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.

Downloaded from Arrow.com.

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.

Downloaded from Arrow.com.

User's Manual

V850ES/KE1

32-bit Single-Chip Microcontrollers

Hardware

μPD703207
μPD703207Y
μPD70F3207H
μPD70F3207HY

Document No. U16892EJ2V0UD00 (2nd edition)
Date Published August 2006 N CP(K)

© NEC Electronics Corporation 2004
Printed in Japan

Downloaded from Arrow.com.

[MEMO]

Downloaded from Arrow.com.

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.

Downloaded from Arrow.com.

Caution: The μPD70F3207H and 70F3207HY use SuperFlash® technology licensed from Silicon Storage Technology, Inc.

IECUBE is a registered trademark of NEC Electronics Corporation in Japan and Germany.
MINICUBE is a registered trademark of NEC Electronics Corporation in Japan and Germany or a trademark in the United States of America.
EEPROM is a trademark of NEC Electronics Corporation.
Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT is a trademark of International Business Machines Corporation.
SPARCstation is a trademark of SPARC International, Inc.
Solaris and SunOS are trademarks of Sun Microsystems, Inc.
SuperFlash is a registered trademark of Silicon Storage Technology, Inc. in several countries including the United States and Japan.

Downloaded from Arrow.com.

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

- **The information in this document is current as of January, 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

Downloaded from Arrow.com.

PREFACE

Readers

This manual is intended for users who wish to understand the functions of the V850ES/KE1 and design application systems using these products.
The target products are as follows.

- μPD703207, 703207Y, 70F3207H, 70F3207HY

Purpose

This manual is intended to give users an understanding of the hardware functions of the V850ES/KE1 shown in the **Organization** below.

Organization

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

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

How to Read This Manual

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

To find the details of a register where the name is known
→ Refer to **APPENDIX C REGISTER INDEX**.

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

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

To understand the overall functions of the V850ES/KE1
→ Read this manual according to the **CONTENTS**.

To know the electrical specifications of the V850ES/KE1
→ Refer to **CHAPTER 24 ELECTRICAL SPECIFICATIONS**

The "yyy bit of the xxx register" is described as the "xxx.yyy bit" in this manual. Note with caution that even if "xxx.yyy" is described as is in a program, however, the compiler/assembler cannot recognize it correctly.

The mark <R> shows major revised points. The revised points can be easily searched by copying an "<R>" in the PDF file and specifying it in the "Find what:" field.

Downloaded from Arrow.com.

Conventions

Data significance: Higher digits on the left and lower digits on the right

Active low representation: $\overline{xxx}$ (overscore over pin or signal name)

Memory map address: Higher addresses on the top and lower addresses on the bottom

Note: Footnote for item marked with **Note** in the text

Caution: Information requiring particular attention

Remark: Supplementary information

Numeric representation: Binary ... xxxx or xxxxB
Decimal ... xxxx
Hexadecimal ... xxxxH

Prefix indicating power of 2 (address space, memory capacity):
K (kilo): $2^{10} = 1,024$
M (mega): $2^{20} = 1,024^2$
G (giga): $2^{30} = 1,024^3$

Related Documents

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

Documents related to V850ES/KE1

Document Name	Document No.
V850ES Architecture User's Manual	U15943E
V850ES/KE1 Hardware User's Manual	This manual

Documents related to development tools (user's manuals)

Document Name		Document No.
QB-V850MINI (On-Chip Debug Emulator)		U17638E
QB-V850ESKX1H (In-Circuit Emulator)		U17214E
CA850 Ver. 3.00 C Compiler Package	Operation	U17293E
	C Language	U17291E
	Assembly Language	U17292E
	Link Directive	U17294E
PM+ Ver. 6.00 Project Manager		U17178E
ID850QB Ver. 3.10 Integrated Debugger	Operation	U17435E
RX850 Ver. 3.20 or Later Real-Time OS	Basics	U13430E
	Installation	U17149E
	Technical	U13431E
	Task Debugger	U17420E
RX850 Pro Ver. 3.20 Real-Time OS	Basics	U13773E
	Installation	U17421E
	Technical	U13772E
	Task Debugger	U17422E
AZ850 Ver. 3.30 System Performance Analyzer		U17423E
PG-FP4 Flash Memory Programmer		U15260E

Downloaded from Arrow.com.

CONTENTS

CHAPTER 1 INTRODUCTION 16
1.1 K1 Series Product Lineup 16
1.1.1 V850ES/Kx1+, V850ES/Kx1 products lineup 16
1.1.2 78K0/Kx1+, 78K0/Kx1 products lineup 19
1.2 Features 22
1.3 Applications 23
1.4 Ordering Information 23
1.5 Pin Configuration (Top View) 24
1.6 Function Block Configuration 27
1.7 Overview of Functions 31

CHAPTER 2 PIN FUNCTIONS 32
2.1 List of Pin Functions 32
2.2 Pin I/O Circuits and Recommended Connection of Unused Pins 36
2.3 Pin I/O Circuits 38

CHAPTER 3 CPU FUNCTIONS 40
3.1 Features 40
3.2 CPU Register Set 41
3.2.1 Program register set 42
3.2.2 System register set 43
3.3 Operating Modes 49
3.4 Address Space 50
3.4.1 CPU address space 50
3.4.2 Wraparound of CPU address space 51
3.4.3 Memory map 52
3.4.4 Areas 54
3.4.5 Recommended use of address space 56
3.4.6 Peripheral I/O registers 58
3.4.7 Special registers 64
3.4.8 Cautions 67

CHAPTER 4 PORT FUNCTIONS 70
4.1 Features 70
4.2 Basic Port Configuration 70
4.3 Port Configuration 71
4.3.1 Port 0 77
4.3.2 Port 3 79
4.3.3 Port 4 84
4.3.4 Port 5 86
4.3.5 Port 7 89
4.3.6 Port 9 90
4.3.7 Port CM 96
4.3.8 Port DL 97
4.4 Block Diagrams 98

Downloaded from Arrow.com.

4.5 Port Register Setting When Alternate Function Is Used 117
4.6 Cautions 121
4.6.1 Cautions on bit manipulation instruction for port n register (Pn) 121
4.6.2 Hysteresis characteristics 122

CHAPTER 5 CLOCK GENERATION FUNCTION 123
5.1 Overview 123
5.2 Configuration 124
5.3 Register 126
5.4 Operation 130
5.4.1 Operation of each clock 130
5.4.2 Clock output function 130
5.4.3 External clock input function 130
5.5 PLL Function 131
5.5.1 Overview 131
5.5.2 Register 131
5.5.3 Usage 132

CHAPTER 6 16-BIT TIMER/EVENT COUNTER P (TMP) 133
6.1 Overview 133
6.2 Functions 133
6.3 Configuration 134
6.4 Registers 136
6.5 Operation 147
6.5.1 Interval timer mode (TP0MD2 to TP0MD0 bits = 000) 148
6.5.2 External event count mode (TP0MD2 to TP0MD0 bits = 001) 158
6.5.3 External trigger pulse output mode (TP0MD2 to TP0MD0 bits = 010) 166
6.5.4 One-shot pulse output mode (TP0MD2 to TP0MD0 bits = 011) 178
6.5.5 PWM output mode (TP0MD2 to TP0MD0 bits = 100) 185
6.5.6 Free-running timer mode (TP0MD2 to TP0MD0 bits = 101) 194
6.5.7 Pulse width measurement mode (TP0MD2 to TP0MD0 bits = 110) 211
6.5.8 Timer output operations 217
6.6 Eliminating Noise on Capture Trigger Input Pin (TIP0a) 218
6.7 Cautions 220

CHAPTER 7 16-BIT TIMER/EVENT COUNTER 0 221
7.1 Functions 221
7.2 Configuration 222
7.3 Registers 228
7.4 Operation 235
7.4.1 Interval timer operation 235
7.4.2 Square wave output operation 238
7.4.3 External event counter operation 241
7.4.4 Operation in clear & start mode entered by TI010 pin valid edge input 244
7.4.5 Free-running timer operation 254
7.4.6 PPG output operation 263
7.4.7 One-shot pulse output operation 266
7.4.8 Pulse width measurement operation 271

Downloaded from Arrow.com.

7.5 Special Use of TM01 280
7.5.1 Rewriting CR011 register during TM01 operation 280
7.5.2 Setting LVS01 and LVR01 bits 280
7.6 Cautions 282

CHAPTER 8 8-BIT TIMER/EVENT COUNTER 5 286
8.1 Functions 286
8.2 Configuration 287
8.3 Registers 290
8.4 Operation 293
8.4.1 Operation as interval timer 293
8.4.2 Operation as external event counter 295
8.4.3 Square-wave output operation 296
8.4.4 8-bit PWM output operation 298
8.4.5 Operation as interval timer (16 bits) 301
8.4.6 Operation as external event counter (16 bits) 303
8.4.7 Square-wave output operation (16-bit resolution) 304
8.4.8 Cautions 305

CHAPTER 9 8-BIT TIMER H 306
9.1 Functions 306
9.2 Configuration 306
9.3 Registers 309
9.4 Operation 313
9.4.1 Operation as interval timer/square wave output 313
9.4.2 PWM output mode operation 316
9.4.3 Carrier generator mode operation 322

CHAPTER 10 INTERVAL TIMER, WATCH TIMER 329
10.1 Interval Timer BRG 329
10.1.1 Functions 329
10.1.2 Configuration 329
10.1.3 Registers 331
10.1.4 Operation 333
10.2 Watch Timer 334
10.2.1 Functions 334
10.2.2 Configuration 334
10.2.3 Registers 335
10.2.4 Operation 337
10.3 Cautions 338

CHAPTER 11 WATCHDOG TIMER FUNCTIONS 340
11.1 Watchdog Timer 1 340
11.1.1 Functions 340
11.1.2 Configuration 342
11.1.3 Registers 342
11.1.4 Operation 344
11.2 Watchdog Timer 2 346

Downloaded from Arrow.com.

11.2.1 Functions 346
11.2.2 Configuration 347
11.2.3 Registers 347
11.2.4 Operation 349

CHAPTER 12 REAL-TIME OUTPUT FUNCTION (RTO) 350
12.1 Function 350
12.2 Configuration 351
12.3 Registers 352
12.4 Operation 354
12.5 Usage 355
12.6 Cautions 355
12.7 Security Function 356

CHAPTER 13 A/D CONVERTER 358
13.1 Function 358
13.2 Configuration 359
13.3 Registers 361
13.4 Operation 368
13.4.1 Basic operation 368
13.4.2 A/D conversion operation 369
13.4.3 Power fail monitoring function 369
13.4.4 Setting method 370
13.5 Cautions 371
13.6 How to Read A/D Converter Characteristics Table 376

CHAPTER 14 ASYNCHRONOUS SERIAL INTERFACE (UART) 380
14.1 Features 380
14.2 Configuration 381
14.3 Registers 383
14.4 Interrupt Requests 389
14.5 Operation 390
14.5.1 Data format 390
14.5.2 Transmit operation 391
14.5.3 Continuous transmission operation 393
14.5.4 Receive operation 397
14.5.5 Reception error 398
14.5.6 Parity types and corresponding operation 400
14.5.7 Receive data noise filter 401
14.6 Dedicated Baud Rate Generator n (BRGn) 402
14.6.1 Baud rate generator n (BRGn) configuration 402
14.6.2 Serial clock generation 403
14.6.3 Baud rate setting example 406
14.6.4 Allowable baud rate range during reception 407
14.6.5 Transfer rate during continuous transmission 409
14.7 Cautions 409

CHAPTER 15 CLOCKED SERIAL INTERFACE 0 (CSI0) 410

Downloaded from Arrow.com.

15.1 Features 410
15.2 Configuration 411
15.3 Registers 414
15.4 Operation 423
15.4.1 Transmission/reception completion interrupt request signal (INTCSI0n) 423
15.4.2 Single transfer mode 425
15.4.3 Continuous transfer mode 428
15.5 Output Pins 436

CHAPTER 16 I^2C BUS 437
16.1 Features 437
16.2 Configuration 440
16.3 Registers 442
16.4 Functions 456
16.4.1 Pin configuration 456
16.5 I^2C Bus Definitions and Control Methods 457
16.5.1 Start condition 457
16.5.2 Addresses 458
16.5.3 Transfer direction specification 459
16.5.4 $\overline{ACK}$ 460
16.5.5 Stop condition 461
16.5.6 Wait state 462
<R> 16.5.7 Wait state cancellation method 464
16.6 I^2C Interrupt Request Signals (INTIIC0) 465
16.6.1 Master device operation 466
16.6.2 Slave device operation (when receiving slave address data (address match)) 469
16.6.3 Slave device operation (when receiving extension code) 473
16.6.4 Operation without communication 477
16.6.5 Arbitration loss operation (operation as slave after arbitration loss) 478
16.6.6 Operation when arbitration loss occurs (no communication after arbitration loss) 480
16.7 Interrupt Request Signal (INTIIC0) Generation Timing and Wait Control 487
16.8 Address Match Detection Method 488
16.9 Error Detection 488
16.10 Extension Code 489
16.11 Arbitration 490
16.12 Wakeup Function 491
16.13 Communication Reservation 492
16.13.1 When communication reservation function is enabled (IICF0.IICRSV0 bit = 0) 492
16.13.2 When communication reservation function is disabled (IICF0.IICRSV0 bit = 1) 495
16.14 Cautions 496
16.15 Communication Operations 497
16.15.1 Master operation in single master system 498
16.15.2 Master operation in multimaster system 499
16.15.3 Slave operation 502
16.16 Timing of Data Communication 505

CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION 512
17.1 Overview 512

Downloaded from Arrow.com.

17.1.1 Features 512
17.2 Non-Maskable Interrupts 515
17.2.1 Operation 518
17.2.2 Restore 519
17.2.3 NP flag 520
17.3 Maskable Interrupts 521
17.3.1 Operation 521
17.3.2 Restore 523
17.3.3 Priorities of maskable interrupts 524
17.3.4 Interrupt control register (xxlCn) 528
17.3.5 Interrupt mask registers 0, 1, 3 (IMR0, IMR1, IMR3) 530
17.3.6 In-service priority register (ISPR) 531
17.3.7 ID flag 532
17.3.8 Watchdog timer mode register 1 (WDTM1) 533
17.4 External Interrupt Request Input Pins (NMI, INTP0 to INTP6) 534
17.4.1 Noise elimination 534
17.4.2 Edge detection 534
17.5 Software Exceptions 537
17.5.1 Operation 537
17.5.2 Restore 538
17.5.3 EP flag 539
17.6 Exception Trap 540
17.6.1 Illegal opcode 540
17.6.2 Debug trap 542
17.7 Multiple Interrupt Servicing Control 544
17.8 Interrupt Response Time 546
17.9 Periods in Which Interrupts Are Not Acknowledged by CPU 547
17.10 Cautions 547

CHAPTER 18 KEY INTERRUPT FUNCTION 548
18.1 Function 548
18.2 Register 549

CHAPTER 19 STANDBY FUNCTION 550
19.1 Overview 550
19.2 Registers 553
19.3 HALT Mode 556
19.3.1 Setting and operation status 556
19.3.2 Releasing HALT mode 556
19.4 IDLE Mode 558
19.4.1 Setting and operation status 558
19.4.2 Releasing IDLE mode 559
19.5 STOP Mode 561
19.5.1 Setting and operation status 561
19.5.2 Releasing STOP mode 562
19.5.3 Securing oscillation stabilization time when STOP mode is released 564
19.6 Subclock Operation Mode 565
19.6.1 Setting and operation status 565

Downloaded from Arrow.com.

19.6.2 Releasing subclock operation mode 565
19.7 Sub-IDLE Mode 567
19.7.1 Setting and operation status 567
19.7.2 Releasing sub-IDLE mode 568

CHAPTER 20 RESET FUNCTION 570
20.1 Overview 570
20.2 Configuration 570
20.3 Operation 571

CHAPTER 21 ROM CORRECTION FUNCTION 575
21.1 Overview 575
21.2 Registers 576
21.3 ROM Correction Operation and Program Flow 577

CHAPTER 22 FLASH MEMORY 579
22.1 Features 579
22.2 Memory Configuration 580
22.3 Functional Outline 581
22.4 Rewriting by Dedicated Flash Programmer 585
22.4.1 Programming environment 585
22.4.2 Communication mode 586
22.4.3 Flash memory control 591
22.4.4 Selection of communication mode 592
22.4.5 Communication commands 593
22.4.6 Pin connection 594
22.5 Rewriting by Self Programming 599
22.5.1 Overview 599
22.5.2 Features 600
22.5.3 Standard self programming flow 601
22.5.4 Flash functions 602
22.5.5 Pin processing 602
22.5.6 Internal resources used 603

<R> **CHAPTER 23 ON-CHIP DEBUG FUNCTION 604**
23.1 ROM Security Function 604
23.1.1 Security ID 604
23.1.2 Setting 605
23.2 Cautions 606

CHAPTER 24 ELECTRICAL SPECIFICATIONS 607

CHAPTER 25 PACKAGE DRAWINGS 627

<R> **CHAPTER 26 RECOMMENDED SOLDERING CONDITIONS 629**

APPENDIX A DEVELOPMENT TOOLS 631
A.1 Software Package 634

Downloaded from Arrow.com.

A.2 Language Processing Software 634
A.3 Control Software 634
A.4 Debugging Tools (Hardware) 635
A.4.1 When using IECUBE QB-V850ESKX1H 635
<R> A.4.2 When using MINICUBE QB-V850MINI 637
A.5 Debugging Tools (Software) 639
A.6 Embedded Software 640
A.7 Flash Memory Writing Tools 640

APPENDIX B INSTRUCTION SET LIST 641
B.1 Conventions 641
B.2 Instruction Set (in Alphabetical Order) 644

APPENDIX C REGISTER INDEX 651

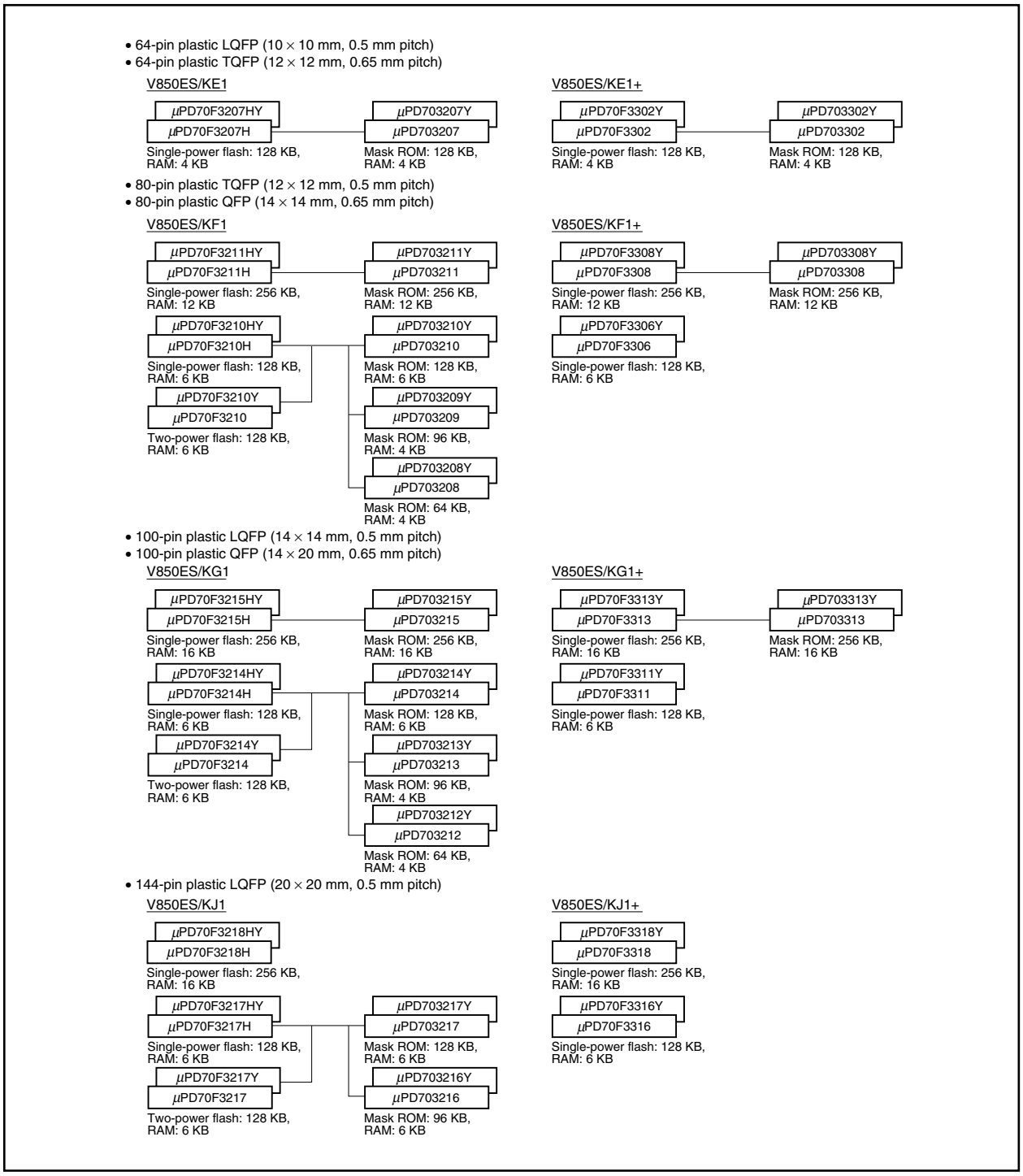
<R> **APPENDIX D REVISION HISTORY 656**
D.1 Major Revisions in This Edition 656

Downloaded from Arrow.com.

CHAPTER 1 INTRODUCTION

1.1 K1 Series Product Lineup

1.1.1 V850ES/Kx1+, V850ES/Kx1 products lineup



Downloaded from Arrow.com.

The function list of the V850ES/Kx1+ is shown below.

Product Name		V850ES/KE1+		V850ES/KF1+			V850ES/KG1+			V850ES/KJ1+	
Number of pins		64 pins		80 pins			100 pins			144 pins	
Internal memory (KB)	Mask ROM	128	–	–	256	–	–	256	–	–	–
	Flash memory	–	128	128	–	256	128	–	256	128	256
	RAM	4		6	12		6	16		6	16
Supply voltage		2.7 to 5.5 V									
Minimum instruction execution time		50 ns @20 MHz									
Clock	X1 input	2 to 10 MHz									
	Subclock	32.768 kHz									
	Internal oscillation	240 kHz (TYP.)									
Port	CMOS input	8		8			8			16	
	CMOS I/O	41 (4)Note 1		57 (6)Note 1			72 (8)Note 1			106 (12)Note 1	
	N-ch open-drain I/O	2		2			4			6	
Timer	16-bit (TMP)	1 ch		1 ch			1 ch			1 ch	
	16-bit (TM0)	1 ch		2 ch			4 ch			6 ch	
	8-bit (TM5)	2 ch		2 ch			2 ch			2 ch	
	8-bit (TMH)	2 ch		2 ch			2 ch			2 ch	
	Interval timer	1 ch		1 ch			1 ch			1 ch	
	Watch	1 ch		1 ch			1 ch			1 ch	
	WDT1	1 ch		1 ch			1 ch			1 ch	
	WDT2	1 ch		1 ch			1 ch			1 ch	
RTO		6 bits × 1 ch		6 bits × 1 ch			6 bits × 1 ch			6 bits × 2 ch	
Serial interface	CSI	2 ch		2 ch			2 ch			3 ch	
	Automatic transmit/receive 3-wire CSI	–		1 ch			2 ch			2 ch	
	UART	1 ch		1 ch			2 ch			2 ch	
	UART supporting LIN-bus	1 ch		1 ch			1 ch			1 ch	
	I^2CNote 2	1 ch		1 ch			1 ch			2 ch	
External bus	Address space	–		128 KB			3 MB			15 MB	
	Address bus	–		16 bits			22 bits			24 bits	
	Mode	–		Multiplex only			Multiplex/separate				
DMA controller		–		–			4 ch			4 ch	
10-bit A/D converter		8 ch		8 ch			8 ch			16 ch	
8-bit D/A converter		–		–			2 ch			2 ch	
Interrupt	External	9		9			9			9	
	Internal	26/27Note 2		29/30Note 2			41/42Note 2			46/48Note 2	
Key return input		8 ch		8 ch			8 ch			8 ch	
Reset	$\overline{\text{RESET}}$ pin	Provided									
	POC	2.7 V or less fixed									
	LVI	3.1 V/3.3 V ±0.15 V or 3.5 V/3.7 V/3.9 V/4.1 V/4.3 V ±0.2 V (selectable by software)									
	Clock monitor	Provided (monitor by internal oscillator)									
	WDT1	Provided									
	WDT2	Provided									
ROM correction		4								None	
Regulator		None		Provided							
Standby function		HALT/IDLE/STOP/sub-IDLE mode									
Operating ambient temperature		T_A = –40 to +85°C									

Notes 1. Figures in parentheses indicate the number of pins for which the N-ch open-drain output can be selected by software.

2. Only in products with an I^2C bus (Y products)

Downloaded from Arrow.com.

The function list of the V850ES/Kx1 is shown below.

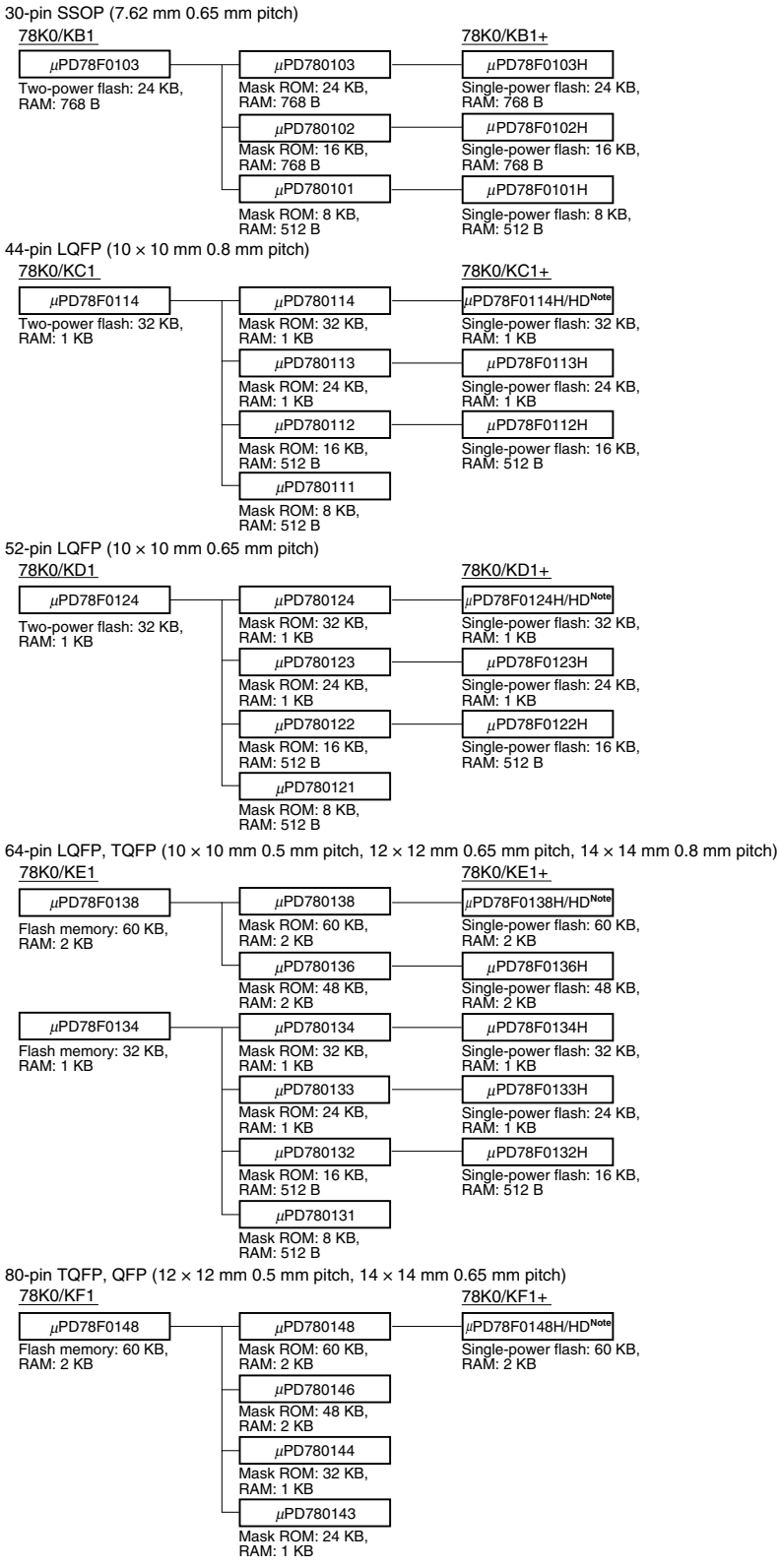
Product Name		V850ES/KE1		V850ES/KF1					V850ES/KG1					V850ES/KJ1		
Number of pins		64 pins		80 pins					100 pins					144 pins		
Internal memory (KB)	Mask ROM	128	–	64/96	128	–	256	–	64/96	128	–	256	–	96/128	–	–
	Flash memory	–	128	–	–	128	–	256	–	–	128	–	256	–	128	256
	RAM	4		4	6		12		4	6		16		6		16
Supply voltage		2.7 to 5.5 V														
Minimum instruction execution time		50 ns @20 MHz														
Clock	X1 input	2 to 10 MHz														
	Subclock	32.768 kHz														
	Internal oscillation	–														
Port	CMOS input	8		8					8					16		
	CMOS I/O	41 (4)[Note 1]		57 (6)[Note 1]					72 (8)[Note 1]					106 (12)[Note 1]		
	N-ch open-drain I/O	2		2					4					6		
Timer	16-bit (TMP)	1 ch		–			1 ch		–			1 ch		–		1 ch
	16-bit (TM0)	1 ch		2 ch					4 ch					6 ch		
	8-bit (TM5)	2 ch		2 ch					2 ch					2 ch		
	8-bit (TMH)	2 ch		2 ch					2 ch					2 ch		
	Interval timer	1 ch		1 ch					1 ch					1 ch		
	Watch	1 ch		1 ch					1 ch					1 ch		
	WDT1	1 ch		1 ch					1 ch					1 ch		
	WDT2	1 ch		1 ch					1 ch					1 ch		
RTO		6 bits × 1 ch		6 bits × 1 ch					6 bits × 1 ch					6 bits × 2 ch		
Serial interface	CSI	2 ch		2 ch					2 ch					3 ch		
	Automatic transmit/receive 3-wire CSI	–		1 ch					2 ch					2 ch		
	UART	2 ch		2 ch					2 ch					3 ch		
	UART supporting LIN-bus	–		–					–					–		
	I^2C[Note 2]	1 ch		1 ch					1 ch					2 ch		
External bus	Address space	–		128 KB					3 MB					15 MB		
	Address bus	–		16 bits					22 bits					24 bits		
	Mode	–		Multiplex only					Multiplex/separate							
DMA controller		–		–					–					–		
10-bit A/D converter		8 ch		8 ch					8 ch					16 ch		
8-bit D/A converter		–		–					2 ch					2 ch		
Interrupt	External	8		8					8					8		
	Internal	25/26[Note 2]		25/26[Note 2]			28/29[Note 2]		30/31[Note 2]			33/34[Note 2]		38/40[Note 2]		41/43[Note 2]
Key return input		8 ch		8 ch					8 ch					8 ch		
Reset	$\overline{\text{RESET}}$ pin	Provided														
	POC	None														
	LVI	None														
	Clock monitor	None														
	WDT1	Provided														
	WDT2	Provided														
ROM correction		4														
Regulator		None		Provided												
Standby function		HALT/IDLE/STOP/sub-IDLE mode														
Operating ambient temperature		T_A = –40 to +85°C														

Notes 1. Figures in parentheses indicate the number of pins for which the N-ch open-drain output can be selected by software.

2. Only in products with an I^2C bus (Y products)

Downloaded from Arrow.com.

1.1.2 78K0/Kx1+, 78K0/Kx1 products lineup



Note Products with an on-chip function

Downloaded from Arrow.com.

The list of functions in the 78K0/Kx1 is shown below.

Item \ Part Number		78K0/KB1			78K0/KC1			78K0/KD1			78K0/KE1					78K0/KF1		
Number of pins		30 pins			44 pins			52 pins			64 pins					80 pins		
Internal memory (KB)	Mask ROM	8	16/24	–	8/16	24/32	–	8/16	24/32	–	8/16	24/32	–	48/60	–	24/32	48/60	–
	Flash memory	–		24	–		32	–		32	–		32	–	60	–		60
	RAM	0.5	0.75		0.5	1		0.5	1		0.5	1		2		1	2	
Power supply voltage		V_{DD} = 2.5 to 5.5 V[Notes 1, 2]																
Minimum instruction execution time		0.166 μs (when 12 MHz, V_{DD} = 4.0 to 5.5 V) 0.2 μs (when 10 MHz, V_{DD} = 3.5 to 5.5 V) 0.238 μs (when 8.38 MHz, V_{DD} = 3.0 to 5.5 V) 0.4 μs (when 5 MHz, V_{DD} = 2.5 to 5.5 V)			<Connect REGC pin to V_{DD}> 0.166 μs (when 12 MHz, V_{DD} = 4.0 to 5.5 V) 0.2 μs (when 10 MHz, V_{DD} = 3.5 to 5.5 V) 0.238 μs (when 8.38 MHz, V_{DD} = 3.0 to 5.5 V) 0.4 μs (when 5 MHz, V_{DD} = 2.5 to 5.5 V)										
Clock	X1 input	2 to 12 MHz																
	Subclock	–			32.768 kHz													
	Internal oscillation	240 kHz (TYP.)																
Port	CMOS I/O	17			19			26			38					54		
	CMOS input	4			8													
	CMOS output	1																
	N-ch open-drain I/O	–			4													
Timer	16 bits (TM0)	1 ch									2 ch					1 ch	2 ch	
	8 bits (TM5)	1 ch			2 ch													
	8 bits (TMH)	2 ch																
	For watch	–			1 ch													
	WDT	1 ch																
Serial interface	3-wire CSI[Note 3]	1 ch									2 ch					1 ch	2 ch	
	Automatic transmit/ receive 3-wire CSI	–														1 ch		
	UART[Note 3]	–			1 ch													
	UART supporting LIN-bus	1 ch																
10-bit A/D converter		4 ch			8 ch													
Interrupt	External	6			7			8			9					9		
	Internal	11	12		15						16	19				17	20	
Key return input		–			4 ch			8 ch										
Reset	$\overline{\text{RESET}}$ pin	Provided																
	POC	2.85 V ±0.15 V/3.5 V ±0.20 V (selectable by mask option)																
	LVI	2.85 V/3.1 V/3.3 V ±0.15 V/3.5 V/3.7 V/3.9 V/4.1 V/4.3 V ±0.2 V (selectable by software)																
	Clock monitor	Provided																
	WDT	Provided																
Clock output/buzzer output		–						Clock output only			Provided							
Multiplier/divider		–									16 bits × 16 bits, 32 bits ÷ 16 bits							
ROM correction		–												Provided		–		
Standby function		HALT/STOP mode																
Operating ambient temperature		Standard and special (A) grade products: –40 to +85°C Special (A1) grade products: –40 to +110°C (mask ROM version), –40 to +105°C (flash memory version) Special (A2) grade products: –40 to +125°C (mask ROM version)																

Notes 1. If the POC circuit detection voltage (V_{POC}) is used with 2.85 V ±0.15 V, then use the products in the voltage range of 3.0 to 5.5 V.

2. If the POC circuit detection voltage (V_{POC}) is used with 3.5 V ±0.2 V, then use the products in the voltage range of 3.7 to 5.5 V.

3. Select either of the functions of these alternate-function pins.

Downloaded from Arrow.com.

The list of functions in the 78K0/Kx1+ is shown below.

Item \ Part Number		78K0/KB1+		78K0/KC1+		78K0/KD1+		78K0/KE1+			78K0/KF1+
Number of pins		30 pins		44 pins		52 pins		64 pins			80 pins
Internal memory (KB)	Flash memory	8	16/24	16	24/32	16	24/32	16	24/32	48/60	60
	RAM	0.5	0.75	0.5	1	0.5	1	0.5	1	2	2
Power supply voltage		V_{DD} = 2.5 to 5.5 V (with internal oscillation clock or subclock: V_{DD} = 2.0 to 5.5 V[Note 1])									
Minimum instruction execution time		0.125 μs (when 16 MHz, V_{DD} = 4.0 to 5.5 V), 0.2 μs (when 10 MHz, V_{DD} = 3.5 to 5.5 V), 0.238 μs (when 8.38 MHz, V_{DD} = 3.0 to 5.5 V), 0.4 μs (when 5 MHz, V_{DD} = 2.5 to 5.5 V)									
Clock	Crystal/ceramic	2 to 16 MHz									
	RC	3 to 4 MHz				–					
	Subclock	–		32.768 kHz							
	Internal oscillation	240 kHz (TYP.)									
Ports	CMOS I/O	17		19		26		38			54
	CMOS input	4		8							
	CMOS output	1									
	N-ch open-drain I/O	–		4							
Timer	16 bits (TM0)	1 ch						2 ch			
	8 bits (TM5)	1 ch		2 ch							
	8 bits (TMH)	2 ch									
	For watch	–		1 ch							
	WDT	1 ch									
Serial interface	3-wire CSI[Note 2]	1 ch						2 ch			
	Automatic transmit/ receive 3-wire CSI	–									1 ch
	UART[Note 2]	–	1 ch								
	UART supporting LIN-bus	1 ch									
10-bit A/D converter		4 ch		8 ch							
Interrupts	External	6		7		8		9			9
	Internal	11	12	15				16	19		20
Key return input		–		4 ch		8 ch					
Reset	$\overline{\text{RESET}}$ pin	Provided									
	POC	2.1 V ±0.1 V (detection voltage is fixed)									
	LVI	2.35 V/2.6 V/2.85 V/3.1 V/3.3 V ±0.15 V/3.5 V/3.7 V/3.9 V/4.1 V/4.3 V ±0.2 V (selectable by software)									
	Clock monitor	Provided									
	WDT	Provided									
Clock output/buzzer output		–				Clock output only		Provided			
External bus interface		–									Provided
Multiplier/divider		–						16 bits × 16 bits, 32 bits ÷ 16 bits			
ROM correction		–								Provided	–
Self-programming function		Provided									
Product with on-chip debug function		μPD78F0114HD, 78F0124HD, 78F0138HD, 78F0148HD									
Standby function		HALT/STOP mode									
Operating ambient temperature		T_A = –40 to +85°C									

Notes 1. Because the POC circuit detection voltage (V_{POC}) is 2.1 V ±0.1 V, use the products in the voltage range of 2.2 to 5.5 V.

2. Select either of the functions of these alternate-function pins.

Downloaded from Arrow.com.

1.2 Features

- Minimum instruction execution time: 50 ns (operation at main clock (f_{XX}) = 20 MHz)
- General-purpose registers: 32 bits × 32 registers
- CPU features: Signed multiplication (16 × 16 → 32): 1 to 2 clocks
  (Instructions without creating register hazards can be continuously executed in parallel)
  Saturated operations (overflow and underflow detection functions are included)
  32-bit shift instruction: 1 clock
  Bit manipulation instructions
  Load/store instructions with long/short format
- Memory space: 64 MB of linear address space
  - Internal memory
    μPD703207, 703207Y (Mask ROM: 128 KB/RAM: 4 KB)
    μPD70F3207H, 70F3207HY (Flash memory: 128 KB/RAM: 4 KB)
- Interrupts and exceptions
  Non-maskable interrupts: 3 sources
  Maskable interrupts: 30 sources (μPD703207, 70F3207H)
  31 sources (μPD703207Y, 70F3207HY)
  Software exceptions: 32 sources
  Exception trap: 1 source
- I/O lines: Total: 51
- Key interrupt function
- Timer function
  16-bit timer/event counter P: 1 channel
  16-bit timer/event counter 0: 1 channel
  8-bit timer/event counter 5: 2 channels
  8-bit timer H: 2 channels
  8-bit interval timer BRG: 1 channel
  Watch timer/interval timer: 1 channel
  Watchdog timers
  Watchdog timer 1 (also usable as oscillation stabilization timer): 1 channel
  Watchdog timer 2: 1 channel
- Serial interface
  Asynchronous serial interface (UART): 2 channels
  3-wire serial I/O (CSI0): 2 channels
  I^2C bus interface (I^2C): 1 channel (μPD703207Y, 70F3207HY)
- A/D converter: 10-bit resolution × 8 channels
- Real-time output port: 6 bits × 1 channel
- Standby functions: HALT/IDLE/STOP modes, subclock/sub-IDLE modes
- ROM correction: 4 correction addresses specifiable
- Clock generator
  Main clock oscillation (f_X)/subclock oscillation (f_{XT})
  CPU clock (f_{CPU}) 7 steps (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, f_{XT})
  Clock-through mode/PLL mode selectable
- Reset
  - Reset by $\overline{\text{RESET}}$ pin
  - Reset by overflow of watchdog timer 1 (WDTRES1)
  - Reset by overflow of watchdog timer 2 (WDTRES2)

Downloaded from Arrow.com.

○ Package: 64-pin plastic TQFP (12 × 12)
64-pin plastic LQFP (fine pitch) (10 × 10)

1.3 Applications

○ Automotive
- System control of body electrical system (power windows, keyless entry reception, etc.)
- Submicrocontroller of control system

○ Home audio, car audio
○ AV equipment
○ PC peripheral devices (keyboards, etc.)
○ Household appliances
- Outdoor units of air conditioners
- Microwave ovens, rice cookers

○ Industrial devices
- Pumps
- Vending machines
- FA

<R>

1.4 Ordering Information

Part Number	Package	Quality Grade
μPD703207GK-×××-9ET-A	64-pin plastic TQFP (12 × 12)	Standard
μPD703207GB-×××-8EU-A	64-pin plastic LQFP (fine pitch) (10 × 10)	Standard
μPD703207YGK-×××-9ET-A	64-pin plastic TQFP (12 × 12)	Standard
μPD703207YGB-×××-8EU-A	64-pin plastic LQFP (fine pitch) (10 × 10)	Standard
μPD70F3207HGK-9ET-A	64-pin plastic TQFP (12 × 12)	Standard
μPD70F3207HGB-8EU-A	64-pin plastic LQFP (fine pitch) (10 × 10)	Standard
μPD70F3207HYGK-9ET-A	64-pin plastic TQFP (12 × 12)	Standard
μPD70F3207HYGB-8EU-A	64-pin plastic LQFP (fine pitch) (10 × 10)	Standard

Remarks 1. ××× indicates ROM code suffix.

2. Products with -A at the end of the part number are lead-free products.

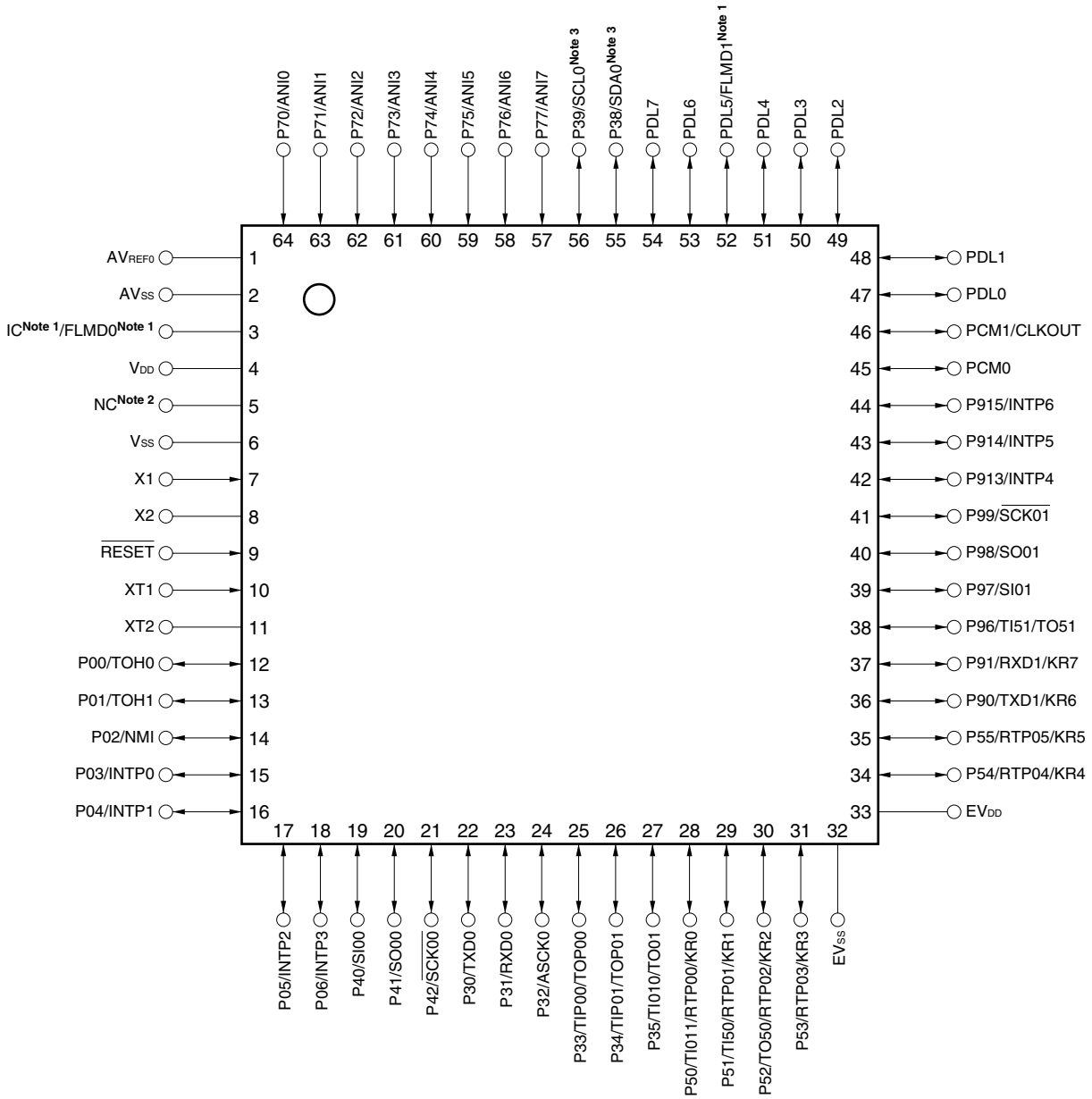
Downloaded from Arrow.com.

1.5 Pin Configuration (Top View)

64-pin plastic TQFP (12 × 12)
64-pin plastic LQFP (fine pitch) (10 × 10)

μPD703207GK-×××-9ET-A	μPD70F3207HGK-9ET-A
μPD703207GB-×××-8EU-A	μPD70F3207HGB-8EU-A
μPD703207YGK-×××-9ET-A	μPD70F3207HYGK-9ET-A
μPD703207YGB-×××-8EU-A	μPD70F3207HYGB-8EU-A

Downloaded from Arrow.com.



Notes 1. IC pin: Connect directly to V_{SS} (μPD703207, 703207Y).

FLMD0 pin: Connect to V_{SS} in normal operation mode (μPD70F3207H, 70F3207HY).

FLMD1 pin: Used only in the μPD70F3207H and 70F3207HY.

2. Leave open.

3. The SCL0 and SDA0 pins can be used only in the μPD703207Y and 70F3207HY.

Caution Make EV_{DD} the same potential as V_{DD}.

Downloaded from Arrow.com.

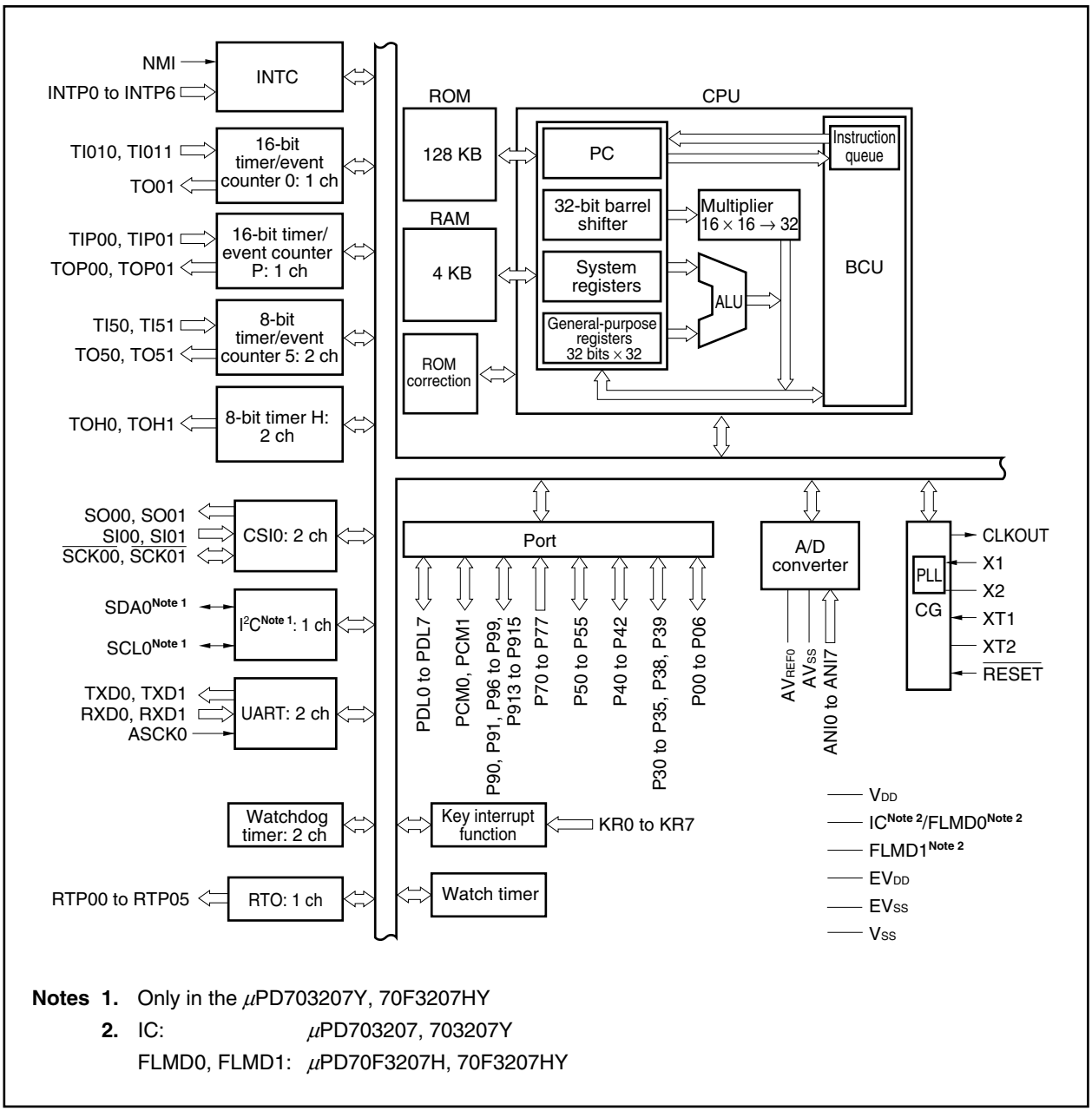
Pin identification

ANI0 to ANI7:	Analog input	PDL0 to PDL7:	Port DL
ASCK0:	Asynchronous serial clock	$\overline{\text{RESET}}$:	Reset
AV_{REF0}:	Analog reference voltage	RTP00 to RTP05:	Real-time output port
AV_{SS}:	Ground for analog	RXD0, RXD1:	Receive data
CLKOUT:	Clock output	$\overline{\text{SCK00}}$, $\overline{\text{SCK01}}$:	Serial clock
EV_{DD}:	Power supply for port	SCL0:	Serial clock
EV_{SS}:	Ground for port	SDA0:	Serial data
FLMD0, FLMD1:	Flash programming mode	SI00, SI01:	Serial input
IC:	Internally connected	SO00, SO01:	Serial output
INTP0 to INTP6:	External interrupt input	TI010, TI011,	
KR0 to KR7:	Key return	TI50, TI51,	
NC:	Non-connection	TIP00, TIP01:	Timer input
NMI:	Non-maskable interrupt request	TO01,	
P00 to P06:	Port 0	TO50, TO51,	
P30 to P35, P38, P39:	Port 3	TOH0, TOH1,	
P40 to P42:	Port 4	TOP00, TOP01:	Timer output
P50 to P55:	Port 5	TXD0, TXD1:	Transmit data
P70 to P77:	Port 7	V_{DD}:	Power supply
P90, P91, P96 to P99,		V_{SS}:	Ground
P913 to P915:	Port 9	X1, X2:	Crystal for main clock
PCM0, PCM1:	Port CM	XT1, XT2:	Crystal for subclock

Downloaded from Arrow.com.

1.6 Function Block Configuration

(1) Internal block diagram



Notes 1. Only in the μPD703207Y, 70F3207HY

2. IC: μPD703207, 703207Y

FLMD0, FLMD1: μPD70F3207H, 70F3207HY

Downloaded from Arrow.com.

(2) Internal units

(a) 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 types of instruction processing.

Other dedicated on-chip hardware, such as a multiplier (16 bits × 16 bits → 32 bits) and a barrel shifter (32 bits) help accelerate complex processing.

(b) Bus control unit (BCU)

The BCU controls the internal bus.

(c) ROM

This consists of a 128 KB mask ROM or flash memory mapped to the address spaces from 0000000H to 001FFFFH.

ROM can be accessed by the CPU in one clock cycle during instruction fetch.

(d) RAM

This consists of a 4 KB RAM mapped to the address spaces from 3FFE000H to 3FFEFFFH.

RAM can be accessed by the CPU in one clock cycle during data access.

(e) Interrupt controller (INTC)

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

(f) Clock generator (CG)

A main clock oscillator and subclock oscillator are provided and generate the main clock oscillation frequency (f_X) and subclock frequency (f_{XT}), respectively.

There are two modes: In the clock-through mode, f_X is used as the main clock frequency (f_{XX}) as is. In the PLL mode, f_X is used multiplied by 4.

The CPU clock frequency (f_{CPU}) can be selected from among f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, and f_{XT}.

(g) Timer/counter

One 16-bit timer/event counter 0 channel, one 16-bit timer/event counter P channel, and two 8-bit timer/event counter 5 channels are incorporated, enabling measurement of pulse intervals and frequency as well as programmable pulse output.

Two 8-bit timer/event counter 5 channels can be connected in cascade to configure a 16-bit timer.

Two 8-bit timer H channels enabling programmable pulse output are provided on chip.

(h) Watch timer

This timer counts the reference time (0.5 seconds) for counting the clock from the subclock (32.768 kHz) or f_{BRG} (32.768 kHz) from the clock generator. At the same time, the watch timer can be used as an interval timer.

Downloaded from Arrow.com.

(i) Watchdog timer

Two watchdog timer channels are provided on chip to detect program loops and system abnormalities. Watchdog timer 1 can be used as an interval timer. When used as a watchdog timer, it generates a non-maskable interrupt request signal (INTWDT1) or system reset signal (WDTRES1) after an overflow occurs. When used as an interval timer, it generates a maskable interrupt request signal (INTWDTM1) after an overflow occurs.

Watchdog timer 2 operates by default following reset release.

It generates a non-maskable interrupt request signal (INTWDT2) or system reset signal (WDTRES2) after an overflow occurs.

(j) Serial interface (SIO)

The V850ES/KE1 includes three kinds of serial interfaces: an asynchronous serial interface (UARTn), a clocked serial interface (CSI0n), and an I^2C bus interface (I^2C0). The μPD703207 and 70F3207H can simultaneously use up to four channels, and the μPD703207Y and 70F3207HY up to five channels.

For UARTn, data is transferred via the TXDn and RXDn pins.

For CSI0n, data is transferred via the SO0n, SI0n, and $\overline{SCK0n}$ pins.

For I^2C0, data is transferred via the SDA0 and SCL0 pins.

I^2C0 is provided only in the μPD703207Y and 70F3207HY.

Remark n = 0, 1

(k) A/D converter

This high-speed, high-resolution 10-bit A/D converter includes 8 analog input pins. Conversion is performed using the successive approximation method.

(l) ROM correction

This function is used to replace part of a program in the mask ROM with that contained in the internal RAM. Up to four correction addresses can be specified.

(m) Key interrupt function

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

(n) Real-time output function

This function transfers 6-bit data set beforehand to output latches upon occurrence of a timer compare register match signal.

A 1-channel 6-bit data real-time output function is provided on chip.

Downloaded from Arrow.com.

(o) Ports

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

Port	I/O	Alternate Function
P0	7-bit I/O	NMI, external interrupt, timer output
P3	8-bit I/O	Serial interface, timer I/O
P4	3-bit I/O	Serial interface
P5	6-bit I/O	Timer I/O, key interrupt function, real-time output function
P7	8-bit input	A/D converter analog input
P9	9-bit I/O	Serial interface, timer I/O, external interrupt, key interrupt function
PCM	2-bit I/O	Clock output
PDL	8-bit I/O	–

Downloaded from Arrow.com.

1.7 Overview of Functions

Part Number		μPD703207, 703207Y	μPD70F3207H, 70F3207HY
Internal memory	ROM	128 KB	128 KB (flash memory)
	High-speed RAM	4 KB	
Memory space		64 MB	
General-purpose registers		32 bits × 32 registers	
Main clock (oscillation frequency)		Ceramic/crystal/external clock	
		When PLL not used (2 to 10 MHz: 2.7 to 5.5 V)	
		When PLL used (2 to 5 MHz: 4.5 to 5.5 V, 2 to 2.5 MHz: 2.7 to 5.5 V)	
Subclock (oscillation frequency)		Crystal/external clock (32.768 kHz)	
Minimum instruction execution time		50 ns (When main clock operated at (f_{XX}) = 20 MHz)	
DSP function		32 × 32 = 64: 200 to 250 ns (at 20 MHz) 32 × 32 + 32 = 32: 300 ns (at 20 MHz) 16 × 16 = 32: 50 to 100 ns (at 20 MHz) 16 × 16 + 32 = 32: 150 ns (at 20 MHz)	
I/O ports		51 • Input: 8 • I/O: 43 (among these, N-ch open-drain output selectable: 4, fixed to N-ch open-drain output: 2)	
Timer		16-bit timer/event counter P: 1 channel 16-bit timer/event counter 0: 1 channel 8-bit timer/event counter 5: 2 channels (16-bit timer/event counter: usable as 1 channel) 8-bit timer H: 2 channels Watch timer: 1 channel 8-bit interval timer: 1 channel Watchdog timer: 2 channels	
Real-time output port		4 bits × 1, 2 bits × 1, or 6 bits × 1	
A/D converter		10-bit resolution × 8 channels	
Serial interface		CSI: 2 channels UART: 2 channels I^2C bus: 1 channel[Note 1] Dedicated baud rate generator: 2 channels	
Interrupt sources		External: 9 (9)[Note 2], internal: 25/26[Note 1]	
Power save function		STOP/IDLE/HALT/sub-IDLE mode	
Operating supply voltage		4.5 to 5.5 V (at 20 MHz)/2.7 to 5.5 V (at 10 MHz)	
Package		64-pin plastic TQFP (12 × 12 mm) 64-pin plastic LQFP (fine pitch) (10 × 10 mm)	

Notes 1. Only in the μPD703207Y, 70F3207HY

2. The figure in parentheses indicates the number of external interrupts for which STOP mode can be released.

Downloaded from Arrow.com.

CHAPTER 2 PIN FUNCTIONS

The names and functions of the pins of the V850ES/KE1 are described below, divided into port pins and non-port pins.

The pin I/O buffer power supplies are divided into two systems; AV_{REF0} and EV_{DD}. The relationship between these power supplies and the pins is shown below.

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

Power Supply	Corresponding Pins
AV_{REF0}	Port 7
EV_{DD}	$\overline{RESET}$, ports 0, 3 to 5, 9, CM, DL

2.1 List of Pin Functions

(1) Port pins

(1/2)

Pin Name	Pin No.	I/O	Pull-up Resistor	Function	Alternate Function
P00	12	I/O	Yes	Port 0 I/O port Input/output can be specified in 1-bit units.	TOH0
P01	13				TOH1
P02	14				NMI
P03	15				INTP0
P04	16				INTP1
P05	17				INTP2
P06	18				INTP3
P30	22	I/O	Yes	Port 3 I/O port Input/output can be specified in 1-bit units. P38 and P39 are fixed to N-ch open-drain output.	TXD0
P31	23				RXD0
P32	24				ASCK0
P33	25				TIP00/TOP00
P34	26				TIP01/TOP01
P35	27				TI010/TO01
P38	55		No[Note 1]		SDA0[Note 2]
P39	56				SCL0[Note 2]
P40	19	I/O	Yes	Port 4 I/O port Input/output can be specified in 1-bit units. P41 and P42 can be specified as N-ch open-drain output in 1-bit units.	SI00
P41	20				SO00
P42	21				$\overline{SCK00}$

Notes 1. An on-chip pull-up resistor can be provided by a mask option (only in the μPD703207, 703207Y).

2. Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

(2/2)

Pin Name	Pin No.	I/O	Pull-up Resistor	Function	Alternate Function
P50	28	I/O	Yes	Port 5 I/O port Input/output can be specified in 1-bit units.	TI011/RTP00/KR0
P51	29				TI50/RTP01/KR1
P52	30				TO50/RTP02/KR2
P53	31				RTP03/KR3
P54	34				RTP04/KR4
P55	35				RTP05/KR5
P70	64	Input	No	Port 7 Input port	ANI0
P71	63				ANI1
P72	62				ANI2
P73	61				ANI3
P74	60				ANI4
P75	59				ANI5
P76	58				ANI6
P77	57				ANI7
P90	36	I/O	Yes	Port 9 I/O port Input/output can be specified in 1-bit units. P98 and P99 can be specified as N-ch open-drain output in 1-bit units.	TXD1/KR6
P91	37				RXD1/KR7
P96	38				TI51/TO51
P97	39				SI01
P98	40				SO01
P99	41				$\overline{\text{SCK01}}$
P913	42				INTP4
P914	43				INTP5
P915	44				INTP6
PCM0	45	I/O	No	Port CM I/O port Input/output can be specified in 1-bit units.	–
PCM1	46				CLKOUT
PDL0	47	I/O	No	Port DL I/O port Input/output can be specified in 1-bit units.	–
PDL1	48				–
PDL2	49				–
PDL3	50				–
PDL4	51				–
PDL5	52				FLMD1[Note]
PDL6	53				–
PDL7	54				–

Note Only in the μPD70F3207H, 70F3207HY

Downloaded from Arrow.com.

(2) Non-port pins

(1/2)

Pin Name	Pin No.	I/O	Pull-up Resistor	Function	Alternate Function
ANI0	64	Input	No	Analog voltage input for A/D converter	P70
ANI1	63				P71
ANI2	62				P72
ANI3	61				P73
ANI4	60				P74
ANI5	59				P75
ANI6	58				P76
ANI7	57				P77
ASCK0	24	Input	Yes	UART0 serial clock input	P32
AV_{REF0}	1	–	–	Reference voltage for A/D converter and positive power supply for alternate-function ports	–
AV_{SS}	2	–	–	Ground potential for A/D converter	–
CLKOUT	46	Output	No	Internal system clock output	PCM1
EV_{DD}	33	–	–	Positive power supply for external	–
EV_{SS}	32	–	–	Ground potential for external	–
FLMD0[Note 1]	3	–	–	Flash programming mode setting pin	–
FLMD1[Note 1]	52				PDL5
IC[Note 2]	8	–	–	Internally connected	–
INTP0	15	Input	Yes	External interrupt request input (maskable, analog noise elimination)	P03
INTP1	16				P04
INTP2	17				P05
INTP3	18				P06
INTP4	42				P913
INTP5	43				P914
INTP6	44				P915
KR0	28	Input	Yes	Key return input	P50/TI011/RTP00
KR1	29				P51/TI50/RTP01
KR2	30				P52/TO50/RTP02
KR3	31				P53/RTP03
KR4	34				P54/RTP04
KR5	35				P55/RTP05
KR6	36				P90/TXD1
KR7	37				P91/RXD1
NC	5	–	–	Not internally connected. Leave open.	–
NMI	14	Input	Yes	External interrupt input (non-maskable, analog noise elimination)	P02
$\overline{RESET}$	9	Input	–	System reset input	–

Notes 1. Only in the μPD70F3207H, 70F3207HY

2. Only in the μPD703207, 703207Y

Downloaded from Arrow.com.

(2/2)

Pin Name	Pin No.	I/O	Pull-up Resistor	Function	Alternate Function
RTP00	28	Output	Yes	Real-time output port	P50/TI011/KR0
RTP01	29				P51/TI50/KR1
RTP02	30				P52/TO50/KR2
RTP03	31				P53/KR3
RTP04	34				P54/KR4
RTP05	35				P55/KR5
RXD0	23	Input	Yes	Serial receive data input for UART0	P31
RXD1	37			Serial receive data input for UART1	P91/KR7
$\overline{SCK00}$	21	I/O	Yes	Serial clock I/O for CSI00 and CSI01 N-ch open-drain output can be specified in 1-bit units.	P42
$\overline{SCK01}$	41				P99
SCL0 Note 1	56	I/O	No Note 2	Serial clock I/O for I^2C0 Fixed to N-ch open-drain output	P39
SDA0 Note 1	55	I/O	No Note 2	Serial transmit/receive data I/O for I^2C0 Fixed to N-ch open-drain output	P38
SI00	19	Input	Yes	Serial receive data input for CSI00	P40
SI01	39			Serial receive data input for CSI01	P97
SO00	20	Output	Yes	Serial transmit data output for CSI00 and CSI01 N-ch open-drain output can be specified in 1-bit units.	P41
SO01	40				P98
TI010	27	Input	Yes	Capture trigger input/external event input for TM01	P35/TO01
TI011	28			Capture trigger input for TM01	P50/RTP00/KR0
TI50	29			External event input for TM50	P51/RTP01/KR1
TI51	38			External event input for TM51	P96/TO51
TIP00	25			Capture trigger input/external event input/external clock input for TMP0	P33/TOP00
TIP01	26			Capture trigger input	P34/TOP01
TO01	27	Output	Yes	Timer output for TM01	P35/TI010
TO50	30			Timer output for TM50	P52/RTP02/KR2
TO51	38			Timer output for TM51	P96/TI51
TOH0	12			Timer output for TMH0	P00
TOH1	13			Timer output for TMH1	P01
TOP00	25			Timer output for TMP0	P33/TIP00
TOP01	26				P34/TIP01
TXD0	22	Output	Yes	Serial transmit data output for UART0	P30
TXD1	36			Serial transmit data output for UART1	P90/KR6
V_{DD}	4	–	–	Positive power supply pin for internal	–
V_{SS}	6	–	–	Ground potential for internal	–
X1	7	Input	No	Connecting resonator for main clock	–
X2	8	–	No		–
XT1	10	Input	No	Connecting resonator for subclock	–
XT2	11	–	No		–

Notes 1. Only in the μPD703207Y, 70F3207HY

2. An on-chip pull-up resistor can be provided by a mask option (only in the μPD703207, 703207Y).

Downloaded from Arrow.com.

2.2 Pin I/O Circuits and Recommended Connection of Unused Pins

(1/2)

Pin	Alternate Function	Pin No.	I/O Circuit Type	Recommended Connection
P00	TOH0	12	5-A	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P01	TOH1	13		
P02	NMI	14	5-W	
P03 to P06	INTP0 to INTP3	15 to 18		
P30	TXD0	22	5-A	
P31	RXD0	23	5-W	
P32	ASCK0	24		
P33	TIP00/TOP00	25		
P34	TIP01/TOP01	26		
P35	TI010/TO01	27		
<R> P38	SDA0[Note 1]	55	13-AE[Note 2]	
			13-AD[Note 3]	
<R> P39	SCL0[Note 1]	56	13-AE[Note 2]	
			13-AD[Note 3]	
P40	SI00	19	5-W	
P41	SO00	20	10-E	
P42	$\overline{SCK00}$	21	10-F	
P50	TI011/RTP00/KR0	28	8-A	
P51	TI50/RTP01/KR1	29		
P52	TO50/RTP02/KR2	30		
P53	RTP03/KR3	31		
P54	RTP04/KR4	34	10-A	
P55	RTP05/KR5	35		
P70 to P77	ANI0 to ANI7	64 to 57	9-C	Connect to AV_{REF0} or AV_{SS}.
P90	TXD1/KR6	36	8-A	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P91	RXD1/KR7	37		
P96	TI51/TO51	38		
P97	SI01	39	5-W	
P98	SO01	40	10-E	
P99	$\overline{SCK01}$	41	10-F	
P913 to P915	INTP4 to INTP6	42 to 44	5-W	
PCM0	–	45	5	
PCM1	CLKOUT	46		
PDL0 to PDL4	–	47 to 51		
PDL5	FLMD1[Note 3]	52		
PDL6, PDL7	–	53, 54		
AV_{REF0}	–	1	–	Directly connect to V_{DD}.
AV_{SS}	–	2	–	–

Notes 1. Only in the μPD703207Y, 70F3207HY

2. Only in the μPD703207, 703207Y

3. Only in the μPD70F3207H, 70F3207HY

Downloaded from Arrow.com.

(2/2)

Pin	Alternate Function	Pin No.	I/O Circuit Type	Recommended Connection
EV_{DD}	–	33	–	–
EV_{SS}	–	32	–	–
FLMD0[Note 1]	–	3	–	Connect to V_{SS} in normal operation mode.
IC[Note 2]	–	3	–	Directly connect to EV_{SS} or V_{SS} or pull down with a 10 kΩ resistor.
NC	–	5	–	Leave open.
$\overline{RESET}$	–	9	2	–
V_{DD}	–	4	–	–
V_{SS}	–	6	–	–
X1	–	7	–	–
X2	–	8	–	–
XT1	–	10	16	Directly connect to V_{SS}[Note 3].
XT2	–	11	16	Leave open.

Notes 1. Only in the μPD70F3207H, 70F3207HY

2. Only in the μPD703207, 703207Y

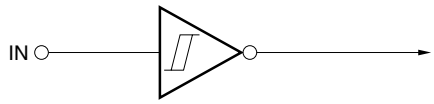
3. Be sure to set the PSMR.XTSTP bit to 1 when this pin is not used.

Downloaded from Arrow.com.

2.3 Pin I/O Circuits

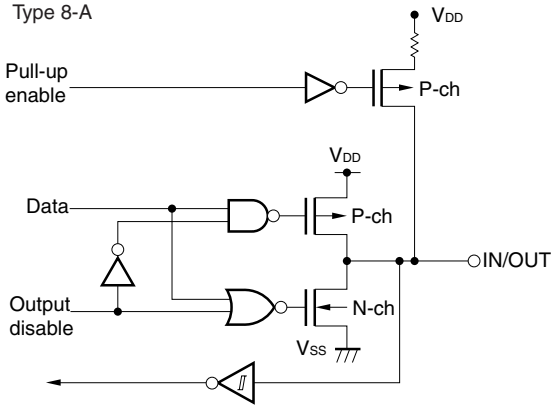
(1/2)

Type 2

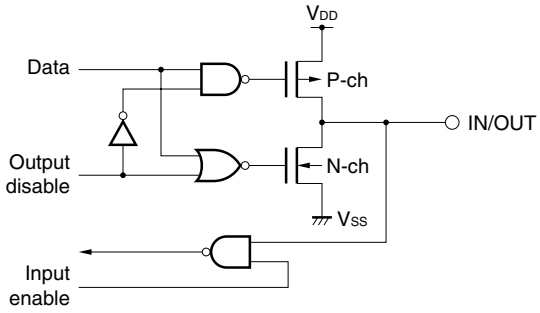


Schmitt-triggered input with hysteresis characteristics

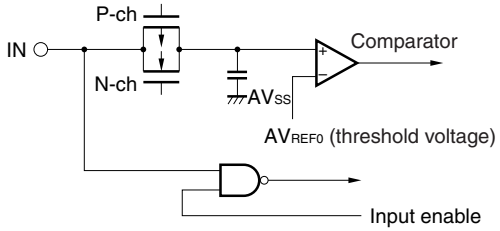
Type 8-A



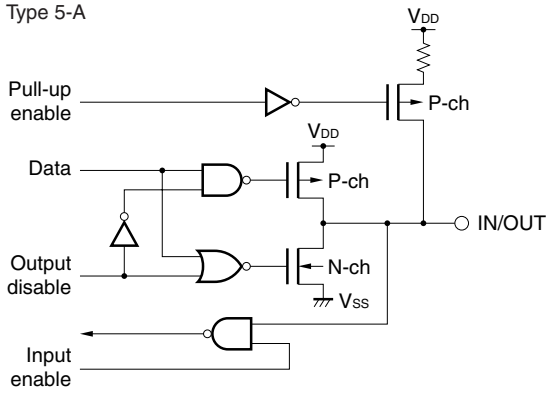
Type 5



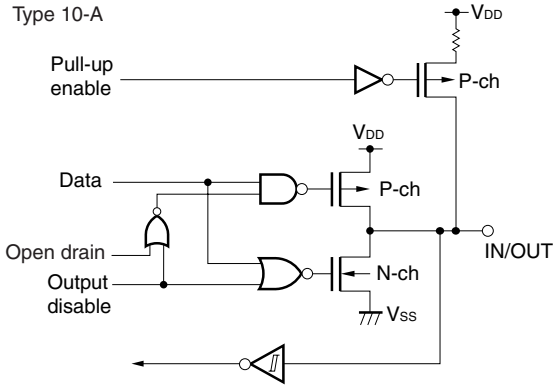
Type 9-C



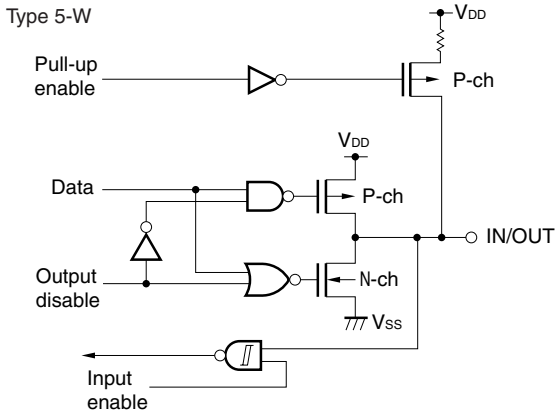
Type 5-A



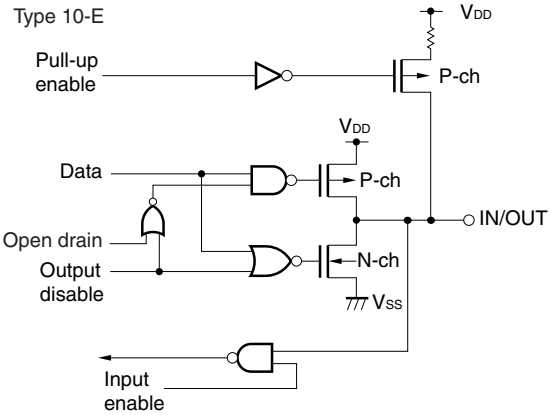
Type 10-A



Type 5-W



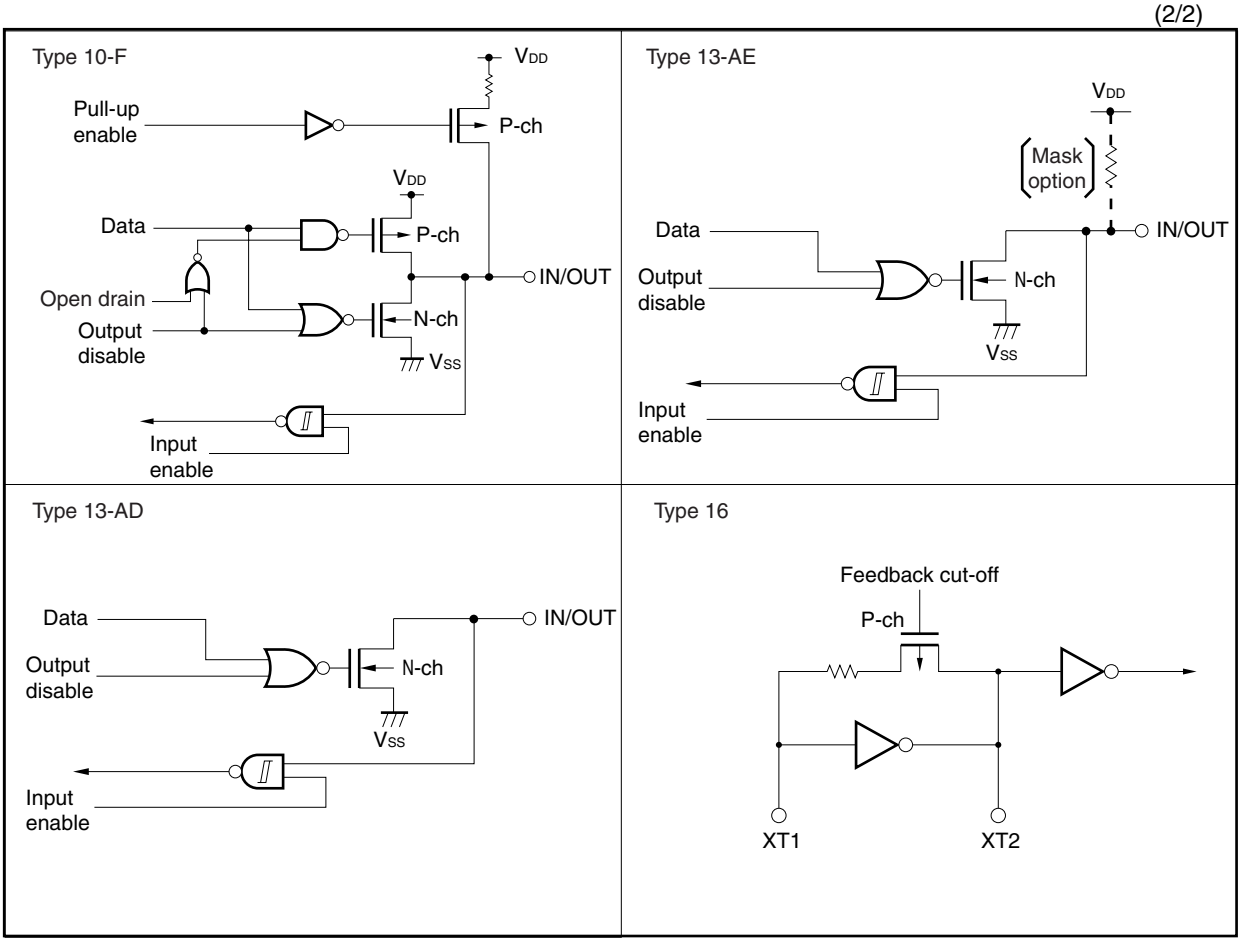
Type 10-E



Downloaded from Arrow.com.

(2/2)

<R>



Remark Read V_{DD} as EV_{DD}. Also, read V_{SS} as EV_{SS}.

Downloaded from Arrow.com.

CHAPTER 3 CPU FUNCTIONS

The CPU of the V850ES/KE1 is based on the RISC architecture and executes most instructions in one clock cycle by using 5-stage pipeline control.

3.1 Features

- Number of instructions: 83
- Minimum instruction execution time: 50.0 ns (@ 20 MHz operation: 4.5 to 5.5 V)
<R> 100 ns (@ 10 MHz operation: 2.7 to 5.5 V)
- Memory space Program (physical address) space: 64 MB linear
  Data (logical address) space: 4 GB linear
- General-purpose registers: 32 bits × 32
- Internal 32-bit architecture
- 5-stage pipeline control
- Multiply/divide instructions
- Saturated operation instructions
- 32-bit shift instruction: 1 clock
- Load/store instruction with long/short format
- Four types of bit manipulation instructions
  - SET1
  - CLR1
  - NOT1
  - TST1

Downloaded from Arrow.com.

3.2 CPU Register Set

The CPU registers of the V850ES/KE1 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers have 32-bit width.

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

(1) Program register set

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

31		0
PC	(Program counter)	

(2) System register set

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

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

ECR	(Interrupt source register)

PSW	(Program status word)

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

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

CTBP	(CALLT base pointer)

Downloaded from Arrow.com.

3.2.1 Program register set

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

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

Thirty-two general-purpose registers, r0 to r31, are available. All 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 always holds 0 and is used for operations that use 0 and offset 0 addressing. r30 is used as a base pointer when performing memory access with the SLD and SST instructions.

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, and they must be restored to the registers after the registers have been used. There are cases when r2 is used by the real-time OS. If r2 is not used by the real-time OS, r2 can be used as a variable register.

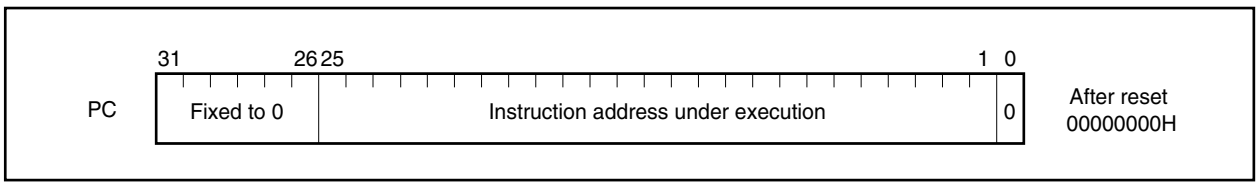
Table 3-1. Program Registers

Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating 32-bit immediate
r2	Address/data variable register (when r2 is not used by the real-time OS to be used)	
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	Used to access global variable in data area
r5	Text pointer	Register to indicate the start of the text area (area for placing program code)
r6 to r29	Address/data variable register	
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 (PC)

This register holds the address of the instruction under 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 bit 26, it is ignored.

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



Downloaded from Arrow.com.

3.2.2 System register set

System registers control the status of the CPU and hold interrupt information.

Read from and write to system registers are performed by setting the system register numbers shown below with the system register load/store instructions (LDSR, STSR instructions).

Table 3-2. System Register Numbers

System Register No.	System Register Name	Operand Specification Enabled	
		LDSR Instruction	STSR Instruction
0	Interrupt status saving register (EIPC)[Note 1]	Yes	Yes
1	Interrupt status saving register (EIPSW)[Note 1]	Yes	Yes
2	NMI status saving register (FEPC)[Note 1]	Yes	Yes
3	NMI status saving register (FEPSW)[Note 1]	Yes	Yes
4	Interrupt source register (ECR)	No	Yes
5	Program status word (PSW)	Yes	Yes
6 to 15	Reserved numbers for future function expansion (The operation is not guaranteed if accessed.)	No	No
16	CALLT execution status saving register (CTPC)	Yes	Yes
17	CALLT execution status saving register (CTPSW)	Yes	Yes
<R> 18	Exception/debug trap status saving register (DBPC)	Yes[Note 2]	Yes[Note 2]
<R> 19	Exception/debug trap status saving register (DBPSW)	Yes[Note 2]	Yes[Note 2]
20	CALLT base pointer (CTBP)	Yes	Yes
21 to 31	Reserved numbers for future function expansion (The operation is not guaranteed if accessed.)	No	No

Notes 1. Since only one set of these registers is available, the contents of this register must be saved by the program when multiple interrupt servicing is enabled.

<R> **2.** These registers can be accessed only during the interval between the execution of the DBTRAP instruction or illegal opcode and the DBRET instruction.

Caution Even if bit 0 of EIPC, FEPC, or CTPC is set (1) by the LDSR instruction, bit 0 is ignored during return with the RETI instruction following interrupt servicing (because bit 0 of PC is fixed to 0). When setting a value to EIPC, FEPC, and CTPC, set an even number (bit 0 = 0).

Downloaded from Arrow.com.

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

There are two interrupt status saving registers, EIPC and EIPSW.

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

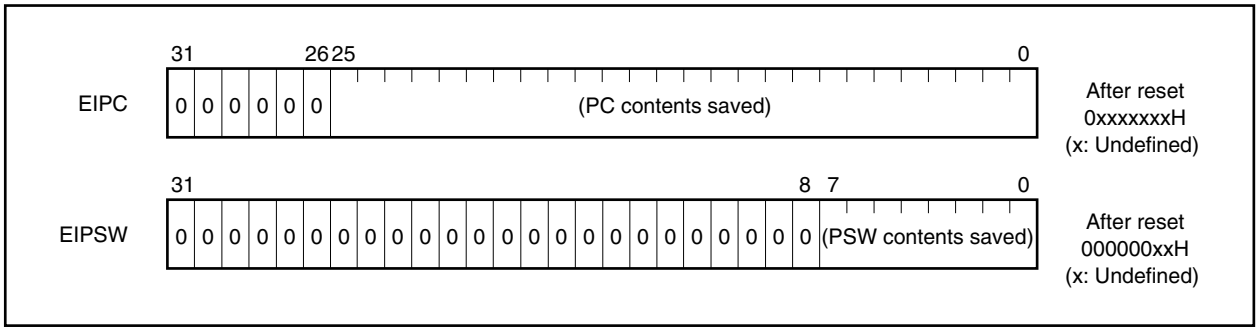
The address of the next instruction following the instruction executed when a software exception or maskable interrupt occurs is saved to EIPC, except for some instructions (refer to **17.9 Periods in Which Interrupts Are Not Acknowledged by CPU**).

The current PSW contents are saved to EIPSW.

Since there is only one set of interrupt status saving registers, the contents of these registers must be saved by the program when multiple interrupt servicing is enabled.

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

When the RETI instruction is executed, the values in EIPC and EIPSW are restored to the PC and PSW, respectively.



Downloaded from Arrow.com.

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

There are two NMI status saving registers, FEPC and FEPSW.

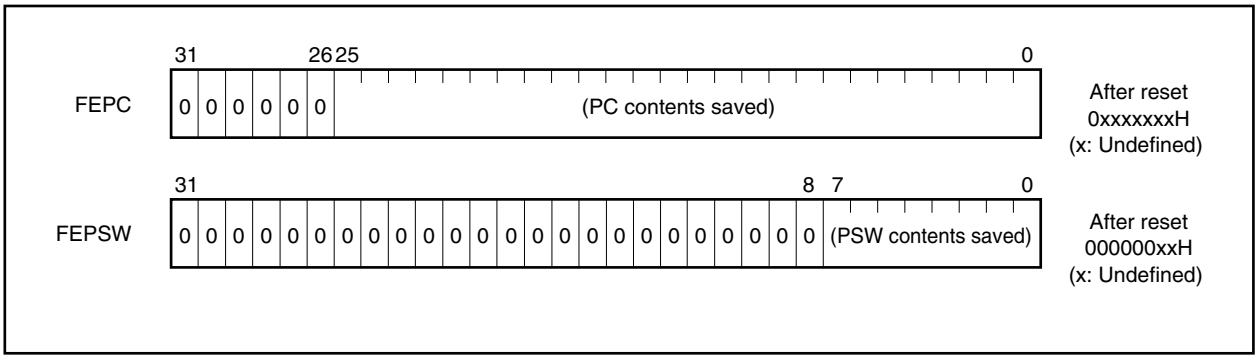
Upon occurrence of a non-maskable interrupt (NMI), the contents of the program counter (PC) are saved to FEPC and the contents of the program status word (PSW) are saved to FEPSW.

The address of the next instruction following the instruction executed when a non-maskable interrupt occurs is saved to FEPC, except for some instructions.

The current PSW contents are saved to FEPSW.

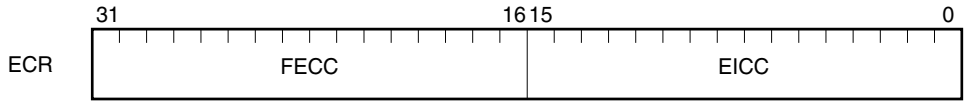
Since there is only one set of NMI status saving registers, the contents of these registers must be saved by the program when multiple interrupt servicing is performed.

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



(3) Interrupt source register (ECR)

Upon occurrence of an interrupt or an exception, the interrupt source register (ECR) holds the source of an interrupt or an exception. The value held by ECR is the exception code coded for each interrupt source. This register is a read-only register, and thus data cannot be written to it using the LDSR instruction.



Bit position	Bit name	Description
31 to 16	FECC	Non-maskable interrupt (NMI) exception code
15 to 0	EICC	Exception, maskable interrupt exception code

Downloaded from Arrow.com.

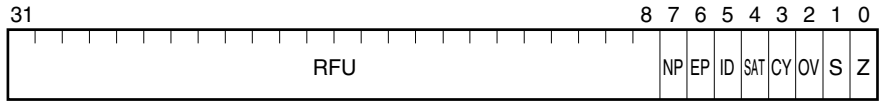
(4) Program status word (PSW)

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

When the contents of this register are changed using the LDSR instruction, the new contents become valid immediately following completion of LDSR instruction execution. Interrupt request acknowledgment is held pending while a write to the PSW is being executed by the LDSR instruction.

Bits 31 to 8 are reserved (fixed to 0) for future function expansion.

(1/2)



Bit position	Flag name	Description
31 to 8	RFU	Reserved field. Fixed to 0.
7	NP	Indicates that non-maskable interrupt (NMI) servicing is in progress. This flag is set to 1 when an NMI request is acknowledged, and disables multiple interrupts. 0: NMI servicing not in progress 1: NMI servicing in progress
6	EP	Indicates that exception processing is in progress. This flag is set to 1 when an exception occurs. Moreover, interrupt requests can be acknowledged even when this bit is set. 0: Exception processing not in progress 1: Exception processing in progress
5	ID	Indicates whether maskable interrupt request acknowledgment is enabled. 0: Interrupt enabled 1: Interrupt disabled
4	SAT[Note]	Indicates that the result of executing a saturated operation instruction has overflowed and that the calculation result is saturated. Since this is a cumulative flag, it is set to 1 when the result of a saturated operation instruction becomes saturated, and it is not cleared to 0 even if the operation results of successive instructions do not become saturated. This flag is neither set nor cleared when arithmetic operation instructions are executed. 0: Not saturated 1: Saturated
3	CY	Indicates whether carry or borrow occurred as the result of an operation. 0: No carry or borrow occurred 1: Carry or borrow occurred
2	OV[Note]	Indicates whether overflow occurred during an operation. 0: No overflow occurred 1: Overflow occurred.
1	S[Note]	Indicates whether the result of an operation is negative. 0: Operation result is positive or 0. 1: Operation result is negative.
0	Z	Indicates whether operation result is 0. 0: Operation result is not 0. 1: Operation result is 0.

Remark **Note** is explained on the following page.

Downloaded from Arrow.com.

(2/2)

Note During saturated operation, the saturated operation results are determined by the contents of the OV flag and S flag. The SAT flag is set (to 1) only when the OV flag is set (to 1) during saturated operation.

Operation result status	Flag status			Saturated operation result
	SAT	OV	S	
Maximum positive value exceeded	1	1	0	7FFFFFFFH
Maximum negative value exceeded	1	1	1	80000000H
Positive (maximum value not exceeded)	Holds value before operation	0	0	Actual operation result
Negative (maximum value not exceeded)			1	

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

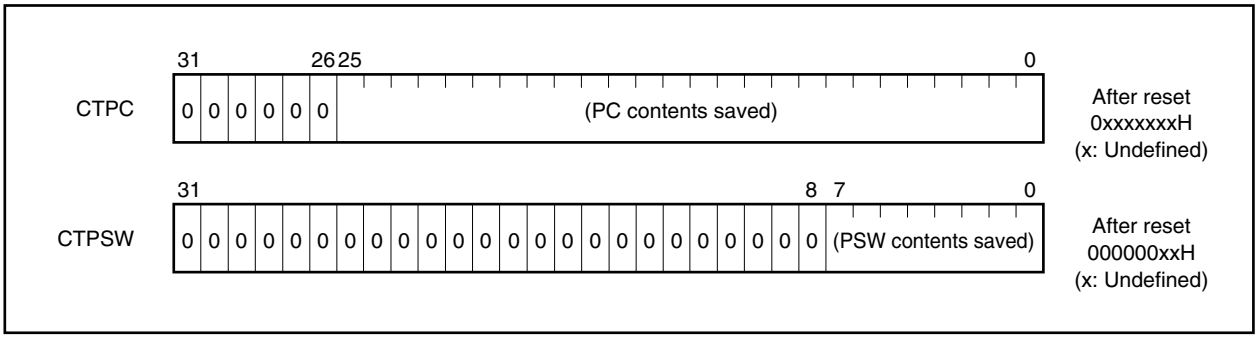
There are two CALLT execution status saving registers, CTPC and CTPSW.

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

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

The current PSW contents are saved to CTPSW.

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



Downloaded from Arrow.com.

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

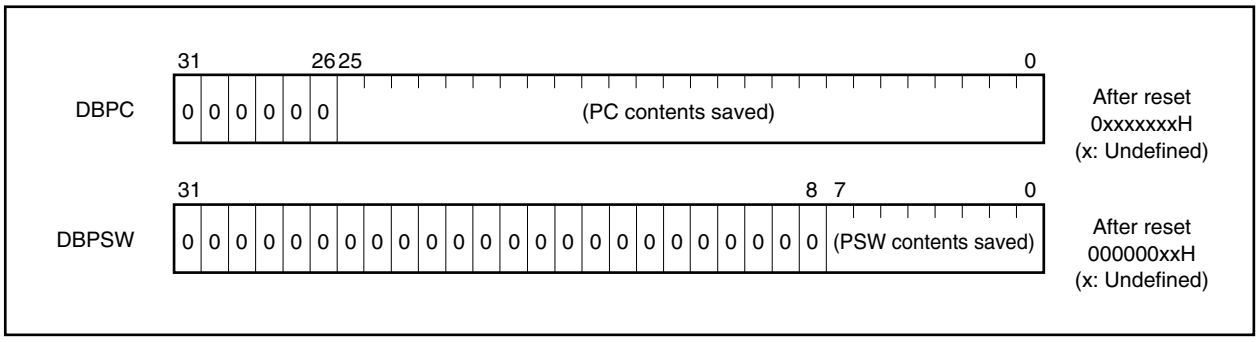
There are two exception/debug trap status saving registers, DBPC and DBPSW.

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

The contents saved to DBPC consist of the address of the next instruction after the instruction executed when an exception trap or debug trap occurs.

The current PSW contents are saved to DBPSW.

