0	PDL0 (Input mode)	PMCDL
	PDL1/D1	PDL1 (Input mode)	
	PDL2/D2	PDL2 (Input mode)	
	PDL3/D3	PDL3 (Input mode)	
	PDL4/D4	PDL4 (Input mode)	
	PDL5/D5	PDL5 (Input mode)	
	PDL6/D6	PDL6 (Input mode)	
	PDL7/D7	PDL7 (Input mode)	
	PDL8/D8	PDL8 (Input mode)	
	PDL9/D9	PDAL9 (Input mode)	
	PDL10/D10	PDL10 (Input mode)	
	PDL11/D11	PDL11 (Input mode)	
	PDL12/D12	PDL12 (Input mode)	
	PDL13/D13	PDL13 (Input mode)	
	PDL14/D14	PDL14 (Input mode)	
	PDL15/D15	PDL15 (Input mode)	

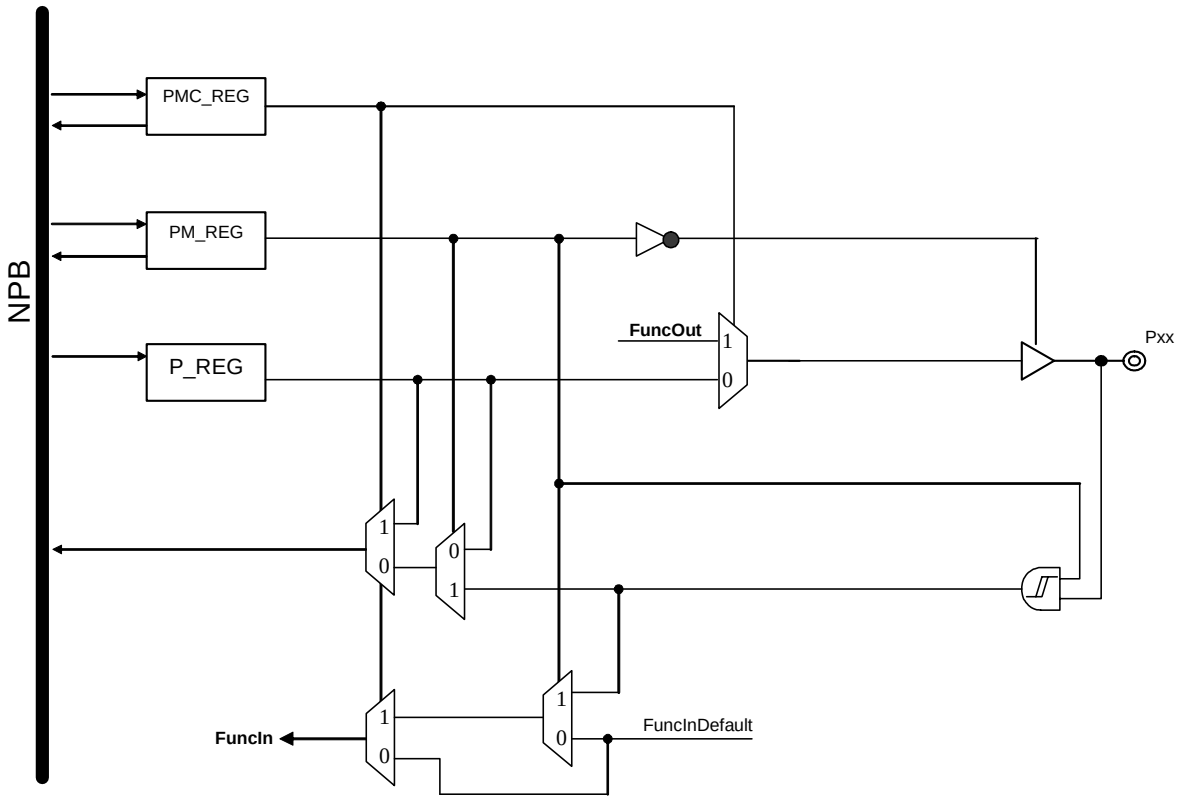
(3) Port block diagram

Figure 15-2: Type A Block Diagram



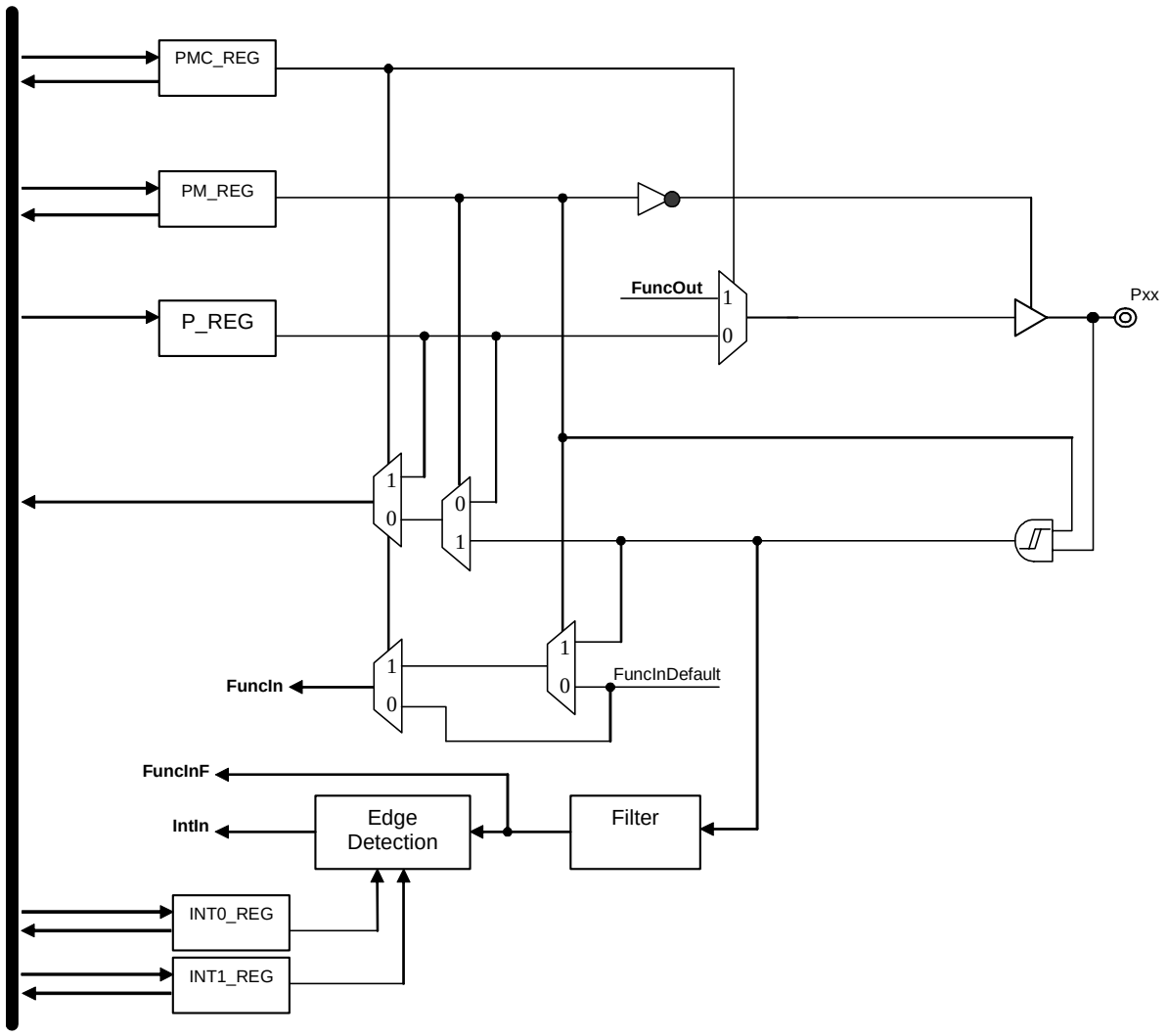
Remark: Port function for PAH, PAL, PCS, PCT

Figure 15-3: Type B Block Diagram



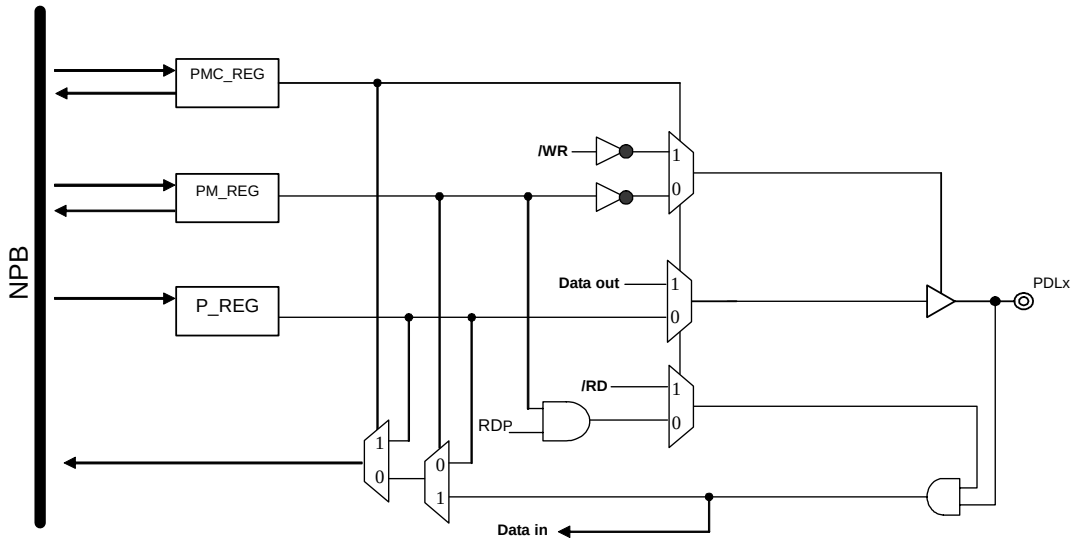
Remark: Port function for P02, P1x, P31, P33, P34, P4x.

Figure 15-4: Type C & D Block Diagram



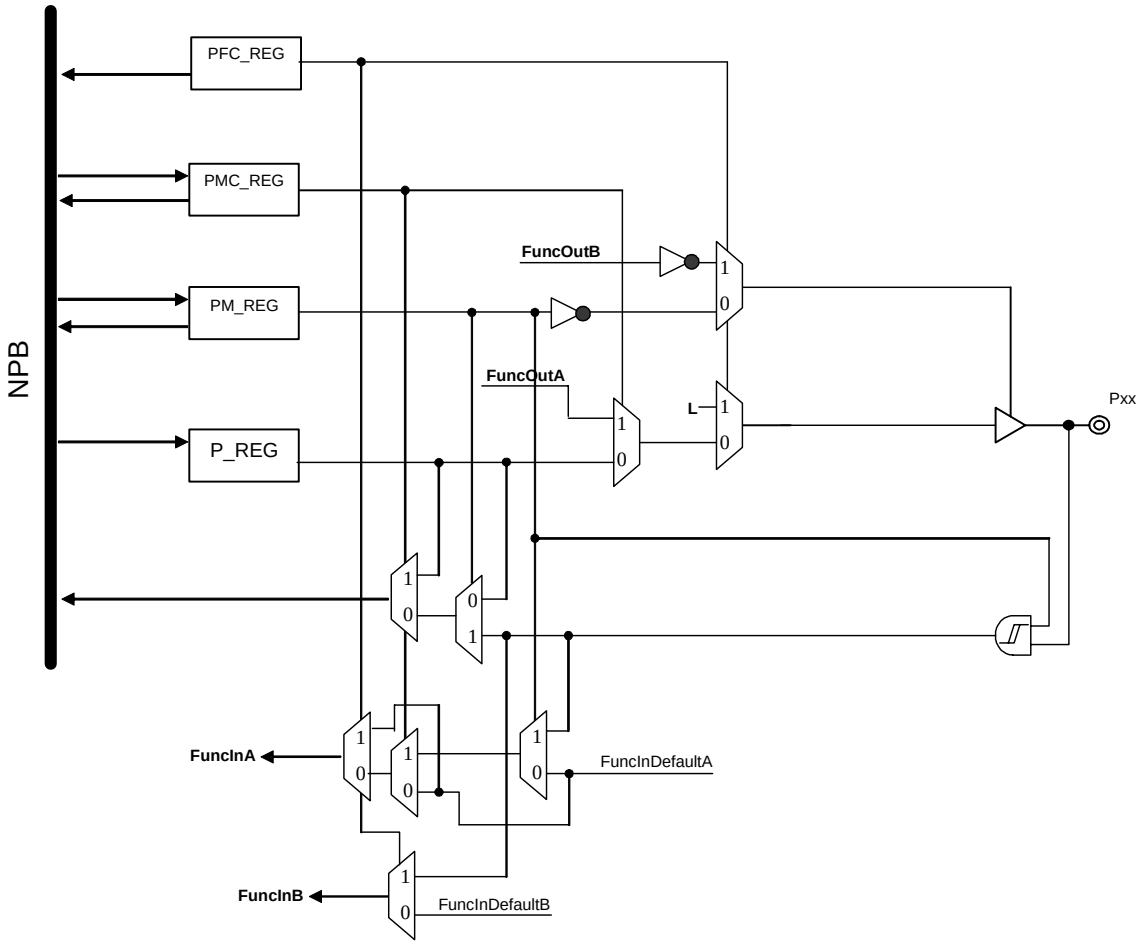
Remark: Port function for P00, P01, P2x, P30, P32

Figure 15-5: Type I Block Diagram



Remark: Port function for PDL

Figure 15-6: Type L Block Diagram



Remark: Port function for P35, P36

15.3 Pin Functions of Each Port

15.3.1 Port 0

Port 0 is a 3-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function.

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

Figure 15-7: Port 0 (P0)

	7	6	5	4	3	2	1	0	Address	At Reset
P0	-	-	-	-	-	P02	P01	P00	FFFFF400H	undefined

Bit Position	Bit Name	Function
2 to 0	P0n (n = 2 to 0)	Input/output port

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

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

Besides functioning as a port, in control mode, it also can operate as the Non maskable interrupt and external interrupt request input.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port 0	P00	NMI	External interrupt request, non maskable interrupt request	
	P01	INTP0		
	P02			

Remarks: **1.** If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register P01 is 07H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(2) Setting in input/output mode and control mode

Port 0 is set in input/output mode using the Port 0 mode register (PM0). In control mode, it is set using the port 0 mode control register (PMC0).

(a) Port 0 mode register (PM0)

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

Figure 15-8: Port 0 Mode Register (PM0)

	7	6	5	4	3	2	1	0	Address	At Reset
PM0	0	0	0	0	0	PM02	PM01	PM00	FFFFF420H	07H

Bit Position	Bit Name	Function
2 to 0	PM0n (n = 2 to 0)	Specifies input/output mode of P0n pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port 0 mode control register (PMC0)

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

Figure 15-9: Port 0 Mode Control Register (PMC0)

	7	6	5	4	3	2	1	0	Address	At Reset
PMC0	0	0	0	0	0	0	0	PMC00	FFFFF440H	00H

Bit Position	Bit Name	Function
0	PMC00	Specifies operation mode of P00 pin 0: Input/output port mode 1: NMI input mode

15.3.2 Port 1

Port 1 is an 8-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function[Note 1].

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

Figure 15-10: Port 1 (P1)

	7	6	5	4	3	2	1	0	Address	At Reset
P1	P17	P16	P15	P14	P13	P12	P11	P10	FFFFF402H	00H[Note 2]

Bit Position	Bit Name	Function
7 to 0	P1n (n = 7 to 0)	Input/output port

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

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

Besides functioning as a port, in control mode, it also can operate as the serial interface (AFCAN0 to AFCAN4) input/output.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port 1	P10	CRXD0	Serial interface (AFCAN0 to AFCAN4) input/output.	
	P11	CTXD0		
	P12	CRXD1		
	P13	CTXD1		
	P14	CRXD2		
	P15	CTXD2		
	P16	CRXD3		
	P17	CTXD3		

Notes: 1. If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register P1 is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(2) Setting in input/output mode and control mode

Port 1 is set in input/output mode using the Port 1 mode register (PM1). In control mode, it is set using the Port 1 mode control register (PMC1).

(a) Port 1 mode register (PM1)

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

Figure 15-11: Port 1 Mode Register (PM1)

	7	6	5	4	3	2	1	0	Address	At Reset
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FFFFF422H	FFH

Bit Position	Bit Name	Function
7 to 0	PM1n (n = 7 to 0)	Specifies input/output mode of P1n pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port 1 mode control register (PMC1)

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

Figure 15-12: Port 1 Mode Control Register (PMC1)

	7	6	5	4	3	2	1	0	Address	At Reset
PMC1	PMC17	PMC16	PMC15	PMC14	PMC13	PMC12	PMC11	PMC10	FFFFF442H	00H

Bit Position	Bit Name	Function
7	PMC17	Specifies operation mode of P17 pin 0: Input/output port mode 1: CTXD3 output mode
6	PMC16	Specifies operation mode of P16 pin 0: Input/output port mode 1: CRXD3 input mode
5	PMC15	Specifies operation mode of P15 pin 0: Input/output port mode 1: CTXD2 output mode
4	PMC14	Specifies operation mode of P14 pin 0: Input/output port mode 1: CRXD2 input mode
3	PMC13	Specifies operation mode of P13 pin 0: Input/output port mode 1: CTXD1 output mode
2	PMC12	Specifies operation mode of P12 pin 0: Input/output port mode 1: CRXD1 input mode
1	PMC11	Specifies operation mode of P11 pin 0: Input/output port mode 1: CTXD2 output mode
0	PMC10	Specifies operation mode of P10 pin 0: Input/output port mode 1: CRXD2 input mode

15.3.3 Port 2

Port 2 is an 8-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function.

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

Figure 15-13: Port 2 (P2)

	7	6	5	4	3	2	1	0	Address	At Reset
P2	P27	P26	P25	P24	P23	P22	P21	P20	FFFFF404H	00H

Bit Position	Bit Name	Function
7 to 0	P2n (n = 7 to 0)	Input/output port

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

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

Besides functioning as a port, in control mode, it also can operate as the serial interface (CRXD5) or real-time pulse unit or external interrupt request input/output.

Remarks: **1.** If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register P2 is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port 2	P20	TXP00/INTP3	Serial interface (CRXD5), real time pulse unit, external interrupt request input/output.	
	P21	TXP01/INTP4		
	P22	TXP10/INTP5		
	P23	TXP11/INTP6		
	P24	TXP20/INTP7		
	P25	TXP21/INTP8		
	P26	INTP9		
	P27	INTP10/CRXD5		

(2) Setting in input/output mode and control mode

Port 2 is set in input/output mode using the Port 2 mode register (PM2). In control mode, it is set using the Port 2 mode control register (PMC2).

(a) Port 2 mode register (PM2)

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

Figure 15-14: Port 2 Mode Register (PM2)

	7	6	5	4	3	2	1	0	Address	At Reset
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FFFFF424H	FFH

Bit Position	Bit Name	Function
7 to 0	PM2n (n = 7 to 0)	Specifies input/output mode of P2n pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port 2 mode control register (PMC2)

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

Figure 15-15: Port 2 Mode Control Register (PMC2)

	7	6	5	4	3	2	1	0	Address	At Reset
PMC2	PMC27	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	PMC20	FFFFF442H	00H

Bit Position	Bit Name	Function
7	PMC27	Specifies operation mode of P27 pin 0: Input/output port mode 1: CRXD5 input mode
6	PMC26	Specifies operation mode of P26 pin 0: Input/output port mode 1: TTRGP2 input mode
5	PMC25	Specifies operation mode of P25 pin 0: Input/output port mode 1: TXP21 input/output mode
4	PMC24	Specifies operation mode of P24 pin 0: Input/output port mode 1: TXP20 input/output mode
3	PMC23	Specifies operation mode of P23 pin 0: Input/output port mode 1: TXP11 input/output mode
2	PMC22	Specifies operation mode of P22 pin 0: Input/output port mode 1: TXP10 input/output mode
1	PMC21	Specifies operation mode of P21 pin 0: Input/output port mode 1: TXP01 input/output mode
0	PMC20	Specifies operation mode of P20 pin 0: Input/output port mode 1: TXP00 input/output mode

15.3.4 Port 3

Port 3 is a 7-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function[Note 1].

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

Figure 15-16: Port 3 (P3)

	7	6	5	4	3	2	1	0	Address	At Reset
P3	0	36	P35	P34	P33	P32	P31	P30	FFFFF406	00H[Note 2]

Bit Position	Bit Name	Function
6 to 0	P3n (n = 6 to 0)	Input/output port

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

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

Besides functioning as a port, in control mode, it also can operate as serial interface (UART, CSI, I²C) input/output and external interrupt request input.

Notes: 1. If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register P3 is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port 3	P30	RXDA0/INTP1	Serial interfaces (UART, CSI, I²C) input/output or external interrupt request input.	
	P31	TXDA0		
	P32	RXDA1/INP2		
	P33	TXDA1		
	P34	SIB2		
	P35	SOB2/SDA		
	P35	$\overline{\text{SCKB2}}$/$\overline{\text{SCL}}$		

(2) Setting in input/output mode and control mode

Port 3 is set in input/output mode using the Port 3 mode register (PM3). In control mode, it is set using the Port 3 mode control register (PMC3).

(a) Port 3 mode register (PM3)

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

Figure 15-17: Port 3 Mode Register (PM3)

	7	6	5	4	3	2	1	0	Address	At Reset
PM3	0	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FFFFF428H	7FH

Bit Position	Bit Name	Function
6 to 0	PM2n (n = 6 to 0)	Specifies input/output mode of P2n pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port 3 mode control register (PMC3)

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

Figure 15-18: Port 3 Mode Control Register (PMC3)

	7	6	5	4	3	2	1	0	Address	At Reset
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	FFFFF448H	00H

Bit Position	Bit Name	Function
6	PMC36	Specifies operation mode of P35 pin 0: Input/output port mode 1: $\overline{SCKB2}$ or $\overline{SCL}$ input output mode
5	PMC35	Specifies operation mode of P35 pin 0: Input/output port mode 1: SOB2 output or SDA input/output mode
4	PMC34	Specifies operation mode of P34 pin 0: Input/output port mode 1: SIB2 input mode
3	PMC33	Specifies operation mode of P33 pin 0: Input/output port mode 1: TXDA1 output mode
2	PMC32	Specifies operation mode of P32 pin 0: Input/output port mode 1: RXDA1 input mode
1	PMC31	Specifies operation mode of P31 pin 0: Input/output port mode 1: TXDA0 output mode
0	PMC30	Specifies operation mode of P30 pin 0: Input/output port mode 1: RXDA0 input mode

(c) Port 3 function control register (PFC3)

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

Figure 15-19: Port 3 Function Control Register (PFC3)

	7	6	5	4	3	2	1	0	Address	At Reset
PFC3	0	PFC36	PFC35	0	0	0	0	0	FFFFF444H	00H

Bit Position	Bit Name	Function
6	PFC36	Specifies operation mode of P35 pin 0: $\overline{\text{SCKB2}}$ input/output mode 1: $\overline{\text{SCL}}$ input/output mode
5	PFC35	Specifies operation mode of P35 pin 0: SOB2 output mode 1: SDA input/output mode

15.3.5 Port 4

Port 4 is a 8-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function.

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

Figure 15-20: Port 4 (P4)

	7	6	5	4	3	2	1	0	Address	At Reset
P4	P47	P46	P45	P44	P43	P42	P41	P40	FFFFF408H	00H

Bit Position	Bit Name	Function
7 to 0	P4n (n = 7 to 0)	Input/output port

Remark: In Input Mode: When the P4 register is read, the pin levels at that time are read. Writing to the P4 register writes the values to that register. This does not affect the input pins.

In Output Mode: When the P4 register is read, the values of P4 are read. Writing to the P4 register writes the values to that register and those values are immediately output.

Besides functioning as a port, in control mode, it also can operate as serial interfaces (CSI, AFCAN) input/output.

Remarks: **1.** If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register P4 is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port 4	P40	SIB0	Serial interface input /output.	
	P41	SOB0		
	P42	$\overline{\text{SCKB0}}$		
	P43	SIB1		
	P44	SOB1		
	P45	$\overline{\text{SCKB1}}$		
	P46	CRXD4		
	P47	CTXD4		

(2) Setting in input/output mode and control mode

Port 4 is set in input/output mode using the Port 4 mode register (PM4). In control mode, it is set using the Port 4 mode control register (PMC4).

(a) Port 4 mode register (PM4)

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

Figure 15-21: Port 4 Mode Register (PM4)

	7	6	5	4	3	2	1	0	Address	At Reset
PM4	P47	P46	PM45	PM44	PM43	PM42	PM41	PM40	FFFFF428H	FFH

Bit Position	Bit Name	Function
7 to 0	PM4n (n = 7 to 0)	Specifies input/output mode of P4n pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port 4 mode control register (PMC4)

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

Figure 15-22: Port 4 Mode Control Register (PMC4)

	7	6	5	4	3	2	1	0	Address	At Reset
PMC4	PMC7	PMC6	PMC45	PMC44	PMC43	PMC42	PMC41	PMC40	FFFFF448H	00H

Bit Position	Bit Name	Function
7	PMC47	Specifies operation mode of P47 pin 0: Input/output port mode 1: CTXD4 output mode
6	PMC46	Specifies operation mode of P46 pin 0: Input/output port mode 1: CRXD4 input mode
5	PMC45	Specifies operation mode of P45 pin 0: Input/output port mode 1: $\overline{\text{SCKB1}}$ input / output mode
4	PMC44	Specifies operation mode of P44 pin 0: Input/output port mode 1: SOB1 output mode
3	PMC43	Specifies operation mode of P43 pin 0: Input/output port mode 1: SIB1 input mode
2	PMC42	Specifies operation mode of P42 pin 0: Input/output port mode 1: $\overline{\text{SCKB0}}$ input/output mode
1	PMC41	Specifies operation mode of P41 pin 0: Input/output port mode 1: SOB0 output mode
0	PMC40	Specifies operation mode of P40 pin 0: Input/output port mode 1: SIB0 input mode

15.3.6 Port 7

Port 7 is an 8-bit input port which is shared with the ADC input channels ANI0 to ANI7. Port 7 holds the digital input values of the A/D input channels ANI0 to ANI3 (P70 to P73). Port mode and port mode control are not available for port 7.