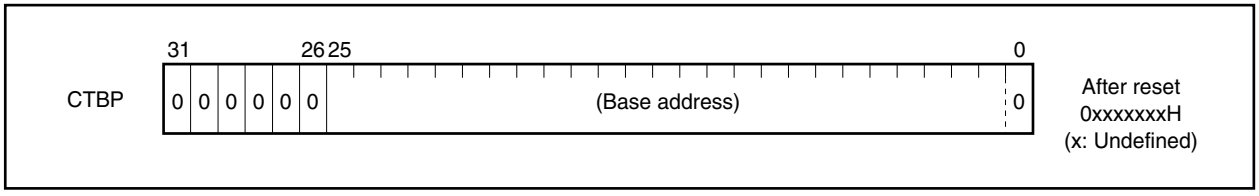
Bits 31 to 26 of DBPC and bits 31 to 8 of DBPSW are reserved (fixed to 0) for future function expansion.



(7) CALLT base pointer (CTBP)

The CALLT base pointer (CTBP) is used to specify table addresses and generate target addresses (bit 0 is fixed to 0).

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



Downloaded from Arrow.com.

3.3 Operating Modes

The V850ES/KE1 has the following operating modes.

(1) Normal operating mode

After the system has been released from the reset state, execution branches to the reset entry address of the internal ROM, and instruction processing is started.

(2) Flash memory programming mode

This mode is valid only in flash memory versions (μPD70F3207H and 70F3207HY).

When this mode is specified, the internal flash memory can be programmed by using a flash programmer.

(a) Specifying operating mode

The operating mode is specified according to the status (input level) of the FLMD0 and FLMD1 pins.

In the normal operating mode, input a low level to the FLMD0 pin during the reset period.

A high level is input to the FLMD0 pin by the flash programmer in the flash memory programming mode if a flash programmer is connected. In the self-programming mode, input a high level to this pin from an external circuit.

Fix the specification of these pins in the application system and do not change the setting of these pins during operation.

FLMD0	FLMD1	Operating Mode
L	×	Normal operating mode
H	L	Flash memory programming mode
H	H	Setting prohibited

Remark H: High level
L: Low level
×: don't care

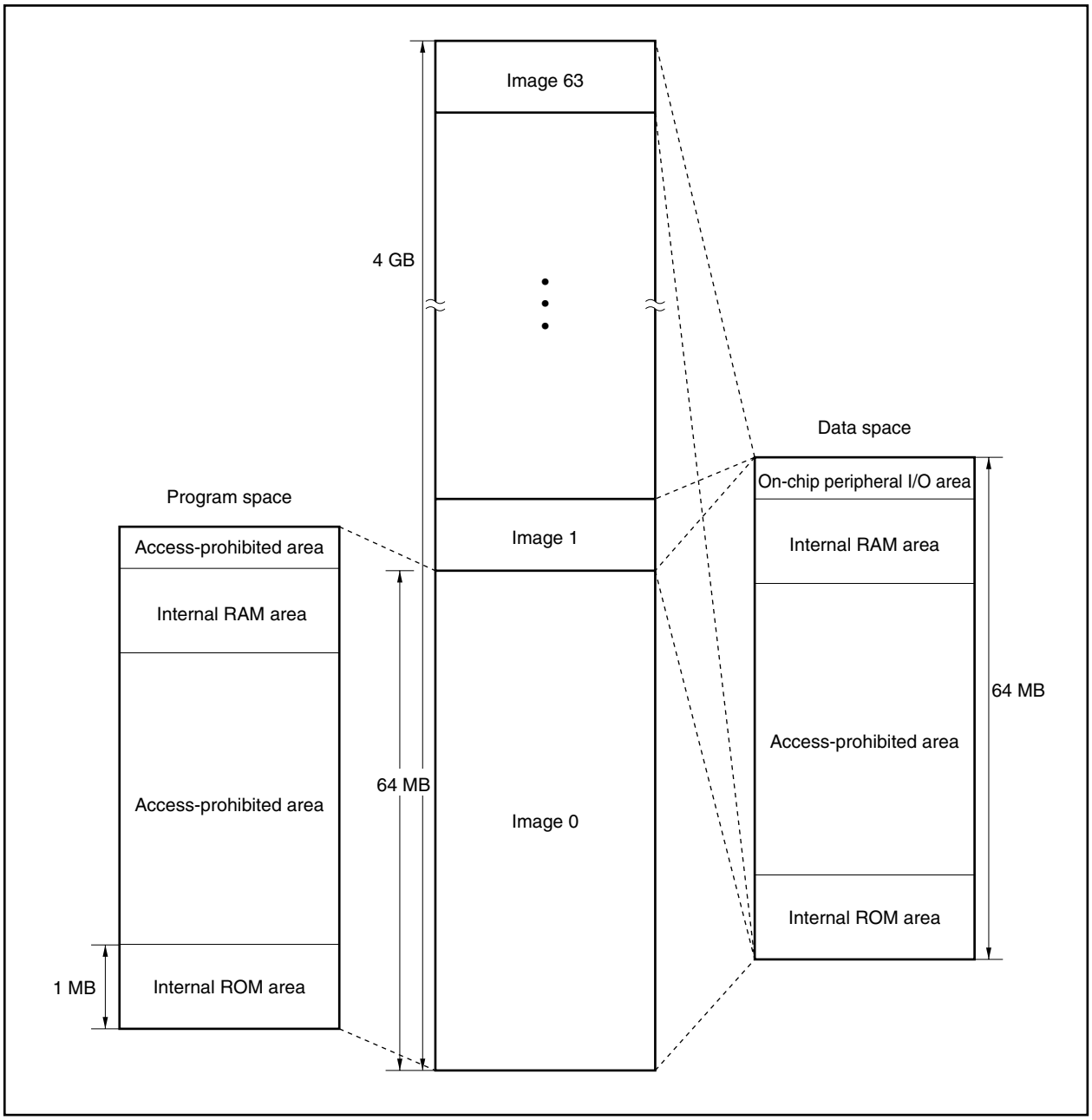
Downloaded from Arrow.com.

3.4 Address Space

3.4.1 CPU address space

For instruction addressing, an internal ROM area of up to 1 MB, and an internal RAM area are supported in a linear address space (program space) of up to 64 MB. For operand addressing (data access), up to 4 GB of a linear address space (data space) is supported. The 4 GB address space, however, is viewed as 64 images of a 64 MB physical address space. This means that the same 64 MB physical address space is accessed regardless of the value of bits 31 to 26.

Figure 3-1. Address Space Image



Downloaded from Arrow.com.

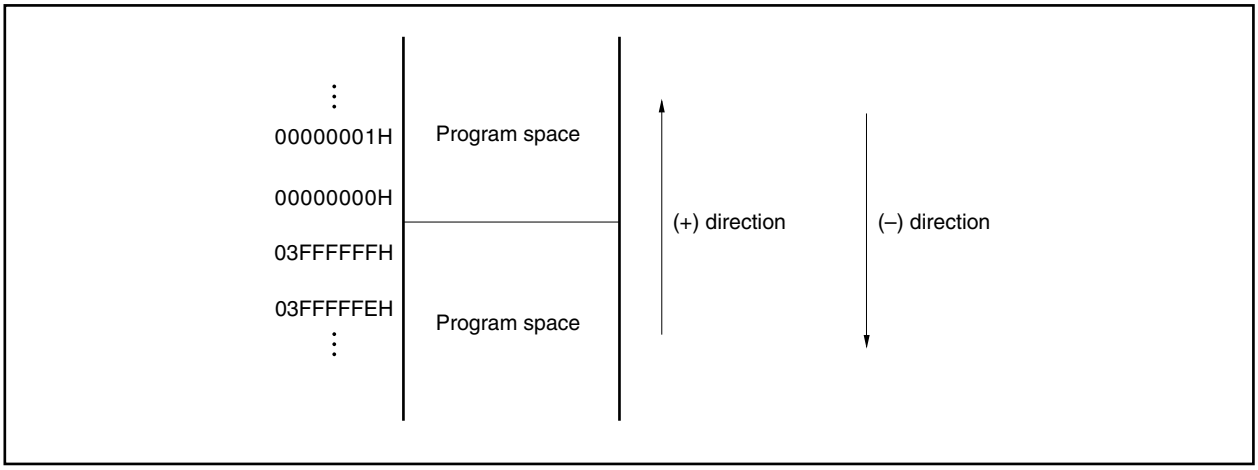
3.4.2 Wraparound of CPU address space

(1) Program space

Of the 32 bits of the program counter (PC), the higher 6 bits are fixed to 0 and only the lower 26 bits are valid. Even if a carry or borrow occurs from bit 25 to bit 26 as a result of branch address calculation, the higher 6 bits ignore this and remain 0.

Therefore, the lower-limit address of the program space, 00000000H, and the upper-limit address, 03FFFFFFH, are contiguous addresses, and the program space is wrapped around at the boundary of these addresses.

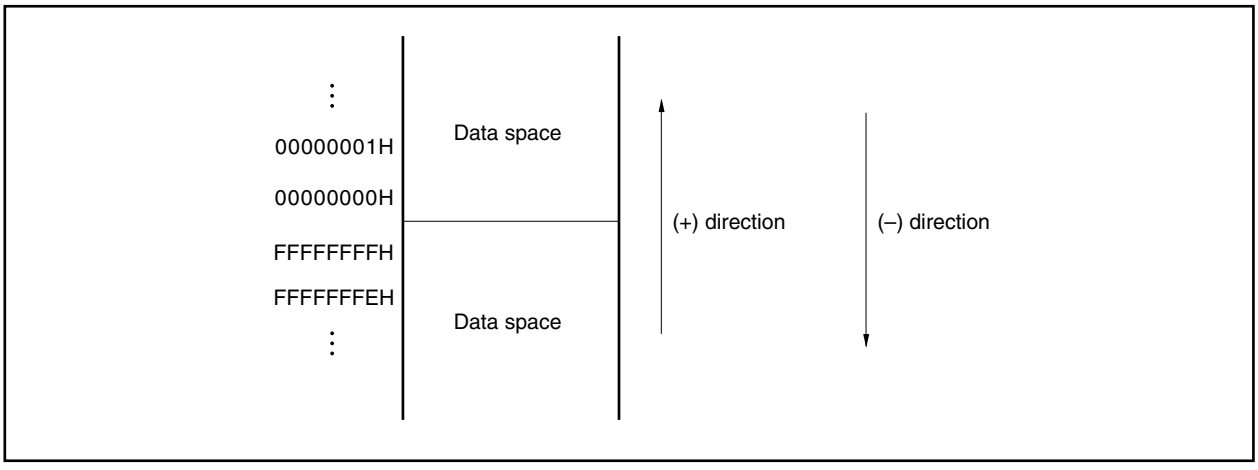
Caution No instructions can be fetched from the 4 KB area of 03FFF000H to 03FFFFFFH because this area is an on-chip peripheral I/O area. Therefore, do not execute any branch operation instructions in which the destination address will reside in any part of this area.



(2) Data space

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

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

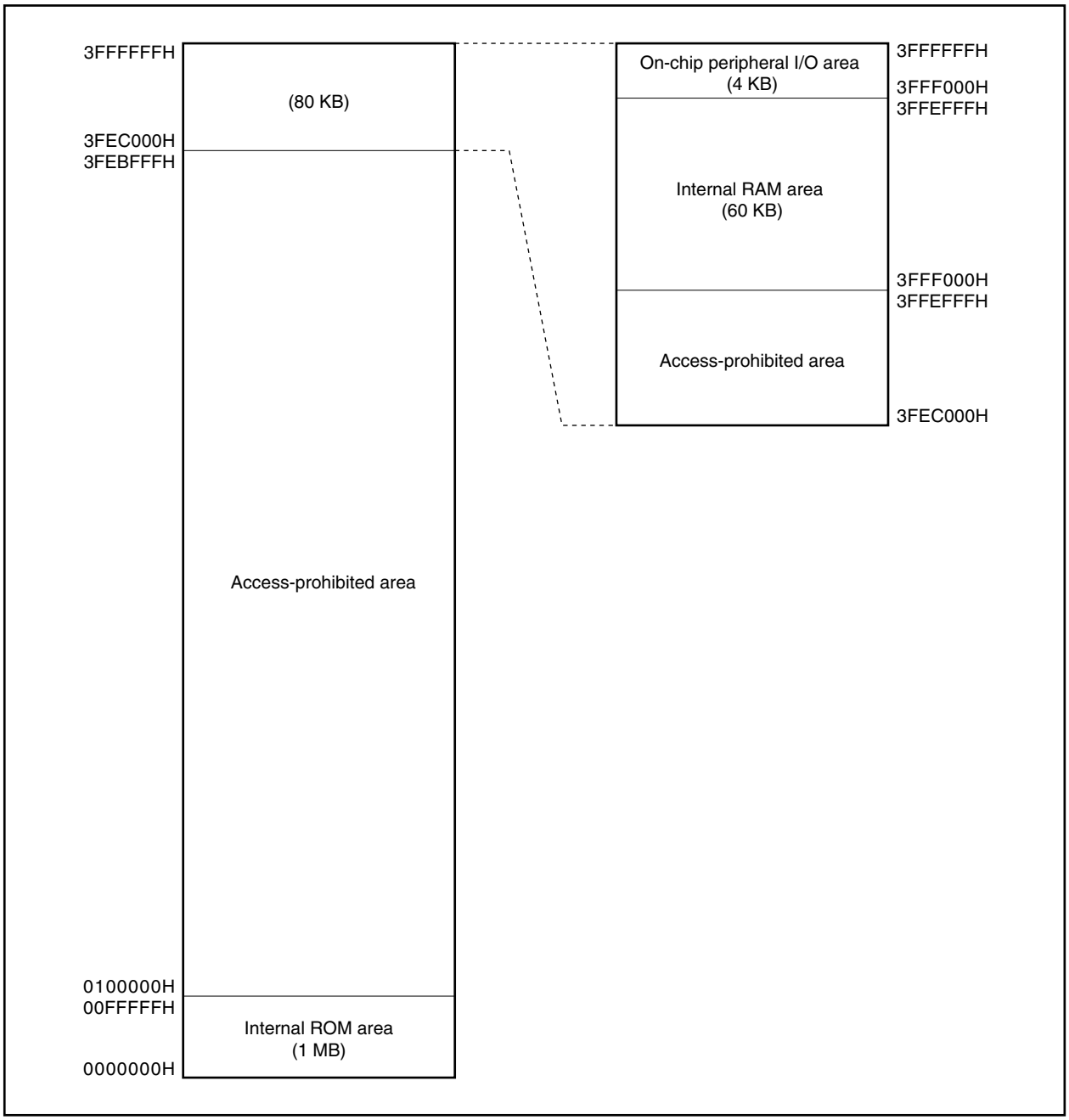


Downloaded from Arrow.com.

3.4.3 Memory map

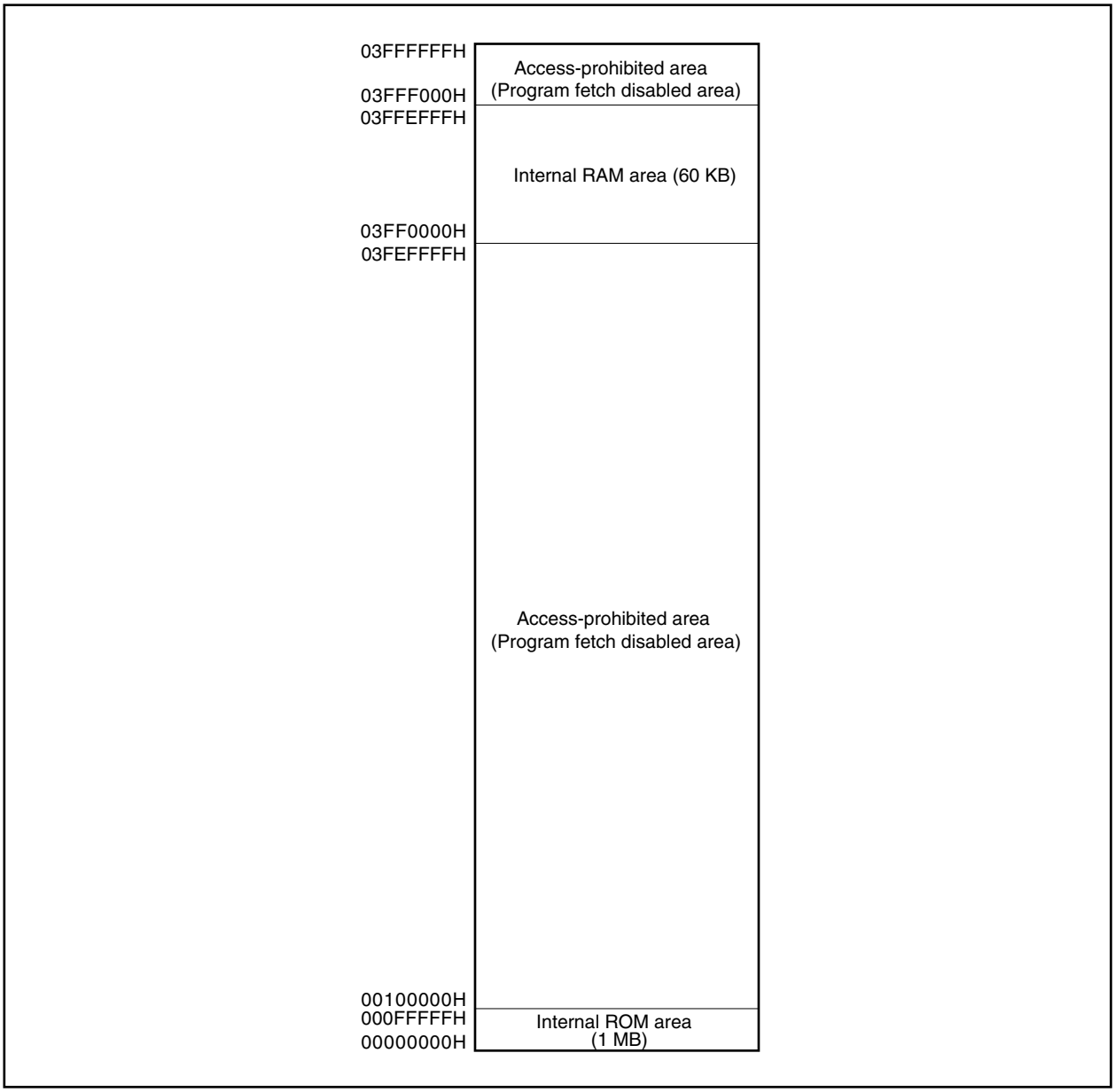
The V850ES/KE1 has reserved areas as shown below.

Figure 3-2. Data Memory Map (Physical Addresses)



Downloaded from Arrow.com.

Figure 3-3. Program Memory Map



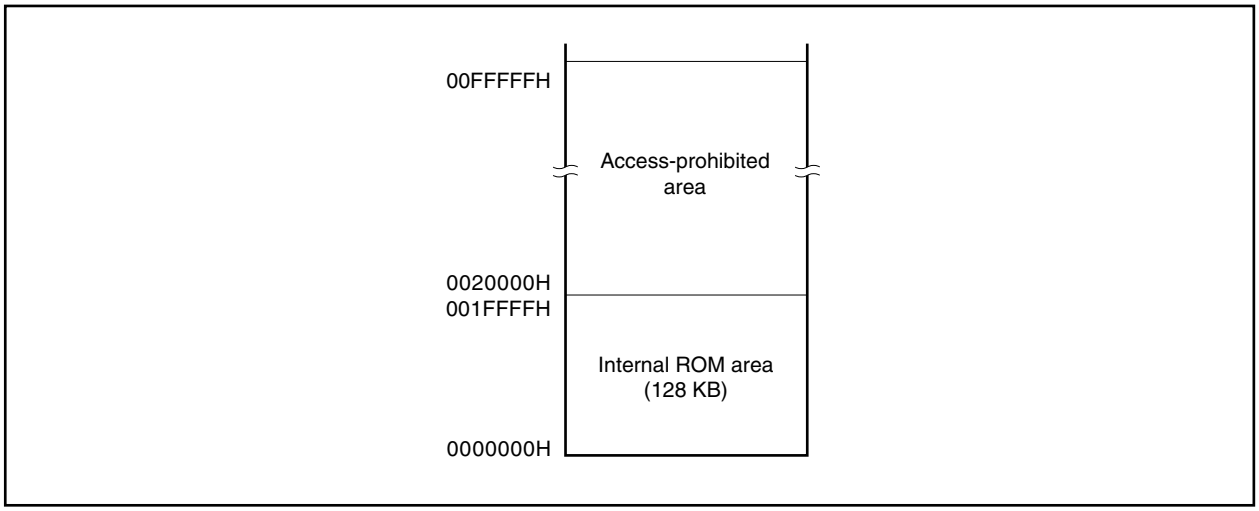
Downloaded from Arrow.com.

3.4.4 Areas

(1) Internal ROM area

An area of 1 MB from 0000000H to 00FFFFFH is reserved for the internal ROM area.
A 128 KB area from 0000000H to 001FFFFH is provided in the V850ES/KE1.
Addresses 0020000H to 00FFFFFH are an access-prohibited area.

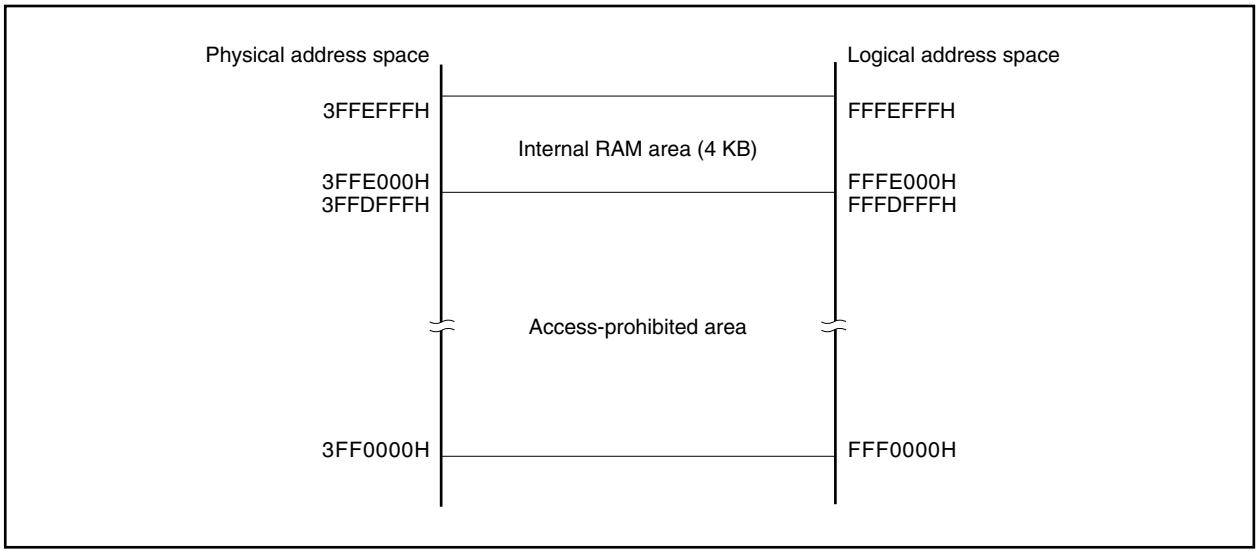
Figure 3-4. Internal ROM Area (128 KB)



(2) Internal RAM area

An area of 60 KB maximum from 3FF0000H to 3FFEFFFH is reserved for the internal RAM area.
A 4 KB area from 3FFE000H to 3FFEFFFH is provided as physical internal RAM in the V850ES/KE1.
Addresses 3FF0000H to 3FFDFFFH are an access-prohibited area.

Figure 3-5. Internal RAM Area (4 KB)

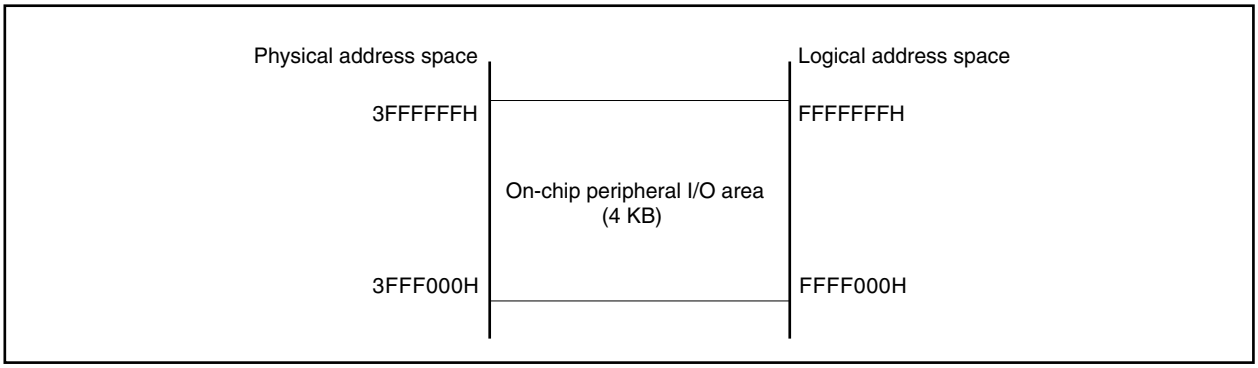


Downloaded from Arrow.com.

(3) On-chip peripheral I/O area

A 4 KB area from 3FFF000H to 3FFFFFFH is reserved as the on-chip peripheral I/O area.

Figure 3-6. On-Chip Peripheral I/O Area



Peripheral I/O registers assigned with functions such as on-chip peripheral I/O operation mode specification and state monitoring are mapped to the on-chip peripheral I/O area. Program fetches are not allowed in this area.

Cautions 1. If word access of a register is attempted, halfword access to the word area is performed twice, first for the lower bits, then for the higher bits, ignoring the lower 2 address bits.

2. If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits become undefined if the access is a read operation. If a write access is performed, only the data in the lower 8 bits is written to the register.

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.

(4) Number of clocks for access

The following table shows the number of base clocks required for accessing each resource.

Area (Bus Width) / Bus Cycle Type	Internal ROM (32 Bits)	Internal RAM (32 Bits)	On-Chip Peripheral I/O (16 Bits)
Instruction fetch (normal access)	1	1 Note 1	–
Instruction fetch (branch)	2	2 Note 1	–
Operand data access	3	1	3 Note 2

Notes 1. If the access conflicts with a data access, the number of clock is incremented by 1.

2. This value varies depending on the setting of the VSWC register.

Remark Unit: Clocks/access

Downloaded from Arrow.com.

3.4.5 Recommended use of address space

The architecture of the V850ES/KE1 requires that a register that serves as a pointer be secured for address generation when operand data in the data space is accessed. The address stored in this pointer ±32 KB can be directly accessed by an instruction for operand data. Because the number of general-purpose registers that can be used as a pointer is limited, however, by keeping the performance from dropping during address calculation when a pointer value is changed, as many general-purpose registers as possible can be secured for variables, and the program size can be reduced.

(1) Program space

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

To use the internal RAM area as the program space, access the addresses 3FFE000H to 3FFEFFFH (4 KB).

(2) Data space

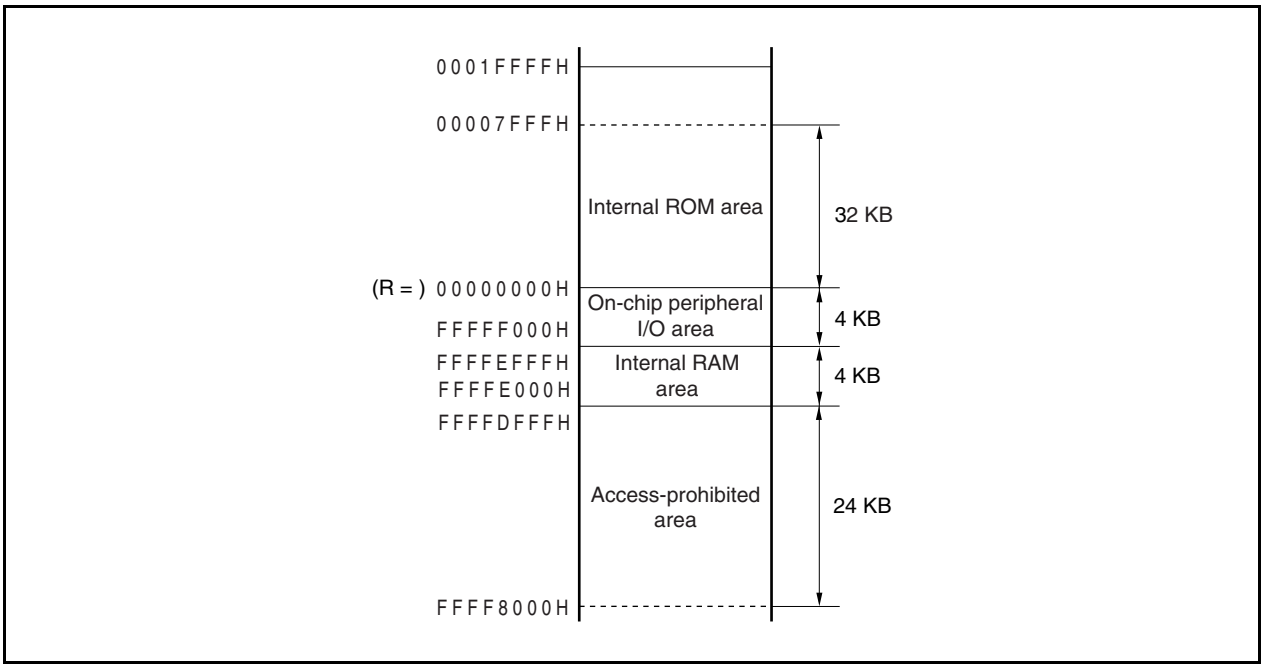
With the V850ES/KE1, it seems that there are sixty-four 64 MB address spaces on the 4 GB CPU address space. Therefore, the least significant bit (bit 25) of a 26-bit address is sign-extended to 32 bits and allocated as an address.

(a) Application example of wraparound

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

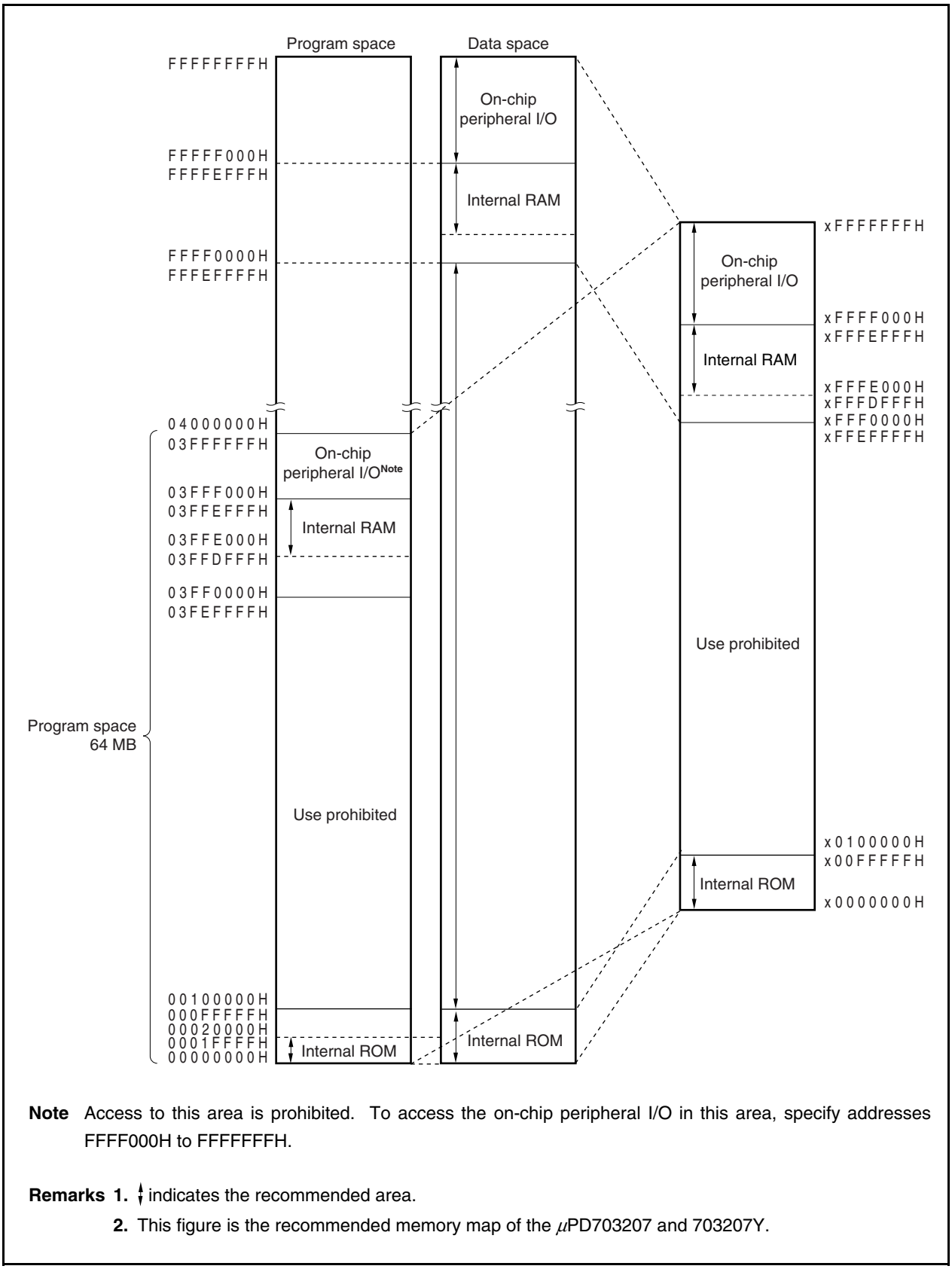
The zero register (r0) is a register fixed to 0 by hardware, and practically eliminates the need for registers dedicated to pointers.

Example: μPD703207, 703207Y



Downloaded from Arrow.com.

Figure 3-7. Recommended Memory Map



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

Remarks 1. ↕ indicates the recommended area.

2. This figure is the recommended memory map of the μPD703207 and 703207Y.

Downloaded from Arrow.com.

3.4.6 Peripheral I/O registers

(1/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit			After Reset
				1	8	16	
FFFFF004H	Port DL register	PDL	R/W	√	√		00H[Note 1]
FFFFF00CH	Port CM register	PCM	R/W	√	√		00H[Note 1]
FFFFF024H	Port DL mode register	PMDL	R/W	√	√		FFH
FFFFF02CH	Port CM mode register	PMCM	R/W	√	√		FFH
FFFFF04CH	Port CM mode control register	PMCCM	R/W	√	√		00H
FFFFF06EH	System wait control register	VSWC	R/W	√	√		77H
FFFFF100H	Interrupt mask register 0	IMR0	R/W			√	FFFFH
FFFFF100H	Interrupt mask register 0L	IMR0L	R/W	√	√		FFH
FFFFF101H	Interrupt mask register 0H	IMR0H	R/W	√	√		FFH
FFFFF102H	Interrupt mask register 1	IMR1	R/W			√	FFFFH
FFFFF102H	Interrupt mask register 1L	IMR1L	R/W	√	√		FFH
FFFFF103H	Interrupt mask register 1H	IMR1H	R/W	√	√		FFH
FFFFF106H	Interrupt mask register 3	IMR3	R/W			√	FFFFH
FFFFF106H	Interrupt mask register 3L	IMR3L	R/W	√	√		FFH
FFFFF110H	Interrupt control register	WDT1IC	R/W	√	√		47H
FFFFF112H	Interrupt control register	PIC0	R/W	√	√		47H
FFFFF114H	Interrupt control register	PIC1	R/W	√	√		47H
FFFFF116H	Interrupt control register	PIC2	R/W	√	√		47H
FFFFF118H	Interrupt control register	PIC3	R/W	√	√		47H
FFFFF11AH	Interrupt control register	PIC4	R/W	√	√		47H
FFFFF11CH	Interrupt control register	PIC5	R/W	√	√		47H
FFFFF11EH	Interrupt control register	PIC6	R/W	√	√		47H
FFFFF124H	Interrupt control register	TM0IC10	R/W	√	√		47H
FFFFF126H	Interrupt control register	TM0IC11	R/W	√	√		47H
FFFFF128H	Interrupt control register	TM5IC0	R/W	√	√		47H
FFFFF12AH	Interrupt control register	TM5IC1	R/W	√	√		47H
FFFFF12CH	Interrupt control register	CSI0IC0	R/W	√	√		47H
FFFFF12EH	Interrupt control register	CSI0IC1	R/W	√	√		47H
FFFFF130H	Interrupt control register	SREIC0	R/W	√	√		47H
FFFFF132H	Interrupt control register	SRIC0	R/W	√	√		47H
FFFFF134H	Interrupt control register	STIC0	R/W	√	√		47H
FFFFF136H	Interrupt control register	SREIC1	R/W	√	√		47H
FFFFF138H	Interrupt control register	SRIC1	R/W	√	√		47H
FFFFF13AH	Interrupt control register	STIC1	R/W	√	√		47H
FFFFF13CH	Interrupt control register	TMHIC0	R/W	√	√		47H
FFFFF13EH	Interrupt control register	TMHIC1	R/W	√	√		47H
FFFFF142H	Interrupt control register	IICIC0[Note 2]	R/W	√	√		47H
FFFFF144H	Interrupt control register	ADIC	R/W	√	√		47H
FFFFF146H	Interrupt control register	KRIC	R/W	√	√		47H

Notes 1. The output latch is 00H. When input, the pin status is read.

2. Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

(2/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit			After Reset
				1	8	16	
FFFFF148H	Interrupt control register	WTIIC	R/W	√	√		47H
FFFFF14AH	Interrupt control register	WTIC	R/W	√	√		47H
FFFFF14CH	Interrupt control register	BRGIC	R/W	√	√		47H
FFFFF174H	Interrupt control register	TP0OVIC	R/W	√	√		47H
FFFFF176H	Interrupt control register	TP0CCIC0	R/W	√	√		47H
FFFFF178H	Interrupt control register	TP0CCIC1	R/W	√	√		47H
FFFFF1FAH	In-service priority register	ISPR	R	√	√		00H
FFFFF1FCH	Command register	PRCMD	W		√		Undefined
FFFFF1FEH	Power save control register	PSC	R/W	√	√		00H
FFFFF200H	A/D converter mode register	ADM	R/W	√	√		00H
FFFFF201H	Analog input channel specification register	ADS	R/W		√		00H
FFFFF202H	Power fail comparison mode register	PFM	R/W	√	√		00H
FFFFF203H	Power fail comparison threshold register	PFT	R/W		√		00H
FFFFF204H	A/D conversion result register	ADCR	R			√	Undefined
FFFFF205H	A/D conversion result register H	ADCRH	R		√		Undefined
FFFFF300H	Key return mode register	KRM	R/W	√	√		00H
FFFFF400H	Port 0 register	P0	R/W	√	√		00H[Note]
FFFFF406H	Port 3 register	P3	R/W			√	0000H[Note]
FFFFF406H	Port 3 register L	P3L	R/W	√	√		00H[Note]
FFFFF407H	Port 3 register H	P3H	R/W	√	√		00H[Note]
FFFFF408H	Port 4 register	P4	R/W	√	√		00H[Note]
FFFFF40AH	Port 5 register	P5	R/W	√	√		00H[Note]
FFFFF40EH	Port 7 register	P7	R		√		Undefined
FFFFF412H	Port 9 register	P9	R/W			√	0000H[Note]
FFFFF412H	Port 9 register L	P9L	R/W	√	√		00H[Note]
FFFFF413H	Port 9 register H	P9H	R/W	√	√		00H[Note]
FFFFF420H	Port 0 mode register	PM0	R/W	√	√		FFH
FFFFF426H	Port 3 mode register	PM3	R/W			√	FFFFH
FFFFF426H	Port 3 mode register L	PM3L	R/W	√	√		FFH
FFFFF427H	Port 3 mode register H	PM3H	R/W	√	√		FFH
FFFFF428H	Port 4 mode register	PM4	R/W	√	√		FFH
FFFFF42AH	Port 5 mode register	PM5	R/W	√	√		FFH
FFFFF432H	Port 9 mode register	PM9	R/W			√	FFFFH
FFFFF432H	Port 9 mode register L	PM9L	R/W	√	√		FFH
FFFFF433H	Port 9 mode register H	PM9H	R/W	√	√		FFH
FFFFF440H	Port 0 mode control register	PMC0	R/W	√	√		00H
FFFFF446H	Port 3 mode control register	PMC3	R/W			√	0000H
FFFFF446H	Port 3 mode control register L	PMC3L	R/W	√	√		00H
FFFFF447H	Port 3 mode control register H	PMC3H	R/W	√	√		00H
FFFFF448H	Port 4 mode control register	PMC4	R/W	√	√		00H
FFFFF44AH	Port 5 mode control register	PMC5	R/W	√	√		00H

Note The output latch is 00H or 0000H. When input, the pin status is read.

Downloaded from Arrow.com.

(3/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit			After Reset
				1	8	16	
FFFFF452H	Port 9 mode control register	PMC9	R/W			√	0000H
FFFFF452H	Port 9 mode control register L	PMC9L	R/W	√	√		00H
FFFFF453H	Port 9 mode control register H	PMC9H	R/W	√	√		00H
FFFFF466H	Port 3 function control register	PFC3	R/W	√	√		00H
FFFFF46AH	Port 5 function control register	PFC5	R/W	√	√		00H
FFFFF472H	Port 9 function control register	PFC9	R/W			√	0000H
FFFFF472H	Port 9 function control register L	PFC9L	R/W	√	√		00H
FFFFF473H	Port 9 function control register H	PFC9H	R/W	√	√		00H
FFFFF580H	8-bit timer H mode register 0	TMHMD0	R/W	√	√		00H
FFFFF581H	8-bit timer H carrier control register 0	TMCYC0	R/W	√	√		00H
FFFFF582H	8-bit timer H compare register 00	CMP00	R/W		√		00H
FFFFF583H	8-bit timer H compare register 01	CMP01	R/W		√		00H
FFFFF590H	8-bit timer H mode register 1	TMHMD1	R/W	√	√		00H
FFFFF591H	8-bit timer H carrier control register 1	TMCYC1	R/W	√	√		00H
FFFFF592H	8-bit timer H compare register 10	CMP10	R/W		√		00H
FFFFF593H	8-bit timer H compare register 11	CMP11	R/W		√		00H
FFFFF5A0H	TMP0 control register 0	TP0CTL0	R/W	√	√		00H
FFFFF5A1H	TMP0 control register 1	TP0CTL1	R/W	√	√		00H
FFFFF5A2H	TMP0 I/O control register 0	TP0IOC0	R/W	√	√		00H
FFFFF5A3H	TMP0 I/O control register 1	TP0IOC1	R/W	√	√		00H
FFFFF5A4H	TMP0 I/O control register 2	TP0IOC2	R/W	√	√		00H
FFFFF5A5H	TMP0 option register 0	TP0OPT0	R/W	√	√		00H
FFFFF5A6H	TMP0 capture/compare register 0	TP0CCR0	R/W			√	0000H
FFFFF5A8H	TMP0 capture/compare register 1	TP0CCR1	R/W			√	0000H
FFFFF5AAH	TMP0 counter read buffer register	TP0CNT	R			√	0000H
FFFFF5C0H	16-bit timer counter 5	TM5	R			√	0000H
FFFFF5C0H	8-bit timer counter 50	TM50	R		√		00H
FFFFF5C1H	8-bit timer counter 51	TM51	R		√		00H
FFFFF5C2H	16-bit timer compare register 5	CR5	R/W			√	0000H
FFFFF5C2H	8-bit timer compare register 50	CR50	R/W		√		00H
FFFFF5C3H	8-bit timer compare register 51	CR51	R/W		√		00H
FFFFF5C4H	Timer clock selection register 5	TCL5	R/W			√	0000H
FFFFF5C4H	Timer clock selection register 50	TCL50	R/W		√		00H
FFFFF5C5H	Timer clock selection register 51	TCL51	R/W		√		00H
FFFFF5C6H	16-bit timer mode control register 5	TMC5	R/W			√	0000H
FFFFF5C6H	8-bit timer mode control register 50	TMC50	R/W	√	√		00H
FFFFF5C7H	8-bit timer mode control register 51	TMC51	R/W	√	√		00H
FFFFF610H	16-bit timer counter 01	TM01	R			√	0000H
FFFFF612H	16-bit timer capture/compare register 010	CR010	R/W			√	0000H
FFFFF614H	16-bit timer capture/compare register 011	CR011	R/W			√	0000H
FFFFF616H	16-bit timer mode control register 01	TMC01	R/W	√	√		00H

Downloaded from Arrow.com.

(4/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit				After Reset
				1	8	16	32	
FFFFF617H	Prescaler mode register 01	PRM01	R/W	√	√			00H
FFFFF618H	Capture/compare control register 01	CRC01	R/W	√	√			00H
FFFFF619H	16-bit timer output control register 01	TOC01	R/W	√	√			00H
FFFFF680H	Watch timer operation mode register	WTM	R/W	√	√			00H
FFFFF6C0H	Oscillation stabilization time selection register	OSTS	R/W		√			01H
FFFFF6C1H	Watchdog timer clock selection register	WDCS	R/W		√			00H
FFFFF6C2H	Watchdog timer mode register 1	WDTM1	R/W	√	√			00H
FFFFF6D0H	Watchdog timer mode register 2	WDTM2	R/W		√			67H
FFFFF6D1H	Watchdog timer enable register	WDTE	R/W		√			9AH
FFFFF6E0H	Real-time output buffer register L0	RTBL0	R/W	√	√			00H
FFFFF6E2H	Real-time output buffer register H0	RTBH0	R/W	√	√			00H
FFFFF6E4H	Real-time output port mode register 0	RTPM0	R/W	√	√			00H
FFFFF6E5H	Real-time output port control register 0	RTPC0	R/W	√	√			00H
FFFFF706H	Port 3 function control expansion register	PFCE3	R/W	√	√			00H
FFFFF802H	System status register	SYS	R/W	√	√			00H
FFFFF806H	PLL control register	PLLCTL	R/W	√	√			01H
FFFFF820H	Power save mode register	PSMR	R/W	√	√			00H
FFFFF828H	Processor clock control register	PCC	R/W	√	√			03H
FFFFF840H	Correction address register 0	CORAD0	R/W				√	00000000H
FFFFF840H	Correction address register 0L	CORAD0L	R/W			√		0000H
FFFFF842H	Correction address register 0H	CORAD0H	R/W			√		0000H
FFFFF844H	Correction address register 1	CORAD1	R/W				√	00000000H
FFFFF844H	Correction address register 1L	CORAD1L	R/W			√		0000H
FFFFF846H	Correction address register 1H	CORAD1H	R/W			√		0000H
FFFFF848H	Correction address register 2	CORAD2	R/W				√	00000000H
FFFFF848H	Correction address register 2L	CORAD2L	R/W			√		0000H
FFFFF84AH	Correction address register 2H	CORAD2H	R/W			√		0000H
FFFFF84CH	Correction address register 3	CORAD3	R/W				√	00000000H
FFFFF84CH	Correction address register 3L	CORAD3L	R/W			√		0000H
FFFFF84EH	Correction address register 3H	CORAD3H	R/W			√		0000H
FFFFF880H	Correction control register	CORCN	R/W	√	√			00H
FFFFF8B0H	Interval timer BRG mode register	PRSM	R/W		√			00H
FFFFF8B1H	Interval timer BRG compare register	PRSCM	R/W		√			00H
FFFFFA00H	Asynchronous serial interface mode register 0	ASIM0	R/W	√	√			01H
FFFFFA02H	Receive buffer register 0	RXB0	R		√			FFH
FFFFFA03H	Asynchronous serial interface status register 0	ASIS0	R		√			00H
FFFFFA04H	Transmit buffer register 0	TXB0	R/W		√			FFH
FFFFFA05H	Asynchronous serial interface transmit status register 0	ASIF0	R	√	√			00H
FFFFFA06H	Clock select register 0	CKSR0	R/W		√			00H
FFFFFA07H	Baud rate generator control register 0	BRGC0	R/W		√			FFH
FFFFFA10H	Asynchronous serial interface mode register 1	ASIM1	R/W	√	√			01H
FFFFFA12H	Receive buffer register 1	RXB1	R		√			FFH

Downloaded from Arrow.com.

(5/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit			After Reset
				1	8	16	
FFFFFA13H	Asynchronous serial interface status register 1	ASIS1	R		√		00H
FFFFFA14H	Transmit buffer register 1	TXB1	R/W		√		FFH
FFFFFA15H	Asynchronous serial interface transmit status register 1	ASIF1	R	√	√		00H
FFFFFA16H	Clock select register 1	CKSR1	R/W		√		00H
FFFFFA17H	Baud rate generator control register 1	BRGC1	R/W		√		FFH
FFFFFB00H	TIP00 noise elimination control register	P0NFC	R/W	√	√		00H
FFFFFB04H	TIP01 noise elimination control register	P1NFC	R/W	√	√		00H
FFFFFC00H	External interrupt falling edge specification register 0	INTF0	R/W	√	√		00H
FFFFFC13H	External interrupt falling edge specification register 9H	INTF9H	R/W	√	√		00H
FFFFFC20H	External interrupt rising edge specification register 0	INTR0	R/W	√	√		00H
FFFFFC33H	External interrupt rising edge specification register 9H	INTR9H	R/W	√	√		00H
FFFFFC40H	Pull-up resistor option register 0	PU0	R/W	√	√		00H
FFFFFC46H	Pull-up resistor option register 3	PU3	R/W	√	√		00H
FFFFFC48H	Pull-up resistor option register 4	PU4	R/W	√	√		00H
FFFFFC4AH	Pull-up resistor option register 5	PU5	R/W	√	√		00H
FFFFFC52H	Pull-up resistor option register 9	PU9	R/W			√	0000H
FFFFFC52H	Pull-up resistor option register 9L	PU9L	R/W	√	√		00H
FFFFFC53H	Pull-up resistor option register 9H	PU9H	R/W	√	√		00H
FFFFFC67H	Port 3 function register H	PF3H	R/W	√	√		00H
FFFFFC68H	Port 4 function register	PF4	R/W	√	√		00H
FFFFFC73H	Port 9 function register H	PF9H	R/W	√	√		00H
FFFFFD00H	Clocked serial interface mode register 00	CSIM00	R/W	√	√		00H
FFFFFD01H	Clocked serial interface clock selection register 0	CSIC0	R/W	√	√		00H
FFFFFD02H	Clocked serial interface receive buffer register 0	SIRB0	R			√	0000H
FFFFFD02H	Clocked serial interface receive buffer register 0L	SIRB0L	R		√		00H
FFFFFD04H	Clocked serial interface transmit buffer register 0	SOTB0	R/W			√	0000H
FFFFFD04H	Clocked serial interface transmit buffer register 0L	SOTB0L	R/W		√		00H
FFFFFD06H	Clocked serial interface read-only receive buffer register 0	SIRBE0	R			√	0000H
FFFFFD06H	Clocked serial interface read-only receive buffer register 0L	SIRBE0L	R		√		00H
FFFFFD08H	Clocked serial interface initial transmit buffer register 0	SOTBF0	R/W			√	0000H
FFFFFD08H	Clocked serial interface initial transmit buffer register 0L	SOTBF0L	R/W		√		00H
FFFFFD0AH	Serial I/O shift register 0	SIO00	R/W			√	00H
FFFFFD0AH	Serial I/O shift register 0L	SIO00L	R/W		√		0000H
FFFFFD10H	Clocked serial interface mode register 01	CSIM01	R/W	√	√		00H
FFFFFD11H	Clocked serial interface clock selection register 1	CSIC1	R/W	√	√		00H
FFFFFD12H	Clocked serial interface receive buffer register 1	SIRB1	R			√	0000H
FFFFFD12H	Clocked serial interface receive buffer register 1L	SIRB1L	R		√		00H
FFFFFD14H	Clocked serial interface transmit buffer register 1	SOTB1	R/W			√	0000H
FFFFFD14H	Clocked serial interface transmit buffer register 1L	SOTB1L	R/W		√		00H
FFFFFD16H	Clocked serial interface read-only receive buffer register 1	SIRBE1	R			√	0000H
FFFFFD16H	Clocked serial interface read-only receive buffer register 1L	SIRBE1L	R		√		00H

Downloaded from Arrow.com.

(6/6)

Address	Function Register Name	Symbol	R/W	Operable Bit Unit			After Reset
				1	8	16	
FFFFFD18H	Clocked serial interface initial transmit buffer register 1	SOTBF1	R/W			√	0000H
FFFFFD18H	Clocked serial interface initial transmit buffer register 1L	SOTBF1L	R/W		√		00H
FFFFFD1AH	Serial I/O shift register 1	SIO01	R/W			√	00H
FFFFFD1AH	Serial I/O shift register 1L	SIO01L	R/W		√		0000H
FFFFFD80H	IIC shift register 0	IIC0Note	R/W		√		00H
FFFFFD82H	IIC control register 0	IICC0Note	R/W	√	√		00H
FFFFFD83H	Slave address register 0	SVA0Note	R/W		√		00H
FFFFFD84H	IIC clock selection register 0	IICCL0Note	R/W	√	√		00H
FFFFFD85H	IIC function expansion register 0	IICX0Note	R/W	√	√		00H
FFFFFD86H	IIC status register 0	IICS0Note	R	√	√		00H
FFFFFD8AH	IIC flag register 0	IICF0Note	R/W	√	√		00H

Note Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

3.4.7 Special registers

Special registers are registers that prevent invalid data from being written when an inadvertent program loop occurs.

The V850ES/KE1 has the following three special registers.

- Power save control register (PSC)
- Processor clock control register (PCC)
- Watchdog timer mode register (WDTM1)

Moreover, there is also the PRCMD register, which is a protection register for write operations to the special registers that prevents the application system from unexpectedly stopping due to an inadvertent program loop. Write access to the special registers is performed with a special sequence and illegal store operations are notified to the SYS register.

(1) Setting data to special registers

Setting data to a special register is done in the following sequence.

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

<4> to <8> Insert NOP instructions (5 instructions)[Note].

[Description Example] When using PSC register (standby mode setting)

```
    ST.B r11, PSMR[r0]  ; PSMR register setting (IDLE, STOP mode setting)
<1> MOV 0x02, r10
<2> ST.B r10, PRCMD[r0] ; PRCMD register write
<3> ST.B r10, PSC[r0]   ; PSC register setting
<4> NOP[Note]           ; Dummy instruction
<5> NOP[Note]           ; Dummy instruction
<6> NOP[Note]           ; Dummy instruction
<7> NOP[Note]           ; Dummy instruction
<8> NOP[Note]           ; Dummy instruction
(next instruction)
```

No special sequence is required to read special registers.

Note When switching to the IDLE mode or the STOP mode (PSC.STP bit = 1), 5 NOP instructions must be inserted immediately after switching is performed.

Cautions 1. Interrupts are not acknowledged for the store instruction for the PRCMD register. This is because continuous execution of store instructions by the program in steps <2> and <3> above is assumed. If another instruction is placed between step <2> and <3>, the above sequence may not be realized when an interrupt is acknowledged for that instruction, which may cause malfunction.

2. The data written to the PRCMD register is dummy data, but use the same register as the general-purpose register used for setting data to the special register (step <3>) when writing to the PRCMD register (step <2>). The same applies to when using a general-purpose register for addressing.

Downloaded from Arrow.com.

(2) Command register (PRCMD)

The PRCMD register is an 8-bit register used to prevent data from being written to registers that may have a large influence on the system, possibly causing the application system to unexpectedly stop, when an inadvertent program loop occurs. Only the first write operation to the special register following the execution of a previously executed write operation to the PRCMD register, is valid.

As a result, register values can be overwritten only using a preset sequence, preventing invalid write operations.

This register can only be written in 8-bit units (if it is read, an undefined value is returned).

After reset: Undefined W Address: FFFFF1FCH

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

Downloaded from Arrow.com.

(3) System status register (SYS)

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

After reset: 00H R/W Address: FFFFF802H

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

PRERR	Detection of protection error
0	Protection error has not occurred
1	Protection error has occurred

The operation conditions of the PRERR flag are described below.

(a) Set conditions (PRERR = 1)

(i) When a write operation to the special register takes place without write operation being performed to the PRCMD register (when step <3> is performed without performing step <2> as described in **3.4.7 (1) Setting data to special registers**).

(ii) When a write operation (including bit manipulation instruction) to an on-chip peripheral I/O register other than a special register is performed following write to the PRCMD register (when <3> in **3.4.7 (1) Setting data to special registers** is not a special register).

Remark Regarding the special registers other than the WDTM register (PCC and PSC registers), even if on-chip peripheral I/O register read (except bit manipulation instruction) (internal RAM access, etc.) is performed in between write to the PRCMD register and write to a special register, the PRERR flag is not set and setting data can be written to the special register.

(b) Clear conditions (PRERR = 0)

(i) When 0 is written to the PRERR flag

(ii) When system reset is performed

Cautions 1. If 0 is written to the PRERR bit of the SYS register that is not a special register immediately following write to the PRCMD register, the PRERR bit becomes 0 (write priority).

2. If data is written to the PRCMD register that is not a special register immediately following write to the PRCMD register, the PRERR bit becomes 1.

Downloaded from Arrow.com.

3.4.8 Cautions

(1) Wait when accessing register

Be sure to set the following register before using the V850ES/KE1.

- System wait control register (VSWC)

After setting the VSWC register, set the other registers as required.

When using an external bus, set the VSWC register and then set the various pins to the control mode by setting the port-related registers.

(a) System wait control register (VSWC)

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

Access to the on-chip peripheral I/O register lasts 3 clocks (during no wait), but in the V850ES/KE1, waits are required according to the internal system clock frequency. Set the values shown below to the VSWC register according to the internal system clock frequency that is used.

This register can be read or written in 8-bit units (Address: FFFFF06EH, After reset: 77H).

Operation Conditions	Internal System Clock Frequency (f_{CLK})	VSWC Register Setting	Number of Waits
4.5 V ≤ V_{DD} ≤ 5.5 V	32 kHz ≤ f_{CLK} < 16.6 MHz	00H	0 (no waits)
	16.6 MHz ≤ f_{CLK} ≤ 20 MHz	01H	1
<R> 4.0 V ≤ V_{DD} < 4.5 V	32 kHz ≤ f_{CLK} ≤ 16 MHz	00H	0 (no waits)
<R> 2.7 V ≤ V_{DD} < 4.0 V	32 kHz ≤ f_{CLK} < 8.3 MHz	00H	0 (no waits)
	8.3 MHz ≤ f_{CLK} ≤ 10 MHz	01H	1

(b) Access to special on-chip peripheral I/O register

This product has two types of internal system buses.

One type is for the CPU bus and the other is for the peripheral bus to interface with low-speed peripheral hardware.

Since the CPU bus clock and peripheral bus clock are asynchronous, if a conflict occurs during access between the CPU and peripheral hardware, illegal data may be passed unexpectedly. Therefore, when accessing peripheral hardware that may cause a conflict, the number of access cycles is changed so that the data is received/passed correctly in the CPU. As a result, the CPU does not shift to the next instruction processing and enters the wait status. When this wait status occurs, the number of execution clocks of the instruction is increased by the number of wait clocks.

Note this with caution when performing real-time processing.

When accessing a special on-chip peripheral I/O register, additional waits may be required further to the waits set by the VSWC register.

The access conditions at that time and the method to calculate the number of waits to be inserted (number of CPU clocks) are shown below.

Number of waits to be inserted = (2 + m) × k (clocks)

Number of accesses to special on-chip peripheral I/O register = 3 + m + (2 + m) × k (clocks)

Downloaded from Arrow.com.

Peripheral Function	Register Name	Access	k
Watchdog timer 1 (WDT1)	WDTM1	Write	1 to 5
	<Calculation of number of waits[Note 1]> k = {(1/fx) × 2/((2 + m)/fCPU)} + 1 fx: Main clock oscillation frequency		
Watchdog timer 2 (WDT2)	WDTM2	Write	3 (fixed)
16-bit timer/event counter P0 (TMP0)	TP0CCR0, TP0CCR1, TP0CNT	Read	1
	<Calculation of number of waits[Note 1]> k = {(1/fxx)/((2 + m)/fCPU)} + 1		
	TP0CCR0, TP0CCR1	Write	0 to 2
	<Calculation of number of waits[Note 1]> k = {(1/fxx) × 5/((2 + m)/fCPU)} A wait occurs when performing continuous write to same register		
16-bit timer/event counter 01 (TM01)	TMC01	Read-modify-write	1 (fixed) A wait occurs during write
I^2C0[Note 2]	IICS0	Read	1 (fixed)
Asynchronous serial interfaces 0, 1 (UART0, UART1)	ASIS0, ASIS1	Read	1 (fixed)
Real-time output function 0 (RTO0)	RTBL0, RTBH0	Write (when RTPC0.RTPOE0 bit = 0)	1
A/D converter	ADM, ADS, PFM, PFT	Write	1 to 2
	ADCR, ADCRH	Read	1 to 2
	<Calculation of number of waits[Note 1]> k = {(1/fxx) × 2/[(2 + m)/fCPU]} + 1		

Notes 1. In the calculation of number of waits, the fractional part of its result must be multiplied by (1/fCPU) and rounded down if (1/fCPU)/(2 + m) or lower, and rounded up if (1/fCPU)/(2 + m) is exceeded.

2. Only in the μPD703207Y, 70F3207HY

<R> **Cautions 1. If fetched from the internal ROM or internal RAM, the number of waits is as shown above. If fetched from the external memory, the number of waits may be decreased below these. The effect of the external memory access cycles varies depending on the wait settings and the like. However, the number of waits shown above is the maximum value, so no higher value is generated.**

2. When the CPU operates on the subclock and no clock is input to the X1 pin, do not access a register in which a wait occurs using an access method that causes a wait. If a wait occurs, it can only be released by a reset.

Remark In the calculation for the number of waits:

fCPU: CPU clock frequency

fXX: Main clock frequency

m: Set value of bits 2 to 0 of the VSWC register

When the VSWC register = 00H: m = 0

When the VSWC register = 01H: m = 1

Downloaded from Arrow.com.

(2) Restriction on conflict between sld instruction and interrupt request

(a) Description

If a conflict occurs between the decode operation of an instruction in <2> immediately before the sld instruction following an instruction in <1> and an interrupt request before the instruction in <1> is complete, the execution result of the instruction in <1> may not be stored in a register.

Instruction <1>

- ld instruction: ld.b, ld.h, ld.w, ld.bu, ld.hu
- sld instruction: sld.b, sld.h, sld.w, sld.bu, sld.hu
- Multiplication instruction: mul, mulh, mulhi, mulu

Instruction <2>

mov reg1, reg2	not reg1, reg2	satsubr reg1, reg2	satsub reg1, reg2
satadd reg1, reg2	satadd imm5, reg2	or reg1, reg2	xor reg1, reg2
and reg1, reg2	tst reg1, reg2	subr reg1, reg2	sub reg1, reg2
add reg1, reg2	add imm5, reg2	cmp reg1, reg2	cmp imm5, reg2
mulh reg1, reg2	shr imm5, reg2	sar imm5, reg2	shl imm5, reg2

<Example>

<i> ld.w [r11], r10
•
•
•
<ii> mov r10, r28
<iii> sld.w 0x28, r10

If the decode operation of the mov instruction <ii> immediately before the sld instruction <iii> and an interrupt request conflict before execution of the ld instruction <i> is complete, the execution result of instruction <i> may not be stored in a register.

(b) Countermeasure

<R> <1> When compiler (CA850) is used

Use CA850 Ver. 2.61 or later because generation of the corresponding instruction sequence can be automatically suppressed.

<2> Countermeasure by assembler

When executing the sld instruction immediately after instruction <ii>, avoid the above operation using either of the following methods.

- Insert a nop instruction immediately before the sld instruction.
- Do not use the same register as the sld instruction destination register in the above instruction <ii> executed immediately before the sld instruction.

Downloaded from Arrow.com.

CHAPTER 4 PORT FUNCTIONS

4.1 Features

- Input-only ports: 8 pins
- I/O ports: 43 pins
  - Fixed to N-ch open-drain output: 2
  - Switchable to N-ch open-drain output: 4
- Input/output can be specified in 1-bit units

4.2 Basic Port Configuration

The V850ES/KE1 incorporates a total of 51 I/O port pins consisting of ports 0, 3 to 5, 7, 9, CM, and DL (including 8 input-only port pins). The port configuration is shown below.

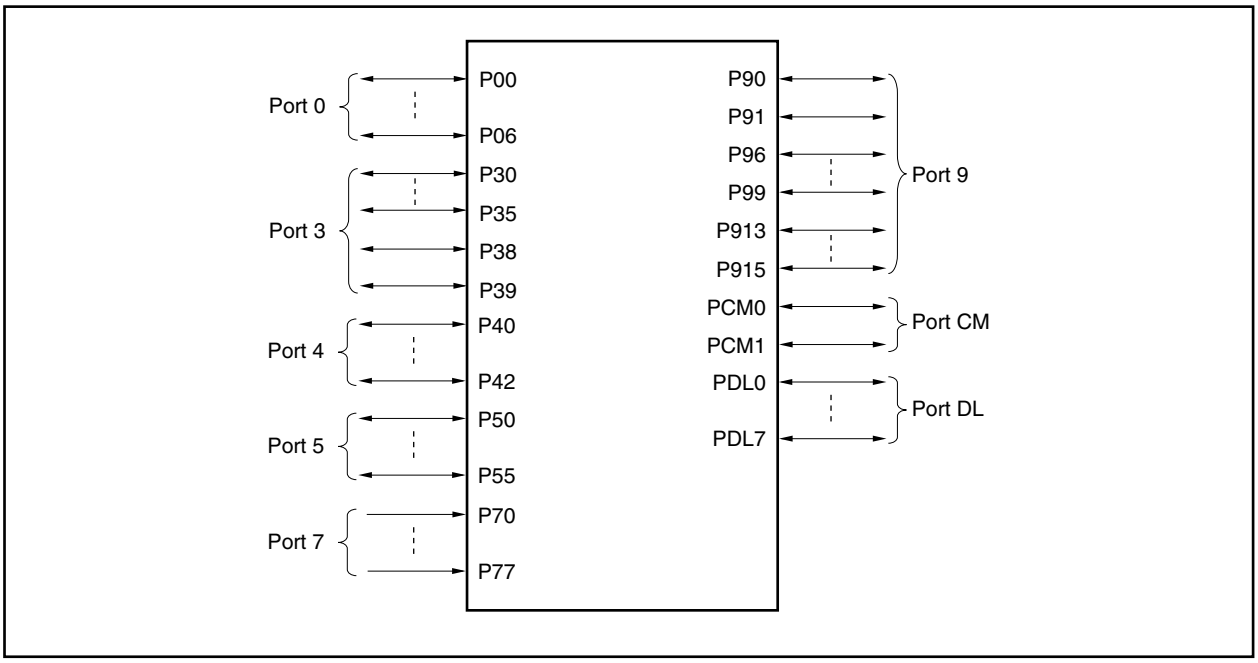


Table 4-1. Pin I/O Buffer Power Supplies

Power Supply	Corresponding Pins
AV_{REF0}	Port 7
EV_{DD}	$\overline{RESET}$, ports 0, 3 to 5, 9, CM, DL

Downloaded from Arrow.com.

4.3 Port Configuration

Table 4-2. Port Configuration

Item	Configuration
Control registers	Port n register (Pn: n = 0, 3 to 5, 7, 9, CM, DL) Port n mode register (PMn: n = 0, 3 to 5, 9, CM, DL) Port n mode control register (PMCn: n = 0, 3 to 5, 9, CM) Port n function control register (PFCn: n = 3, 5, 9) Port 3 function control expansion register (PFCE3) Port n function register (PFn: n = 3, 4, 9) Pull-up resistor option register (PUn: n = 0, 3 to 5, 9)
Ports	Input only: 8 I/O: 43
Pull-up resistors	Software control: 31

Downloaded from Arrow.com.

(1) Port n register (Pn)

Data I/O with external devices is performed by writing to and reading from the Pn register. The Pn register is configured of a port latch that retains the output data and a circuit that reads the pin status.

Each bit of the Pn register corresponds to one pin of port n and can be read or written in 1-bit units.

After reset: 00H[Note] (output latch) R/W

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

Pnm	Control of output data (in output mode)
0	0 is output
1	1 is output

Note Input-only port pins are undefined.

Writing to and reading from the Pn register is executed as follows depending on the setting of each register.

<R> **Table 4-3. Writing to/Reading from Pn Register**

Setting of PMCn Register	Setting of PMn Register	Writing to Pn Register	Reading from Pn Register
Port mode (PMCnm bit = 0)	Output mode (PMnm bit = 0)	Write to the output latch[Note]. The contents of the output latch are output from the pin.	The value of the output latch is read.
	Input mode (PMnm bit = 1)	Write to the output latch[Note]. The status of the pin is not affected.	The pin status is read.
Alternate-function mode (PMCnm bit = 1)	Output mode (PMnm bit = 0)	Write to the output latch[Note]. The status of the pin is not affected. The pin operates as an alternate-function pin.	• When alternate function is output The output status of the alternate function is read. • When alternate function is input The output latch value is read.
	Input mode (PMnm bit = 1)	Write to the output latch[Note]. The status of the pin is not affected. The pin operates as an alternate-function pin.	The pin status is read.

Note The value written to the output latch is retained until a new value is written to the output latch.

Downloaded from Arrow.com.

(2) Port n mode register (PMn)

PMn specifies the input mode/output mode of the port.

Each bit of the PMn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: FFH R/W

	7	6	5	4	3	2	1	0
PMn	PMn7	PMn6	PMn5	PMn4	PMn3	PMn2	PMn1	PMn0

PMnm	Control of I/O mode
0	Output mode
1	Input mode

(3) Port n mode control register (PMCn)

PMCn specifies the port mode/alternate function.

Each bit of the PMCn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PMCn	PMCn7	PMCn6	PMCn5	PMCn4	PMCn3	PMCn2	PMCn1	PMCn0

PMCnm	Specification of operation mode
0	Port mode
1	Alternate function mode

(4) Port n function control register (PFCn)

PFCn is a register that specifies the alternate function to be used when one pin has two or more alternate functions.

Each bit of the PFCn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PFCn	PFCn7	PFCn6	PFCn5	PFCn4	PFCn3	PFCn2	PFCn1	PFCn0

PFCnm	Specification of alternate function
0	Alternate function 1
1	Alternate function 2

Downloaded from Arrow.com.

(5) Port n function control expansion register (PFCEn)

PFCEn is a register that specifies the alternate function to be used when one pin has three or more alternate functions.

Each bit of the PFCEn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PFCEn	PFCEn7	PFCEn6	PFCEn5	PFCEn4	PFCEn3	PFCEn2	PFCEn1	PFCEn0

	7	6	5	4	3	2	1	0
PFCn	PFCn7	PFCn6	PFCn5	PFCn4	PFCn3	PFCn2	PFCn1	PFCn0

PFCEnm	PFCnm	Specification of alternate function
0	0	Alternate function 1
0	1	Alternate function 2
1	0	Alternate function 3
1	1	Alternate function 4

(6) Port n function register (PFn)

PFn is a register that specifies normal output/N-ch open-drain output.

Each bit of the PFn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PFn	PFn7	PFn6	PFn5	PFn4	PFn3	PFn2	PFn1	PFn0

PFnm[Note]	Control of normal output/N-ch open-drain output
0	Normal output (CMOS output)
1	N-ch open-drain output

Note The PFnm bit is valid only when the PMn.PMnm bit is 0 (output mode) regardless of the setting of the PMCn register. When the PMnm bit is 1 (input mode), the set value in the PFn register is invalid.

Example <1> When the value of the PFn register is valid
PFnm bit = 1 … N-ch open-drain output is specified.
PMnm bit = 0 … Output mode is specified.
PMCnm bit = 0 or 1

<2> When the value of the PFn register is invalid
PFnm bit = 0 … N-ch open-drain output is specified.
PMnm bit = 1 … Input mode is specified.
PMCnm bit = 0 or 1

Downloaded from Arrow.com.

(7) Pull-up resistor option register (PUn)

PUn is a register that specifies the connection of an on-chip pull-up resistor.

Each bit of the PUn register corresponds to one pin of port n and can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PUn	PUn7	PUn6	PUn5	PUn4	PUn3	PUn2	PUn1	PUn0

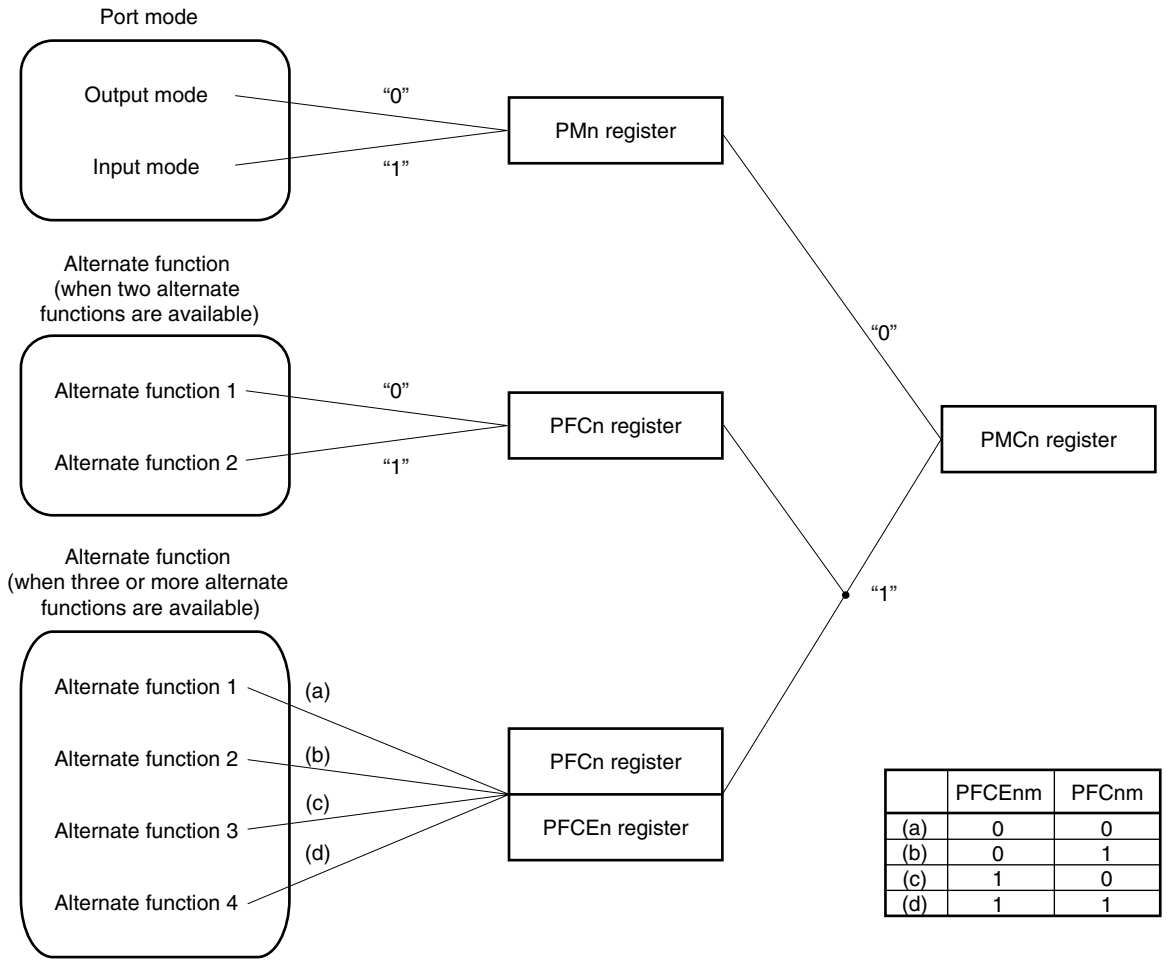
PUnm	Control of on-chip pull-up resistor connection
0	Not connected
1	Connected

Downloaded from Arrow.com.

(8) Port settings

Set the ports as follows.

Figure 4-1. Register Settings and Pin Functions



	PFCEnm	PFCnm
(a)	0	0
(b)	0	1
(c)	1	0
(d)	1	1

Remark Switch to the alternate function using the following procedure.

<1> Set the PFCn and PFCEn registers.
<2> Set the PMCn register.
<3> Set the INTRn or INTFn register (to specify an external interrupt pin).

If the PMCn register is set first, an unintended function may be set while the PFCn and PFCEn registers are being set.

Downloaded from Arrow.com.

4.3.1 Port 0

Port 0 is a 7-bit I/O port for which I/O settings can be controlled in 1-bit units.

Port 0 includes the following alternate functions.

Table 4-4. Alternate-Function Pins of Port 0

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
P00	12	TOH0	Output	Yes	–	D-2
P01	13	TOH1	Output			D-2
P02	14	NMI	Input		Analog noise elimination	H-1
P03	15	INTP0	Input			H-1
P04	16	INTP1	Input			H-1
P05	17	INTP2	Input			H-1
P06	18	INTP3	Input			H-1

Note Software pull-up function

Caution P02 to P06 have hysteresis characteristics when the alternate function is input, but not in the port mode.

(1) Port 0 register (P0)

After reset: 00H (output latch) R/W Address: FFFFF400H

	7	6	5	4	3	2	1	0
P0	0	P06	P05	P04	P03	P02	P01	P00

P0n	Control of output data (in output mode) (n = 0 to 6)
0	0 is output
1	1 is output

(2) Port 0 mode register (PM0)

After reset: FFH R/W Address: FFFFF420H

	7	6	5	4	3	2	1	0
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00

PM0n	Control of I/O mode (n = 0 to 6)
0	Output mode
1	Input mode

Downloaded from Arrow.com.

(3) Port 0 mode control register (PMC0)

After reset: 00H R/W Address: FFFFF440H

	7	6	5	4	3	2	1	0
PMC0	0	PMC06	PMC05	PMC04	PMC03	PMC02	PMC01	PMC00

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

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

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

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

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

PMC01	Specification of P01 pin operation mode
0	I/O port
1	TOH1 output

PMC00	Specification of P00 pin operation mode
0	I/O port
1	TOH0 output

(4) Pull-up resistor option register 0 (PU0)

After reset: 00H R/W Address: FFFFFC40H

	7	6	5	4	3	2	1	0
PU0	0	PU06	PU05	PU04	PU03	PU02	PU01	PU00

PU0n	Control of on-chip pull-up resistor connection (n = 0 to 6)
0	Not connected
1	Connected

Downloaded from Arrow.com.

4.3.2 Port 3

Port 3 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units.

Port 3 includes the following alternate functions.

Table 4-5. Alternate-Function Pins of Port 3

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note 1]	Remark	Block Type
P30	22	TXD0	Output	Yes	–	D-2
P31	23	RXD0	Input			D-1-1
P32	24	ASCK0	Input			D-1-2
P33	25	TIP00/TOP00	I/O			G-7-3
P34	26	TIP01/TOP01	I/O			G-7-3
P35	27	TI010/TO01	I/O			E-6
P38	55	SDA0[Note 2]	I/O	No[Note 3]	N-ch open-drain output	K
P39	56	SCL0[Note 2]	I/O			K

Notes 1. Software pull-up function

2. Only in the μPD703207Y, 70F3207HY

3. An on-chip pull-up resistor can be provided by a mask option (only in the mask ROM versions).

Caution P31 to P35, P38, and P39 have hysteresis characteristics when the alternate function is input, but not in the port mode.

Downloaded from Arrow.com.

(1) Port 3 register (P3)

After reset: 00H (output latch) R/W Address: P3 FFFFF406H,
P3L FFFFF406H, P3H FFFFF407H

	15	14	13	12	11	10	9	8
P3 (P3H[Note])	0	0	0	0	0	0	P39	P38

	7	6	5	4	3	2	1	0
(P3L)	0	0	P35	P34	P33	P32	P31	P30

P3n	Control of output data (in output mode) (n = 0 to 5, 8, 9)
0	0 is output
1	1 is output

Note When reading from or writing to bits 8 to 15 of the P3 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the P3H register.

Remark The P3 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the P3 register are used as the P3H register and as the P3L register, respectively, this register can be read or written in 8-bit or 1-bit units.

(2) Port 3 mode register (PM3)

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

	15	14	13	12	11	10	9	8
PM3 (PM3H[Note])	1	1	1	1	1	1	PM39	PM38

	7	6	5	4	3	2	1	0
(PM3L)	1	1	PM35	PM34	PM33	PM32	PM31	PM30

PM3n	Control of I/O mode (n = 0 to 5, 8, 9)
0	Output mode
1	Input mode

Note When reading from or writing to bits 8 to 15 of the PM3 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PM3H register.

Remark The PM3 register can be read or written in 16-bit units.
When the higher 8 bits and the lower 8 bits of the PM3 register are used as the PM3H register and as the PM3L register, respectively, this register can be read or written in 8-bit or 1-bit units.

Downloaded from Arrow.com.

(3) Port 3 mode control register (PMC3)

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

	15	14	13	12	11	10	9	8
PMC3 (PMC3H[Note 1])	0	0	0	0	0	0	PMC39[Note 2]	PMC38[Note 2]

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

PMC39	Specification of P39 pin operation mode
0	I/O port
1	SCL0 I/O

PMC38	Specification of P38 pin operation mode
0	I/O port
1	SDA0 I/O

PMC35	Specification of P35 pin operation mode
0	I/O port
1	TI010 input/TO01 output

PMC34	Specification of P34 pin operation mode
0	I/O port
1	TIP01 input/TOP01 output

PMC33	Specification of P33 pin operation mode
0	I/O port
1	TIP00 input/TOP00 output

PMC32	Specification of P32 pin operation mode
0	I/O port
1	ASCK0 input

PMC31	Specification of P31 pin operation mode
0	I/O port
1	RXD0 input

PMC30	Specification of P30 pin operation mode
0	I/O port
1	TXD0 output

Notes 1. When reading from or writing to bits 8 to 15 of the PMC3 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PMC3H register.

2. Valid only in the μPD703207Y and 70F3207HY. In all other products, set this bit to 0.

Remark The PMC3 register can be read or written in 16-bit units.
When the higher 8 bits and the lower 8 bits of the PMC3 register are used as the PMC3H register and as the PMC3L register, respectively, this register can be read or written in 8-bit or 1-bit units.

Downloaded from Arrow.com.

(4) Port 3 function register H (PF3H)

After reset: 00H R/W Address: FFFFFC67H

	7	6	5	4	3	2	1	0
PF3H	0	0	0	0	0	0	PF39	PF38

PF3n	Specification of normal port/alternate function (n = 8, 9)
0	When used as normal port (N-ch open-drain output)
1	When used as alternate-function (N-ch open-drain output)

Caution When using P38 and P39 as N-ch open-drain-output alternate-function pins, set in the following sequence.
Be sure to set the port latch to 1 before setting the pin to N-ch open-drain output.
P3n bit = 1 → PF3n bit = 1 → PMC3n bit = 1

(5) Port 3 function control register (PFC3)

After reset: 00H R/W Address: FFFFF466H

	7	6	5	4	3	2	1	0
PFC3	0	0	PFC35	PFC34	PFC33	0	0	0

Caution Always clear bits 0 to 2, 6, and 7 of the PFC3 register to 0.

Remark For details of specification of alternate-function pins, refer to **4.3.2 (8) Specifying alternate-function pins of port 3**.

(6) Port 3 function control expansion register (PFCE3)

After reset: 00H R/W Address: FFFFF706H

	7	6	5	4	3	2	1	0
PFCE3	0	0	0	PFCE34	PFCE33	0	0	0

Remark For details of specification of alternate-function pins, refer to **4.3.2 (8) Specifying alternate-function pins of port 3**.

Downloaded from Arrow.com.

(7) Pull-up resistor option register 3 (PU3)

After reset: 00H R/W Address: FFFFFC46H

	7	6	5	4	3	2	1	0
PU3	0	0	PU35	PU34	PU33	PU32	PU31	PU30

PU3n	Control of on-chip pull-up resistor connection (n = 0 to 5)
0	Not connected
1	Connected

Caution An on-chip pull-up resistor can be provided for P38 and P39 by a mask option (only in the μPD703207, 703207Y).

(8) Specifying alternate-function pins of port 3

Caution When the P3n pin is specified as an alternate function by the PMC3n bit with the PFC3n and PFCE3n bits maintaining the initial value (0), output becomes undefined. Therefore, to specify the P3n pin as an alternate function, set the PFC3n and PFCE3n bits first and then set the PMC3n bit to 1 (n = 3, 4).

PFC35	Specification of Alternate-Function Pin of P35 Pin
0	TI010 input
1	TO01 output

PFCE34	PFC34	Specification of Alternate-Function Pin of P34 Pin
0	0	Setting prohibited
0	1	Setting prohibited
1	0	TIP01 input
1	1	TOP01 output

PFCE33	PFC33	Specification of Alternate-Function Pin of P33 Pin
0	0	Setting prohibited
0	1	Setting prohibited
1	0	TIP00 input
1	1	TOP00 output

Downloaded from Arrow.com.

4.3.3 Port 4

Port 4 is a 3-bit I/O port for which I/O settings can be controlled in 1-bit units.

Port 4 includes the following alternate functions.

Table 4-6. Alternate-Function Pins of Port 4

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
P40	19	SI00	Input	Yes	–	D-1-2
P41	20	SO00	Output		N-ch open-drain output can be selected.	F-1
P42	21	$\overline{\text{SCK00}}$	I/O			F-2

Note Software pull-up function

Caution P40 and P42 have hysteresis characteristics when the alternate function is input, but not in the port mode.

(1) Port 4 register (P4)

After reset: 00H (output latch) R/W Address: FFFFF408H

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

P4n	Control of output data (in output mode) (n = 0 to 2)
0	0 is output
1	1 is output

(2) Port 4 mode register (PM4)

After reset: FFH R/W Address: FFFFF428H

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

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

Downloaded from Arrow.com.

(3) Port 4 mode control register (PMC4)

After reset: 00H R/W Address: FFFFF448H

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

PMC42	Specification of P42 pin operation mode
0	I/O port
1	$\overline{\text{SCK00}}$ I/O

PMC41	Specification of P41 pin operation mode
0	I/O port
1	SO00 output

PMC40	Specification of P40 pin operation mode
0	I/O port
1	SI00 input

(4) Port 4 function register (PF4)

After reset: 00H R/W Address: FFFFFC68H

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

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

Caution When using P41 and P42 as N-ch open-drain-output alternate-function pins, set in the following sequence.
Be sure to set the port latch to 1 before setting the pin to N-ch open-drain output.
P4n bit = 1 → PF4n bit = 1 → PMC4n bit = 1

(5) Pull-up resistor option register 4 (PU4)

After reset: 00H R/W Address: FFFFFC48H

	7	6	5	4	3	2	1	0
PU4	0	0	0	0	0	PU42	PU41	PU40

PU4n	Control of on-chip pull-up resistor connection (n = 0 to 2)
0	Not connected
1	Connected

Downloaded from Arrow.com.

4.3.4 Port 5

Port 5 is a 6-bit I/O port for which I/O settings can be controlled in 1-bit units.

Port 5 includes the following alternate functions.

Table 4-7. Alternate-Function Pins of Port 5

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
P50	28	TI011/RTP00/KR0	I/O	Yes	–	E-5
P51	29	TI50/RTP01/KR1	I/O			E-5
P52	30	TO50/RTP02/KR2	I/O			E-4
P53	31	RTP03/KR3	I/O			E-9
P54	34	RTP04/KR4	I/O			E-9
P55	35	RTP05/KR5	I/O			E-9

Note Software pull-up function

(1) Port 5 register (P5)

After reset: 00H (output latch) R/W Address: FFFFF40AH

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

P5n	Control of output data (in output mode) (n = 0 to 5)
0	0 is output
1	1 is output

(2) Port 5 mode register (PM5)

After reset: FFH R/W Address: FFFFF42AH

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

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

Downloaded from Arrow.com.

(3) Port 5 mode control register (PMC5)

After reset: 00H R/W Address: FFFFF44AH

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

PMC55	Specification of P55 pin operation mode
0	I/O port/KR5 input
1	RTP05 output

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

PMC53	Specification of P53 pin operation mode
0	I/O port/KR3 input
1	RTP03 output

PMC52	Specification of P52 pin operation mode
0	I/O port/KR2 input
1	TO50 output/RTP02 output

PMC51	Specification of P51 pin operation mode
0	I/O port/KR1 input
1	TI50 input/RTP01 output

PMC50	Specification of P50 pin operation mode
0	I/O port/KR0 input
1	TI011 input/RTP00 output

Downloaded from Arrow.com.

(4) Port 5 function control register (PFC5)

Caution When the P5n pin is specified as an alternate function by the PMC5.PMC5n bit with the PFC5n bit maintaining the initial value (0), output becomes undefined. Therefore, to specify the P5n pin as alternate function 2, set the PFC5n bit to 1 first and then set the PMC5n bit to 1 (n = 3 to 5).

After reset: 00H R/W Address: FFFFF46AH

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

PFC55	Specification of alternate-function pin of P55 pin
1	RTP05 output

PFC54	Specification of alternate-function pin of P54 pin
1	RTP04 output

PFC53	Specification of alternate-function pin of P53 pin
1	RTP03 output

PFC52	Specification of alternate-function pin of P52 pin
0	TO50 output
1	RTP02 output

PFC51	Specification of alternate-function pin of P51 pin
0	TI50 input
1	RTP01 output

PFC50	Specification of alternate-function pin of P50 pin
0	TI011 input
1	RTP00 output

(5) Pull-up resistor option register 5 (PU5)

After reset: 00H R/W Address: FFFFFC4AH

	7	6	5	4	3	2	1	0
PU5	0	0	PU55	PU54	PU53	PU52	PU51	PU50

PU5n	Control of on-chip pull-up resistor connection (n = 0 to 5)
0	Not connected
1	Connected

Downloaded from Arrow.com.

4.3.5 Port 7

Port 7 is an 8-bit input-only port for which all the pins are fixed to input.

Port 7 includes the following alternate functions.

Table 4-8. Alternate-Function Pins of Port 7

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
P70	64	ANI0	Input	No	–	A-1
P71	63	ANI1	Input			A-1
P72	62	ANI2	Input			A-1
P73	61	ANI3	Input			A-1
P74	60	ANI4	Input			A-1
P75	59	ANI5	Input			A-1
P76	58	ANI6	Input			A-1
P77	57	ANI7	Input			A-1

Note Software pull-up function

(1) Port 7 register (P7)

After reset: Undefined R Address: FFFFF40EH

	7	6	5	4	3	2	1	0
P7	P77	P76	P75	P74	P73	P72	P71	P70

P7n	Input data read (n = 0 to 7)
0	Input low level
1	Input high level

Downloaded from Arrow.com.

4.3.6 Port 9

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

Port 9 includes the following alternate functions.

Table 4-9. Alternate-Function Pins of Port 9

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
P90	36	TXD1/KR6	I/O	Yes	–	E-9
P91	37	RXD1/KR7	Input			E-7
P96	38	TI51/TO51	I/O			E-9
P97	39	SI01	Input			E-8
P98	40	SO01	Output		N-ch open-drain output can be specified.	G-9
P99	41	$\overline{\text{SCK01}}$	I/O			G-8
P913	42	INTP4	Input		Analog noise elimination	H-3
P914	43	INTP5	Input			H-3
P915	44	INTP6	Input			H-3

Note Software pull-up function

Caution P97, P99, and P913 to P915 have hysteresis characteristics when the alternate function is input, but not in the port mode.

Downloaded from Arrow.com.

(1) Port 9 register (P9)

After reset: 00H (output latch) R/W Address: P9 FFFFF412H,
P9L FFFFF412H, P9H FFFFF413H

	15	14	13	12	11	10	9	8
P9 (P9H[Note])	P915	P914	P913	0	0	0	P99	P98

	7	6	5	4	3	2	1	0
(P9L)	P97	P96	0	0	0	0	P91	P90

P9n	Control of output data (in output mode) (n = 0, 1, 6 to 9, 13 to 15)
0	0 is output
1	1 is output

Note When reading from or writing to bits 8 to 15 of the P9 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the P9H register.

Remark The P9 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the P9 register are used as the P9H register and as the P9L register, respectively, these registers can be read or written in 8-bit or 1-bit units.

(2) Port 9 mode register (PM9)

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

	15	14	13	12	11	10	9	8
PM9 (PM9H[Note])	PM915	PM914	PM913	1	1	1	PM99	PM98

	7	6	5	4	3	2	1	0
(PM9L)	PM97	PM96	1	1	1	1	PM91	PM90

PM9n	Control of I/O mode (n = 0, 1, 6 to 9, 13 to 15)
0	Output mode
1	Input mode

Note When reading from or writing to bits 8 to 15 of the PM9 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PM9H register.

Remark The PM9 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the PM9 register are used as the PM9H register and as the PM9L register, respectively, this register can be read or written in 8-bit or 1-bit units.

(3) Port 9 mode control register (PMC9)

Downloaded from Arrow.com.

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

	15	14	13	12	11	10	9	8
PMC9 (PMC9H[Note])	PMC915	PMC914	PMC913	0	0	0	PMC99	PMC98

	7	6	5	4	3	2	1	0
(PMC9L)	PMC97	PMC96	0	0	0	0	PMC91	PMC90

PMC915	Specification of P915 pin operation mode
0	I/O port
1	INTP6 input

PMC914	Specification of P914 pin operation mode
0	I/O port
1	INTP5 input

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

PMC99	Specification of P99 pin operation mode
0	I/O port
1	$\overline{\text{SCK01}}$ I/O

PMC98	Specification of P98 pin operation mode
0	I/O port
1	SO01 output

PMC97	Specification of P97 pin operation mode
0	I/O port
1	SI01 input

PMC96	Specification of P96 pin operation mode
0	I/O port/TI51 input
1	TO51 output

PMC91	Specification of P91 pin operation mode
0	I/O port/KR7 input
1	RXD1 input

PMC90	Specification of P90 pin operation mode
0	I/O port/KR6 input
1	TXD1 output

Note When reading from or writing to bits 8 to 15 of the PMC9 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PMC9H register.

Remark The PMC9 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the PMC9 register are used as the PMC9H register and as the PMC9L register, respectively, these registers can be read or written in 8-bit or 1-bit units.

Downloaded from Arrow.com.

(4) Port 9 function register H (PF9H)

After reset: 00H R/W Address: FFFFFC73H

	7	6	5	4	3	2	1	0
PF9H	0	0	0	0	0	0	PF99	PF98

PF9n	Control of normal output/N-ch open-drain output (n = 8, 9)
0	Normal output
1	N-ch open-drain output

Caution When using P98 and P99 as N-ch open-drain-output alternate-function pins, set in the following sequence.
Be sure to set the port latch to 1 before setting the pin to N-ch open-drain output.
P9n bit = 1 → PFC9n bit = 0/1 → PF9n bit = 1 → PMC9n bit = 1

Downloaded from Arrow.com.