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

Figure 15-23: Port Function Register 7 (P7)

	7	6	5	4	3	2	1	0	Address	Initial value
P7	0	0	0	0	P73/ ANI3	P72/ ANI2	P71/ ANI1	P70/ ANI0	FFFFF40EH	undefined

Bit Position	Bit Name	Function
3 to 0	P73 to P70	The bits P77 to P70 holds the digital input values of the A/D input channels ANI3 to ANI0.

Remarks: 1. Reading the digital value of the analog input channel ANIx is disabled during A/D conversion operation.

2. x = 0 to 3

15.3.7 Port AH

Port AH is a 4-bit input/output port in which input or output can be specified in 1-bit units. After reset, the port AH pins operate as port pin. Each port bit can be independently configured to port input, port output or peripheral function.

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

Figure 15-24: Port AH (PAH)

	7	6	5	4	3	2	1	0	Address	At Reset
PAH	0	0	0	0	PAH3	PAH2	PAH1	PAH0	FFFFF002H	00H[Note]

Note: The reset value of register PAH is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

Remark: If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

Bit Position	Bit Name	Function
3 to 0	PAHn (n = 3 to 0)	Input/output port

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

In Output Mode: When the AH register is read, the values of PAH are read. Writing to the PAH register writes the values to that register and those values are immediately output.

Besides functioning as a port, in control mode, it also can operate as an address bus.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port AH	PAH3 to PAH0	A19 to A16	Address bus	

(2) Setting in input/output mode and control mode

Port AH is set in input/output mode using the port AH mode register (PMAH). In control mode, it is set using the port AH mode control register (PMCAH).

(a) Port AH mode register (PMAH)

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

Figure 15-25: Port AH Mode Register (PMAH)

	7	6	5	4	3	2	1	0	Address	At Reset
PMAH	0	0	0	0	PMAH3	PMAH2	PMAH1	PMAH0	FFFF022FH	00H

Bit Position	Bit Name	Function
7 to 0	PMAHn (n = 7 to 0)	Specifies input/output mode of PAHn pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port AH mode control register (PMCAH)

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

Figure 15-26: Port AH Mode Control Register (PMCAH)

	7	6	5	4	3	2	1	0	Address	At Reset
PMCAH	0	0	0	0	PMCAH3	PMCAH2	PMCAH1	PMCAH0	FFFFF042H	00H

Bit Position	Bit Name	Function
3 to 0	PMCAHn (n = 3 to 0)	Specifies operation mode of PMCAHn pin 0: Input/output port mode 1: A19 to A16 address output

15.3.8 Port AL

Port AL is a 16-bit input/output port in which input or output can be specified in 1-bit units. After reset, the port AL pins operate as port pin. Each port bit can be independently configured to port input, port output or peripheral function.

This register can be read in 1-bit, 8-bit and 16-bit units.

Figure 15-27: Port AL (PAL)

	15	14	13	12	11	10	9	8	Address	At Reset
PALH	PAL15	PAL14	PAL13	PAL12	PAL11	PAL10	PAL9	PAL8	FFFFF001H	00H[Note]

	7	6	5	4	3	2	1	0	Address	At Reset
PALL	PAL7	PAL6	PAL5	PAL4	PAL3	PAL2	PAL1	PAL0	FFFFF000H	00H[Note]

Note: The reset value of register PAL is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

Remark: If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

Bit Position	Bit Name	Function
15 to 0	PALn (n = 15 to 0)	Input/output port

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

In Output Mode: When the AL register is read, the values of PAL are read. Writing to the PAL register writes the values to that register and those values are immediately output.

Besides functioning as a port, in control mode, it also can operate as an address bus.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port AL	PAL15 to PAL0	A15 to A0	Address bus	

(2) Setting in input/output mode and control mode

Port AL is set in input/output mode using the port AL mode register (PMAL). In control mode, it is set using the port AL mode control register (PMCAL).

(a) Port AL mode register (PMAL)

This register can be read or written in 16- or 8-bit or 1-bit units.

Figure 15-28: Port AL Mode Register (PMAL)

	15	14	13	12	11	10	9	8	Address	At Reset
PMALH	PMAL15	PMAL14	PMAL13	PMAL12	PMAL11	PMAL10	PMAL9	PMAL8	FFFF021FH	00H

	7	6	5	4	3	2	1	0	Address	At Reset
PMALL	PMAL7	PMAL6	PMAL5	PMAL4	PMAL3	PMAL2	PMAL1	PMAL0	FFFF020FH	00H

Bit Position	Bit Name	Function
15 to 0	PMALn (n = 15 to 0)	Specifies input/output mode of PALn pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port AL mode control register (PMCAL)

This register can be read or written in 16-bit or 8-bit or 1-bit units.

Figure 15-29: Port AL Mode Control Register (PMCAL)

	15	14	13	12	11	10	9	8	Address	At Reset
PMCALH	PMCAL15	PMCAL14	PMCAL13	PMCAL12	PMCAL11	PMCAL10	PMCAL9	PMCAL8	FFFFF041H	00H

	7	6	5	4	3	2	1	0	Address	At Reset
PMCALL	PMCAL7	PMCAL6	PMCAL5	PMCAL4	PMCAL3	PMCAL2	PMCAL1	PMCAL0	FFFFF040H	00H

Bit Position	Bit Name	Function
15to 0	PMCALn (n = 15 to 0)	Specifies operation mode of PMCALn pin 0: Input/output port mode 1: A15 to A0 address output

15.3.9 Port CS

Port CS is a 3-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function[Note].

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

Figure 15-30: Port CS (PCS)

	7	6	5	4	3	2	1	0	Address	At Reset
PCS	0	0	0	PCS4	PCS3	0	0	PCS0	FFFFF008H	00H

Note: If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

Remark: The reset value of register PCS is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

Bit Position	Bit Name	Function
4, 3, 0	PCSn (n = 4, 3, 0)	Input/output port

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

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

Besides functioning as a port, in control mode, it also can operate as the chip select signal output when memory is accesses externally.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port CS	PCS4	$\overline{CS0}$	Chip select signal output	
	PCS3	$\overline{CS3}$	Chip select signal output	
	PCS0	$\overline{CS4}$	Chip select signal output	

Caution: In case that a port pin $\overline{CS0}$, $\overline{CS3}$ or $\overline{CS4}$ operates as a chip select output port, it is recommended to plug in an external pull up resistor to that pin.

(2) Setting in input/output mode and control mode

Port CS is set in input/output mode using the port CS mode register (PMCS). In control mode, it is set using the port CS mode control register (PMCS).

(a) Port CS mode register (PMCS)

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

Figure 15-31: Port CS Mode Register (PMCS)

	7	6	5	4	3	2	1	0	Address	At Reset
PMCS	0	0	0	PMCS4	PMCS3	0	0	PMCS0	FFFFF028H	19H

Bit Position	Bit Name	Function
4	PMCS4	Specifies input/output mode of PCS4 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)
3	PMCS3	Specifies input/output mode of PCS3 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)
0	PMCS0	Specifies input/output mode of PCS0 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port CS mode control register (PMCCS)

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

Figure 15-32: Port CS Mode Control Register (PMCCS)

	7	6	5	4	3	2	1	0	Address	At Reset
PMCCS	0	0	0	PMCCS4	PMCCS3	0	0	PMCCS0	FFFFF048H	00H

Bit Position	Bit Name	Function
4	PMCCS4	Specifies operation mode of PMCCS4 pin 0: Input/output port mode 1: $\overline{CS4}$ Chip select output
3	PMCCS4	Specifies operation mode of PMCCS3 pin 0: Input/output port mode 1: $\overline{CS3}$ Chip select output
0	PMCCS4	Specifies operation mode of PMCCS0 pin 0: Input/output port mode 1: $\overline{CS0}$ Chip select output

15.3.10 Port CT

Port CT is a 3-bit input/output port in which input or output can be specified in 1-bit units. Each port bit can be independently configured to port input, port output or peripheral function.

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

Figure 15-33: Port CT (PCT)

	7	6	5	4	3	2	1	0	Address	At Reset
PCT	0	0	0	PCT4	0	0	PCT1	PCT0	FFFFF00AH	00H

Bit Position	Bit Name	Function
4, 1, 0	PCTn (n = 4, 1, 0)	Input/output port

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

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

Besides functioning as a port, in control mode, PCT0 and PCT1 can operate as the write strobe signal outputs when memory is accessed externally. PCT4 can also operate as the read strobe signal input.

Remarks: **1.** If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

2. The reset value of register PCT is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port CT	PCT4	$\overline{RD}$	Read strobe output	
	PCT1	$\overline{WR1}$	Upper write strobe signal output	
	PCT0	$\overline{WR0}$	Lower write strobe signal output	

Caution: In case that a port pin PCT0, PCT1 or PCT4 operates as a control signal for the external memory interface ($\overline{WR0}$, $\overline{WR1}$ or $\overline{RD}$), it is recommended to plug in an external pull up resistor to that pin

(2) Setting in input/output mode and control mode

Port CT is set in input/output mode using the port CT mode register (PMCT). In control mode, it is set using the port CT mode control register (PMCT).

(a) Port CT mode register (PMCT)

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

Figure 15-34: Port CT Mode Register (PMCT)

	7	6	5	4	3	2	1	0	Address	At Reset
PMCT	0	0	0	PMCT4	0	0	PMCT1	PMCT0	FFFFF02AH	13H

Bit Position	Bit Name	Function
4	PMCT4	Specifies input/output mode of PCT4 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)
1	PMCT1	Specifies input/output mode of PCT1 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)
0	PMCT0	Specifies input/output mode of PCT0 pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port mode control register (PMCCT)

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

Figure 15-35: Port CT Mode Control Register (PMCCT)

	7	6	5	4	3	2	1	0	Address	At Reset
PMCCT	0	0	0	PMCCT4	0	0	PMCCT1	PMCCT0	FFFFF04AH	00H

Bit Position	Bit Name	Function
4	PMCCT4	Specifies operation mode of PMCCT4 pin 0: Input/output port mode 1: $\overline{RD}$ Read strobe signal output
1	PMCCT1	Specifies operation mode of PMCCT1 pin 0: Input/output port mode 1: $\overline{WR1}$ Upper write strobe signal output
0	PMCCT0	Specifies operation mode of PMCCT0 pin 0: Input/output port mode 1: $\overline{WR0}$ Lower write strobe signal output

15.3.11 Port DL

Port DL is a 16-bit input/output port in which input or output can be specified in 1-bit units. After reset, the port DL pins operate as port pin. Each port bit can be independently configured to port input, port output or peripheral function.

This register can be read in 1-bit, 8-bit and 16-bit units.

Figure 15-36: Port DL (PDL)

	15	14	13	12	11	10	9	8	Address	At Reset
PDLH	PDL15	PDL14	PDL13	PDL12	PDL11	PDL10	PDL9	PDL8	FFFFF005H	00H[Note]

	7	6	5	4	3	2	1	0	Address	At Reset
PDLL	PDL7	PDL6	PDL5	PDL4	PDL3	PDL2	PDL1	PDL0	FFFFF004H	00H[Note]

Note: The reset value of register PDL is 00H. Due to the input mode of the port after reset, the read input value is determined by the port pins.

Remark: If using peripheral functions, the direction setting for the respective port bit is not forced automatically. Port bit direction has to be programmed according to the peripheral function requirement by setting the port mode register.

Bit Position	Bit Name	Function
15 to 0	PDLn (n = 15 to 0)	Input/output port

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

In Output Mode: When the DL register is read, the values of PDL are read. Writing to the PDL register writes the values to that register and those values are immediately output.

Besides functioning as a port, in control mode, it also can operate as an address bus.

(1) Operation in control mode

Port		Alternate Pin Name	Remarks	Block Type
Port DL	PDL15 to PDL0	D15 to D0	Data bus	

(2) Setting in input/output mode and control mode

Port DL is set in input/output mode using the port DL mode register (PMDL). In control mode, it is set using the port DL mode control register (PMCDL).

(a) Port DL mode register (PMDL)

This register can be read or written in 16-bit, 8-bit or 1-bit units.

Figure 15-37: Port DL Mode Register (PMDL)

	15	14	13	12	11	10	9	8	Address	At Reset
PMDLH	PMDL15	PMDL14	PMDL13	PMDL12	PMDL11	PMDL10	PMDL9	PMDL8	FFFF005FH	00H

	7	6	5	4	3	2	1	0	Address	At Reset
PMDLL	PMDL7	PMDL6	PMDL5	PMDL4	PMDL3	PMDL2	PMDL1	PMDL0	FFFF004FH	00H

Bit Position	Bit Name	Function
15 to 0	PMDLn (n = 15 to 0)	Specifies input/output mode of PDLn pin. 0: Output mode (Output buffer on) 1: Input mode (Output buffer off)

(b) Port DL mode control register (PMCDL)

This register can be read or written in 16-bit, 8-bit or 1-bit units.