(5) Port 9 function control register (PFC9)

Caution When the P9n pin is specified as an alternate function by the PMC9.PMC9n bit with the PFC9.PFC9n bit maintaining the initial value (0), output becomes undefined. Therefore, to specify the P9n pin as alternate function 2, set the PFC9n bit to 1 first and then set the PMC9n bit to 1 (n = 0, 1, 6 to 9, 13 to 15).

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

	15	14	13	12	11	10	9	8
PFC9 (PFC9H[Note])	PFC915	PFC914	PFC913	0	0	0	PFC99	PFC98

	7	6	5	4	3	2	1	0
(PFC9L)	PFC97	PFC96	0	0	0	0	PFC91	PFC90

PFC915	Specification of P915 pin operation mode in control mode
1	INTP6 input

PFC914	Specification of P914 pin operation mode in control mode
1	INTP5 input

PFC913	Specification of P913 pin operation mode in control mode
1	INTP4 input

PFC99	Specification of P99 pin operation mode in control mode
1	$\overline{\text{SCK01}}$ I/O

PFC98	Specification of P98 pin operation mode in control mode
1	SO01 output

PFC97	Specification of P97 pin operation mode in control mode
1	SI01 input

PFC96	Specification of P96 pin operation mode in control mode
1	TO51 output

PFC91	Specification of P91 pin operation mode in control mode
1	RXD1 input

PFC90	Specification of P90 pin operation mode in control mode
1	TXD1 output

Note When reading from or writing to bits 8 to 15 of the PFC9 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PFC9H register.

Remark The PFC9 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the PFC9 register are used as the PFC9H register and as the PFC9L register, respectively, these registers can be read or written in 8-bit or 1-bit units.

Downloaded from Arrow.com.

(6) Pull-up resistor option register 9 (PU9)

After reset: 0000H R/W Address: PU9 FFFFFC52H,
PU9L FFFFFC52H, PU9H FFFFFC53H

	15	14	13	12	11	10	9	8
PU9 (PU9H[Note])	PU915	PU914	PU913	0	0	0	PU99	PU98

	7	6	5	4	3	2	1	0
(PU9L)	PU97	PU96	0	0	0	0	PU91	PU90

PU9n	Control of on-chip pull-up resistor connection (n = 0, 1, 6 to 9, 13 to 15)
0	Not connected
1	Connected

Note When reading from or writing to bits 8 to 15 of the PU9 register in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the PU9H register.

Remark The PU9 register can be read or written in 16-bit units.
However, when the higher 8 bits and the lower 8 bits of the PU9 register are used as the PU9H register and as the PU9L register, respectively, these registers can be read or written in 8-bit or 1-bit units.

Downloaded from Arrow.com.

4.3.7 Port CM

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

Port CM includes the following alternate functions.

Table 4-10. Alternate-Function Pins of Port CM

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
PCM0	45	–	–	No	–	B-1
PCM1	46	CLKOUT	Output			C-2

Note Software pull-up function

(1) Port CM register (PCM)

After reset: 00H (output latch) R/W Address: FFFFF00CH

	7	6	5	4	3	2	1	0
PCM	0	0	0	0	0	0	PCM1	PCM0

PCMn	Control of output data (in output mode) (n = 0, 1)
0	0 is output
1	1 is output

(2) Port CM mode register (PMCM)

After reset: FFH R/W Address: FFFFF02CH

	7	6	5	4	3	2	1	0
PMCM	1	1	1	1	1	1	PMCM1	PMCM0

PMCMn	Control of I/O mode (n = 0, 1)
0	Output mode
1	Input mode

(3) Port CM mode control register (PMCCM)

After reset: 00H R/W Address: FFFFF04CH

	7	6	5	4	3	2	1	0
PMCCM	0	0	0	0	0	0	PMCCM1	0

PMCCM1	Specification of PCM1 pin operation mode
0	I/O port
1	CLKOUT output

Downloaded from Arrow.com.

4.3.8 Port DL

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

Port DL includes the following alternate functions.

Table 4-11. Alternate-Function Pins of Port DL

Pin Name	Pin No.	Alternate Function	I/O	PULL[Note]	Remark	Block Type
PDL0	47	–	–	No	–	B-1
PDL1	48	–	–			B-1
PDL2	49	–	–			B-1
PDL3	50	–	–			B-1
PDL4	51	–	–			B-1
PDL5	52	–	–			B-1
PDL6	53	–	–			B-1
PDL7	54	–	–			B-1

Note Software pull-up function

(1) Port DL register (PDL)

After reset: 00H (output latch) R/W Address: FFFFF004H

	7	6	5	4	3	2	1	0
PDL	PDL7	PDL6	PDL5	PDL4	PDL3	PDL2	PDL1	PDL0

PDLn	Control of output data (in output mode) (n = 0 to 7)
0	0 is output
1	1 is output

(2) Port DL mode register (PMDL)

After reset: FFFFH R/W Address: FFFFF024H

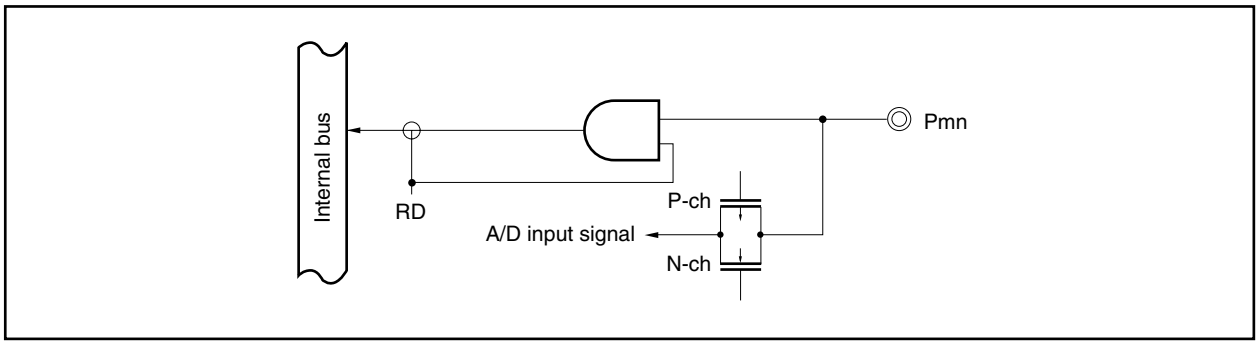
	7	6	5	4	3	2	1	0
PMDL	PMDL7	PMDL6	PMDL5	PMDL4	PMDL3	PMDL2	PMDL1	PMDL0

PMDLn	Control of I/O mode (n = 0 to 7)
0	Output mode
1	Input mode

Downloaded from Arrow.com.

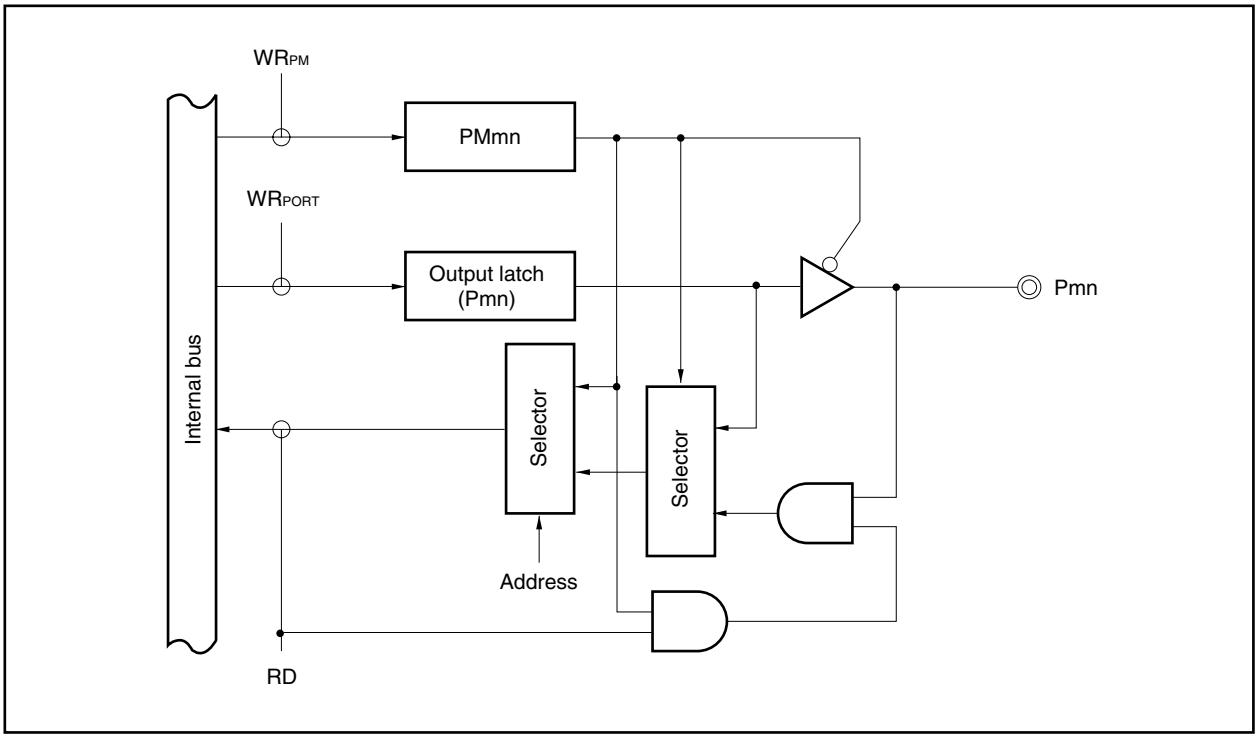
4.4 Block Diagrams

Figure 4-2. Block Diagram of Type A-1



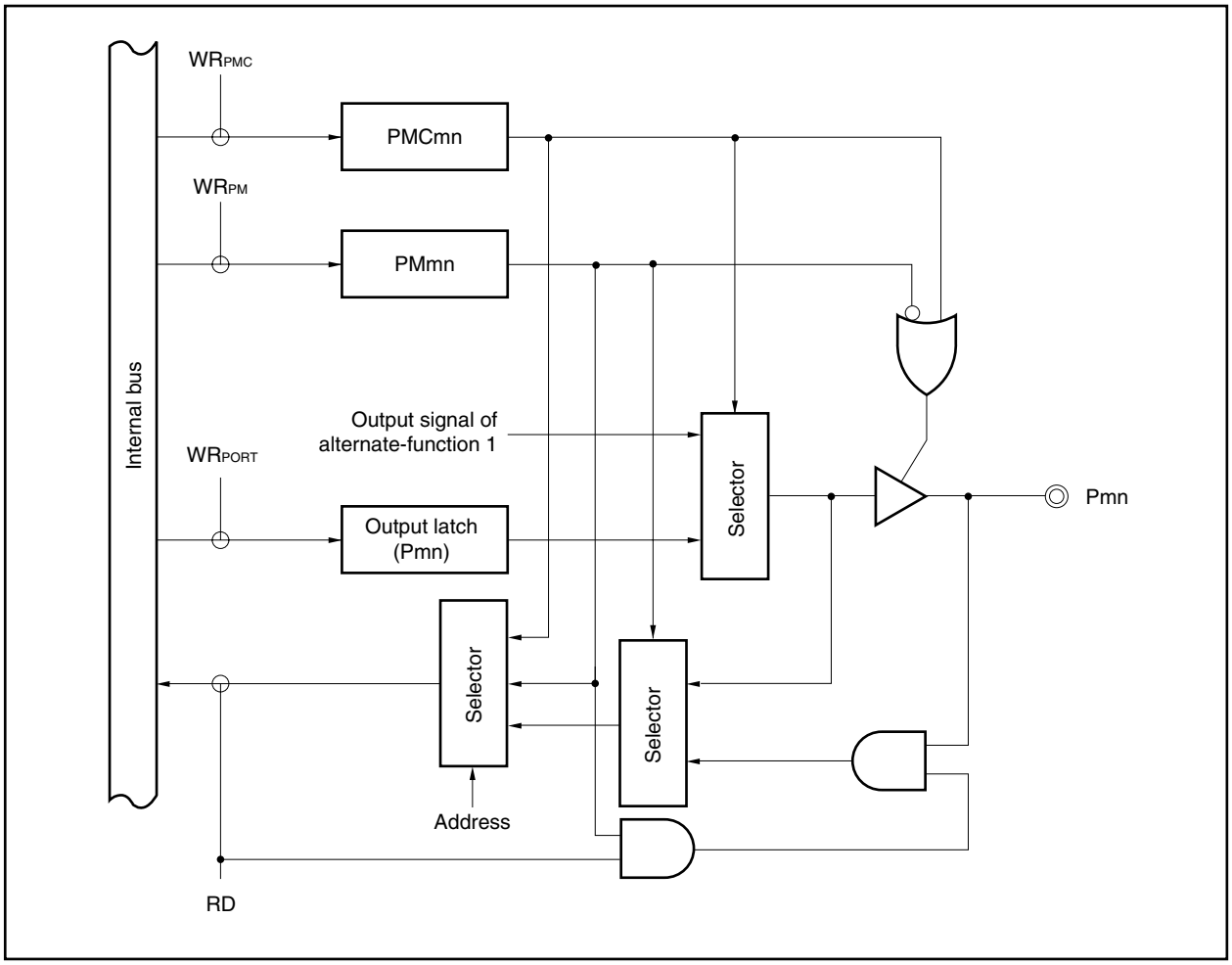
Downloaded from Arrow.com.

Figure 4-3. Block Diagram of Type B-1



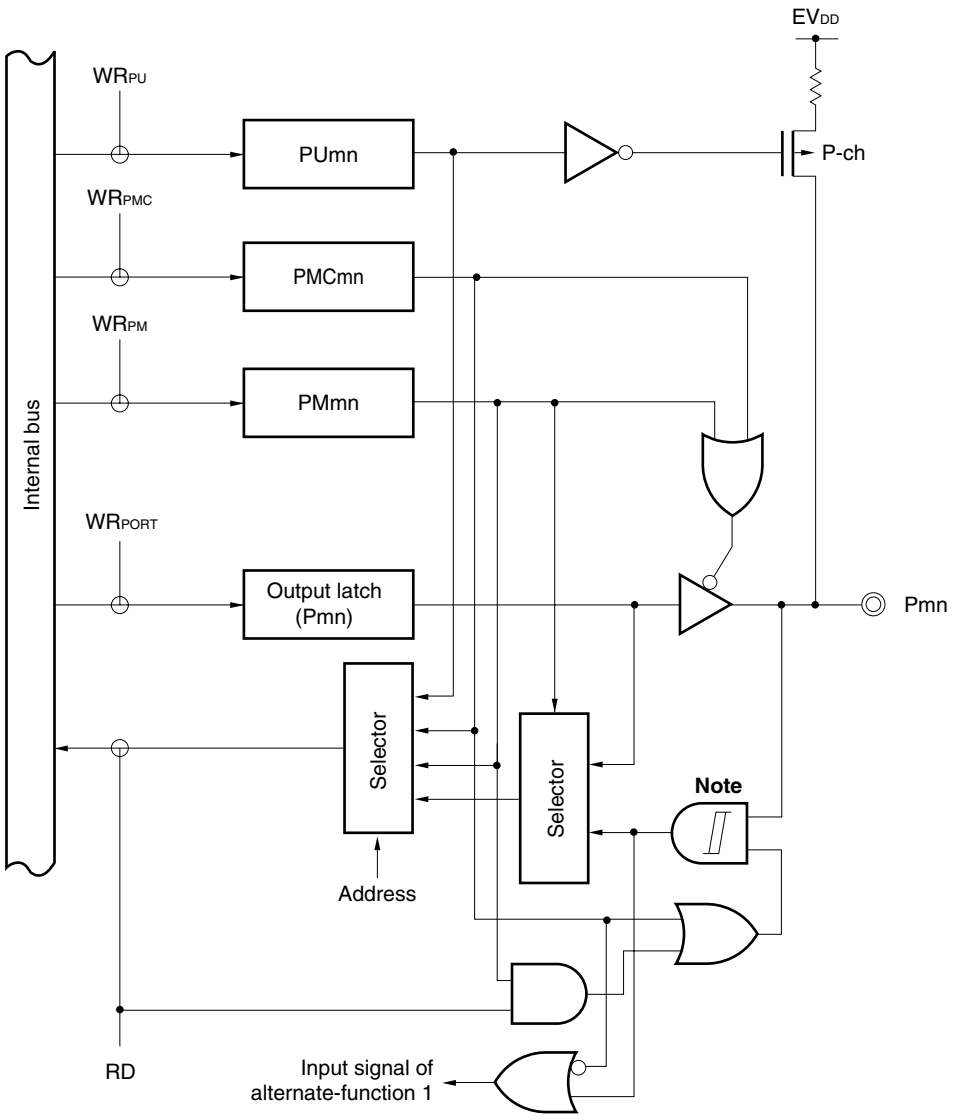
Downloaded from Arrow.com.

Figure 4-4. Block Diagram of Type C-2



Downloaded from Arrow.com.

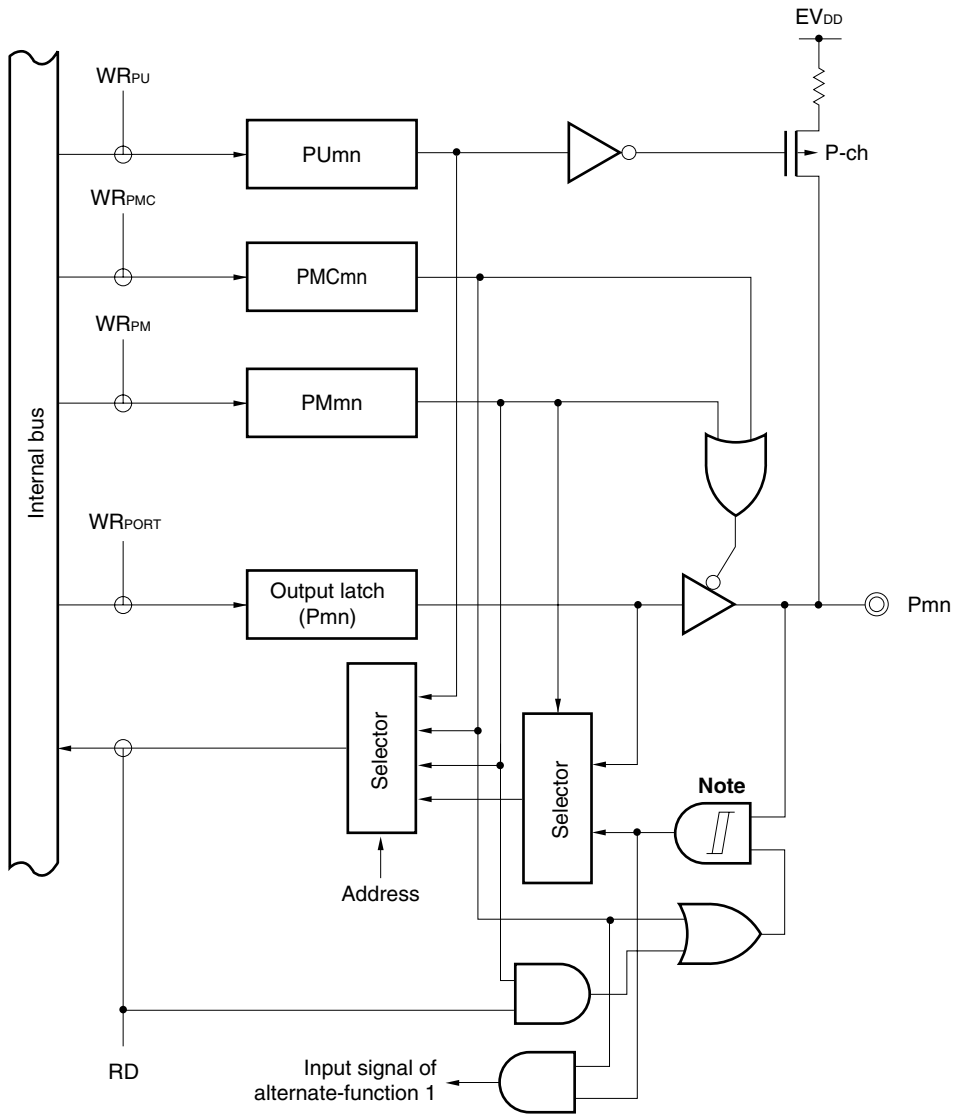
Figure 4-5. Block Diagram of Type D-1-1



Note There are no hysteresis characteristics in the port mode.

Downloaded from Arrow.com.

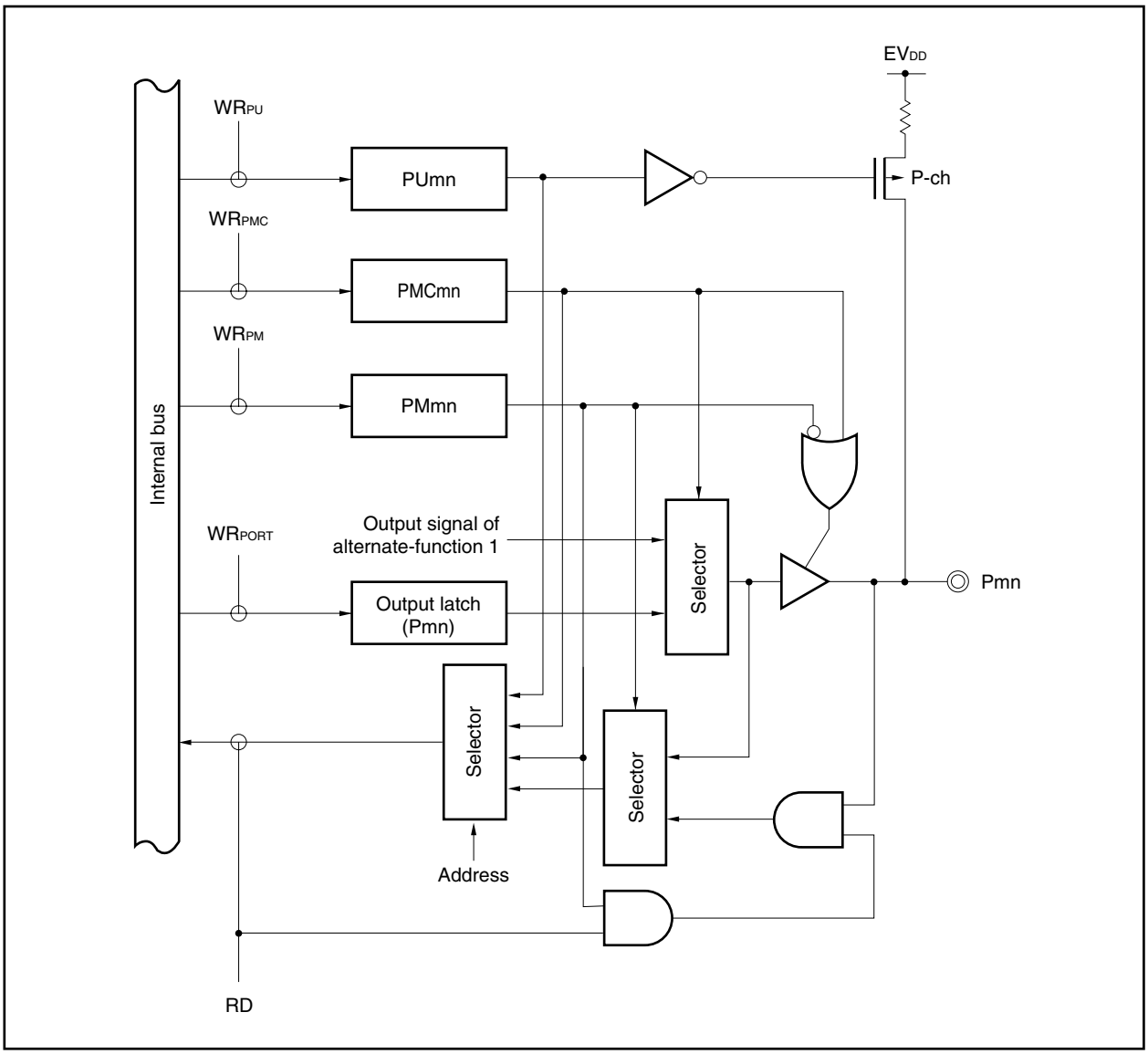
Figure 4-6. Block Diagram of Type D-1-2



Note There are no hysteresis characteristics in the port mode.

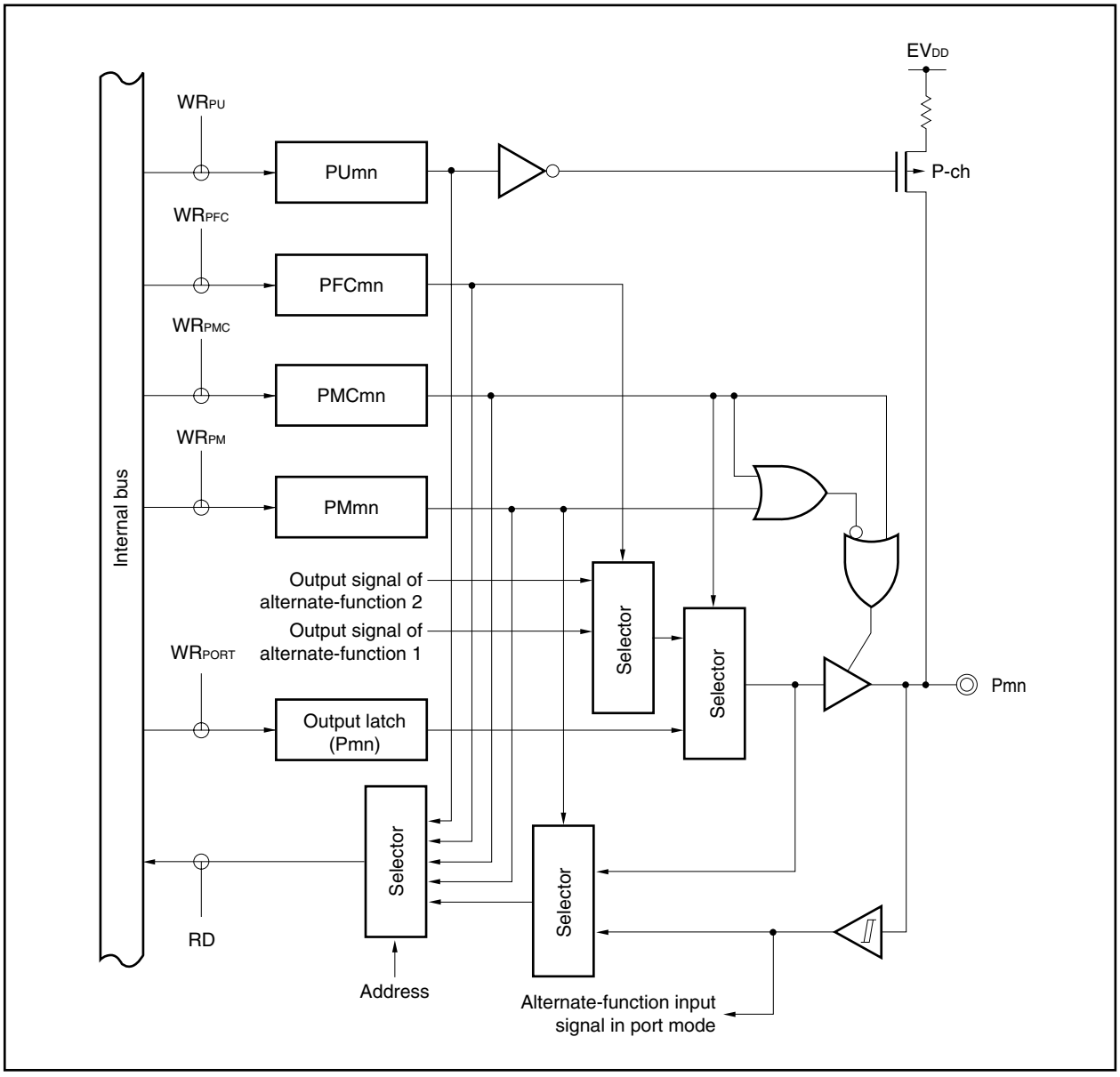
Downloaded from Arrow.com.

Figure 4-7. Block Diagram of Type D-2



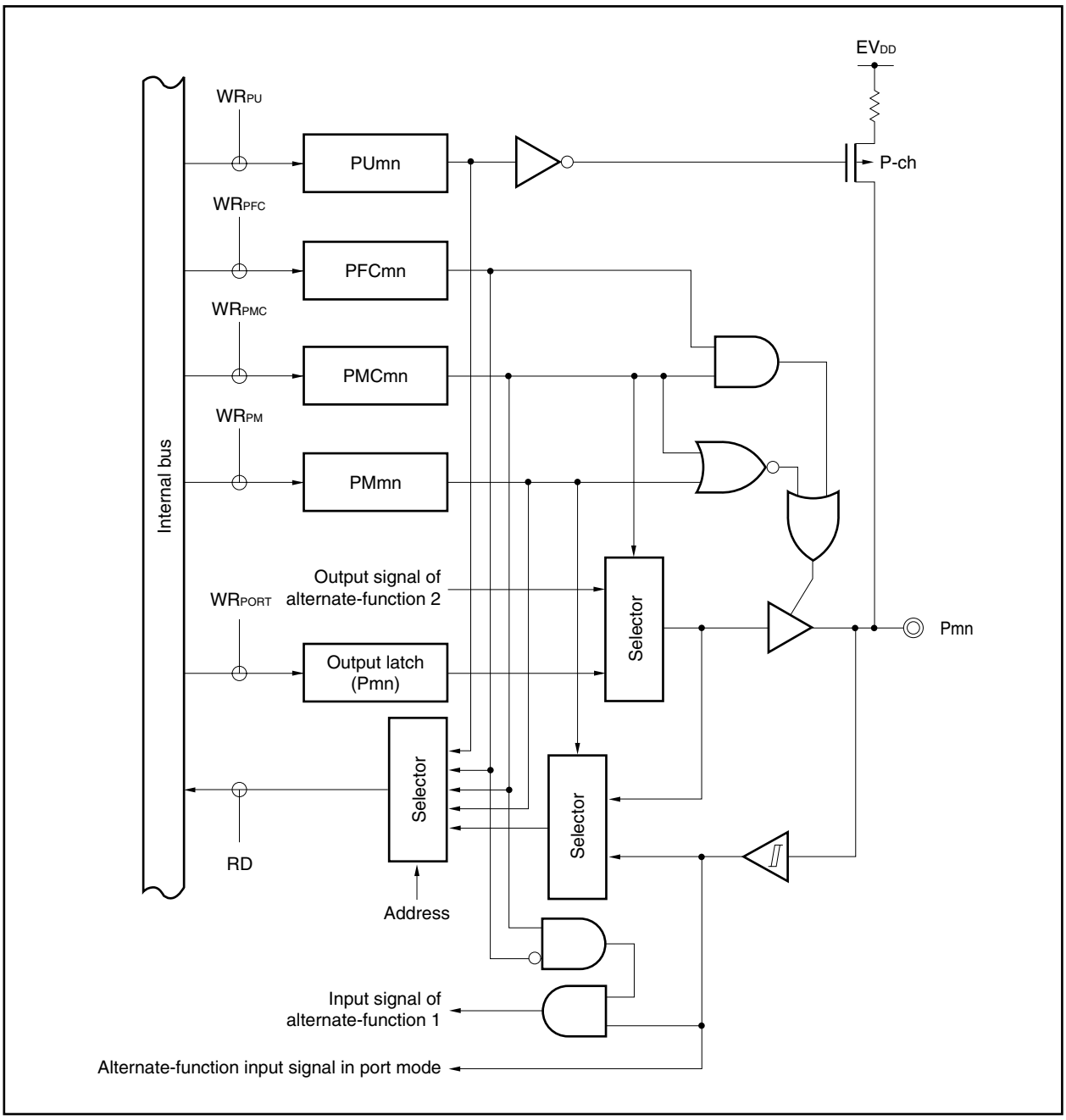
Downloaded from Arrow.com.

Figure 4-8. Block Diagram of Type E-4



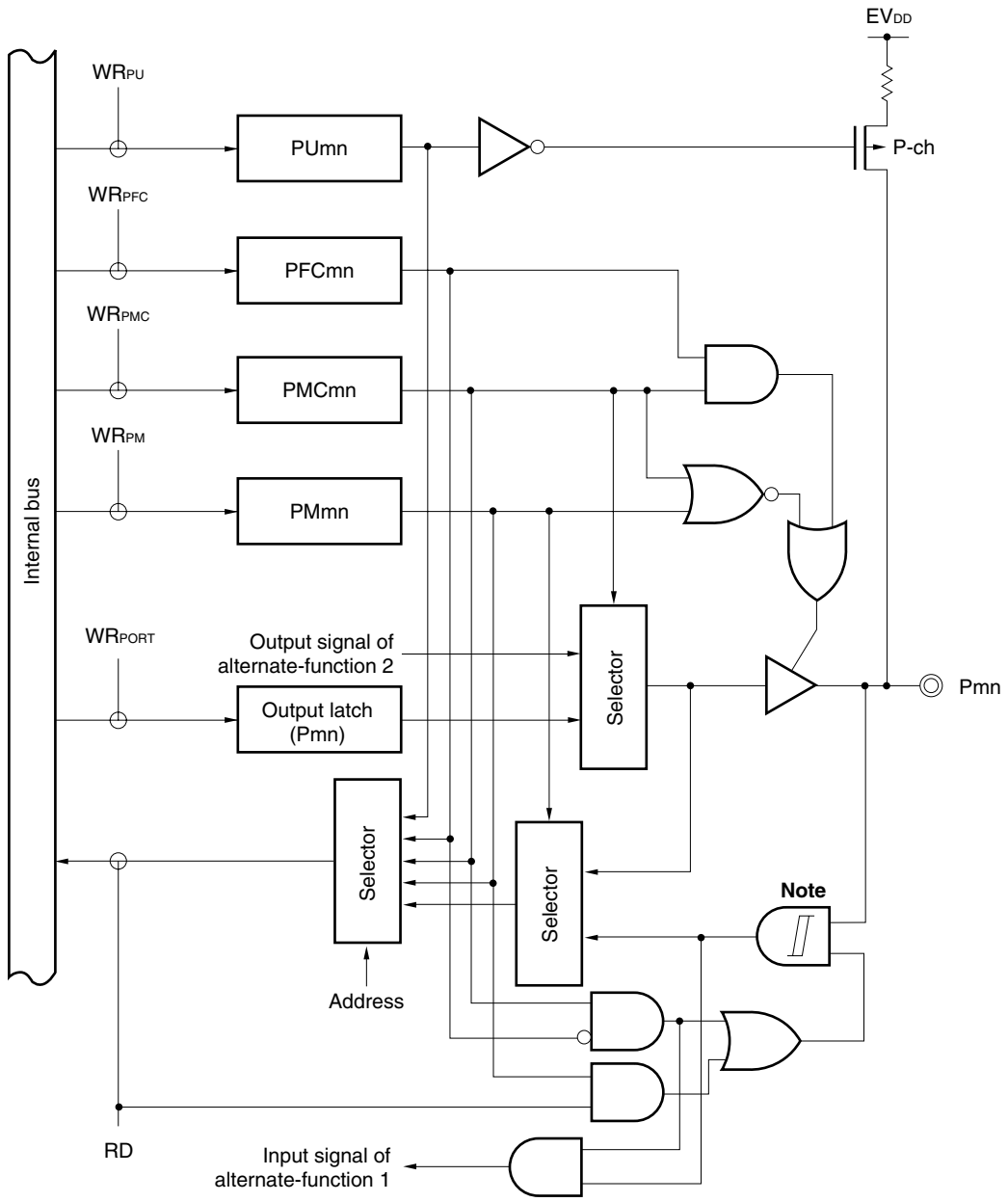
Downloaded from Arrow.com.

Figure 4-9. Block Diagram of Type E-5



Downloaded from Arrow.com.

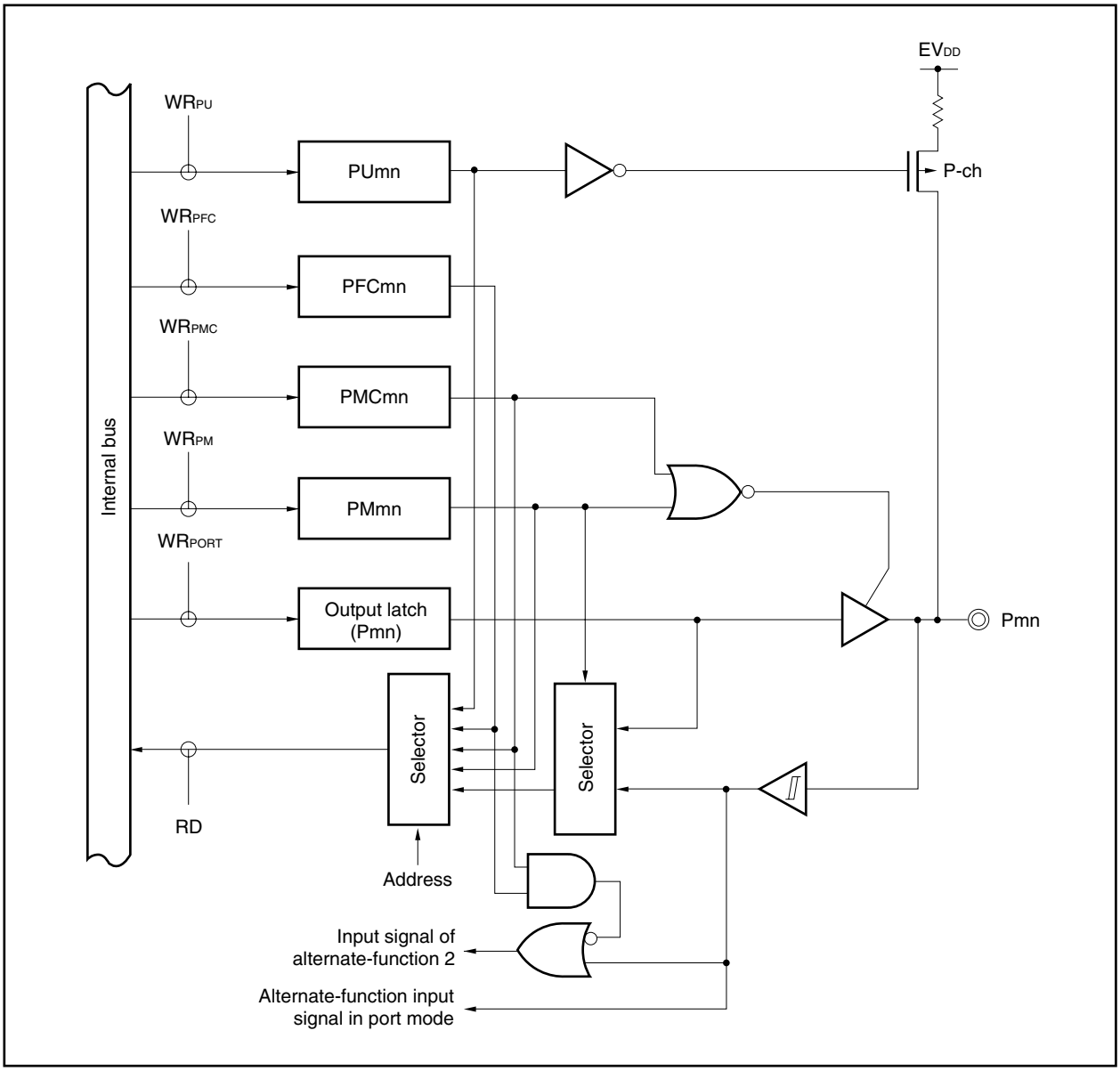
Figure 4-10. Block Diagram of Type E-6



Note There are no hysteresis characteristics in the port mode.

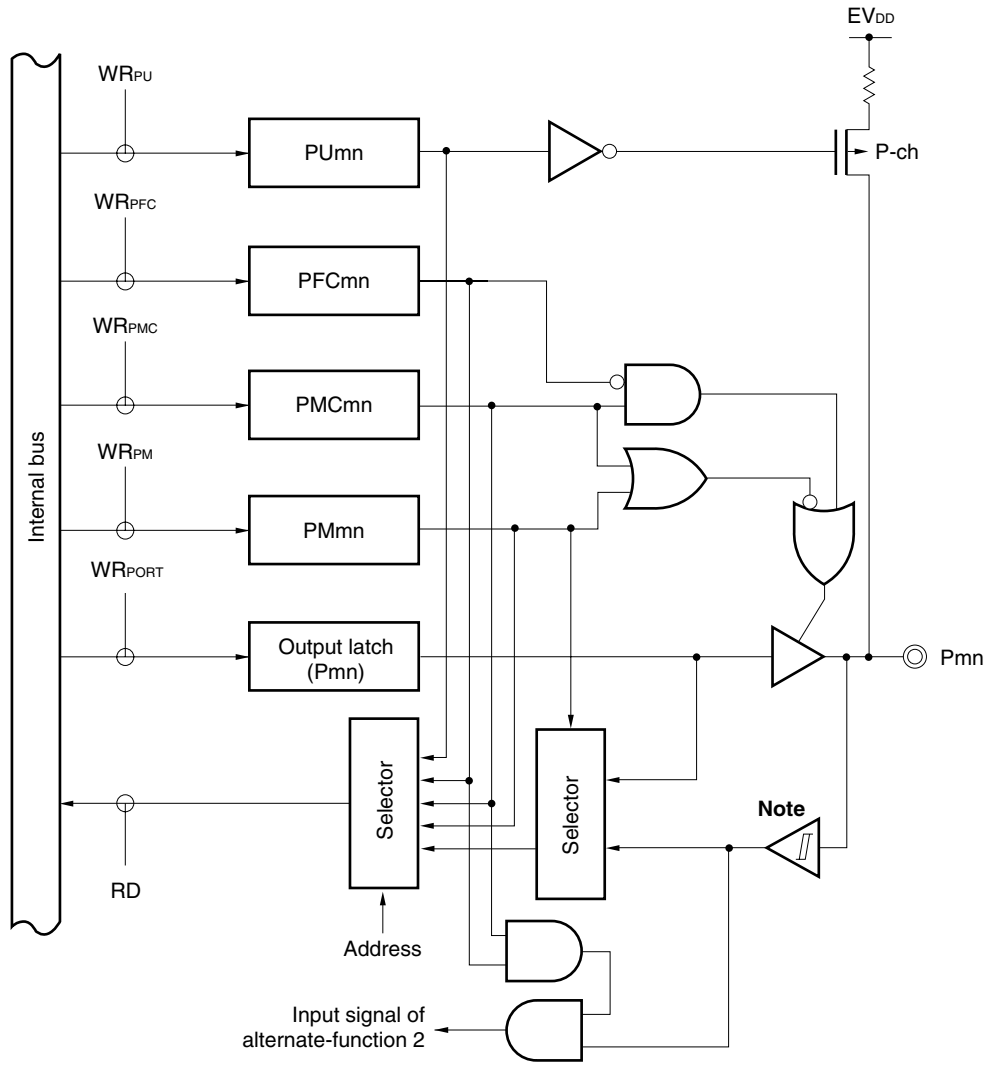
Downloaded from Arrow.com.

Figure 4-11. Block Diagram of Type E-7



Downloaded from Arrow.com.

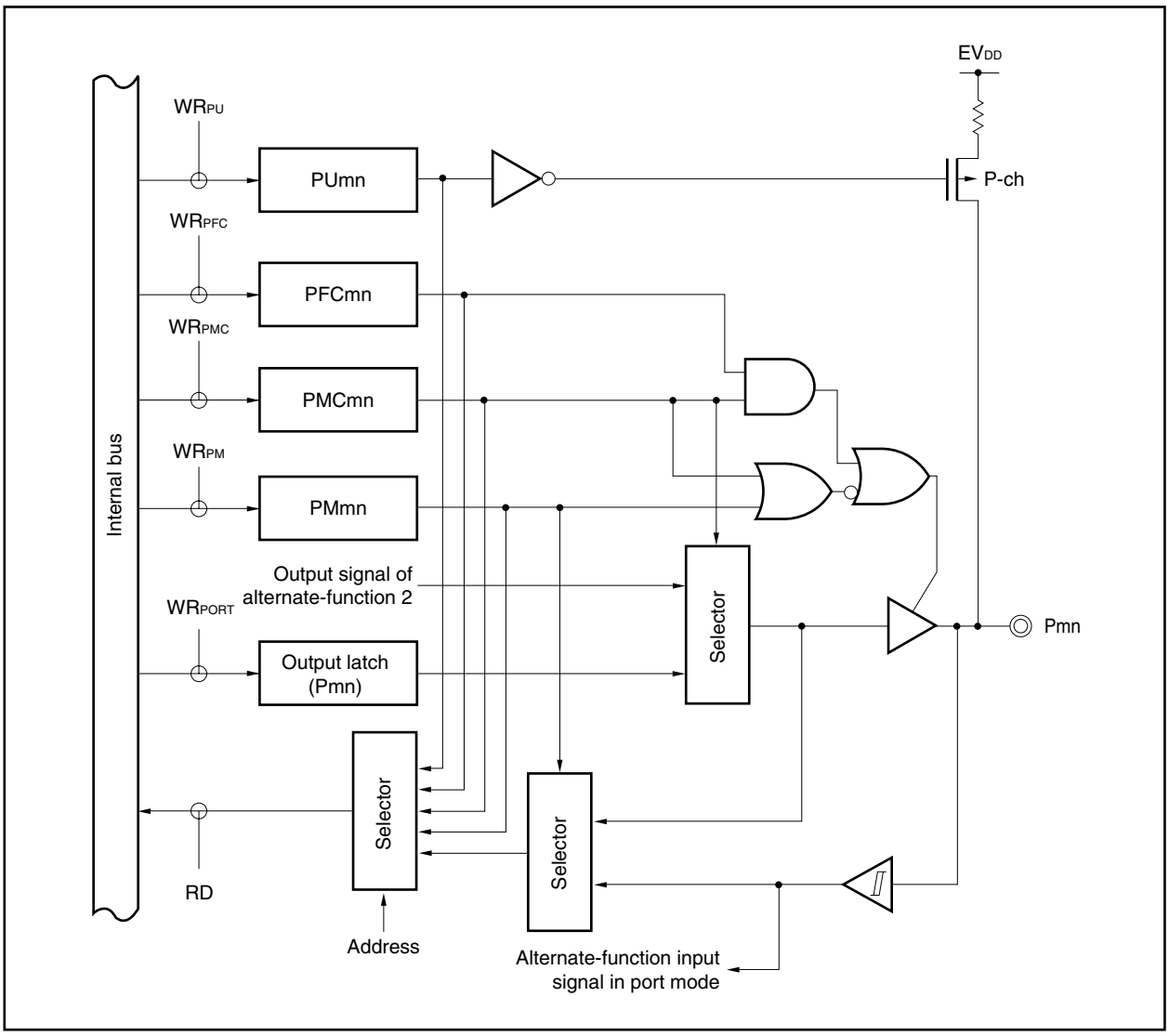
Figure 4-12. Block Diagram of Type E-8



Note There are no hysteresis characteristics in the port mode.

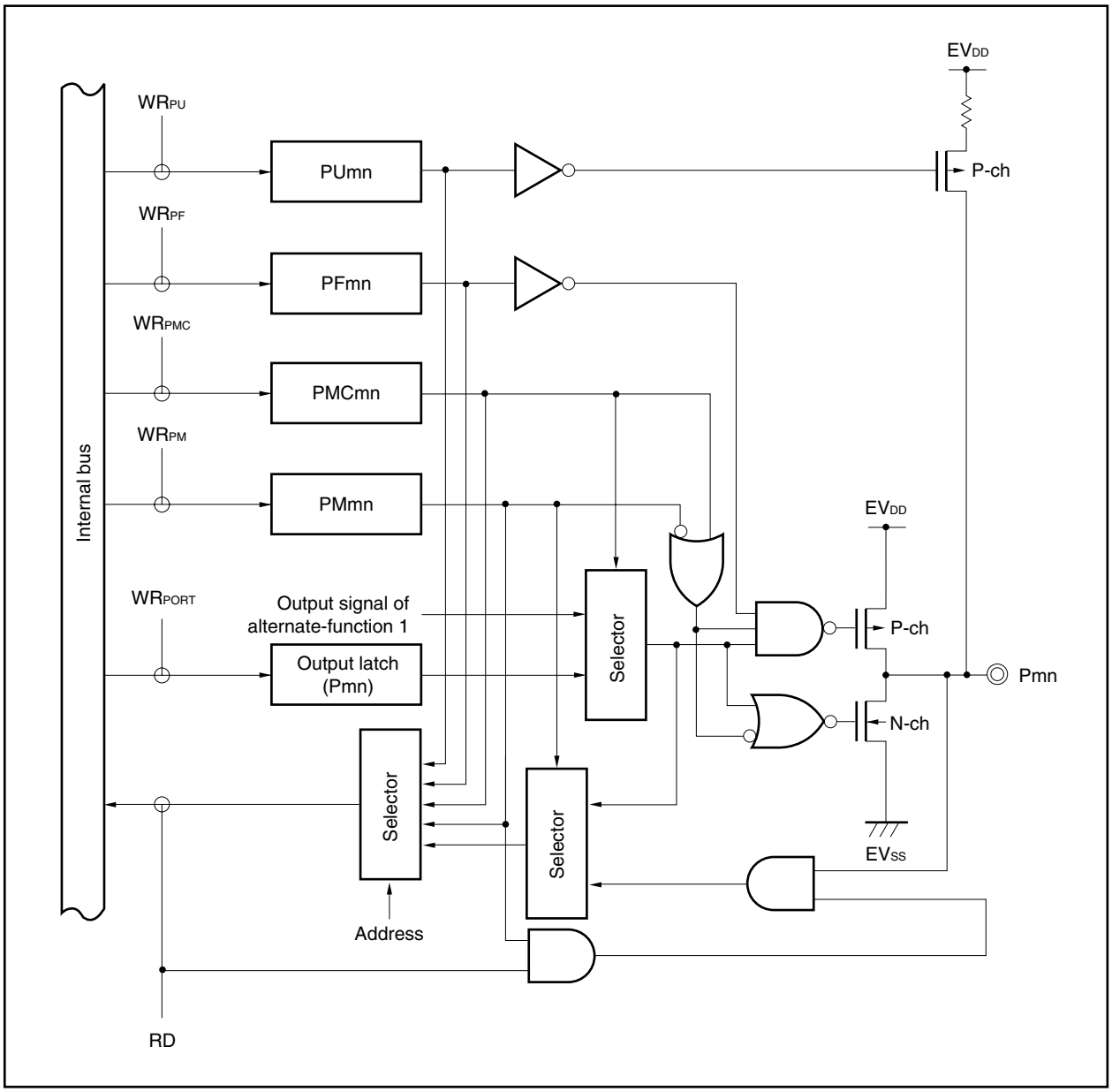
Downloaded from Arrow.com.

Figure 4-13. Block Diagram of Type E-9



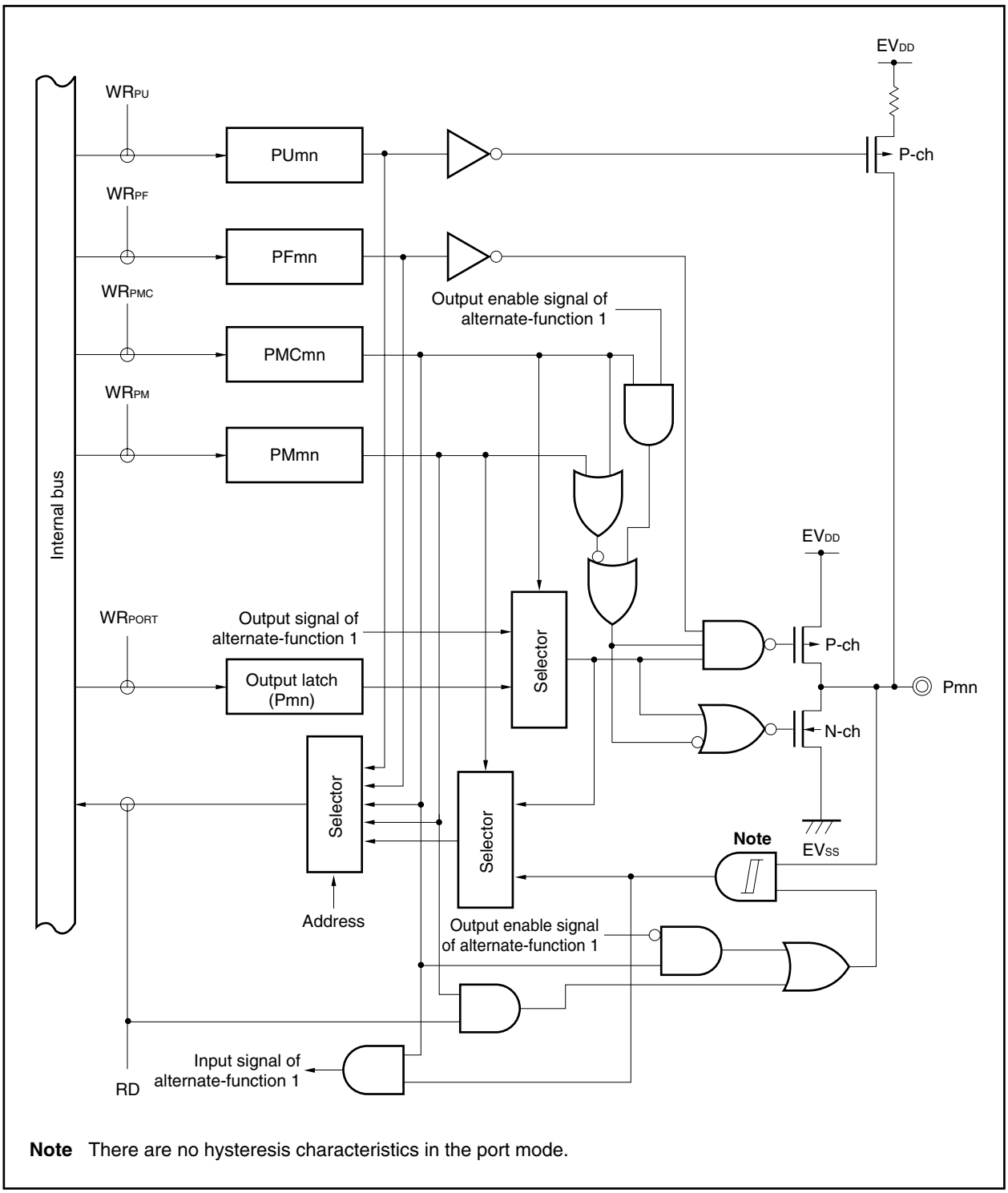
Downloaded from Arrow.com.

Figure 4-14. Block Diagram of Type F-1



Downloaded from Arrow.com.

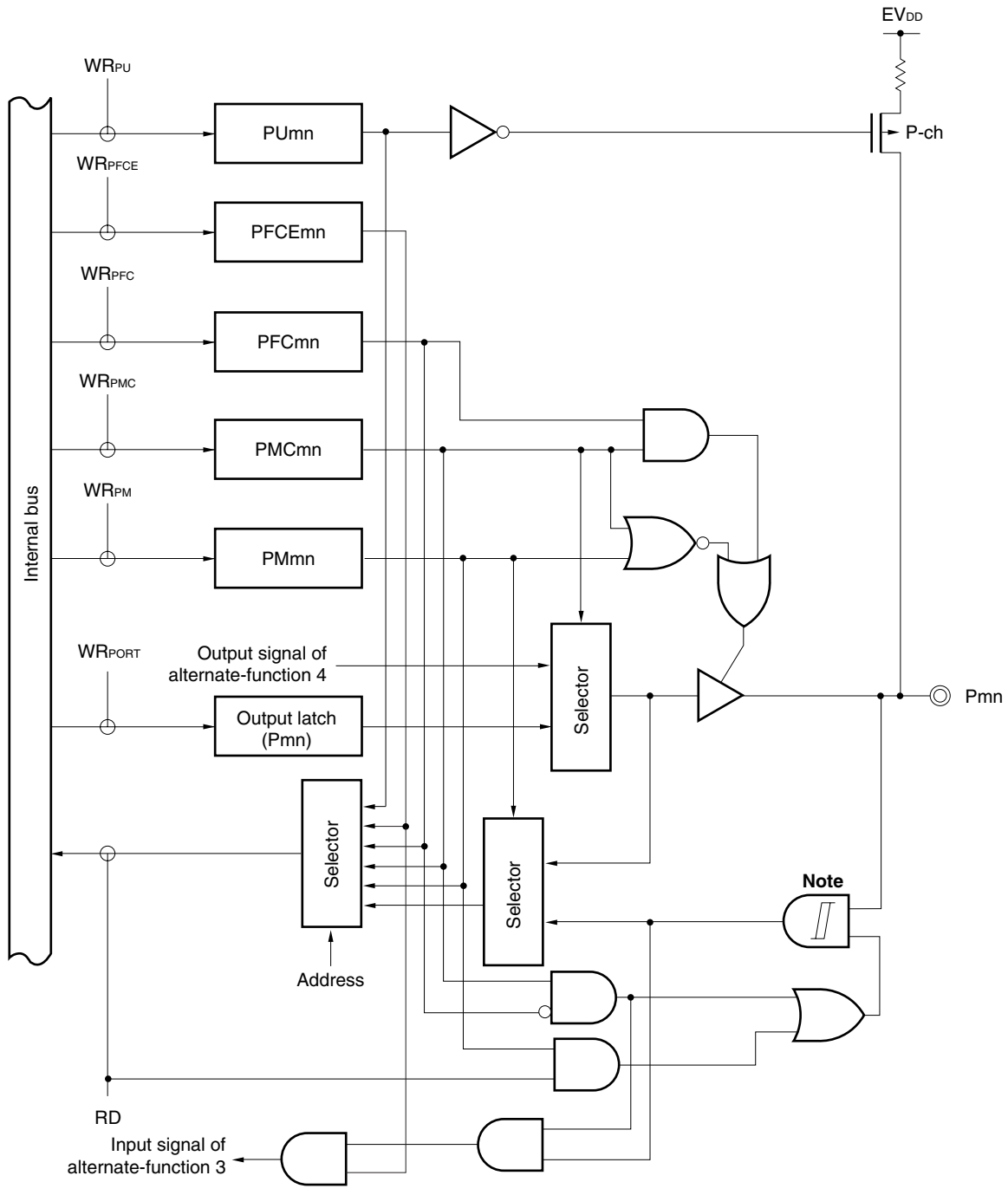
Figure 4-15. Block Diagram of Type F-2



Note There are no hysteresis characteristics in the port mode.

Downloaded from Arrow.com.

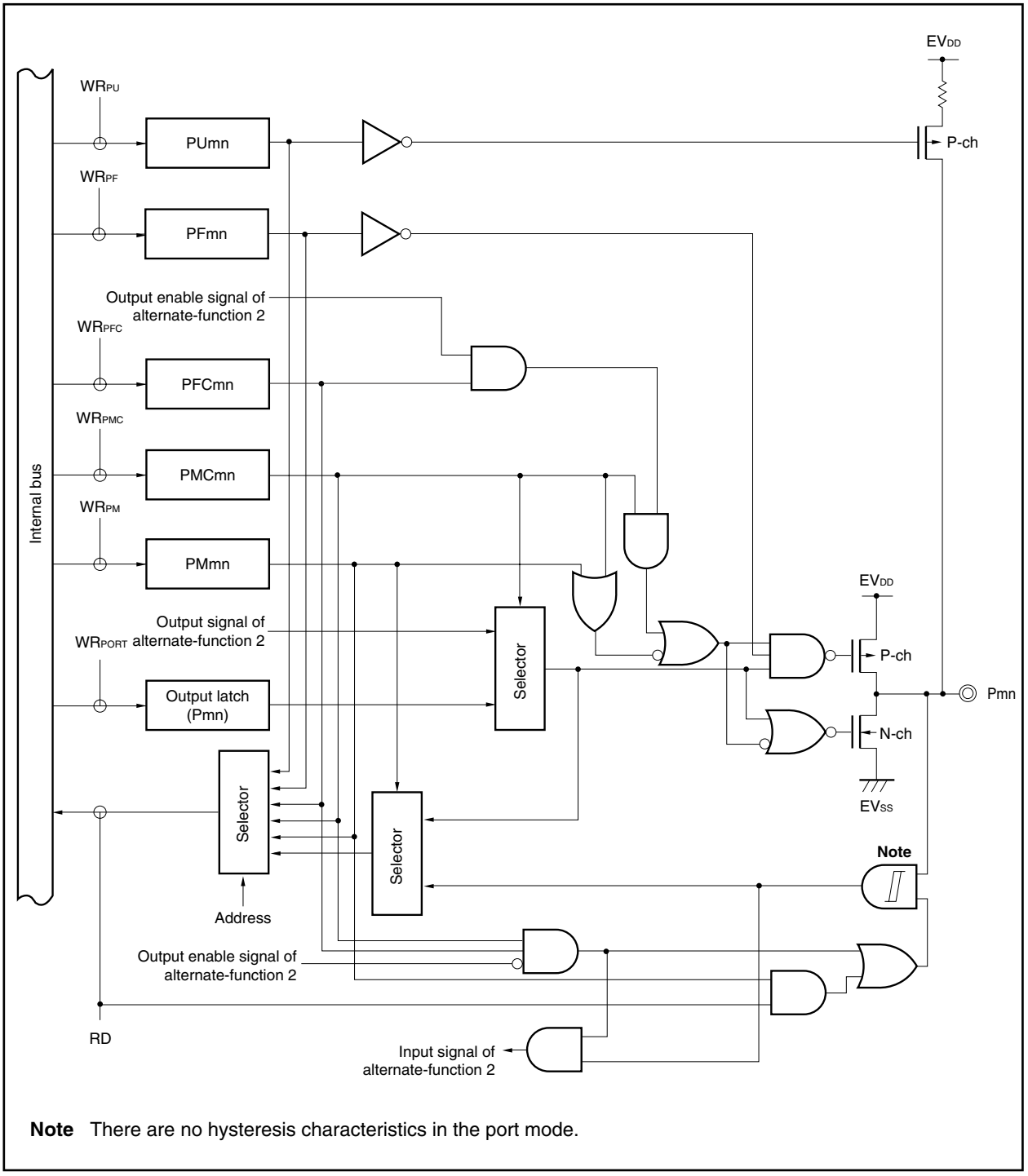
Figure 4-16. Block Diagram of Type G-7-3



Note There are no hysteresis characteristics in the port mode.

Downloaded from Arrow.com.

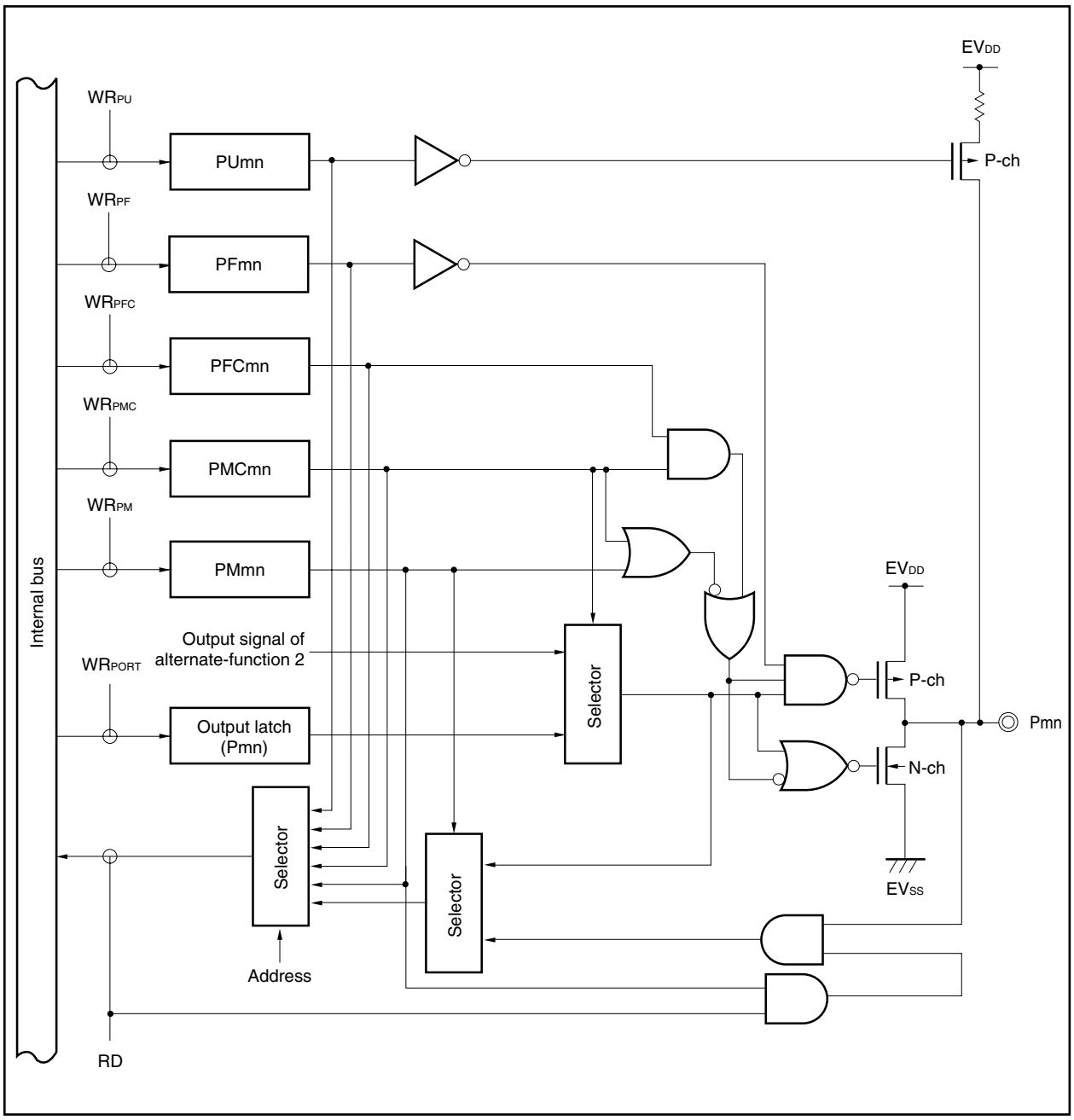
Figure 4-17. Block Diagram of Type G-8



Note There are no hysteresis characteristics in the port mode.

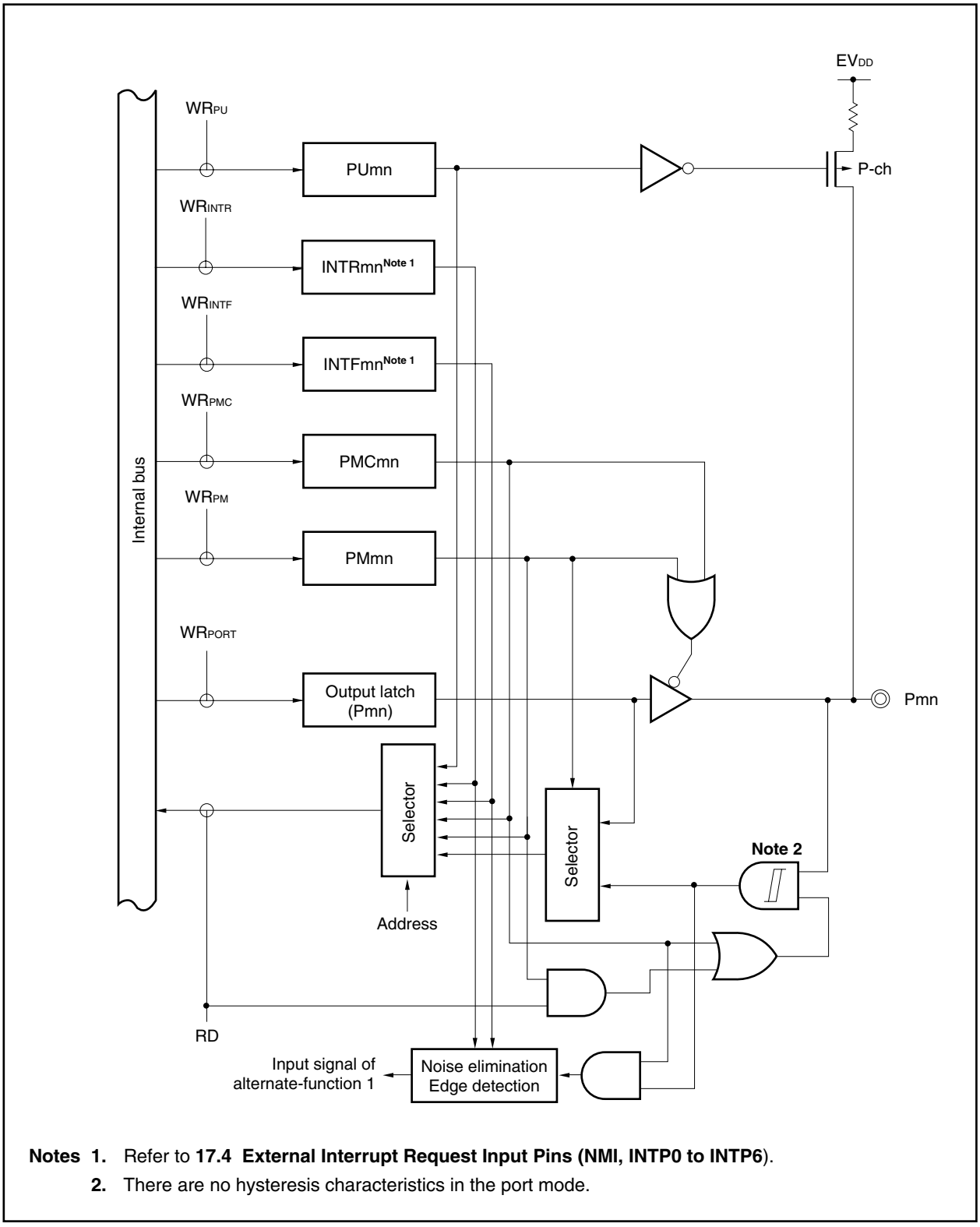
Downloaded from Arrow.com.

Figure 4-18. Block Diagram of Type G-9



Downloaded from Arrow.com.

Figure 4-19. Block Diagram of Type H-1

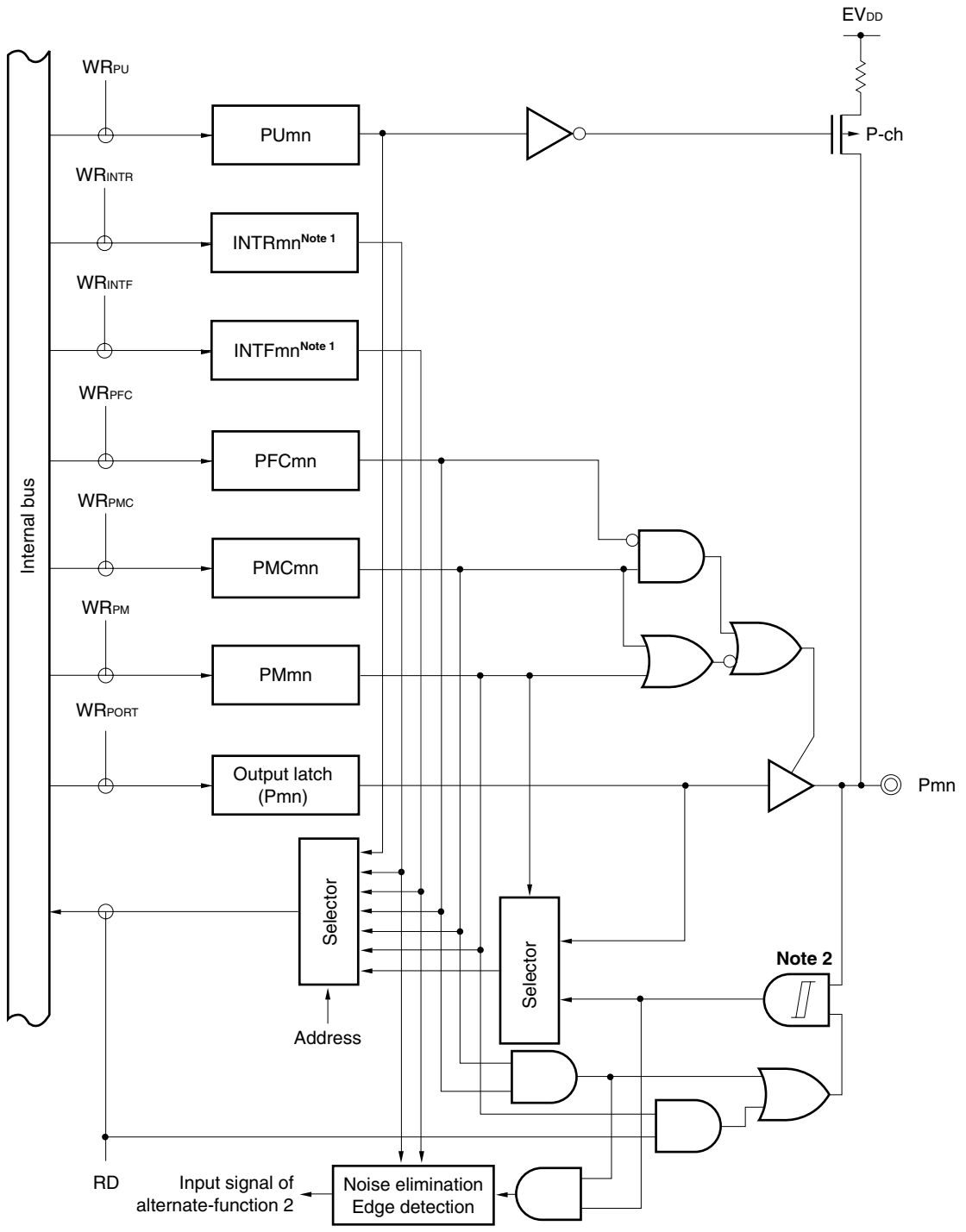


Notes 1. Refer to **17.4 External Interrupt Request Input Pins (NMI, INTP0 to INTP6)**.

2. There are no hysteresis characteristics in the port mode.

Downloaded from Arrow.com.

Figure 4-20. Block Diagram of Type H-3

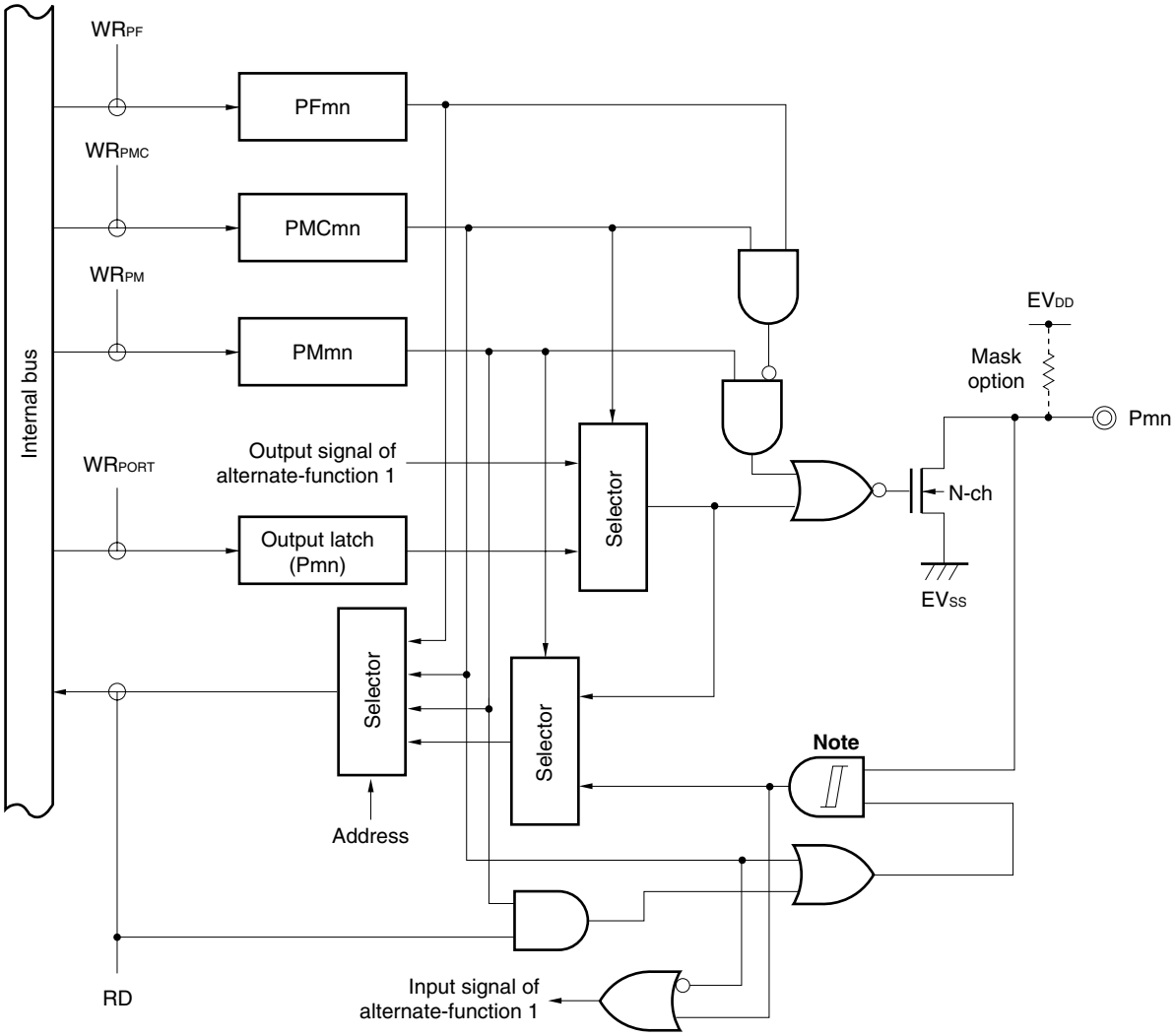


Notes 1. Refer to **17.4 External Interrupt Request Input Pins (NMI, INTP0 to INTP6)**.

2. There are no hysteresis characteristics in the port mode.

Downloaded from Arrow.com.

Figure 4-21. Block Diagram of Type K



Note There are no hysteresis characteristics in the port mode.

4.5 Port Register Setting When Alternate Function Is Used

Table 4-12 shows the port register settings when each port is used for an alternate function.

When using a port pin as an alternate-function pin, refer to description of each pin.

Downloaded from Arrow.com.

Downloaded from Arrow.com.

Table 4-12. Settings When Port Pins Are Used for Alternate Functions (1/3)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Function Name	I/O					
P00	TOH0	Output	P00 = Setting not required	PM00 = Setting not required	PMC00 = 1	–	–
P01	TOH1	Output	P01 = Setting not required	PM01 = Setting not required	PMC01 = 1	–	–
P02	NMI	Input	P02 = Setting not required	PM02 = Setting not required	PMC02 = 1	–	–
P03	INTP0	Input	P03 = Setting not required	PM03 = Setting not required	PMC03 = 1	–	–
P04	INTP1	Input	P04 = Setting not required	PM04 = Setting not required	PMC04 = 1	–	–
P05	INTP2	Input	P05 = Setting not required	PM05 = Setting not required	PMC05 = 1	–	–
P06	INTP3	Input	P06 = Setting not required	PM06 = Setting not required	PMC06 = 1	–	–
P30	TXD0	Output	P30 = Setting not required	PM30 = Setting not required	PMC30 = 1	–	–
P31	RXD0	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	–
P32	ASCK0	Input	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	–	–
P33	TIP00	Input	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	PFC33 = 0	PFCE33 (PFCE3) = 1
	TOP00	Output	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	PFC33 = 1	PFCE33 (PFCE3) = 1
P34	TIP01	Input	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	PFC34 = 0	PFCE34 (PFCE3) = 1
	TOP01	Output	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	PFC34 = 1	PFCE34 (PFCE3) = 1
P35	TI010	Input	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	PFC35 = 0	–
	TO01	Output	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	PFC35 = 1	–
P38	SDA0[Note]	I/O	P38 = 1	PM38 = Setting not required	PMC38 = 1	–	PF38 (PF3H) = 1
P39	SCL0[Note]	I/O	P39 = 1	PM39 = Setting not required	PMC39 = 1	–	PF39 (PF3H) = 1
P40	SI00	Input	P40 = Setting not required	PM40 = Setting not required	PMC40 = 1	–	–
P41	SO00	Output	P41 = Setting not required	PM41 = Setting not required	PMC41 = 1	–	PF41 (PF4) = Don't care
P42	$\overline{\text{SCK00}}$	I/O	P42 = Setting not required	PM42 = Setting not required	PMC42 = 1	–	PF42 (PF4) = Don't care

Note Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

Table 4-12. Settings When Port Pins Are Used for Alternate Functions (2/3)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Function Name	I/O					
P50	TI011	Input	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFC50 = 0	–
	RTP00	Output	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFC50 = 1	–
	KR0	Input	P50 = Setting not required	PM50 = 1	PMC50 = 0	PFC50 = Setting not required	KRM0 (KRM) = 1
P51	TI50	Input	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFC51 = 0	–
	RTP01	Output	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFC51 = 1	–
	KR1	Input	P51 = Setting not required	PM51 = 1	PMC51 = 0	PFC51 = Setting not required	KRM1 (KRM) = 1
P52	TO50	Output	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFC52 = 0	–
	RTP02	Output	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFC52 = 1	–
	KR2	Input	P52 = Setting not required	PM52 = 1	PMC52 = 0	PFC52 = Setting not required	KRM2 (KRM) = 1
P53	RTP03	Output	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFC53 = 1	–
	KR3	Input	P53 = Setting not required	PM53 = 1	PMC53 = 0	PFC53 = Setting not required	KRM3 (KRM) = 1
P54	RTP04	Output	P54 = Setting not required	PM54 = Setting not required	PMC54 = 1	PFC54 = 1	–
	KR4	Input	P54 = Setting not required	PM54 = 1	PMC54 = 0	PFC54 = Setting not required	KRM4 (KRM) = 1
P55	RTP05	Output	P55 = Setting not required	PM55 = Setting not required	PMC55 = 1	PFC55 = 1	–
	KR5	Input	P55 = Setting not required	PM55 = 1	PMC55 = 0	PFC55 = Setting not required	KRM5 (KRM) = 1
P70	ANI0	Input	P70 = Setting not required	–	–	–	–
P71	ANI1	Input	P71 = Setting not required	–	–	–	–
P72	ANI2	Input	P72 = Setting not required	–	–	–	–
P73	ANI3	Input	P73 = Setting not required	–	–	–	–
P74	ANI4	Input	P74 = Setting not required	–	–	–	–
P75	ANI5	Input	P75 = Setting not required	–	–	–	–
P76	ANI6	Input	P76 = Setting not required	–	–	–	–
P77	ANI7	Input	P77 = Setting not required	–	–	–	–

Downloaded from Arrow.com.

Table 4-12. Settings When Port Pins Are Used for Alternate Functions (3/3)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Function Name	I/O					
P90	TXD1	Output	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFC90 = 1	–
	KR6	Input	P90 = Setting not required	PM90 = 1	PMC90 = 0	PFC90 = Setting not required	KRM6 (KRM) = 1
P91	RXD1	Input	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFC91 = 1	–
	KR7	Input	P91 = Setting not required	PM91 = 1	PMC91 = 0	PFC91 = Setting not required	KRM7 (KRM) = 1
P96	TI51	Input	P96 = Setting not required	PM96 = 1	PMC96 = 0	PFC96 = Setting not required	–
	TO51	Output	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFC96 = 1	–
P97	SI01	Input	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFC97 = 1	–
P98	SO01	Output	P98 = Setting not required	PM98 = Setting not required	PMC98 = 1	PFC98 = 1	PF98 (PF9) = Don't care
P99	$\overline{\text{SCK01}}$	I/O	P99 = Setting not required	PM99 = Setting not required	PMC99 = 1	PFC99 = 1	PF98 (PF9) = Don't care
P913	INTP4	Input	P913 = Setting not required	PM913 = Setting not required	PMC913 = 1	PFC913 = 1	–
P914	INTP5	Input	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFC914 = 1	–
P915	INTP6	Input	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFC915 = 1	–
PCM1	CLKOUT	Output	PCM1 = Setting not required	PMCM1 = Setting not required	PMCCM1 = 1	–	–

4.6 Cautions

4.6.1 Cautions on bit manipulation instruction for port n register (Pn)

When a 1-bit manipulation instruction is executed on a port that provides both input and output functions, the value of the output latch of an input port that is not subject to manipulation may be written in addition to the targeted bit.

Therefore, it is recommended to rewrite the output latch when switching a port from input mode to output mode.

<Example> When PDL0 is an output port, PDL1 to PDL7 are input ports (all pin statuses are high level), and the value of the port latch is 00H, if the output of output port PDL0 is changed from low level to high level via a bit manipulation instruction, the value of the port latch is FFH.

Explanation: The targets of writing to and reading from the Pn register of a port whose PMnm bit is 1 are the output latch and pin status, respectively.

A bit manipulation instruction is executed in the following order in the V850ES/KE1.

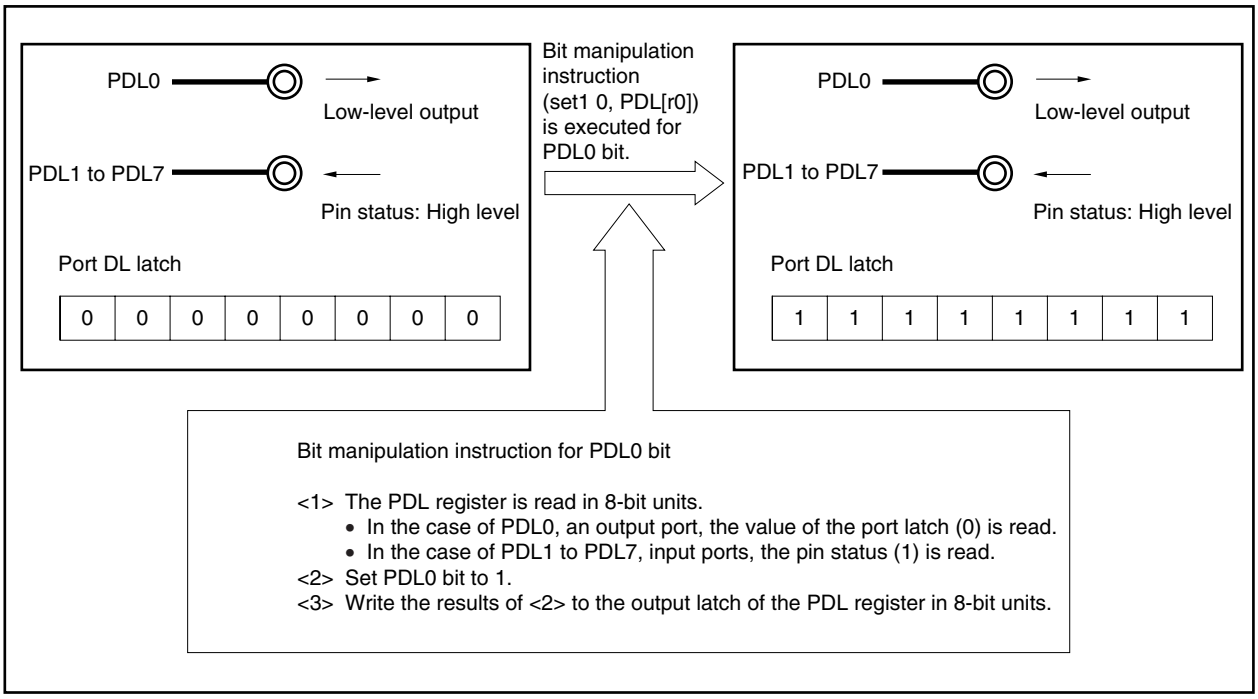
<1> The Pn register is read in 8-bit units.
<2> The targeted one bit is manipulated.
<3> The Pn register is written in 8-bit units.

In step <1>, the value of the output latch (0) of PDL0, which is an output port, is read, while the pin statuses of PDL1 to PDL7, which are input ports, are read. If the pin statuses of PDL1 to PDL7 are high level at this time, the read value is FEH.

The value is changed to FFH by the manipulation in <2>.

FFH is written to the output latch by the manipulation in <3>.

Figure 4-22. Bit Manipulation Instruction (PDL0)



Downloaded from Arrow.com.

4.6.2 Hysteresis characteristics

In port mode, the following ports do not have hysteresis characteristics.

P02 to P06
P31 to P35, P38, P39
P40, P42
P97, P99, P913 to P915

Downloaded from Arrow.com.

CHAPTER 5 CLOCK GENERATION FUNCTION

5.1 Overview

The following clock generation functions are available.

<R>

- ○ Main clock oscillator
  <In PLL (×4) mode>
  - f_X = 2 to 5 MHz (f_{XX} = 8 to 20 MHz: 4.5 V ≤ V_{DD} ≤ 5.5 V)
  - f_X = 2 to 4 MHz (f_{XX} = 8 to 16 MHz: 4.0 V ≤ V_{DD} ≤ 5.5 V)
  - f_X = 2 to 2.5 MHz (f_{XX} = 8 to 10 MHz: 2.7 V ≤ V_{DD} ≤ 5.5 V)

  <In clock through mode>
  - f_X = 2 to 10 MHz (f_{XX} = 2 to 10 MHz: 2.7 V ≤ V_{DD} ≤ 5.5 V)

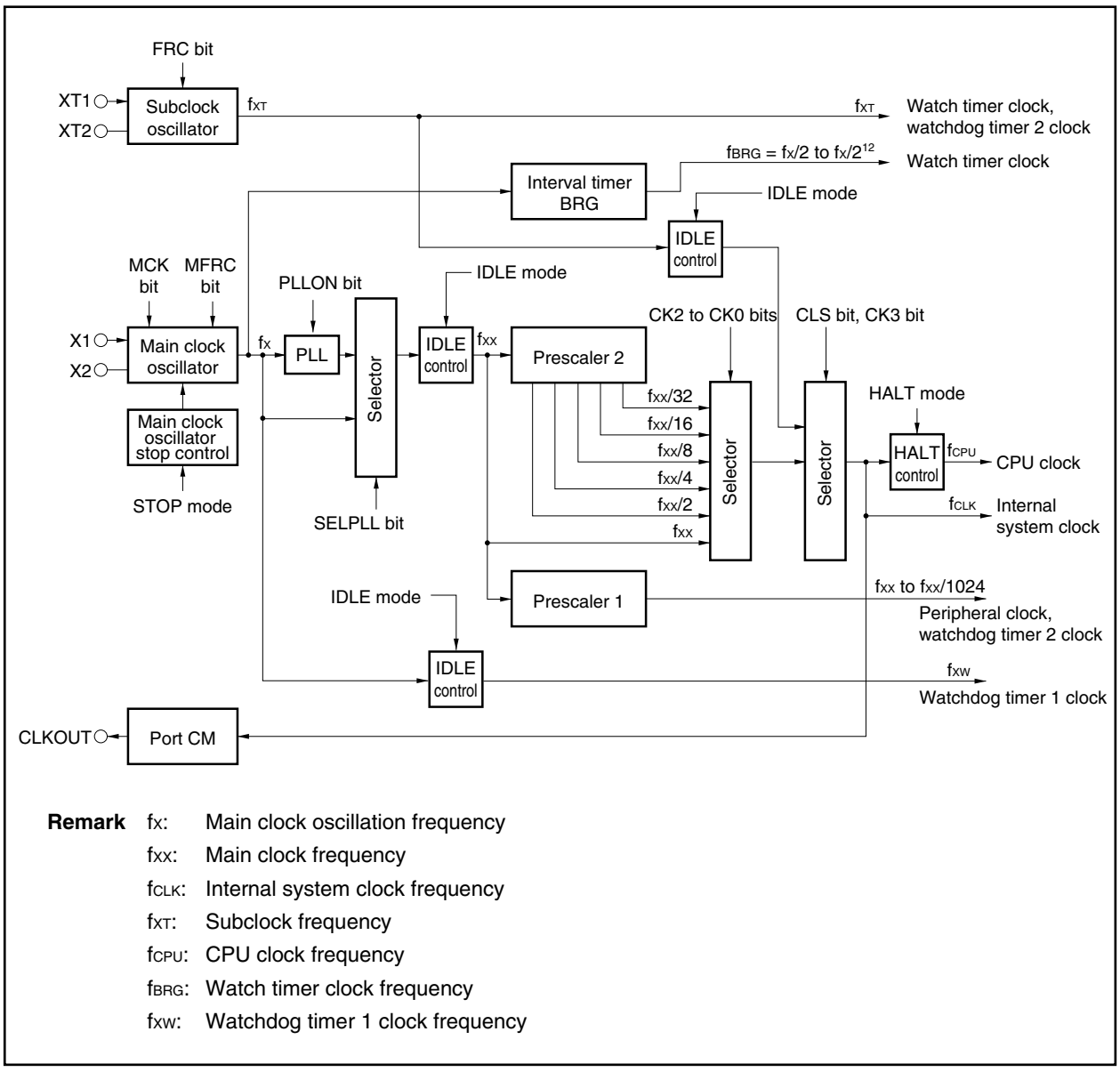
- ○ Subclock oscillator
  - f_{XT} = 32.768 kHz
- ○ Multiplication (×4) function by PLL (Phase Locked Loop)
  - Clock-through mode/PLL mode selectable
  - Usable voltage: V_{DD} = 2.7 to 5.5 V
- ○ Internal system clock generation
  - 7 steps (f_{XX}, f_{XX}/2, f_{XX}/4, f_{XX}/8, f_{XX}/16, f_{XX}/32, f_{XT})
- ○ Peripheral clock generation
- ○ Clock output function

Remark f_X: Main clock oscillation frequency
f_{XX}: Main clock frequency
f_{XT}: Subclock frequency

Downloaded from Arrow.com.

5.2 Configuration

Figure 5-1. Clock Generator



Remark f_X: Main clock oscillation frequency
f_{XX}: Main clock frequency
f_{CLK}: Internal system clock frequency
f_{XT}: Subclock frequency
f_{CPU}: CPU clock frequency
f_{BRG}: Watch timer clock frequency
f_{XW}: Watchdog timer 1 clock frequency

Downloaded from Arrow.com.

<R> **(1) Main clock oscillator**

The main clock oscillator oscillates the following frequencies (f_X).