Figure 15-38: Port DL Mode Control Register (PMCDL)

	15	14	13	12	11	10	9	8	Address	At Reset
PMCDLH	PMCDL15	PMCDL14	PMCDL13	PMCDL12	PMCDL11	PMCDL10	PMCDL9	PMCDL8	FFFFF045H	00H

	7	6	5	4	3	2	1	0	Address	At Reset
PMCDLL	PMCDL7	PMCDL6	PMCDL5	PMCDL4	PMCDL3	PMCDL2	PMCDL1	PMCDL0	FFFFF044H	00H

Bit Position	Bit Name	Function
15to 0	PMCDLn (n = 15 to 0)	Specifies operation mode of PMCDLn pin 0: Input/output port mode 1: D15 to D0 address output

Chapter 16 Flash Memory

The V850E/CG3 CarGate+ is equipped with embedded FLASH memory

When fetching an instruction, 4 bytes of the flash memory can be accessed in 1 clock in the same manner as the mask ROM versions.

The flash memory can be written mounted on the target board (on-board write), by connecting a dedicated flash programmer to the target system.

Flash memory is commonly used in the following development environments and applications.

- For altering software after solder-mounting the V850E/CG3 on the target system
- For differentiating software in small-scale production of various models.
- For data adjustment when starting mass production

16.1 Features

- 4-byte/1-clock access (for instruction fetch access)
- All-area erase or block erase
- Communication with dedicated flash programmer via serial interface
- Single power supply Flash. No additional external erase/write voltage needed
- On-board programming
- Flash memory programming by self-programming **Notes1, 2**

Notes: 1. To use self-programming it has to be ensured, that MODE0 is set to high. For single chip operation MODE0 has to be low, therefore MODE0 has to be switchable, e.g. by using a port.

2. For details please refer to the Application Note “Self Programming” U15352EE2V0AN00 or higher.

16.2 Erase Unit

The units in which the 256 KB flash memory can be erased in following.

(1) All-area erase

The area xx000000H to xx03FFFFH can be erased at the same time.

(2) Block erase

The flash memory can be erased in block units.

Block 0:	32 KB
Block 1:	32 KB
Block 2:	96 KB
Block 3:	96 KB

16.3 Writing with Flash Programmer

A dedicated flash programmer can be used for on-board or off-board writing of the flash memory.

(1) On-board programming

The contents of the flash memory can be rewritten with the V850E/CG3 mounted on the target system. Mount a connector that connects the dedicated flash programmer on the target system.

(2) Off-board programming

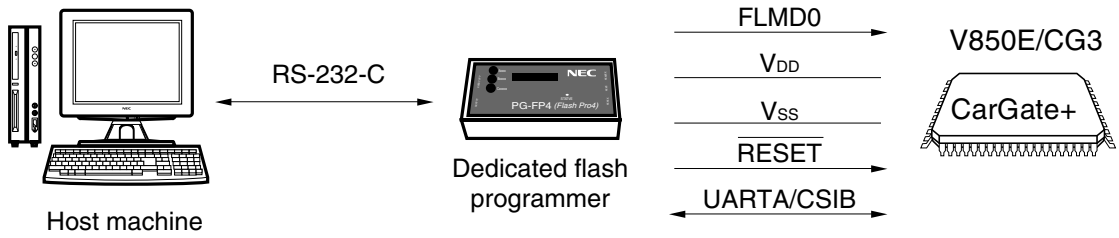
The flash memory of the V850E/CG3 can be written before the device is mounted on the target system, by using a dedicated program adapter (FA series).

Remark: The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

16.4 Programming Environment

The environment necessary to write a program to the flash memory of the V850E/CG3 is shown below.

Figure 16-1: Environment to Write Program to Flash Memory



A host machine is required for controlling the dedicated flash programmer.
UARTA0 or CSIB0 is used as the interface between the dedicated flash programmer and the V850E/CG3 to manipulate the flash programmer by writing or erasing. To write the flash memory off-board, a dedicated program adapter (FA series) is necessary.

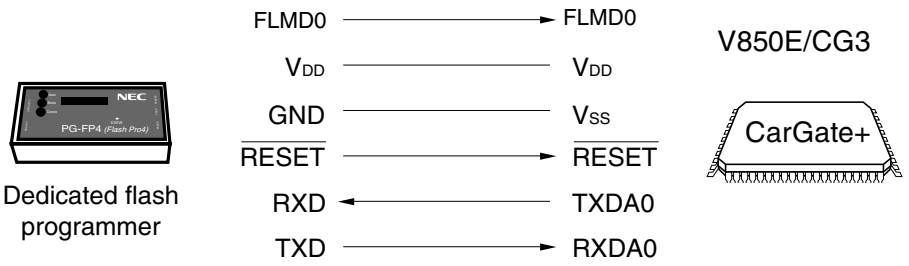
16.5 Communication Mode

Serial communication is performed between the dedicated flash programmer and the V850E/CG3 by using UARTA0 or CSIB0 of the V850E/CG3.

(1) UARTA0

Transfer rate: 4,800 to 76,800 bps

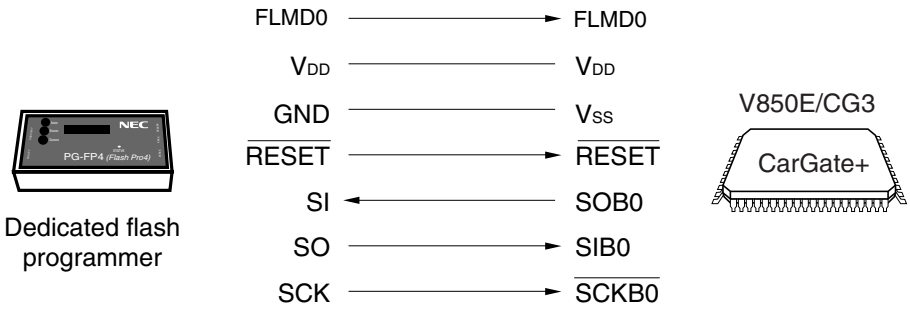
Figure 16-2: Communication with Dedicated Flash Programmer (UARTA0)



(2) CSIB0

Serial clock: Up to 1 MHz (MSB first)

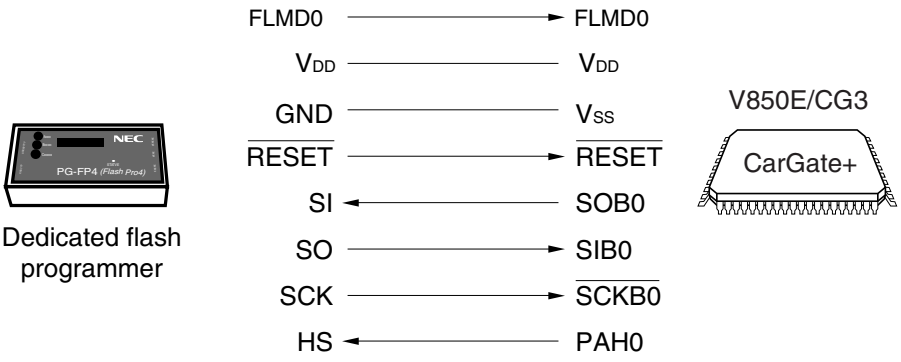
Figure 16-3: Communication with Dedicated Flash Programmer (CSIB0)



(3) CSIB0 + HS

Serial clock: Up to 1 MHz (MSB first)

Figure 16-4: Communication with Dedicated Flash Programmer (CSIB0+HS)



The dedicated flash programmer outputs a transfer clock and the V850E/CG3 operates as a salve.
If the PG-FP4 is used as the flash programmer, it generates the following signals for the V850E/CG3.
For details, refer to the **PG-FP4 User's Manual (U15260E)**.

Table 16-1: Signals Generated by Dedicated Flash Programmer (PG-FP4)

PG-FP4			V850E/CG3	Connection		
Signal Name	I/O	Pin Function	Pin Name	CSIB0	UARTA0	CSIB0 + HS
MODE0	Output	Write enable/disable	MODE0	O	O	O
V_{DD}	I/O	V_{DD} voltage generation/voltage monitor	V_{DD}	O	O	O
GND	–	Ground	V_{SS}	O	O	O
CLK	Output	Clock output to V850E/CG3	X1	×	×	×
$\overline{RESET}$	Output	Reset signal	$\overline{RESET}$	O	O	O
SI/RXD	Input	Receive signal	SOB0/TXDA0	O	O	O
SO/TXD	Output	Transmit signal	SIB0/RXDA0	O	O	O
$\overline{SCK}$	Output	Transfer clock	$\overline{SCKB0}$	O	×	O
HS	Input	Handshake signal for CSIB0 + HS communication	PAH0	×	×	O

Remark: O: Always connected.
×: Does not need to be connected.

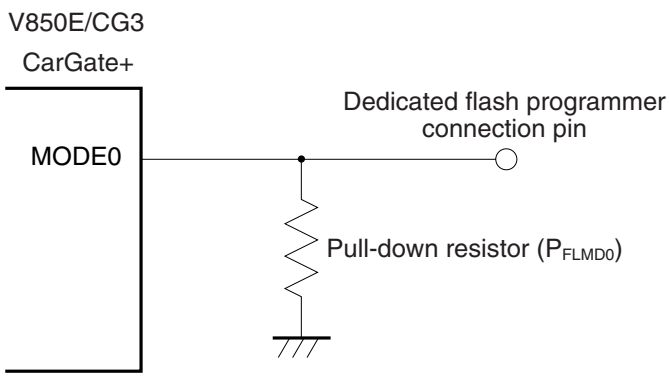
16.6 Pin Connection

A connector must be mounted on the target system to connect the dedicated flash programmer for on-board writing. In addition, a function to switch between the normal operation mode and flash memory programming mode must be provided on the board.
When the flash memory programming mode is set, all the pins not used for flash memory programming are in the same status as that immediately after reset. Therefore, all the ports go into an output high-impedance state, and the pins must be processed correctly if the external device does not recognize the output high-impedance state.

16.6.1 MODE0 pin

In the normal operation mode, 0 V is input to the MODE0 pin. In the flash memory programming mode, the V_{DD} write voltage is supplied to the MODE0 pin. An example of connection of the MODE0 pin is shown below.

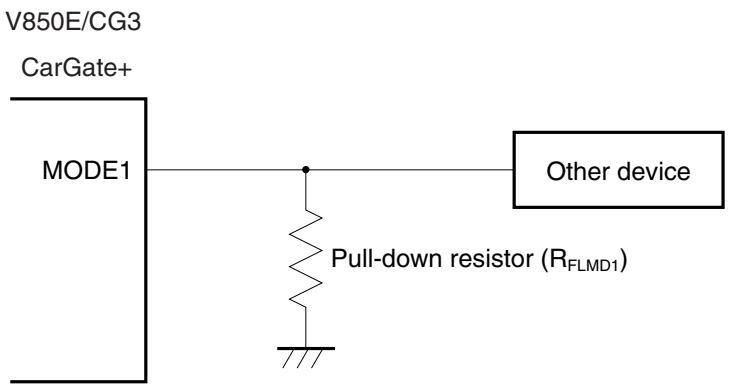
Figure 16-5: Example of Connection of MODE0 Pin



16.6.2 MODE1 pin

If 0 V is input to the MODE0 pin, the MODE1 pin does not function. If V_{DD} is supplied to the MODE0 pin, 0 V must be input to the MODE1 pin to set the flash memory programming mode. An example of the connection of the MODE1 pin is shown below.

Figure 16-6: Example of Connection of MODE1 Pin



Caution: If a V_{DD} signal is input to the MODE1 pin from an other device during on-board writing and immediately after reset, isolate this signal.

Table 16-2: Relationship Between MODE0 and MODE1 Pins and Operation Mode

MODE0	MODE1	Operation Mode
0 V	×	Normal operation mode
V_{DD}	0 V	Flash memory programming mode
V_{DD}	V_{DD}	Setting prohibited

Remark: ×: Don't care

16.6.3 Serial interface pins

The pins used by each serial interface are shown in the table below.

Table 16-3: Pins Used by Each Serial Interface

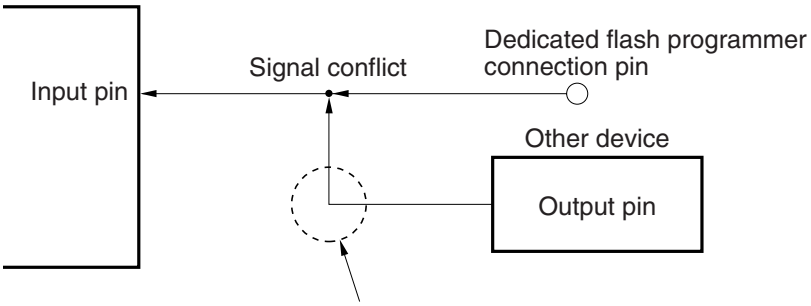
Serial Interface	Pins
CSIB0	SOB0, SIB0, SCKB0
CSIB0 + HS	SOB0, SIB0, SCKB0, PAH0
UARTA0	TXDA0, RXDA0

When connecting a dedicated flash programmer to a serial interface pin that is connected to another device on board, exercise care so that signal conflict and malfunction of the other device do not occur.