- f_X = 2 to 5 MHz (V_{DD} = 4.5 to 5.5 V, in PLL mode)
- f_X = 2 to 4 MHz (V_{DD} = 4.0 to 5.5 V, in PLL mode)
- f_X = 2 to 2.5 MHz (V_{DD} = 2.7 to 5.5 V, in PLL mode)
- f_X = 2 to 10 MHz (V_{DD} = 2.7 to 5.5 V, in clock-through mode)

(2) Subclock oscillator

The subclock oscillator oscillates a frequency of 32.768 kHz (f_{XT}).

(3) Main clock oscillator stop control

This circuit generates a control signal that stops oscillation of the main clock oscillator.

Oscillation of the main clock oscillator is stopped in the STOP mode or when the PCC.MCK bit = 1 (valid only when the PCC.CLS bit = 1).

(4) Prescaler 1

This prescaler generates the clock (f_{XX} to f_{XX}/1024) to be supplied to the following on-chip peripheral functions: TMP0, TM01, TM50, TM51, TMH0, TMH1, CSI00, CSI01, UART0, UART1, I^2C0, ADC, and WDT2

(5) Prescaler 2

This circuit divides the main clock (f_{XX}).

The clock generated by prescaler 2 (f_{XX} to f_{XX}/32) is supplied to the selector that generates the CPU clock (f_{CPU}) and internal system clock (f_{CLK}).

f_{CLK} is the clock supplied to the INTC, ROM correction, ROM, and RAM blocks, and can be output from the CLKOUT pin.

(6) Interval timer BRG

This circuit divides the clock (f_X) generated by the main clock oscillator to a specific frequency (32.768 kHz) and supplies that clock to the watch timer block.

For details, refer to **CHAPTER 10 INTERVAL TIMER, WATCH TIMER**.

(7) PLL

This circuit multiplies the clock (f_X) generated by the main clock oscillator.

It operates in two modes: clock-through mode in which f_X is output as is, and PLL mode in which a multiplied clock is output. These modes can be selected by using the PLLCTL.SELPLL bit.

Operation of the PLL can be started or stopped by the PLLCTL.PLLON bit.

Downloaded from Arrow.com.

5.3 Register

(1) Processor clock control register (PCC)

The PCC register is a special register. Data can be written to this register only in combination of specific sequences (refer to **3.4.7 Special registers**).

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

Reset sets PCC to 03H.

(1/2)

After reset: 03H R/W Address: FFFFF828H

	7	<6>	5	<4>	<3>	2	1	0
PCC	FRC	MCK	MFRC	CLS[Note]	CK3	CK2	CK1	CK0

FRC	Use of subclock on-chip feedback resistor
0	Used
1	Not used

MCK	Control of main clock oscillator
0	Oscillation enabled
1	Oscillation stopped

- Even if the MCK bit is set to 1 while the system is operating with the main clock as the CPU clock, the operation of the main clock does not stop. It stops after the CPU clock has been changed to the subclock.
- When the main clock is stopped and the device is operating on the subclock, clear the MCK bit to 0 and wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.

MFRC	Use of main clock on-chip feedback resistor
0	Used
1	Not used

CLS[Note]	Status of CPU clock (f_{CPU})
0	Main clock operation
1	Subclock operation

Note The CLS bit is a read-only bit.

Downloaded from Arrow.com.

(2/2)

CK3	CK2	CK1	CK0	Clock selection (f_{CLK}/f_{CPU})
0	0	0	0	f_{XX}
0	0	0	1	$f_{XX}/2$
0	0	1	0	$f_{XX}/4$
0	0	1	1	$f_{XX}/8$ (default value)
0	1	0	0	$f_{XX}/16$
0	1	0	1	$f_{XX}/32$
0	1	1	×	Setting prohibited
1	×	×	×	f_{XT}

Cautions 1. Do not change the CPU clock (by using the CK3 to CK0 bits) while CLKOUT is being output.

2. Use a bit manipulation instruction to manipulate the CK3 bit. When using an 8-bit manipulation instruction, do not change the set values of the CK2 to CK0 bits.

3. When the CPU operates on the subclock and no clock is input to the X1 pin, do not access a register in which a wait occurs using an access method that causes a wait (refer to 3.4.8 (1) (b) Access to special on-chip peripheral I/O register for details of the access methods). If a wait occurs, it can only be released by a reset.

Remark ×: don't care

Downloaded from Arrow.com.

(a) Example of setting main clock operation → subclock operation

<1> CK3 bit ← 1: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<2> Subclock operation: Read the CLS bit to check if subclock operation has started. It takes the following time after the CK3 bit is set until subclock operation is started.

Max.: $1/f_{XT}$ (1/subclock frequency)

<3> MCK bit ← 1: Set the MCK bit to 1 only when stopping the main clock.

Cautions 1. When stopping the main clock, stop the PLL.

2. If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied, then change to the subclock operation mode.

Internal system clock (f_{CLK}) > Subclock (f_{XT}: 32.768 kHz) × 4

Remark Internal system clock (f_{CLK}): Clock generated from the main clock (f_{XX}) by setting bits CK2 to CK0

[Description example]

```
<1> _SET_SUB_RUN :
    st.b        r0, PRCMD[r0]
    set1        3, PCC[r0]          -- CK3 bit ← 1
<2> _CHECK_CLS :
    tst1        4, PCC[r0]          -- Wait until subclock operation starts.
    bz          _CHECK_CLS
<3> _STOP_MAIN_CLOCK :
    st.b        r0, PRCMD[r0]
    set1        6, PCC[r0]          -- MCK bit ← 1, main clock is stopped
```

Remark The above description is an example. Note with caution that the CLS bit is read in a closed loop in <2>.

Downloaded from Arrow.com.

(b) Example of setting subclock operation → main clock operation

<1> MCK bit ← 0: Main clock starts oscillating

<2> Insert waits by the program and wait until the oscillation stabilization time of the main clock elapses.

<3> CK3 bit ← 0: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<4> Main clock operation: It takes the following time after the CK3 bit is set until main clock operation is started.

Max.: 1/f_{XT} (1/subclock frequency)

Therefore, insert one NOP instruction immediately after setting the CK3 bit to 0 or read the CLS bit to check if main clock operation has started.

[Description example]

```
<1> _START_MAIN_OSC :
    st.b        r0, PRCMD[r0]          -- Release of protection of special registers
    clr1        6, PCC[r0]             -- Main clock starts oscillating
<2> movea       0x55, r0, r11          -- Wait for oscillation stabilization time
    _WAIT_OST :
    nop
    nop
    nop
    addi        -1, r11, r11
    mp          r0, r11
    bne         _PROGRAM_WAIT
<3> st.b        r0, PRCMD[r0]
    clr1        3, PCC[r0]             -- CK3 ← 0
<4> _CHECK_CLS :
    tst1        4, PCC[r0]             -- Wait until main clock operation starts
    bnz         _CHECK_CLS
```

Remark The above description is an example. Note with caution that the CLS bit is read in a closed loop in <4>.

Downloaded from Arrow.com.

5.4 Operation

5.4.1 Operation of each clock

The following table shows the operation status of each clock.

Table 5-1. Operation Status of Each Clock

Register Setting and Operation Status / Target Clock	PCC Register								
	CLS bit = 0, MCK bit = 0					CLS bit = 1, MCK bit = 0		CLS bit = 1, MCK bit = 1	
	During reset	During oscillation stabilization time count	HALT mode	IDLE mode	STOP mode	Subclock mode	Sub-IDLE mode	Subclock mode	Sub-IDLE mode
Main clock oscillator (f_X)	×	○	○	○	×	○	○	×	×
Subclock oscillator (f_{XT})	○	○	○	○	○	○	○	○	○
CPU clock (f_{CPU})	×	×	×	×	×	○	×	○	×
Internal system clock (f_{CLK})	×	×	○	×	×	○	×	○	×
Peripheral clock (f_{XX} to f_{XX}/1024)	×	×	○	×	×	○	×	×	×
WT clock (main)	×	○	○	○	×	○	○	×	×
WT clock (sub)	○	○	○	○	○	○	○	○	○
WDT1 clock (f_{XW})	×	○	○	○	×	○	○	×	×
WDT2 clock (main)	×	×	○	×	×	○	×	×	×
WDT2 clock (sub)	○	○	○	○	○	○	○	○	○

Remark ○: Operable
×: Stopped

5.4.2 Clock output function

The clock output function is used to output the internal system clock (f_{CLK}) from the CLKOUT pin.

The internal system clock (f_{CLK}) is selected by using the PCC.CK3 to PCC.CK0 bits.

The CLKOUT pin functions alternately as the PCM1 pin and functions as a clock output pin if so specified by the control register of port CM.

The status of the CLKOUT pin is the same as the internal system clock in Table 5-1 and the pin can output the clock when it is in the operable status. It outputs a low level in the stopped status. However, the port mode (PCM1: input mode) is selected until the CLKOUT pin output is set after reset. Consequently, the CLKOUT pin goes into a high-impedance state.

5.4.3 External clock input function

An external clock can be directly input to the oscillator. Input the clock to the X1 pin and its inverse signal to the X2 pin. Set the PCC.MFRC bit to 1 (on-chip feedback resistor not used). Note, however, that oscillation stabilization time is inserted even in the external clock mode.

Downloaded from Arrow.com.

5.5 PLL Function

5.5.1 Overview

The PLL function is used to output the operating clock of the CPU and on-chip peripheral function at a frequency 4 times higher than the oscillation frequency, and select the clock-through mode.

When PLL function is used: Input clock = 2 to 5 MHz (f_{XX}: 8 to 20 MHz)
Clock-through mode: Input clock = 2 to 10 MHz (f_{XX}: 2 to 10 MHz)

5.5.2 Register

(1) PLL control register (PLLCTL)

The PLLCTL register is an 8-bit register that controls the security function of PLL and RTO.
This register can be read or written in 8-bit or 1-bit units.
Reset sets PLLCTL to 01H.

After reset: 01H R/W Address: FFFFF806H

	7	6	5	4	3	<2>	<1>	<0>
PLLCTL	0	0	0	0	0	RTOST0[Note]	SELPLL	PLLON

PLLON	PLL operation control
0	PLL stopped
1	PLL operating

SELPLL	PLL clock slelection
0	Clock-through operation
1	PLL operation

Note For the RTOST0 bit, refer to **CHAPTER 12 REAL-TIME OUTPUT FUNCTION (RTO)**.

Caution Be sure to clear bits 4 to 7 to "0". Changing bit 3 does not affect the operation.

Downloaded from Arrow.com.

5.5.3 Usage

(1) When PLL is used

- After reset has been released, the PLL operates (PLLCTL.PLLON bit = 1), but because the default mode is the clock-through mode (PLLCTL.SELPLL bit = 0), select the PLL mode (SELPLL bit = 1).
- To set the STOP mode in which the main clock is stopped, or to set the IDLE mode, first select the clock-through mode and then stop the PLL. To return from the IDLE or STOP mode, first enable PLL operation (PLLON bit = 1), and then select the PLL mode (SELPLL bit = 1).
- To enable the PLL operation, first set the PLLON bit to 1, wait for 200 μs, and then set the SELPLL bit to 1. To stop the PLL, first select the clock-through mode (SELPLL bit = 0), wait for 8 clocks or more, and then stop the PLL (PLLON bit = 0).

(2) When PLL is not used

- The clock-through mode (SELPLL bit = 0) is selected after reset has been released, but the PLL is operating (PLLON bit = 1) and must therefore be stopped (PLLON bit = 0).

Remark The PLL is operable in the IDLE mode. To realize low power consumption, stop the PLL. Be sure to stop the PLL when shifting to the STOP mode.

Downloaded from Arrow.com.

CHAPTER 6 16-BIT TIMER/EVENT COUNTER P (TMP)

Timer P (TMP) is a 16-bit timer/event counter.

6.1 Overview

An outline of TMP0 is shown below.

- Clock selection: 8 ways
- Capture trigger input pins: 2
- External event count input pins: 1
- External trigger input pins: 1
- Timer/counters: 1
- Capture/compare registers: 2
- Capture/compare match interrupt request signals: 2
- Timer output pins: 2

6.2 Functions

TMP0 has the following functions.

- Interval timer
- External event counter
- External trigger pulse output
- One-shot pulse output
- PWM output
- Free-running timer
- Pulse width measurement

Downloaded from Arrow.com.

6.3 Configuration

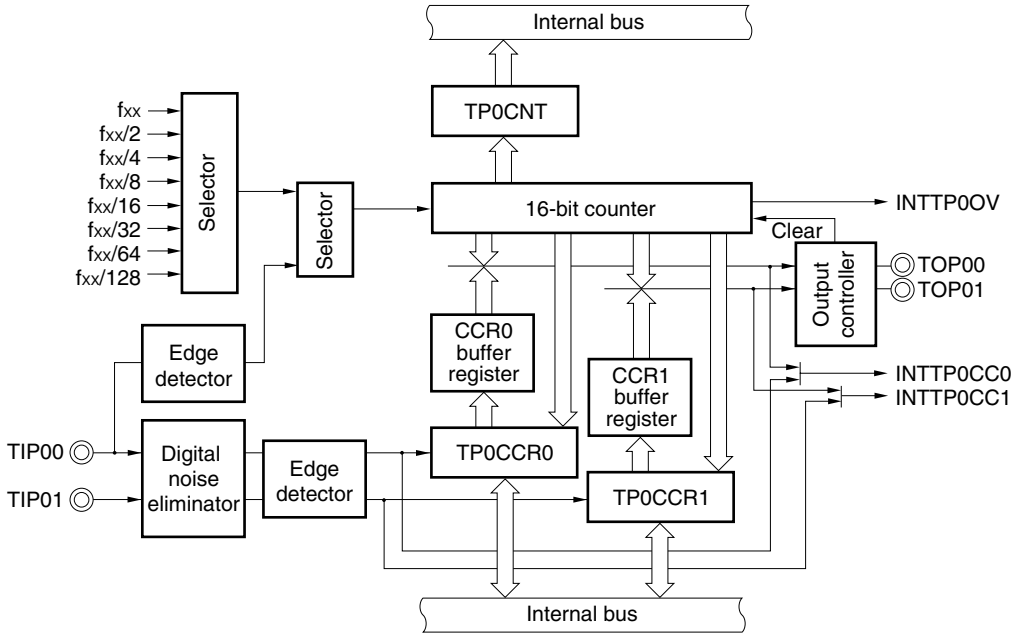
TMP0 includes the following hardware.

Table 6-1. Configuration of TMP0

Item	Configuration
Timer register	16-bit counter
Registers	TMP0 capture/compare registers 0, 1 (TP0CCR0, TP0CCR1) TMP0 counter read buffer register (TP0CNT) CCR0, CCR1 buffer registers
Timer inputs	2 (TIP00[Note], TIP01 pins)
Timer outputs	2 (TOP00, TOP01 pins)
Control registers	TMP0 control registers 0, 1 (TP0CTL0, TP0CTL1) TMP0 I/O control registers 0 to 2 (TP0IOC0 to TP0IOC2) TMP0 option register 0 (TP0OPT0)

Note The TIP00 pin functions alternately as a capture trigger input signal, external event count input signal, and external trigger input signal.

Figure 6-1. Block Diagram of TMP0



Remark fxx: Main clock frequency

Downloaded from Arrow.com.

(1) 16-bit counter

This 16-bit counter can count internal clocks or external events.

The count value of this counter can be read by using the TP0CNT register.

When the TP0CTL0.TP0CE bit = 0, the value of the 16-bit counter is FFFFH. If the TP0CNT register is read at this time, 0000H is read.

Reset sets the TP0CE bit to 0. Therefore, the 16-bit counter is set to FFFFH.

(2) CCR0 buffer register

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TP0CCR0 register is used as a compare register, the value written to the TP0CCR0 register is transferred to the CCR0 buffer register. When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTP0CC0) is generated.

The CCR0 buffer register cannot be read or written directly.

The CCR0 buffer register is cleared to 0000H after reset, as the TP0CCR0 register is cleared to 0000H.

(3) CCR1 buffer register

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TP0CCR1 register is used as a compare register, the value written to the TP0CCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTP0CC1) is generated.

The CCR1 buffer register cannot be read or written directly.

The CCR1 buffer register is cleared to 0000H after reset, as the TP0CCR1 register is cleared to 0000H.

(4) Edge detector

This circuit detects the valid edges input to the TIP00 and TIP01 pins. No edge, rising edge, falling edge, or both the rising and falling edges can be selected as the valid edge by using the TP0IOC1 and TP0IOC2 registers.

(5) Output controller

This circuit controls the output of the TOP00 and TOP01 pins. The output controller is controlled by the TP0IOC0 register.

(6) Selector

This selector selects the count clock for the 16-bit counter. Eight types of internal clocks or an external event can be selected as the count clock.

(7) Digital noise eliminator

This circuit is valid only when the TIP0a pin is used as a capture trigger input pin.

This circuit is controlled by the TIP0a noise elimination register (PaNFC).

Remark a = 0, 1

Downloaded from Arrow.com.

6.4 Registers

(1) TMP0 control register 0 (TP0CTL0)

The TP0CTL0 register is an 8-bit register that controls the operation of TMP0.

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

Reset sets this register to 00H.

The same value can always be written to the TP0CTL0 register by software.

After reset: 00H R/W Address: FFFFF5A0H

	<7>	6	5	4	3	2	1	0
TP0CTL0	TP0CE	0	0	0	0	TP0CKS2	TP0CKS1	TP0CKS0

TP0CE	TMP0 operation control
0	TMP0 operation disabled (TMP0 reset asynchronously[Note]).
1	TMP0 operation enabled. TMP0 operation started.

TP0CKS2	TP0CKS1	TP0CKS0	Internal count clock selection
0	0	0	f_{XX}
0	0	1	$f_{XX}/2$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/8$
1	0	0	$f_{XX}/16$
1	0	1	$f_{XX}/32$
1	1	0	$f_{XX}/64$
1	1	1	$f_{XX}/128$

Note TP0OPT0.TP0OVF bit, 16-bit counter, timer output (TOP00, TOP01 pins)

Cautions 1. Set the TP0CKS2 to TP0CKS0 bits when the TP0CE bit = 0. When the value of the TP0CE bit is changed from 0 to 1, the TP0CKS2 to TP0CKS0 bits can be set simultaneously.

2. Be sure to clear bits 3 to 6 to "0".

Remark f_{XX}: Main clock frequency

Downloaded from Arrow.com.

(2) TMP0 control register 1 (TP0CTL1)

The TP0CTL1 register is an 8-bit register that controls the operation of TMP0.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF5A1H

	7	<6>	<5>	4	3	2	1	0
TP0CTL1	0	TP0EST	TP0EEE	0	0	TP0MD2	TP0MD1	TP0MD0

TP0EST	Software trigger control
0	–
1	Generate a valid signal for external trigger input. • In one-shot pulse output mode: A one-shot pulse is output with writing 1 to the TP0EST bit as the trigger. • In external trigger pulse output mode: A PWM waveform is output with writing 1 to the TP0EST bit as the trigger.

TP0EEE	Count clock selection
0	Disable operation with external event count input. (Perform counting with the count clock selected by the TP0CTL0.TP0CK0 to TP0CTL0.TP0CK2 bits.)
1	Enable operation with external event count input. (Perform counting at the valid edge of the external event count input signal.)
The TP0EEE bit selects whether counting is performed with the internal count clock or the valid edge of the external event count input.	

TP0MD2	TP0MD1	TP0MD0	Timer mode selection
0	0	0	Interval timer mode
0	0	1	External event count mode
0	1	0	External trigger pulse output mode
0	1	1	One-shot pulse output mode
1	0	0	PWM output mode
1	0	1	Free-running timer mode
1	1	0	Pulse width measurement mode
1	1	1	Setting prohibited

Cautions 1. The TP0EST bit is valid only in the external trigger pulse output mode or one-shot pulse output mode. In any other mode, writing 1 to this bit is ignored.

2. External event count input is selected in the external event count mode regardless of the value of the TP0EEE bit.

3. Set the TP0EEE and TP0MD2 to TP0MD0 bits when the TP0CTL0.TP0CE bit = 0. (The same value can be written when the TP0CE bit = 1.) The operation is not guaranteed when rewriting is performed with the TP0CE bit = 1. If rewriting was mistakenly performed, clear the TP0CE bit to 0 and then set the bits again.

4. Be sure to clear bits 3, 4, and 7 to "0".

Downloaded from Arrow.com.

(3) TMP0 I/O control register 0 (TP0IOC0)

The TP0IOC0 register is an 8-bit register that controls the timer output (TOP00, TOP01 pins).

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF5A2H

	7	6	5	4	3	<2>	1	<0>
TP0IOC0	0	0	0	0	TP0OL1	TP0OE1	TP0OL0	TP0OE0

TP0OL1	TOP01 pin output level setting[Note]
0	TOP01 pin high-level start
1	TOP01 pin low-level start

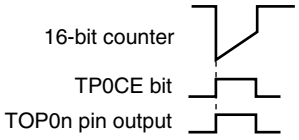
TP0OE1	TOP01 pin output setting
0	Timer output disabled • When TP0OL1 bit = 0: Low level is output from the TOP01 pin • When TP0OL1 bit = 1: High level is output from the TOP01 pin
1	Timer output enabled (a square wave is output from the TOP01 pin).

TP0OL0	TOP00 pin output level setting[Note]
0	TOP00 pin high-level start
1	TOP00 pin low-level start

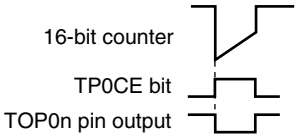
TP0OE0	TOP00 pin output setting
0	Timer output disabled • When TP0OL0 bit = 0: Low level is output from the TOP00 pin • When TP0OL0 bit = 1: High level is output from the TOP00 pin
1	Timer output enabled (a square wave is output from the TOP00 pin).

Note The output level of the timer output pin (TOP0n) specified by the TP0OLn bit is shown below (n = 0, 1).

• When TP0OLn bit = 0



• When TP0OLn bit = 1



Cautions 1. Rewrite the TP0OL1, TP0OE1, TP0OL0, and TP0OE0 bits when the TP0CTL0.TP0CE bit = 0. (The same value can be written when the TP0CE bit = 1.) If rewriting was mistakenly performed, clear the TP0CE bit to 0 and then set the bits again.

2. Even if the TP0OLa bit is manipulated when the TP0CE and TP0OEa bits are 0, the TOP0a pin output level varies (a = 0, 1).

Downloaded from Arrow.com.

(4) TMP0 I/O control register 1 (TP0IOC1)

The TP0IOC1 register is an 8-bit register that controls the valid edge of the capture trigger input signals (TIP00, TIP01 pins).

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF5A3H

	7	6	5	4	3	2	1	0
TP0IOC1	0	0	0	0	TP0IS3	TP0IS2	TP0IS1	TP0IS0

TP0IS3	TP0IS2	Capture trigger input signal (TIP01 pin) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TP0IS1	TP0IS0	Capture trigger input signal (TIP00 pin) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TP0IS3 to TP0IS0 bits when the TP0CTL0.TP0CE bit = 0. (The same value can be written when the TP0CE bit = 1.) If rewriting was mistakenly performed, clear the TP0CE bit to 0 and then set the bits again.

2. The TP0IS3 to TP0IS0 bits are valid only in the free-running timer mode and the pulse width measurement mode. In all other modes, a capture operation is not possible.

Downloaded from Arrow.com.

(5) TMP0 I/O control register 2 (TP0IOC2)

The TP0IOC2 register is an 8-bit register that controls the valid edge of the external event count input signal (TIP00 pin) and external trigger input signal (TIP00 pin).

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF5A4H

	7	6	5	4	3	2	1	0
TP0IOC2	0	0	0	0	TP0EES1	TP0EES0	TP0ETS1	TP0ETS0

TP0EES1	TP0EES0	External event count input signal (TIP00 pin) valid edge setting
0	0	No edge detection (external event count invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TP0ETS1	TP0ETS0	External trigger input signal (TIP00 pin) valid edge setting
0	0	No edge detection (external trigger invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Cautions 1. Rewrite the TP0EES1, TP0EES0, TP0ETS1, and TP0ETS0 bits when the TP0CTL0.TP0CE bit = 0. (The same value can be written when the TP0CE bit = 1.) If rewriting was mistakenly performed, clear the TP0CE bit to 0 and then set the bits again.

2. The TP0EES1 and TP0EES0 bits are valid only when the TP0CTL1.TP0EEE bit = 1 or when the external event count mode (TP0CTL1.TP0MD2 to TP0CTL1.TP0MD0 bits = 001) has been set.

<R>

3. The TP0ETS1 and TP0ETS0 bits are valid only when the external trigger pulse output mode (TP0CTL1.TP0MD2 to TP0CTL1.TP0MD0 bits = 010) or the one-shot pulse output mode (TP0CTL1.TP0MD2 to TP0CTL1.TP0MD0 bits = 011) is set.

Downloaded from Arrow.com.

(6) TMP0 option register 0 (TP0OPT0)

The TP0OPT0 register is an 8-bit register used to set the capture/compare operation and detect an overflow.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF5A5H

	7	6	5	4	3	2	1	<0>
TP0OPT0	0	0	TP0CCS1	TP0CCS0	0	0	0	TP0OVF

TP0CCS1	TP0CCR1 register capture/compare selection
0	Compare register selected
1	Capture register selected
The TP0CCS1 bit setting is valid only in the free-running timer mode.	

TP0CCS0	TP0CCR0 register capture/compare selection
0	Compare register selected
1	Capture register selected
The TP0CCS0 bit setting is valid only in the free-running timer mode.	

TP0OVF	TMP0 overflow detection flag
Set (1)	Overflow occurred
Reset (0)	TP0OVF bit 0 written or TP0CTL0.TP0CE bit = 0
• The TP0OVF bit is set when the 16-bit counter count value overflows from FFFFH to 0000H in the free-running timer mode or the pulse width measurement mode. • An interrupt request signal (INTTP0OV) is generated at the same time that the TP0OVF bit is set to 1. The INTTP0OV signal is not generated in modes other than the free-running timer mode and the pulse width measurement mode. • The TP0OVF bit is not cleared even when the TP0OVF bit or the TP0OPT0 register are read when the TP0OVF bit = 1. • The TP0OVF bit can be both read and written, but the TP0OVF bit cannot be set to 1 by software. Writing 1 has no influence on the operation of TMP0.	

Cautions 1. Rewrite the TP0CCS1 and TP0CCS0 bits when the TP0CE bit = 0. (The same value can be written when the TP0CE bit = 1.) If rewriting was mistakenly performed, clear the TP0CE bit to 0 and then set the bits again.

2. Be sure to clear bits 1 to 3, 6, and 7 to "0".

Downloaded from Arrow.com.

(7) TMP0 capture/compare register 0 (TP0CCR0)

The TP0CCR0 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TP0OPT0.TP0CCS0 bit. In the pulse width measurement mode, the TP0CCR0 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TP0CCR0 register can be read or written during operation.

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

Reset sets this register to 0000H.

Caution Accessing the TP0CCR0 register is disabled during subclock operation with the main clock stopped. For details, refer to 3.4.8 (1) (b).

After reset: 0000H R/W Address: FFFFF5A6H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TP0CCR0																

Downloaded from Arrow.com.

(a) Function as compare register

The TP0CCR0 register can be rewritten even when the TP0CTL0.TP0CE bit = 1.

The set value of the TP0CCR0 register is transferred to the CCR0 buffer register. When the value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTP0CC0) is generated. If TOP00 pin output is enabled at this time, the output of the TOP00 pin is inverted.

When the TP0CCR0 register is used as a cycle register in the interval timer mode, external event count mode, external trigger pulse output mode, one-shot pulse output mode, or PWM output mode, the value of the 16-bit counter is cleared (0000H) if its count value matches the value of the CCR0 buffer register.

(b) Function as capture register

When the TP0CCR0 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TP0CCR0 register if the valid edge of the capture trigger input pin (TIP00 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TP0CCR0 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIP00 pin) is detected.

Even if the capture operation and reading the TP0CCR0 register conflict, the correct value of the TP0CCR0 register can be read.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

Table 6-2. Function of Capture/Compare Register in Each Mode and How to Write Compare Register

Operation Mode	Capture/Compare Register	How to Write Compare Register
Interval timer	Compare register	Anytime write
External event counter	Compare register	Anytime write
External trigger pulse output	Compare register	Batch write
One-shot pulse output	Compare register	Anytime write
PWM output	Compare register	Batch write
Free-running timer	Capture/compare register	Anytime write
Pulse width measurement	Capture register	–

Downloaded from Arrow.com.

(8) TMP0 capture/compare register 1 (TP0CCR1)

The TP0CCR1 register can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TP0OPT0.TP0CCS1 bit. In the pulse width measurement mode, the TP0CCR1 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TP0CCR1 register can be read or written during operation.

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

Reset sets this register to 0000H.

Caution Accessing the TP0CCR1 register is disabled during subclock operation with the main clock stopped. For details, refer to 3.4.8 (1) (b).

After reset: 0000H R/W Address: FFFFF5A8H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TP0CCR1																

Downloaded from Arrow.com.

(a) Function as compare register

The TP0CCR1 register can be rewritten even when the TP0CTL0.TP0CE bit = 1.

The set value of the TP0CCR1 register is transferred to the CCR1 buffer register. When the value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTP0CC1) is generated. If TOP01 pin output is enabled at this time, the output of the TOP01 pin is inverted.

(b) Function as capture register

When the TP0CCR1 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TP0CCR1 register if the valid edge of the capture trigger input pin (TIP01 pin) is detected. In the pulse width measurement mode, the count value of the 16-bit counter is stored in the TP0CCR1 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIP01 pin) is detected.

Even if the capture operation and reading the TP0CCR1 register conflict, the correct value of the TP0CCR1 register can be read.

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

Table 6-3. Function of Capture/Compare Register in Each Mode and How to Write Compare Register

Operation Mode	Capture/Compare Register	How to Write Compare Register
Interval timer	Compare register	Anytime write
External event counter	Compare register	Anytime write
External trigger pulse output	Compare register	Batch write
One-shot pulse output	Compare register	Anytime write
PWM output	Compare register	Batch write
Free-running timer	Capture/compare register	Anytime write
Pulse width measurement	Capture register	–

Downloaded from Arrow.com.

(9) TMP0 counter read buffer register (TP0CNT)

The TP0CNT register is a read buffer register that can read the count value of the 16-bit counter.

If this register is read when the TP0CTL0.TP0CE bit = 1, the count value of the 16-bit timer can be read.

This register is read-only, in 16-bit units.

The value of the TP0CNT register is cleared to 0000H when the TP0CE bit = 0. If the TP0CNT register is read at this time, the value of the 16-bit counter (FFFFH) is not read, but 0000H is read.

The value of the TP0CNT register is cleared to 0000H after reset, as the TP0CE bit is cleared to 0.

Caution Accessing the TP0CNT register is disabled during subclock operation with the main clock stopped. For details, refer to 3.4.8 (1) (b).

After reset: 0000H R Address: FFFFF5AAH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TP0CNT																

Downloaded from Arrow.com.

6.5 Operation

TMP0 can perform the following operations.

Operation	TP0CTL1.TP0EST Bit (Software Trigger Bit)	TIP00 Pin (External Trigger Input)	Capture/Compare Register Setting	Compare Register Write
Interval timer mode	Invalid	Invalid	Compare only	Anytime write
External event count mode[Note 1]	Invalid	Invalid	Compare only	Anytime write
External trigger pulse output mode[Note 2]	Valid	Valid	Compare only	Batch write
One-shot pulse output mode[Note 2]	Valid	Valid	Compare only	Anytime write
PWM output mode	Invalid	Invalid	Compare only	Batch write
Free-running timer mode	Invalid	Invalid	Switching enabled	Anytime write
Pulse width measurement mode[Note 2]	Invalid	Invalid	Capture only	Not applicable

Notes 1. To use the external event count mode, specify that the valid edge of the TIP00 pin capture trigger input is not detected (by clearing the TP0IOC1.TP0IS1 and TP0IOC1.TP0IS0 bits to “00”).

2. When using the external trigger pulse output mode, one-shot pulse output mode, and pulse width measurement mode, select the internal clock as the count clock (by clearing the TP0CTL1.TP0EEE bit to 0).

Downloaded from Arrow.com.

6.5.1 Interval timer mode (TP0MD2 to TP0MD0 bits = 000)

In the interval timer mode, an interrupt request signal (INTTP0CC0) is generated at the specified interval if the TP0CTL0.TP0CE bit is set to 1. A square wave whose half cycle is equal to the interval can be output from the TOP00 pin.

Usually, the TP0CCR1 register is not used in the interval timer mode.

Figure 6-2. Configuration of Interval Timer

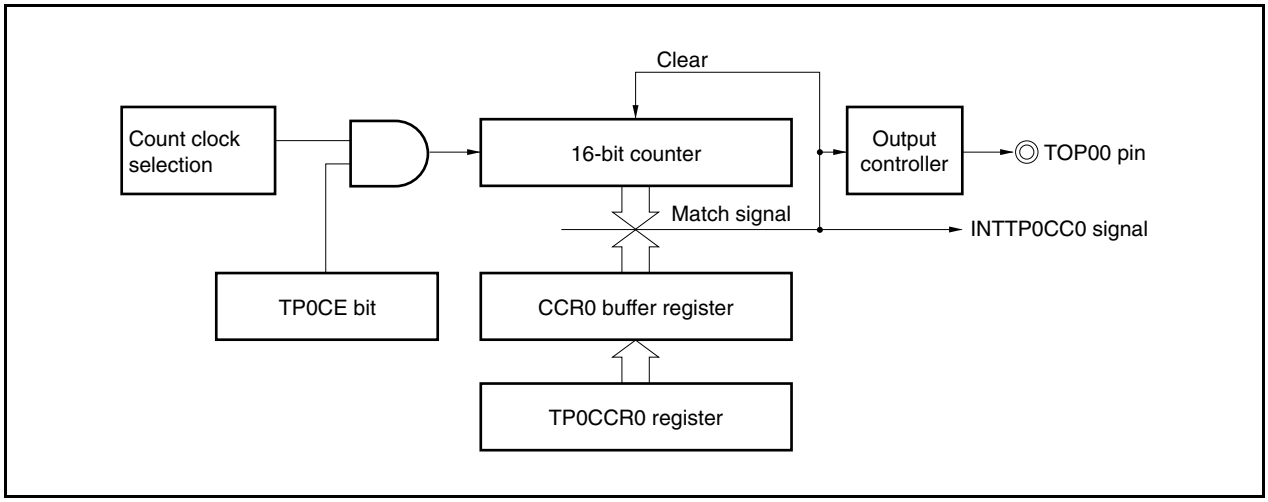
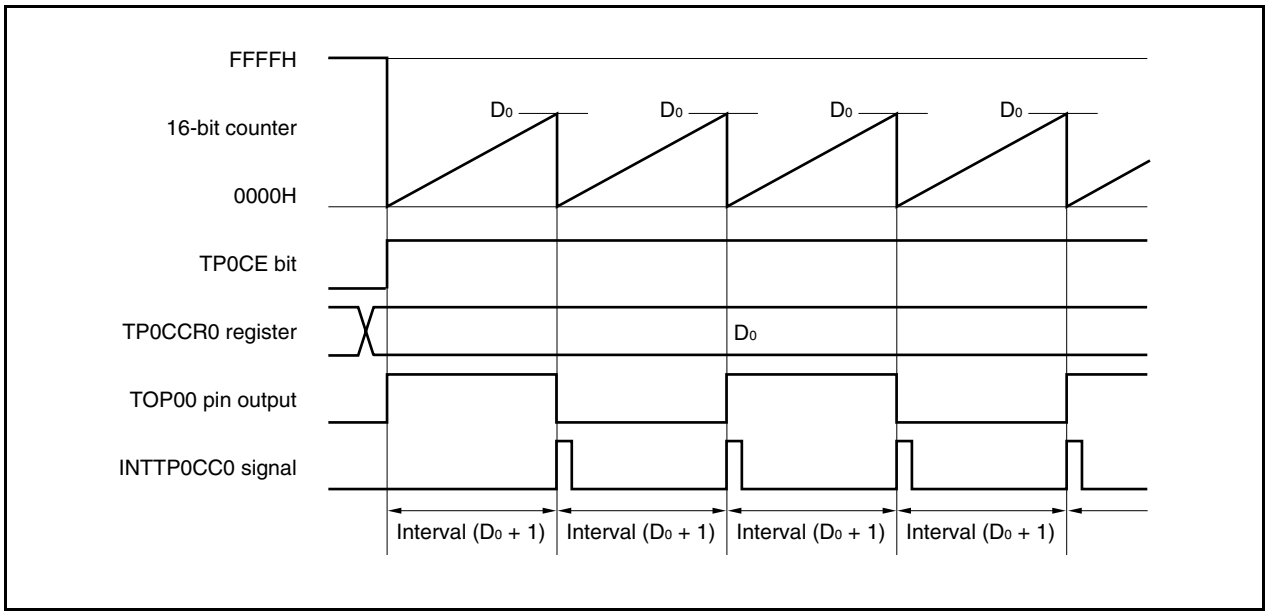


Figure 6-3. Basic Timing of Operation in Interval Timer Mode



Downloaded from Arrow.com.

When the TP0CE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting. At this time, the output of the TOP00 pin is inverted. Additionally, the set value of the TP0CCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, the output of the TOP00 pin is inverted, and a compare match interrupt request signal (INTTP0CC0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TP0CCR0 register + 1) × Count clock cycle

Figure 6-4. Register Setting for Interval Timer Mode Operation (1/2)

(a) TMP0 control register 0 (TP0CTL0)

	TP0CE					TP0CKS2	TP0CKS1	TP0CKS0
TP0CTL0	0/1	0	0	0	0	0/1	0/1	0/1

Select count clock

0: Stop counting
1: Enable counting

(b) TMP0 control register 1 (TP0CTL1)

<R>

			TP0EST	TP0EEE			TP0MD2	TP0MD1	TP0MD0
TP0CTL1	0	0	0	0/1[Note]	0	0	0	0	0

0, 0, 0:
Interval timer mode

0: Operate on count clock selected by TP0CKS0 to TP0CKS2 bits
1: Count with external event count input signal

(c) TMP0 I/O control register 0 (TP0IOC0)

					TP0OL1	TP0OE1	TP0OL0	TP0OE0
TP0IOC0	0	0	0	0	0/1	0/1	0/1	0/1

0: Disable TOP00 pin output
1: Enable TOP00 pin output

Setting of output level with operation of TOP00 pin disabled
0: Low level
1: High level

0: Disable TOP01 pin output
1: Enable TOP01 pin output

Setting of output level with operation of TOP01 pin disabled
0: Low level
1: High level

<R>

Note This bit can be set to 1 only when the interrupt request signals (INTTP0CC0 and INTTP0CC1) are masked by the interrupt mask flags (TP0CCMK0 and TP0CCMK1) and timer output (TOP01) is performed at the same time. However, set the TP0CCR0 and TP0CCR1 registers to the same value (refer to **6.5.1 (2) (d) Operation of TP0CCR1 register**).

Downloaded from Arrow.com.

Figure 6-4. Register Setting for Interval Timer Mode Operation (2/2)

(d) TMP0 counter read buffer register (TP0CNT)

By reading the TP0CNT register, the count value of the 16-bit counter can be read.

(e) TMP0 capture/compare register 0 (TP0CCR0)

If the TP0CCR0 register is set to D_0, the interval is as follows.

Interval = $(D_0 + 1) \times$ Count clock cycle

(f) TMP0 capture/compare register 1 (TP0CCR1)

Usually, the TP0CCR1 register is not used in the interval timer mode. However, the set value of the TP0CCR1 register is transferred to the CCR1 buffer register. A compare match interrupt request signal (INTTP0CC1) is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

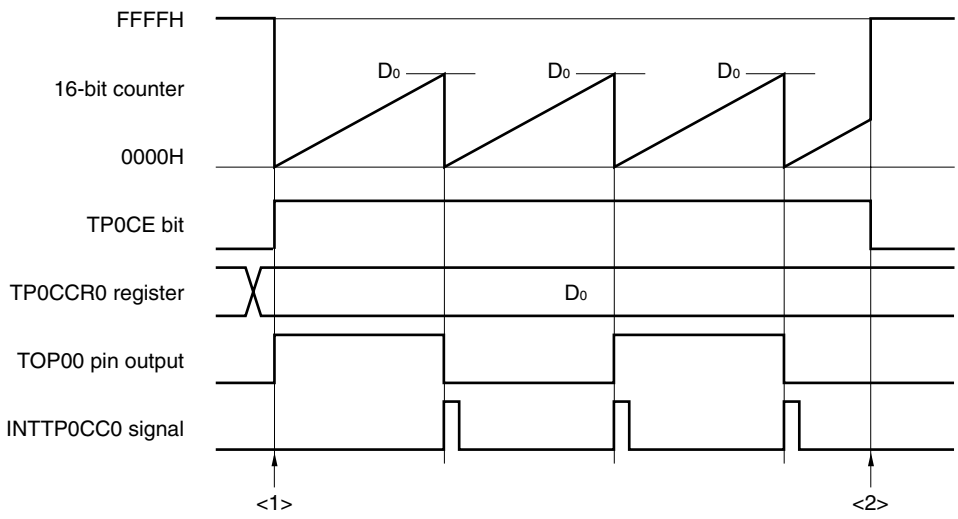
Therefore, mask the interrupt request by using the corresponding interrupt mask flag (TP0CCMK1).

Remark TMP0 I/O control register 1 (TP0IOC1), TMP0 I/O control register 2 (TP0IOC2), and TMP0 option register 0 (TP0OPT0) are usually not used in the interval timer mode.
However, set the TP0IOC2 register to use the external event count input.

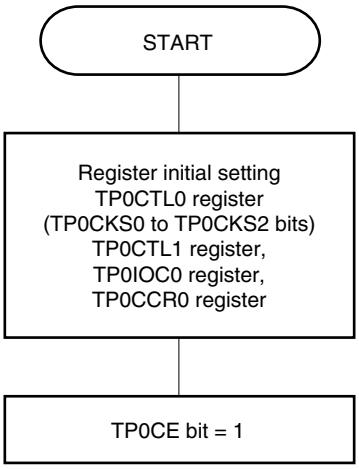
Downloaded from Arrow.com.

(1) Interval timer mode operation flow

Figure 6-5. Software Processing Flow in Interval Timer Mode



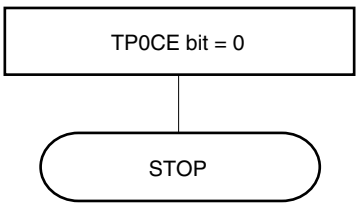
<1> Count operation start flow



Initial setting of these registers is performed before setting the TP0CE bit to 1.

The TP0CKS0 to TP0CKS2 bits can be set at the same time when counting has been started (TP0CE bit = 1).

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TP0CE bit to 0.

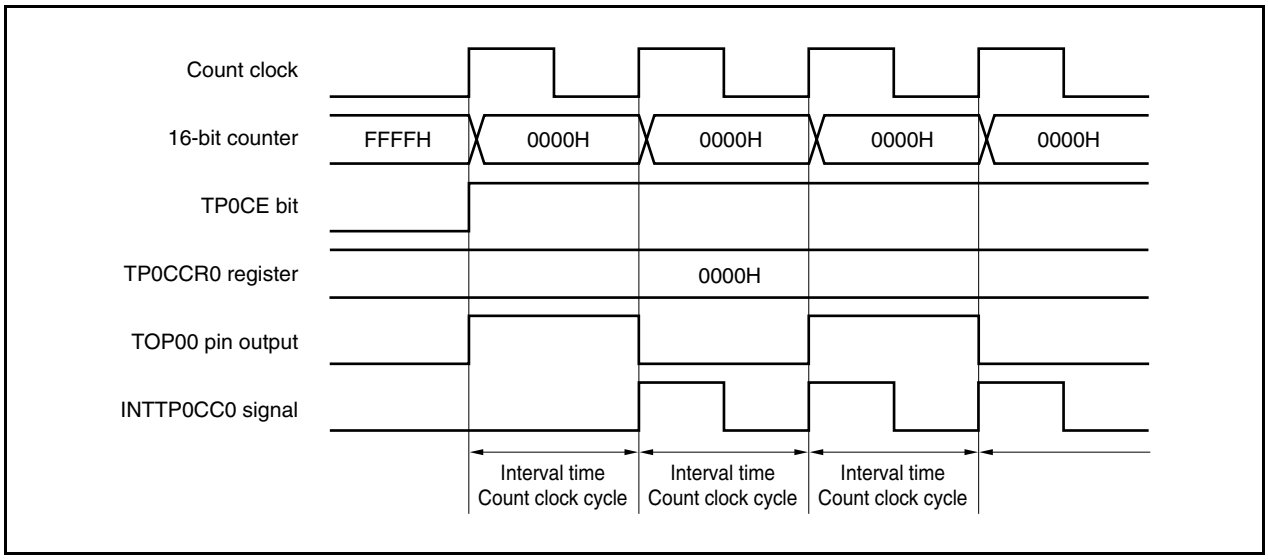
Downloaded from Arrow.com.

(2) Interval timer mode operation timing

(a) Operation if TP0CCR0 register is cleared to 0000H

If the TP0CCR0 register is cleared to 0000H, the INTTP0CC0 signal is generated at each count clock, and the output of the TOP00 pin is inverted.

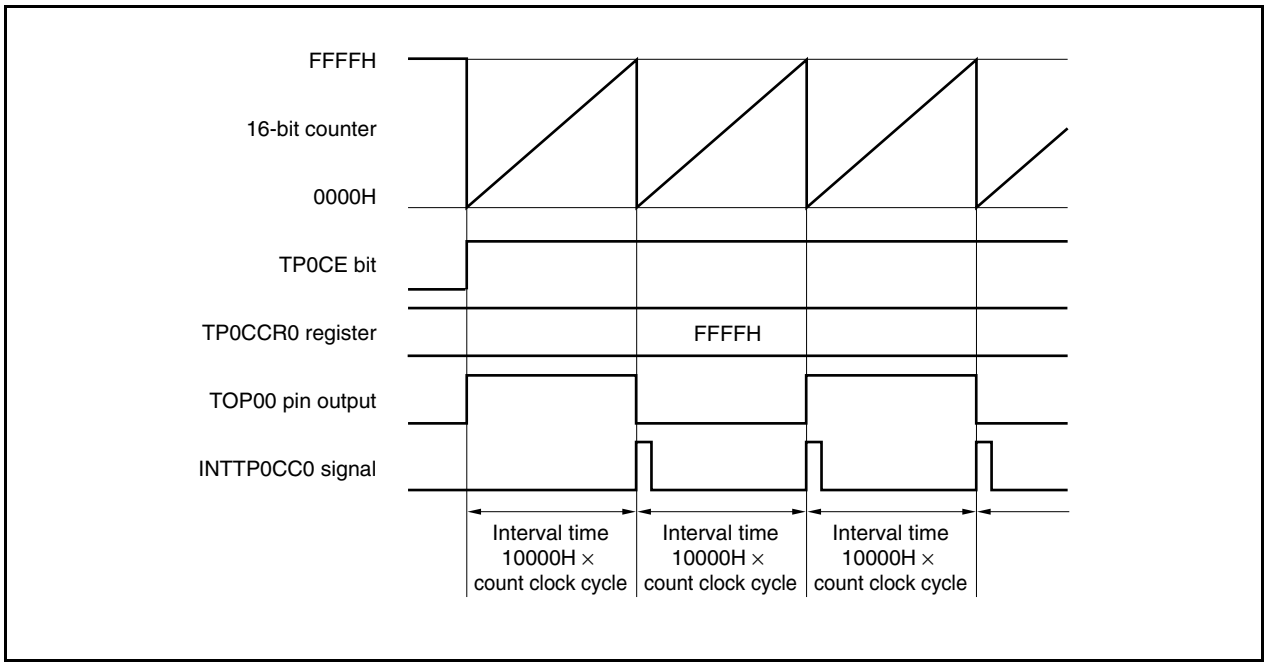
The value of the 16-bit counter is always 0000H.



Downloaded from Arrow.com.

(b) Operation if TP0CCR0 register is set to FFFFH

If the TP0CCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH. The counter is cleared to 0000H in synchronization with the next count-up timing. The INTTP0CC0 signal is generated and the output of the TOP00 pin is inverted. At this time, an overflow interrupt request signal (INTTP0OV) is not generated, nor is the overflow flag (TP0OPT0.TP0OVF bit) set to 1.

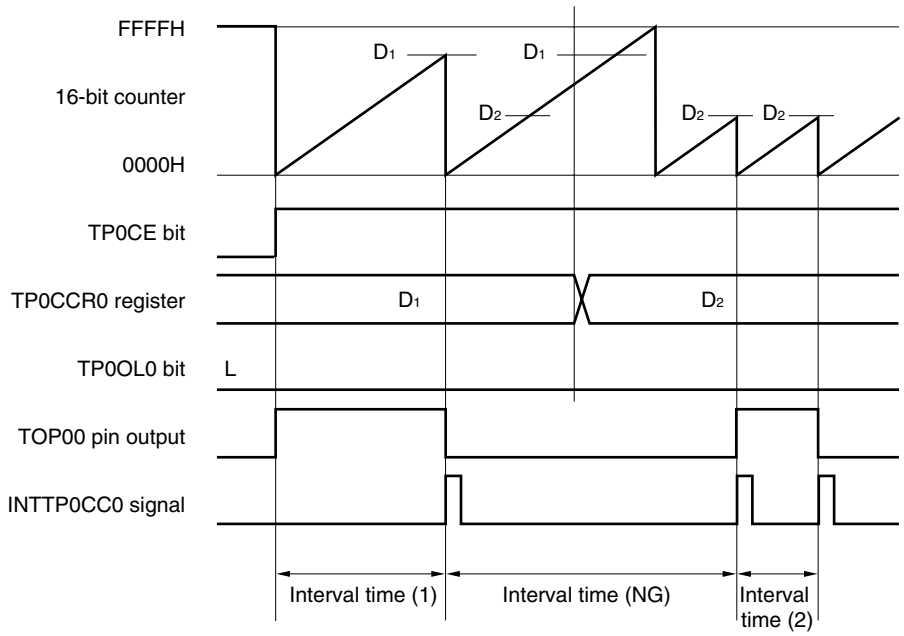


Downloaded from Arrow.com.

(c) Notes on rewriting TP0CCR0 register

To change the value of the TP0CCR0 register to a smaller value, stop counting once and then change the set value.

If the value of the TP0CCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



Remark Interval time (1): $(D_1 + 1) \times$ Count clock cycle
Interval time (NG): $(10000H + D_2 + 1) \times$ Count clock cycle
Interval time (2): $(D_2 + 1) \times$ Count clock cycle

If the value of the TP0CCR0 register is changed from D_1 to D_2 while the count value is greater than D_2 but less than D_1, the count value is transferred to the CCR0 buffer register as soon as the TP0CCR0 register has been rewritten. Consequently, the value of the 16-bit counter that is compared is D_2.

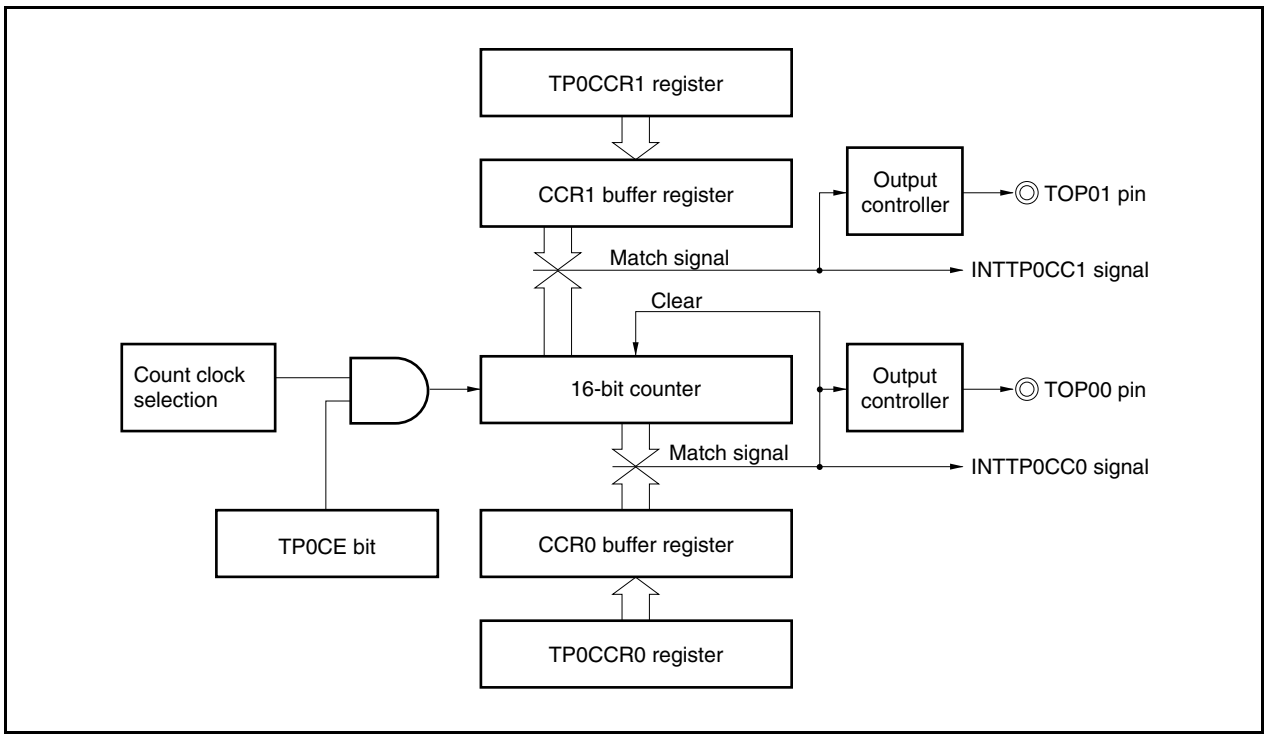
Because the count value has already exceeded D_2, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches D_2, the INTTP0CC0 signal is generated and the output of the TOP00 pin is inverted.

Therefore, the INTTP0CC0 signal may not be generated at the interval time "$(D_1 + 1) \times$ Count clock cycle" or "$(D_2 + 1) \times$ Count clock cycle" originally expected, but may be generated at an interval of "$(10000H + D_2 + 1) \times$ Count clock period".

Downloaded from Arrow.com.

(d) Operation of TP0CCR1 register

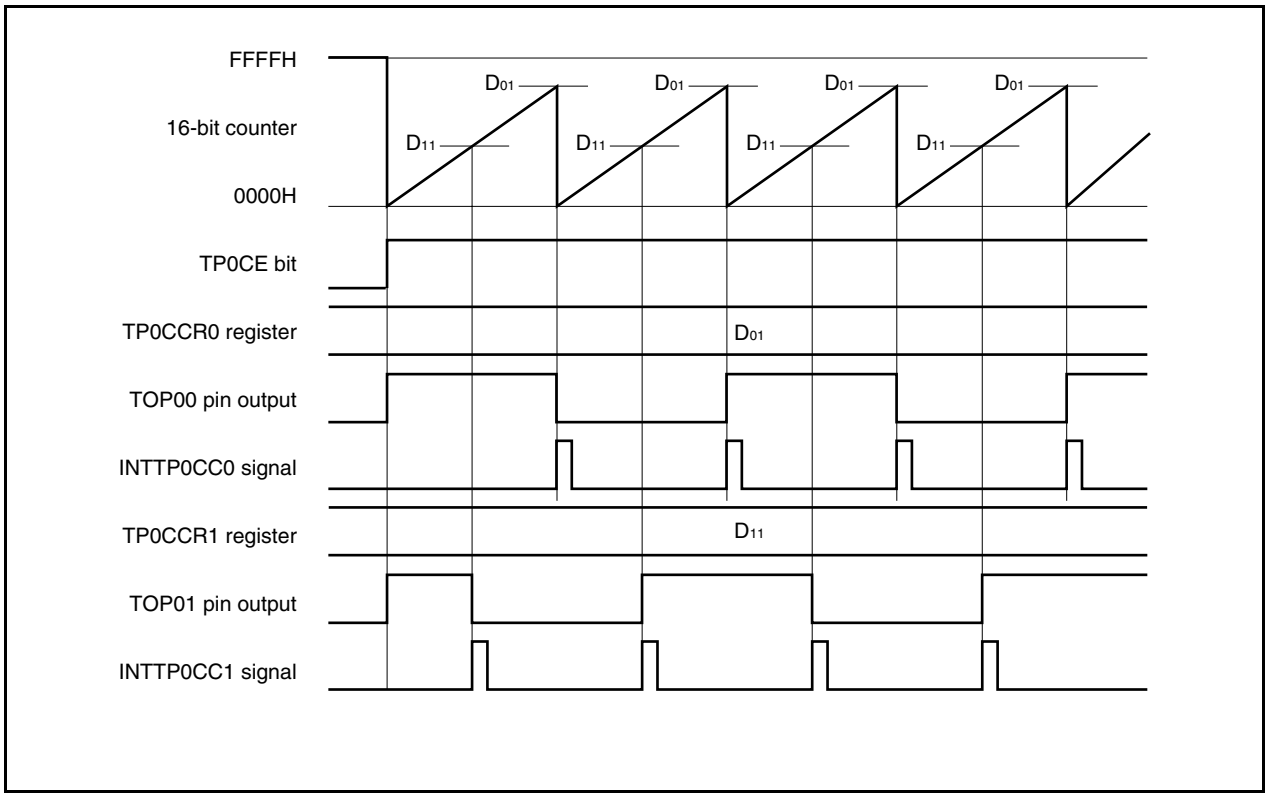
Figure 6-6. Configuration of TP0CCR1 Register



Downloaded from Arrow.com.

If the set value of the TP0CCR1 register is less than the set value of the TP0CCR0 register, the INTTP0CC1 signal is generated once per cycle. At the same time, the output of the TOP01 pin is inverted. The TOP01 pin outputs a square wave with the same cycle as that output by the TOP00 pin.

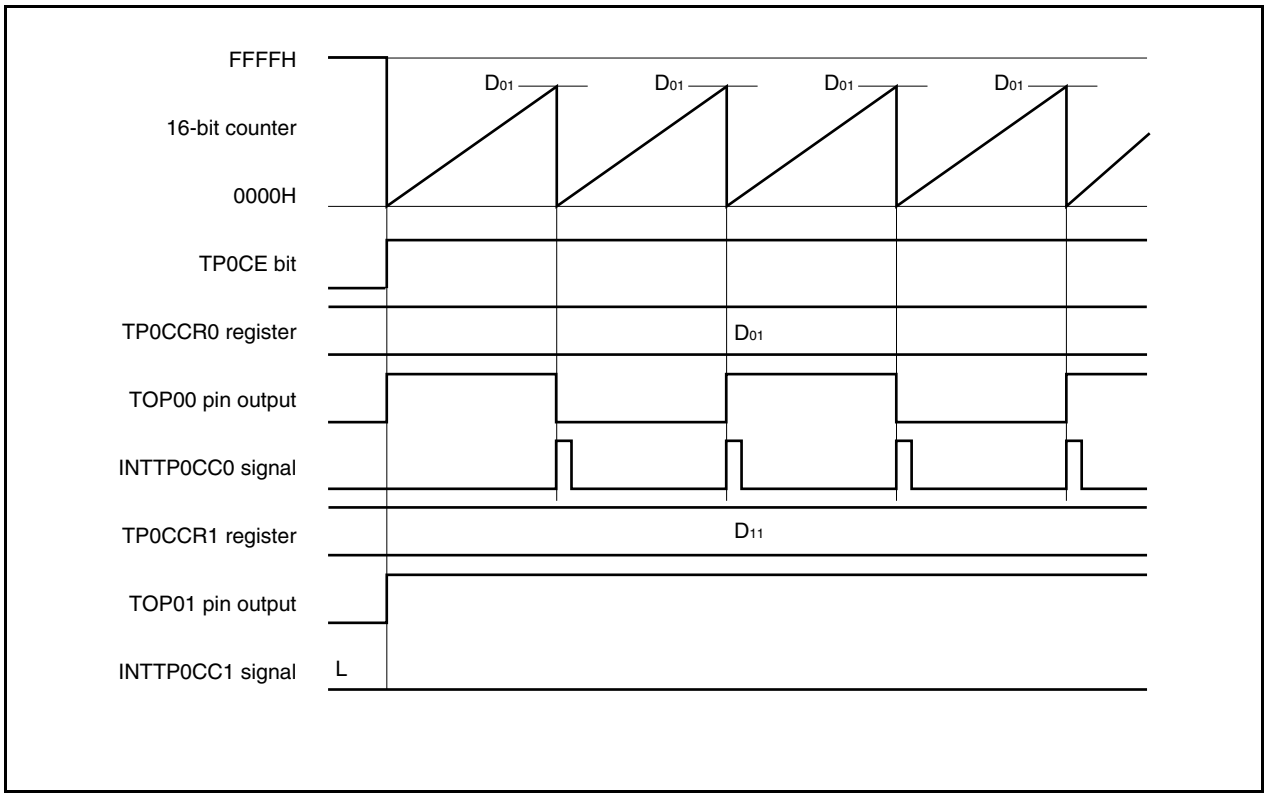
Figure 6-7. Timing Chart When $D_{01} \geq D_{11}$



Downloaded from Arrow.com.

If the set value of the TP0CCR1 register is greater than the set value of the TP0CCR0 register, the count value of the 16-bit counter does not match the value of the TP0CCR1 register. Consequently, the INTTP0CC1 signal is not generated, nor is the output of the TOP01 pin changed.

Figure 6-8. Timing Chart When $D_{01} < D_{11}$



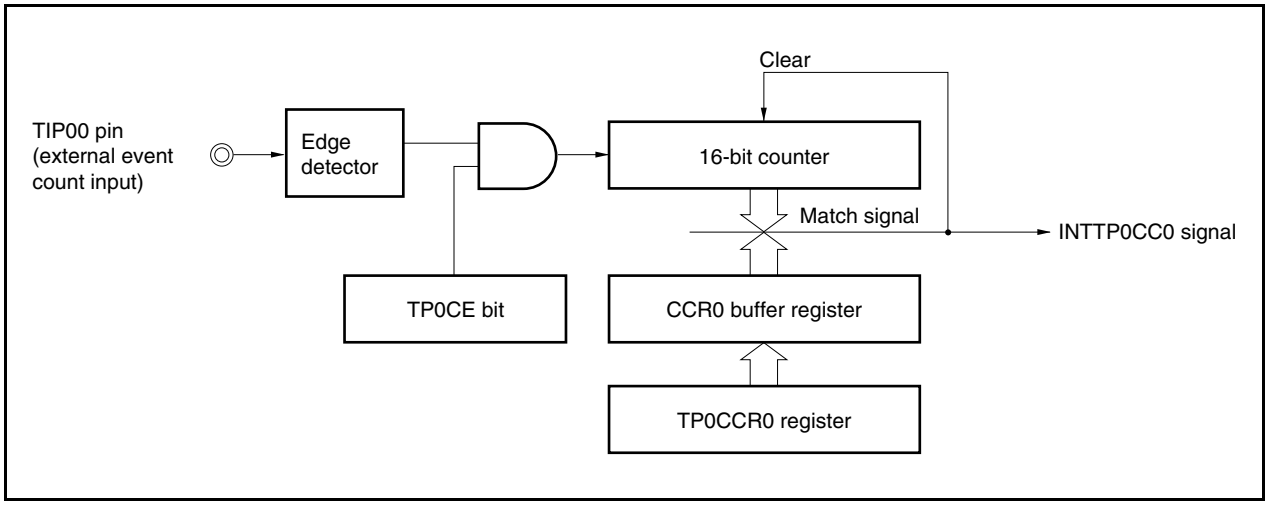
Downloaded from Arrow.com.

6.5.2 External event count mode (TP0MD2 to TP0MD0 bits = 001)

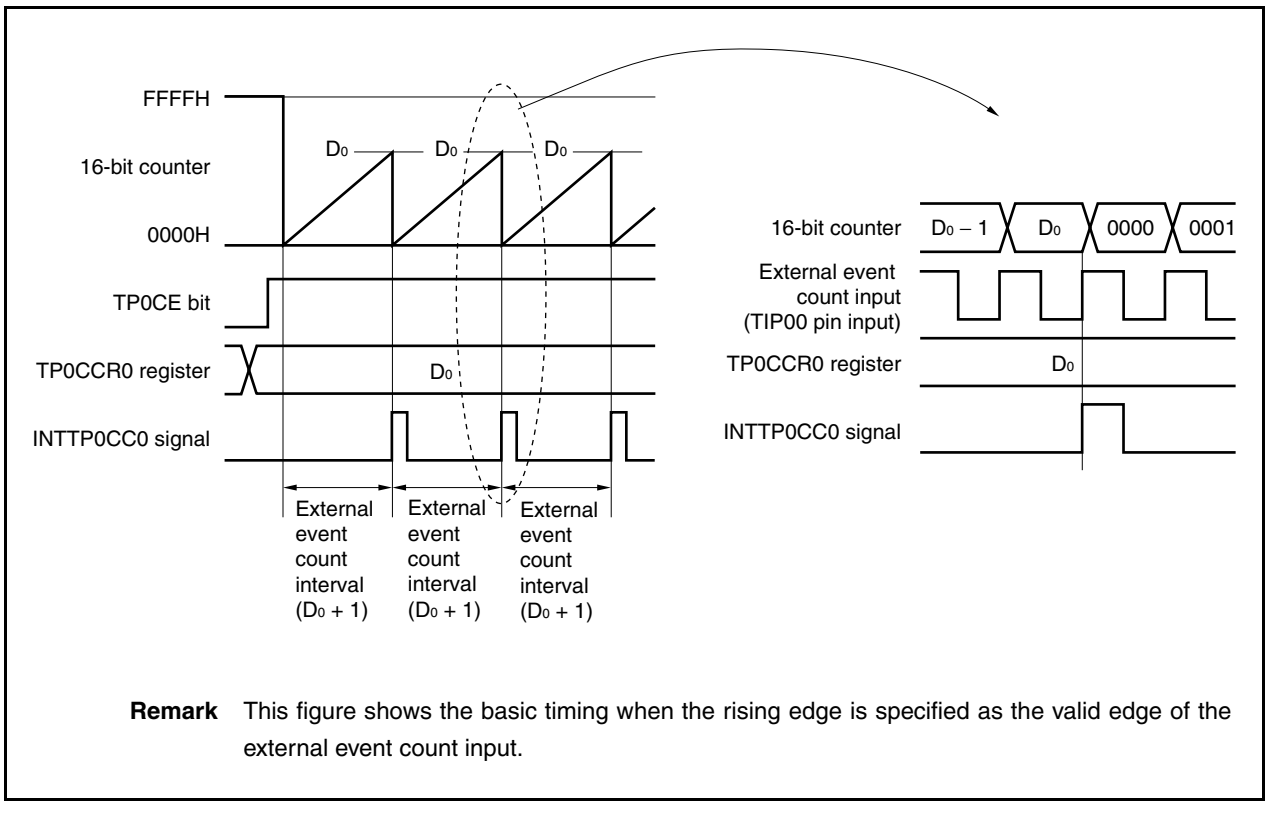
In the external event count mode, the valid edge of the external event count input is counted when the TP0CTL0.TP0CE bit is set to 1, and an interrupt request signal (INTTP0CC0) is generated each time the specified number of edges have been counted. The timer output (TOP00, TOP01 pins) cannot be used.

Usually, the TP0CCR1 register is not used in the external event count mode.

Figure 6-9. Configuration in External Event Count Mode



<R> **Figure 6-10. Basic Timing in External Event Count Mode**



Remark This figure shows the basic timing when the rising edge is specified as the valid edge of the external event count input.

Downloaded from Arrow.com.

When the TP0CE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H. The counter counts each time the valid edge of external event count input is detected. Additionally, the set value of the TP0CCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, and a compare match interrupt request signal (INTTP0CC0) is generated.

The INTTP0CC0 signal is generated each time the valid edge of the external event count input has been detected (set value of TP0CCR0 register + 1) times.

Figure 6-11. Register Setting for Operation in External Event Count Mode (1/2)

(a) TMP0 control register 0 (TP0CTL0)

	TP0CE					TP0CKS2	TP0CKS1	TP0CKS0
TP0CTL0	0/1	0	0	0	0	0	0	0

0: Stop counting
1: Enable counting

(b) TMP0 control register 1 (TP0CTL1)

		TP0EST	TP0EEE			TP0MD2	TP0MD1	TP0MD0
TP0CTL1	0	0	0	0	0	0	0	1

0, 0, 1:
External event count mode

(c) TMP0 I/O control register 0 (TP0IOC0)

					TP0OL1	TP0OE1	TP0OL0	TP0OE0
TP0IOC0	0	0	0	0	0	0	0	0

0: Disable TOP00 pin output

0: Disable TOP01 pin output

(d) TMP0 I/O control register 2 (TP0IOC2)

					TP0EES1	TP0EES0	TP0ETS1	TP0ETS0
TP0IOC2	0	0	0	0	0/1	0/1	0	0

Select valid edge
of external event
count input

Downloaded from Arrow.com.

Figure 6-11. Register Setting for Operation in External Event Count Mode (2/2)

(e) TMP0 counter read buffer register (TP0CNT)

The count value of the 16-bit counter can be read by reading the TP0CNT register.

(f) TMP0 capture/compare register 0 (TP0CCR0)

If D_0 is set to the TP0CCR0 register, the counter is cleared and a compare match interrupt request signal (INTTP0CC0) is generated when the number of external event counts reaches (D_0 + 1).

(g) TMP0 capture/compare register 1 (TP0CCR1)

Usually, the TP0CCR1 register is not used in the external event count mode. However, the set value of the TP0CCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTP0CC1) is generated.

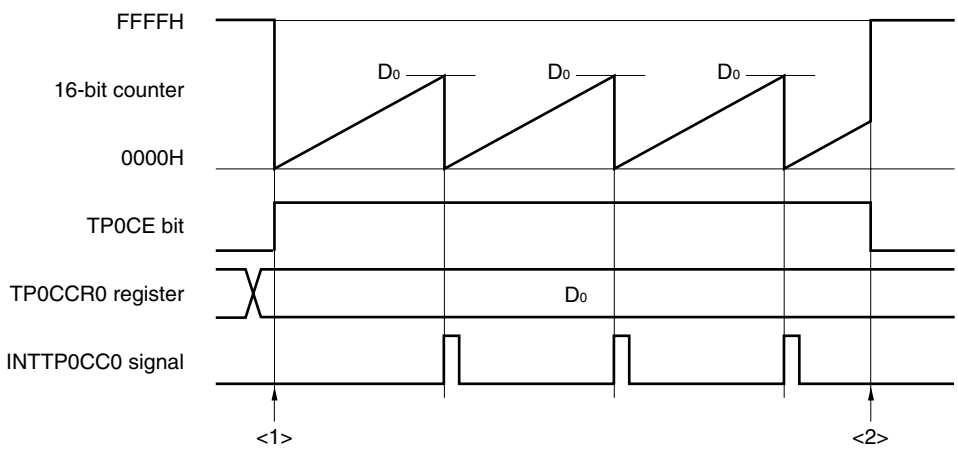
Therefore, mask the interrupt signal by using the interrupt mask flag (TP0CCMK1).

Remark TMP0 I/O control register 1 (TP0IOC1) and TMP0 option register 0 (TP0OPT0) are not used in the external event count mode.

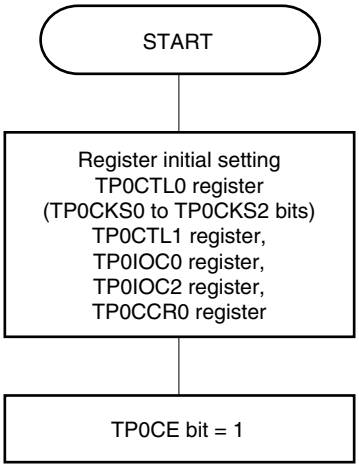
Downloaded from Arrow.com.

(1) External event count mode operation flow

Figure 6-12. Flow of Software Processing in External Event Count Mode



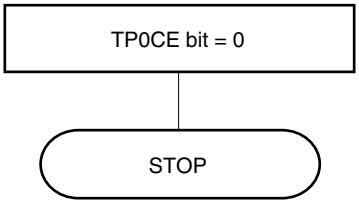
<1> Count operation start flow



Initial setting of these registers is performed before setting the TP0CE bit to 1.

The TP0CKS0 to TP0CKS2 bits can be set at the same time when counting has been started (TP0CE bit = 1).

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TP0CE bit to 0.

Downloaded from Arrow.com.

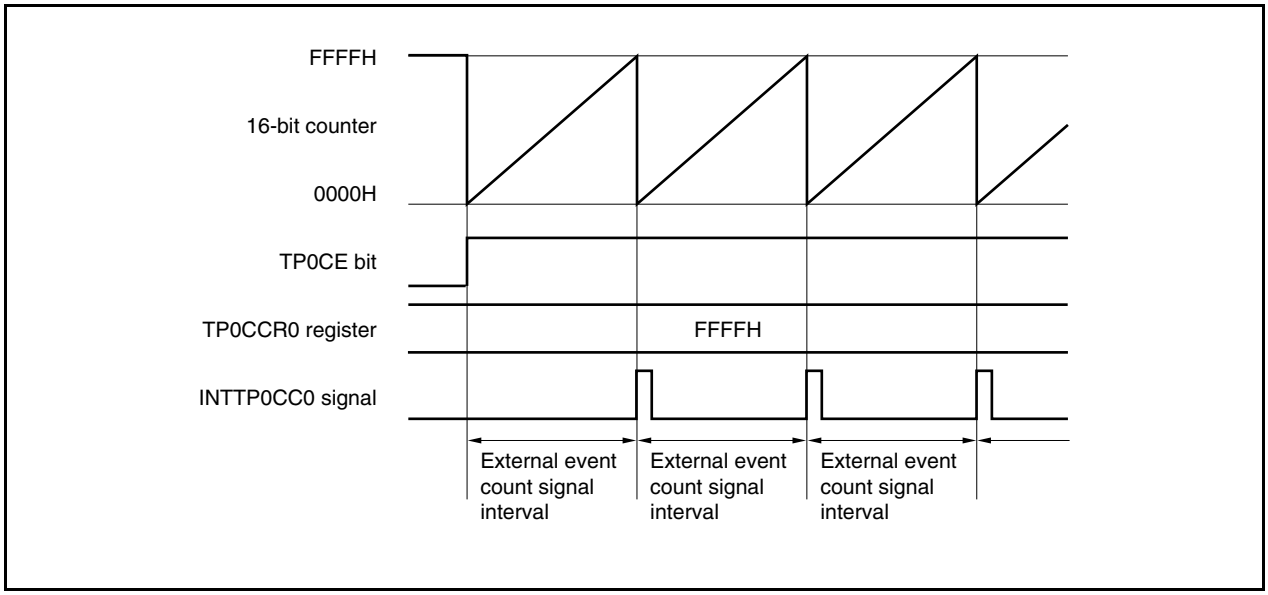
(2) Operation timing in external event count mode

<R> **Cautions 1. In the external event count mode, do not set the TP0CCR0 and TP0CCR1 registers to 0000H.**

<R> **2. In the external event count mode, use of the timer output is disabled. If performing timer output using external event count input, set the interval timer mode, and select the operation enabled by the external event count input for the count clock (TP0CTL1.TP0MD2 to TP0CTL1.TP0MD0 bits = 000, TP0CTL1.TP0EEE bit = 1).**

(a) Operation if TP0CCR0 register is set to FFFFH

If the TP0CCR0 register is set to FFFFH, the 16-bit counter counts to FFFFH each time the valid edge of the external event count signal has been detected. The 16-bit counter is cleared to 0000H in synchronization with the next count-up timing, and the INTTP0CC0 signal is generated. At this time, the TP0OPT0.TP0OVF bit is not set.

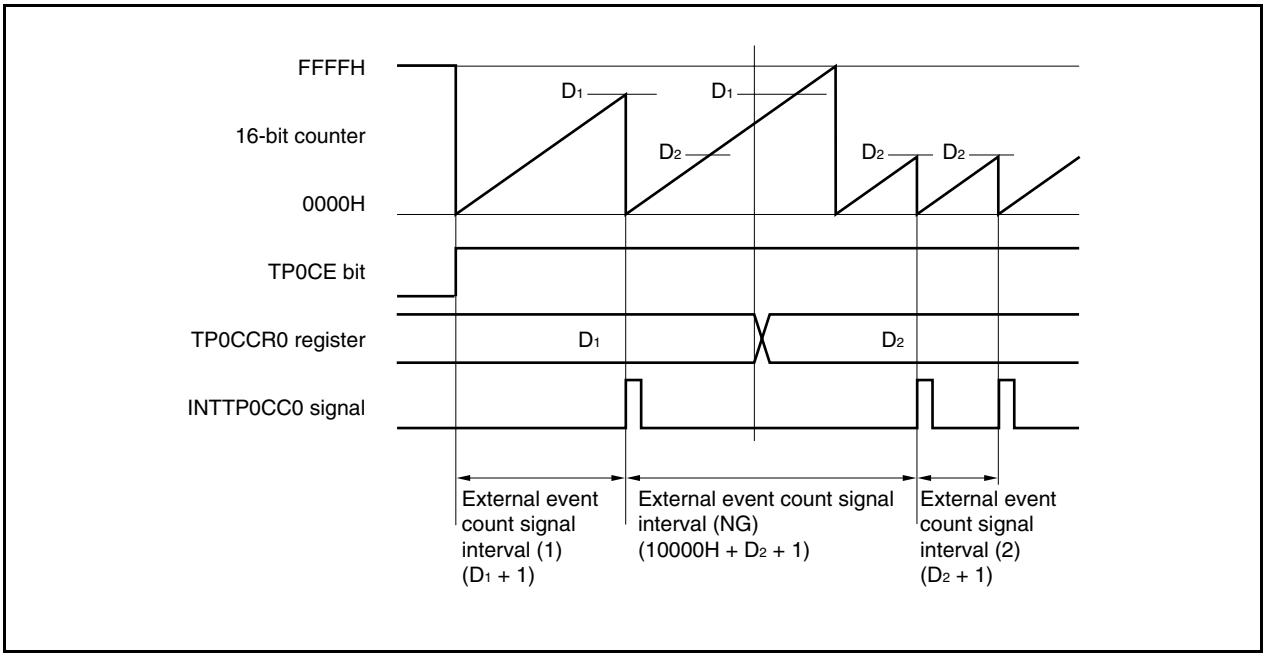


Downloaded from Arrow.com.

(b) Notes on rewriting the TP0CCR0 register

To change the value of the TP0CCR0 register to a smaller value, stop counting once and then change the set value.

If the value of the TP0CCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



If the value of the TP0CCR0 register is changed from D_1 to D_2 while the count value is greater than D_2 but less than D_1, the count value is transferred to the CCR0 buffer register as soon as the TP0CCR0 register has been rewritten. Consequently, the value that is compared with the 16-bit counter is D_2.

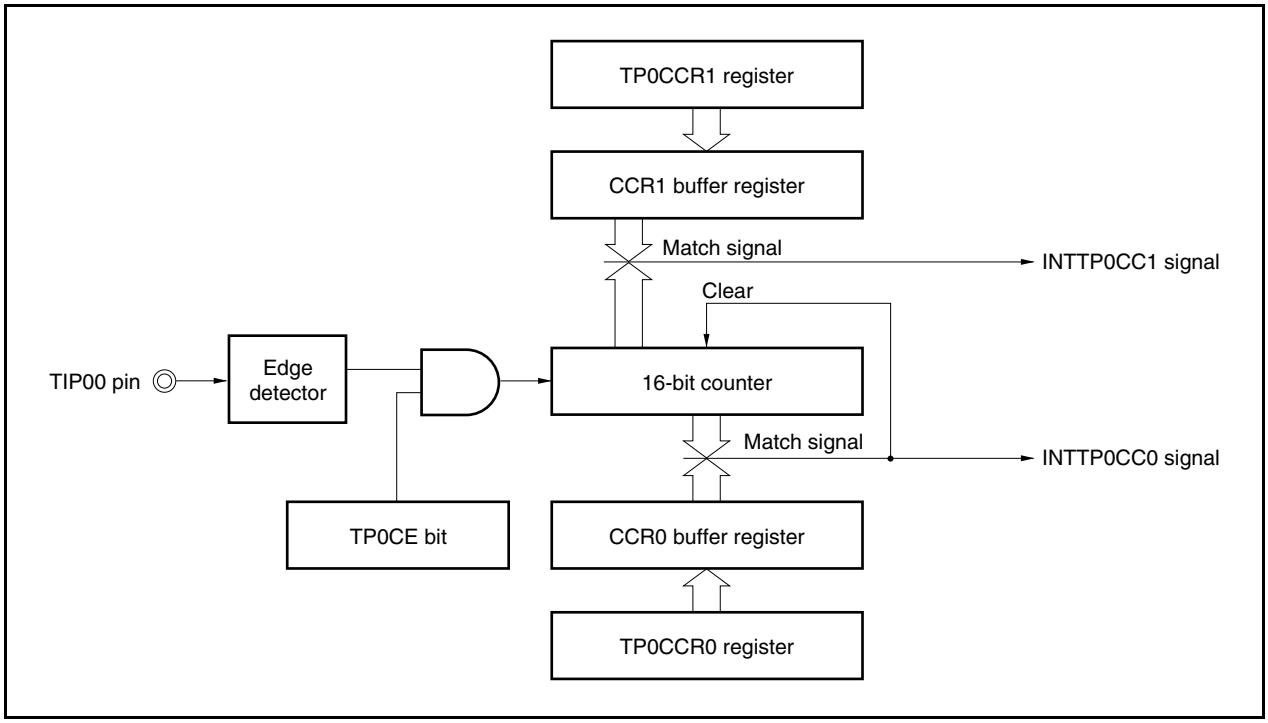
Because the count value has already exceeded D_2, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches D_2, the INTTP0CC0 signal is generated.

Therefore, the INTTP0CC0 signal may not be generated at the valid edge count of "$(D_1 + 1)$ times" or "$(D_2 + 1)$ times" originally expected, but may be generated at the valid edge count of "$(10000H + D_2 + 1)$ times".

Downloaded from Arrow.com.

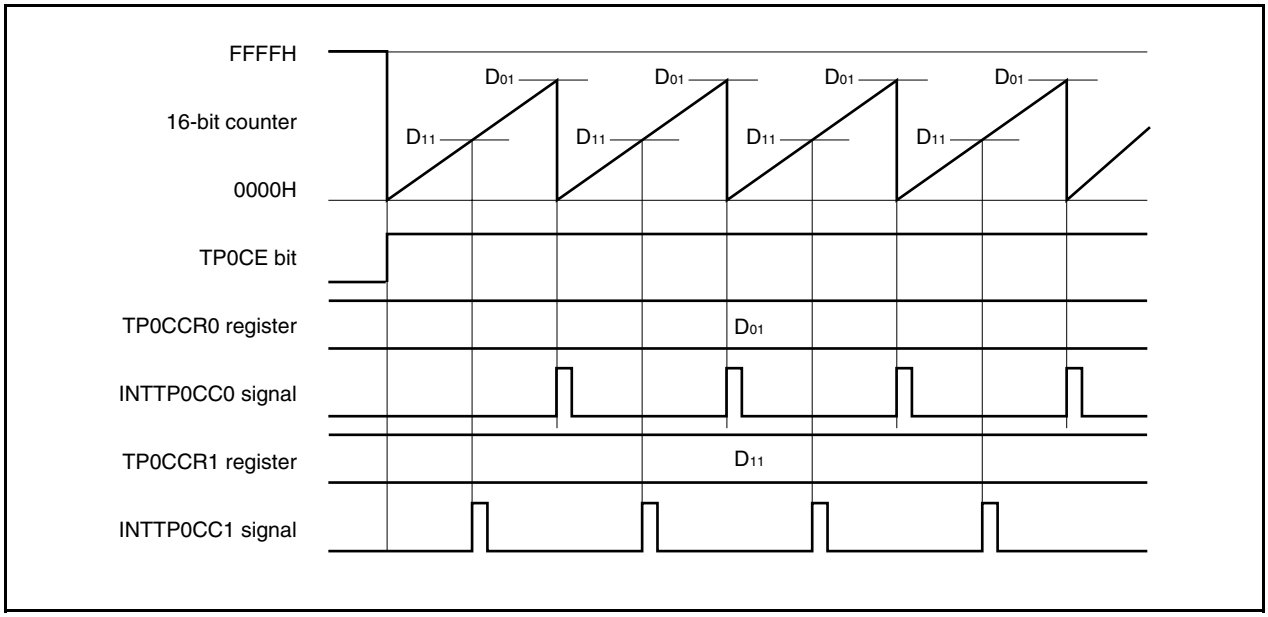
(c) Operation of TP0CCR1 register

Figure 6-13. Configuration of TP0CCR1 Register



If the set value of the TP0CCR1 register is smaller than the set value of the TP0CCR0 register, the INTTP0CC1 signal is generated once per cycle.

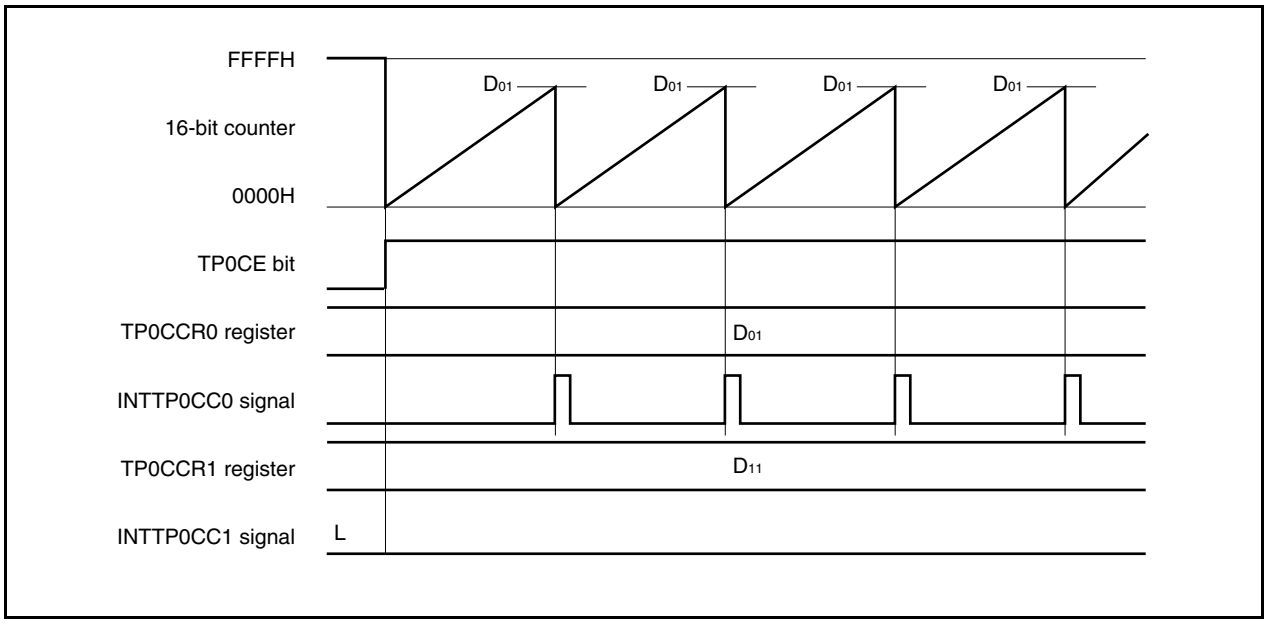
Figure 6-14. Timing Chart When $D_{01} \geq D_{11}$



Downloaded from Arrow.com.

If the set value of the TP0CCR1 register is greater than the set value of the TP0CCR0 register, the INTTP0CC1 signal is not generated because the count value of the 16-bit counter and the value of the TP0CCR1 register do not match.

Figure 6-15. Timing Chart When $D_{01} < D_{11}$



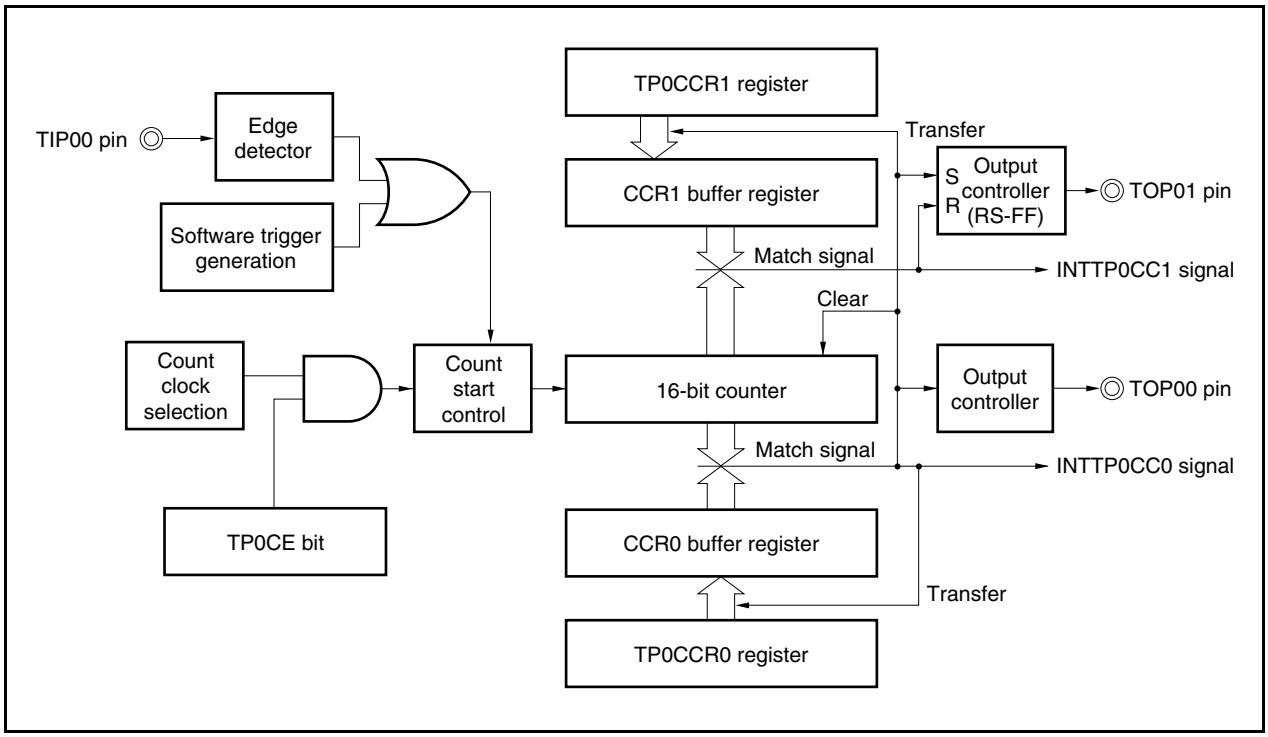
Downloaded from Arrow.com.

6.5.3 External trigger pulse output mode (TP0MD2 to TP0MD0 bits = 010)

In the external trigger pulse output mode, 16-bit timer/event counter P waits for a trigger when the TP0CTL0.TP0CE bit is set to 1. When the valid edge of an external trigger input signal is detected, 16-bit timer/event counter P starts counting, and outputs a PWM waveform from the TOP01 pin.

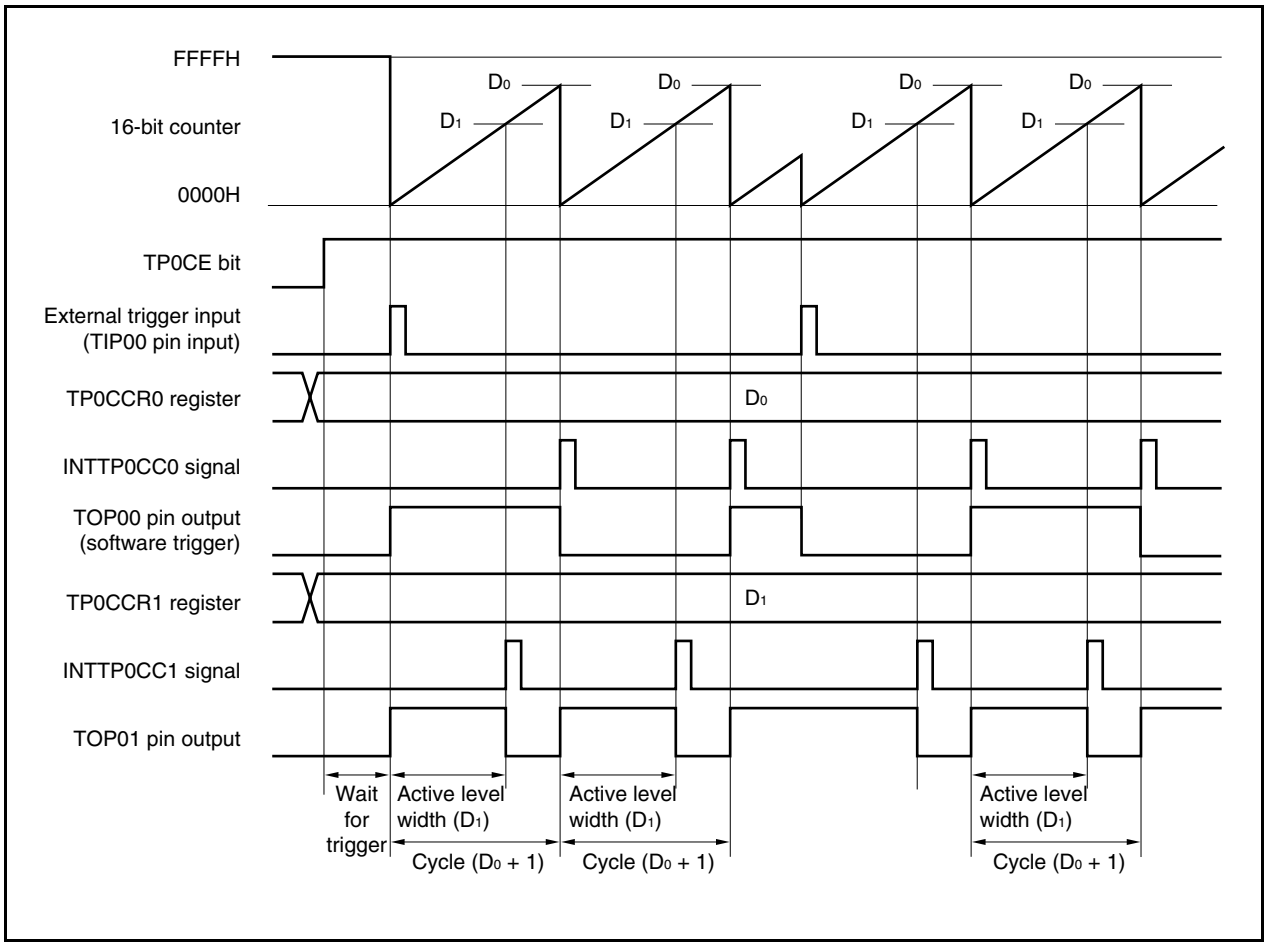
Pulses can also be output by generating a software trigger instead of using the external trigger. When using a software trigger, a square wave that has one cycle of the PWM waveform as half its cycle can also be output from the TOP00 pin.