(1) Conflict of signals

When the dedicated flash programmer (output) is connected to a serial interface pin (input) connected to another device (output), a signal conflict occurs. To avoid this signal conflict, isolate the connection with the other device, or place the other device in an output high-impedance state.

Figure 16-7: Signal Conflict (Input Pin of Serial Interface)

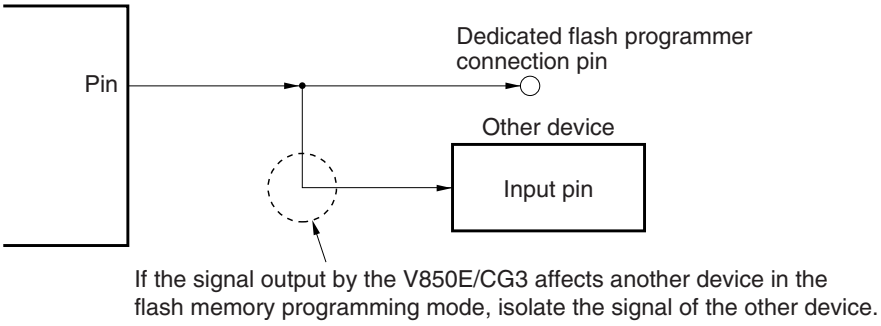


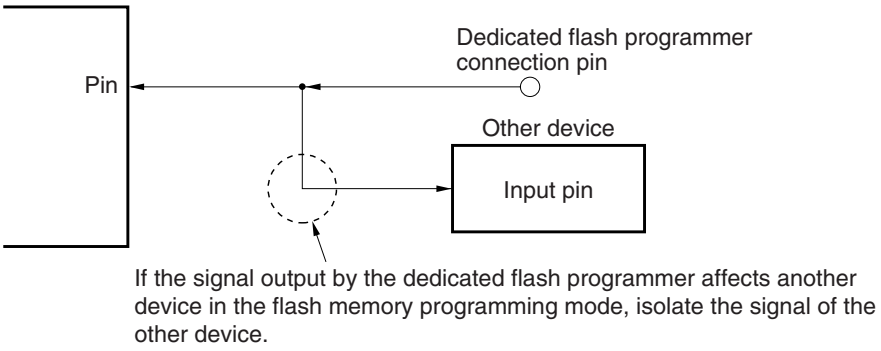
In the flash memory programming mode, the signal output by an other device conflicts with the signal output by the dedicated flash programmer. Isolate the signal of the other device.

(2) Abnormal operation of other device

When the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) connected to another device (input), a signal is output to the other device, causing a malfunction. To avoid this malfunction, isolate the connection with the other device, or set so that the signal input to the other device is ignored.

Figure 16-8: Abnormal Operation of Other Device

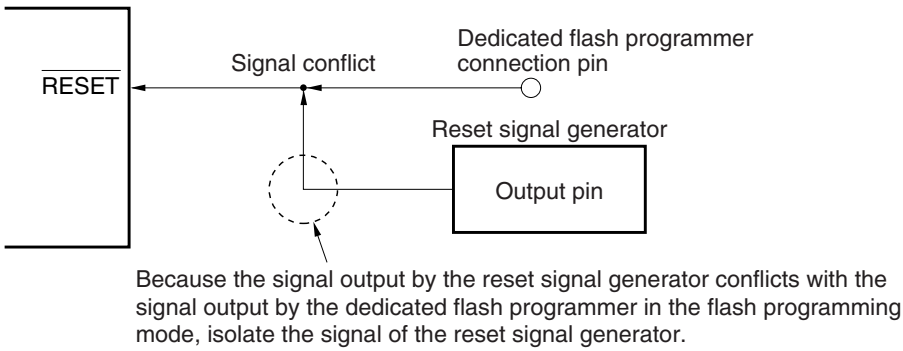




16.6.4 $\overline{\text{RESET}}$ pin

When the reset signal of the dedicated flash programmer is connected to the $\overline{\text{RESET}}$ pin connected to a reset signal generator on board, a signal conflict occurs. To avoid this signal conflict, isolate the connection with the reset signal generator.
If a reset signal is input from the user system in flash memory programming mode, the programming operation is not performed correctly. Do not input a reset signal other than that from the dedicated flash programmer.

Figure 16-9: Signal Conflict ($\overline{\text{RESET}}$ Pin)



16.6.5 Port pins (including NMI)

All the port pins, including the pin connected to the dedicated flash programmer, go into an output high-impedance state in the flash memory programming mode. If there is a problem such as that the external device connected to a port prohibits the output high-impedance state, connect the port to V_{DD} or V_{SS} via a resistor.

16.6.6 Other signal pins

Connect the X1 and X2 pins in the same status as in the normal operation mode.

16.6.7 Power supply

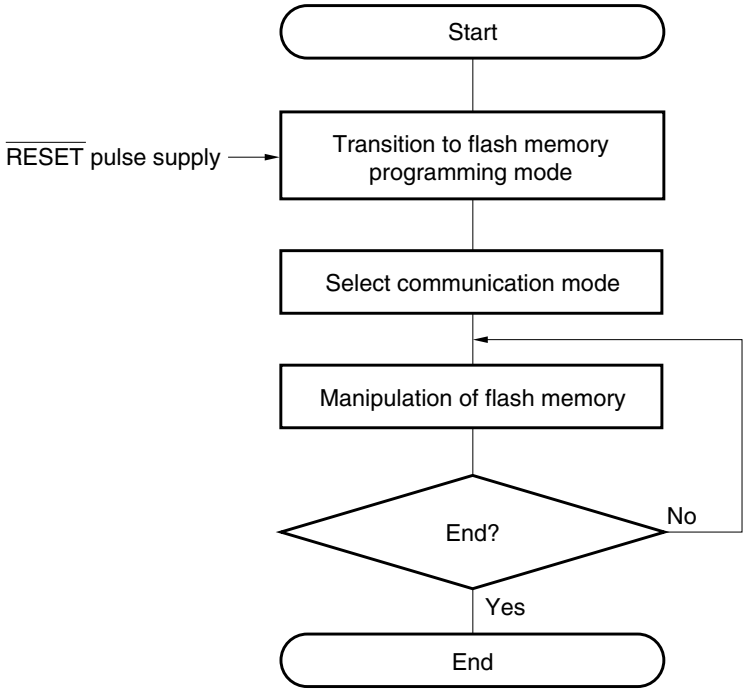
Supply the same power to the power supply pins (V_{DD}, V_{SS}, AV_{SS}, BV_{DD}, BV_{SS}, AV_{DD}) as in the normal operation mode.

16.7 Programming Method

16.7.1 Flash memory control

The procedure to manipulate the flash memory is illustrated below.

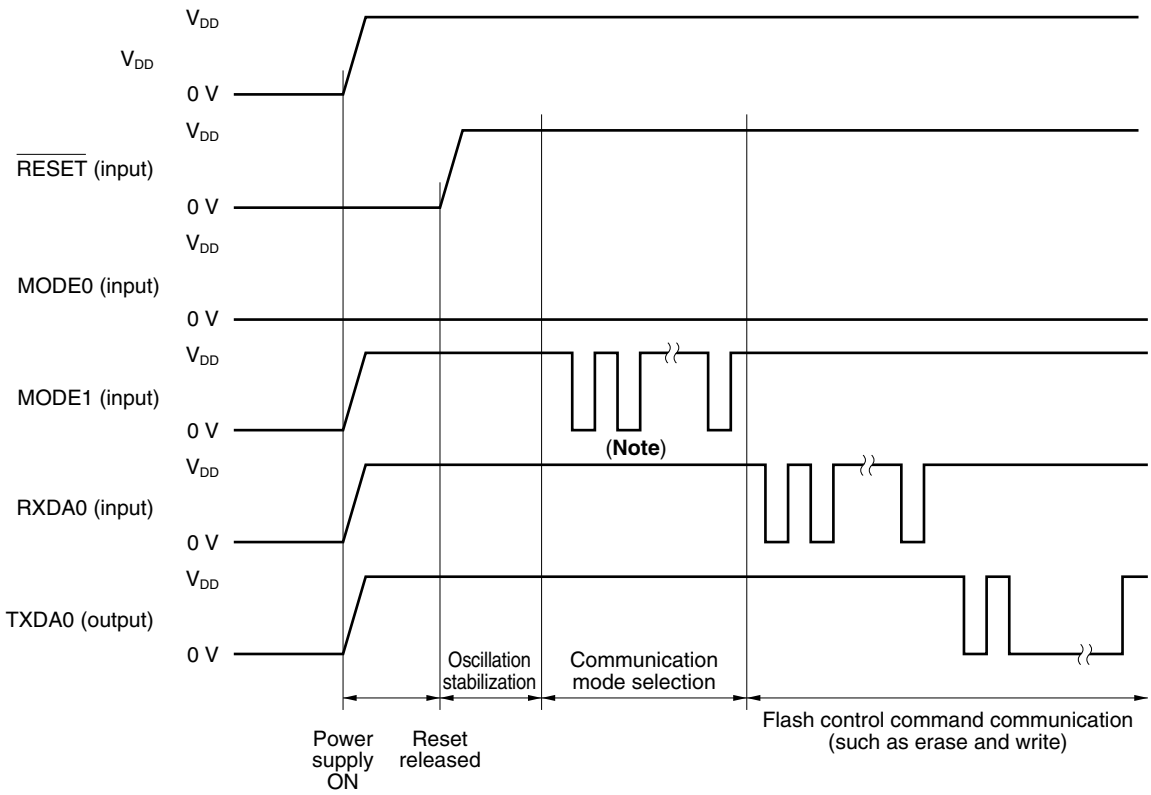
Figure 16-10: Flash Memory Manipulation Procedure



16.7.2 Flash memory programming mode

To rewrite the contents of the flash memory by using the dedicated flash programmer, set the V850E/CG3 in the flash memory programming mode.
To set the mode, set the MODE0 and MODE1 pins, and release reset.
Change the mode by using a jumper when performing on-board writing.

Figure 16-11: Flash Memory Programming Mode



Note: The number of clocks to be inserted differs depending on the communication mode. For details, refer to Table 16-4, “Communication Modes,” on page 797.

16.7.3 Selecting communication mode

In the V850E/CG3 the communication mode is selected by inputting pulses (up to 12 pulses) to the MODE0 pin after the flash memory programming mode is set. These MODE pulses are generated by the dedicated flash programmer.
The relationship between the number of pulses and the communication mode is shown in the table below.

Table 16-4: Communication Modes

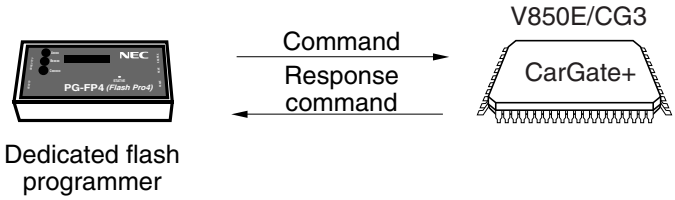
MODE Pulse	Communication Mode	Remark
8	CSIB0	V850E/CG3 operate as slave. MSB first
11	CSIB0 + HS	V850E/CG3 operate as slave. MSB first
0	UARTA0	Communication rate: 9,600 bps (after reset), LSB first
Others	RFU	Setting prohibited

Caution: When UARTA is selected, the receive clock is calculated based on the reset command that is sent from the dedicated flash programmer after reception of the MODE pulse.

16.7.4 Communication commands

The V850E/CG3 communicate with the dedicated flash programmer via commands. The commands sent by the dedicated flash programmer to the V850E/CG3 are called commands, and the response signals sent by the V850E/CG3 to the flash programmer are called response commands.

Figure 16-12: Communication Commands



The following table lists the flash memory control commands of the V850E/CG3. All these commands are issued by the programmer, and the V850E/CG3 perform the corresponding processing.

Table 16-5: Flash Memory Control Commands

Classification	Command Name	Support			Function
		CSIB	CSIB+HS	UARTA	
Blank check	Block blank check command	×	×	×	Checks erasure status of entire memory.
Erase	Chip erase command	×	×	×	Erases all memory contents.
	Block erase command	×	×	×	Erases memory contents of specified block.
Write	Write command	×	×	×	Writes data by specifying write address and number of bytes to be written, and executes verify check.
Verify	Verify command	×	×	×	Compares input data with all memory contents.
System setting and control	Reset command	×	×	×	Escapes from each status.
	Oscillation frequency setting command	×	×	×	Sets oscillation frequency.
	Baud rate setting command	–	–	×	Sets baud rate when UART is used.
	Silicon signature command	×	×	×	Reads silicon signature information.
	Version acquisition command	×	×	×	Reads version information of device.
	Status command	×	×	–	Acquires operation status.
	Security setting command	×	×	×	Sets security of chip erasure, block erasure, and writing.

The V850E/CG3 return a response command in response to the command issued by the flash programmer. The response commands sent by the V850E/CG3 are listed below.

Table 16-6: Response Commands

Response Command Name	Function
ACK	Acknowledges command/data.
NAK	Acknowledges illegal command/data.

[MEMO]

Appendix A Instruction Set List

A.1 Convention

(a) Register symbols used to describe operands

Register Symbol	Explanation
reg1	General registers: Used as source registers
reg2	General registers: Used mainly as destination registers. Also used as source register in some instructions.
reg3	General registers: Used mainly to store the remainders of division results and the higher order 3 bits of multiplication results.
bit#3	33-bit data for specifying the bit number
immX	X bit immediate data
dispX	X bit displacement data
regID	System register number
vector	5-bit data that specifies the trap vector (00H to 1FH)
cccc	4-bit data that shows the conditions code
sp	Stack pointer (SP)
ep	Element pointer (r30)
listX	X item register list

(b) Register symbols used to describe opcodes

Register Symbol	Explanation
R	1-bit data of a code that specifies reg1 or regID
r	1-bit data of the code that specifies reg2
w	1-bit data of the code that specifies reg3
d	1-bit displacement data
I	1-bit immediate data (indicates the higher bits of immediate data)
i	1-bit immediate data
cccc	4-bit data that shows the condition codes
CCCC	4-bit data that shows the condition codes of Bcond instruction
bbb	3-bit data for specifying the bit number
L	1-bit data that specifies a program register in the register list
S	1-bit data that specifies a system register in the register list

(c) Register symbols used in operation

Register Symbol	Explanation
←	Input for
GR [ ]	General register
SR [ ]	System register
zero-extend (n)	Expand n with zeros until word length.
sign-extend (n)	Expand n with signs until word length.
load-memory (a, b)	Read size b data from address a.
store-memory (a, b, c)	Write data b into address a in size c.
load-memory-bit (a, b)	Read bit b of address a.
store-memory-bit (a, b, c)	Write c to bit b of address a.
saturated (n)	Execute saturated processing of n (n is a 2's complement). If, as a result of calculations, n ≥ 7FFFFFFFH, let it be 7FFFFFFFH. n ≤ 80000000H, let it be 80000000H.
result	Reflects the results in a flag.
Byte	Byte (8 bits)
Half-word	Half word (16 bits)
Word	Word (32 bits)
+	Addition
–	Subtraction
ll	Bit concatenation
×	Multiplication
÷	Division
%	Remainder from division results
AND	Logical product
OR	Logical sum
XOR	Exclusive OR
NOT	Logical negation
logically shift left by	Logical shift left
logically shift right by	Logical shift right
arithmetically shift right by	Arithmetic shift right

(d) Register symbols used in an execution clock

Register Symbol	Explanation
I	If executing another instruction immediately after executing the first instruction (issue).
r	If repeating execution of the same instruction immediately after executing the first instruction (repeat).
l	If using the results of instruction execution in the instruction immediately after the execution (latency).

(e) Register symbols used in flag operations

Identifier	Explanation
(Blank)	No change
0	Clear to 0
X	Set or cleared in accordance with the results.
R	Previously saved values are restored.

(f) Condition codes

Condition Name (cond)	Condition Code (cccc)	Condition Formula	Explanation
V	0 0 0 0	OV = 1	Overflow
NV	1 0 0 0	OV = 0	No overflow
C/L	0 0 0 1	CY = 1	Carry Lower (Less than)
NC/NL	1 0 0 1	CY = 0	No carry Not lower (Greater than or equal)
Z/E	0 0 1 0	Z = 1	Zero Equal
NZ/NE	1 0 1 0	Z = 0	Not zero Not equal
NH	0 0 1 1	(CY or Z) = 1	Not higher (Less than or equal)
H	1 0 1 1	(CY or Z) = 0	Higher (Greater than)
N	0 1 0 0	S = 1	Negative
P	1 1 0 0	S = 0	Positive
T	0 1 0 1	—	Always (Unconditional)
SA	1 1 0 1	SAT = 1	Saturated
LT	0 1 1 0	(S xor OV) = 1	Less than signed
GE	1 1 1 0	(S xor OV) = 0	Greater than or equal signed
LE	0 1 1 1	((S xor OV) or Z) = 1	Less than or equal signed
GT	1 1 1 1	((S xor OV) or Z) = 0	Greater than signed

A.2 Instruction Set (In Alphabetical Order)

(1/4)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
ADD	reg1,reg2	rrrrr001110RRRRR	GR[reg2]←GR[reg2] + GR[reg1]	1	1	1	×	×	×	×	
	imm5,reg2	rrrrr010010iiiii	GR[reg2]←GR[reg2] + sign-extend(imm5)	1	1	1	×	×	×	×	
ADDI	imm16,reg1,reg2	rrrrr110000RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] + sign-extend(imm16)	1	1	1	×	×	×	×	
AND	reg1,reg2	rrrrr001010RRRRR	GR[reg2]←GR[reg2] AND GR[reg1]	1	1	1		0	×	×	
ANDI	imm16,reg1,reg2	rrrrr110110RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] AND zero-extend(imm16)	1	1	1		0	0	×	×
Bcond	disp9	ddddd1011dddcccc Note 1	if conditions are satisfied then PC←PC+sign-extend(disp9) — When conditions are satisfied	2 Note 2	2 Note 2	2 Note 2					
			When conditions are not satisfied	1	1	1					
BSH	reg2,reg3	rrrrr11111100000 wwwww01101000010	GR[reg3]←GR[reg2] (23: 16) ll GR[reg2] (31: 24) ll GR[reg2] (7: 0) ll GR[reg2] (15: 8)	1	1	1	×	0	×	×	
BSW	reg2,reg3	rrrrr11111100000 wwwww01101000000	GR[reg3]←GR[reg2] (7: 0) ll GR[reg2] (15: 8) ll GR[reg2] (23: 16) ll GR[reg2] (31: 24)	1	1	1	×	0	×	×	
CALLT	imm6	0000001000iiiiii	CTPC←PC + 2(return PC) CTPSW←PSW adr←CTBP+zero-extend(imm6 logically shift left by 1) PC←CTBP+zero-extend(Load-memory(adr,Half-word))	4	4	4					
CLR1	bit#3, disp16[reg1]	10bbb111110RRRRR ddddddddddddddd	adr←GR[reg1] + sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,0)	3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100100	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,0)	3 Note 3	3 Note 3	3 Note 3				×	
CMOV	cccc,imm5,reg2, reg3	rrrrr111111iiiii wwwww011000cccc0	if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2]	1	1	1					
	cccc,reg1,reg2, reg3	rrrrr111111RRRRR wwwww011001cccc0	if conditions are satisfied then GR[reg3]←GR[reg1] else GR[reg3]←GR[reg2]	1	1	1					
CMP	reg1,reg2	rrrrr001111RRRRR	result←GR[reg2] – GR[reg1]	1	1	1	×	×	×	×	
	imm5,reg2	rrrrr010011iiiii	result←GR[reg2] – sign-extend(imm5)	1	1	1	×	×	×	×	
CTRET		0000011111100000 0000000101000100	PC←CTPC PSW←CTPSW	3	3	3	R	R	R	R	R
DI		0000011111100000 0000000101100000	PSW.ID←1	1	1	1					
DISPOSE	imm5,list12	00000011001iiiiiL LLLLLLLLLLL00000	sp←sp + zero-extend(imm5 logically shift left by 2) GR[reg in list12]←Load-memory(sp,Word) sp←sp + 4 repeat 2 steps above until all regs in list12 are loaded	N+1 Note 4	N+1 Note 4	N+1 Note 4					
	imm5,list12,[reg1]	00000011001iiiiiL LLLLLLLLLLLRRRRR Note 5	sp←sp + zero-extend(imm5 logically shift left by 2) R[reg in list12]←Load-memory(sp,Word) sp←sp + 4 repeat 2 steps above until all regs in list12 are loaded PC←GR[reg1]	N+3 Note 4	N+3 Note 4	N+3 Note 4					
DIV	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01011000000	GR[reg2]←GR[reg2] ÷ GR[reg1] GR[reg3]←GR[reg2] % GR[reg1]	35	35	35					
DIVH	reg1,reg2	rrrrr000010RRRRR	GR[reg2]←GR[reg2] ÷ GR[reg1]Note 6	35	35	35		×	×	×	
	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01010000000	GR[reg2]←GR[reg2] ÷ GR[reg1]Note 6 GR[reg3]←GR[reg2] % GR[reg1]	35	35	35		×	×	×	
DIVHU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01010000010	GR[reg2]←GR[reg2] ÷ GR[reg1]Note 6 GR[reg3]←GR[reg2] % GR[reg1]	34	34	34		×	×	×	
DIVU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01011000010	GR[reg2]←GR[reg2] ÷ GR[reg1] GR[reg3]←GR[reg2] % GR[reg1]	34	34	34		×	×	×	
EI		1000011111100000 0000000101100000	PSW.ID←0	1	1	1					
HALT		0000011111100000 0000000100100000	Stop	1	1	1					
HSW	reg2,reg3	rrrrr11111100000 wwwww01101000100	GR[reg3]←GR[reg2](15: 0) ll GR[reg2] (31: 16)	1	1	1	×	0	×	×	
JARL	disp22,reg2	rrrrr11110ddddddd ddddddddddddddd0 Note 7	GR[reg2]←PC + 4 PC←PC + sign-extend(disp22)	2	2	2					
JMP	[reg1]	00000000011RRRRR	PC←GR[reg1]	3	3	3					

(2/4)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
JR	disp22	0000011110dddddd dddddddddddddddd0 Note 7	PC←PC + sign-extend(disp22)	2	2	2					
LD.B	disp16[reg1],reg2	rrrrr111000RRRRR ddddddddddddddddd	adr←GR[reg1] + sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 11					
LD.BU	disp16[reg1],reg2	rrrrr11110bRRRRR ddddddddddddddd1 Notes 8, 10	adr←GR[reg1] + sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	Note 11					
LD.H	disp16[reg1],reg2	rrrrr111001RRRRR dddddddddddddddd0 Note 8	adr←GR[reg1] + sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Half-word))	1	1	Note 11					
LDSR	reg2,regID	rrrrr111111RRRRR 0000000000100000 Note 12	SR[regID]←GR[reg2] — Other than regID = PSW	1	1	1					
			SR[regID]←GR[reg2] — regID = PSW	1	1	1	×	×	×	×	×
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1]+sign-exend(disp16) GR[reg2]←zero-extend(Load-memory(adr,half-word)	1	1	Note 11					
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1] + sign-exend(disp16) GR[reg2]←Load-memory(adr,Word)	1	1	Note 9					
MOV	reg1,reg2	rrrrr000000RRRRR	GR[reg2]←GR[reg1]	1	1	1					
	imm5,reg2	rrrrr010000iiiii	GR[reg2]←sign-extend(imm5)	1	1	1					
	imm32,reg1	00000110001RRRRR iiiiiiiiiiiiiiii iiiiiiiiiiiiiiii	GR[reg1]←imm32	2	2	2					
MOVEA	imm16,reg1,reg2	rrrrr110001RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] + sign-extend(imm16)	1	1	1					
MOVHI	imm16,reg1,reg2	rrrrr110010RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] + (imm16 ll 0^{16})	1	1	1					
MUL	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100000	GR[reg3] ll GR[reg2]←GR[reg2] × GR[reg1]	1	2 Note 14	2					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII00 **Note 13**	GR[reg3] ll GR[reg2]←GR[reg2] × sign-extend(imm9)	1	2 Note 14	2					
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2]**Note 6** × GR[reg1]**Note 6**	1	1	2					
	imm5,reg2	rrrrr010111iiiii	GR[reg2]←GR[reg2]**Note 6** × sign-extend(imm5)	1	1	2					
MULHI	imm16,reg1,reg2	rrrrr110111RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]**Note 6** × imm16	1	1	2					
MULU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]	1	2 Note 14	2					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII10 Note 13	GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9)	1	2 Note 14	2					
NOP		0000000000000000	Pass at least one clock cycle doing nothing.	1	1	1					
NOT	reg1,reg2	rrrrr000001RRRRR	GR[reg2]←NOT(GR[reg1])	1	1	1		0	×	×	
NOT1	bit#3,disp16[reg1]	01bbb111110RRRRR ddddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,Z flag)	3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100010	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,Z flag)	3 Note 3	3 Note 3	3 Note 3				×	
OR	reg1,reg2	rrrrr001000RRRRR	GR[reg2]←GR[reg2]OR GR[reg1]	1	1	1		0	×	×	
ORI	imm16,reg1,reg2	rrrrr110100RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]OR zero-extend(imm16)	1	1	1		0	×	×	
PREPARE	list12,imm5	0000011110iiiiiL LLLLLLLLLLL00001	Store-memory(sp – 4,GR[reg in list12],Word) sp←sp – 4 repeat 1 step above until all regs in list12 are stored sp←sp – zero-extend(imm5)	n+1 Note 4	n+1 Note 4	n+1 Note 4					
	list12,imm5, sp/imm**Note 15**	0000011110iiiiiL LLLLLLLLLLLff011 imm16/imm32 Note 16	Store-memory(sp – 4,GR[reg in list12],Word) sp←sp – 4 repeat 1 step above until all regs in list12 are stored sp←sp – zero-extend(imm5) ep←sp/imm	n+2 Note 4 Note 17	n+2 Note 4 Note 17	n+2 Note 4 Note 17					