Figure 6-16. Configuration in External Trigger Pulse Output Mode



Downloaded from Arrow.com.

Figure 6-17. Basic Timing in External Trigger Pulse Output Mode



<R> 16-bit timer/event counter P waits for a trigger when the TP0CE bit is set to 1. When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting at the same time, and outputs a PWM waveform from the TOP01 pin. If the trigger is generated again while the counter is operating, the counter is cleared to 0000H and restarted. (The output of the TOP00 pin is inverted. The TOP01 pin outputs a high level regardless of the status (high/low) when a trigger occurs.)

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TP0CCR1 register) × Count clock cycle
Cycle = (Set value of TP0CCR0 register + 1) × Count clock cycle
Duty factor = (Set value of TP0CCR1 register)/(Set value of TP0CCR0 register + 1)

The compare match interrupt request signal INTTP0CC0 is generated when the 16-bit counter counts next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal INTTP0CC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TP0CCRa register is transferred to the CCRa buffer register when the count value of the 16-bit counter matches the value of the CCRa buffer register and the 16-bit counter is cleared to 0000H.

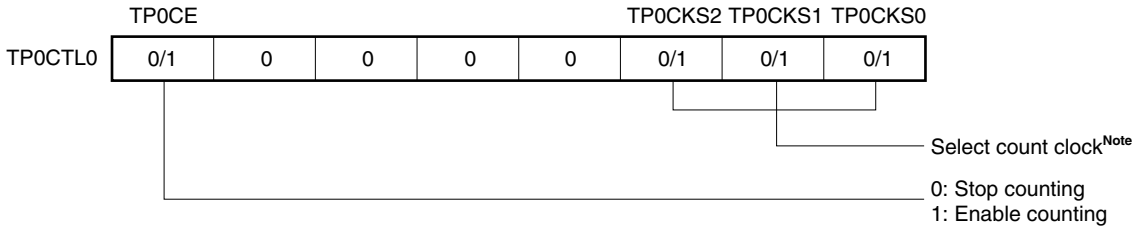
The valid edge of an external trigger input signal, or setting the software trigger (TP0CTL1.TP0EST bit) to 1 is used as the trigger.

Remark a = 0, 1

Downloaded from Arrow.com.

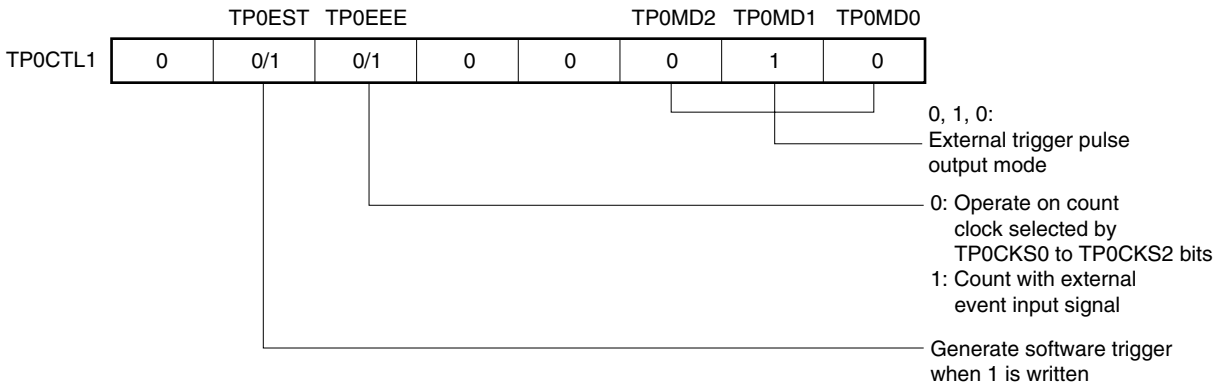
Figure 6-18. Setting of Registers in External Trigger Pulse Output Mode (1/2)

(a) TMP0 control register 0 (TP0CTL0)

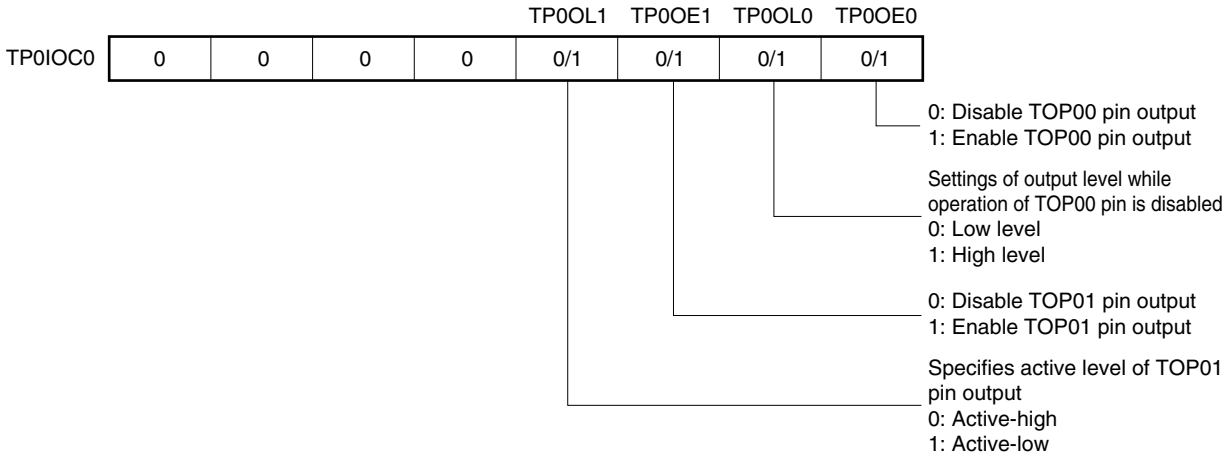


Note The setting is invalid when the TP0CTL1.TP0EEE bit = 1.

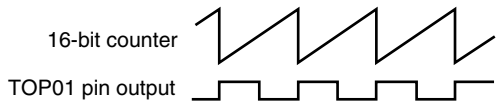
(b) TMP0 control register 1 (TP0CTL1)



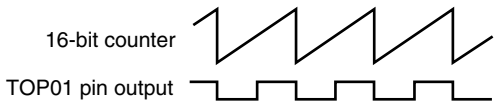
(c) TMP0 I/O control register 0 (TP0IOC0)



- When TP0OL1 bit = 0



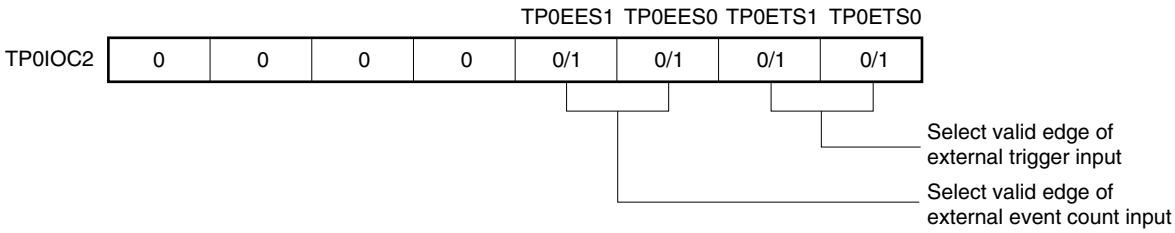
- When TP0OL1 bit = 1



Downloaded from Arrow.com.

Figure 6-18. Setting of Registers in External Trigger Pulse Output Mode (2/2)

(d) TMP0 I/O control register 2 (TP0IOC2)



(e) TMP0 counter read buffer register (TP0CNT)

The value of the 16-bit counter can be read by reading the TP0CNT register.

(f) TMP0 capture/compare registers 0 and 1 (TP0CCR0 and TP0CCR1)

If D_0 is set to the TP0CCR0 register and D_1 to the TP0CCR1 register, the cycle and active level of the PWM waveform are as follows.

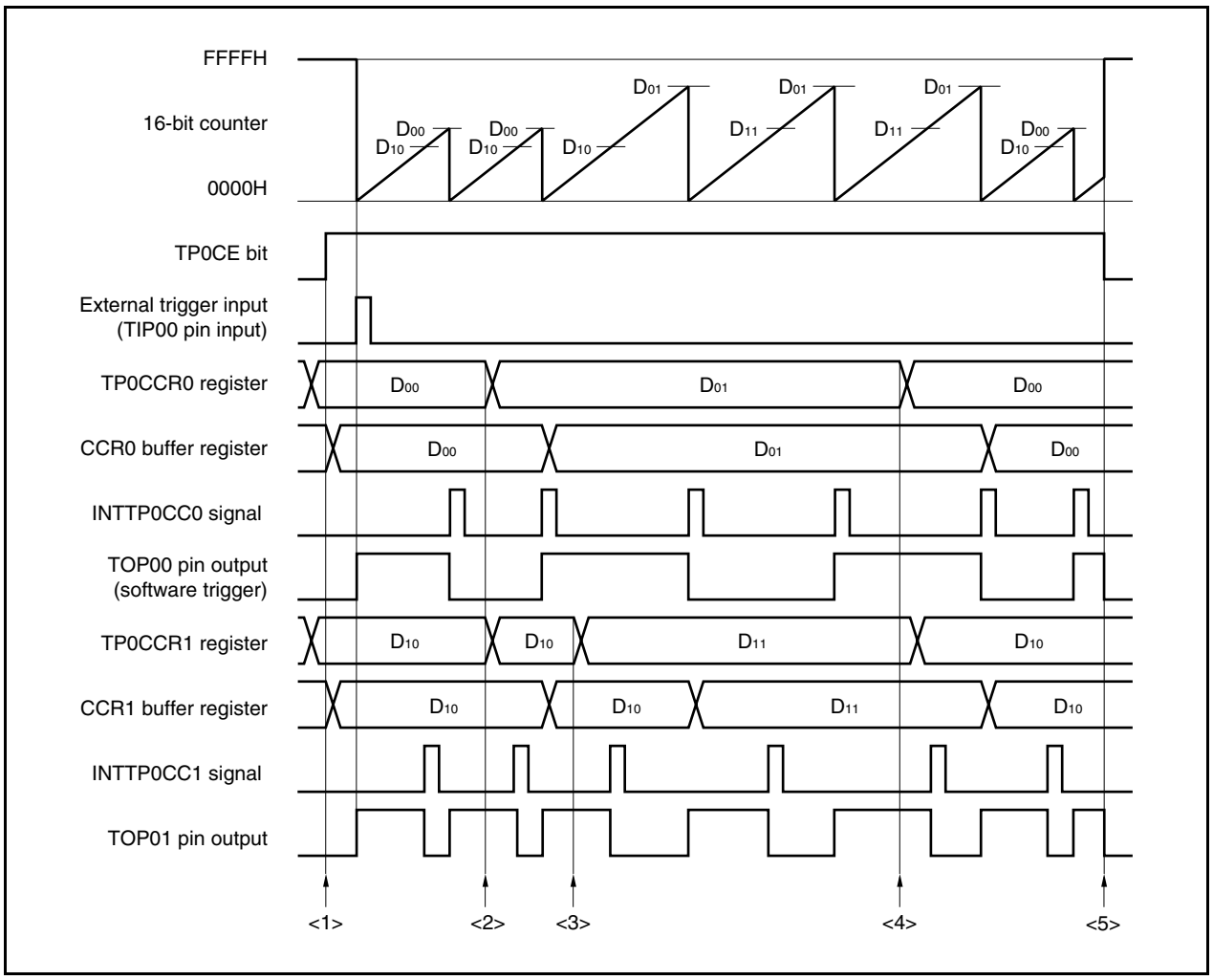
Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

Remark TMP0 I/O control register 1 (TP0IOC1) and TMP0 option register 0 (TP0OPT0) are not used in the external trigger pulse output mode.

Downloaded from Arrow.com.

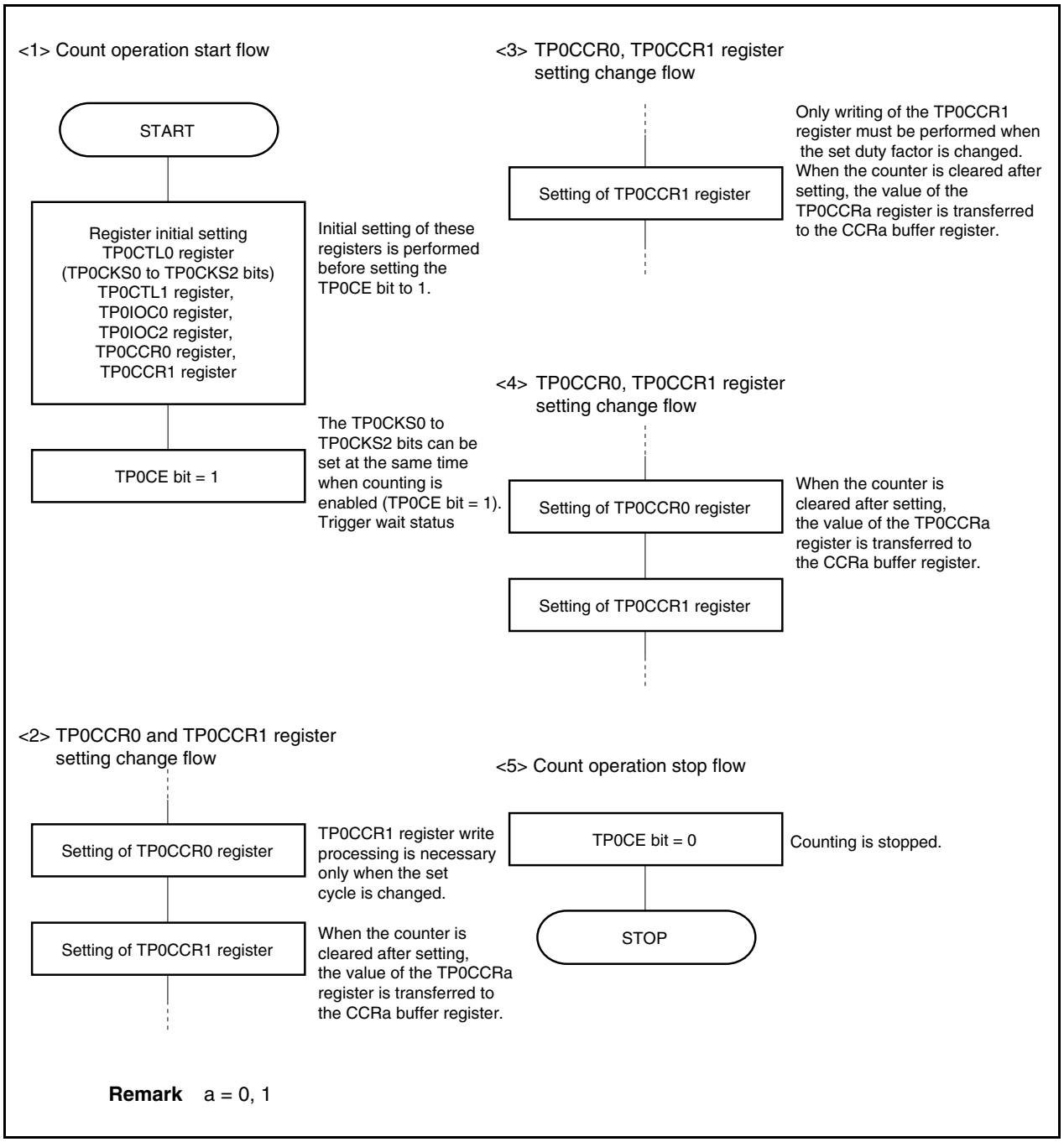
(1) Operation flow in external trigger pulse output mode

Figure 6-19. Software Processing Flow in External Trigger Pulse Output Mode (1/2)



Downloaded from Arrow.com.

Figure 6-19. Software Processing Flow in External Trigger Pulse Output Mode (2/2)



Remark a = 0, 1

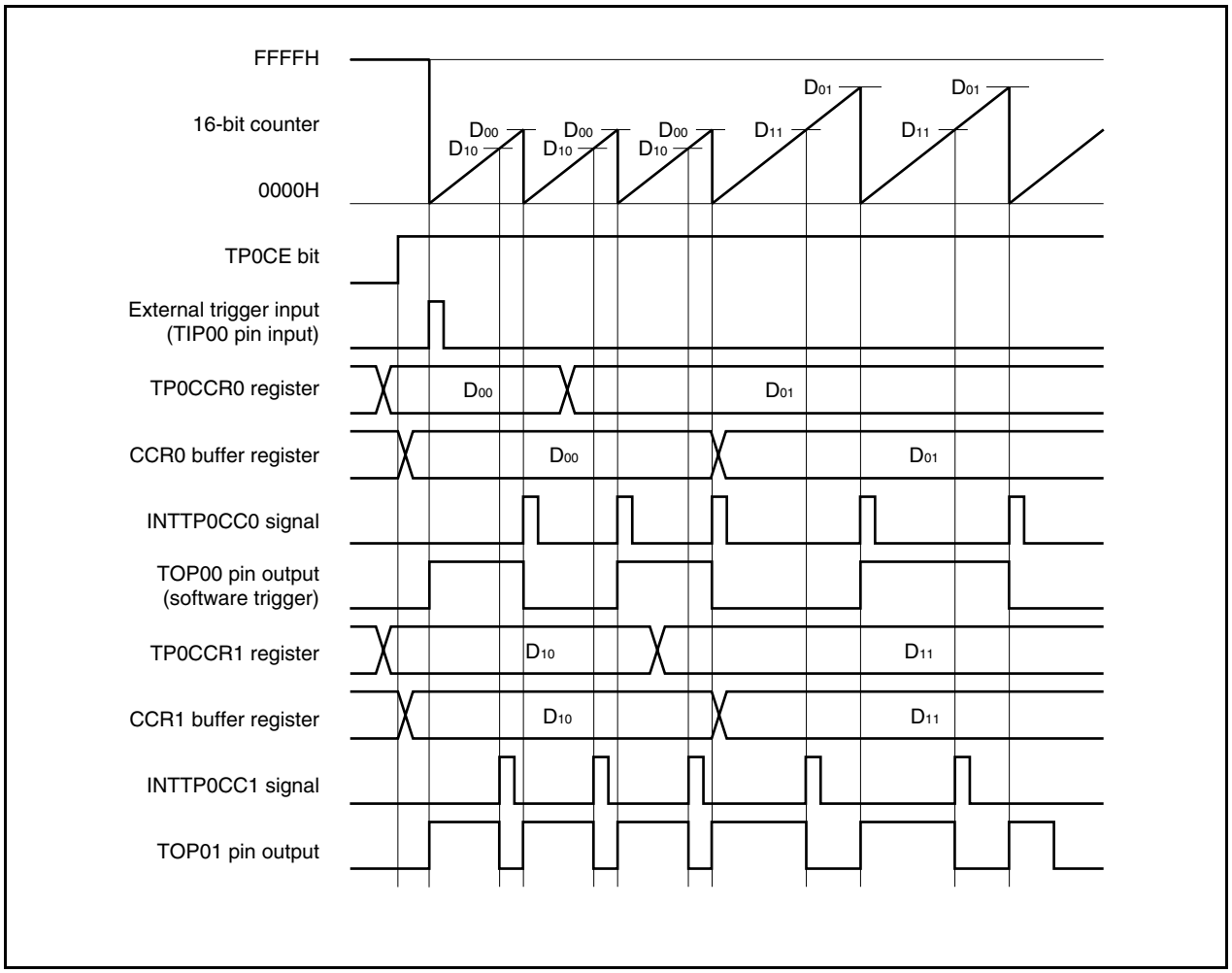
Downloaded from Arrow.com.

(2) External trigger pulse output mode operation timing

(a) Note on changing pulse width during operation

To change the PWM waveform while the counter is operating, write the TP0CCR1 register last.

Rewrite the TP0CCRa register after writing the TP0CCR1 register after the INTTP0CC0 signal is detected.



Downloaded from Arrow.com.

In order to transfer data from the TP0CCRa register to the CCRa buffer register, the TP0CCR1 register must be written.

To change both the cycle and active level width of the PWM waveform at this time, first set the cycle to the TP0CCR0 register and then set the active level width to the TP0CCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TP0CCR0 register, and then write the same value to the TP0CCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TP0CCR1 register has to be set.

After data is written to the TP0CCR1 register, the value written to the TP0CCRa register is transferred to the CCRa buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

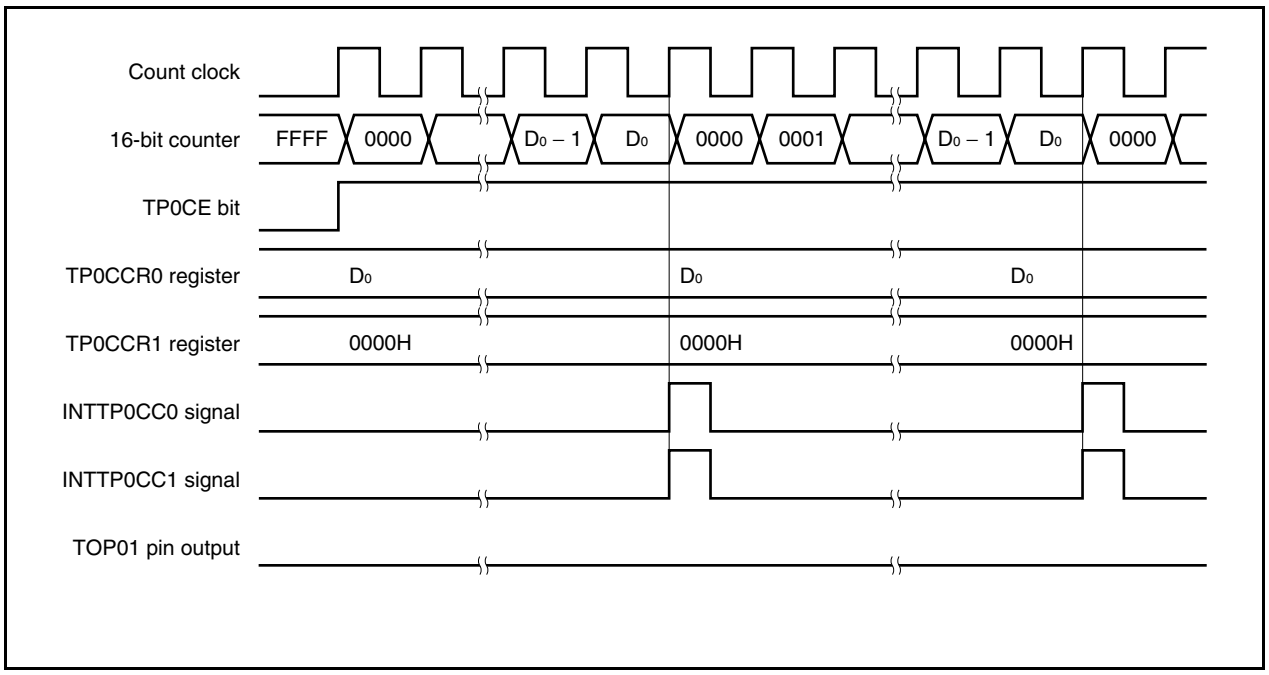
To write the TP0CCR0 or TP0CCR1 register again after writing the TP0CCR1 register once, do so after the INTTP0CC0 signal is generated. Otherwise, the value of the CCRa buffer register may become undefined because the timing of transferring data from the TP0CCRa register to the CCRa buffer register conflicts with writing the TP0CCRa register.

Remark a = 0, 1

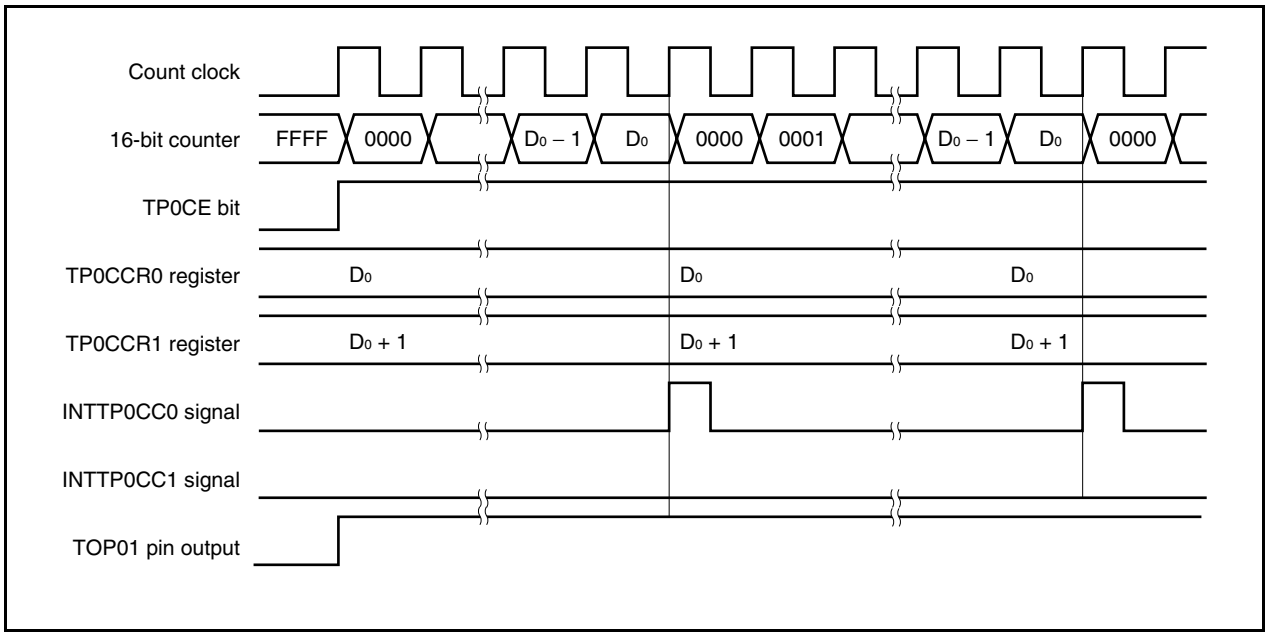
Downloaded from Arrow.com.

(b) 0%/100% output of PWM waveform

To output a 0% waveform, clear the TP0CCR1 register to 0000H. If the set value of the TP0CCR0 register is FFFFH, the INTTP0CC1 signal is generated periodically.



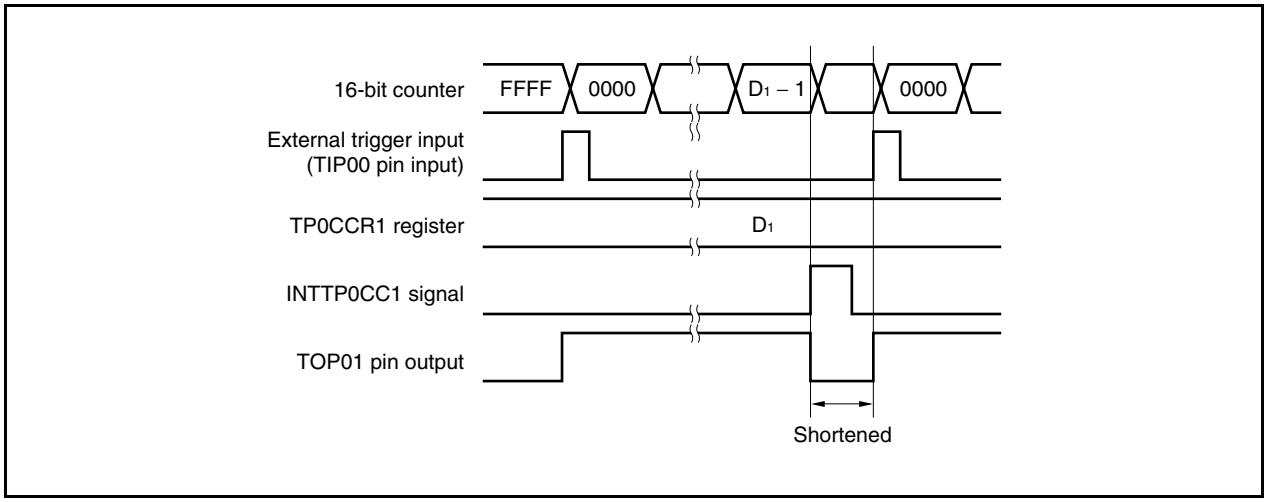
To output a 100% waveform, set a value of (set value of TP0CCR0 register + 1) to the TP0CCR1 register. If the set value of the TP0CCR0 register is FFFFH, 100% output cannot be produced.



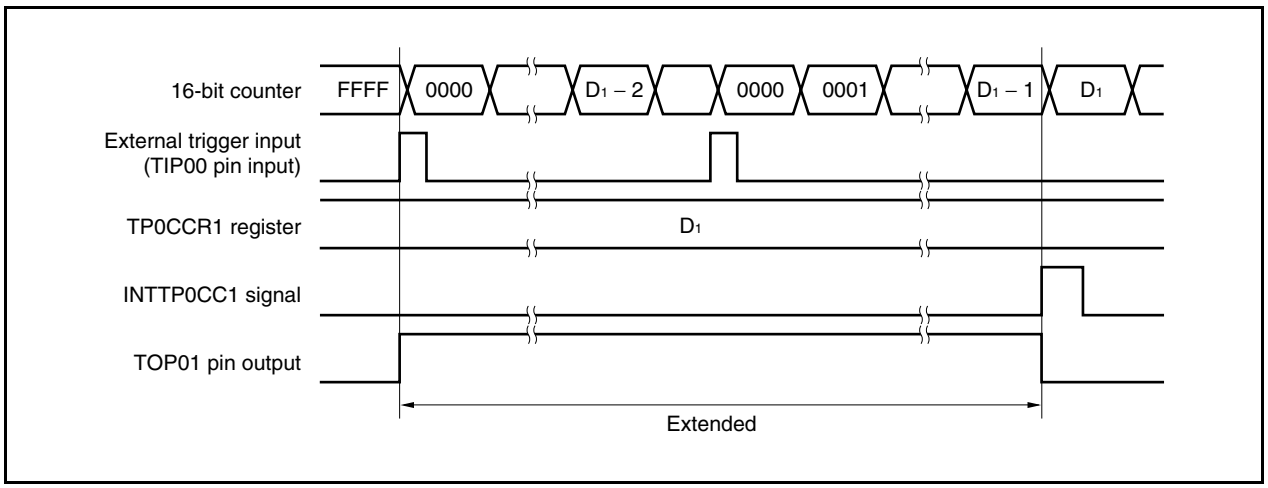
Downloaded from Arrow.com.

(c) Conflict between trigger detection and match with TP0CCR1 register

If the trigger is detected immediately after the INTTP0CC1 signal is generated, the 16-bit counter is immediately cleared to 0000H, the output signal of the TOP01 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.



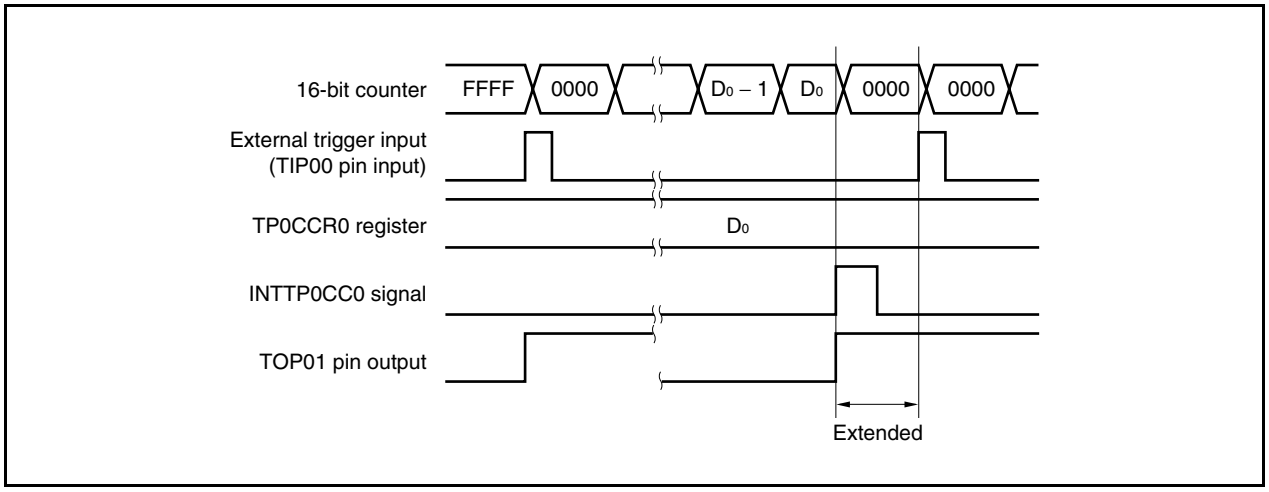
If the trigger is detected immediately before the INTTP0CC1 signal is generated, the INTTP0CC1 signal is not generated, and the 16-bit counter is cleared to 0000H and continues counting. The output signal of the TOP01 pin remains active. Consequently, the active period of the PWM waveform is extended.



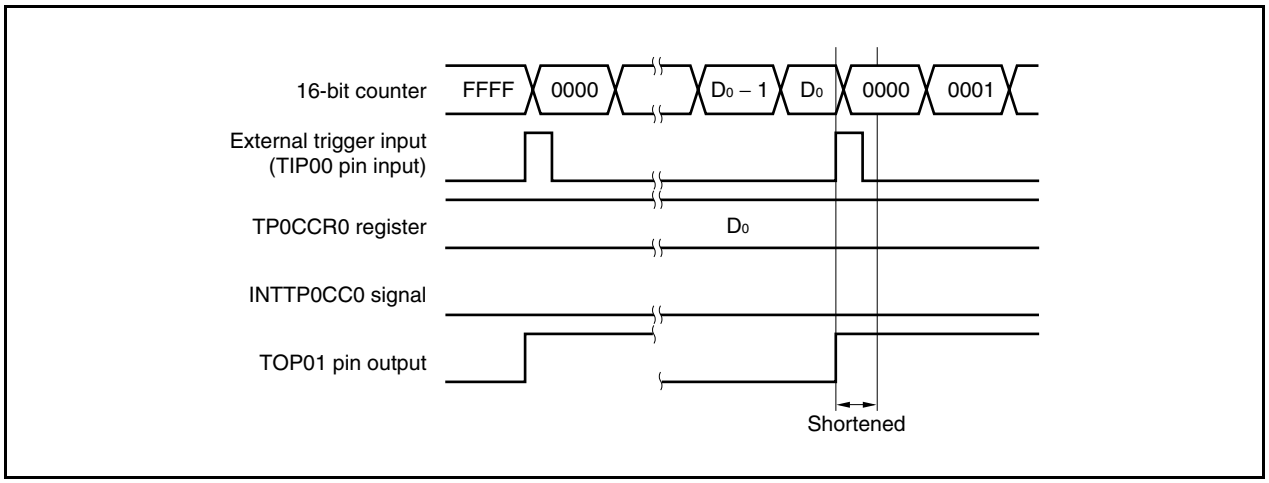
Downloaded from Arrow.com.

(d) Conflict between trigger detection and match with TP0CCR0 register

If the trigger is detected immediately after the INTTP0CC0 signal is generated, the 16-bit counter is cleared to 0000H and continues counting up. Therefore, the active period of the TOP01 pin is extended by time from generation of the INTTP0CC0 signal to trigger detection.



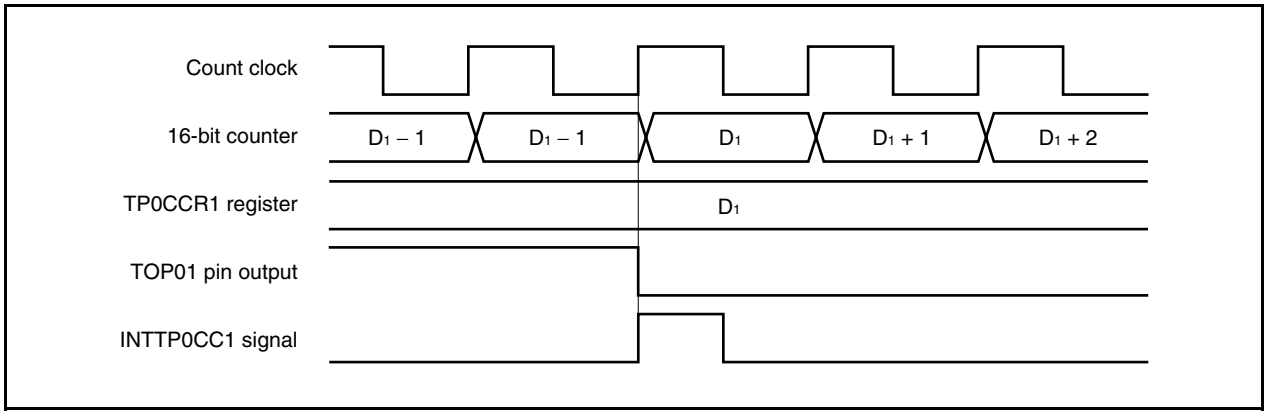
If the trigger is detected immediately before the INTTP0CC0 signal is generated, the INTTP0CC0 signal is not generated. The 16-bit counter is cleared to 0000H, the TOP01 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.



Downloaded from Arrow.com.

(e) Generation timing of compare match interrupt request signal (INTTP0CC1)

The timing of generation of the INTTP0CC1 signal in the external trigger pulse output mode differs from the timing of other INTTP0CC1 signals; the INTTP0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TP0CCR1 register.



Usually, the INTTP0CC1 signal is generated in synchronization with the next count up, after the count value of the 16-bit counter matches the value of the TP0CCR1 register.

In the external trigger pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the timing of changing the output signal of the TOP01 pin.

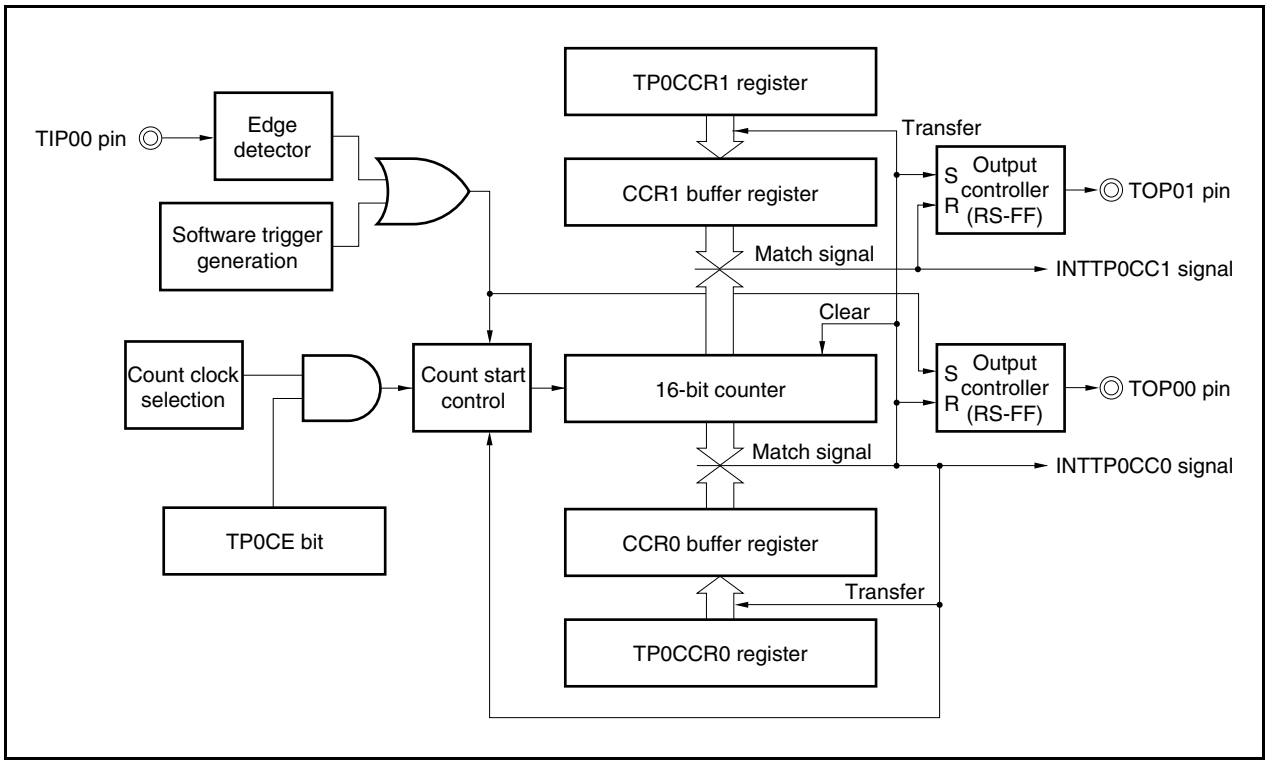
Downloaded from Arrow.com.

6.5.4 One-shot pulse output mode (TP0MD2 to TP0MD0 bits = 011)

In the one-shot pulse output mode, 16-bit timer/event counter P waits for a trigger when the TP0CTL0.TP0CE bit is set to 1. When the valid edge of an external trigger input is detected, 16-bit timer/event counter P starts counting, and outputs a one-shot pulse from the TOP01 pin.

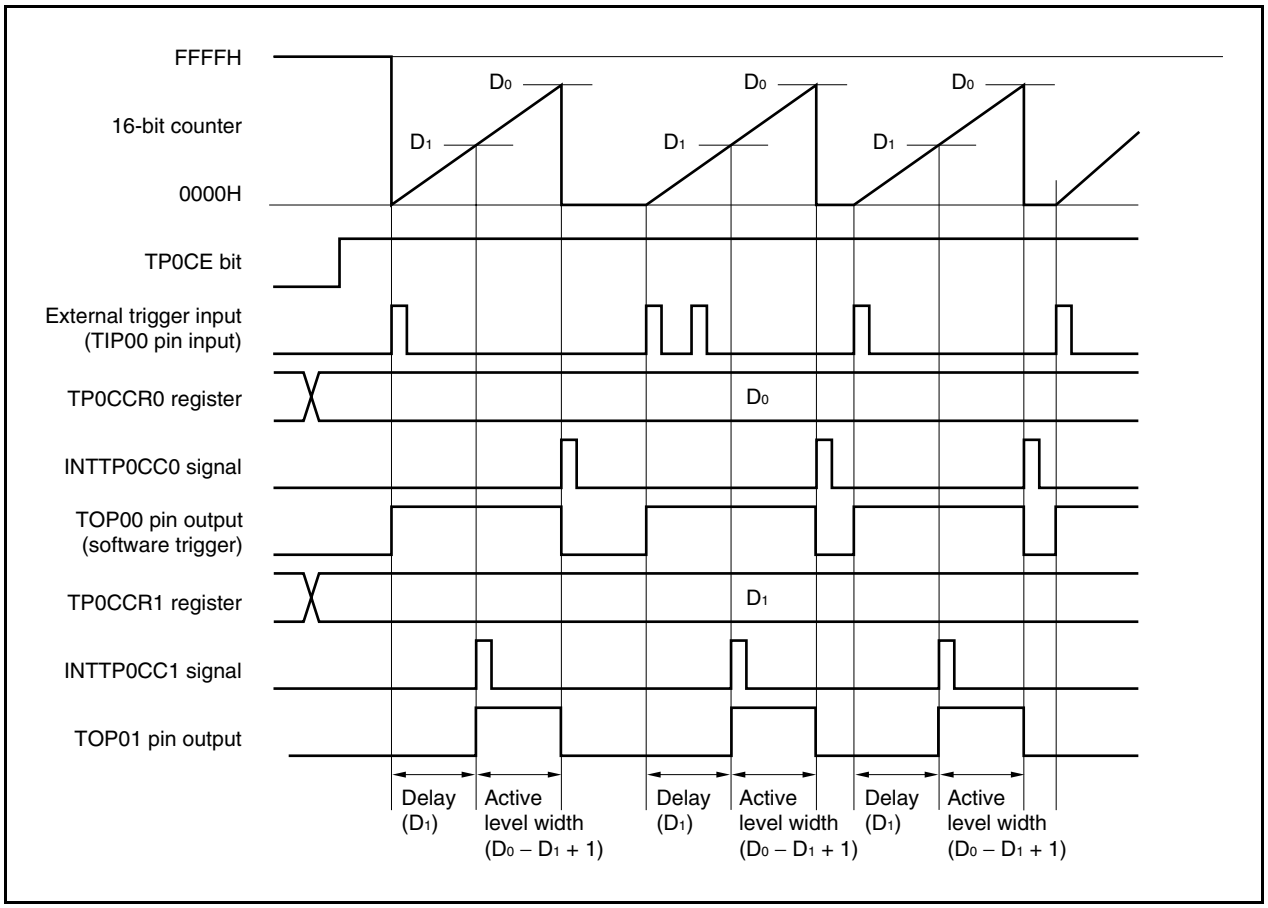
Instead of the external trigger, a software trigger can also be generated to output the pulse. When the software trigger is used, the TOP00 pin outputs the active level while the 16-bit counter is counting, and the inactive level when the counter is stopped (waiting for a trigger).

Figure 6-20. Configuration in One-Shot Pulse Output Mode



Downloaded from Arrow.com.

Figure 6-21. Basic Timing in One-Shot Pulse Output Mode



When the TP0CE bit is set to 1, 16-bit timer/event counter P waits for a trigger. When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a one-shot pulse from the TOP01 pin. After the one-shot pulse is output, the 16-bit counter is set to FFFFH, stops counting, and waits for a trigger. If a trigger is generated again while the one-shot pulse is being output, it is ignored.

The output delay period and active level width of the one-shot pulse can be calculated as follows.

Output delay period = (Set value of TP0CCR1 register) × Count clock cycle

Active level width = (Set value of TP0CCR0 register − Set value of TP0CCR1 register + 1) × Count clock cycle

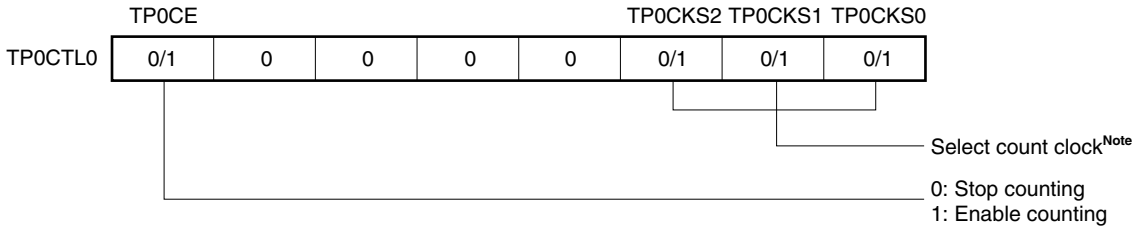
The compare match interrupt request signal INTTP0CC0 is generated when the 16-bit counter counts after its count value matches the value of the CCR0 buffer register. The compare match interrupt request signal INTTP0CC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The valid edge of an external trigger input or setting the software trigger (TP0CTL1.TP0EST bit) to 1 is used as the trigger.

Downloaded from Arrow.com.

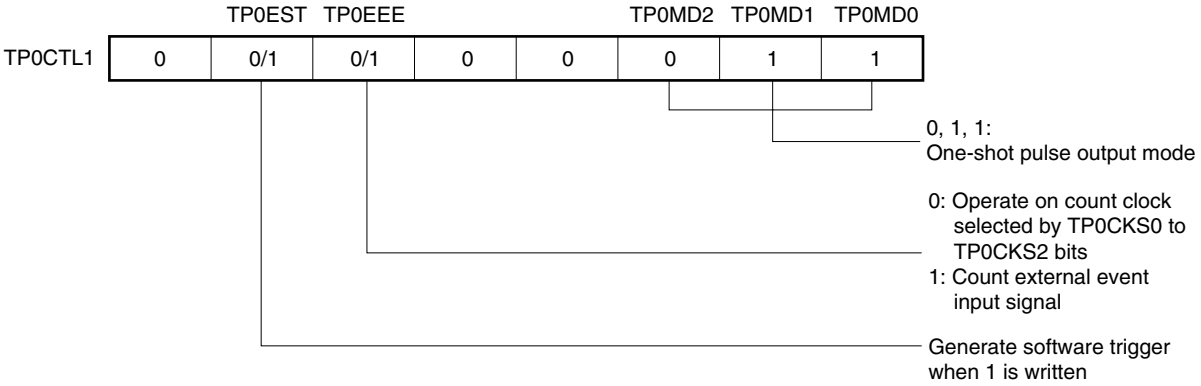
Figure 6-22. Setting of Registers in One-Shot Pulse Output Mode (1/2)

(a) TMP0 control register 0 (TP0CTL0)

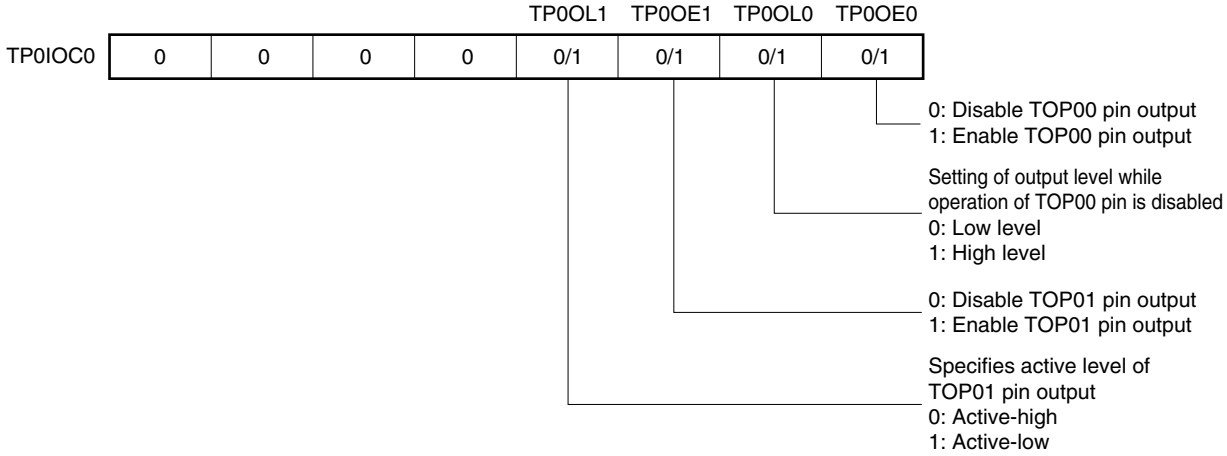


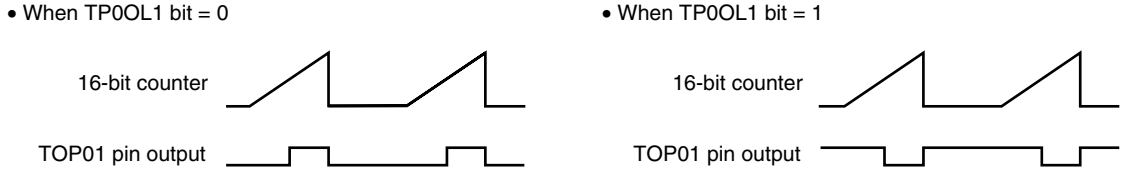
Note The setting is invalid when the TP0CTL1.TP0EEE bit = 1.

(b) TMP0 control register 1 (TP0CTL1)



(c) TMP0 I/O control register 0 (TP0IOC0)

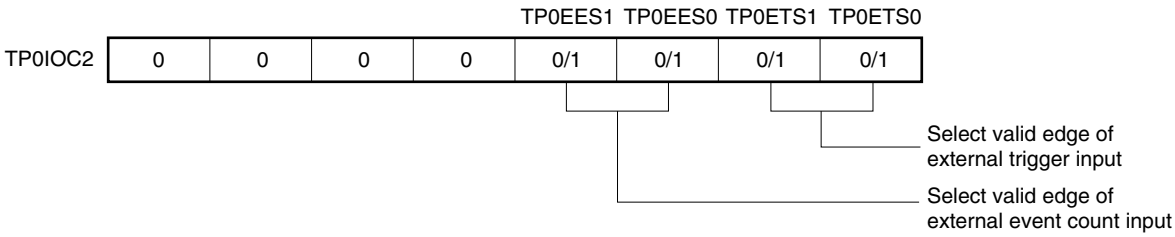




Downloaded from Arrow.com.

Figure 6-22. Setting of Registers in One-Shot Pulse Output Mode (2/2)

(d) TMP0 I/O control register 2 (TP0IOC2)



(e) TMP0 counter read buffer register (TP0CNT)

The value of the 16-bit counter can be read by reading the TP0CNT register.

(f) TMP0 capture/compare registers 0 and 1 (TP0CCR0 and TP0CCR1)

If D_0 is set to the TP0CCR0 register and D_1 to the TP0CCR1 register, the active level width and output delay period of the one-shot pulse are as follows.

Active level width = $(D_1 - D_0 + 1) \times$ Count clock cycle

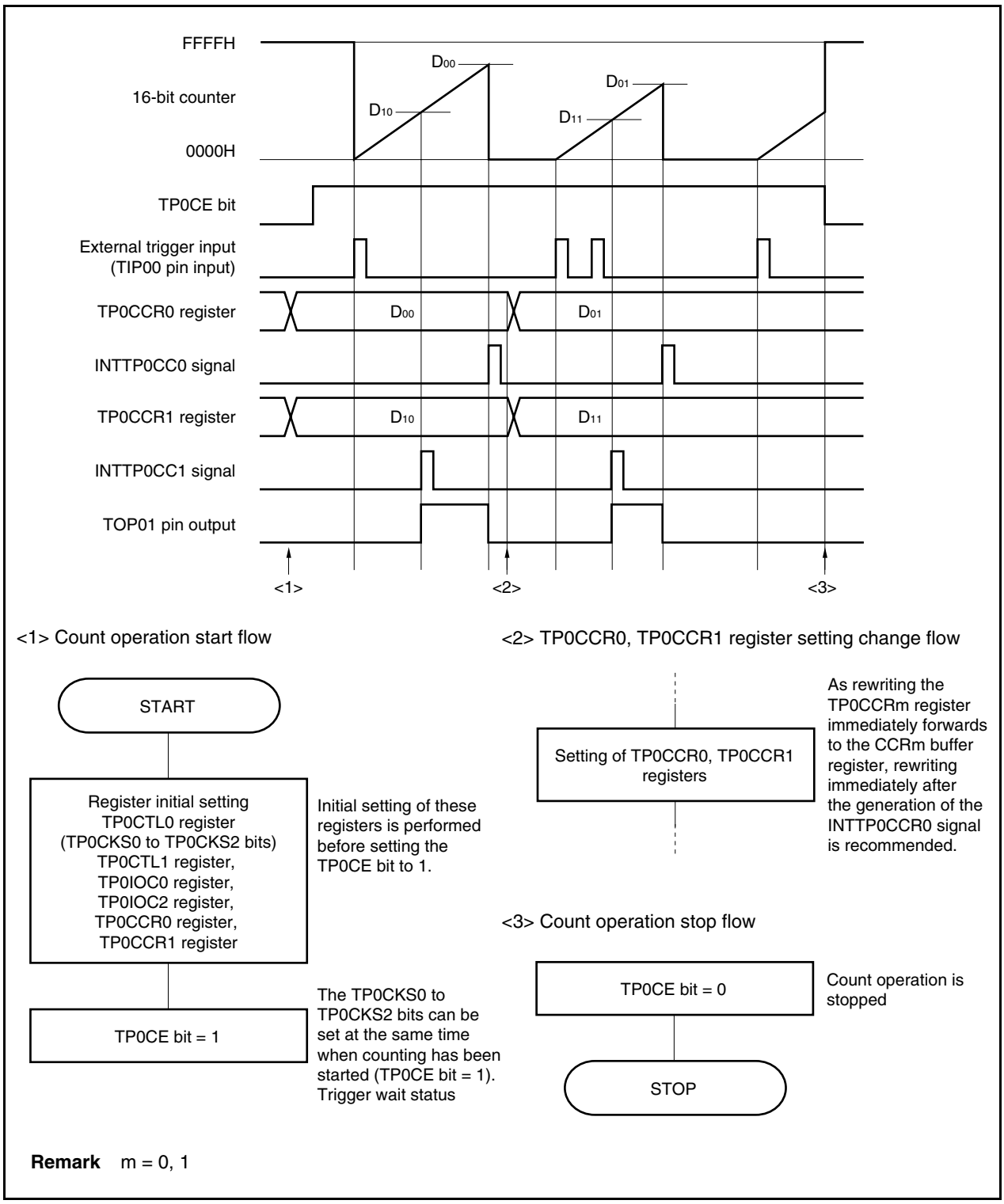
Output delay period = $D_1 \times$ Count clock cycle

Remark TMP0 I/O control register 1 (TP0IOC1) and TMP0 option register 0 (TP0OPT0) are not used in the one-shot pulse output mode.

Downloaded from Arrow.com.

(1) Operation flow in one-shot pulse output mode

<R> **Figure 6-23. Software Processing Flow in One-Shot Pulse Output Mode**



Remark m = 0, 1

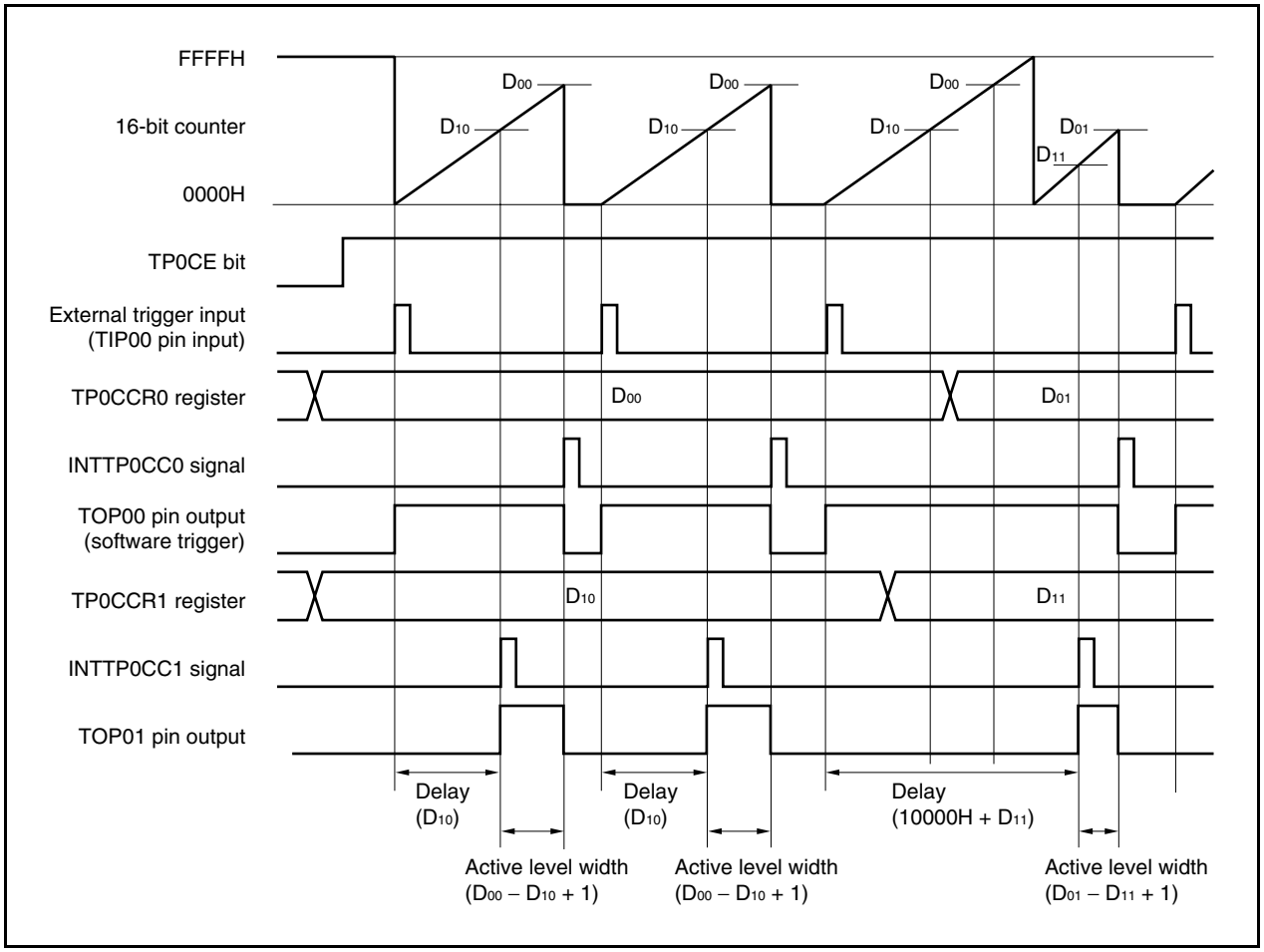
Downloaded from Arrow.com.

(2) Operation timing in one-shot pulse output mode

(a) Note on rewriting TP0CCRa register

To change the set value of the TP0CCRa register to a smaller value, stop counting once, and then change the set value.

If the value of the TP0CCRa register is rewritten to a smaller value during counting, the 16-bit counter may overflow.



<R>

When the TP0CCR0 register is rewritten from D_{00} to D_{01} and the TP0CCR1 register from D_{10} to D_{11} where $D_{00} > D_{01}$ and $D_{10} > D_{11}$, if the TP0CCR1 register is rewritten when the count value of the 16-bit counter is greater than D_{11} and less than D_{10} and if the TP0CCR0 register is rewritten when the count value is greater than D_{01} and less than D_{00}, each set value is reflected as soon as the register has been rewritten and compared with the count value. The counter counts up to FFFFH and then counts up again from 0000H. When the count value matches D_{11}, the counter generates the INTTP0CC1 signal and asserts the TOP01 pin. When the count value matches D_{01}, the counter generates the INTTP0CC0 signal, deasserts the TOP01 pin, and stops counting.

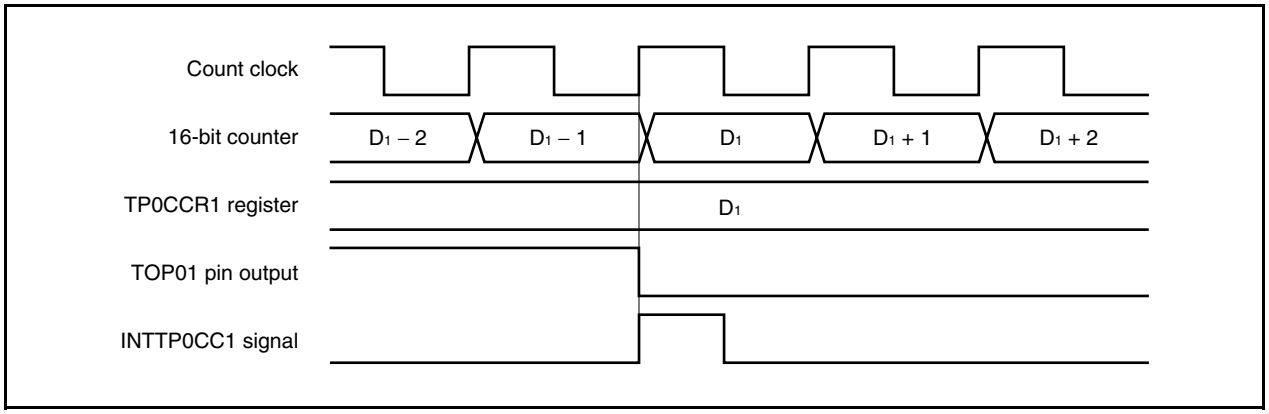
Therefore, the counter may output a pulse with a delay period or active period different from that of the one-shot pulse that is originally expected.

Remark a = 0, 1

Downloaded from Arrow.com.

(b) Generation timing of compare match interrupt request signal (INTTP0CC1)

The generation timing of the INTTP0CC1 signal in the one-shot pulse output mode is different from other INTTP0CC1 signals; the INTTP0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TP0CCR1 register.



Usually, the INTTP0CC1 signal is generated when the 16-bit counter counts up next time after its count value matches the value of the TP0CCR1 register.

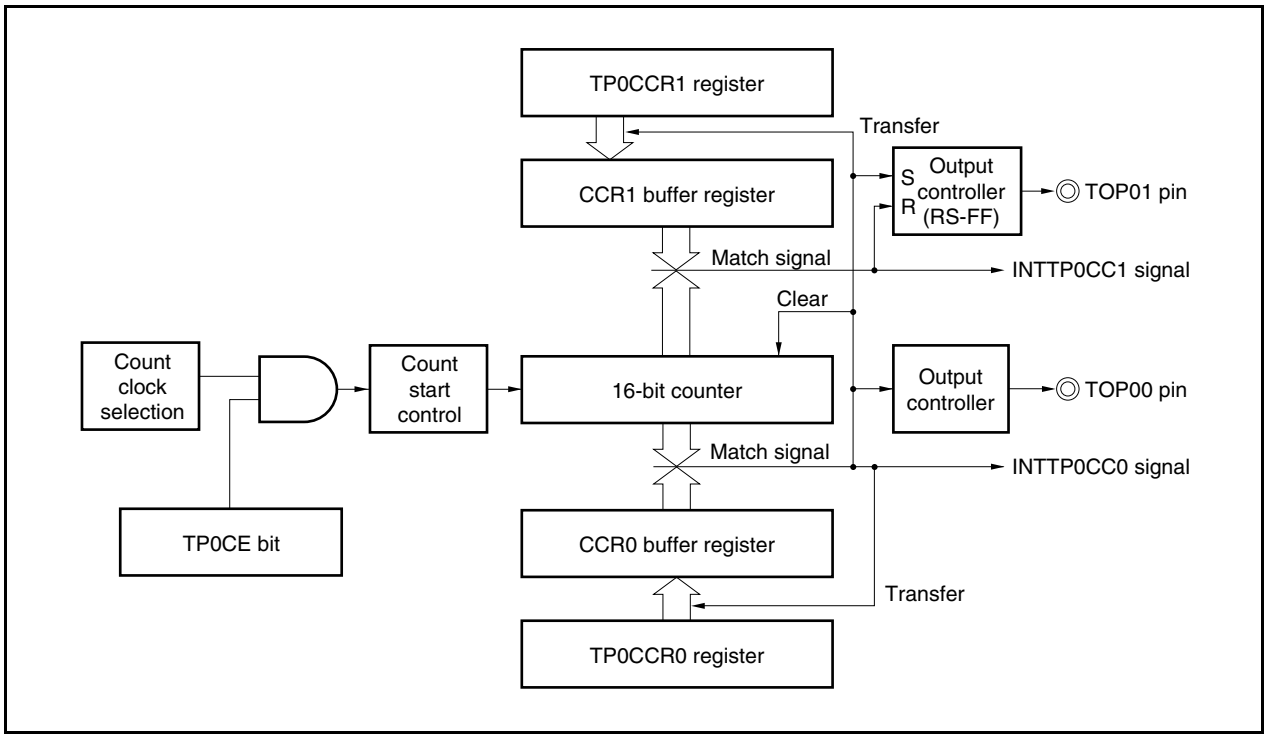
In the one-shot pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the TOP01 pin.

Downloaded from Arrow.com.

6.5.5 PWM output mode (TP0MD2 to TP0MD0 bits = 100)

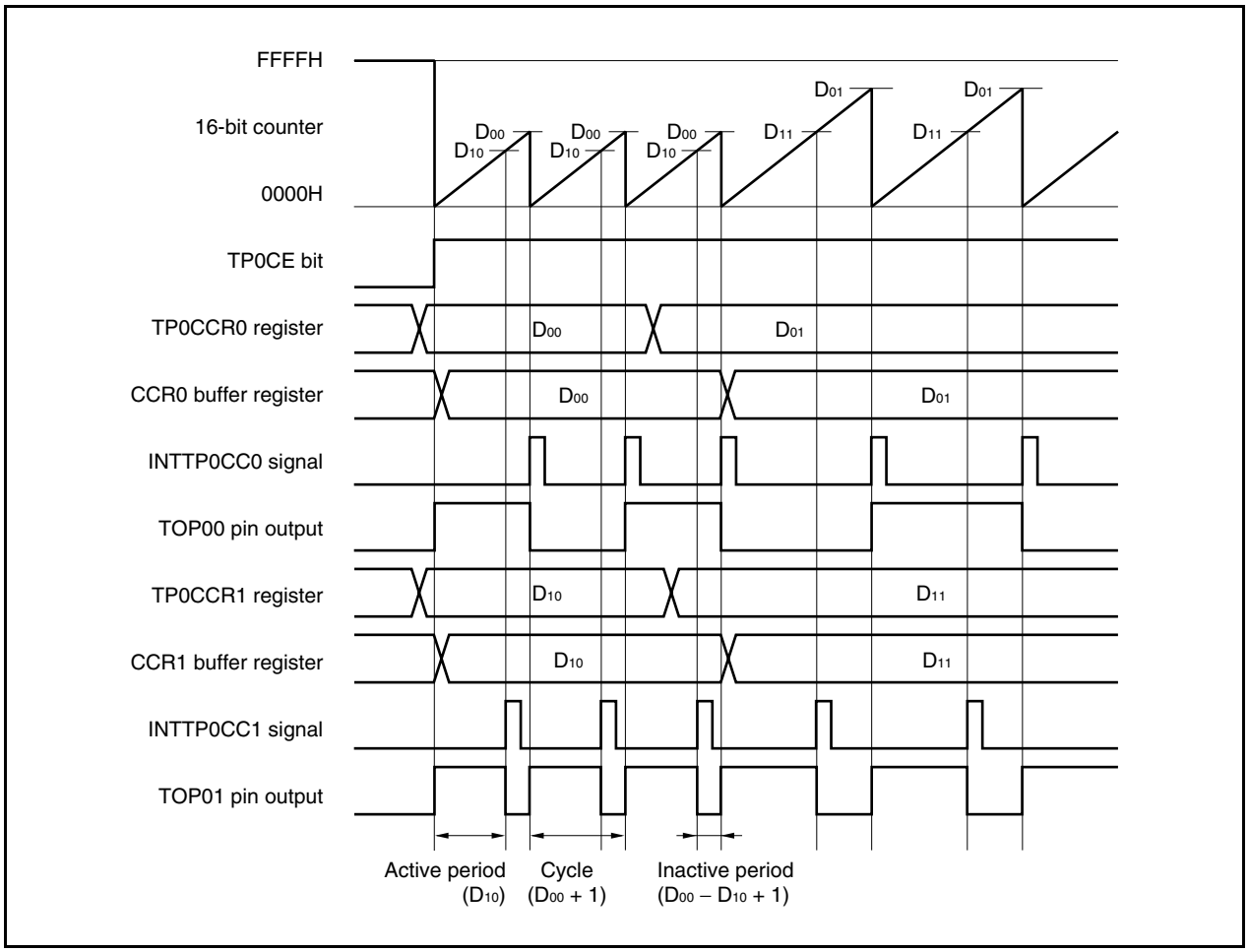
In the PWM output mode, a PWM waveform is output from the TOP01 pin when the TP0CTL0.TP0CE bit is set to 1. In addition, a pulse with one cycle of the PWM waveform as half its cycle is output from the TOP00 pin.

Figure 6-24. Configuration in PWM Output Mode



Downloaded from Arrow.com.

Figure 6-25. Basic Timing in PWM Output Mode



When the TP0CE bit is set to 1, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a PWM waveform from the TOP01 pin.

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TP0CCR1 register) × Count clock cycle
Cycle = (Set value of TP0CCR0 register + 1) × Count clock cycle
Duty factor = (Set value of TP0CCR1 register)/(Set value of TP0CCR0 register + 1)

The PWM waveform can be changed by rewriting the TP0CCRa register while the counter is operating. The newly written value is reflected when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

The compare match interrupt request signal INTTP0CC0 is generated when the 16-bit counter counts next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal INTTP0CC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TP0CCRa register is transferred to the CCRa buffer register when the count value of the 16-bit counter matches the value of the CCRa buffer register and the 16-bit counter is cleared to 0000H.

Remark a = 0, 1

Downloaded from Arrow.com.

Figure 6-26. Register Setting in PWM Output Mode (1/2)

(a) TMP0 control register 0 (TP0CTL0)

	TP0CE					TP0CKS2	TP0CKS1	TP0CKS0
TP0CTL0	0/1	0	0	0	0	0/1	0/1	0/1

Select count clock[Note]

0: Stop counting
1: Enable counting

Note The setting is invalid when the TP0CTL1.TP0EEE bit = 1.

(b) TMP0 control register 1 (TP0CTL1)

<R>

		TP0EST	TP0EEE			TP0MD2	TP0MD1	TP0MD0
TP0CTL1	0	0	0/1	0	0	1	0	0

1, 0, 0:
PWM output mode

0: Operate on count clock selected by TP0CKS0 to TP0CKS2 bits
1: Count with external event count input signal

(c) TMP0 I/O control register 0 (TP0IOC0)

					TP0OL1	TP0OE1	TP0OL0	TP0OE0
TP0IOC0	0	0	0	0	0/1	0/1	0/1	0/1

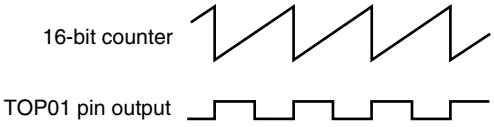
0: Disable TOP00 pin output
1: Enable TOP00 pin output

Setting of output level while operation of TOP00 pin is disabled
0: Low level
1: High level

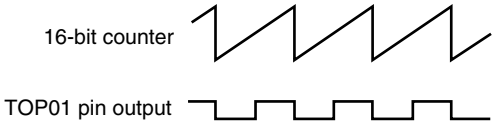
0: Disable TOP01 pin output
1: Enable TOP01 pin output

Specifies active level of TOP01 pin output
0: Active-high
1: Active-low

- When TP0OL1 bit = 0



- When TP0OL1 bit = 1



Downloaded from Arrow.com.

Figure 6-26. Register Setting in PWM Output Mode (2/2)

(d) TMP0 I/O control register 2 (TP0IOC2)

					TP0EES1	TP0EES0	TP0ETS1	TP0ETS0
TP0IOC2	0	0	0	0	0/1	0/1	0	0

Select valid edge of external event count input.

(e) TMP0 counter read buffer register (TP0CNT)

The value of the 16-bit counter can be read by reading the TP0CNT register.

(f) TMP0 capture/compare registers 0 and 1 (TP0CCR0 and TP0CCR1)

If D_0 is set to the TP0CCR0 register and D_1 to the TP0CCR1 register, the cycle and active level of the PWM waveform are as follows.

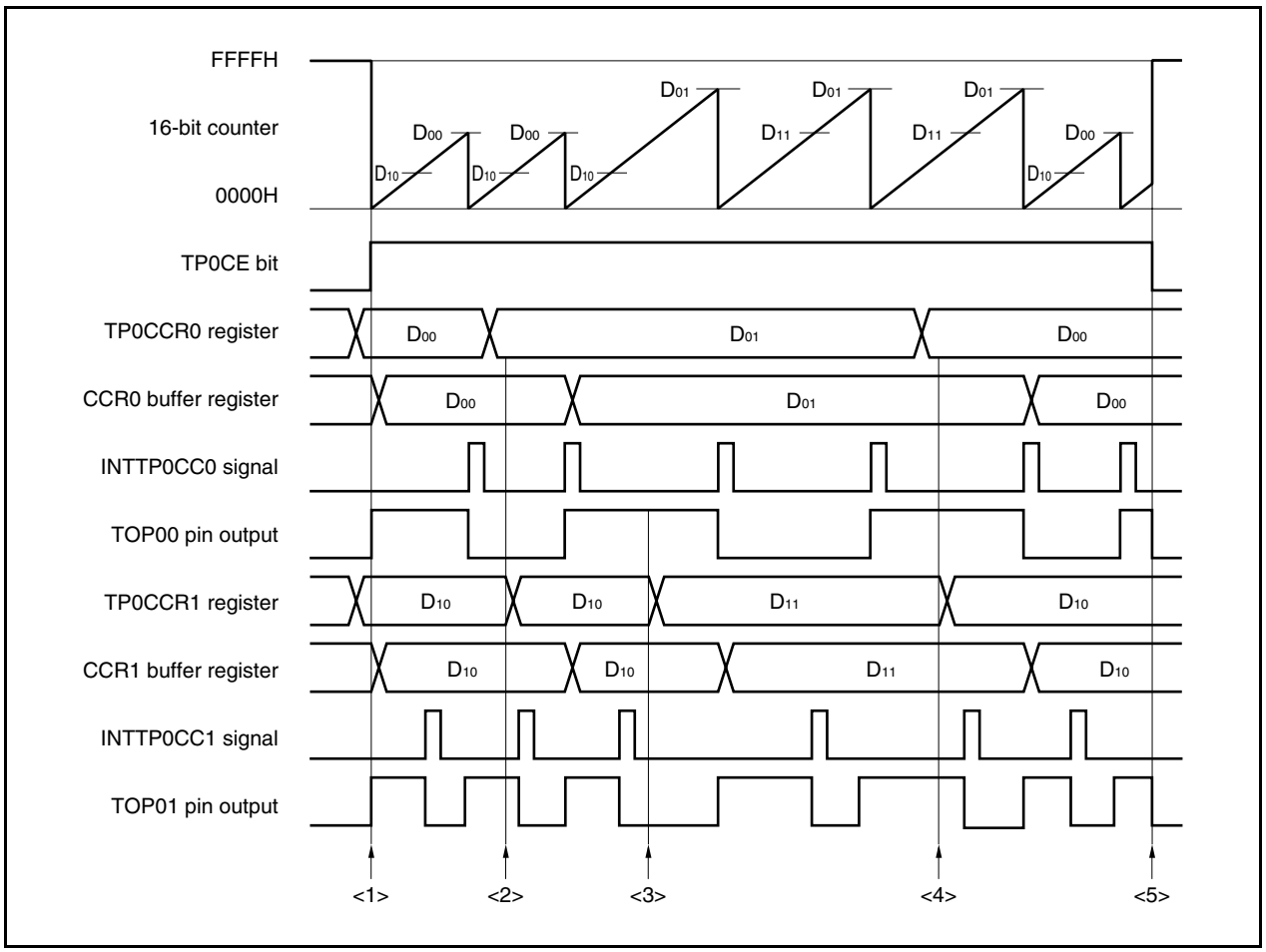
Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

Remark TMP0 I/O control register 1 (TP0IOC1) and TMP0 option register 0 (TP0OPT0) are not used in the PWM output mode.

Downloaded from Arrow.com.

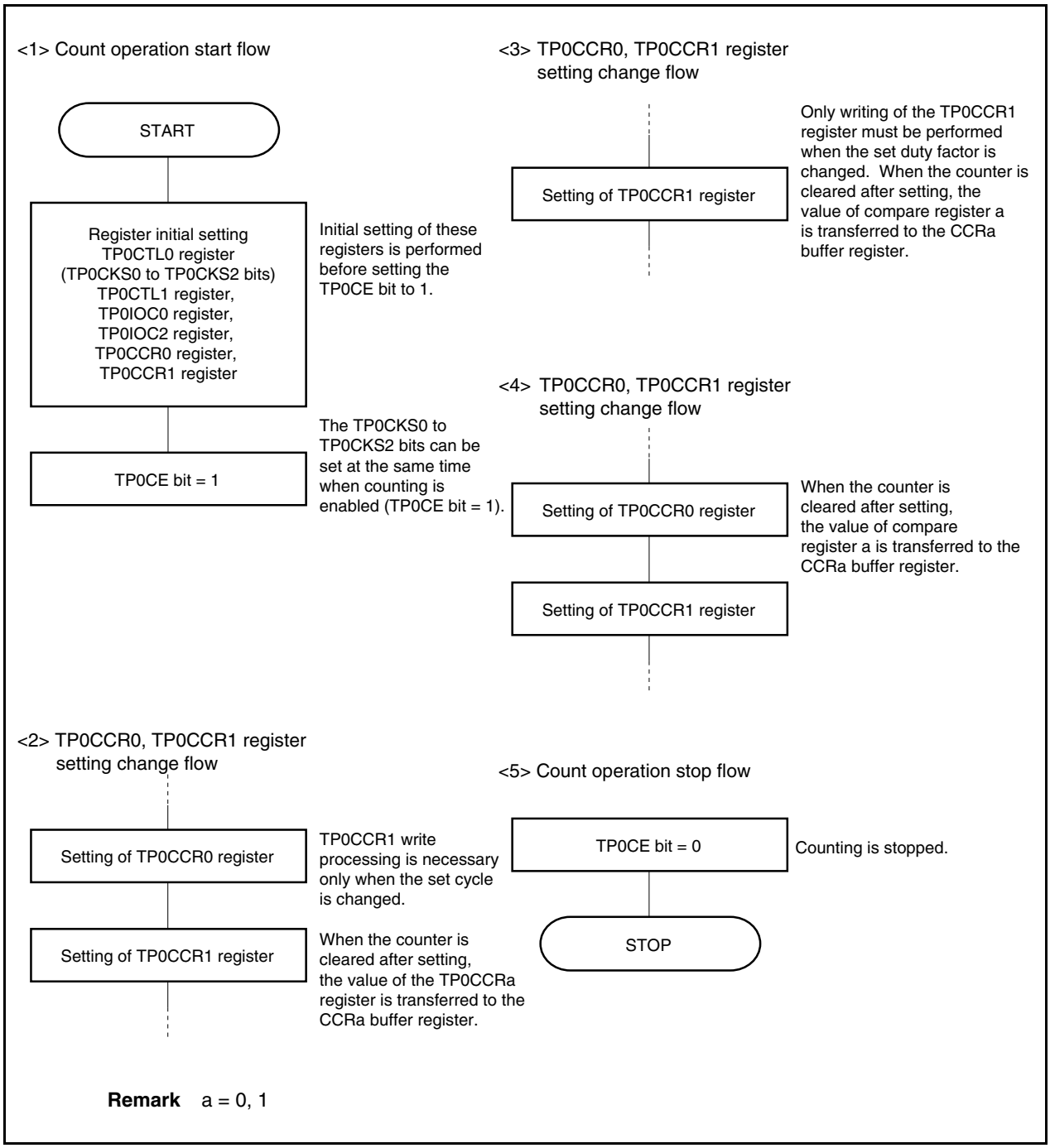
(1) Operation flow in PWM output mode

Figure 6-27. Software Processing Flow in PWM Output Mode (1/2)



Downloaded from Arrow.com.

Figure 6-27. Software Processing Flow in PWM Output Mode (2/2)



Remark a = 0, 1

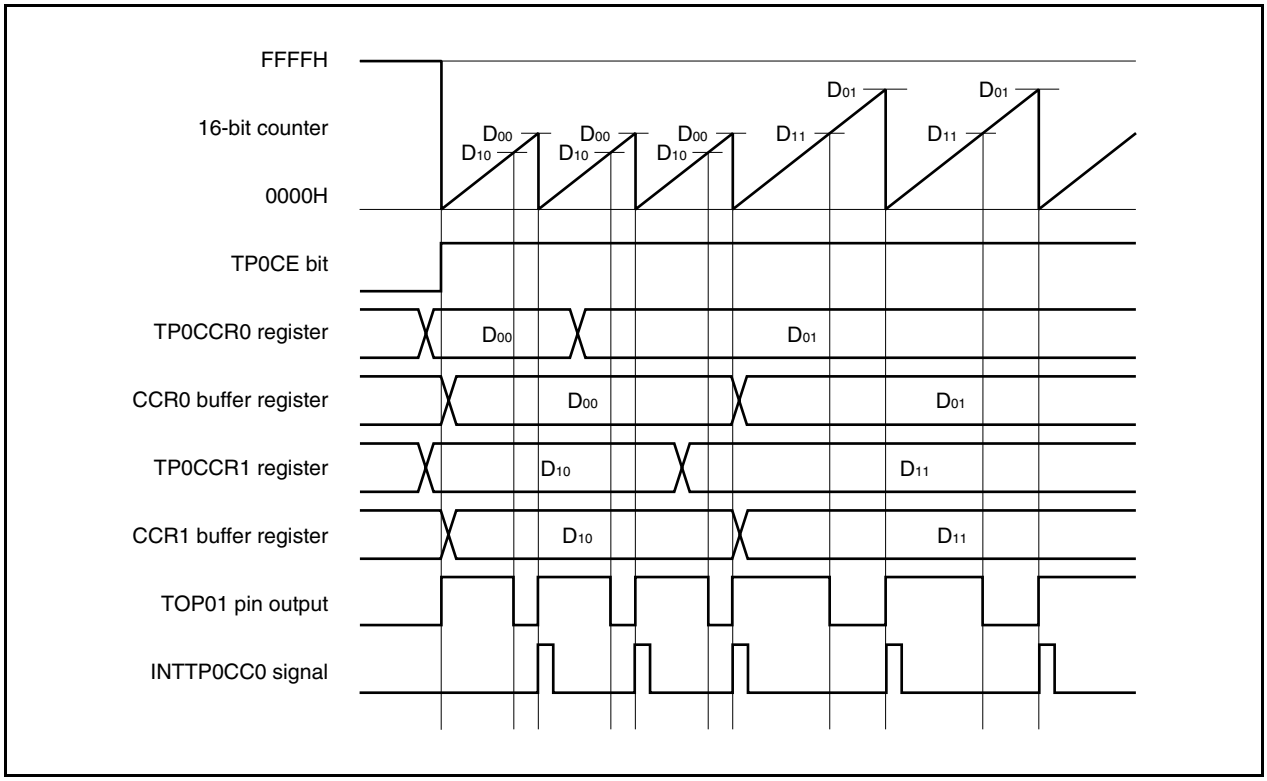
Downloaded from Arrow.com.

(2) PWM output mode operation timing

(a) Changing pulse width during operation

To change the PWM waveform while the counter is operating, write the TP0CCR1 register last.

Rewrite the TP0CCRa register after writing the TP0CCR1 register after the INTTP0CC1 signal is detected.



To transfer data from the TP0CCRa register to the CCRa buffer register, the TP0CCR1 register must be written.

To change both the cycle and active level of the PWM waveform at this time, first set the cycle to the TP0CCR0 register and then set the active level to the TP0CCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TP0CCR0 register, and then write the same value to the TP0CCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TP0CCR1 register has to be set.

After data is written to the TP0CCR1 register, the value written to the TP0CCRa register is transferred to the CCRa buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

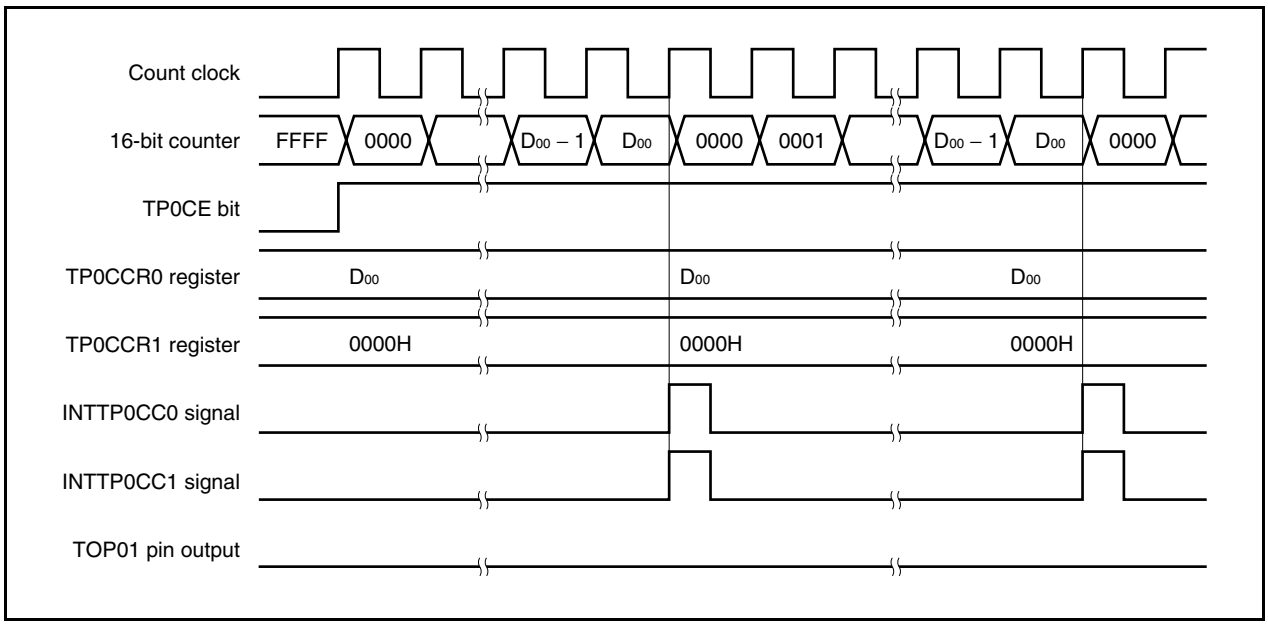
To write the TP0CCR0 or TP0CCR1 register again after writing the TP0CCR1 register once, do so after the INTTP0CC0 signal is generated. Otherwise, the value of the CCRa buffer register may become undefined because the timing of transferring data from the TP0CCRa register to the CCRa buffer register conflicts with writing the TP0CCRa register.

Remark a = 0, 1

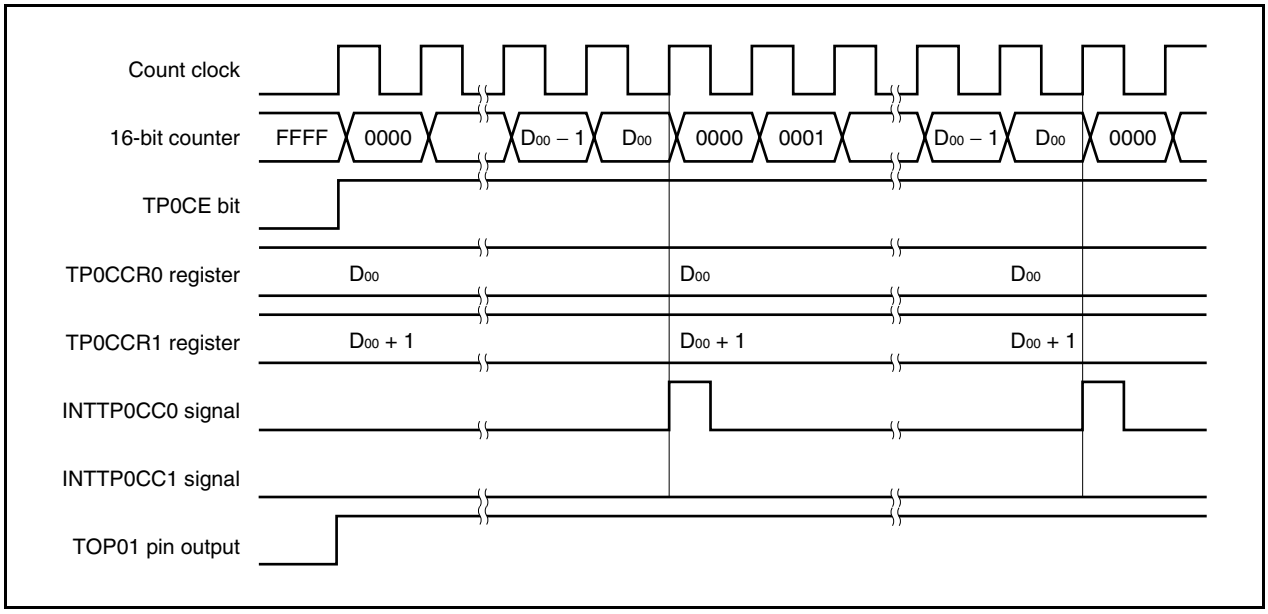
Downloaded from Arrow.com.

(b) 0%/100% output of PWM waveform

To output a 0% waveform, set the TP0CCR1 register to 0000H. If the set value of the TP0CCR0 register is FFFFH, the INTTP0CC1 signal is generated periodically.



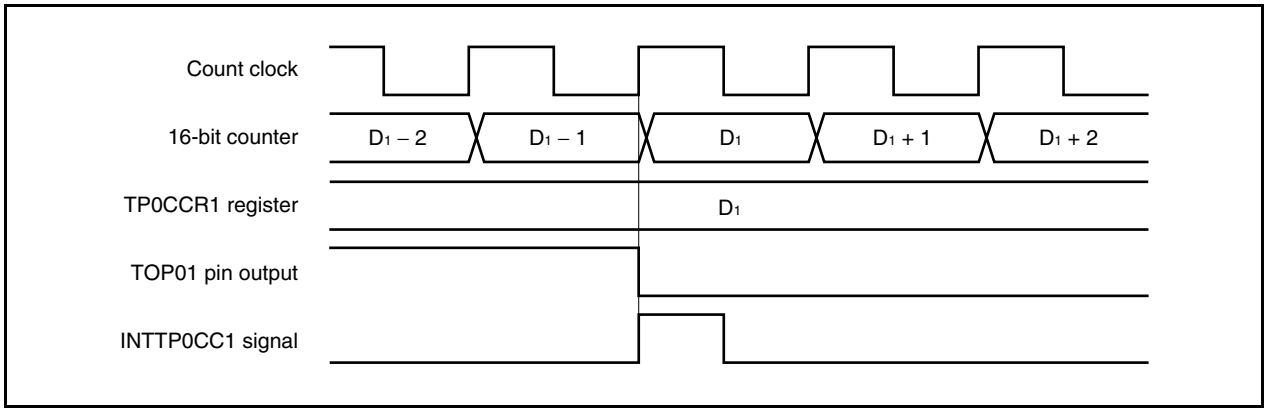
To output a 100% waveform, set a value of (set value of TP0CCR0 register + 1) to the TP0CCR1 register. If the set value of the TP0CCR0 register is FFFFH, 100% output cannot be produced.



Downloaded from Arrow.com.

(c) Generation timing of compare match interrupt request signal (INTTP0CC1)

The timing of generation of the INTTP0CC1 signal in the PWM output mode differs from the timing of other INTTP0CC1 signals; the INTTP0CC1 signal is generated when the count value of the 16-bit counter matches the value of the TP0CCR1 register.



Usually, the INTTP0CC1 signal is generated in synchronization with the next counting up after the count value of the 16-bit counter matches the value of the TP0CCR1 register.

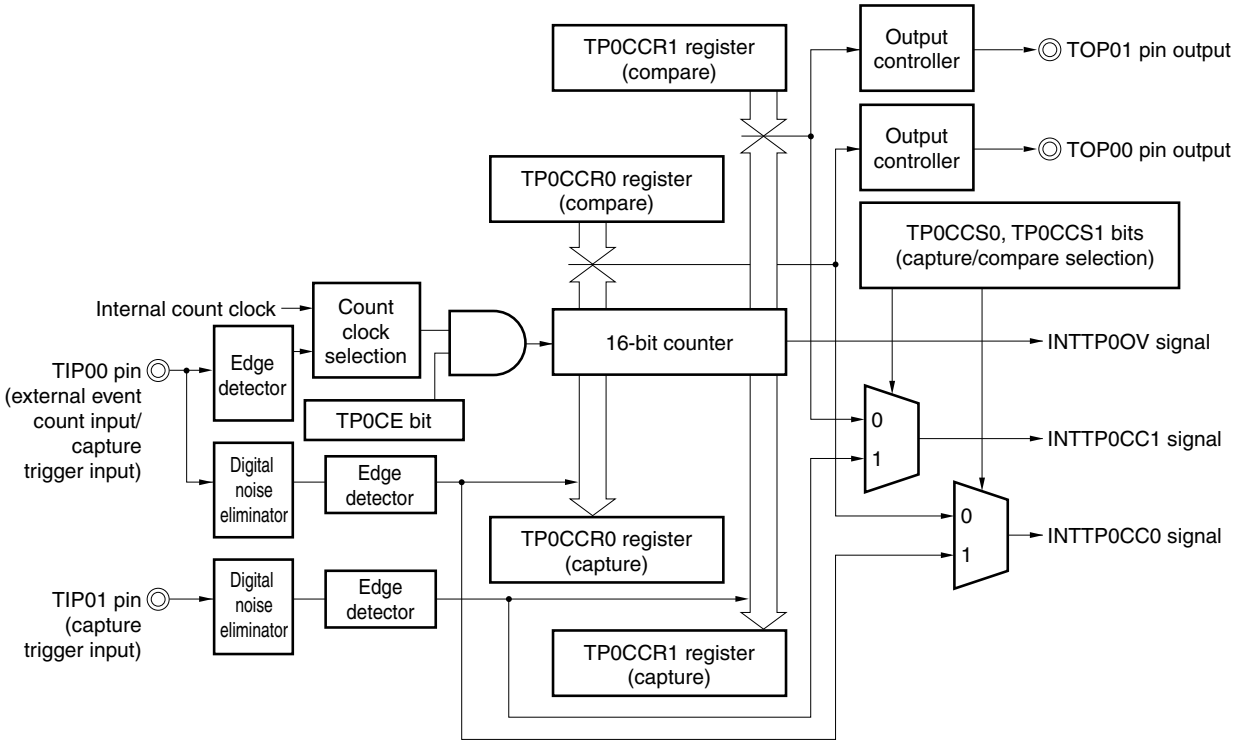
In the PWM output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the output signal of the TOP01 pin.

Downloaded from Arrow.com.

6.5.6 Free-running timer mode (TP0MD2 to TP0MD0 bits = 101)

In the free-running timer mode, 16-bit timer/event counter P starts counting when the TP0CTL0.TP0CE bit is set to 1. At this time, the TP0CCRa register can be used as a compare register or a capture register, depending on the setting of the TP0OPT0.TP0CCS0 and TP0OPT0.TP0CCS1 bits.

Figure 6-28. Configuration in Free-Running Timer Mode



Remark a = 0, 1

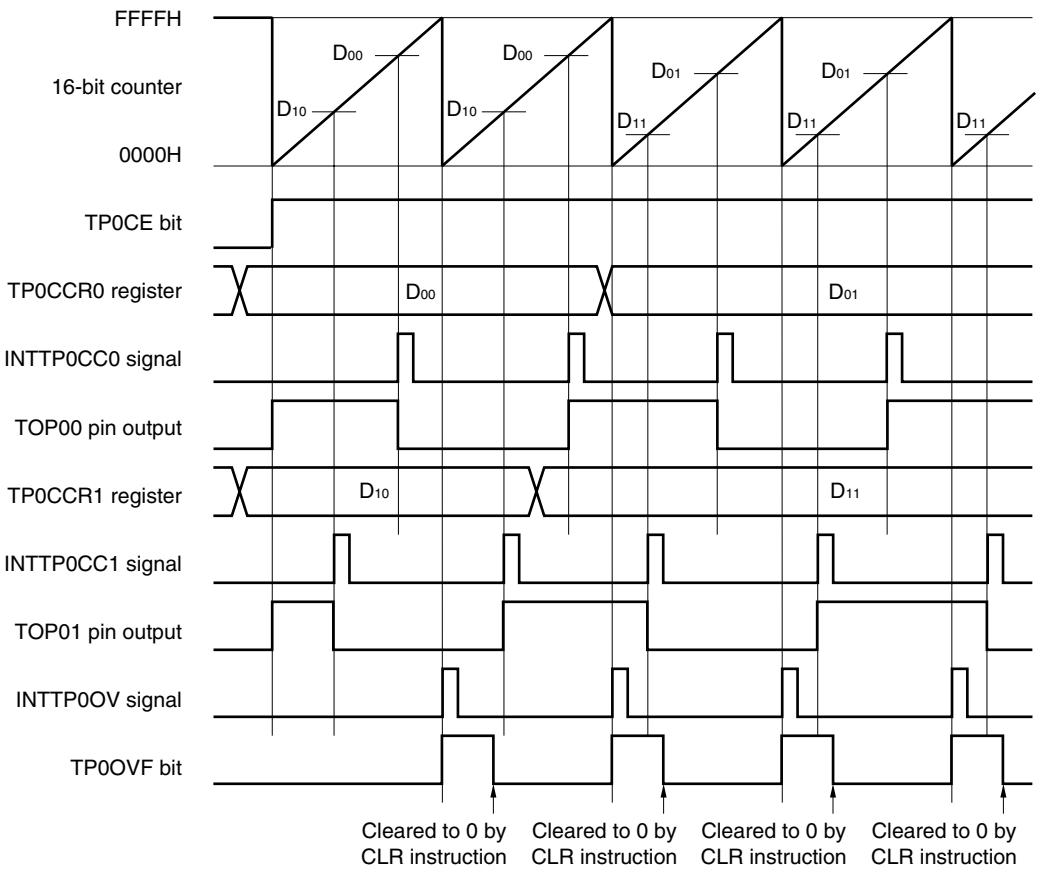
Downloaded from Arrow.com.

When the TP0CE bit is set to 1, 16-bit timer/event counter P starts counting, and the output signals of the TOP00 and TOP01 pins are inverted. When the count value of the 16-bit counter later matches the set value of the TP0CCRa register, a compare match interrupt request signal (INTTP0CCa) is generated, and the output signal of the TOP0a pin is inverted.

The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTP0OV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TP0OPT0.TP0OVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction by software.

The TP0CCRa register can be rewritten while the counter is operating. If it is rewritten, the new value is reflected at that time, and compared with the count value.

Figure 6-29. Basic Timing in Free-Running Timer Mode (Compare Function)



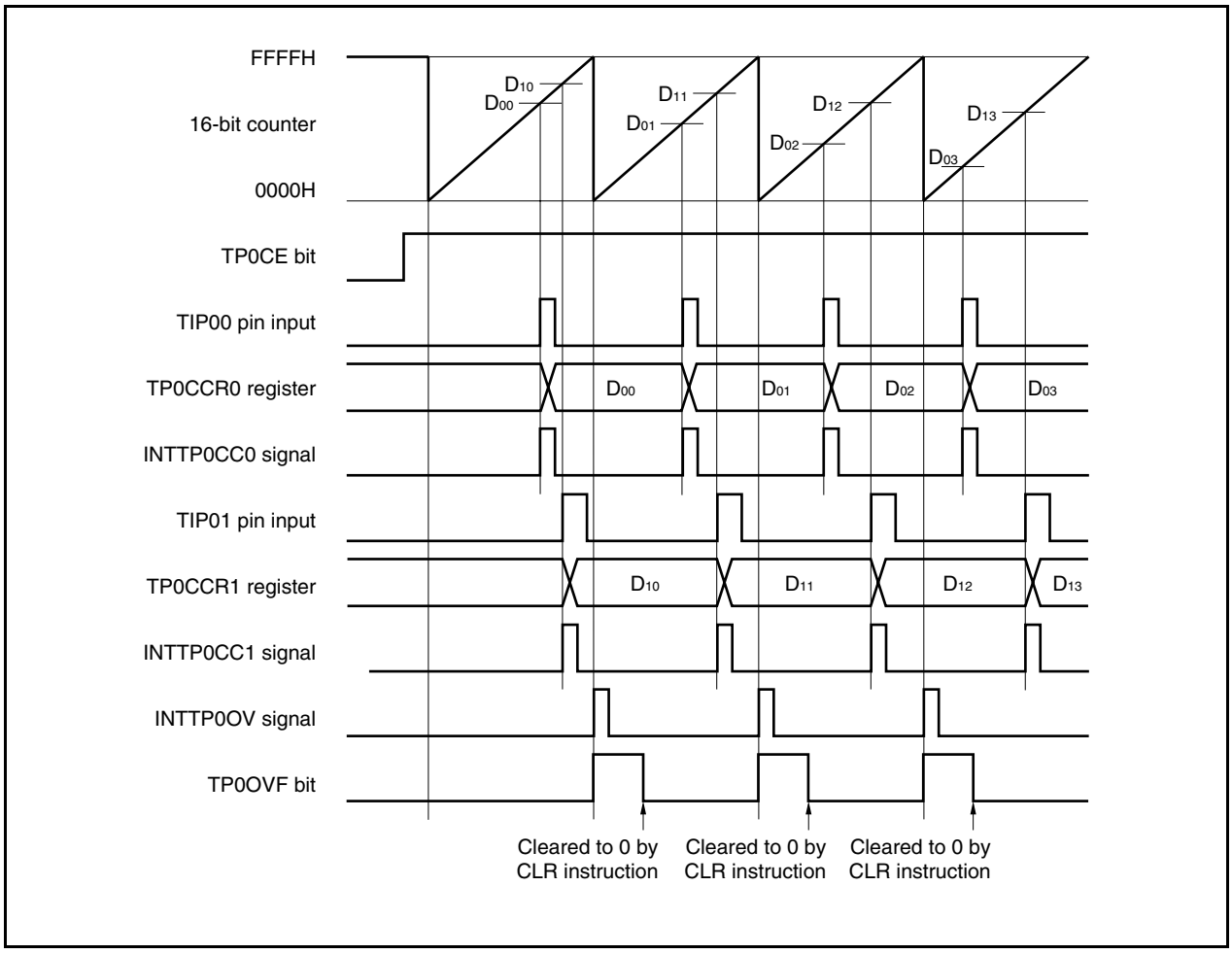
Remark a = 0, 1

Downloaded from Arrow.com.

When the TP0CE bit is set to 1, the 16-bit counter starts counting. When the valid edge input to the TIP0a pin is detected, the count value of the 16-bit counter is stored in the TP0CCRa register, and a capture interrupt request signal (INTTP0CCa) is generated.

The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTP0OV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TP0OPT0.TP0OVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction by software.

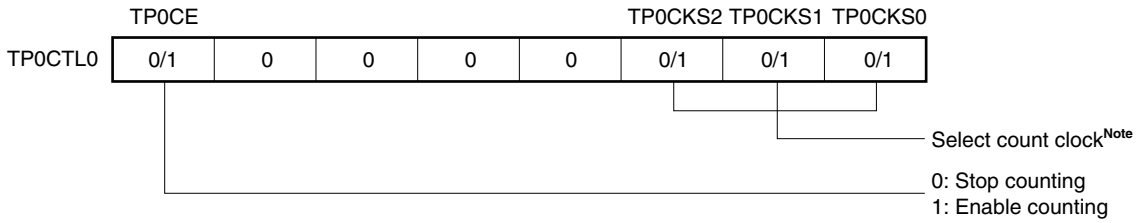
Figure 6-30. Basic Timing in Free-Running Timer Mode (Capture Function)



Downloaded from Arrow.com.

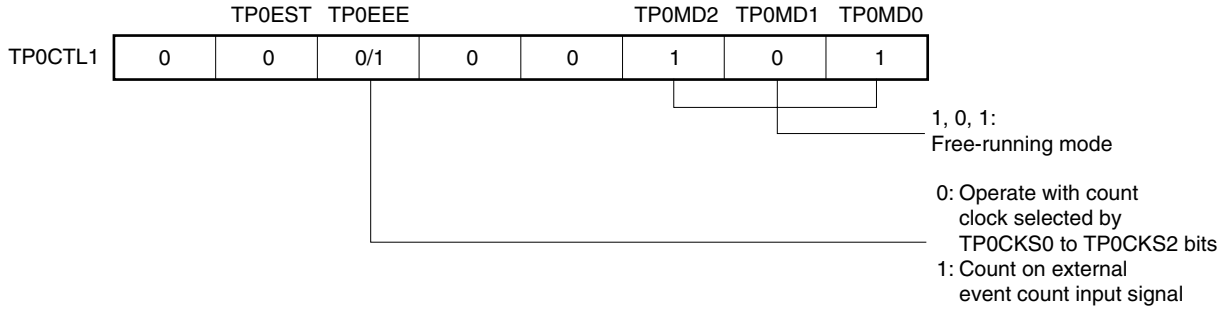
Figure 6-31. Register Setting in Free-Running Timer Mode (1/2)

(a) TMP0 control register 0 (TP0CTL0)

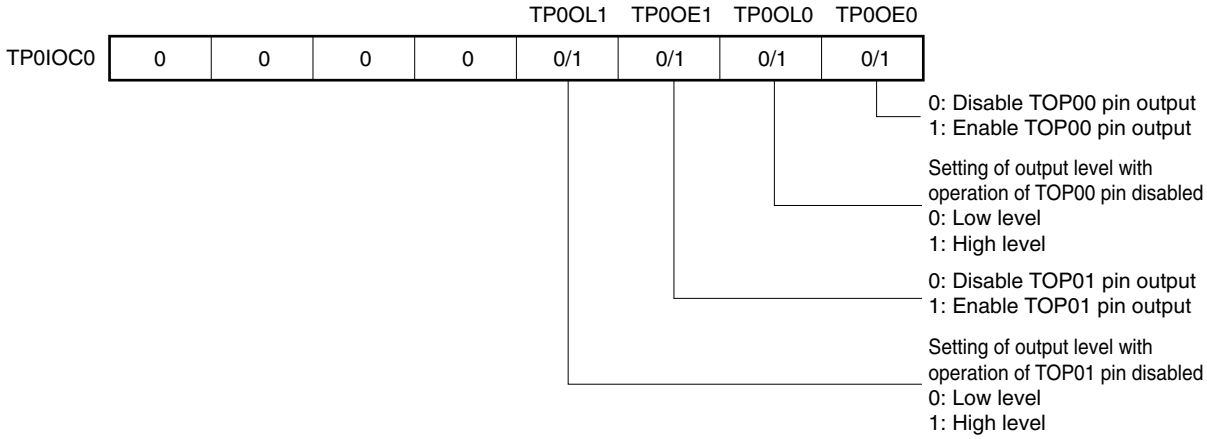


Note The setting is invalid when the TP0CTL1.TP0EEE bit = 1

(b) TMP0 control register 1 (TP0CTL1)



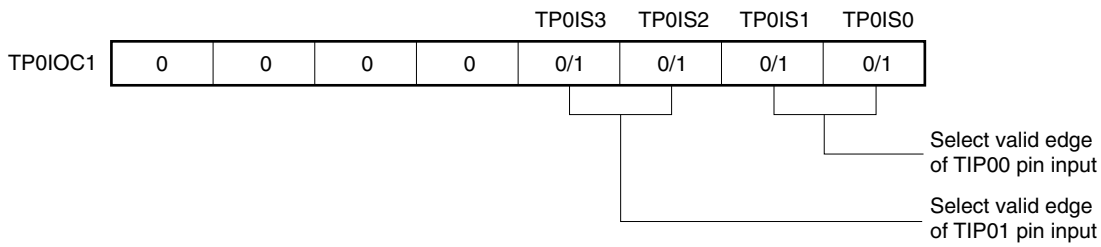
(c) TMP0 I/O control register 0 (TP0IOC0)



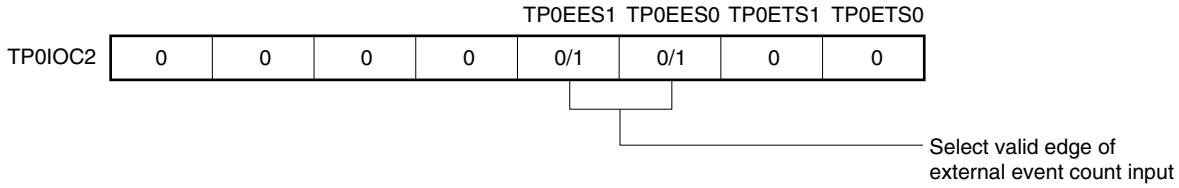
Downloaded from Arrow.com.

Figure 6-31. Register Setting in Free-Running Timer Mode (2/2)

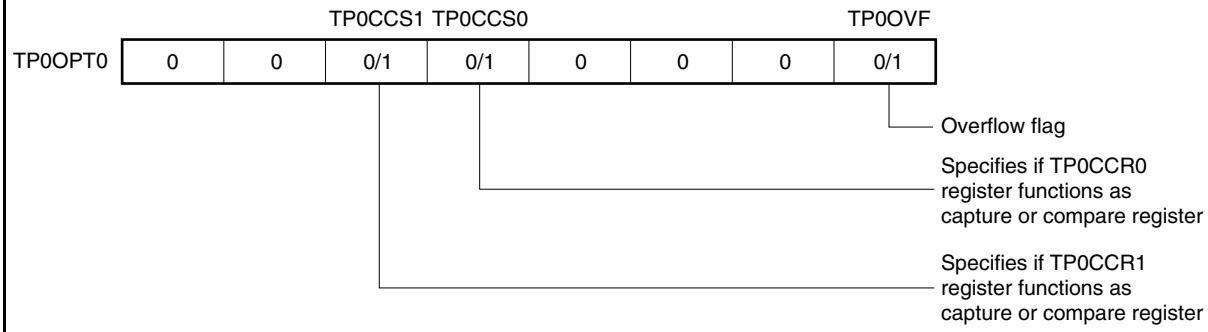
(d) TMP0 I/O control register 1 (TP0IOC1)



(e) TMP0 I/O control register 2 (TP0IOC2)



(f) TMP0 option register 0 (TP0OPT0)



(g) TMP0 counter read buffer register (TP0CNT)

The value of the 16-bit counter can be read by reading the TP0CNT register.

(h) TMP0 capture/compare registers 0 and 1 (TP0CCR0 and TP0CCR1)

These registers function as capture registers or compare registers depending on the setting of the TP0OPT0.TP0CCSa bit.

When the registers function as capture registers, they store the count value of the 16-bit counter when the valid edge input to the TIP0a pin is detected.

When the registers function as compare registers and when D_a is set to the TP0CCRa register, the INTTP0CCa signal is generated when the counter reaches (D_a + 1), and the output signal of the TOP0a pin is inverted.

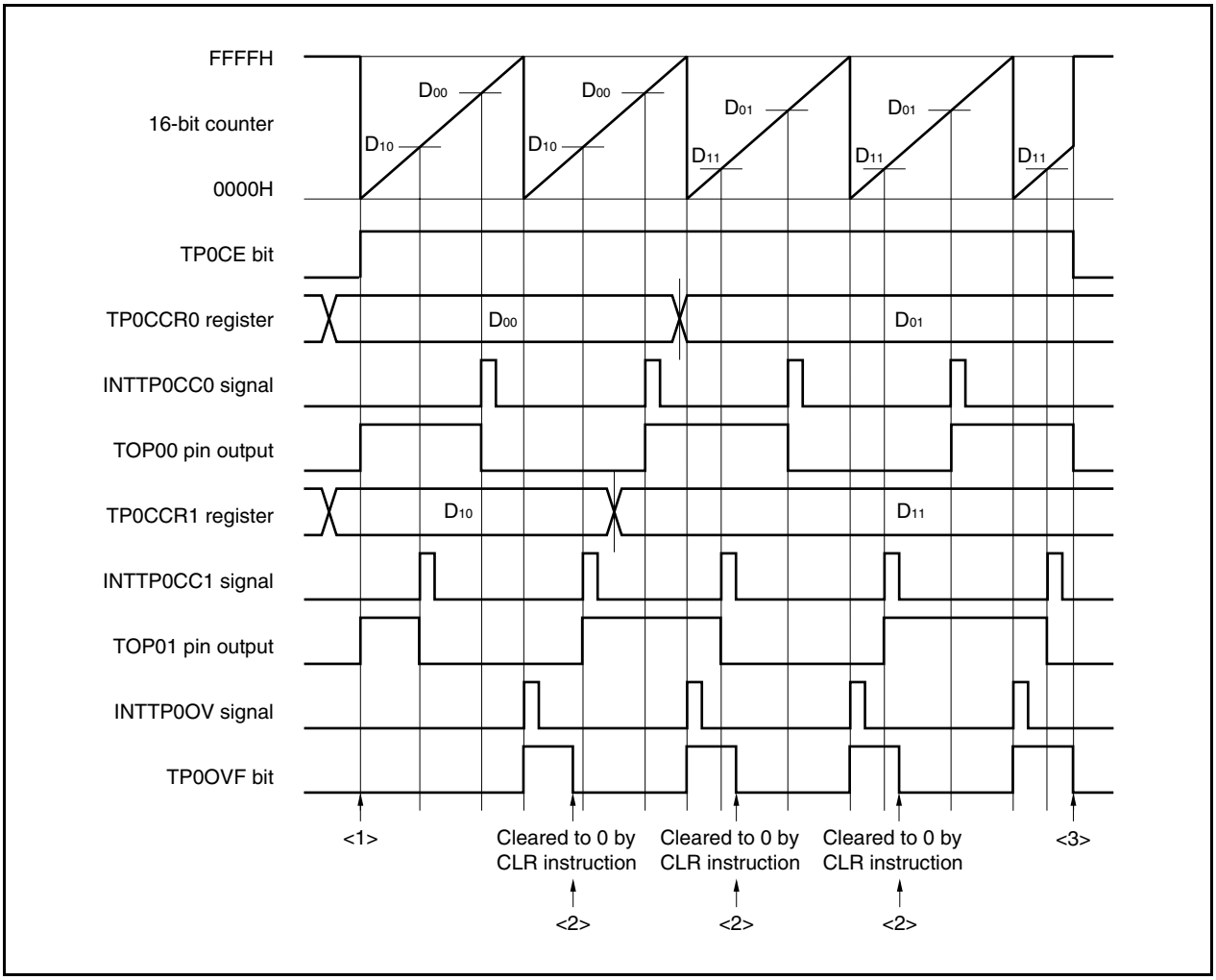
Remark a = 0, 1

Downloaded from Arrow.com.

(1) Operation flow in free-running timer mode

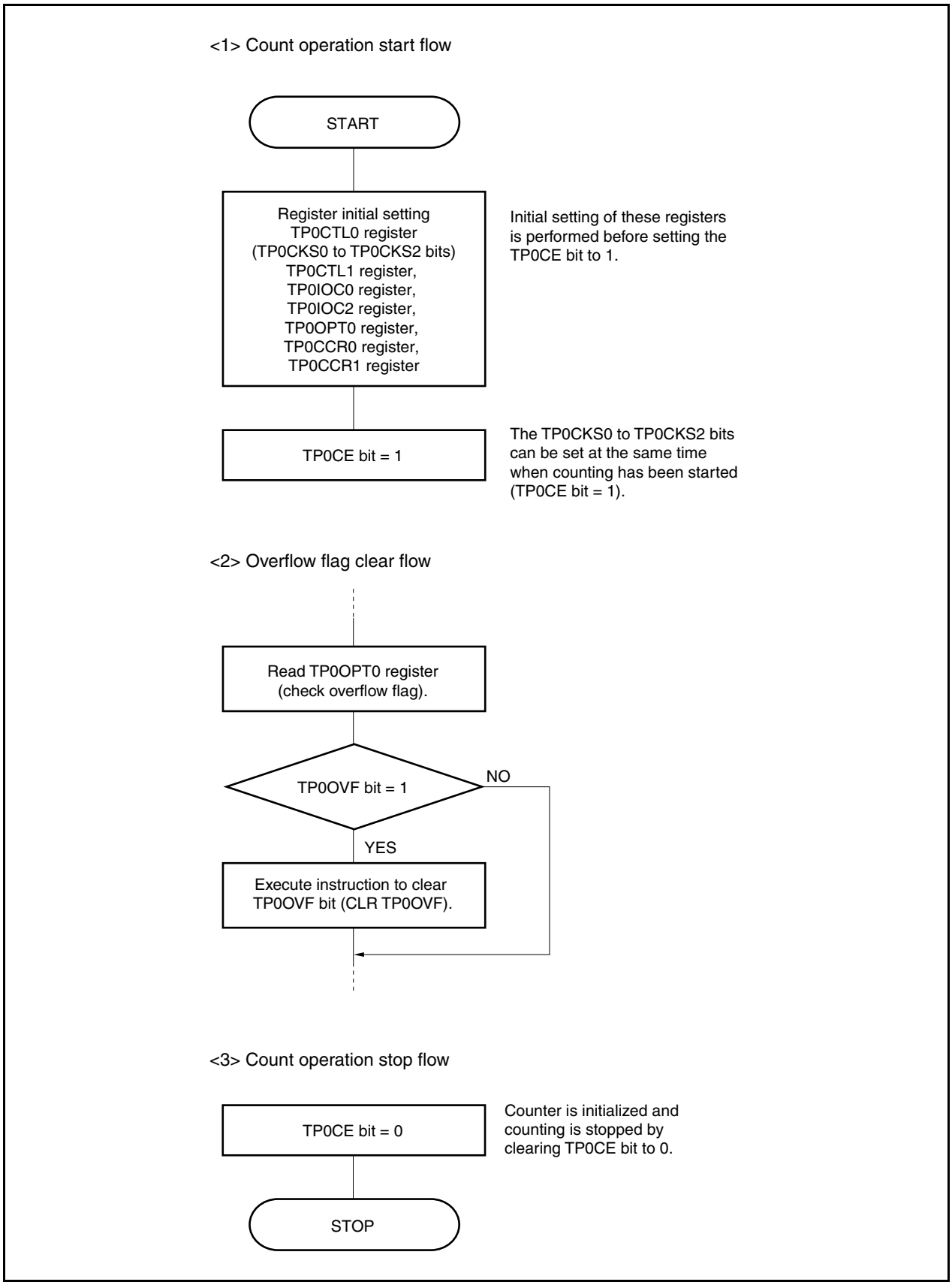
(a) When using capture/compare register as compare register

Figure 6-32. Software Processing Flow in Free-Running Timer Mode (Compare Function) (1/2)



Downloaded from Arrow.com.

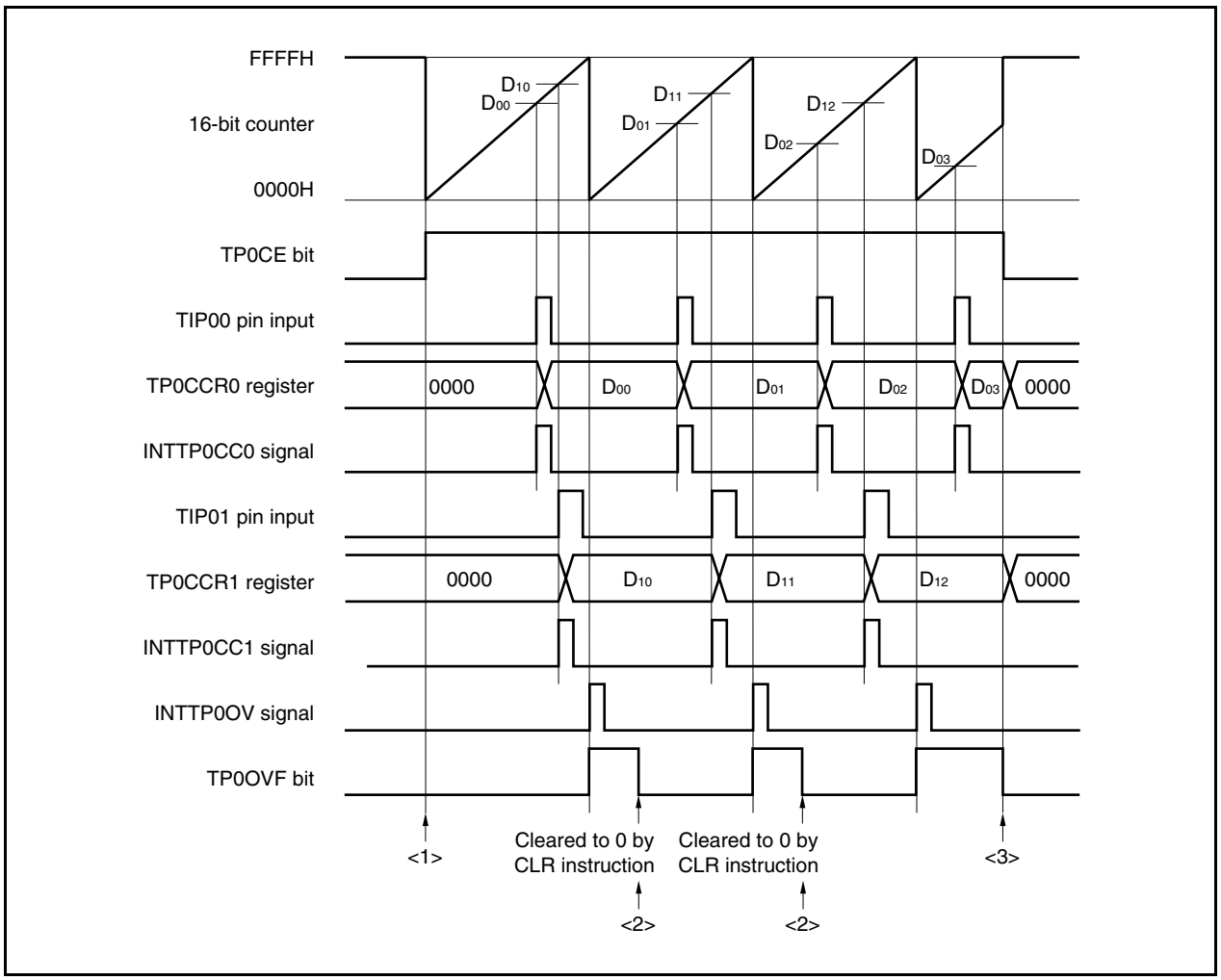
Figure 6-32. Software Processing Flow in Free-Running Timer Mode (Compare Function) (2/2)



Downloaded from Arrow.com.

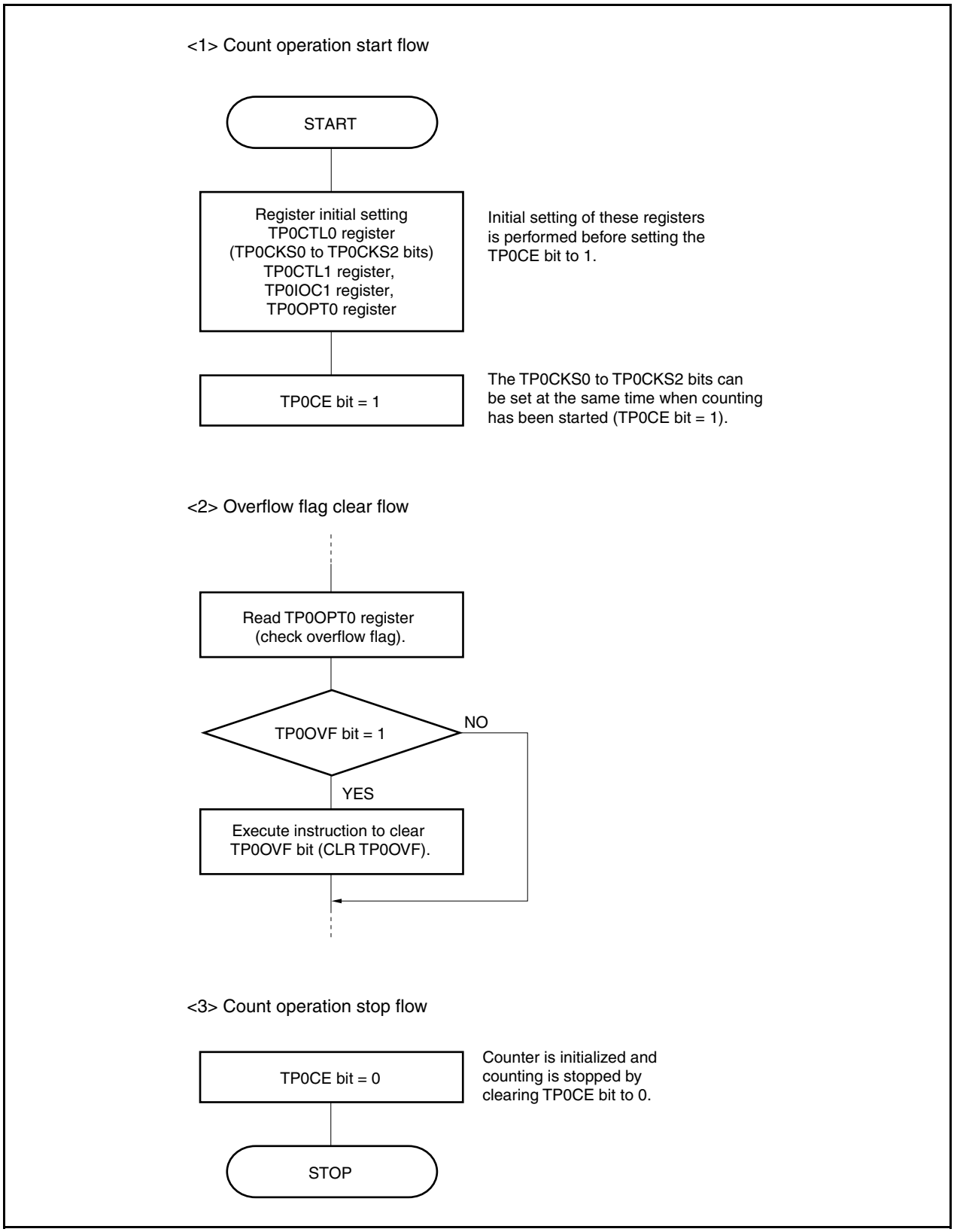
(b) When using capture/compare register as capture register

Figure 6-33. Software Processing Flow in Free-Running Timer Mode (Capture Function) (1/2)



Downloaded from Arrow.com.

Figure 6-33. Software Processing Flow in Free-Running Timer Mode (Capture Function) (2/2)

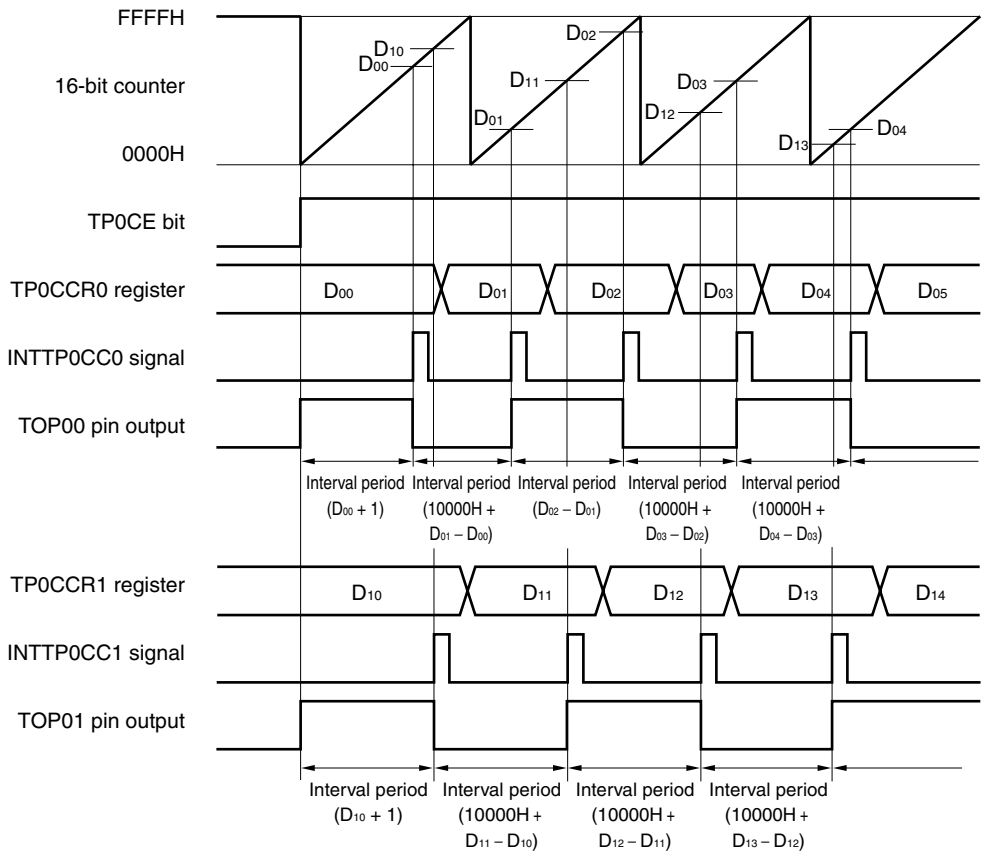


Downloaded from Arrow.com.

(2) Operation timing in free-running timer mode

(a) Interval operation with compare register

When 16-bit timer/event counter P is used as an interval timer with the TP0CCRa register used as a compare register, software processing is necessary for setting a comparison value to generate the next interrupt request signal each time the INTTP0CCa signal has been detected.



When performing an interval operation in the free-running timer mode, two intervals can be set with one channel.

To perform the interval operation, the value of the corresponding TP0CCRa register must be re-set in the interrupt servicing that is executed when the INTTP0CCa signal is detected.

The set value for re-setting the TP0CCRa register can be calculated by the following expression, where "D_a" is the interval period.

Compare register default value: $D_a - 1$

Value set to compare register second and subsequent time: Previous set value + D_a

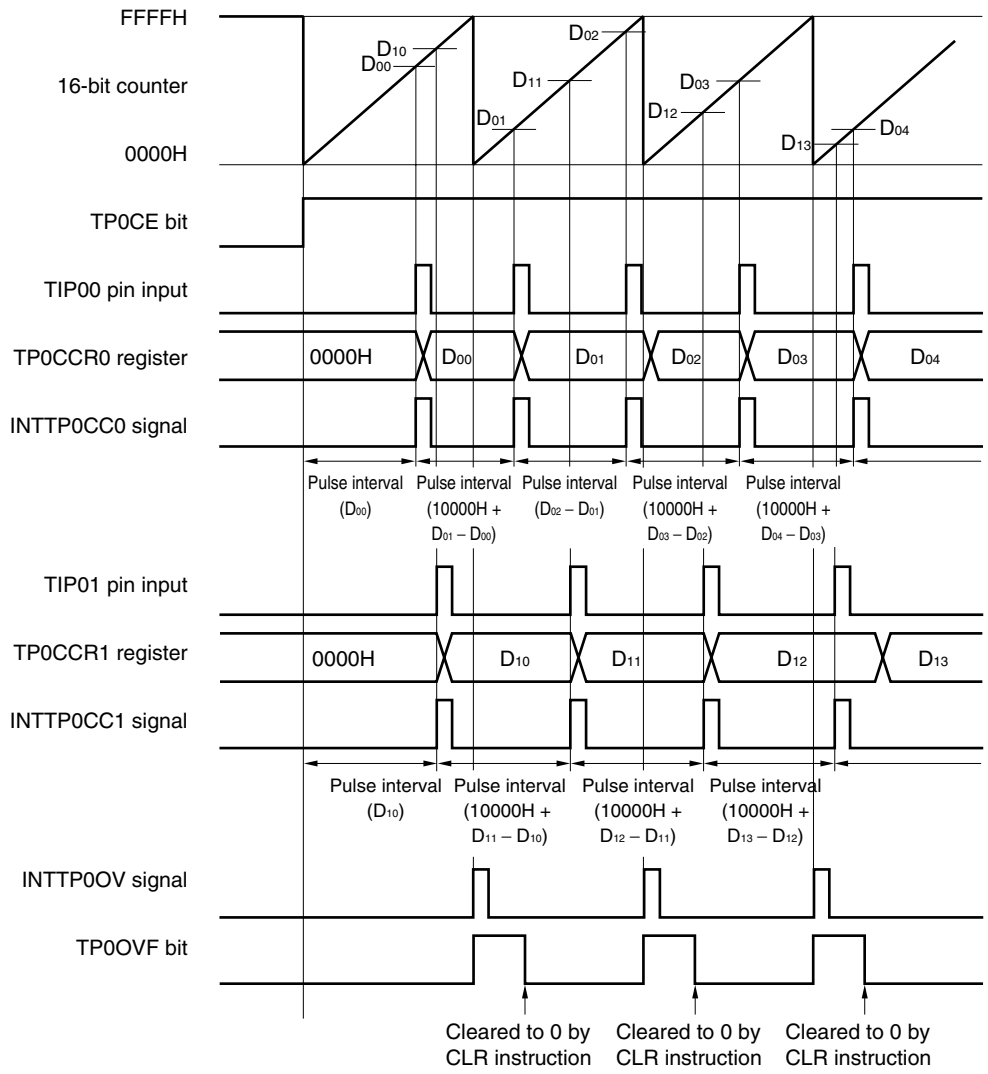
(If the calculation result is greater than FFFFH, subtract 10000H from the result and set this value to the register.)

Remark a = 0, 1

Downloaded from Arrow.com.

(b) Pulse width measurement with capture register

When pulse width measurement is performed with the TP0CCRa register used as a capture register, software processing is necessary for reading the capture register each time the INTTP0CCa signal has been detected and for calculating an interval.



When executing pulse width measurement in the free-running timer mode, two pulse widths can be measured with one channel.

To measure a pulse width, the pulse width can be calculated by reading the value of the TP0CCRa register in synchronization with the INTTP0CCa signal, and calculating the difference between the read value and the previously read value.

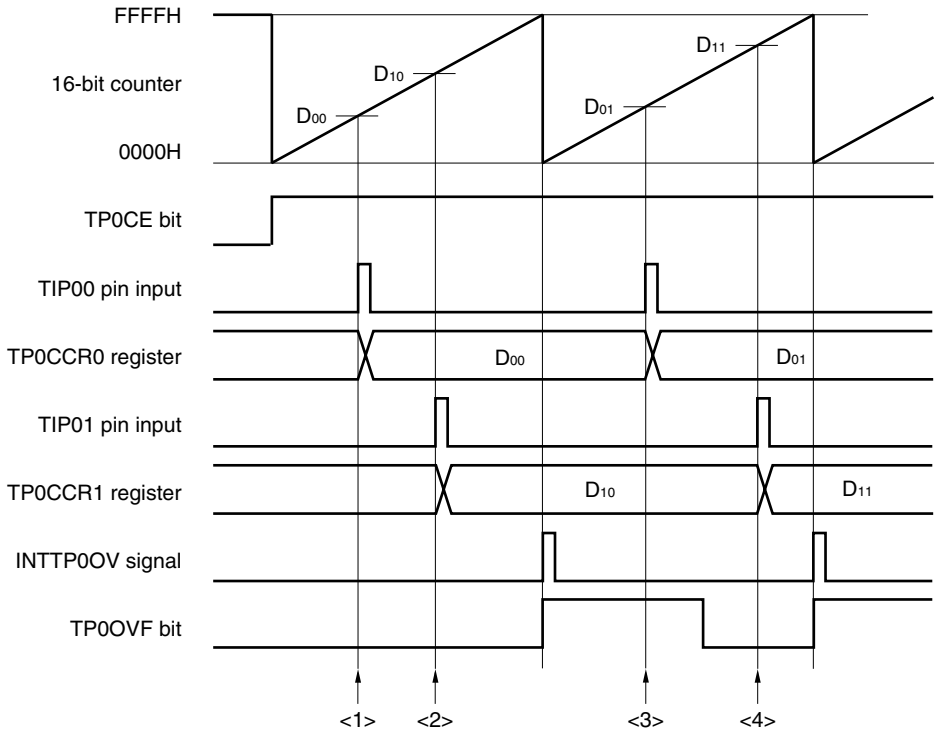
Remark a = 0, 1

Downloaded from Arrow.com.

(c) Processing of overflow when two capture registers are used

Care must be exercised in processing the overflow flag when two capture registers are used. First, an example of incorrect processing is shown below.

Example of incorrect processing when two capture registers are used



The following problem may occur when two pulse widths are measured in the free-running timer mode.

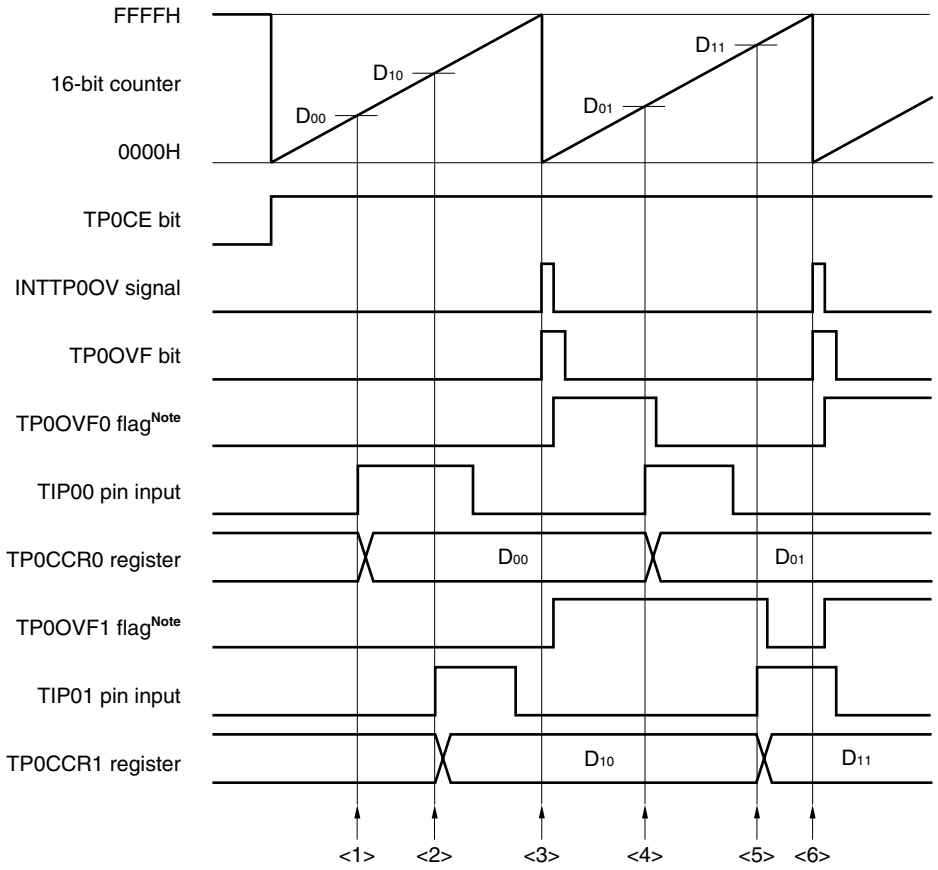
<1> Read the TP0CCR0 register (setting of the default value of the TIP00 pin input).
<2> Read the TP0CCR1 register (setting of the default value of the TIP01 pin input).
<3> Read the TP0CCR0 register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{01} − D_{00}).
<4> Read the TP0CCR1 register.
Read the overflow flag. Because the flag is cleared in <3>, 0 is read.
Because the overflow flag is 0, the pulse width can be calculated by (D_{11} − D_{10}) (incorrect).

When two capture registers are used, and if the overflow flag is cleared to 0 by one capture register, the other capture register may not obtain the correct pulse width.
Use software when using two capture registers. An example of how to use software is shown below.

Downloaded from Arrow.com.

(1/2)

Example when two capture registers are used (using overflow interrupt)



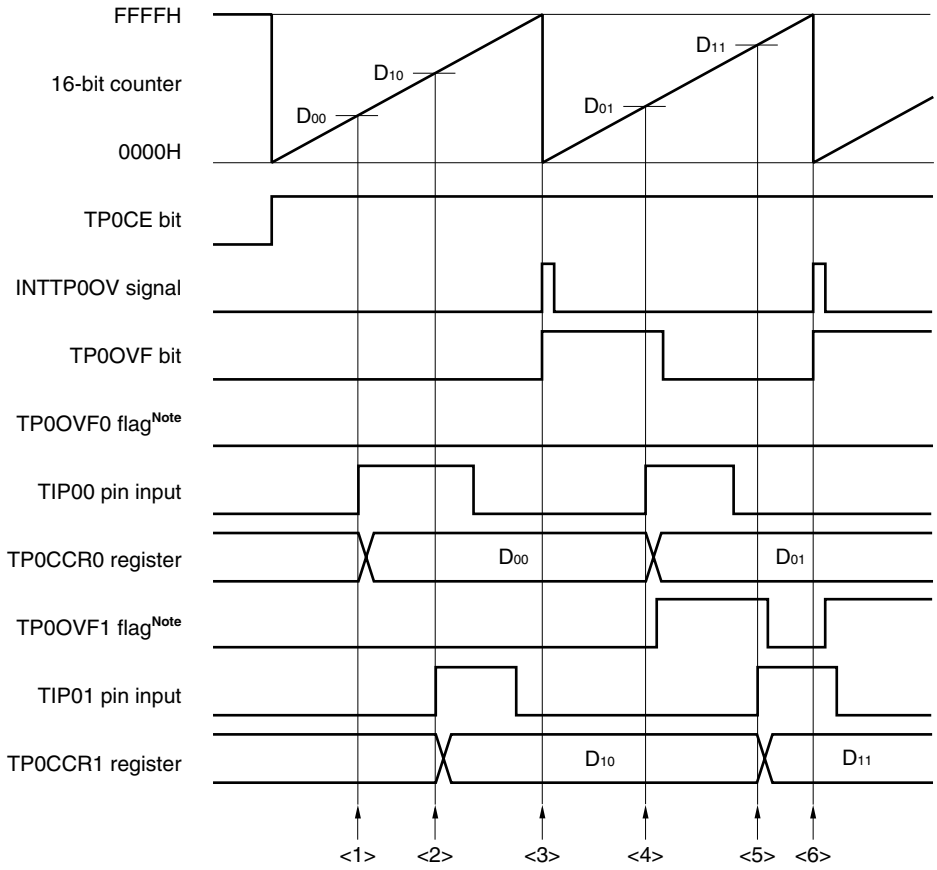
Note The TP0OVF0 and TP0OVF1 flags are set on the internal RAM by software.

<1> Read the TP0CCR0 register (setting of the default value of the TIP00 pin input).
<2> Read the TP0CCR1 register (setting of the default value of the TIP01 pin input).
<3> An overflow occurs. Set the TP0OVF0 and TP0OVF1 flags to 1 in the overflow interrupt servicing, and clear the overflow flag to 0.
<4> Read the TP0CCR0 register.
Read the TP0OVF0 flag. If the TP0OVF0 flag is 1, clear it to 0.
Because the TP0OVF0 flag is 1, the pulse width can be calculated by (10000H + D_{01} – D_{00}).
<5> Read the TP0CCR1 register.
Read the TP0OVF1 flag. If the TP0OVF1 flag is 1, clear it to 0 (the TP0OVF0 flag is cleared in <4>, and the TP0OVF1 flag remains 1).
Because the TP0OVF1 flag is 1, the pulse width can be calculated by (10000H + D_{11} – D_{10}) (correct).
<6> Same as <3>

Downloaded from Arrow.com.

(2/2)

Example when two capture registers are used (without using overflow interrupt)



Note The TP0OVF0 and TP0OVF1 flags are set on the internal RAM by software.

<1> Read the TP0CCR0 register (setting of the default value of the TIP00 pin input).

<2> Read the TP0CCR1 register (setting of the default value of the TIP01 pin input).

<3> An overflow occurs. Nothing is done by software.

<4> Read the TP0CCR0 register.

Read the overflow flag. If the overflow flag is 1, set only the TP0OVF1 flag to 1, and clear the overflow flag to 0.

Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{01} − D_{00}).

<5> Read the TP0CCR1 register.

Read the overflow flag. Because the overflow flag is cleared in <4>, 0 is read.

Read the TP0OVF1 flag. If the TP0OVF1 flag is 1, clear it to 0.

Because the TP0OVF1 flag is 1, the pulse width can be calculated by (10000H + D_{11} − D_{10}) (correct).

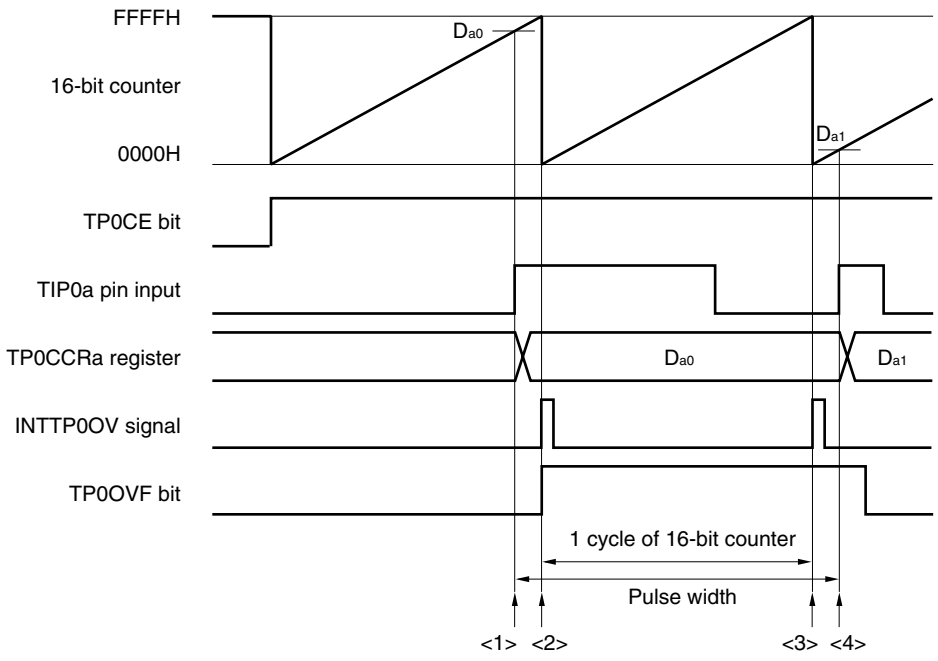
<6> Same as <3>

Downloaded from Arrow.com.

(d) Processing of overflow if capture trigger interval is long

If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next. First, an example of incorrect processing is shown below.

Example of incorrect processing when capture trigger interval is long



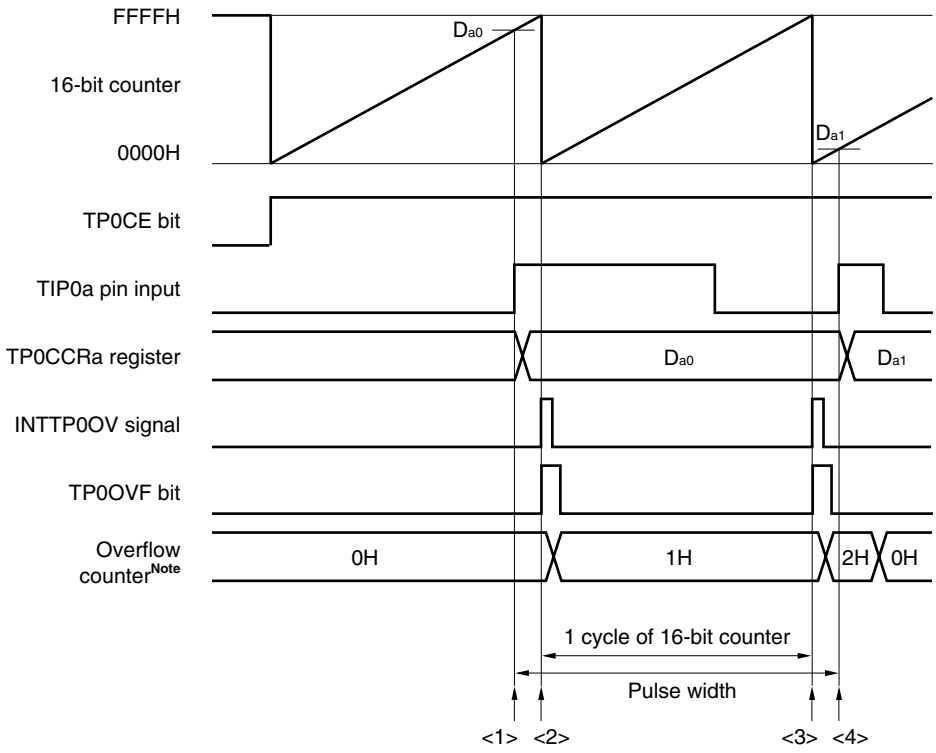
The following problem may occur when long pulse width is measured in the free-running timer mode.

<1> Read the TP0CCRa register (setting of the default value of the TIP0a pin input).
<2> An overflow occurs. Nothing is done by software.
<3> An overflow occurs a second time. Nothing is done by software.
<4> Read the TP0CCRa register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{a1} − D_{a0}) (incorrect).
Actually, the pulse width must be (20000H + D_{a1} − D_{a0}) because an overflow occurs twice.

If an overflow occurs twice or more when the capture trigger interval is long, the correct pulse width may not be obtained.
If the capture trigger interval is long, slow the count clock to lengthen one cycle of the 16-bit counter, or use software. An example of how to use software is shown next.

Downloaded from Arrow.com.

Example when capture trigger interval is long



Note The overflow counter is set arbitrarily by software on the internal RAM.

<1> Read the TP0CCRa register (setting of the default value of the TIP0a pin input).

<2> An overflow occurs. Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<3> An overflow occurs a second time. Increment (+1) the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<4> Read the TP0CCRa register.

Read the overflow counter.

→ When the overflow counter is “N”, the pulse width can be calculated by ($N \times 10000H + D_{a1} - D_{a0}$).

In this example, the pulse width is ($20000H + D_{a1} - D_{a0}$) because an overflow occurs twice.

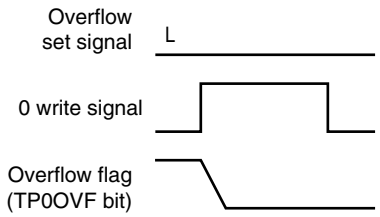
Clear the overflow counter (0H).

Downloaded from Arrow.com.

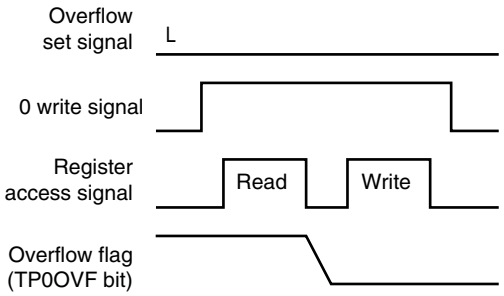
(e) Clearing overflow flag

The overflow flag can be cleared to 0 by clearing the TP0OVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TP0OPT0 register. To accurately detect an overflow, read the TP0OVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.

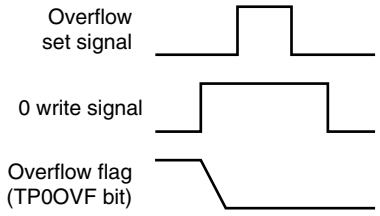
(i) Operation to write 0 (without conflict with setting)



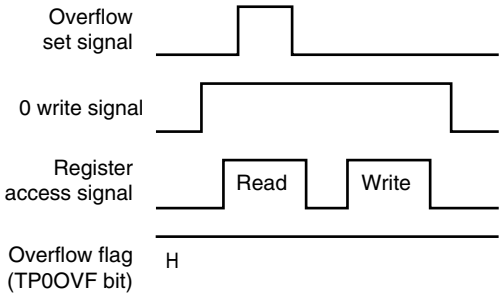
(iii) Operation to clear to 0 (without conflict with setting)



(ii) Operation to write 0 (conflict with setting)



(iv) Operation to clear to 0 (conflict with setting)



To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction. If 0 is written to the overflow flag without checking if the flag is 1, the set information of overflow may be erased by writing 0 ((ii) in the above chart). Therefore, software may judge that no overflow has occurred even when an overflow actually has occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set even after execution of the clear instruction.

Downloaded from Arrow.com.

6.5.7 Pulse width measurement mode (TP0MD2 to TP0MD0 bits = 110)

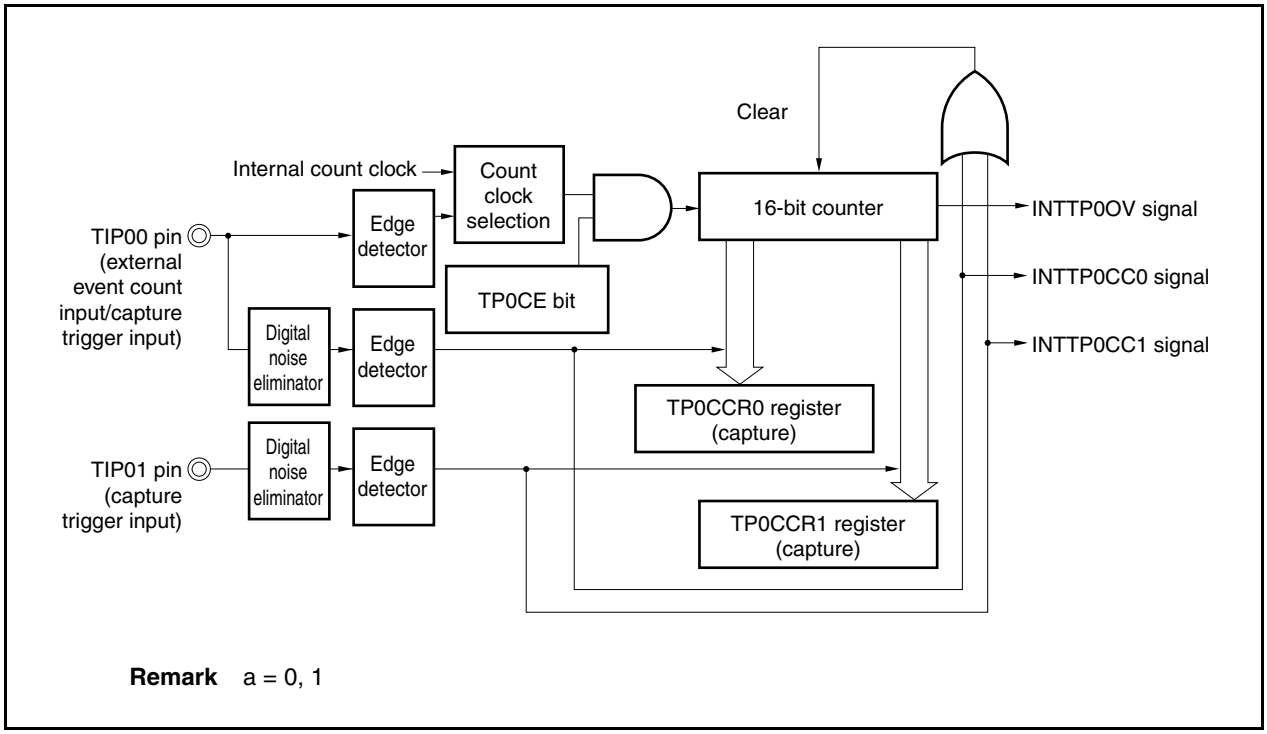
In the pulse width measurement mode, 16-bit timer/event counter P starts counting when the TP0CTL0.TP0CE bit is set to 1. Each time the valid edge input to the TIP0a pin has been detected, the count value of the 16-bit counter is stored in the TP0CCRa register, and the 16-bit counter is cleared to 0000H.

The interval of the valid edge can be measured by reading the TP0CCRa register after a capture interrupt request signal (INTTP0CCa) occurs.

Select either the TIP00 or TIP01 pin as the capture trigger input pin. Specify “No edge detected” by using the TP0IOC1 register for the unused pins.

When an external clock is used as the count clock, measure the pulse width of the TIP01 pin because the external clock is fixed to the TIP00 pin. At this time, clear the TP0IOC1.TP0IS1 and TP0IOC1.TP0IS0 bits to 00 (capture trigger input (TIP00 pin): No edge detected).

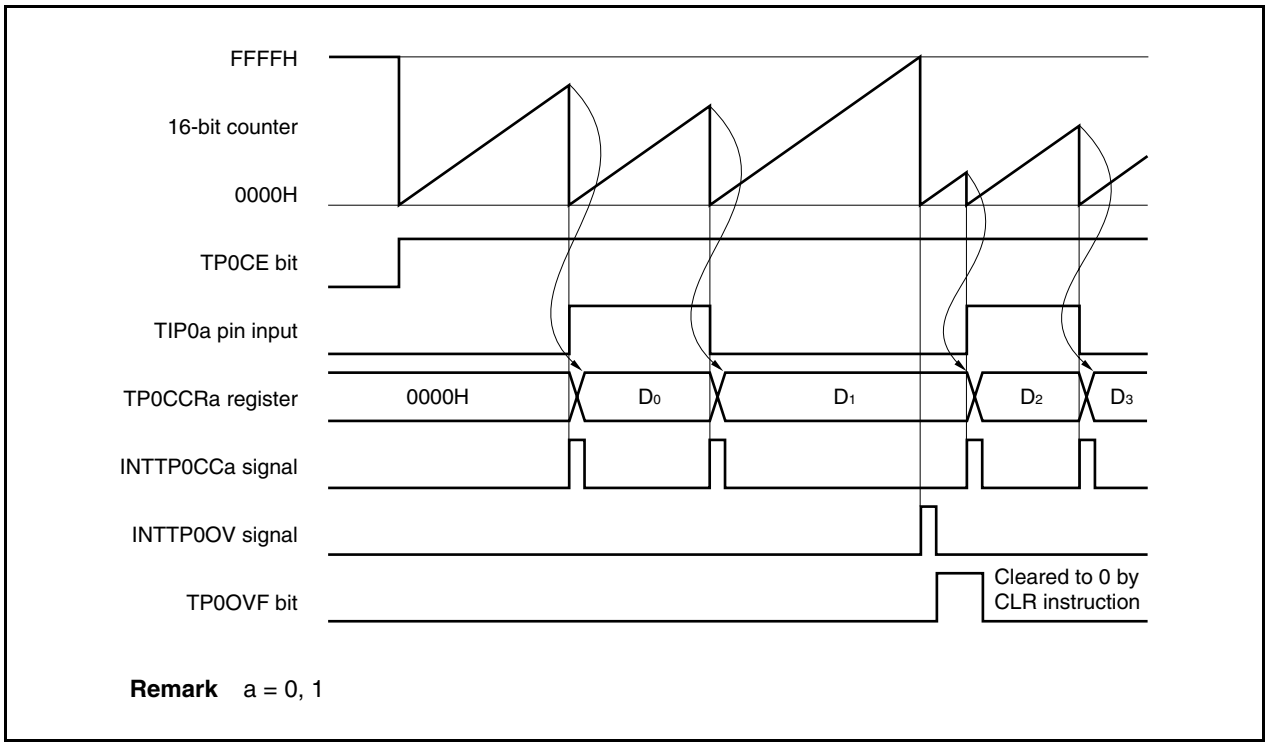
Figure 6-34. Configuration in Pulse Width Measurement Mode



Remark a = 0, 1

Downloaded from Arrow.com.

Figure 6-35. Basic Timing in Pulse Width Measurement Mode



Remark a = 0, 1

When the TP0CE bit is set to 1, the 16-bit counter starts counting. When the valid edge input to the TIP0a pin is later detected, the count value of the 16-bit counter is stored in the TP0CCRa register, the 16-bit counter is cleared to 0000H, and a capture interrupt request signal (INTTP0CCa) is generated.

The pulse width is calculated as follows.

<R> Pulse width = Captured value × Count clock cycle

If the valid edge is not input to the TIP0a pin even when the 16-bit counter counted up to FFFFH, an overflow interrupt request signal (INTTP0OV) is generated at the next count clock, and the counter is cleared to 0000H and continues counting. At this time, the overflow flag (TP0OPT0.TP0OVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction via software.

If the overflow flag is set to 1, the pulse width can be calculated as follows.

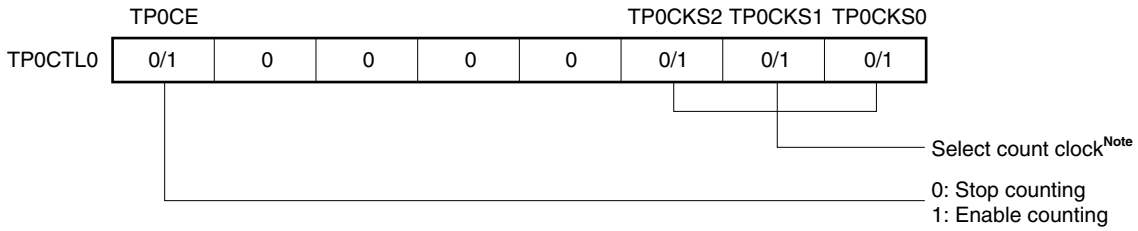
<R> Pulse width = (10000H × TP0OVF bit set (1) count + Captured value) × Count clock cycle

Remark a = 0, 1

Downloaded from Arrow.com.

Figure 6-36. Register Setting in Pulse Width Measurement Mode (1/2)

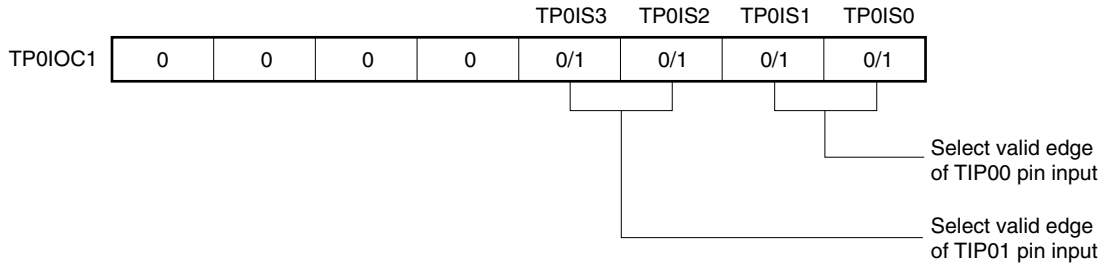
(a) TMP0 control register 0 (TP0CTL0)



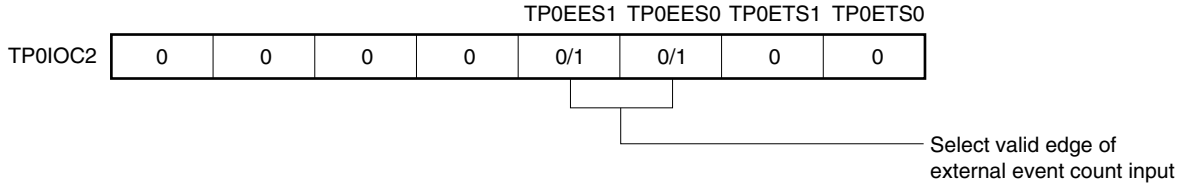
Note Setting is invalid when the TP0EEE bit = 1.

(b) TMP0 control register 1 (TP0CTL1)

(c) TMP0 I/O control register 1 (TP0IOC1)



(d) TMP0 I/O control register 2 (TP0IOC2)



Downloaded from Arrow.com.

Figure 6-36. Register Setting in Pulse Width Measurement Mode (2/2)

(e) TMP0 option register 0 (TP0OPT0)

				TP0CCS1	TP0CCS0			TP0OVF
TP0OPT0	0	0	0	0	0	0	0	0/1

Overflow flag

(f) TMP0 counter read buffer register (TP0CNT)

The value of the 16-bit counter can be read by reading the TP0CNT register.

(g) TMP0 capture/compare registers 0 and 1 (TP0CCR0 and TP0CCR1)

These registers store the count value of the 16-bit counter when the valid edge input to the TIP0a pin is detected.

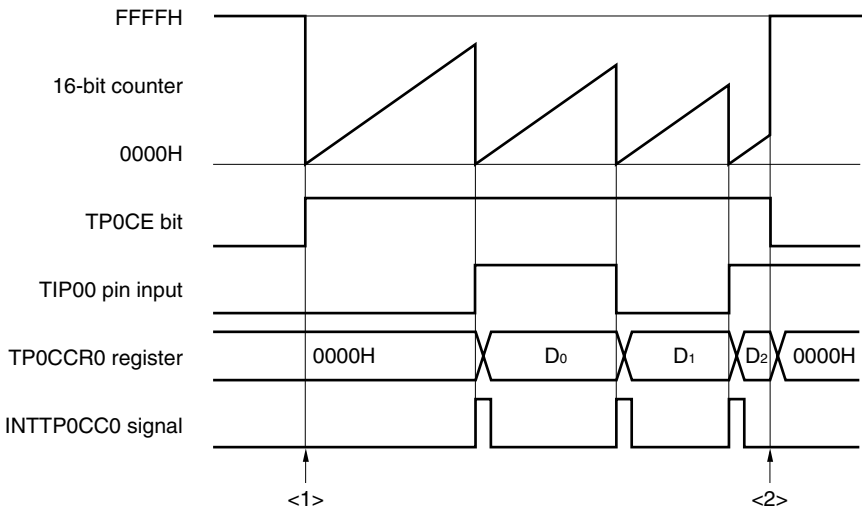
Remarks 1. TMP0 I/O control register 0 (TP0IOC0) is not used in the pulse width measurement mode.

2. a = 0, 1

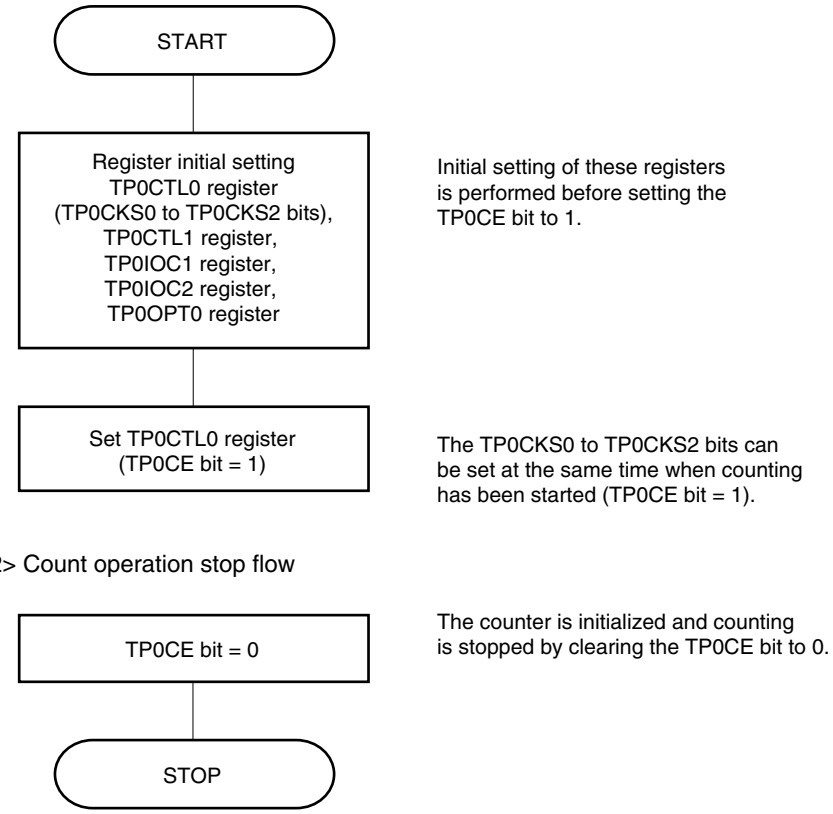
Downloaded from Arrow.com.

(1) Operation flow in pulse width measurement mode

Figure 6-37. Software Processing Flow in Pulse Width Measurement Mode



<1> Count operation start flow



<2> Count operation stop flow

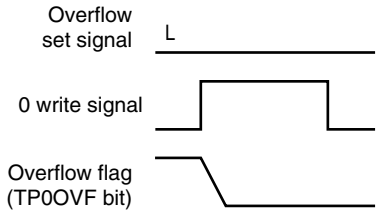
Downloaded from Arrow.com.

(2) Operation timing in pulse width measurement mode

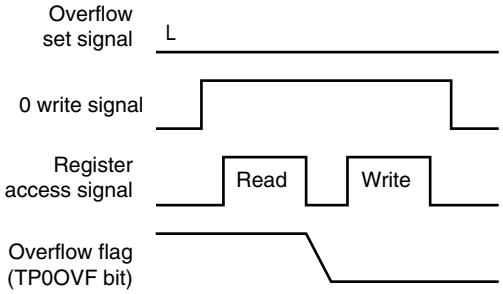
(a) Clearing overflow flag

The overflow flag can be cleared to 0 by clearing the TP0OVF bit to 0 with the CLR instruction and by writing 8-bit data (bit 0 is 0) to the TP0OPT0 register. To accurately detect an overflow, read the TP0OVF bit when it is 1, and then clear the overflow flag by using a bit manipulation instruction.

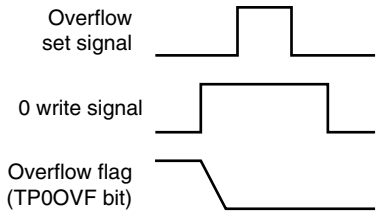
(i) Operation to write 0 (without conflict with setting)



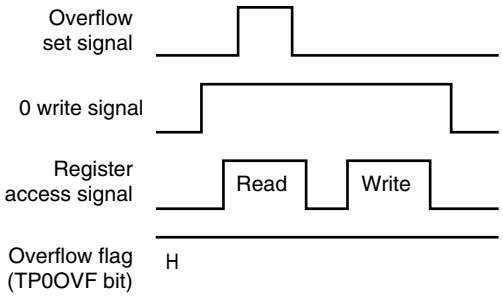
(iii) Operation to clear to 0 (without conflict with setting)



(ii) Operation to write 0 (conflict with setting)



(iv) Operation to clear to 0 (conflict with setting)



To clear the overflow flag to 0, read the overflow flag to check if it is set to 1, and clear it with the CLR instruction. If 0 is written to the overflow flag without checking if the flag is 1, the set information of overflow may be erased by writing 0 ((ii) in the above chart). Therefore, software may judge that no overflow has occurred even when an overflow actually has occurred.

If execution of the CLR instruction conflicts with occurrence of an overflow when the overflow flag is cleared to 0 with the CLR instruction, the overflow flag remains set even after execution of the clear instruction.

Downloaded from Arrow.com.

6.5.8 Timer output operations

The following table shows the operations and output levels of the TOP00 and TOP01 pins.

Table 6-4. Timer Output Control in Each Mode

Operation Mode	TOP01 Pin	TOP00 Pin
Interval timer mode	Square wave output	
External event count mode	Square wave output	–
External trigger pulse output mode	External trigger pulse output	Square wave output
One-shot pulse output mode	One-shot pulse output	
PWM output mode	PWM output	
Free-running timer mode	Square wave output (only when compare function is used)	
Pulse width measurement mode	–	

Table 6-5. Truth Table of TOP00 and TOP01 Pins Under Control of Timer Output Control Bits

TP0IOC0.TP0OLa Bit	TP0IOC0.TP0OEa Bit	TP0CTL0.TP0CE Bit	Level of TOP0a Pin
0	0	×	Low-level output
	1	0	Low-level output
		1	Low level immediately before counting, high level after counting is started
1	0	×	High-level output
	1	0	High-level output
		1	High level immediately before counting, low level after counting is started

Remark a = 0, 1

Downloaded from Arrow.com.

6.6 Eliminating Noise on Capture Trigger Input Pin (TIP0a)

The TIP0a pin has a digital noise eliminator.

However, this circuit is valid only when the pin is used as a capture trigger input pin; it is invalid when the pin is used as an external event count input pin or external trigger input pin.

Digital noise can be eliminated by specifying the alternate function of the TIP0a pin using the PMC3, PFC3, and PFCE3 registers.

The number of times of sampling can be selected from three or two by using the PaNFC.PaNFSTS bit. The sampling clock can be selected from fxx, fxx/2, fxx/4, fxx/16, fxx/32, or fxx/64, by using the PaNFC.PaNFC2 to PaNFC.PaNFC0 bits.

(1) TIP0a noise elimination control register (PaNFC)

This register is used to select the sampling clock and the number of times of sampling for eliminating digital noise.

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

Reset input clears this register to 00H.

After reset: 00H R/W Address: P0NFC FFFFFB00H, P1NFC FFFFFB04H

	7	6	5	4	3	2	1	0
PaNFC (a = 0, 1)	0	PaNFSTS	0	0	0	PaNFC2	PaNFC1	PaNFC0

PaNFSTS	Setting of number of times of sampling for eliminating digital noise
0	Number of times of sampling = 3
1	Number of times of sampling = 2

PaNFC2	PaNFC1	PaNFC0	Sampling clock selection
0	0	0	fxx
0	0	1	fxx/2
0	1	0	fxx/4
0	1	1	fxx/16
1	0	0	fxx/32
1	0	1	fxx/64
Other than above			Setting prohibited

Cautions 1. Enable starting the 16-bit counter of TMP0 (TP0CTL.TP0CE bit = 1) after the lapse of the sampling clock period × number of times of sampling.

2. Be sure to clear bits 7, 5 to 3 to "0".

Downloaded from Arrow.com.

<Setting procedure>

<1> Select the number of times of sampling and the sampling clock by using the PaNFC register.
<2> Select the alternate function (of the TIP0a pin) by using the PMC3, PFC3, and PFCE3 registers.
<3> Set the operating mode of TMP0 (such as the capture mode or the valid edge of the capture trigger).
<4> Enable the TMP0 count operation.

<Noise elimination width>

The digital noise elimination width (t_{WTIPa}) is as follows, where T is the sampling clock period and M is the number of times of sampling.

- $t_{WTIPa} < (M - 1)T$: Accurately eliminated as noise
- $(M - 1)T \leq t_{WTIPa} < MT$: Eliminated as noise or detected as valid edge
- $t_{WTIPa} \geq MT$: Accurately detected as valid edge

Therefore, a pulse width of MT or longer must be input so that the valid edge of the capture trigger input can be accurately detected.

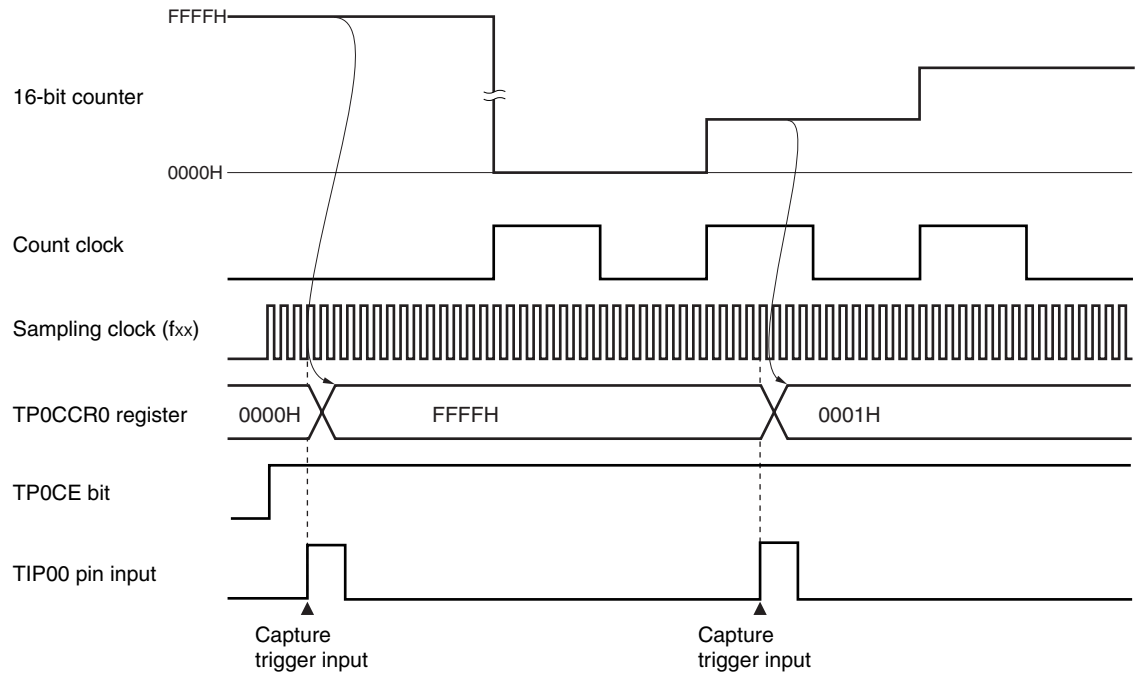
Downloaded from Arrow.com.

6.7 Cautions

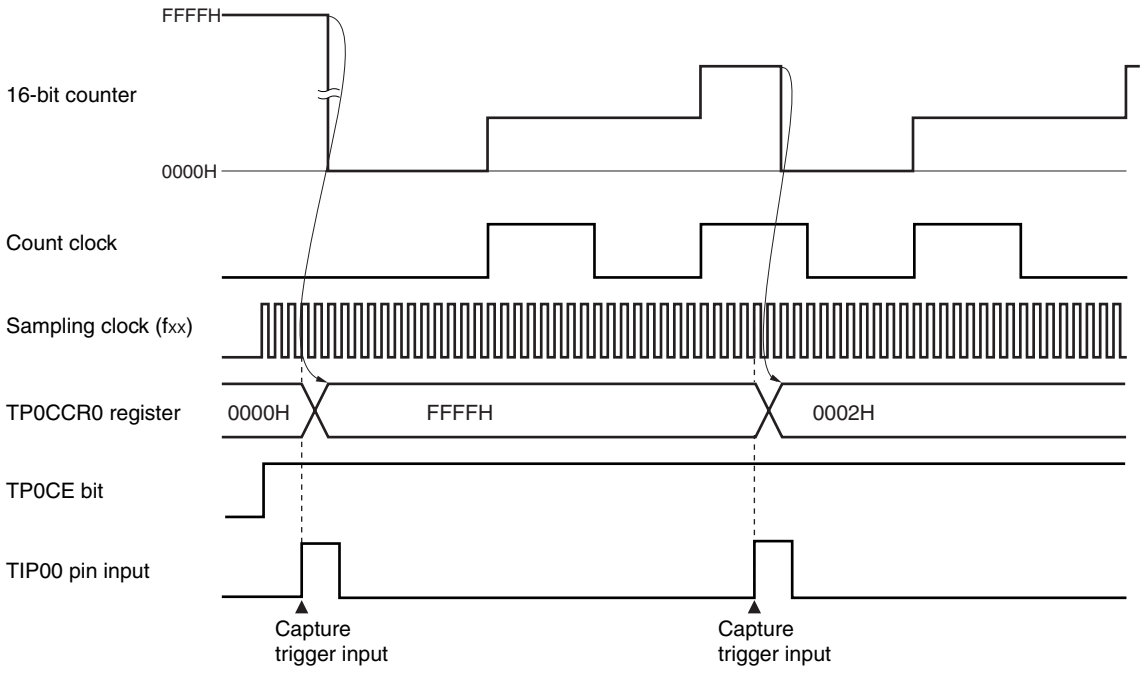
(1) Capture operation

When the capture operation is used and fxx/8, fxx/16, fxx/32, fxx/64, fxx/128, or the external event count (TP0CLT1.TP0EEE bit = 1) is selected as the count clock, FFFFH, not 0000H, may be captured in the TP0CCRn register if the capture trigger is input immediately after the TP0CE bit is set to 1.

(a) Free-running timer mode



(b) Pulse width measurement mode



Downloaded from Arrow.com.

<R>

CHAPTER 7 16-BIT TIMER/EVENT COUNTER 0

In the V850ES/KE1, one channel of 16-bit timer/event counter 0 is provided.

7.1 Functions

16-bit timer/event counter 01 has the following functions.

(1) Interval timer

16-bit timer/event counter 01 generates an interrupt request at the preset time interval.

(2) Square-wave output

16-bit timer/event counter 01 can output a square wave with any selected frequency.

(3) External event counter

16-bit timer/event counter 01 can measure the number of pulses of an externally input signal.

(4) One-shot pulse output (16-bit timer/event counters 00 and 01 only)

16-bit timer/event counter 01 can output a one-shot pulse whose output pulse width can be set freely.

(5) PPG output

16-bit timer/event counter 01 can output a rectangular wave whose frequency and output pulse width can be set freely.

(6) Pulse width measurement

16-bit timer/event counter 01 can measure the pulse width of an externally input signal.

Downloaded from Arrow.com.

7.2 Configuration

16-bit timer/event counter 01 includes the following hardware.

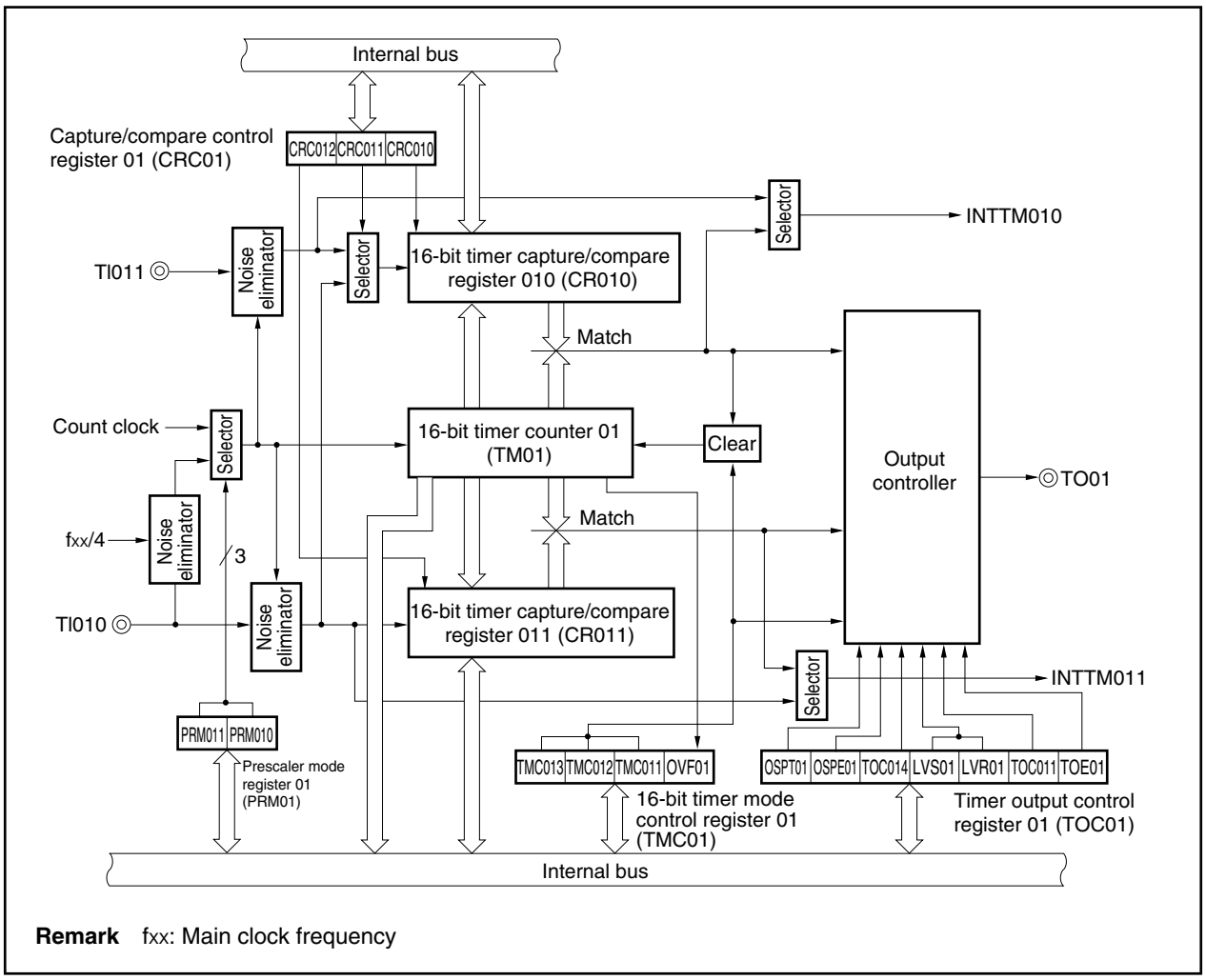
Table 7-1. Configuration of 16-bit Timer/Event Counter 01

Item	Configuration
Timer/counter	16-bit timer counter 01 × 1 (TM01)
Register	16-bit timer capture/compare registers: 16-bit × 2 (CR010, CR011)
Timer input	2 (TI010, TI011 pins)
Timer output	1 (TO01 pin), output controller
Control registers[Note]	16-bit timer mode control register 01 (TMC01) Capture/compare control register 01 (CRC01) 16-bit timer output control register 01 (TOC01) Prescaler mode register 01 (PRM01)

Note To use the TI010, TI011, and TO01 pin functions, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

The block diagram is shown below.

Figure 7-1. Block Diagram of 16-bit Timer/Event Counter 01



Remark fxx: Main clock frequency

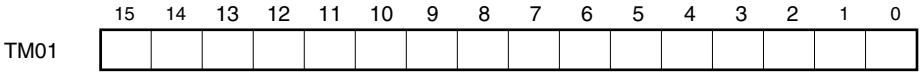
Downloaded from Arrow.com.

(1) 16-bit timer counter 01 (TM01)

The TM01 register is a 16-bit read-only register that counts count pulses.

The counter is incremented in synchronization with the rising edge of the count clock.

After reset: 0000H R Address: FFFFF610H



The count value of the TM01 register can be read by reading the TM01 register when the values of the TMC01.TMC013 and TMC01.TMC012 bits are other than 00. The value of the TM01 register is 0000H if it is read when the TMC013 and TMC012 bits are 00.

The count value is reset to 0000H in the following cases.

- At reset signal generation
- If the TMC013 and TMC012 bits are cleared to 00
- If the valid edge is input to the TI010 pin in the mode in which the clear & start occurs when inputting the valid edge to the TI010 pin
- If the TM01 register and the CR010 register match in the mode in which the clear & start occurs when the TM01 register and the CR010 register match
- If the TOC01.OSPT01 bit is set to 1 in one-shot pulse output mode or if a valid edge is input to the TI010 pin.

Downloaded from Arrow.com.

(2) 16-bit timer capture/compare register 010 (CR010), 16-bit timer capture/compare register 011 (CR011)

The CR010 and CR011 registers are 16-bit registers that are used with a capture function or comparison function selected by using the CRC01 register.