(3/4)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
RETI		0000011111100000 0000000101000000	if PSW.EP=1 then PC ←EIPC PSW ←EIPSW else if PSW.NP=1 then PC ←FEPC PSW ←FEPSW else PC←EIPC PSW ←EIPSW	3	3	3	R	R	R	R	R
SAR	reg1,reg2	rrrrr111111RRRRR 0000000010100000	GR[reg2]←GR[reg2] arithmetically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010101iiiii	GR[reg2]←GR[reg2] arithmetically shift right by zero-extend (imm5)	1	1	1	×	0	×	×	
SASF	cccc,reg2	rrrrr1111110cccc 0000001000000000	if conditions are satisfied then GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000001H else GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000000H	1	1	1					
SATADD	reg1,reg2	rrrrr000110RRRRR	GR[reg2]←saturated(GR[reg2]+GR[reg1])	1	1	1	×	×	×	×	×
	imm5,reg2	rrrrr010001iiiii	GR[reg2]←saturated(GR[reg2]+sign-extend(imm5)	1	1	1	×	×	×	×	×
SATSUB	reg1,reg2	rrrrr000101RRRRR	GR[reg2]←saturated(GR[reg2]–GR[reg1])	1	1	1	×	×	×	×	×
SATSUBI	imm16,reg1,reg2	rrrrr110011RRRRR iiiiiiiiiiiiiiii	GR[reg2]←saturated(GR[reg1]–sign-extend(imm16)	1	1	1	×	×	×	×	×
SATSUBR	reg1,reg2	rrrrr000100RRRRR	GR[reg2]←saturated(GR[reg1]–GR[reg2])	1	1	1	×	×	×	×	×
SETF	cccc,reg2	rrrrr1111110cccc 0000000000000000	If conditions are satisfied then GR[reg2]←00000001H else GR[reg2]←00000000H	1	1	1					
SET1	bit#3,disp16[reg1]	00bbb111110RRRRR ddddddddddddddd	adr←GR[reg1] + sign-extend(disp16) Z flag←Not (Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,1)	3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100000	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,1)	3 Note 3	3 Note 3	3 Note 3				×	
SHL	reg1,reg2	rrrrr111111RRRRR 0000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010110iiiii	GR[reg2]←GR[reg2] logically shift left by zero-extend(imm5)	1	1	1	×	0	×	×	
SHR	reg1,reg2	rrrrr111111RRRRR 0000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010100iiiii	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	×	0	×	×	
SLD.B	disp7[ep],reg2	rrrrr0110ddddddd	adr←ep + zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 9					
SLD.BU	disp4[ep],reg2	rrrrr0000110dddd Note 18	adr←ep + zero-extend(disp4) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	Note 9					
SLD.H	disp8[ep],reg2	rrrrr1000ddddddd Note 19	adr←ep + zero-extend(disp8) GR[reg2]←sign-extend(Load-memory(adr,Half-word))	1	1	Note 9					
SLD.HU	disp5[ep],reg2	rrrrr0000111dddd Notes 18, 20	adr←ep+zero-extend(disp5) GR[reg2]←zero-extend(Load-memory(adr,Half-word))	1	1	Note 9					
SLD.W	disp8[ep],reg2	rrrrr1010ddddddd0 Note 21	adr←ep + zero-extend(disp8) GR[reg2]←Load-memory(adr,Word)	1	1	Note 9					
SST.B	reg2,disp7[ep]	rrrrr0111ddddddd	adr←ep + zero-extend(disp7) Store-memory(adr,GR[reg2],Byte)	1	1	1					
SST.H	reg2,disp8[ep]	rrrrr1001ddddddd Note 19	adr←ep + zero-extend(disp8) Store-memory(adr,GR[reg2],Half-word)	1	1	1					
SST.W	reg2,disp8[ep]	rrrrr1010ddddddd1 Note 21	adr←ep + zero-extend(disp8) Store-memory(adr,GR[reg2],Word)	1	1	1					
ST.B	reg2,disp16[reg1]	rrrrr111010RRRRR ddddddddddddddddd	adr←GR[reg1] + sign-extend(disp16) Store-memory(adr,GR[reg2],Byte)	1	1	1					
ST.H	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd0 Note 8	adr←GR[reg1] + sign-extend(disp16) Store-memory (adr,GR[reg2], Half-word)	1	1	1					
ST.W	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1] + sign-extend(disp16) Store-memory (adr,GR[reg2], Word)	1	1	1					
STSR	regID,reg2	rrrrr111111RRRRR 0000000001000000	GR[reg2]←SR[regID]	1	1	1					
SUB	reg1,reg2	rrrrr001101RRRRR	GR[reg2]←GR[reg2]–GR[reg1]	1	1	1	×	×	×	×	
SUBR	reg1,reg2	rrrrr001100RRRRR	GR[reg2]←GR[reg1]–GR[reg2]	1	1	1	×	×	×	×	

(4/4)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SWITCH	reg1	00000000010RRRRR	adr←(PC+2) + (GR [reg1] logically shift left by 1) PC←(PC+2) + (sign-extend (Load-memory (adr,Half-word))) logically shift left by 1	5	5	5					
SXB	reg1	00000000101RRRRR	GR[reg1]←sign-extend (GR[reg1] (7: 0))	1	1	1					
SXH	reg1	00000000111RRRRR	GR[reg1]←sign-extend (GR[reg1] (15: 0))	1	1	1					
TRAP	vector	00000111111iiiii 0000000100000000	EIPC←PC+4 (Return PC) EIPSW←PSW ECR.EICC←Interrupt Code PSW.EP←1 PSW.ID←1 PC←00000040H (when vector is 00H to 0FH) 00000050H (when vector is 10H to 1FH)	3	3	3					
TST	reg1,reg2	rrrrr001011RRRRR	result←GR[reg2] AND GR[reg1]	1	1	1		0	×	×	
TST1	bit#3,disp16[reg1]	11bbb111110RRRRR ddddddddddddddd	adr←GR[reg1] + sign-extend(disp16) Z flag←Not (Load-memory-bit (adr,bit#3))	3 Note 3	3 Note 3	3 Note 3				×	
	reg2, [reg1]	rrrrr111111RRRRR 0000000011100110	adr←GR[reg1] Z flag←Not (Load-memory-bit (adr,reg2))	3 Note 3	3 Note 3	3 Note 3				×	
XOR	reg1,reg2	rrrrr001001RRRRR	GR[reg2]←GR[reg2] XOR GR[reg1]	1	1	1		0	×	×	
XORI	imm16,reg1,reg2	rrrrr110101RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] XOR zero-extend (imm16)	1	1	1		0	×	×	
ZXB	reg1	00000000100RRRRR	GR[reg1]←zero-extend (GR[reg1] (7: 0))	1	1	1					
ZXH	reg1	00000000110RRRRR	GR[reg1]←zero-extend (GR[reg1] (15: 0))	1	1	1					

Notes: 1. ddddddd: Higher 8 bits of disp9.

2. 3 clocks if the final instruction includes PSW write access.

3. If there is no wait state (3 + the number of read access wait states).

4. n is the total number of list X load registers. (According to the number of wait states. Also, if there are no wait states, n is the number of list X registers.)

5. RRRRR: other than 00000.

6. The lower half word data only are valid.

7. dddddddddddddddddddddd: The higher 21 bits of disp22.

8. ddddddddddddddd: The higher 15 bits of disp16.

9. According to the number of wait states (1 if there are no wait states).

10. b: bit 0 of disp16.

11. According to the number of wait states (2 if there are no wait states).

12. In this instruction, for convenience of mnemonic description, the source register is made reg2, but the reg1 field is used in the opcode. Therefore, the meaning of register specification in the mnemonic description and in the opcode differs from other instructions.

rrrrr= regID specification
RRRRR= reg2 specification

13. iiiii: Lower 5 bits of imm9.
IIII: Lower 4 bits of imm9.

14. In the case of reg2 = reg3 (the lower 32 bits of the results are not written in the register) or reg3 = r0 (the higher 32 bits of the results are not written in the register), shortened by 1 clock.

15. sp/imm: specified by bits 19 and 20 of the sub-opcode.

16. `ff = 00`: Load sp in ep.
`10`: Load sign expanded 16-bit immediate data (bits 47 to 32) in ep.
`11`: Load 32-bit immediate data (bits 63 to 32) in ep.

17. If imm = imm32, n + 3 clocks.

18. `rrrrr`: Other than 00000.

19. `ddddddd`: Higher 7 bits of disp8.

20. `dddd`: Higher 4 bits of disp5.

21. `dddddd`: Higher 6 bits of disp8.

Appendix B Transition Diagrams RXONLY Channel and DIAG Channel

Table B-1: Transition Diagram with upper 16 message buffer assigned to DIAG channel

After Transition		Before Transition							
RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG
		INIT	INIT	INIT	NORMAL	INIT	SLEEP	INIT	STOP
INIT	INIT	-		no transition		inhibited		inhibited	
INIT	NORMAL	no transition		-		no transition		inhibited	
INIT	SLEEP	inhibited		no transition		-		no transition	
INIT	STOP	inhibited		inhibited		no transition		-	
NORMAL	INIT	DIAG to RXONLY		DIAG to RXONLY		inhibited		inhibited	
NORMAL	NORMAL	DIAG to RXONLY		DIAG to RXONLY		DIAG to RXONLY		inhibited	
NORMAL	SLEEP	inhibited		DIAG to RXONLY		DIAG to RXONLY		DIAG to RXONLY	
NORMAL	STOP	inhibited		inhibited		DIAG to RXONLY		DIAG to RXONLY	
SLEEP	INIT	inhibited		inhibited		inhibited		inhibited	
SLEEP	NORMAL	inhibited		inhibited		inhibited		inhibited	
SLEEP	SLEEP	inhibited		inhibited		inhibited		inhibited	
SLEEP	STOP	inhibited		inhibited		inhibited		inhibited	
STOP	INIT	inhibited		inhibited		inhibited		inhibited	
STOP	NORMAL	inhibited		inhibited		inhibited		inhibited	
STOP	SLEEP	inhibited		inhibited		inhibited		inhibited	
STOP	STOP	inhibited		inhibited		inhibited		inhibited	

Comments:

1	- No transition possible or transition is not executed in one step
2	- Transition is needed. It is not executed right away, but after power save mode is released case 5 may become applicable.
3	- No transition possible or transition is not executed in one step
4	- Buffer assignment changes; transition flow needed.
5	- Buffer assignment changes, but transition flow can be executed as whole macro is in SLEEP or STOP mode.

Table B-2: Transition Diagram with upper 16 message buffer assigned to DIAG channel

After Transition		Before Transition							
RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG
		STOP	INIT	STOP	NORMAL	STOP	SLEEP	STOP	STOP
INIT	INIT	inhibited		inhibited		inhibited		inhibited	
INIT	NORMAL	inhibited		inhibited		inhibited		inhibited	
INIT	SLEEP	inhibited		inhibited		inhibited		inhibited	
INIT	STOP	inhibited		inhibited		inhibited		inhibited	
NORMAL	INIT	inhibited		inhibited		inhibited		inhibited	
NORMAL	NORMAL	inhibited		inhibited		inhibited		inhibited	
NORMAL	SLEEP	inhibited		inhibited		inhibited		inhibited	
NORMAL	STOP	inhibited		inhibited		inhibited		inhibited	
SLEEP	INIT	DIAG to RXONLY		DIAG to RXONLY		inhibited		inhibited	
SLEEP	NORMAL	DIAG to RXONLY		DIAG to RXONLY		DIAG to RXONLY		inhibited	
SLEEP	SLEEP	inhibited		DIAG to RXONLY		DIAG to RXONLY		DIAG to RXONLY	
SLEEP	STOP	inhibited		inhibited		DIAG to RXONLY		DIAG to RXONLY	
STOP	INIT	-		no transition		inhibited		inhibited	
STOP	NORMAL	no transition		-		no transition		inhibited	
STOP	SLEEP	inhibited		no transition		-		no transition	
STOP	STOP	inhibited		inhibited		no transition		-	

Comments:

1 - No transition possible or transition is not executed in one step

2 - Transition is needed. It is not executed right away, but after power save mode is released case 5 may become applicable.

3 - No transition possible or transition is not executed in one step

4 - Buffer assignment changes; transition flow needed.

5 - Buffer assignment changes, but transition flow can be executed as whole macro is in SLEEP or STOP mode.