Change of the value of the CR010 register while the timer is operating (TMC01.TMC013 and TMC01.TMC012 bits = other than 00) is prohibited.

The value of the CR011 register can be changed during operation if the value has been set in a specific way. For details, see **7.5.1 Rewriting CR011 register during TM01 operation**.

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

Reset sets these registers to 0000H.

(a) 16-bit timer capture/compare register 010 (CR010)

After reset: 0000H R/W Address: FFFFF612H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR010																

(i) When the CR010 register is used as a compare register

The value set in the CR010 register is constantly compared with the TM01 register count value, and an interrupt request signal (INTTM010) is generated if they match. The value is held until the CR010 register is rewritten.

(ii) When the CR010 register is used as a capture register

The count value of the TM01 register is captured to the CR010 register when a capture trigger is input.

As the capture trigger, an edge of a phase reverse to that of the TI010 pin or the valid edge of the TI011 pin can be selected by using the CRC01 or PRM01 register.

Downloaded from Arrow.com.

(b) 16-bit timer capture/compare register 011 (CR011)

After reset: 0000H R/W Address: FFFFF614H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR011																

(i) When using the CR011 register as a compare register

The value set to the CR011 register and the count value of the TM01 register are always compared and when these values match, an interrupt request signal (INTTM011) is generated.

(ii) When using the CR011 register as a capture register

The TM01 register count value is captured to the CR011 register by inputting a capture trigger.

The valid edge of the TI010 pin can be selected as the capture trigger. The valid edge of the TI010 pin is set with the PRM01 register.

Cautions 1. When the P35 pin is used as the valid edge of TI010, it cannot be used as timer output (TO01). Moreover, when the P35 pin is used as TO01, it cannot be used as the valid edge of TI010.

2. If clearing of the TMC013 and TMC012 bits to 00 and input of the capture trigger conflict, then the captured data is undefined.

3. To change the mode from the capture mode to the comparison mode, first clear the TMC013 and TMC012 bits to 00, and then change the setting.

A value that has been once captured remains stored in the CR010 and CR011 registers unless the device is reset. If the mode has been changed to the comparison mode, be sure to set a comparison value.

Downloaded from Arrow.com.

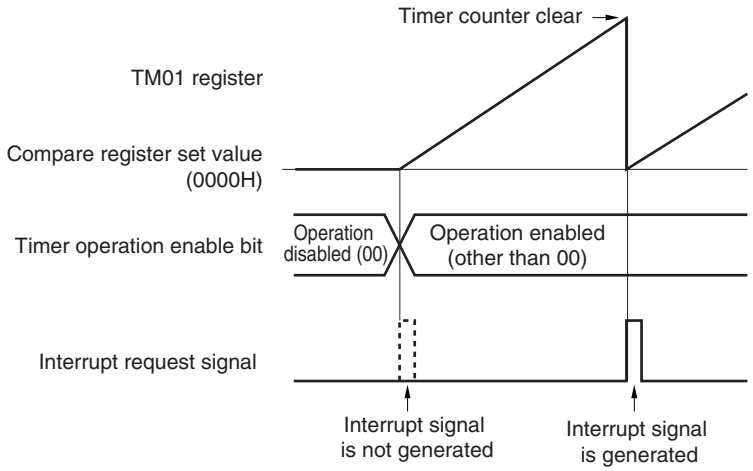
(c) Setting range when used as compare register

When the CR010 or CR011 register is used as a compare register, set it as shown below.

Operation	CR010 Register	CR011 Register
• Operation as interval timer • Operation as square-wave output • Operation as external event counter	0000H < N ≤ FFFFH	0000H[Note] ≤ M ≤ FFFFH Normally, this setting is not used. Mask the match interrupt signal (INTTM011).
• Operation in the clear & start mode entered by TI010 pin valid edge input • Operation as free-running timer	0000H[Note] ≤ N ≤ FFFFH	0000H[Note] ≤ M ≤ FFFFH
• Operation as PPG output	M < N ≤ FFFFH	0000H[Note] ≤ M < N
• Operation as one-shot pulse output	0000H[Note] ≤ N ≤ FFFFH (N ≠ M)	0000H[Note] ≤ M ≤ FFFFH (M ≠ N)

Note When 0000H is set, a match interrupt immediately after the timer operation does not occur and timer output is not changed, and the first match timing is as follows. A match interrupt occurs at the timing when the timer counter (TM01 register) is changed from 0000H to 0001H.

- When the timer counter is cleared due to overflow
- When the timer counter is cleared due to TI010 pin valid edge (when clear & start mode is entered by TI010 pin valid edge input)
- When the timer counter is cleared due to compare match (when clear & start mode is entered by match between TM01 and CR010 (CR010 = other than 0000H, CR011 = 0000H))

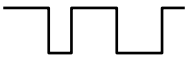
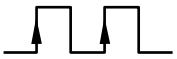
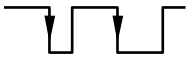
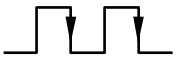
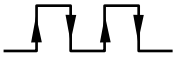
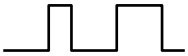
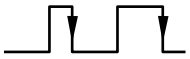


Remarks 1. N: CR010 register set value
M: CR011 register set value

2. For details of operation enable bits (TMC01.TMC013, TMC01.TMC012 bits), refer to **7.3 (1) 16-bit timer mode control register 01 (TMC01)**.

Downloaded from Arrow.com.

Table 7-2. Capture Operation of CR010 and CR011 Registers

External Input Signal / Capture Operation	TI010 Pin Input		TI011 Pin Input	
Capture operation of CR010 register	CRC011 bit = 1 TI010 pin input (reverse phase) 	Set values of ES101 and ES100 Position of edge to be captured	CRC011 bit = 0 TI011 pin input 	Set values of ES111 and ES110 Position of edge to be captured
		01: Rising		01: Rising 
		00: Falling 		00: Falling 
		11: Both edges (cannot be captured)		11: Both edges 
	Interrupt signal	INTTM010 signal is not generated even if value is captured.	Interrupt signal	INTTM010 signal is generated each time value is captured.
Capture operation of CR011 register	TI010 pin input[Note] 	Set values of ES101 and ES100 Position of edge to be captured		
		01: Rising		
		00: Falling 		
		11: Both edges		
	Interrupt signal	INTTM011 signal is generated each time value is captured.		

Note The capture operation of the CR011 register is not affected by the setting of the CRC011 bit.

Caution To capture the count value of the TM01 register to the CR010 register by using the phase reverse to that input to the TI010 pin, the interrupt request signal (INTTM010) is not generated after the value has been captured. If the valid edge is detected on the TI011 pin during this operation, the capture operation is not performed but the INTTM010 signal is generated as an external interrupt signal. To not use the external interrupt, mask the INTTM010 signal.

Remark CRC011: See **7.3 (2) Capture/compare control register 01 (CRC01)**.
ES111, ES110, ES101, ES100: See **7.3 (4) Prescaler mode register 01 (PRM01)**.

Downloaded from Arrow.com.

7.3 Registers

Registers used to control 16-bit timer/event counter 01 are shown below.

- 16-bit timer mode control register 01 (TMC01)
- Capture/compare control register 01 (CRC01)
- 16-bit timer output control register 01 (TOC01)
- Prescaler mode register 01 (PRM01)

Remark To use the TI010, TI011, and TO01 pin functions, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

(1) 16-bit timer mode control register 01 (TMC01)

TMC01 is an 8-bit register that sets the 16-bit timer/event counter 01 operation mode, the TM01 register clear mode, and output timing, and detects an overflow.

Rewriting TMC01 is prohibited during operation (when the TMC013 and TMC012 bits = other than 00). However, it can be changed when the TMC013 and TMC012 bits are cleared to 00 (stopping operation) and when the OVF01 bit is cleared to 0.

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

Reset sets this register to 00H.

Cautions 1. 16-bit timer/event counter 01 starts operation at the moment the TMC013 and TMC012 bits are set to values other than 00 (operation stop mode), respectively. Set the TMC013 and TMC012 bits to 00 to stop the operation.

2. When the main clock is stopped and the CPU is operating on the subclock, do not access the TMC01 register using an access method that causes a wait.
For details, refer to 3.4.8 (1) (b).

3. Be sure to clear bits 7 to 4 and 1 to “0”.

Downloaded from Arrow.com.

After reset: 00H R/W Address: FFFFF616H

	7	6	5	4	3	2	1	<0>
TMC01	0	0	0	0	TMC013	TMC012	0	OVF01

TMC013	TMC012	Enable operation of 16-bit timer/event counter 01
0	0	Disables TM01 operation. Stops supplying operating clock. Clears 16-bit timer counter (TM01).
0	1	Free-running timer mode
1	0	Clear & start mode entered by TI010 pin valid edge input[Note]
1	1	Clear & start mode entered upon a match between TM01 and CR010

OVF01	TM01 register overflow flag
Clear (0)	Clears OVF01 to 0 or TMC01.TMC013 and TMC01.TMC012 = 00
Set (1)	Overflow occurs.
OVF01 is set to 1 when the value of TM01 changes from FFFFH to 0000H in all the operation modes (free-running timer mode, clear & start mode entered by TI010 pin valid edge input, and clear & start mode entered upon a match between TM01 and CR010). It can also be set to 1 by writing 1 to the OVF01 bit.	

Note The TI010 pin valid edge is set by the PRM01 register.

Caution Be sure to clear bits 7 to 4 and 1 to "0".

Downloaded from Arrow.com.

(2) Capture/compare control register 01 (CRC01)

The CRC01 register is the register that controls the operation of the CR010 and CR011 registers.

Changing the value of the CRC01 register is prohibited during operation (when the TMC01.TMC013 and TMC01.TMC012 bits = other than 00).

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF618H

	7	6	5	4	3	2	1	0
CRC01	0	0	0	0	0	CRC012	CRC011	CRC010

CRC012	CR011 register operating mode selection
0	Operates as compare register
1	Operates as capture register

CRC011	CR010 register capture trigger selection
0	Captures on valid edge of TI011 pin
1	Captures on valid edge of TI010 pin by reverse phase[Note]
The valid edge of the TI011 and TI010 pin is set by the PRM01 register. If PRM01.ES101 and PRM01.ES100 are set to 11 (both edges) when CRC011 is 1, the valid edge of the TI010 pin cannot be detected.	

CRC010	CR010 register operating mode selection
0	Operates as compare register
1	Operates as capture register
If TMC013 and TMC012 are set to 11 (clear & start mode entered upon a match between TM01 and CR010), be sure to set the CRC010 bit to 0.	

Note When the valid edge is detected from the TI011 pin, the capture operation is not performed but the INTTM010 signal is generated as an external interrupt signal.

Caution To ensure that the capture operation is performed properly, the capture trigger requires a pulse two cycles longer than the count clock selected by the PRM01 register.

Downloaded from Arrow.com.

(3) 16-bit timer output control register 01 (TOC01)

The TOC01 register is an 8-bit register that controls the TO01 pin output.

The TOC01 register can be rewritten while only the OSPT01 bit is operating (when the TMC01.TMC013 and TMC01.TMC012 bits = other than 00). Rewriting the other bits is prohibited during operation.

However, TOC014 can be rewritten during timer operation as a means to rewrite the CR011 register (see **7.5.1 Rewriting CR011 register during TM01 operation**).

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

Reset sets this register to 00H.

Cautions 1. Be sure to set the TOC01 register using the following procedure.

<1> Set the TOC014 and TOC011 bits to 1.

<2> Set only the TOE01 bit to 1.

<3> Set either the LVS01 bit or LVR01 bit to 1.

2. Restrictions are applied to TM01 in which the timer input and output share the same pin. For details, refer to 7.6 (1) Alternate functions of TI010/TO01 pins.

(1/2)

After reset: 00H R/W Address: FFFFF619H

	7	<6>	<5>	4	<3>	<2>	1	<0>
TOC01	0	OSPT01[Note]	OSPE01[Note]	TOC014	LVS01	LVR01	TOC011	TOE01

OSPT01[Note]	One-shot pulse output trigger via software
0	–
1	One-shot pulse output
The value of this bit is always “0” when it is read. If this bit is set to 1, TM01 is cleared and started.	

OSPE01[Note]	One-shot pulse output operation control
0	Successive pulse output
1	One-shot pulse output
One-shot pulse output operates correctly in the free-running timer mode or clear & start mode entered by TI010 pin valid edge input. The one-shot pulse cannot be output in the clear & start mode entered upon a match between the TM01 and CR010 registers.	

TOC014	TO01 pin output control on match between CR011 and TM01 registers
0	Disables inversion operation
1	Enables inversion operation
The interrupt signal (INTTM011) is generated even when the TOC014 bit = 0.	

Note In 16-bit timer/event counter 0, since the timer input and output share the same pin, only the software trigger is valid as the one-shot pulse output.

Downloaded from Arrow.com.

(2/2)

LVS01	LVR01	Setting of TO01 pin output status
0	0	No change
0	1	Initial value of TO01 pin output is low level (TO01 pin output is cleared to 0).
1	0	Initial value of TO01 pin output is high level (TO01 pin output is set to 1).
1	1	Setting prohibited

- The LVS01 and LVR01 bits can be used to set the initial value of the output level of the TO01 pin. If the initial value does not have to be set, leave the LVS01 and LVR01 bits as 00.
- Be sure to set the LVS01 and LVR01 bits when TOE01 = 1.
  The LVS01, LVR01, and TOE01 bits being simultaneously set to 1 is prohibited.
- The LVS01 and LVR01 bits are trigger bits. By setting these bits to 1, the initial value of the output level of the TO01 pin can be set. Even if these bits are cleared to 0, output of the TO01 pin is not affected.
- The values of the LVS01 and LVR01 bits are always 0 when they are read.
- For how to set the LVS01 and LVR01 bits, see **7.5.2 Setting LVS01 and LVR01 bits**.

TOC011	TO01 pin output control on match between CR010 and TM01 registers
0	Disables inversion operation
1	Enables inversion operation

The interrupt signal (INTTM010) is generated even when the TOC011 bit = 0.

TOE01	TO01 pin output control
0	Disables output (TO01 pin output fixed to low level)
1	Enables output

Downloaded from Arrow.com.

(4) Prescaler mode register 01 (PRM01)

The PRM01 register is the register that sets the TM01 register count clock and TI010 and TI011 pin input valid edges.

Rewriting the PRM01 register is prohibited during operation (when the TMC01.TMC013 and TMC01.TMC012 bits = other than 00).

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

Reset sets this register to 00H.

Cautions 1. Do not apply the following setting when setting the PRM011 and PRM010 bits to 11 (to specify the valid edge of the TI010 pin as a count clock).

- **Clear & start mode entered by the TI010 pin valid edge**
- **Setting the TI010 pin as a capture trigger**

2. If the operation of the 16-bit timer/event counter 01 is enabled when the TI010 or TI011 pin is at high level and when the valid edge of the TI010 or TI011 pin is specified to be the rising edge or both edges, the high level of the TI010 or TI011 pin is detected as a rising edge. Note this when the TI010 or TI011 pin is pulled up. However, the rising edge is not detected when the timer operation has been once stopped and is then enabled again.

3. When the P35 pin is used as the valid edge of TI010, it cannot be used as timer output (TO01). Moreover, when the P35 pin is used as timer output (TO01), it cannot be used as the valid edge of TI010.

Downloaded from Arrow.com.

After reset: 00H R/W Address: FFFFF617H

	7	6	5	4	3	2	1	0
PRM01	ES111	ES110	ES101	ES100	0	0	PRM011	PRM010

ES111	ES110	TI011 pin valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

ES101	ES100	TI010 pin valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

PRM011	PRM010	Count clock selection[Note 1]			
		Count Clock	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 10 MHz
0	0	f_{XX}	Setting prohibited	Setting prohibited	100 ns
0	1	$f_{XX}/4$	200 ns	250 ns	400 ns
1	0	INTWT	–	–	–
1	1	Valid edge of TI010[Note 2]	–	–	–

Notes 1. When the internal clock is selected, set so as to satisfy the following conditions:

V_{DD} = 4.0 to 5.5 V: Count clock ≤ 10 MHz

V_{DD} = 2.7 to 4.0 V: Count clock ≤ 5 MHz

2. The external clock requires a pulse longer than two cycles of the internal clock ($f_{XX}/4$).

Downloaded from Arrow.com.

7.4 Operation

7.4.1 Interval timer operation

If the TMC01.TMC013 and TMC01.TMC012 bits are set to 11 (clear & start mode entered upon a match between the TM01 register and the CR010 register), the count operation is started in synchronization with the count clock.

When the value of the TM01 register later matches the value of the CR010 register, the TM01 register is cleared to 0000H and a match interrupt signal (INTTM010) is generated. This INTTM010 signal enables the TM01 register to operate as an interval timer.

Remarks 1. For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 interrupt, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

Figure 7-2. Block Diagram of Interval Timer Operation

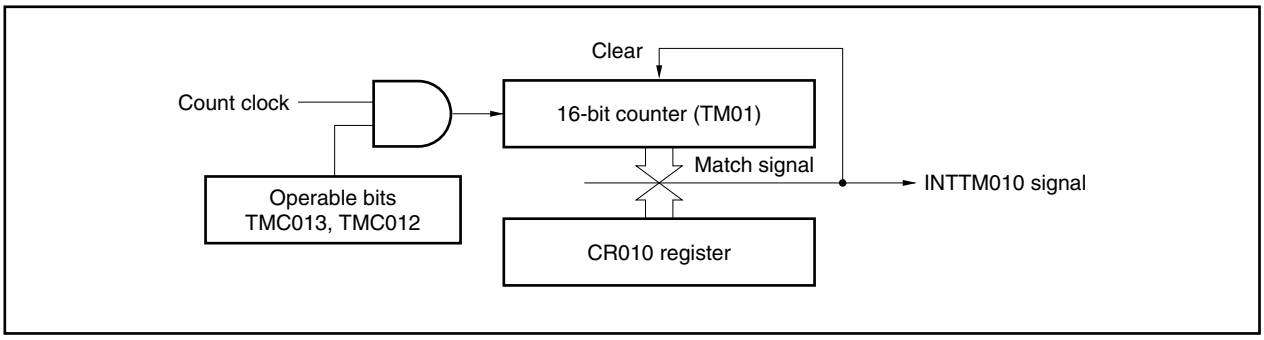
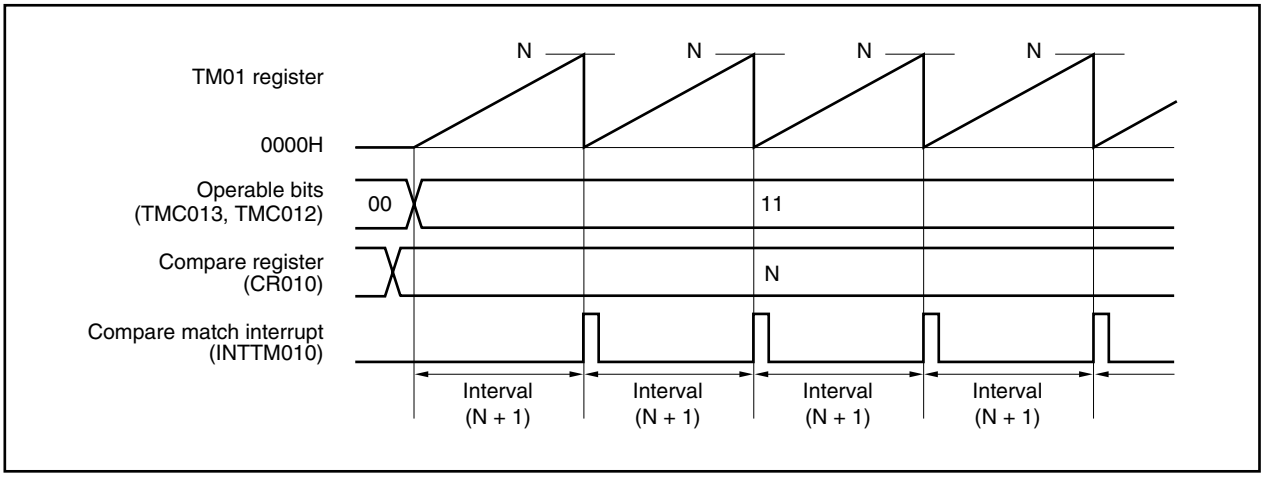


Figure 7-3. Basic Timing Example of Interval Timer Operation



Downloaded from Arrow.com.

Figure 7-4. Example of Register Settings for Interval Timer Operation

(a) 16-bit timer mode control register 01 (TMC01)

				TMC013	TMC012		OVF01
0	0	0	0	1	1	0	0

Clears and starts on match between TM01 and CR010.

(b) Capture/compare control register 01 (CRC01)

					CRC012	CRC011	CRC010
0	0	0	0	0	0	0	0

CR010 used as compare register

(c) 16-bit timer output control register 01 (TOC01)

	OSPT01	OSPE01	TOC014	LVS01	LVR01	TOC011	TOE01
0	0	0	0	0	0	0	0

(d) Prescaler mode register 01 (PRM01)

ES111	ES110	ES101	ES100			PRM011	PRM010
0	0	0	0	0	0	0/1	0/1

Selects count clock.

(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

If M is set to the CR010 register, the interval time is as follows.

Interval time = (M + 1) × Count clock cycle

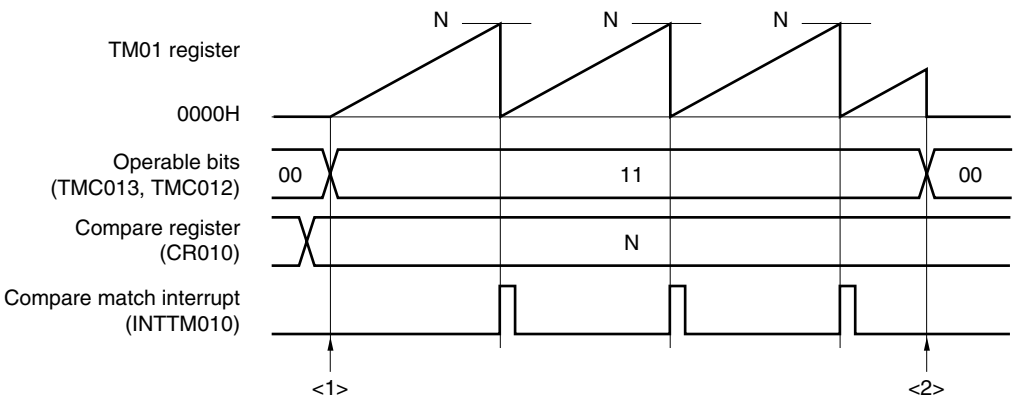
Setting the CR010 register to 0000H is prohibited.

(g) 16-bit capture/compare register 011 (CR011)

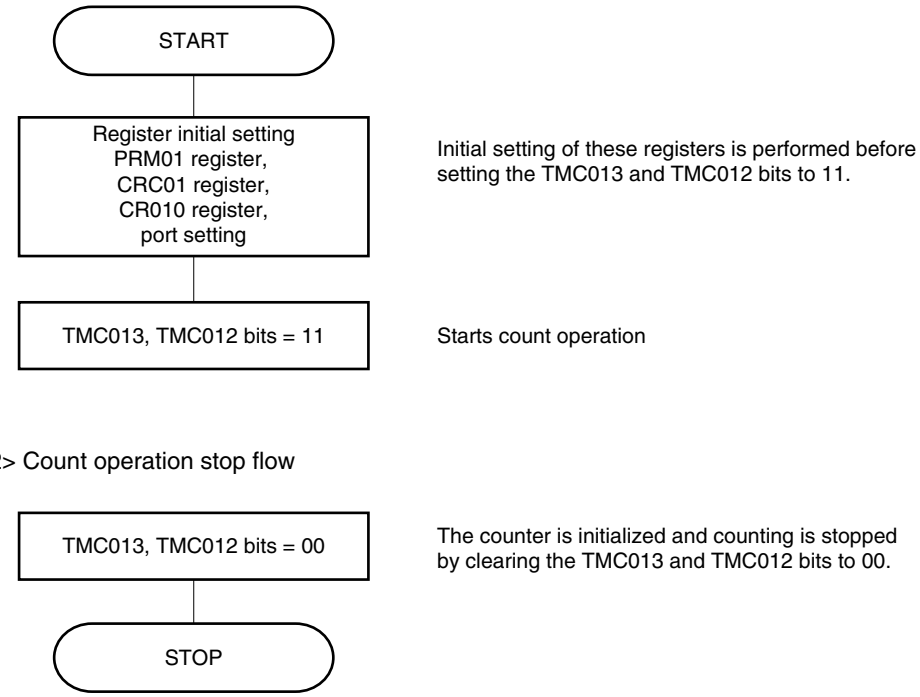
Usually, the CR011 register is not used for the interval timer function. However, a compare match interrupt (INTTM011) is generated when the set value of the CR011 register matches the value of the TM01 register. Therefore, mask the interrupt request by using the interrupt mask flag (TM0MK11).

Downloaded from Arrow.com.

Figure 7-5. Example of Software Processing for Interval Timer Function



<1> Count operation start flow



<2> Count operation stop flow

Downloaded from Arrow.com.

7.4.2 Square wave output operation

When 16-bit timer/event counter 01 operates as an interval timer (see **7.4.1**), a square wave can be output from the TO01 pin by setting the TOC01 register to 03H.

When the TMC01.TMC013 and TMC01.TMC012 bits are set to 11 (count clear & start mode entered upon a match between the TM01 register and the CR010 register), the counting operation is started in synchronization with the count clock.

When the value of the TM01 register later matches the value of the CR010 register, the TM01 register is cleared to 0000H, an interrupt signal (INTTM010) is generated, and output of the TO01 pin is inverted. This TO01 pin output that is inverted at fixed intervals enables TO01 to output a square wave.

Remarks 1. For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 interrupt, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

3. Restrictions are applied to TM01. For details, refer to **7.6 (1) Alternate functions of TI010/TO01 pins.**

Figure 7-6. Block Diagram of Square Wave Output Operation

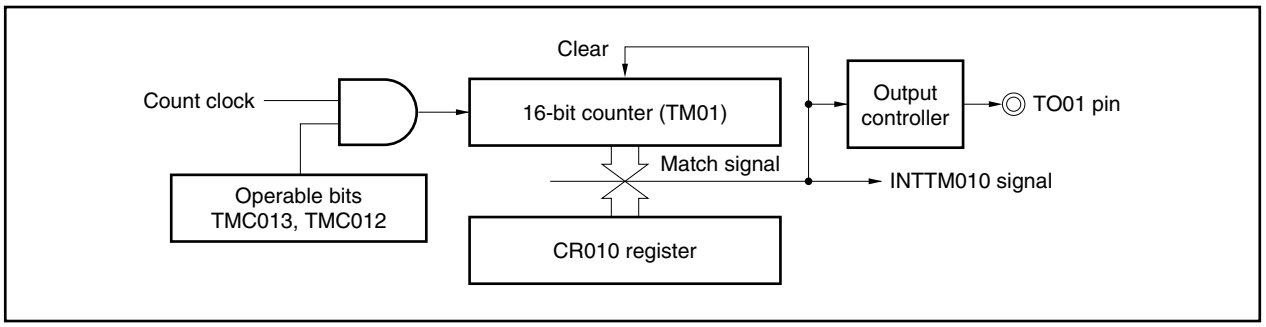
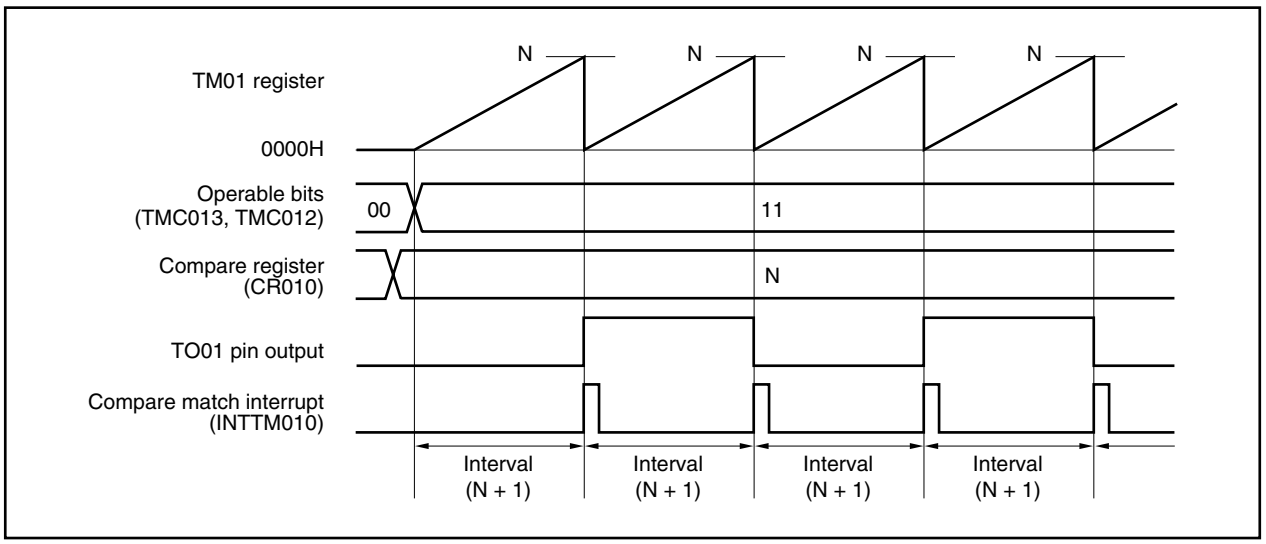


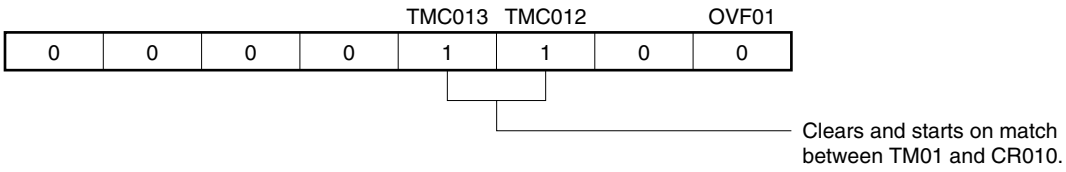
Figure 7-7. Basic Timing Example of Square Wave Output Operation



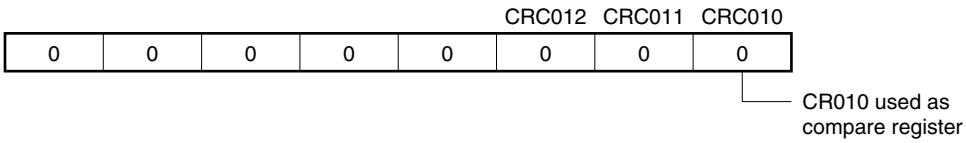
Downloaded from Arrow.com.

Figure 7-8. Example of Register Settings for Square Wave Output Operation

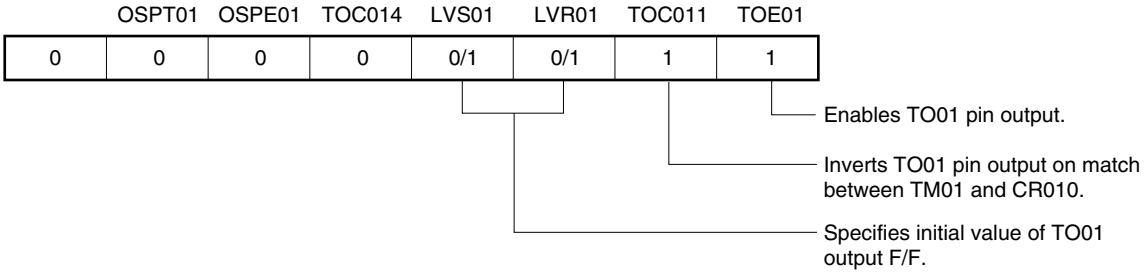
(a) 16-bit timer mode control register 01 (TMC01)



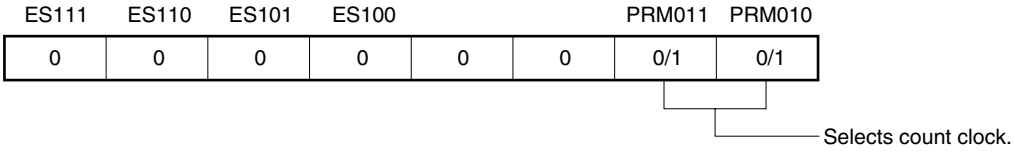
(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)



(d) Prescaler mode register 01 (PRM01)



(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

If M is set to the CR010 register, the square wave frequency is as follows.

$$1 / [2 \times (M + 1) \times \text{Count clock cycle}]$$

Setting the CR010 register to 0000H is prohibited.

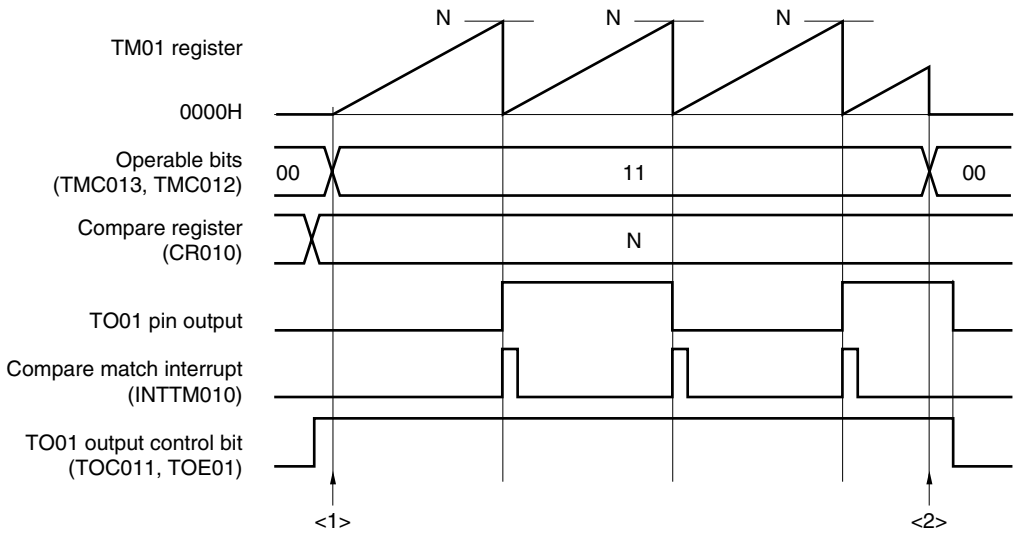
(g) 16-bit capture/compare register 011 (CR011)

Usually, the CR011 register is not used for the square wave output function. However, a compare match interrupt (INTTM011) is generated when the set value of the CR011 register matches the value of the TM01 register.

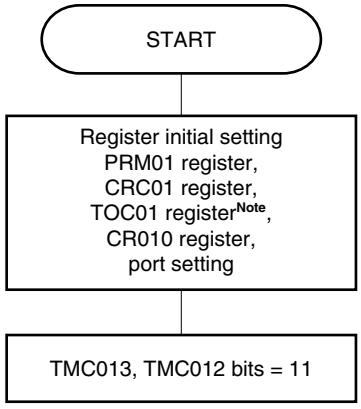
Therefore, mask the interrupt request by using the interrupt mask flag (TM0MK11).

Downloaded from Arrow.com.

Figure 7-9. Example of Software Processing for Square Wave Output Function



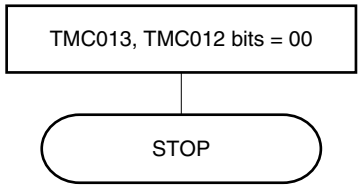
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC013 and TMC012 bits to 11.

Starts count operation.

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC013 and TMC012 bits to 00.

Note Care must be exercised when setting the TOC01 register. For details, see **7.3 (3) 16-bit timer output control register 01 (TOC01)**.

Downloaded from Arrow.com.

7.4.3 External event counter operation

When the PRM01.PRM011 and PRM01.PRM010 bits are set to 11 (for counting up with the valid edge of the TI010 pin) and the TMC01.TMC013 and TMC01.TMC012 bits are set to 11, the valid edge of an external event input is counted, and a match interrupt signal indicating matching between the TM01 register and the CR010 register (INTTM010) is generated.

To input the external event, the TI010 pin is used. Therefore, the timer/event counter cannot be used as an external event counter in the clear & start mode entered by the TI010 pin valid edge input (when the TMC013 and TMC012 bits = 10).

The INTTM010 signal is generated with the following timing.

- Timing of generation of INTTM010 signal (second time or later)
  = Number of times of detection of valid edge of external event input × (Set value of the CR010 register + 1)

However, the first match interrupt immediately after the timer/event counter has started operating is generated with the following timing.

- Number of times of detection of valid edge of external event input × (Set value of the CR010 register + 2)

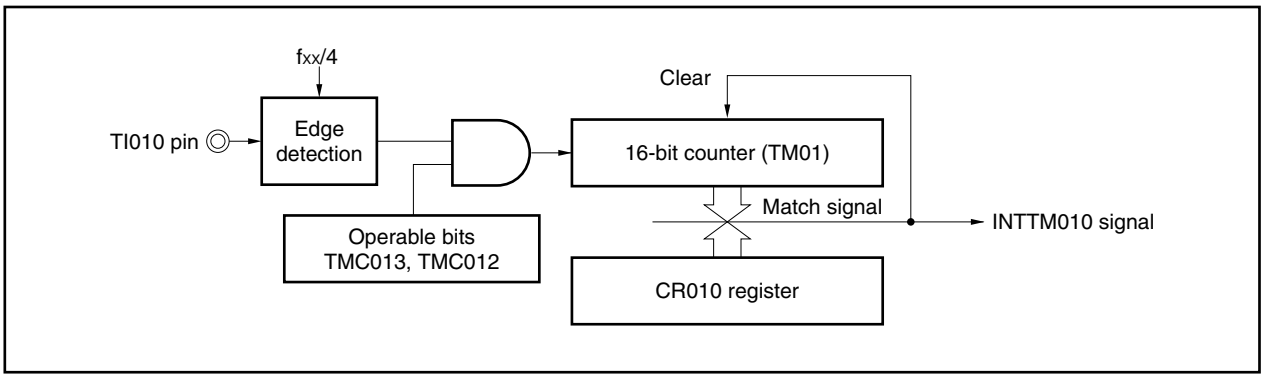
To detect the valid edge, the signal input to the TI010 pin is sampled during the clock cycle of fxx/4. The valid edge is not detected until it is detected two times in a row. Therefore, a noise with a short pulse width can be eliminated.

Remarks 1. For the alternate-function pin (TI010) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 interrupts, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

3. Restrictions are applied to TM01. For details, refer to **7.6 (1) Alternate functions of TI010/TO01 pins.**

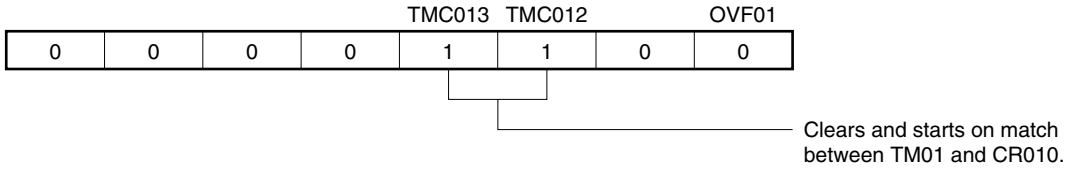
Figure 7-10. Block Diagram of External Event Counter Operation



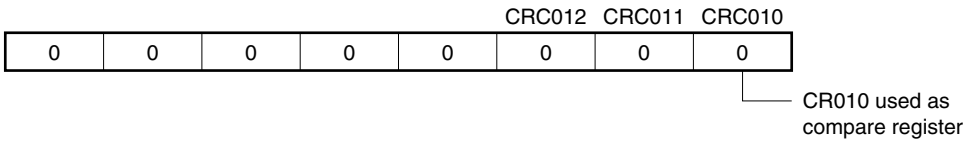
Downloaded from Arrow.com.

Figure 7-11. Example of Register Settings in External Event Counter Mode

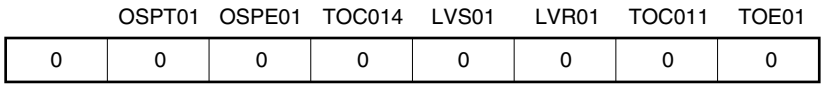
(a) 16-bit timer mode control register 01 (TMC01)



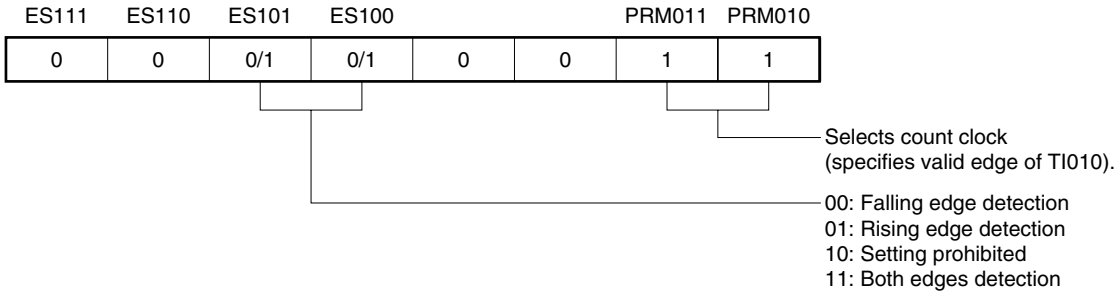
(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)



(d) Prescaler mode register 01 (PRM01)



(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

If M is set to the CR010 register, the interrupt signal (INTTM010) is generated when the number of external events reaches (M + 1).

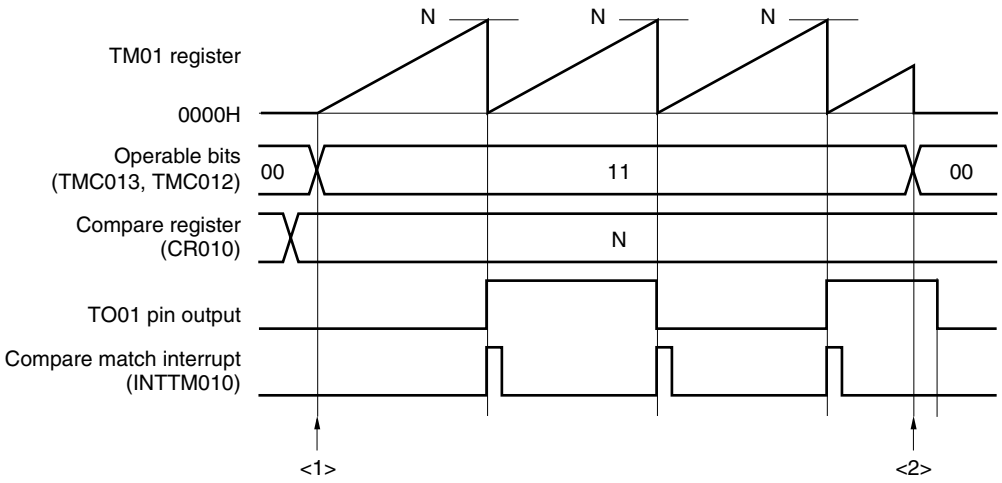
Setting the CR010 register to 0000H is prohibited.

(g) 16-bit capture/compare register 011 (CR011)

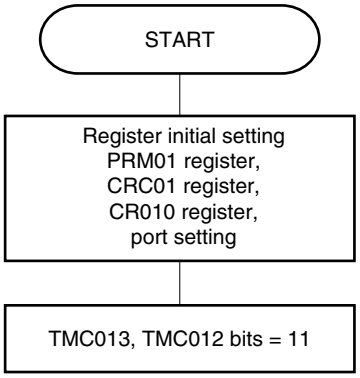
When this register's value matches the count value of the TM01 register, an interrupt signal (INTTM011) is generated. The count value of the TM01 register is not cleared.

Downloaded from Arrow.com.

Figure 7-12. Example of Software Processing in External Event Counter Mode



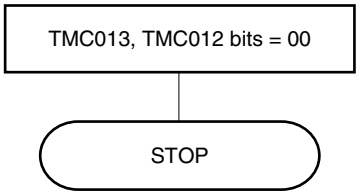
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC013 and TMC012 bits to 11.

Starts count operation.

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC013 and TMC012 bits to 00.

Downloaded from Arrow.com.

7.4.4 Operation in clear & start mode entered by TI010 pin valid edge input

When the TMC01.TMC013 and TMC01.TMC012 bits are set to 10 (clear & start mode entered by the TI010 pin valid edge input) and the count clock (set by the PRM01 register) is supplied to the timer/event counter, the TM01 register starts counting up. When the valid edge of the TI010 pin is detected during the counting operation, the TM01 register is cleared to 0000H and starts counting up again. If the valid edge of the TI010 pin is not detected, the TM01 register overflows and continues counting.

The valid edge of the TI010 pin is a cause to clear the TM01 register. Starting the counter is not controlled immediately after the start of the operation.

The CR010 and CR011 registers are used as compare registers and capture registers.

(a) When the CR010 and CR011 registers are used as compare registers

Signals INTTM010 and INTTM011 are generated when the value of the TM01 register matches the value of the CR010 and CR011 registers.

(b) When the CR010 and CR011 registers are used as capture registers

The count value of the TM01 register is captured to the CR010 register and the INTTM010 signal is generated when the valid edge is input to the TI011 pin (or when the phase reverse to that of the valid edge is input to the TI010 pin).

When the valid edge is input to the TI010 pin, the count value of the TM01 register is captured to the CR011 register and the INTTM011 signal is generated. As soon as the count value has been captured, the counter is cleared to 0000H.

Cautions 1. Do not set the count clock as the valid edge of the TI010 pin (RPM01.PRM011 and RPM01.PRM010 bits = 11). When the PRM011 and PRM010 bits = 11, the TM01 register is cleared.

2. Restrictions are applied to TM01 in which the timer input and output share the same pin. For details, refer to 7.6 (1) Alternate functions of TI010/TO01 pins.

Remarks 1. For the alternate-function pin (TI010) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 interrupt, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

Downloaded from Arrow.com.

(1) Operation in clear & start mode entered by TI010 pin valid edge input (CR010 register: compare register, CR011 register: compare register)

Figure 7-13. Block Diagram of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Compare Register, CR011 Register: Compare Register)

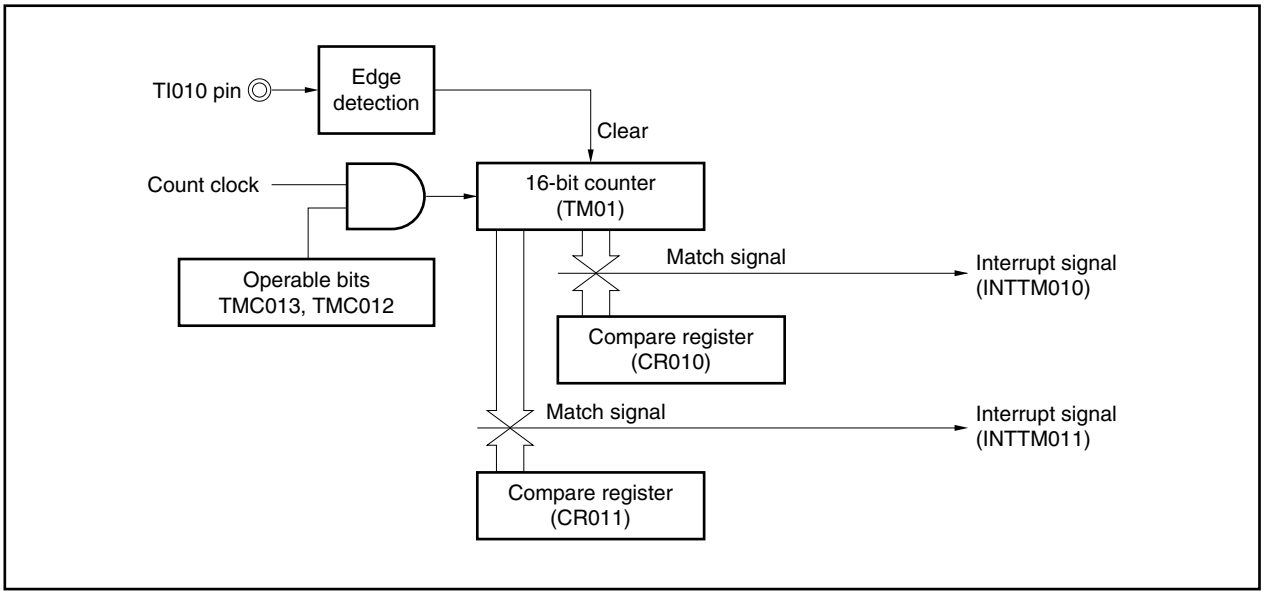
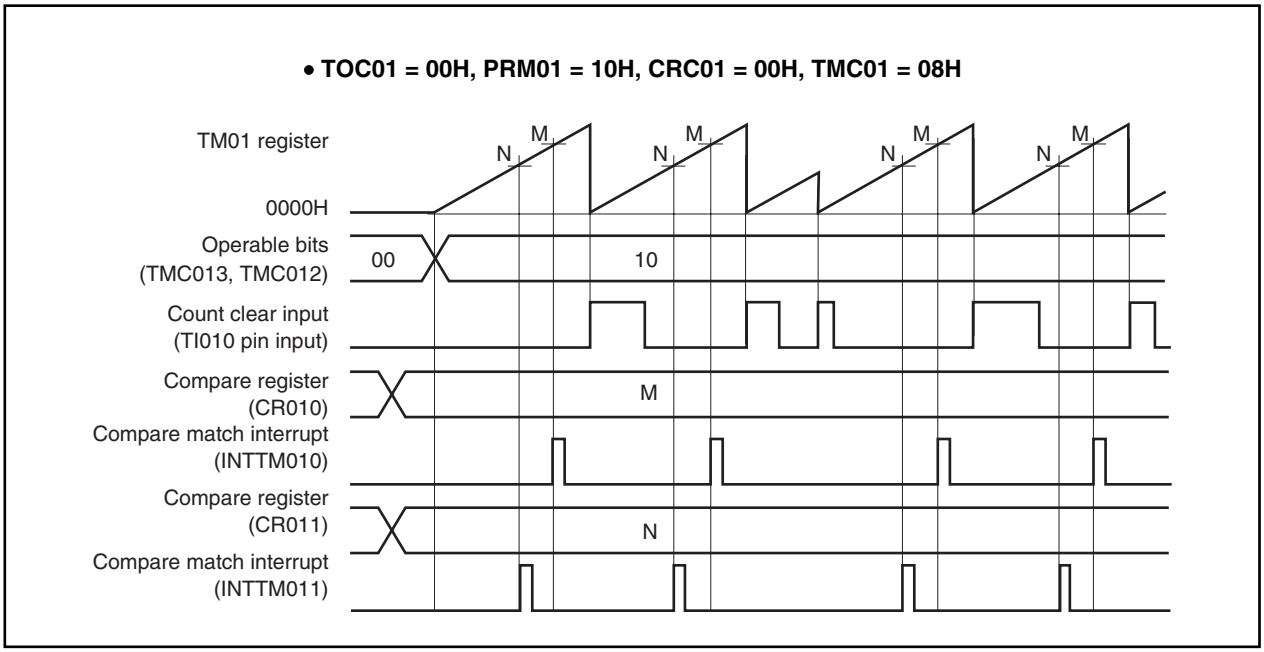


Figure 7-14. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Compare Register, CR011 Register: Compare Register)



Downloaded from Arrow.com.

(2) Operation in clear & start mode entered by TI010 pin valid edge input (CR010 register: compare register, CR011 register: capture register)

Figure 7-15. Block Diagram of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Compare Register, CR011 Register: Capture Register)

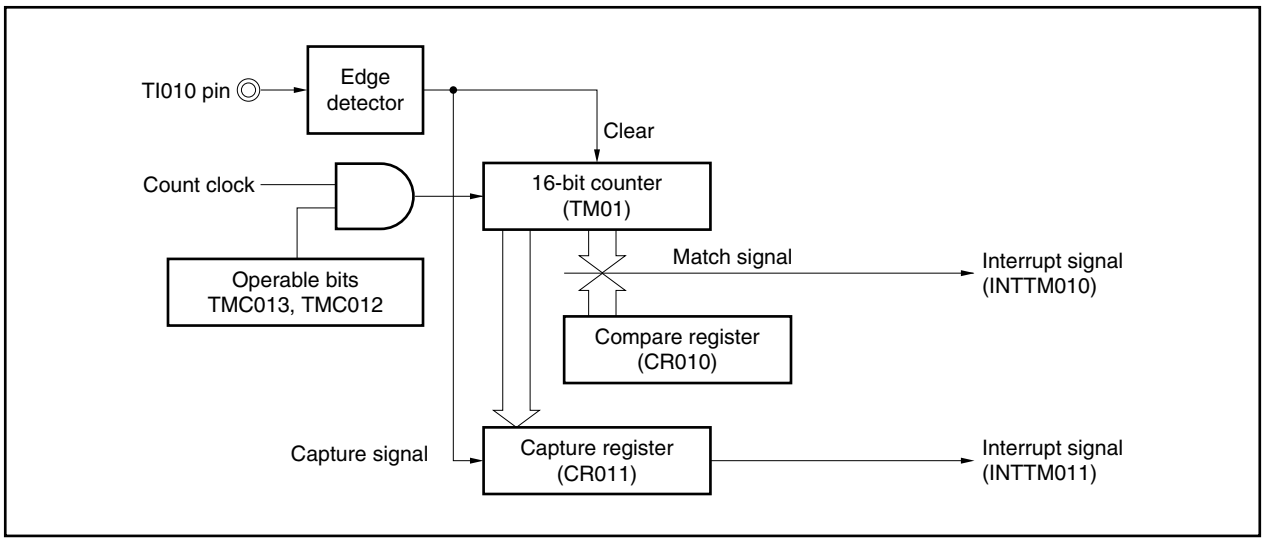
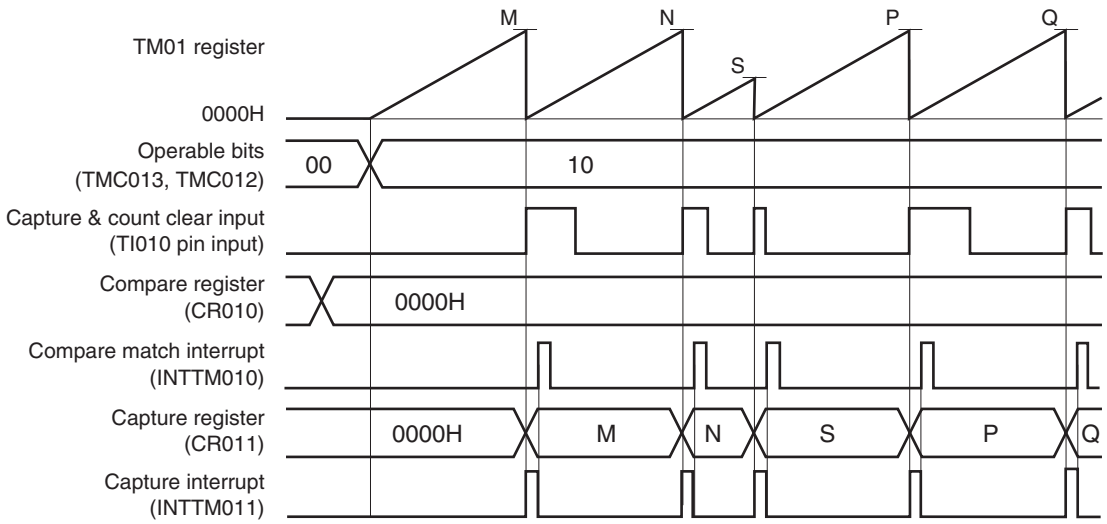


Figure 7-16. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Compare Register, CR011 Register: Capture Register)

• TOC01 = 00H, PRM01 = 10H, CRC01, = 04H, TMC01 = 08H, CR010 = 0000H



The count value is captured to the CR011 register and the TM01 register is cleared (to 0000H) when the valid edge of the TI010 pin is detected. When the count value of the TM01 register is 0000H, a compare match interrupt signal (INTTM010) is generated.

Downloaded from Arrow.com.

(3) Operation in clear & start mode entered by TI010 pin valid edge input (CR010 register: capture register, CR011 register: compare register)

Figure 7-17. Block Diagram of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Compare Register)

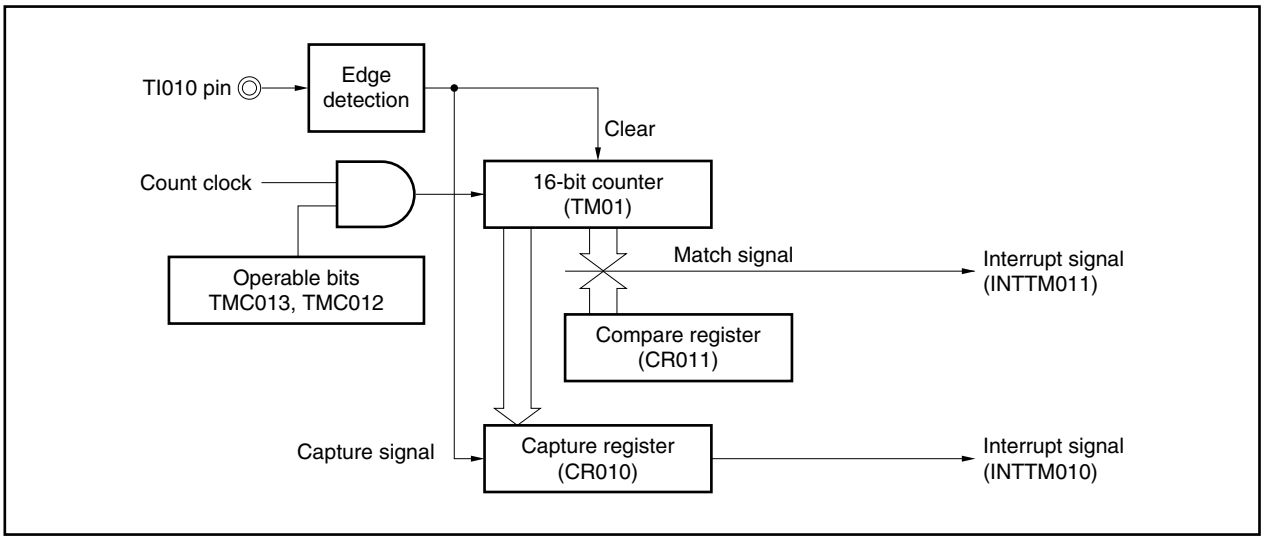
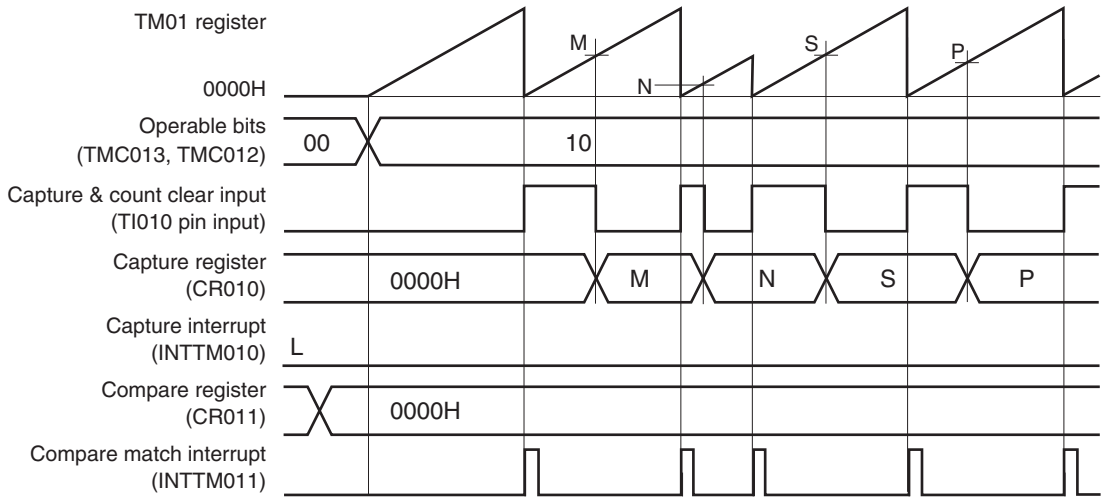


Figure 7-18. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Compare Register)

• TOC01 = 00H, PRM01 = 10H, CRC01 = 03H, TMC01 = 08H, CR011 = 0000H



The TM01 register is cleared at the rising edge detection of the TI010 pin and it is captured to the CR010 register at the falling edge detection of the TI010 pin.

When the CRC01.CRC011 bit is set to 1, the count value of the TM01 register is captured to CR010 in the phase reverse to that of the signal input to the TI010 pin, but the capture interrupt signal (INTTM010) is not generated. However, the INTTM010 signal is generated when the valid edge of the TI011 pin is detected. Mask the INTTM010 signal when it is not used.

Downloaded from Arrow.com.

(4) Operation in clear & start mode entered by TI010 pin valid edge input (CR010 register: capture register, CR011 register: capture register)

Figure 7-19. Block Diagram of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Capture Register)

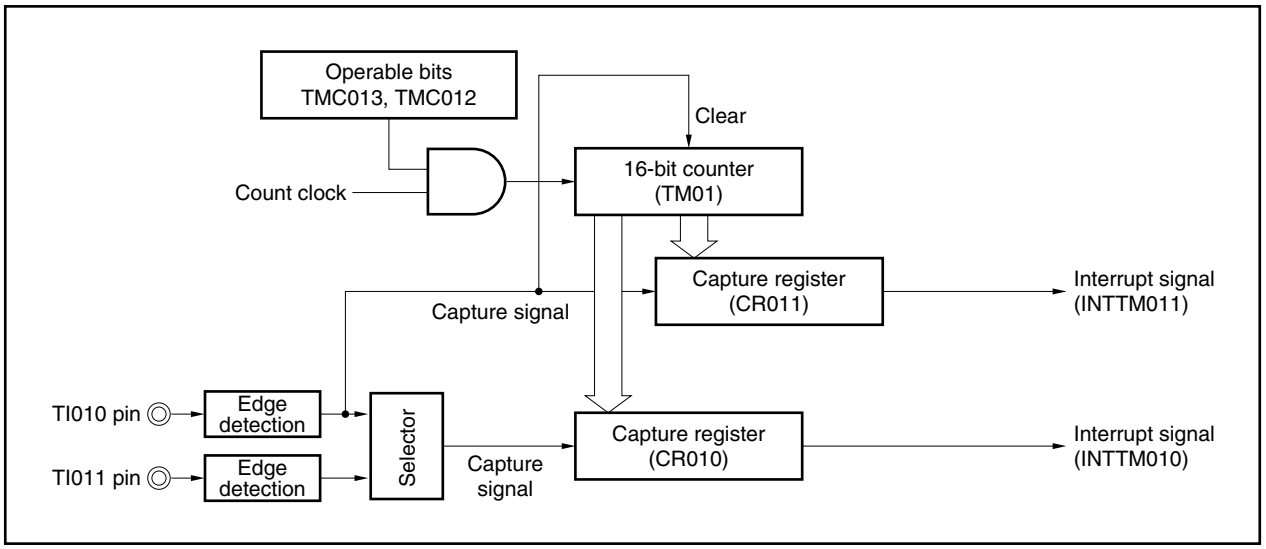
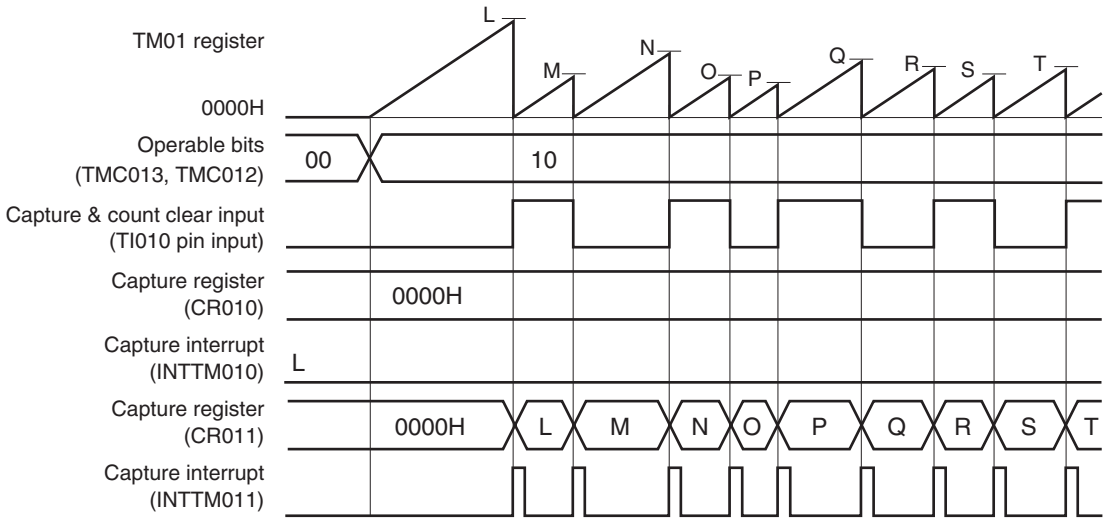


Figure 7-20. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Capture Register) (1/3)

(a) TOC01 = 00H, PRM01 = 30H, CRC01 = 05H, TMC01 = 08H



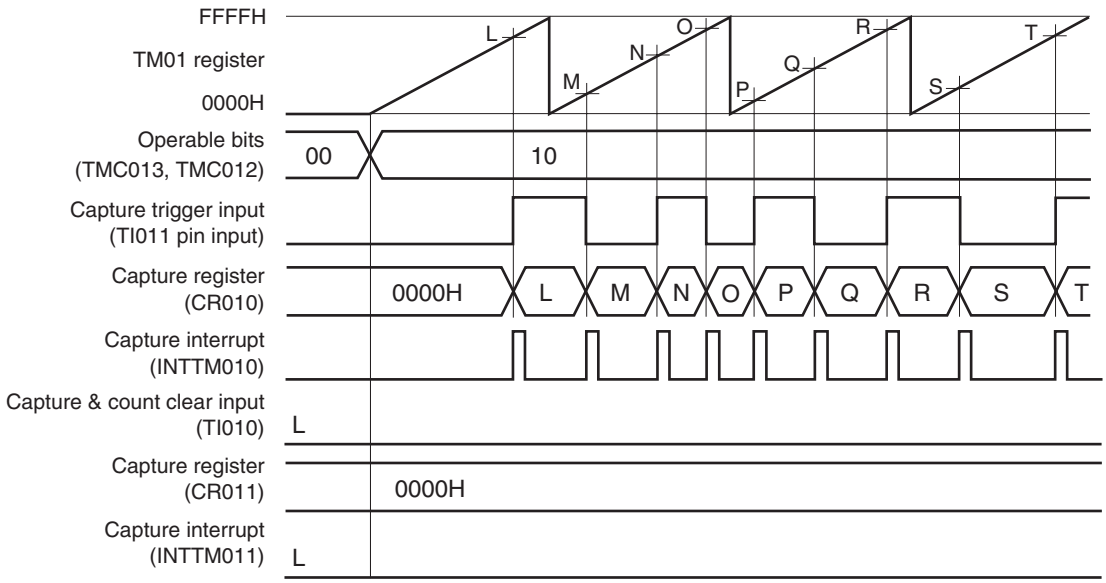
This is an application example where the count value is captured to the CR011 register and the TM01 register is cleared, when the rising or falling edge of the TI010 pin is detected.

When the edge of the TI011 pin is detected, an interrupt signal (INTTM010) is generated. Mask the INTTM010 signal when it is not used.

Downloaded from Arrow.com.

Figure 7-20. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Capture Register) (2/3)

(b) TOC01 = 00H, PRM01 = C0H, CRC01 = 05H, TMC01 = 08H

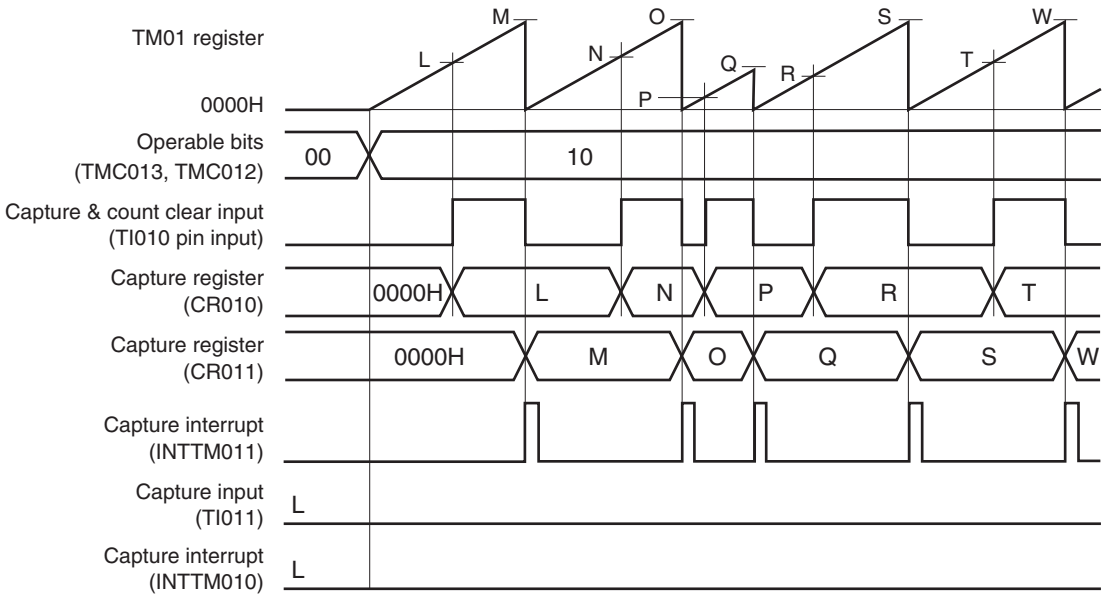


This is a timing example where an edge is not input to the TI010 pin, in an application where the count value is captured to the CR010 register when the rising or falling edge of the TI011 pin is detected.

Downloaded from Arrow.com.

Figure 7-20. Timing Example of Clear & Start Mode Entered by TI010 Pin Valid Edge Input (CR010 Register: Capture Register, CR011 Register: Capture Register) (3/3)

(c) TOC01 = 00H, PRM01 = 00H, CRC01 = 07H, TMC01 = 08H



This is an application example where the pulse width of the signal input to the TI010 pin is measured.
By setting the CRC01 register, the count value can be captured to the CR010 register in the phase reverse to the falling edge of the TI010 pin (i.e., rising edge) and to the CR011 register at the falling edge of the TI010 pin.
The high- and low-level widths of the input pulse can be calculated by the following expressions.

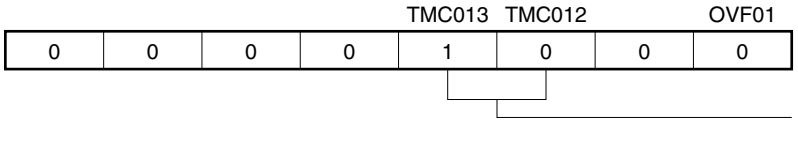
- High-level width = [CR011 register value] – [CR010 register value] × [Count clock cycle]
- Low-level width = [CR010 register value] × [Count clock cycle]

If the reverse phase of the TI010 pin is selected as a trigger to capture the count value to the CR010 register, the INTTM010 signal is not generated. Read the values of the CR010 and CR011 registers to measure the pulse width immediately after the INTTM011 signal is generated.
However, if the valid edge specified by the PRM01.ES111 and PRM01.ES110 bits is input to the TI011 pin, the count value is not captured but the INTTM010 signal is generated. To measure the pulse width of the TI010 pin, mask the INTTM010 signal when it is not used.

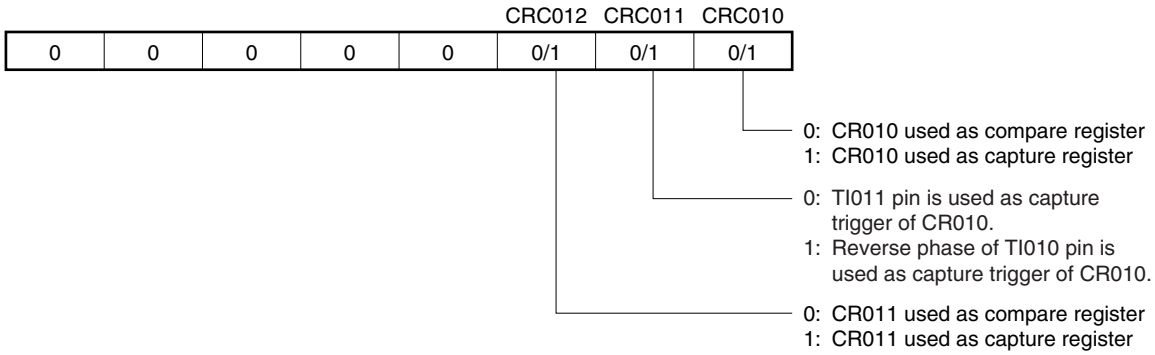
Downloaded from Arrow.com.

Figure 7-21. Example of Register Settings in Clear & Start Mode Entered by TI010 Pin Valid Edge Input (1/2)

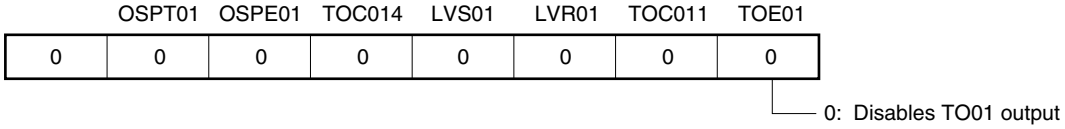
(a) 16-bit timer mode control register 01 (TMC01)



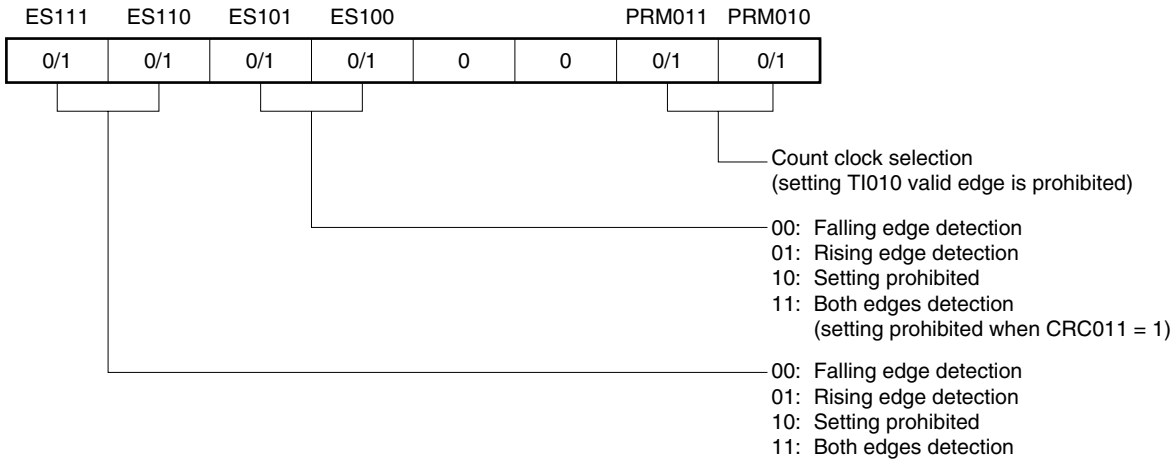
(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)



(d) Prescaler mode register 01 (PRM01)



Downloaded from Arrow.com.

Figure 7-21. Example of Register Settings in Clear & Start Mode Entered by TI010 Pin Valid Edge Input (2/2)

(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

When this register is used as a compare register and when its value matches the count value of the TM01 register, an interrupt signal (INTTM010) is generated. The count value of the TM01 register is not cleared.

To use this register as a capture register, select either the TI010 or TI011 pin input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of the TM01 register is stored in the CR010 register.

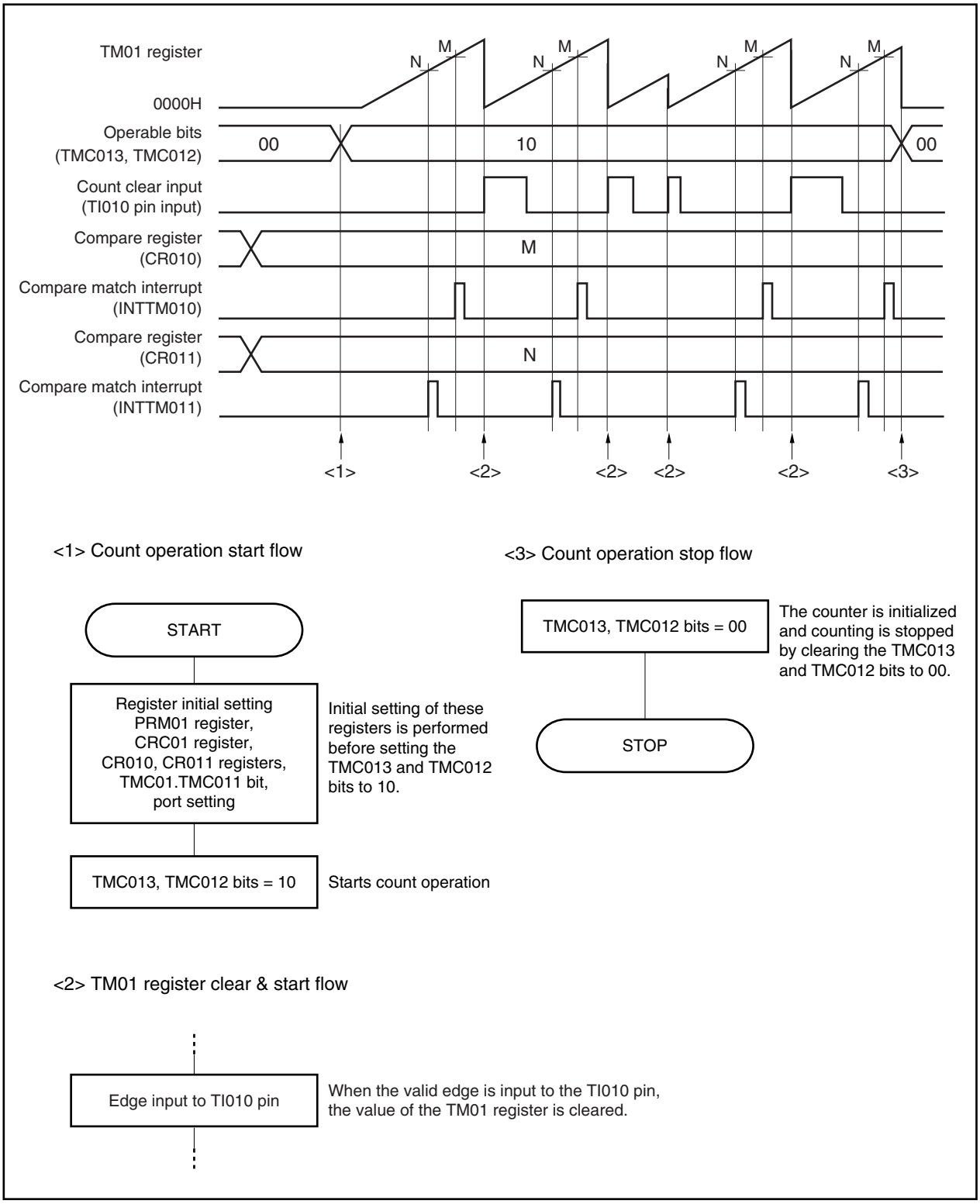
(g) 16-bit capture/compare register 011 (CR011)

When this register is used as a compare register and when its value matches the count value of the TM01 register, an interrupt signal (INTTM011) is generated. The count value of the TM01 register is not cleared.

When this register is used as a capture register, the TI010 pin input is used as a capture trigger. When the valid edge of the capture trigger is detected, the count value of the TM01 register is stored in the CR011 register.

Downloaded from Arrow.com.

Figure 7-22. Example of Software Processing in Clear & Start Mode Entered by TI010 Pin Valid Edge Input



Downloaded from Arrow.com.

7.4.5 Free-running timer operation

When the TMC01.TMC013 and TMC01.TMC012 bits are set to 01 (free-running timer mode), 16-bit timer/event counter 01 continues counting up in synchronization with the count clock. When it has counted up to FFFFH, the overflow flag (TMC01.OVF01 bit) is set to 1 at the next clock, and the TM01 register is cleared (to 0000H) and continues counting. Clear the OVF01 bit to 0 by executing the CLR instruction via software.

The following three types of free-running timer operations are available.

- Both the CR010 and CR011 registers are used as compare registers.
- Either the CR010 register or CR011 register is used as a compare register and the other is used as a capture register.
- Both the CR010 and CR011 registers are used as capture registers.

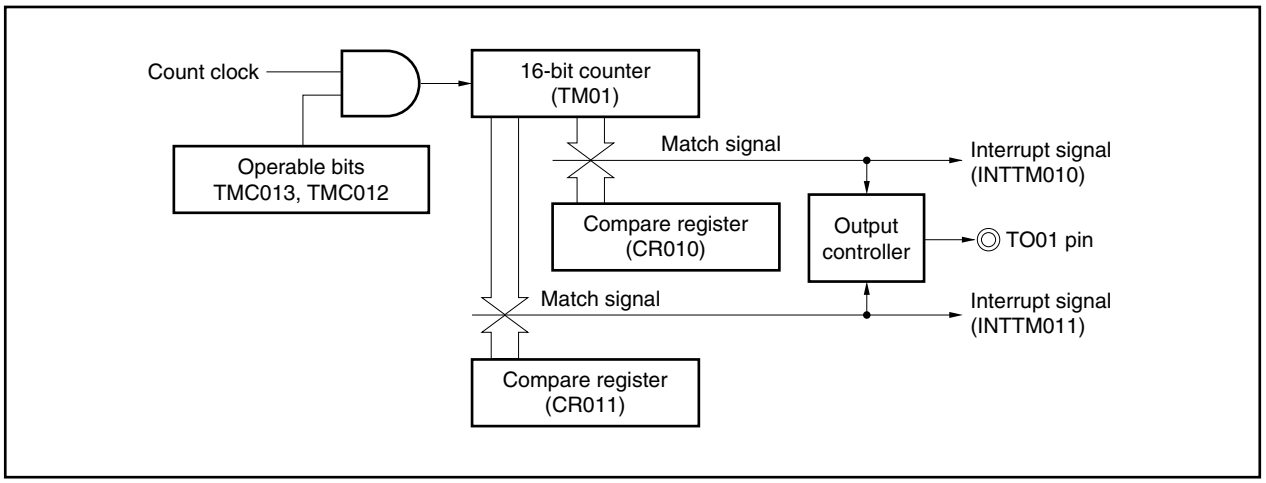
Remarks 1. For the alternate-function pin (TO01) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 and INTTM011 interrupts, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

3. Restrictions are applied to TM01. For details, refer to **7.6 (1) Alternate functions of TI010/TO01 pins.**

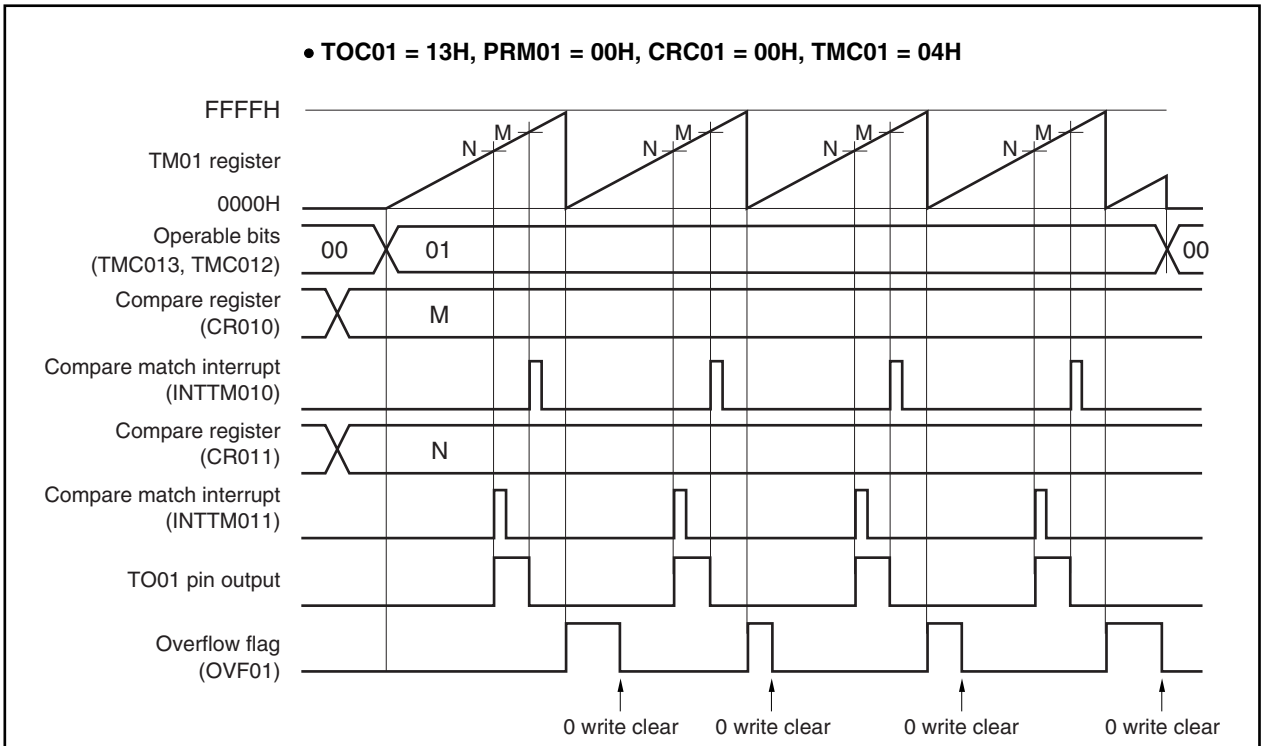
(1) Free-running timer mode operation (CR010 register: compare register, CR011 register: compare register)

Figure 7-23. Block Diagram of Free-Running Timer Mode (CR010 Register: Compare Register, CR011 Register: Compare Register)



Downloaded from Arrow.com.

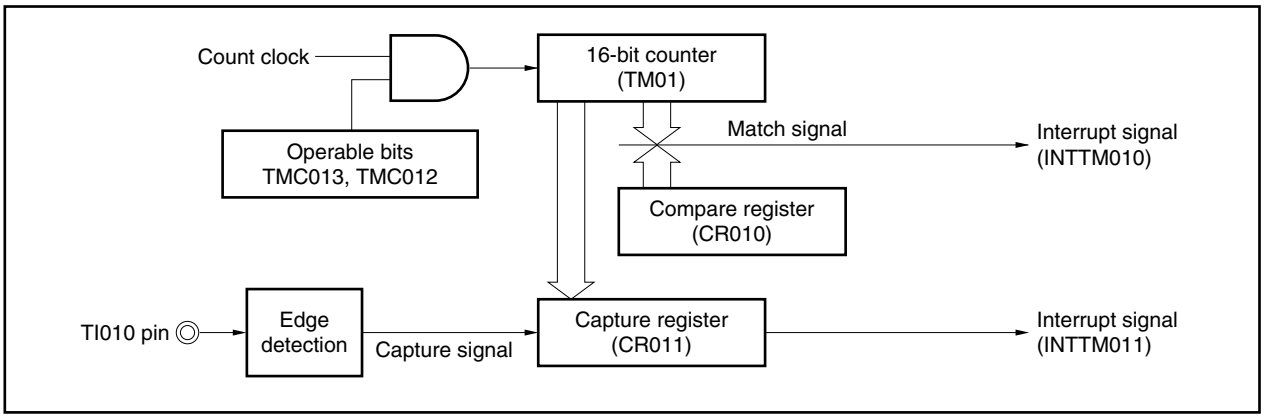
Figure 7-24. Timing Example of Free-Running Timer Mode
(CR010 Register: Compare Register, CR011 Register: Compare Register)



This is an application example where two compare registers are used in the free-running timer mode.
The output level of the TO01 pin is reversed each time the count value of the TM01 register matches the set values of the CR010 and CR011 registers. When the count value matches the register value, the INTTM010 or INTTM011 signal is generated.

(2) Free-running timer mode operation
(CR010 register: compare register, CR011 register: capture register)

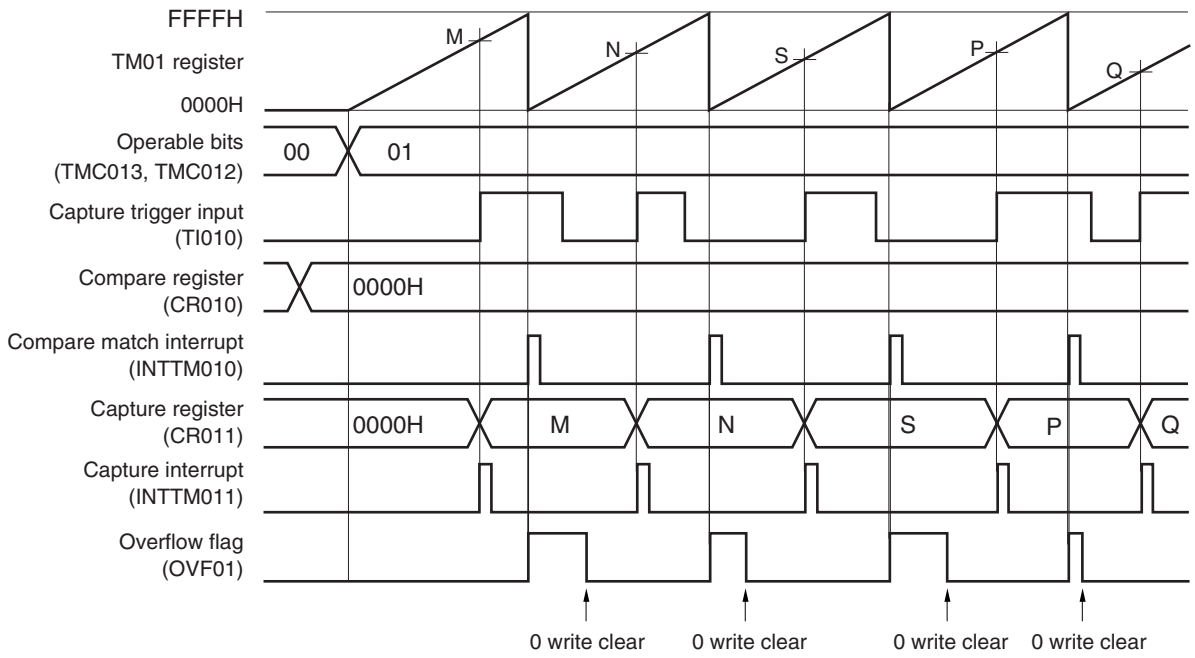
Figure 7-25. Block Diagram of Free-Running Timer Mode
(CR010 Register: Compare Register, CR011 Register: Capture Register)



Downloaded from Arrow.com.

Figure 7-26. Timing Example of Free-Running Timer Mode
(CR010 Register: Compare Register, CR011 Register: Capture Register)

• TOC01 = 00H, PRM01 = 10H, CRC01 = 04H, TMC01 = 04H



This is an application example where a compare register and a capture register are used at the same time in the free-running timer mode.

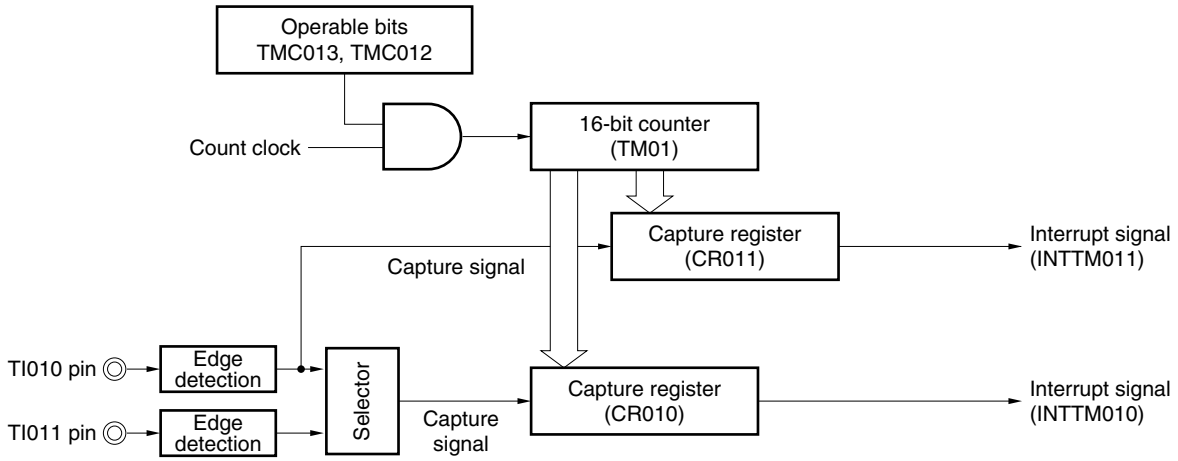
In this example, the INTTM010 signal is generated each time the count value of the TM01 register matches the set value of the CR010 register (compare register). (Because the TI010 and TO01 pins are alternate functions, the timer output (TO01) cannot be used.) In addition, the INTTM011 signal is generated and the count value of the TM01 register is captured to the CR011 register each time the valid edge of the TI010 pin is detected.

Downloaded from Arrow.com.