Table B-3: Transition Diagram with upper 16 message buffer assigned to RXONLY channel

After Transition		Before Transition							
RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG
		NORMAL	INIT	NORMA	NORMAL	NORMA	SLEEP	NORMA	STOP
INIT	INIT	RXONLY to DIAG		RXONLY to DIAG		inhibited		inhibited	
INIT	NORMAL	RXONLY to DIAG		RXONLY to DIAG		RXONLY to DIAG		inhibited	
INIT	SLEEP	inhibited		RXONLY to DIAG		RXONLY to DIAG		RXONLY to DIAG	
INIT	STOP	inhibited		inhibited		RXONLY to DIAG		RXONLY to DIAG	
NORMAL	INIT	-		no transition		inhibited		inhibited	
NORMAL	NORMAL	no transition		-		no transition		inhibited	
NORMAL	SLEEP	inhibited		no transition		-		no transition	
NORMAL	STOP	inhibited		inhibited		no transition		-	
SLEEP	INIT	no transition		no transition		inhibited		inhibited	
SLEEP	NORMAL	no transition		no transition		no transition		inhibited	
SLEEP	SLEEP	inhibited		no transition		no transition		no transition	
SLEEP	STOP	inhibited		inhibited		no transition		no transition	
STOP	INIT	inhibited		inhibited		inhibited		inhibited	
STOP	NORMAL	inhibited		inhibited		inhibited		inhibited	
STOP	SLEEP	inhibited		inhibited		inhibited		inhibited	
STOP	STOP	inhibited		inhibited		inhibited		inhibited	

Comments:

1 - No transition possible or transition is not executed in one step

2 - Transition is needed. It is not executed right away, but after power save mode is released case 5 may become applicable.

3 - No transition possible or transition is not executed in one step

4 - Buffer assignment changes; transition flow needed.

5 - Buffer assignment changes, but transition flow can be executed as whole macro is in SLEEP or STOP mode.

Table B-4: Transition Diagram with upper 16 message buffer assigned to RXONLY channel

After Transition		Before Transition							
RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG	RXONLY	DIAG
		SLEEP	INIT	SLEEP	NORMAL	SLEEP	SLEEP	SLEEP	STOP
INIT	INIT	inhibited		inhibited		inhibited		inhibited	
INIT	NORMAL	inhibited		inhibited		inhibited		inhibited	
INIT	SLEEP	inhibited		inhibited		inhibited		inhibited	
INIT	STOP	inhibited		inhibited		inhibited		inhibited	
NORMAL	INIT	no transition		no transition		inhibited		inhibited	
NORMAL	NORMAL	no transition		no transition		no transition		inhibited	
NORMAL	SLEEP	inhibited		no transition		no transition		no transition	
NORMAL	STOP	inhibited		inhibited		no transition		no transition	
SLEEP	INIT	-		no transition		inhibited		inhibited	
SLEEP	NORMAL	no transition		-		no transition		inhibited	
SLEEP	SLEEP	inhibited		no transition		-		no transition	
SLEEP	STOP	inhibited		inhibited		no transition		-	
STOP	INIT	RXONLY to DIAG		RXONLY to DIAG		inhibited		inhibited	
STOP	NORMAL	RXONLY to DIAG		RXONLY to DIAG		RXONLY to DIAG		inhibited	
STOP	SLEEP	inhibited		RXONLY to DIAG		RXONLY to DIAG		RXONLY to DIAG	
STOP	STOP	inhibited		inhibited		RXONLY to DIAG		RXONLY to DIAG	

Comments:

1 - No transition possible or transition is not executed in one step

2 - Transition is needed. It is not executed right away, but after power save mode is released case 5 may become applicable.

3 - No transition possible or transition is not executed in one step

4 - Buffer assignment changes; transition flow needed.

5 - Buffer assignment changes, but transition flow can be executed as whole macro is in SLEEP or STOP mode.

Appendix C Index

A

address set up wait states . . . 277
address setup wait control register . . . 213
address space . . . 63
address wait control register . . . 213
ANI00 to ANI11 . . . 50
ASC . . . 213
AVDD . . . 50
AVREF . . . 51
AVSS . . . 50

B

BCC . . . 214
BCT0 . . . 195
BCT1 . . . 195
BEC . . . 198
BPC . . . 186
BSC . . . 197
Bus cycle configuration registers 0, 1(BCT0, BCT1) . . . 195
Bus cycle control register . . . 214
Bus size configuration register . . . 197
Byte access (8 bits) . . . 200

C

CALLT . . . 58
CAN Global Automatic Block Transmission Control Register . . . 533
CAN Global Automatic Block Transmission Delay Register . . . 536
CAN Global Clock Selection Register . . . 532
CAN Global Configuration Register . . . 535, 700
CAN Global Control Register . . . 530
CAN Message Configuration Register . . . 564
CAN Message Control Register m . . . 566
CAN Message Data Byte Register . . . 561
CAN Message Data Length Register m . . . 563
CAN Message ID Register m . . . 565
CAN Module Bit Rate Prescaler Register . . . 550
CAN Module Bit Rate Register . . . 551
CAN Module Control Register . . . 539
CAN Module Error Counter Register . . . 545
CAN Module Information Register . . . 544
CAN Module Interrupt Enable Register . . . 546
CAN Module Interrupt Status Register . . . 548
CAN Module Last Error Code Register . . . 543
CAN Module Last In-Pointer Register . . . 553
CAN Module Last Out-Pointer Register . . . 556
CAN Module Mask Control Register . . . 537
CAN Module Receive History List Register . . . 554
CAN Module Time Stamp Register . . . 559
CAN Module Transmit History List Register . . . 557
CGMABT . . . 699
CGMABTD . . . 701
CGMCS . . . 698
CGMCTRL . . . 697
Chip area selection control registers 0, 1(CSC0, CSC1) . . . 193
CKC . . . 218

Clock control register 218
Clock sources 217
CnBRP 550, 722
CnBRP_R 734
CnBTR 551, 723
CnBTR_R 735
CnCTRL 539, 715
CnCTRL_R 729
CnERC 545
CnGMABTD 536
CnGMCONF 535, 700
CnGMCS 532
CnGMCTR 530
CnIE 546, 720
CnIE_R 732
CnINFO 544, 719
CnINTS 548, 721
CnINTS_R 733
CnLEC 543, 718
CnLEC_R 731
CnLIPT 553, 724
CnLIPT_R 736
CnLOPT 556, 726
CnMASK1H 712
CnMASK1L 712
CnMASK2H 712
CnMASK2L 712
CnMASK3H 713
CnMASK3L 713
CnMASK4H 714
CnMASK4L 714
CnMASKaH 537
CnMASKaL 537
CnMCONFm 564
CnMCTRLm 566
CnMDATAxm 561
CnMDATAzm 561
CnMDLCm 563
CnMIDHm 565
CnMIDLm 565
CnRGPT 554, 725
CnTGPT 557, 727
CnTIDMH 707
CnTIDML 707
CnTIDRmH 702
CnTIDRmL 702
CnTS 559, 728
CnTS_R 737
COM0 to COM1 50
Command register 188
CSC0 193
CSC1 193
CTBP 58
CTPC 58
CTPSW 58
CVDD 50

D

Data space 65
Data wait control registers 0, 1 212
DBPC 58
DBPSW 58
DWC0 283
DWC0, DWC1 212
DWC1 283

E

ECR 58, 59
EIPC 58
EIPSW 58
Element pointer 57
Endian configuration register 198
EP 321
Exception status flag 321
exception table 67
Exception trap 322
External I/O 277
External ROM 277

F

FEPC 58
FEPSW 58
Functions of each port 751
Functions of each port pin on reset and registers that set port or control mode 752

G

general registers 55, 57
Global pointer 57

H

Halfword access (16 bits) 202
HALT mode 223, 224
handler addresses 67

I

ID 312
IDLE mode 223, 226
idle states 277
Illegal opcode definition 322
images 64
IMR0 to IMR3 310
In-service priority register 311
instruction cycle 55
Internal RAM area 69
Interrupt control register 307
Interrupt mask registers 0 to 3 310
Interrupt mode register 0 315
Interrupt mode register 1 316
Interrupt mode register 2 317
Interrupt mode register 3 318
Interrupt response time 326
Interrupt Source Register 59
Interrupt Vector Table 294
INTM0 315
INTM1 316

INTM2 317
INTM3 318
ISPR 311

L
Link pointer 57
low-power systems 221

M
Maskable interrupt status flag 312
Maskable interrupts 300
Priorities 303
MCONFm 744
MCTRLm 747
MDATA0m 741
MDATA1m 741
MDATA2m 741
MDATA3m 741
MDATA4m 741
MDATA5m 742
MDATA6m 742
MDATA7m 742
MDLCm 743
MIDHm 746
MIDLm 746
MODE0 to MODE3 50
Multiple interrupt processing control 324

N
NMI 50
NMI pin input 296
NMI0 298
Non-maskable interrupt 296
Non-maskable interrupt status flag 299
Non-port pins 36
NP 299

O
off-page 283
on-page 283

P
P0 759
P1 761
P2 764
P3 767
P4 771
Page ROM configuration register 287
PAH 775
PAL 777
PC 57
PCS 779
PCT 781
PDL 783
peripheral I/O 70
Peripheral I/O Registers 72
PFC3 770
Pin functions 33, 39

Pin I/O circuits 54
pipeline 55
PLL synthesizer 217
PM0 760
PM1 762
PM2 765
PM3 768
PM4 772
PMAH 776
PMAL 778
PMC0 760
PMC1 763
PMC2 766
PMC3 769
PMC4 773
PMCAH 776
PMCAL 778
PMCCS 780
PMCCT 782
PMCDL 784
PMCS 780
PMCT 782
PMDL 784
Port 0 759
Port 1 . . . . . 39, 40, 761
Port 2 . . . . . 41, 764
Port 3 . . . . . 42, 43, 767
Port 4 771
Port 6 44
Port 7 774
Port AH 45, 46, 48, 775
Port AL 777
Port configuration 750
Port CS 47, 779
Port CT 49, 781
Port DL 783
Port functions 749
Port pins 33
Power save control register 229
Power save mode register 230
Power save modes 217
Power saving functions 221
PRC 287
PRCMD 188
Program counter 57
program register 56
Program register set 56
Program space 65
Program Status Word 60
Programmable peripheral I/O registers 185
PSC 229
PSM 230
PSW 58, 60

R

RESET 50
ROMC 283

S

Saturation 61
Software exception 319
Software STOP mode 223, 227
Specific Registers 187
SRAM 277
Stack pointer 57
system register 56
System register set 56
System registers 58

T

Text pointer 57

V

VCMPM 299
VDD5 50
Voltage comparator mode register 299
VSS5 50
VSWC 189

W

Wait function 191
wait states 189
Word access (32 bits) 204
Wrap-around 65

X

X1 50
X2 50

Z

Zero register 57

Appendix D Revision History

The following shows the revision history up to present. Application portions signifies the chapter of each edition.

(1/2)

Edition No.	Major items revised	Revised Sections
EE2V1	I/O Circuit Type for MODE0 and Reset changed in **Chapter 2.3 Pin I/O Circuit and Connection of Unused Pins**	Chapter 2, p.52
	Type 2 port type added in **Figure 2-1**	Chapter 2, p.54
	Removed description of PHCMD and PHS **Chapter 3.8 Specific Registers**	Chapter 3, p.188
		Chapter 3, p.189
	PRCMD remains valid!	Chapter 3, p.190
	Removed caution in **Figure 5-2 Clock Control Register (CKC)**	Chapter 5, p.218
	Change of Alternate Function Pin description in **Table 6-2: TMP Pin List**	Chapter 6, p.232
	Added Note in **Figure 6-11**, **Figure 6-12** and **Figure 6-13** and renaming of TSOUTD to TSOUTR	Chapter 6, p.245
		Chapter 6, p.246
		Chapter 6, p.247
	INTP10 description added in **Figure 8-21**	Chapter 8, p.317
	Advanced NMI detection opened in **Figure 8-22**	Chapter 8, p.318
	ADC caution added	Chapter 9, p.329
	Conversion Mode Setting Example **Table 9-2** revised	Chapter 9, p.336
	Note in **Figure 10-5** added	Chapter 10, p.367
	Figure 11-5; CBnTSF caution added	Chapter 11, p.400
	Chapter **AFCAN** revised	Chapter 13
	Chapter **DAFCAN** revised	Chapter 14
	PMC01 bit removed from **Figure 15-9**	Chapter 15, p.760
	PDLH & PDLL address values corrected **Figure 15-36**	Chapter 15, p.783
	Appendix B - Transition Diagrams RXONLY Channel & DIAG Channel added	AppB, p.809

(2/2)

Edition No.	Major items revised	Revised Sections

NEC

Facsimile Message

From:

Name

Company

Tel. FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America
NEC Electronics America Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Marketing Services & Publishing
Fax: +49(0)-211-6503-1344

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

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

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-6250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: +81- 44-435-9608

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	☐	☐	☐	☐
Technical Accuracy	☐	☐	☐	☐
Organization	☐	☐	☐	☐

CS 99.1