(3) Free-running timer mode operation

(CR010 register: capture register, CR011 register: capture register)

Figure 7-27. Block Diagram of Free-Running Timer Mode
(CR010 Register: Capture Register, CR011 Register: Capture Register)

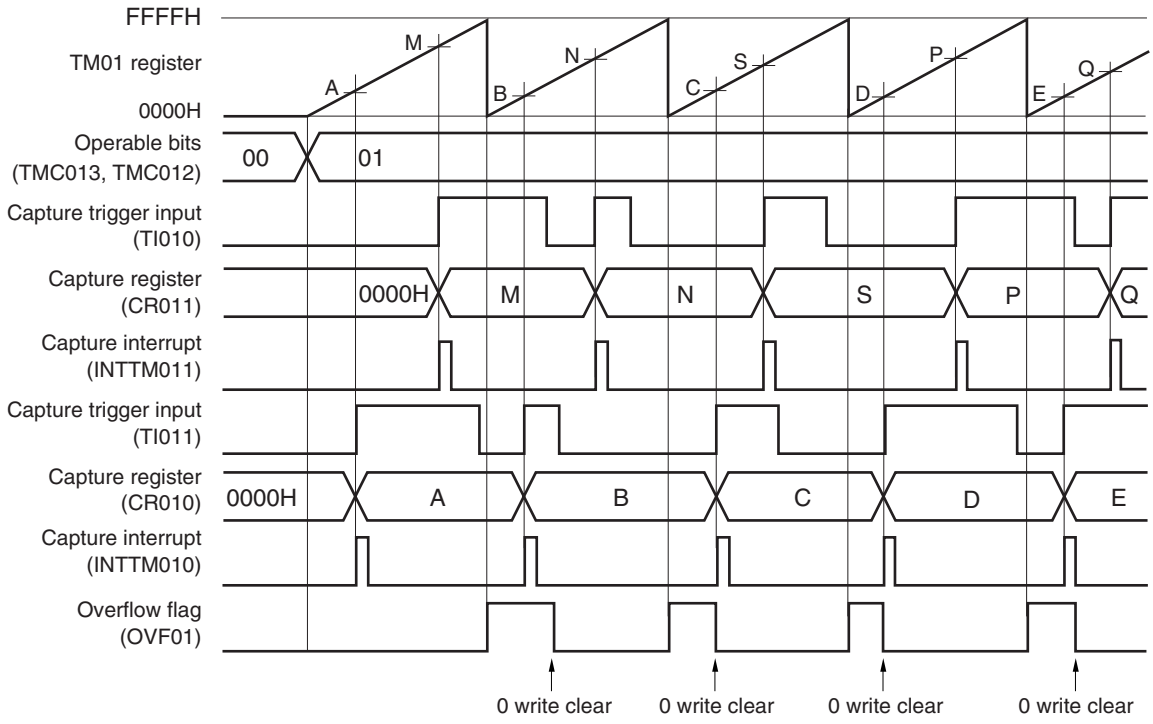


Remark If both the CR010 and CR011 registers are used as capture registers in the free-running timer mode, the TO01 pin cannot be used.

Downloaded from Arrow.com.

Figure 7-28. Timing Example of Free-Running Timer Mode (CR010 Register: Capture Register, CR011 Register: Capture Register) (1/2)

(a) TOC01 = 00H, PRM01 = 50H, CRC01 = 05H, TMC01 = 04H



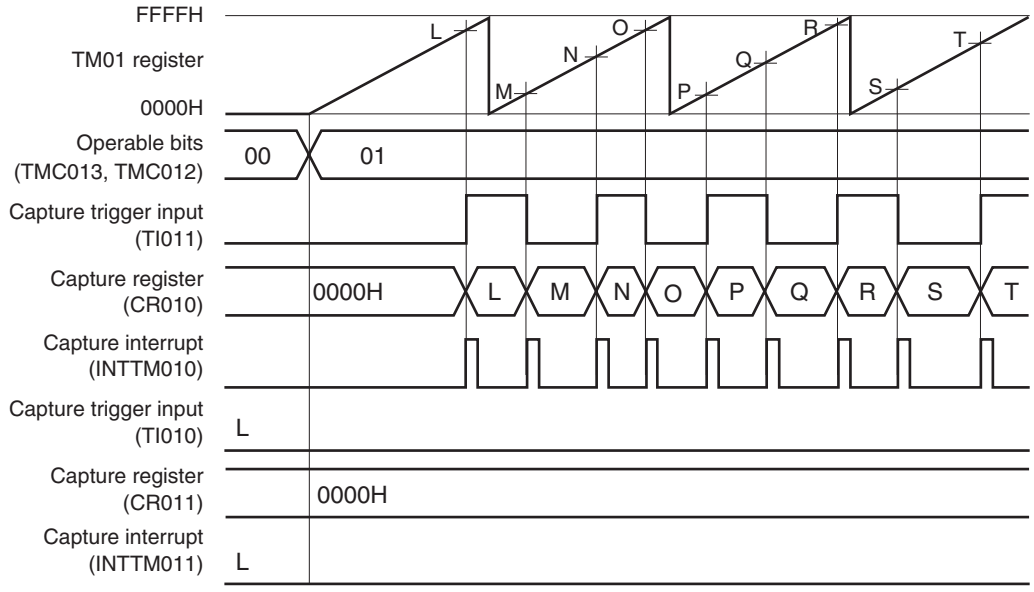
This is an application example where the count values that have been captured at the valid edges of separate capture trigger signals are stored in separate capture registers in the free-running timer mode.

The count value is captured to the CR011 register when the valid edge of the TI010 pin input is detected and to the CR010 register when the valid edge of the TI011 pin input is detected.

Downloaded from Arrow.com.

Figure 7-28. Timing Example of Free-Running Timer Mode (CR010 Register: Capture Register, CR011 Register: Capture Register) (2/2)

(b) TOC01 = 00H, PRM01 = C0H, CRC01 = 05H, TMC01 = 04H



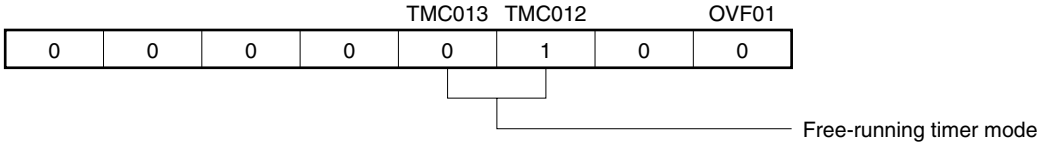
This is an application example where both the edges of the TI011 pin are detected and the count value is captured to the CR010 register in the free-running timer mode.

When both the CR010 and CR011 registers are used as capture registers and when the valid edge of only the TI011 pin is to be detected, the count value cannot be captured to the CR011 register.

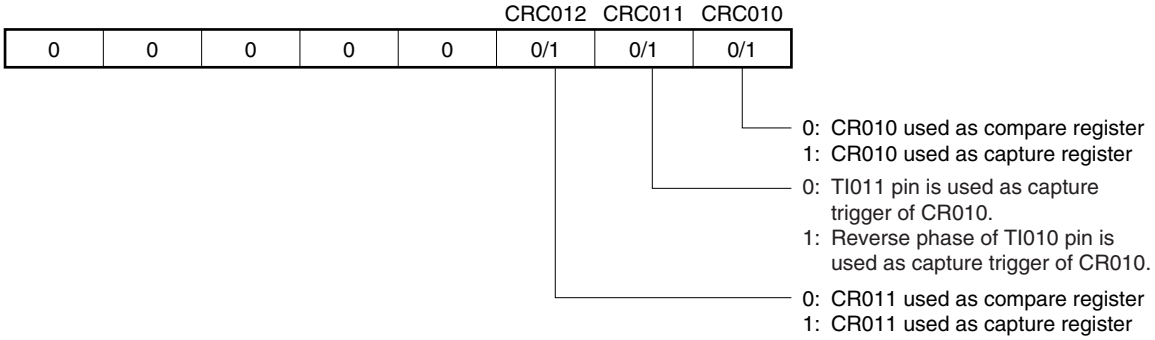
Downloaded from Arrow.com.

Figure 7-29. Example of Register Settings in Free-Running Timer Mode (1/2)

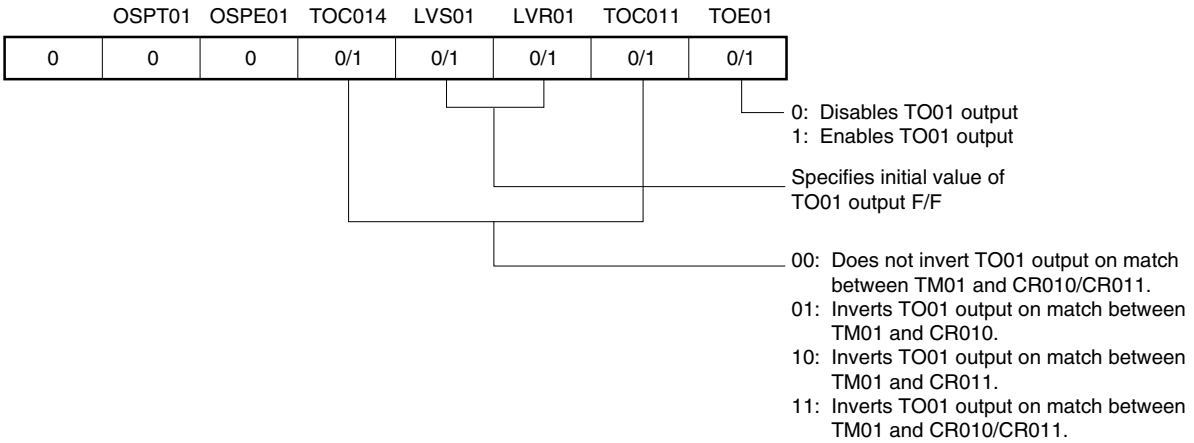
(a) 16-bit timer mode control register 01 (TMC01)



(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)[Note]

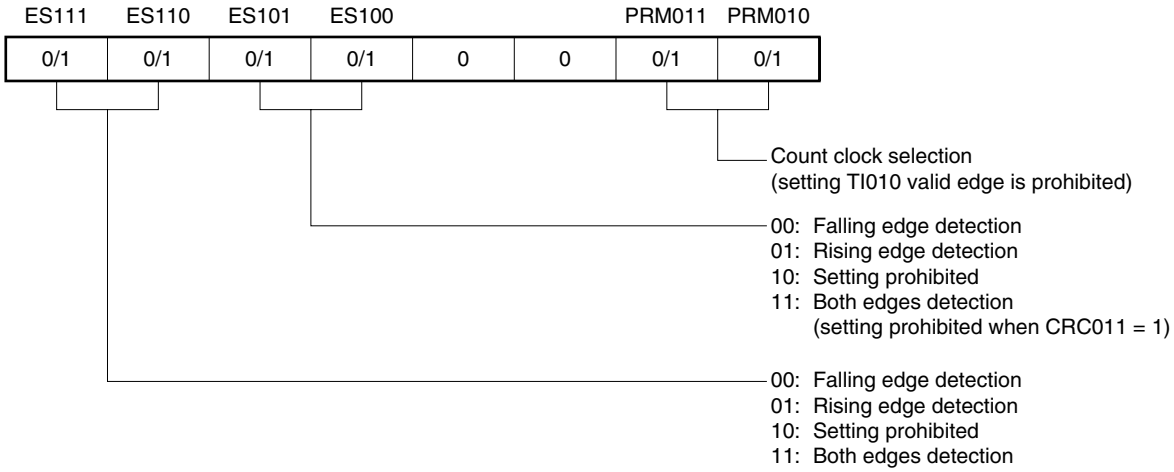


Note When this register is captured by the valid edge of the TI010 pin, the timer output (TO01 pin) cannot be used because TI010 and TO01 share the same pin. Set the TOC01 register to 00H.

Downloaded from Arrow.com.

Figure 7-29. Example of Register Settings in Free-Running Timer Mode (2/2)

(d) Prescaler mode register 01 (PRM01)



(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

When this register is used as a compare register and when its value matches the count value of the TM01 register, an interrupt signal (INTTM010) is generated. The count value of the TM01 register is not cleared.

To use this register as a capture register, select either the TI010 or TI011 pin input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of the TM01 register is stored in the CR010 register.

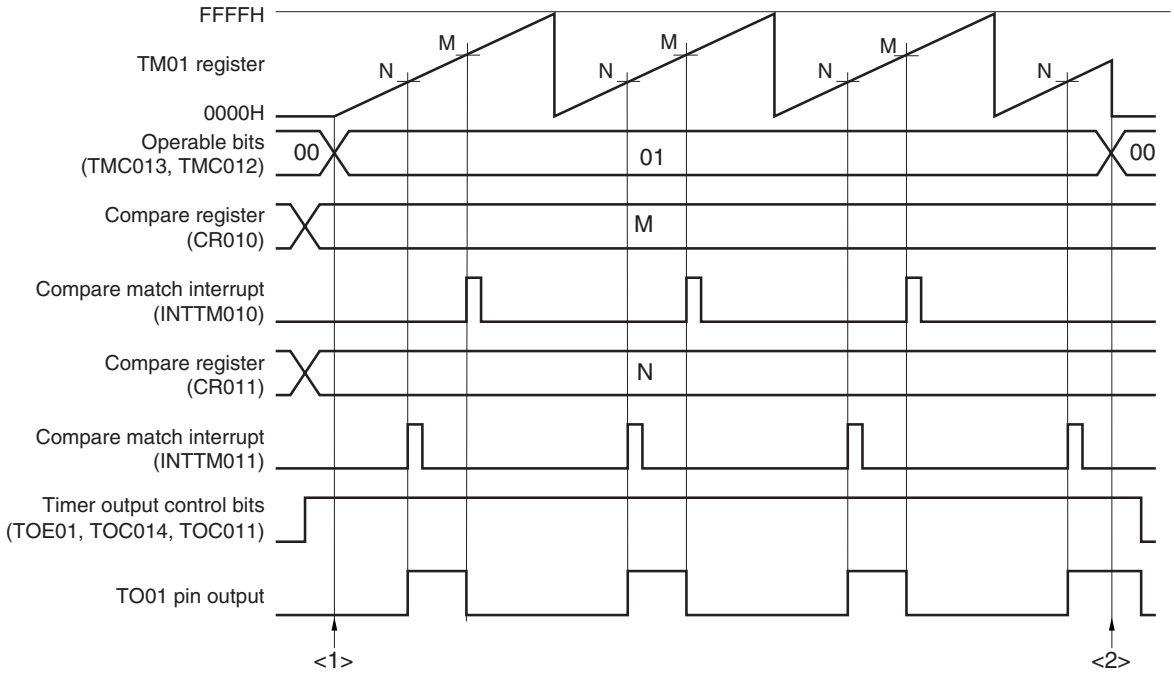
(g) 16-bit capture/compare register 011 (CR011)

When this register is used as a compare register and when its value matches the count value of the TM01 register, an interrupt signal (INTTM011) is generated. The count value of the TM01 register is not cleared.

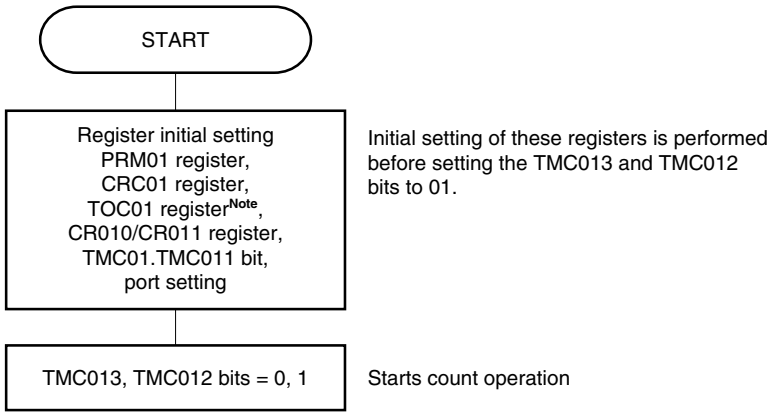
When this register is used as a capture register, the TI010 pin input is used as a capture trigger. When the valid edge of the capture trigger is detected, the count value of the TM01 register is stored in the CR011 register.

Downloaded from Arrow.com.

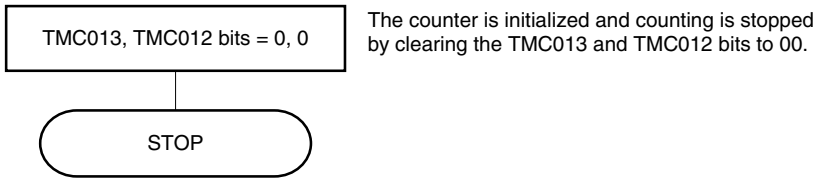
Figure 7-30. Example of Software Processing in Free-Running Timer Mode



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting the TOC01 register. For details, see **7.3 (3) 16-bit timer output control register 01 (TOC01)**. In addition, when this register is captured by the valid edge of the TI010 pin, the timer output (TO01 pin) cannot be used because TI010 and TO01 share the same pin. Set the TOC01 register to 00H.

Downloaded from Arrow.com.

7.4.6 PPG output operation

A rectangular wave having a pulse width set in advance by the CR011 register is output from the TO01 pin as a PPG (Programmable Pulse Generator) signal during a cycle set by the CR010 register when the TMC01.TMC013 and TMC01.TMC012 bits are set to 11 (clear & start upon a match between the TM01 register and the CR010 register).

The pulse cycle and duty factor of the pulse generated as the PPG output are as follows.

- Pulse cycle = (Set value of the CR010 register + 1) × Count clock cycle
- Duty = (Set value of the CR011 register + 1) / (Set value of the CR010 register + 1)

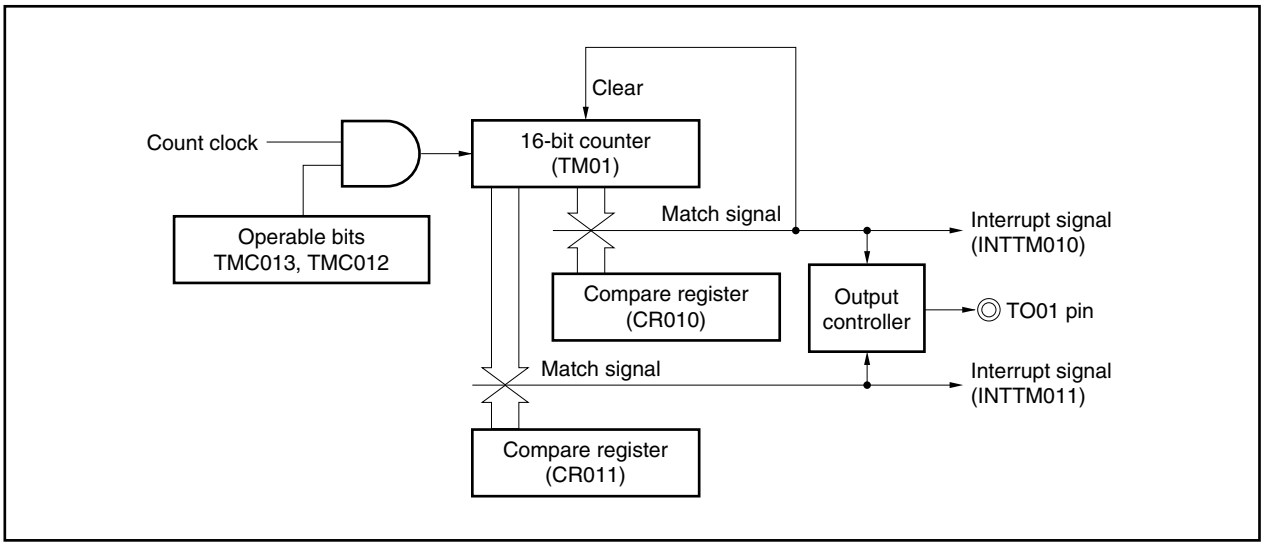
Caution To change the duty factor (value of the CR011 register) during operation, see 7.5.1 Rewriting CR011 register during TM01 operation.

Remarks 1. For the alternate-function pin (TO01) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 and INTTM011 interrupts, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

3. Restrictions are applied to TM01 in which the timer input and output share the same pin. For details, refer to **7.6 (1) Alternate functions of TI010/TO01 pins.**

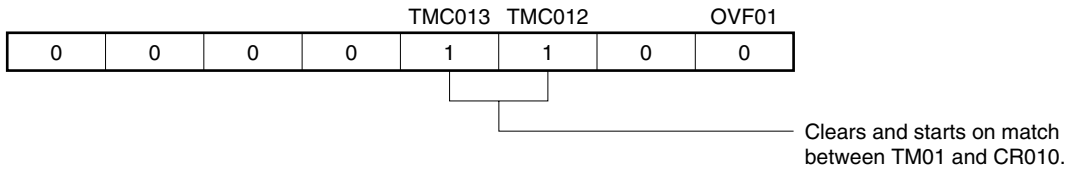
Figure 7-31. Block Diagram of PPG Output Operation



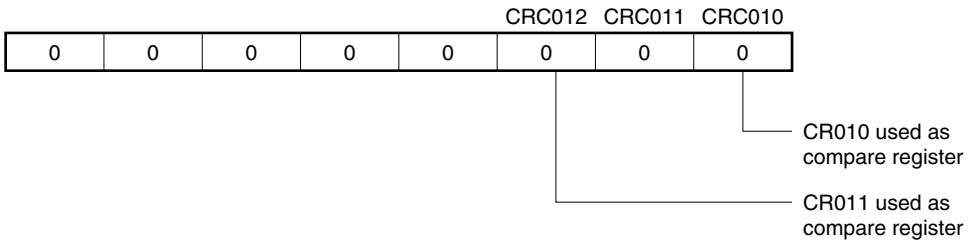
Downloaded from Arrow.com.

Figure 7-32. Example of Register Settings for PPG Output Operation

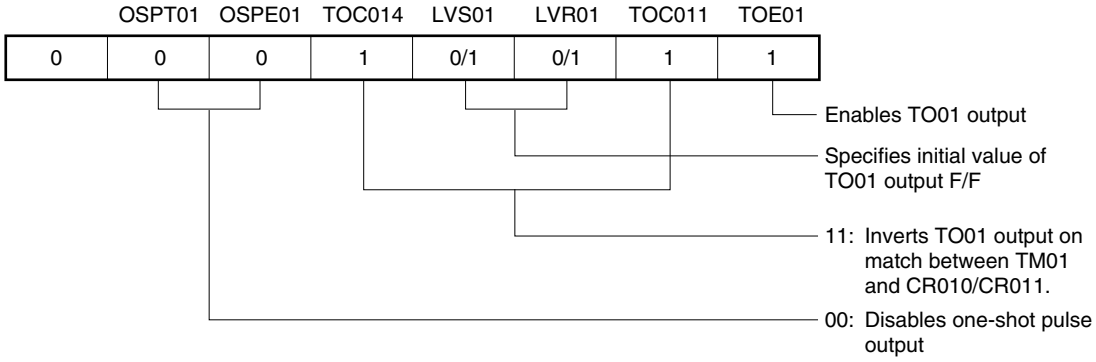
(a) 16-bit timer mode control register 01 (TMC01)



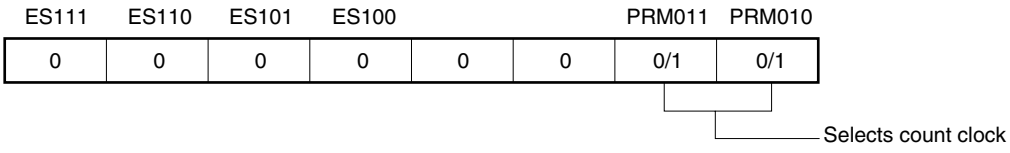
(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)



(d) Prescaler mode register 01 (PRM01)



(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

An interrupt signal (INTTM010) is generated when the value of this register matches the count value of the TM01 register.

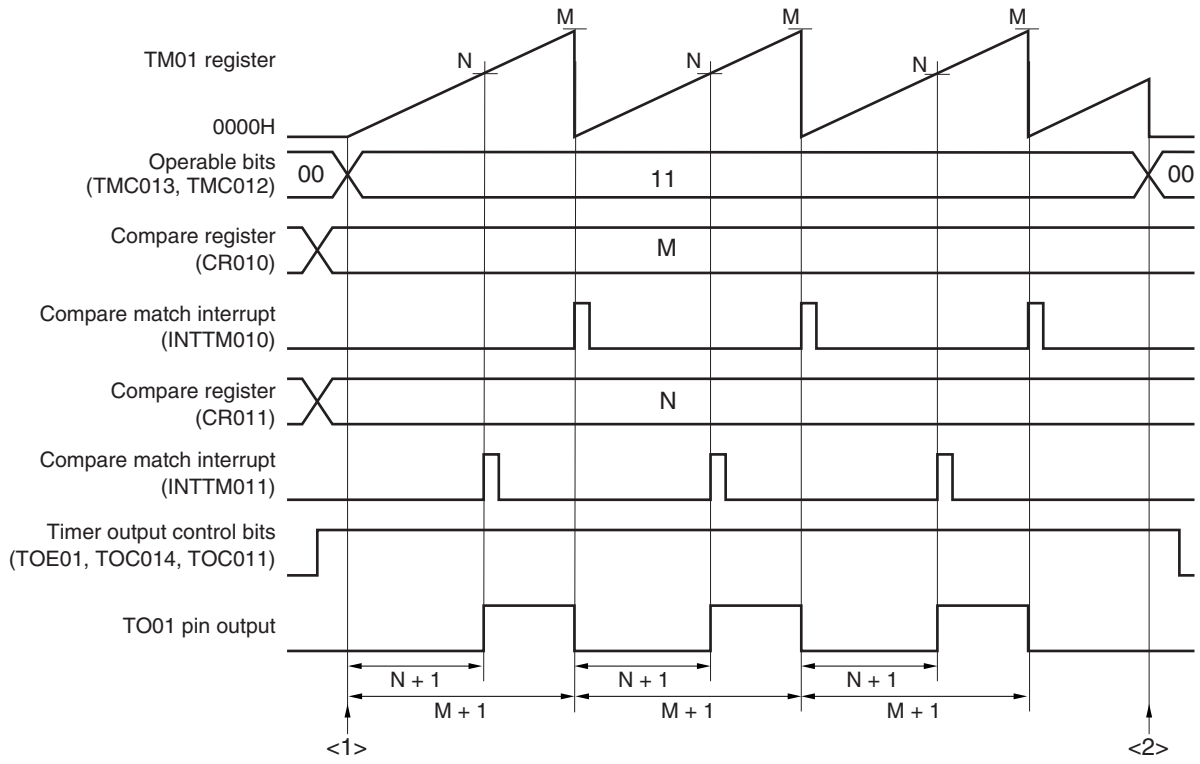
(g) 16-bit capture/compare register 011 (CR011)

An interrupt signal (INTTM011) is generated when the value of this register matches the count value of the TM01 register. The count value of the TM01 register is not cleared.

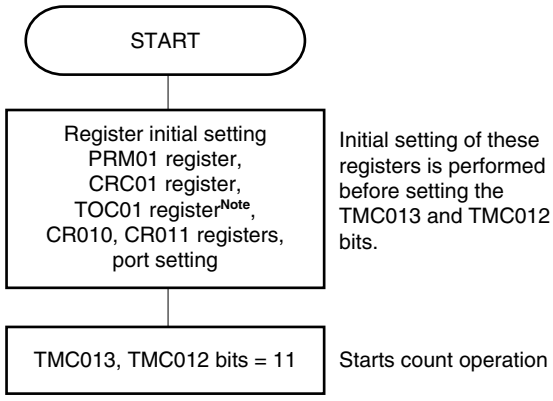
Caution Set values to the CR010 and CR011 registers such that the condition 0000H ≤ CR011 < CR010 ≤ FFFFH is satisfied.

Downloaded from Arrow.com.

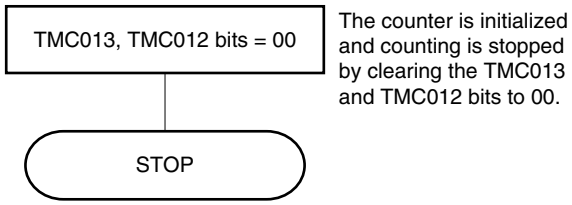
Figure 7-33. Example of Software Processing for PPG Output Operation



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting the TOC01 register. For details, see **7.3 (3) 16-bit timer output control register 01 (TOC01)**.

Remark PPG pulse cycle = (M + 1) × Count clock cycle
PPG duty = (N + 1)/(M + 1)

Downloaded from Arrow.com.

7.4.7 One-shot pulse output operation

A one-shot pulse can be output by setting the TMC01.TMC013 and TMC01.TMC012 bits to 01 (free-running timer mode) or to 10 (clear & start mode entered by the TI010 pin valid edge) and setting the TOC01.OSPE01 bit to 1.

When the TOC01.OSPT01 is set to 1 during timer operation, clearing & starting of the TM01 register is triggered, and a pulse of the difference between the values of the CR010 and CR011 registers is output only once from the TO01 pin.

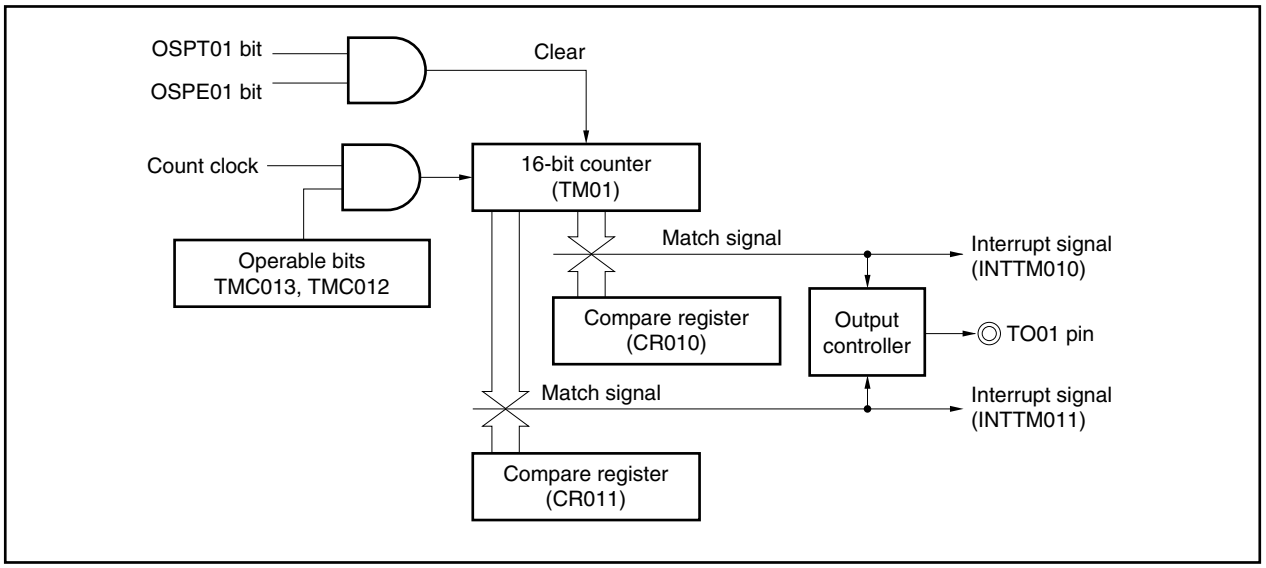
Cautions 1. Do not input the trigger again (setting OSPT01 to 1) while the one-shot pulse is output. To output the one-shot pulse again, generate the trigger after the current one-shot pulse output has completed.

2. Restrictions are applied to TM01 in which the timer input and output share the same pin. For details, refer to 7.6 (1) Alternate functions of TI010/TO01 pins.

Remarks 1. For the alternate-function pin (TI01) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 and INTTM011 interrupts, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

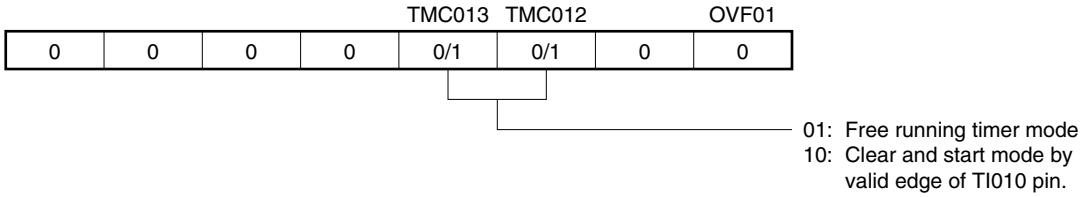
Figure 7-34. Block Diagram of One-Shot Pulse Output Operation



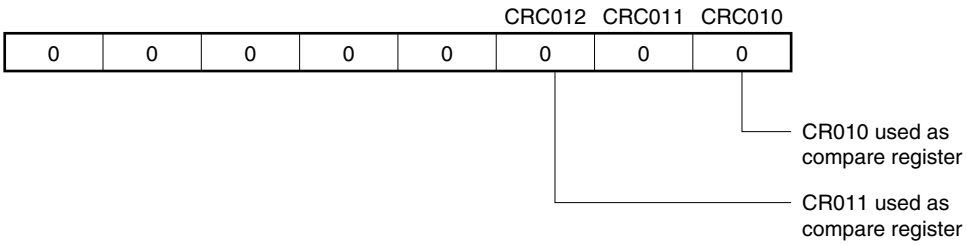
Downloaded from Arrow.com.

Figure 7-35. Example of Register Settings for One-Shot Pulse Output Operation (1/2)

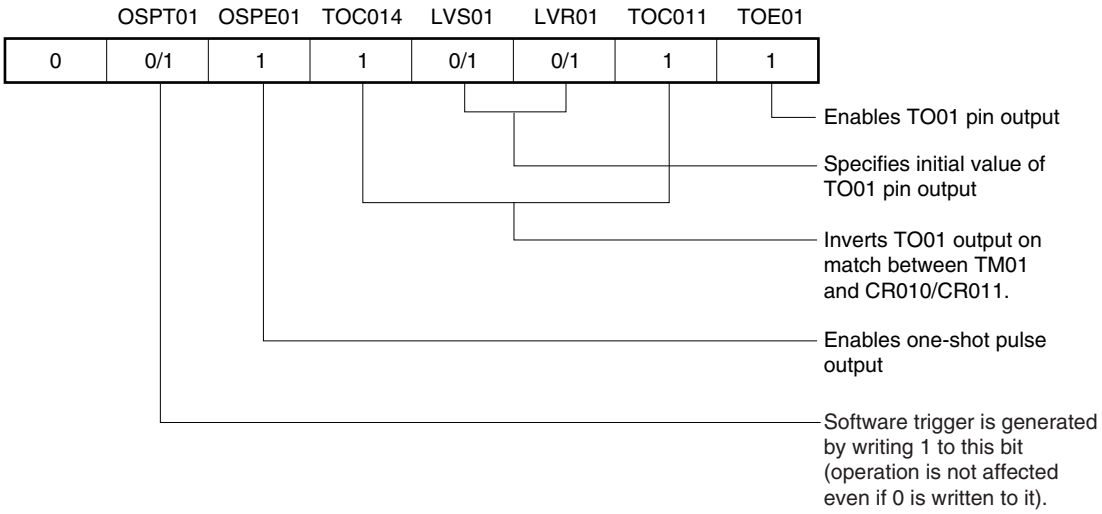
(a) 16-bit timer mode control register 01 (TMC01)



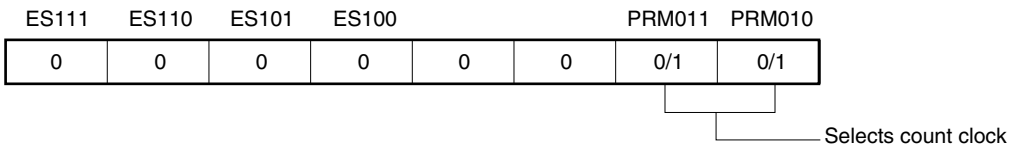
(b) Capture/compare control register 01 (CRC01)



(c) 16-bit timer output control register 01 (TOC01)



(d) Prescaler mode register 01 (PRM01)



Downloaded from Arrow.com.

Figure 7-35. Example of Register Settings for One-Shot Pulse Output Operation (2/2)

(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

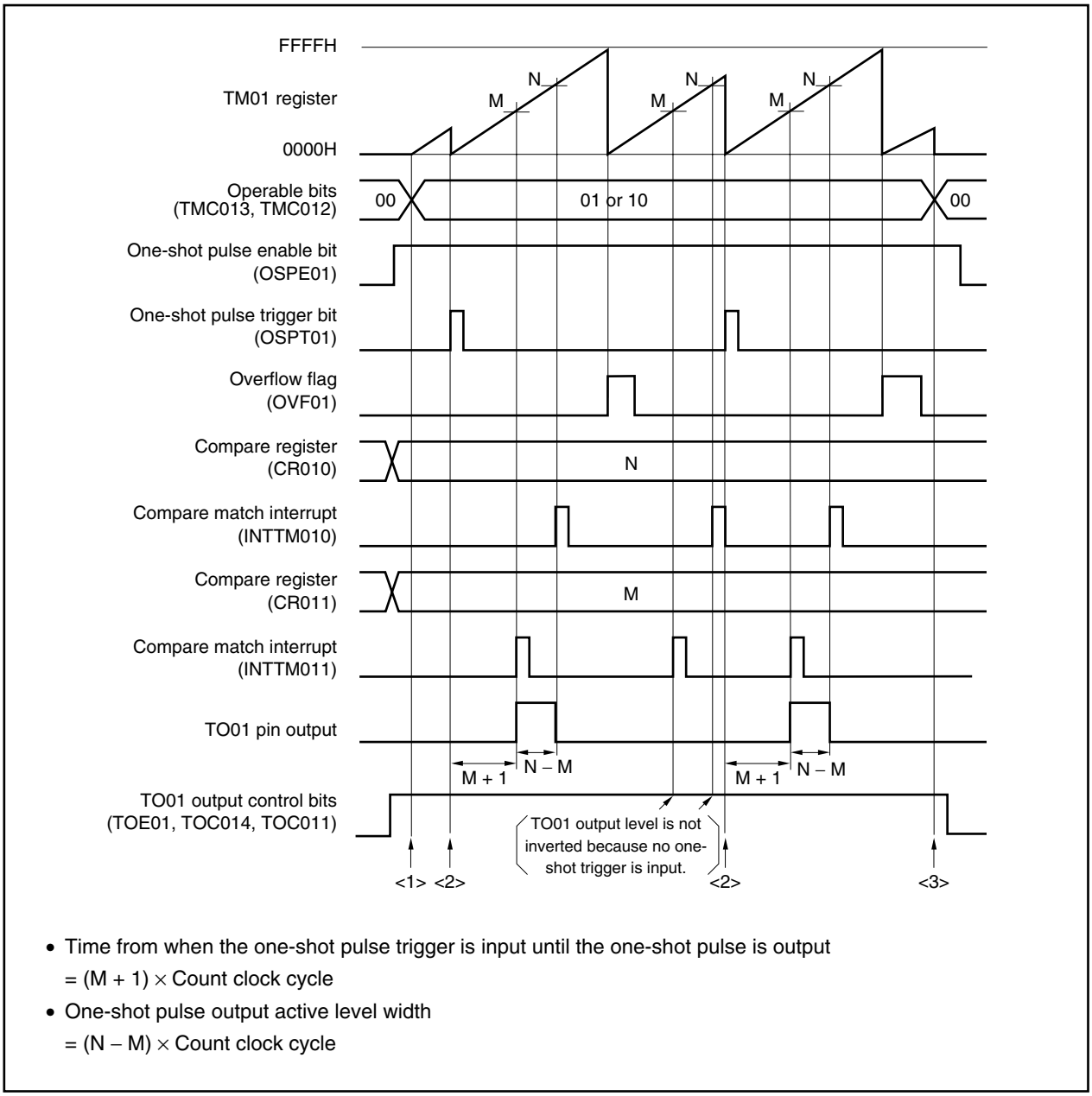
This register is used as a compare register when a one-shot pulse is output. When the value of the TM01 register matches that of the CR010 register, an interrupt signal (INTTM010) is generated and the output level of the TO01 pin is inverted.

(g) 16-bit capture/compare register 011 (CR011)

This register is used as a compare register when a one-shot pulse is output. When the value of the TM01 register matches that of the CR011 register, an interrupt signal (INTTM011) is generated and the output level of the TO01 pin is inverted.

Downloaded from Arrow.com.

Figure 7-36. Example of Software Processing for One-Shot Pulse Output Operation (1/2)

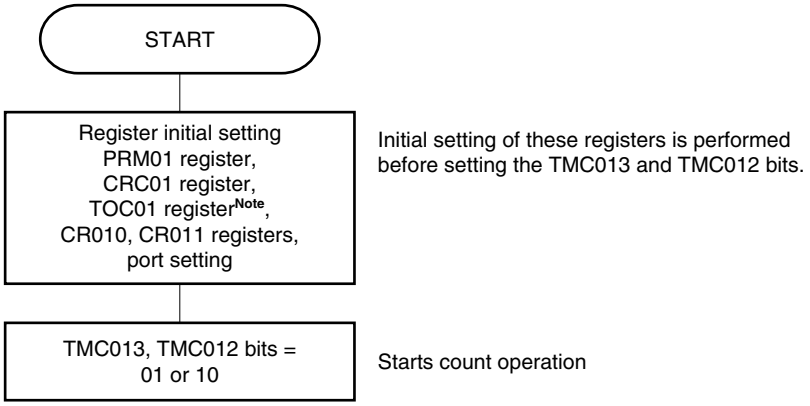


- Time from when the one-shot pulse trigger is input until the one-shot pulse is output
  = (M + 1) × Count clock cycle
- One-shot pulse output active level width
  = (N – M) × Count clock cycle

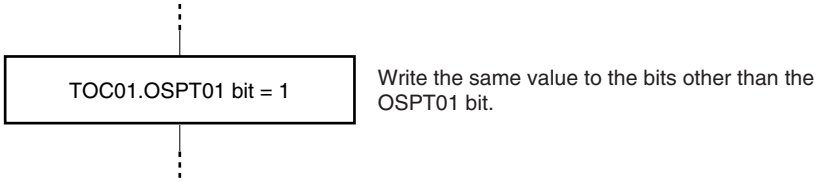
Downloaded from Arrow.com.

Figure 7-36. Example of Software Processing for One-Shot Pulse Output Operation (2/2)

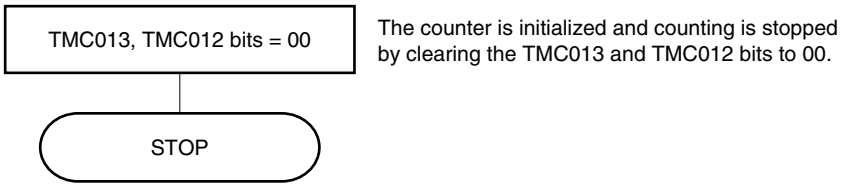
<1> Count operation start flow



<2> One-shot trigger input flow



<3> Count operation stop flow



Note Care must be exercised when setting the TOC01 register. For details, see **7.3 (3) 16-bit timer output control register 01 (TOC01)**.

Downloaded from Arrow.com.

7.4.8 Pulse width measurement operation

The TM01 register can be used to measure the pulse width of the signal input to the TI010 and TI011 pins.

Measurement can be accomplished by operating the 16-bit timer/event counter 01 in the free-running timer mode or by restarting the timer in synchronization with the signal input to the TI010 pin.

When an interrupt is generated, read the value of the valid capture register and measure the pulse width. Check the TMC01.OVF01 flag. If it is set (to 1), clear it to 0 by software.

Figure 7-37. Block Diagram of Pulse Width Measurement (Free-Running Timer Mode)

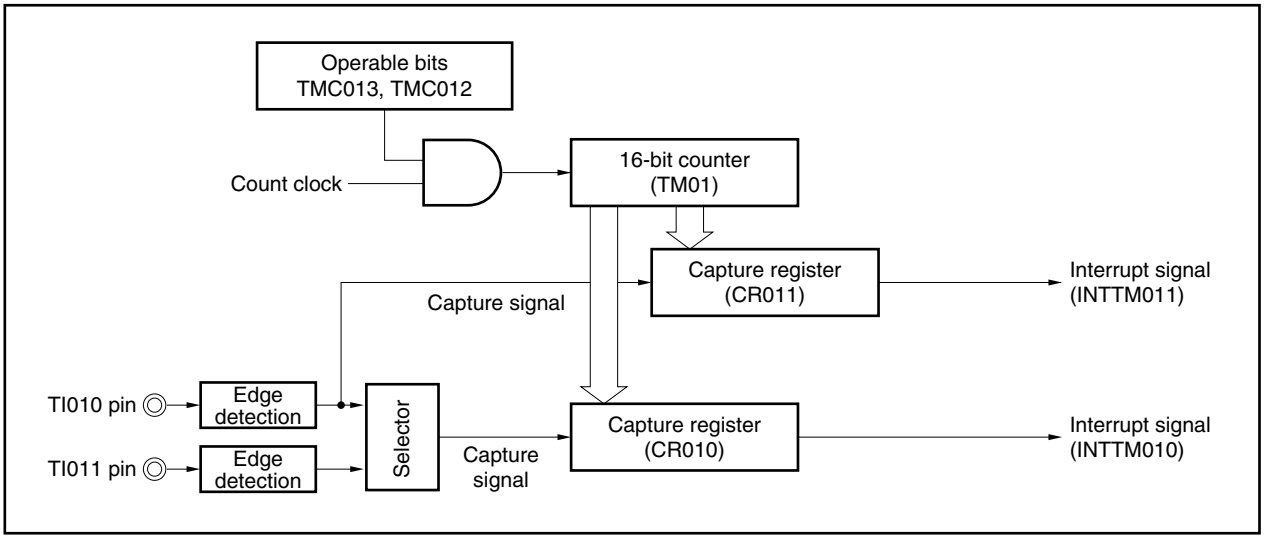
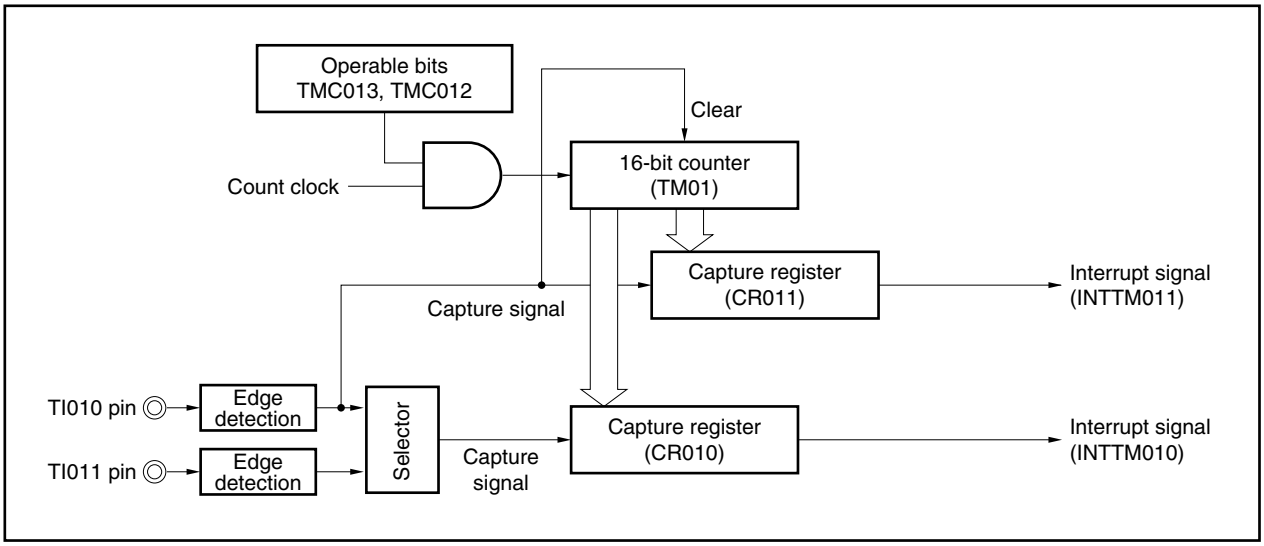


Figure 7-38. Block Diagram of Pulse Width Measurement (Clear & Start Mode Entered by TI010 Pin Valid Edge Input)



Downloaded from Arrow.com.

A pulse width can be measured in the following three ways.

- Measuring the pulse width by using two input signals of the TI010 and TI011 pins (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI010 pin (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI010 pin (clear & start mode entered by the TI010 pin valid edge input)

Remarks 1. For the alternate-function pin (TO01) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For enabling the INTTM010 and INTTM011 interrupts, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

3. Restrictions are applied to TM01. For details, refer to **7.6 (1) Alternate functions of TI010/TO01 pins.**

Downloaded from Arrow.com.

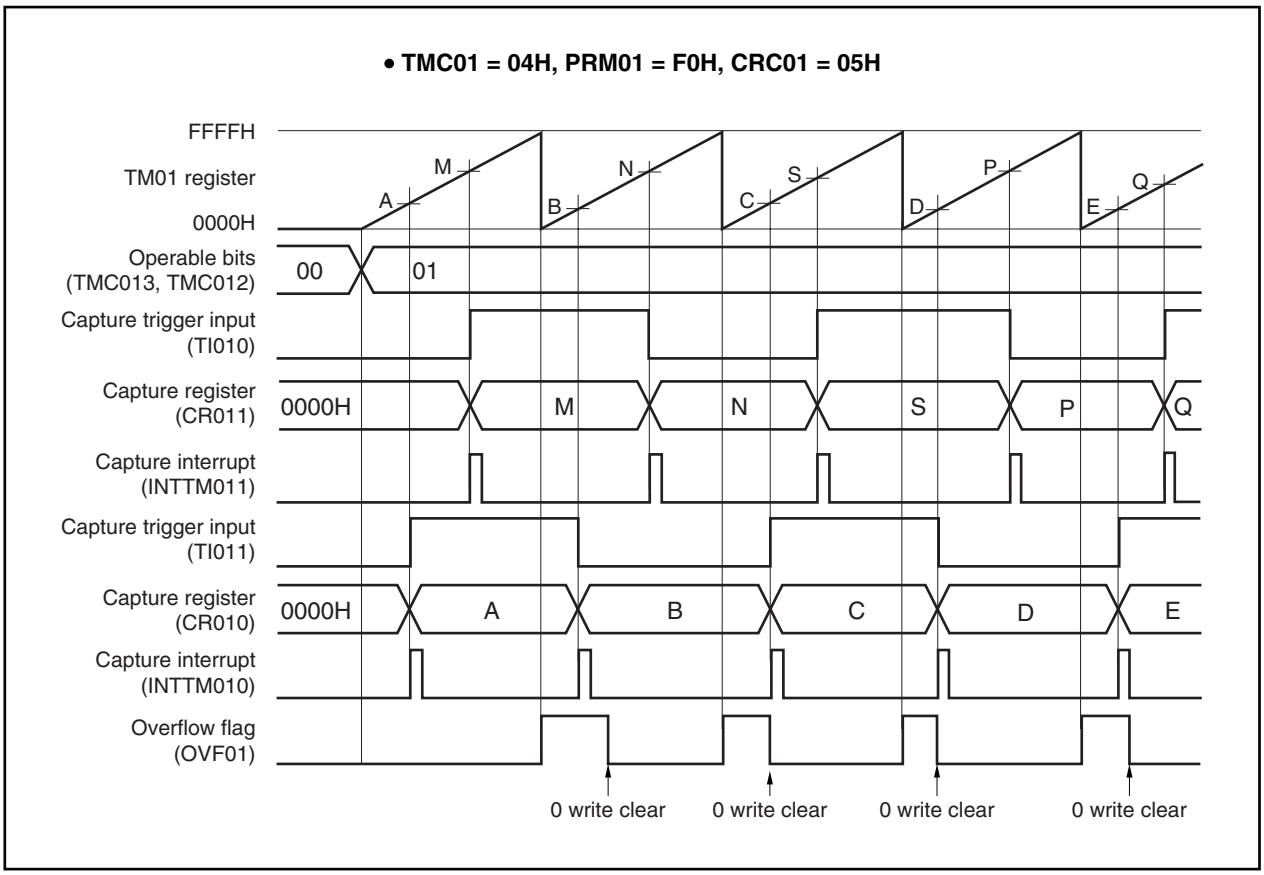
(1) Measuring the pulse width by using two input signals of the TI010 and TI011 pins (free-running timer mode)

Set the free-running timer mode (the TMC01.TMC013 and TMC01.TMC012 bits = 01). When the valid edge of the TI010 pin is detected, the count value of the TM01 register is captured to the CR011 register. When the valid edge of the TI011 pin is detected, the count value of the TM01 register is captured to the CR010 register. Specify detection of both the edges of the TI010 and TI011 pins.

By this measurement method, the previous count value is subtracted from the count value captured by the edge of each input signal. Therefore, save the previously captured value to a separate register in advance.

If an overflow occurs, the value becomes negative if the previously captured value is simply subtracted from the current captured value and, therefore, a borrow occurs (the PSW.CY bit is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear the TMC01.OVF01 bit to 0.

Figure 7-39. Timing Example of Pulse Width Measurement (1)



Downloaded from Arrow.com.

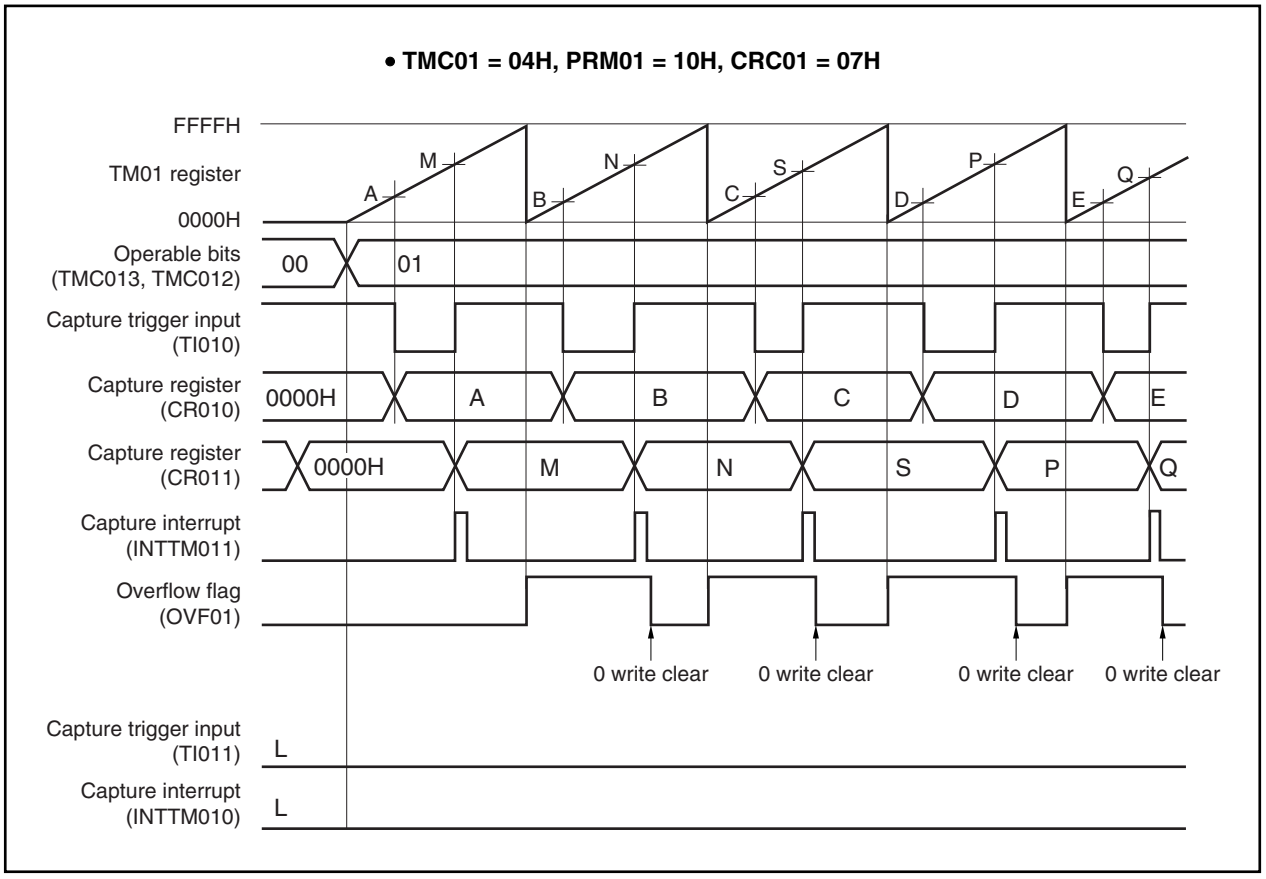
(2) Measuring the pulse width by using one input signal of the TI010 pin (free-running timer mode)

Set the free-running timer mode (the TMC01.TMC013 and TMC01.TMC012 bits = 01). The count value of the TM01 register is captured to the CR010 register in the phase reverse to the valid edge detected on the TI010 pin. When the valid edge of the TI010 pin is detected, the count value of the TM01 register is captured to the CR011 register.

By this measurement method, values are stored in separate capture registers when a width from one edge to another is measured. Therefore, the capture values do not have to be saved. By subtracting the value of one capture register from that of another, a high-level width, low-level width, and cycle are calculated.

If an overflow occurs, the value becomes negative if one captured value is simply subtracted from another and, therefore, a borrow occurs (the PSW.CY bit is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear the TMC01.OVF01 bit to 0.

Figure 7-40. Timing Example of Pulse Width Measurement (2)



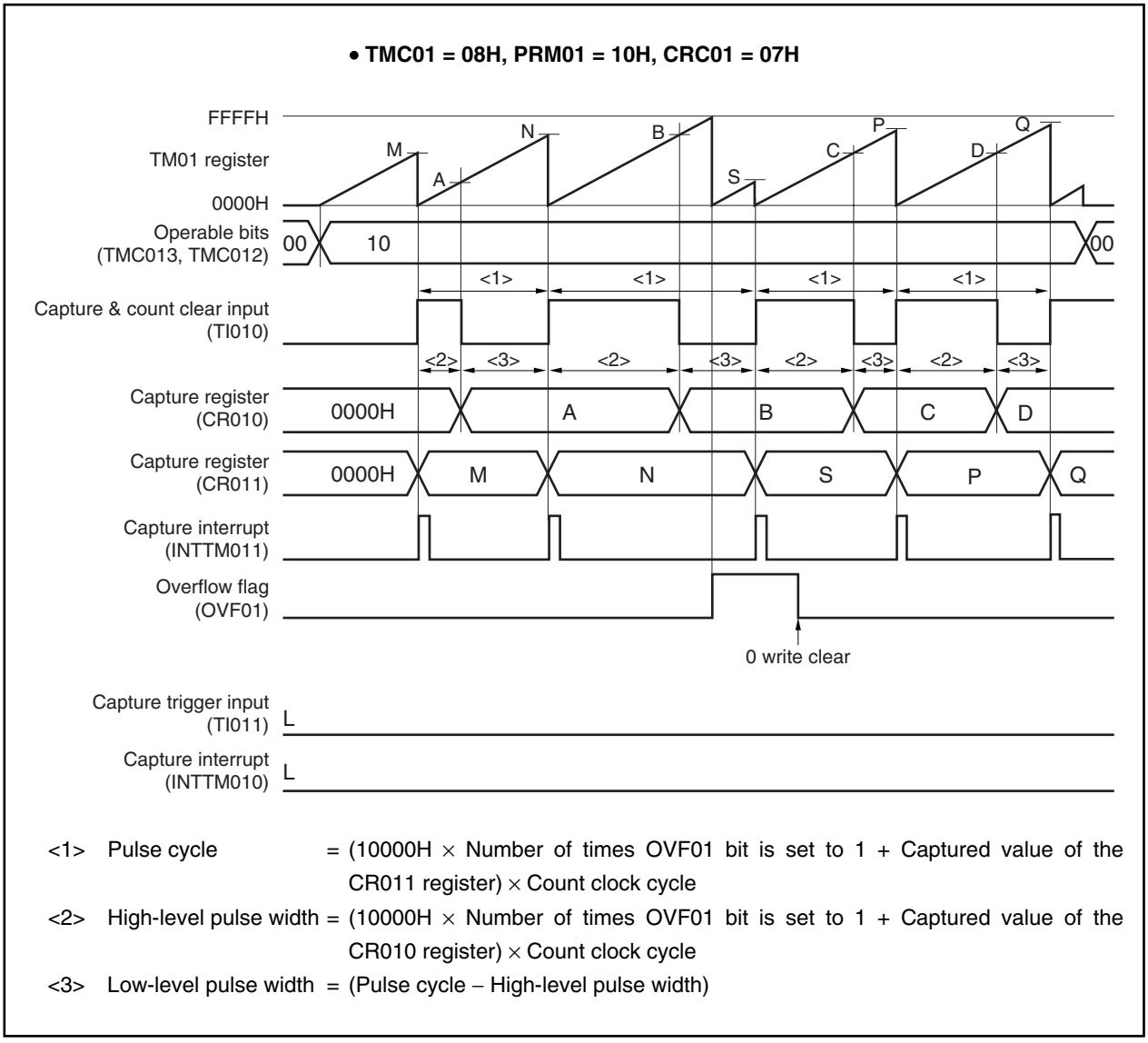
Downloaded from Arrow.com.

(3) Measuring the pulse width by using one input signal of the TI010 pin (clear & start mode entered by the TI010 pin valid edge input)

Set the clear & start mode entered by the TI010 pin valid edge (the TMC01.TMC013 and TMC01.TMC012 bits = 10). The count value of the TM01 register is captured to the CR010 register in the phase reverse to the valid edge of the TI010 pin, and the count value of the TM01 register is captured to the CR011 register and the TM01 register is cleared (0000H) when the valid edge of the TI010 pin is detected. Therefore, a cycle is stored in the CR011 register if the TM01 register does not overflow.

If an overflow occurs, take the value that results from adding 10000H to the value stored in the CR011 register as a cycle. Clear the TMC01.OVF01 bit to 0.

Figure 7-41. Timing Example of Pulse Width Measurement (3)



<1> Pulse cycle = (10000H × Number of times OVF01 bit is set to 1 + Captured value of the CR011 register) × Count clock cycle

<2> High-level pulse width = (10000H × Number of times OVF01 bit is set to 1 + Captured value of the CR010 register) × Count clock cycle

<3> Low-level pulse width = (Pulse cycle − High-level pulse width)

Downloaded from Arrow.com.

Figure 7-42. Example of Register Settings for Pulse Width Measurement Operation (1/2)

(a) 16-bit timer mode control register 01 (TMC01)

				TMC013	TMC012		OVF01
0	0	0	0	0/1	0/1	0	0

01: Free running timer mode
10: Clear and start mode entered by valid edge of TI010 pin.

(b) Capture/compare control register 01 (CRC01)

					CRC012	CRC011	CRC010
0	0	0	0	0	1	0/1	1

CR010 used as capture register

0: TI011 pin is used as capture trigger of CR010.
1: Reverse phase of TI010 pin is used as capture trigger of CR010.

CR011 used as capture register

(c) 16-bit timer output control register 01 (TOC01)

	OSPT01	OSPE01	TOC014	LVS01	LVR01	TOC011	TOE01
0	0	0	0	0	0	0	0

(d) Prescaler mode register 01 (PRM01)

ES111	ES110	ES101	ES100			PRM011	PRM010
0/1	0/1	0/1	0/1	0	0	0/1	0/1

Selects count clock
(setting valid edge of TI010 is prohibited)

00: Falling edge detection
01: Rising edge detection
10: Setting prohibited
11: Both edges detection
(setting when CRC011 = 1 is prohibited)

00: Falling edge detection
01: Rising edge detection
10: Setting prohibited
11: Both edges detection

Downloaded from Arrow.com.

Figure 7-42. Example of Register Settings for Pulse Width Measurement Operation (2/2)

(e) 16-bit timer counter 01 (TM01)

By reading the TM01 register, the count value can be read.

(f) 16-bit capture/compare register 010 (CR010)

This register is used as a capture register. Either the TI010 or TI011 pin is selected as a capture trigger. When a specified edge of the capture trigger is detected, the count value of the TM01 register is stored in the CR010 register.

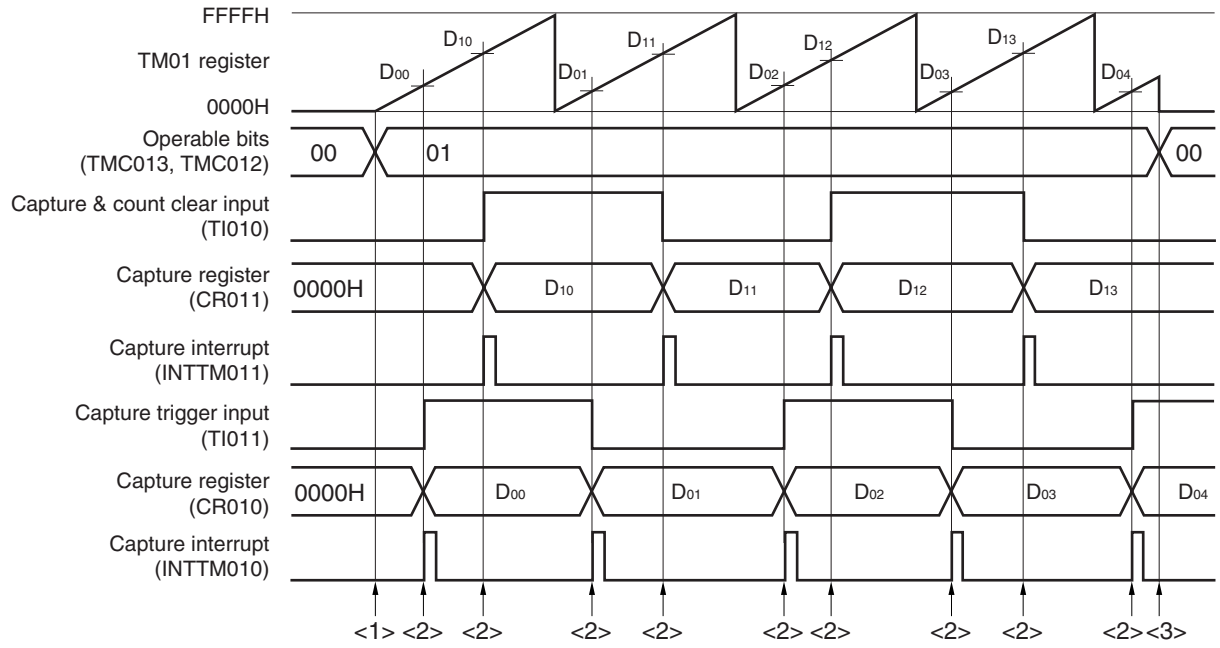
(g) 16-bit capture/compare register 011 (CR011)

This register is used as a capture register. The signal input to the TI010 pin is used as a capture trigger. When the capture trigger is detected, the count value of the TM01 register is stored in the CR011 register.

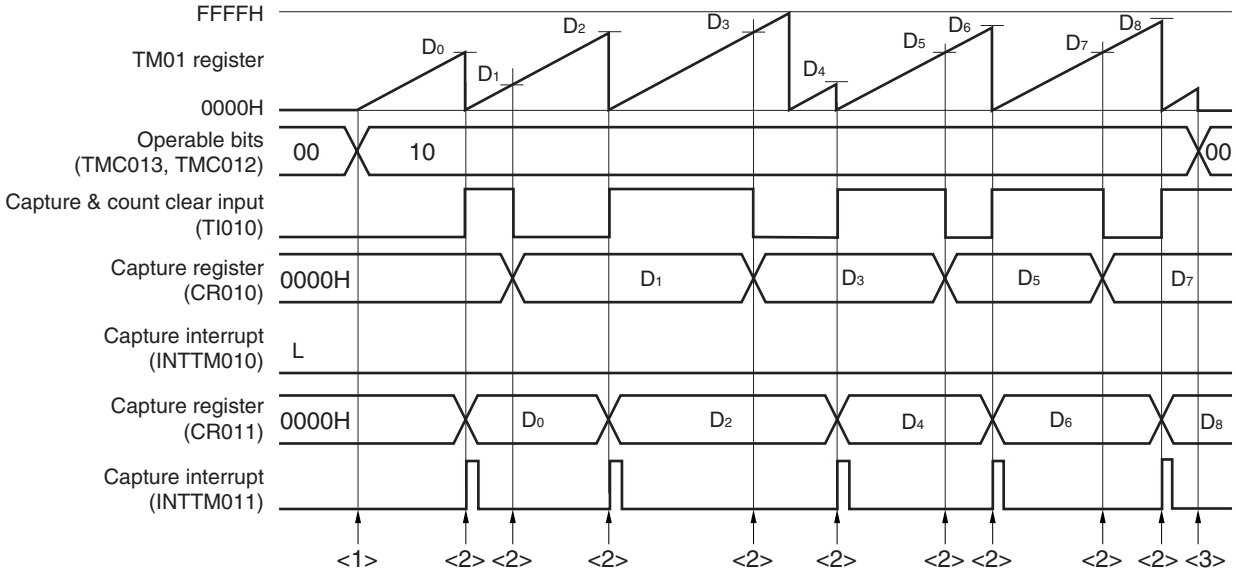
Downloaded from Arrow.com.

Figure 7-43. Example of Software Processing for Pulse Width Measurement (1/2)

(a) Example of free-running timer mode

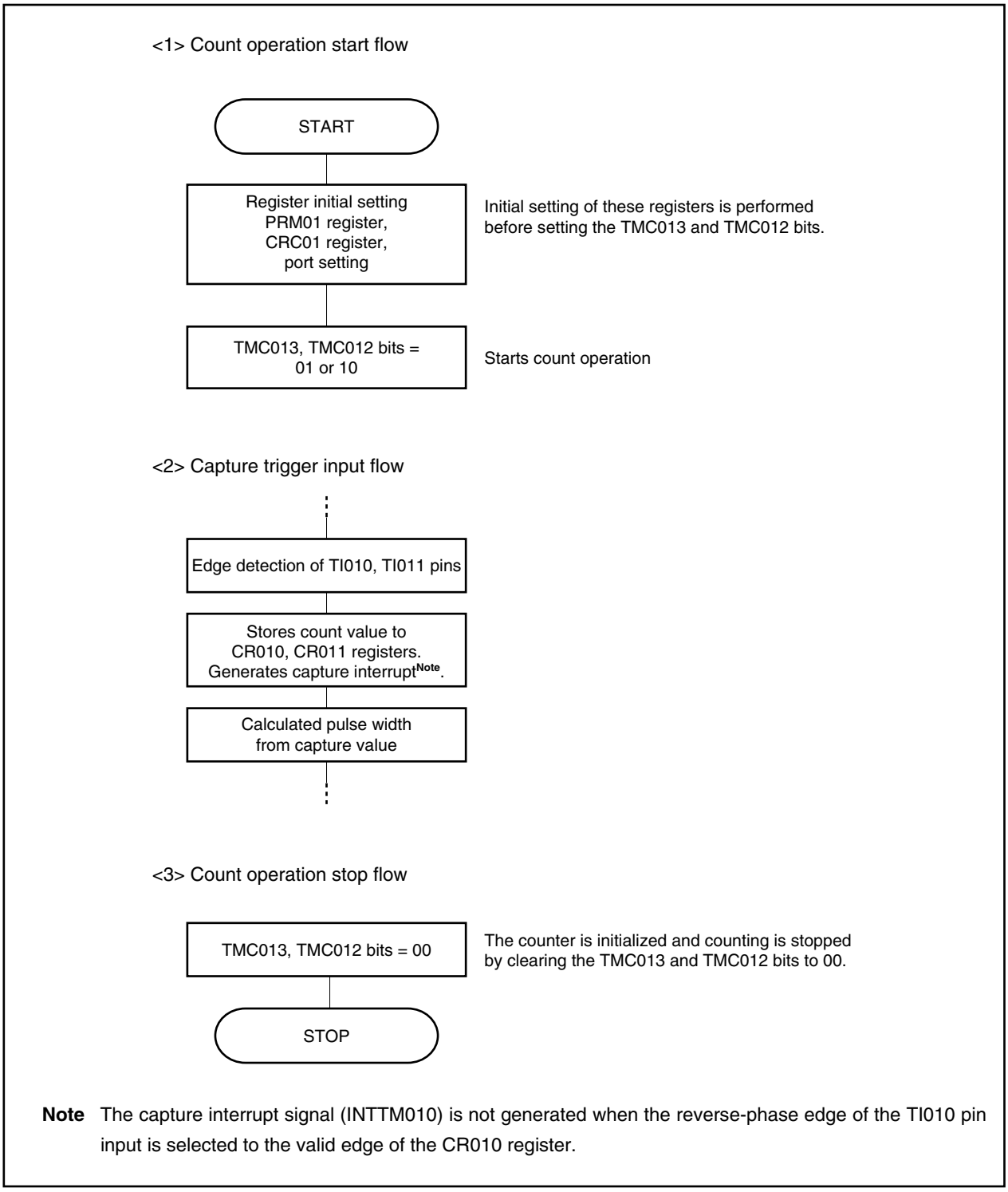


(b) Example of clear & start mode entered by TI010 pin valid edge



Downloaded from Arrow.com.

Figure 7-43. Example of Software Processing for Pulse Width Measurement (2/2)



Note The capture interrupt signal (INTTM010) is not generated when the reverse-phase edge of the TI010 pin input is selected to the valid edge of the CR010 register.

Downloaded from Arrow.com.

7.5 Special Use of TM01

7.5.1 Rewriting CR011 register during TM01 operation

In principle, rewriting the CR010 and CR011 registers of the V850ES/KE1 when they are used as compare registers is prohibited while the TM01 register is operating (TMC01.TMC013 and TMC01.TMC012 bits = other than 00).

However, the value of the CR011 register can be changed, even while the TM01 register is operating, using the following procedure if the CR011 register is used for PPG output and the duty factor is changed (change the value of the CR011 register immediately after its value matches the value of the TM01 register. If the value of the CR011 register is changed immediately before its value matches the TM01 register, an unexpected operation may be performed).

Procedure for changing value of the CR011 register

<1> Disable interrupt INTTM011 (TM0IC10.TM0MK11 bit = 1).
<2> Disable reversal of the timer output when the value of the TM01 register matches that of the CR011 register (TOC01.TOC014 bit = 0).
<3> Change the value of the CR011 register.
<4> Wait for one cycle of the count clock of the TM01 register.
<5> Enable reversal of the timer output when the value of the TM01 register matches that of the CR011 register (TOC01.TOC014 bit = 1).
<6> Clear the interrupt flag of INTTM011 to 0 (TM0IC10.TM0IF11 bit = 0).
<7> Enable interrupt INTTM011 (TM0IC10.TM0MK11 bit = 0).

Remark For the TM0IC10 register, see **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

7.5.2 Setting LVS01 and LVR01 bits

(1) Usage of the LVS01 and LVR01 bits

The TOC01.LVS01 and TOC01.LVR01 bits are used to set the default value of the TO01 pin output and to invert the timer output without enabling the timer operation (TMC01.TMC013 and TMC01.TMC012 bits = 00). Clear the LVS01 and LVR01 bits to 00 (default value: low-level output) when software control is unnecessary.

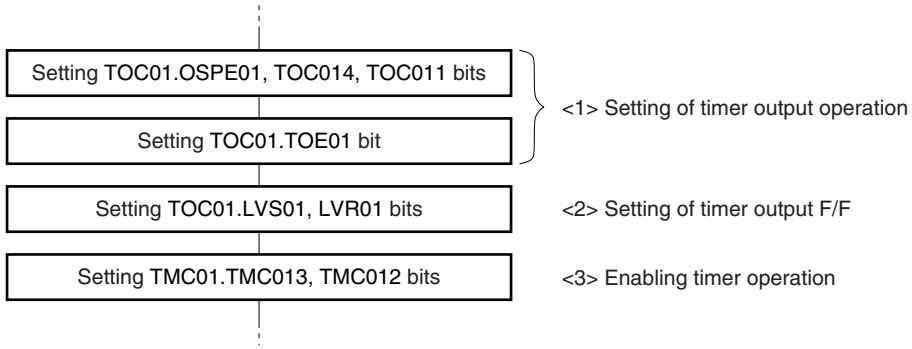
LVS01 Bit	LVR01 Bit	Timer Output Status
0	0	Not changed (low-level output)
0	1	Cleared (low-level output)
1	0	Set (high-level output)
1	1	Setting prohibited

Downloaded from Arrow.com.

(2) Setting the LVS01 and LVR01 bits

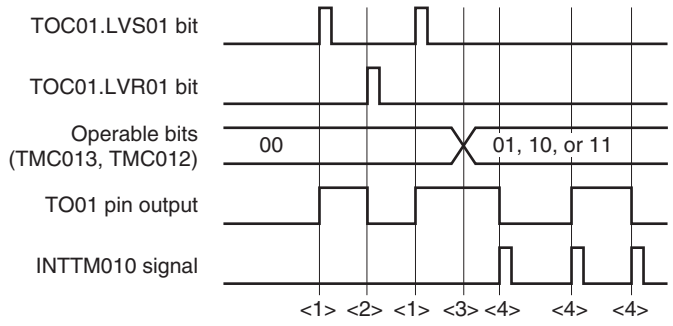
Set the LVS01 and LVR01 bits using the following procedure.

Figure 7-44. Example of Flow for Setting LVS01 and LVR01 Bits



Caution Be sure to set the LVS01 and LVR01 bits following steps <1>, <2>, and <3> above. Step <2> can be performed after <1> and before <3>.

Figure 7-45. Timing Example of LVR01 and LVS01 Bits



<1> The TO01 pin output goes high when the LVS01 and LVR01 bits = 10.

<2> The TO01 pin output goes low when the LVS01 and LVR01 bits = 01 (the pin output remains unchanged from the high level even if the LVS01 and LVR01 bits are cleared to 00).

<3> The timer starts operating when the TMC013 and TMC012 bits are set to 01, 10, or 11. Because the LVS01 and LVR01 bits were set to 10 before the operation was started, the TO01 pin output starts from the high level. After the timer starts operating, setting the LVS01 and LVR01 bits is prohibited until the TMC013 and TMC012 bits = 00 (disabling the timer operation).

<4> The output level of the TO01 pin is inverted each time an interrupt signal (INTTM010) is generated.

Downloaded from Arrow.com.

7.6 Cautions

(1) Alternate functions of TI010/TO01 pins

In the V850ES/KE1, the timer input and output share the same pins shown in Table 7-3.

Restrictions are applied to each channel, as shown in Table 7-4.

Table 7-3. Timer I/O Pins

Channel	Pin	Alternate Function	Remarks
TM01	TI010	P35/TO01	Shares the pin with TO01.
	TI011	P50/KR0/RTP00	–
	TO01	P35/TI010	Shares the pin with TI010.

Table 7-4. Restrictions for Each Channel of 16-bit Timer/Event Counter 0

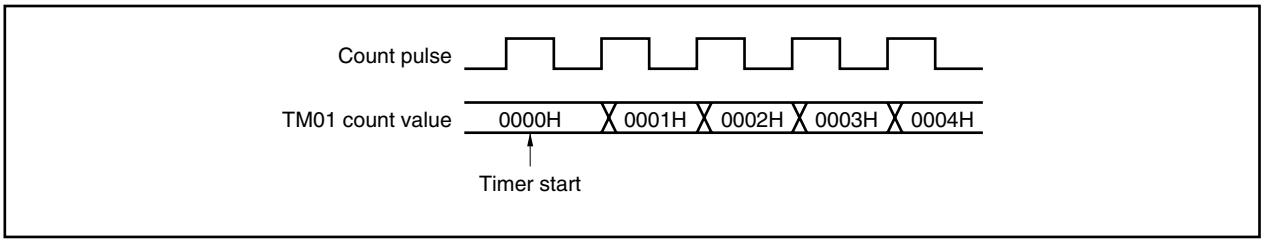
Operation	Restriction
Operation as interval timer	–
Operation as square-wave output	Setting the TI010 valid edge (PRM01.PRM011, PRM01.PRM010 bits = 11) for the count clock is prohibited since the TI010 and TO01 share the same pin (only the internal clock can be used).
Operation as external event counter	Use of the timer output (TO01) is prohibited since the TI010 and TO01 share the same pin. Set the TOC01 register to 00H.
Operation as clear & start mode by valid edge input of TI010 pin	Use of the timer output (TO01) is prohibited since the TI010 and TO01 share the same pin. Set the TOC01 register to 00H.
Operation as free-running timer	Use of the timer output (TO01) is prohibited when TI010 valid edge (PRM01.PRM011, PRM01.PRM010 bits = 11) is set for the count clock, since the TI010 and TO01 share the same pin. When the timer output (TO01) is used, setting the TI010 valid edge (PRM01.PRM011, PRM01.PRM010 bits = 11) for the count clock is prohibited (only the internal clock can be used).
Operation as PPG output	Setting the TI010 valid edge (PRM01.PRM011, PRM01.PRM010 bits = 11) for the count clock is prohibited since the TI010 and TO01 share the same pin (only the internal clock can be used).
Operation as one-shot pulse output	Use of the hardware trigger by the TI010 pin valid edge is prohibited. The software trigger (TOC01.OSPT01 bit) can be used.
Operation as pulse width measurement	Use of the timer output (TO01) is prohibited since the TI010 and TO01 share the same pin. Set the TOC01 register to 00H.

Downloaded from Arrow.com.

(2) Error on starting timer

An error of up to 1 clock occurs before the match signal is generated after the timer has been started. This is because the count of the TM01 register is started asynchronously to the count pulse.

Figure 7-46. Count Start Timing of TM01 Register



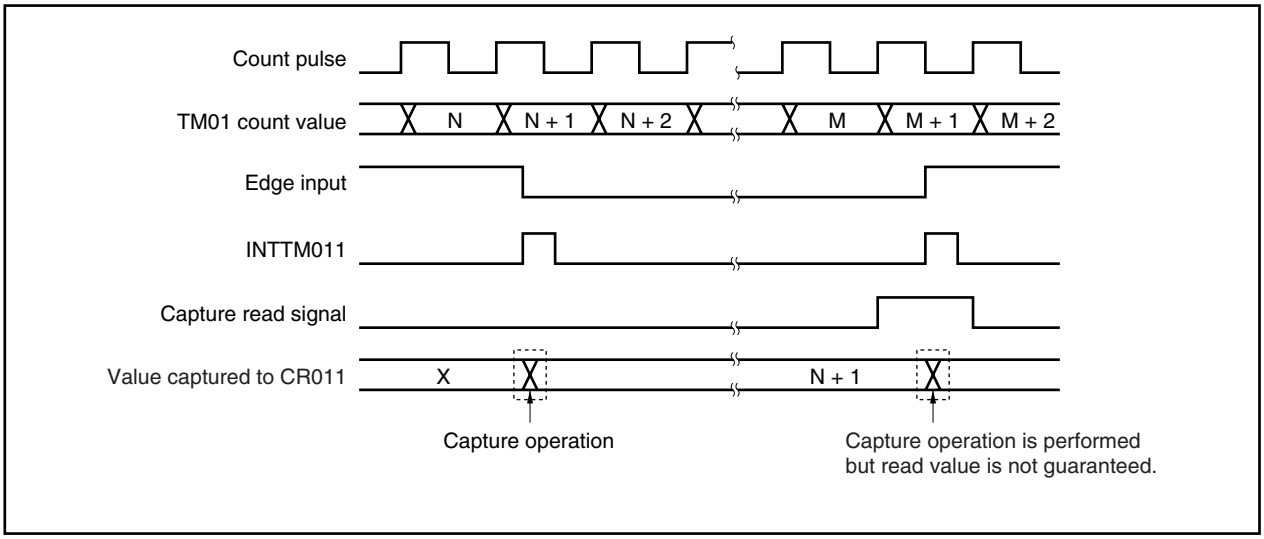
(3) Setting CR010 and CR011 registers (in the mode in which clear & start occurs upon match between TM01 register and CR010 register)

Set the CR010 and CR011 registers to a value other than 0000H (when using these registers as external event counters, one-pulse count operation is not possible).

(4) Data hold timing of capture register

(a) If the valid edge of the TI011/TI010 pin is input while the CR010/CR011 register is read, the CR010/CR011 register performs capture operation, but the read value at this time is not guaranteed. However, the interrupt request signal (INTTM010/INTTM011) is generated as a result of detection of the valid edge.

Figure 7-47. Data Hold Timing of Capture Register



(b) The values of the CR010 and CR011 registers are not guaranteed after 16-bit timer/event counter 01 has stopped.

Downloaded from Arrow.com.

(5) Setting valid edge

Set the valid edge of the TI010 pin while the timer operation is stopped (TMC01.TMC013 and TMC01.TMC012 bits = 00). Set the valid edge by using the PRM01.ES100 and PRM01.ES101 bits.

(6) Re-triggering one-shot pulse

Make sure that the trigger is not generated while an active level is being output in the one-shot pulse output mode. Be sure to input the next trigger after the current active level is output.

(7) Operation of OVF01 flag

(a) Setting of OVF01 flag

The TMC01.OVF01 flag is set to 1 in the following case in addition to when the TM01 register overflows.

Select the mode in which clear & start occurs upon match between the TM01 register and the CR010 register.

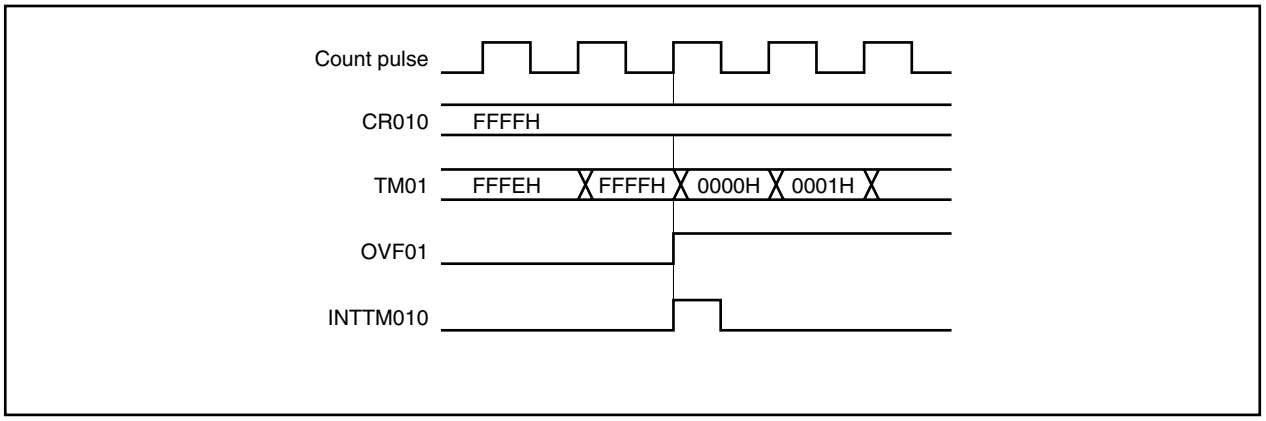
↓

Set the CR010 register to FFFFH

↓

When the TM01 register is cleared from FFFFH to 0000H upon match with the CR010 register

Figure 7-48. Operation Timing of OVF01 Flag



(b) Clearing of OVF01 flag

After the TM01 register overflows, clearing OVF01 flag is invalid and set (1) again even if the OVF01 flag is cleared (0) before the next count clock is counted (before TM01 register becomes 0001H).

(8) One-shot pulse output

One-shot pulse output operates normally in either the free-running timer mode or the mode in which clear & start occurs on the valid edge of the TI010 pin. In the mode in which clear & start occurs upon match between the TM01 register and the CR010 register, one-shot pulse output is not possible.

Downloaded from Arrow.com.

(9) Capture operation

(a) If valid edge of TI010 pin is specified for count clock

If the valid edge of the TI010 pin is specified for the count clock, the capture register that specified the TI010 pin as the trigger does not operate normally.

(b) To ensure that signals input from TI011 and TI010 pins are correctly captured

To accurately capture the count value, the pulse input to the TI010 and TI011 pins as a capture trigger must be wider than two count clocks selected by the PRM01 register.

(c) Interrupt signal generation

Although a capture operation is performed at the falling edge of the count clock, an interrupt request signal (INTTM010, INTTM011) is generated at the rising edge of the next count clock.

(d) Caution when CRC01.CRC011 bit is set to 1

When the count value of the TM01 register is captured to the CR010 register in the phase reverse to the signal input to the TI010 pin, the interrupt request signal (INTTM010) is not generated after the count value is captured. If the valid edge is detected on the TI011 pin during this operation, the capture operation is not performed but the INTTM010 signal is generated as an external interrupt signal. Mask the INTTM010 signal when the external interrupt is not used.

(10) Edge detection

(a) Specifying valid edge after reset

If the operation of the 16-bit timer/event counter 01 is enabled after reset and while the TI010 or TI011 pin is at high level and when the rising edge or both the edges are specified as the valid edge of the TI010 or TI011 pin, then the high level of the TI010 or TI011 pin is detected as the rising edge. Note this when the TI010 or TI011 pin is pulled up. However, the rising edge is not detected when the operation is once stopped and then enabled again.

(b) Sampling clock for noise elimination

The sampling clock for noise elimination differs depending on whether the valid edge of TI010 is used for the count clock or as a capture trigger. In the former case, sampling is performed using $f_{XX}/4$, and in the latter case, sampling is performed using the count clock selected by the PRM01 register.

When the signal input to the TI010 pin is sampled and the valid level is detected two times in a row, the valid edge is detected. Therefore, noise having a short pulse width can be eliminated.

Remark f_{XX}: Main clock frequency

Downloaded from Arrow.com.

CHAPTER 8 8-BIT TIMER/EVENT COUNTER 5

In the V850ES/KE1, two channels of 8-bit timer/event counter 5 are provided.

8.1 Functions

8-bit timer/event counter 5n has the following two modes (n = 0, 1).

- Mode using 8-bit timer/event counter alone (individual mode)
- Mode using cascade connection (16-bit resolution: cascade connection mode)

These two modes are described below.

(1) Mode using 8-bit timer/event counter alone (individual mode)

8-bit timer/event counter 5n operates as an 8-bit timer/event counter.
The following functions can be used.

- Interval timer
- External event counter
- Square-wave output
- PWM output

(2) Mode using cascade connection (16-bit resolution: cascade connection mode)

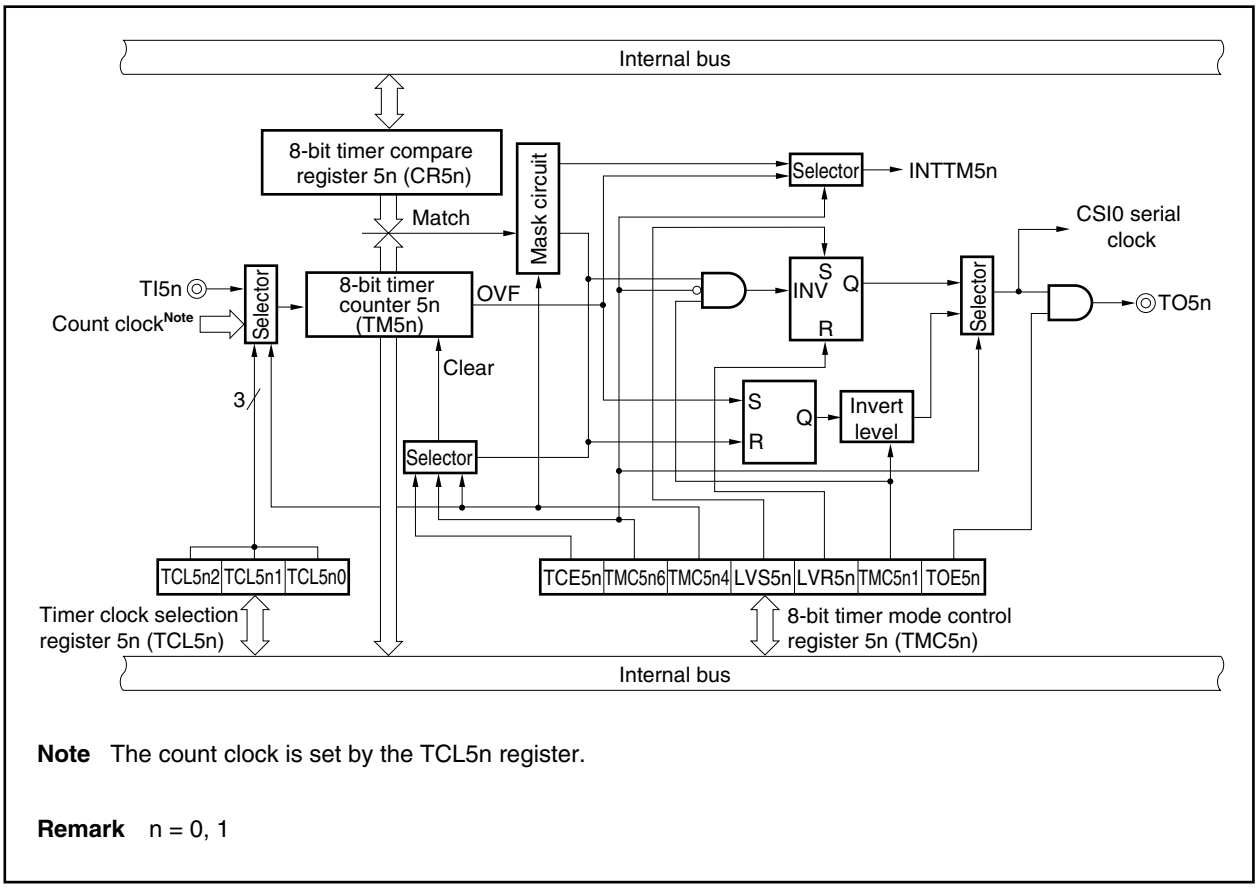
8-bit timer/event counter 5n operates as a 16-bit timer/event counter by connecting the TM5n register in cascade. The following functions can be used.

- Interval timer with 16-bit resolution
- External event counter with 16-bit resolution
- Square-wave output with 16-bit resolution

The block diagram of 8-bit timer/event counter 5n is shown next.

Downloaded from Arrow.com.

Figure 8-1. Block Diagram of 8-bit Timer/Event Counter 5n



Note The count clock is set by the TCL5n register.

Remark n = 0, 1

8.2 Configuration

8-bit timer/event counter 5n includes the following hardware.

Table 8-1. Configuration of 8-bit Timer/Event Counter 5n

Item	Configuration
Timer registers	8-bit timer counter 5n (TM5n) 16-bit timer counter 5 (TM5): Only when using cascade connection
Registers	8-bit timer compare register 5n (CR5n) 16-bit timer compare register 5 (CR5): Only when using cascade connection
Timer output	1 (TO5n pin)
Control registers Note	Timer clock selection register 5n (TCL5n) 8-bit timer mode control register 5n (TMC5n) 16-bit timer mode control register 5 (TMC5): Only when using cascade connection

Note When using the functions of the TI5n and TO5n pins, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

Remark n = 0, 1

Downloaded from Arrow.com.

(1) 8-bit timer counter 5n (TM5n)

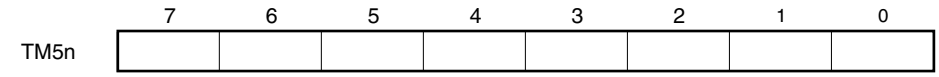
The TM5n register is an 8-bit read-only register that counts the count pulses.

The counter is incremented in synchronization with the rising edge of the count clock.

Through cascade connection, the TM5n registers can be used as a 16-bit timer.

When using the TM50 register and the TM51 register in cascade as a 16-bit timer, these registers can be read only in 16-bit units. Therefore, read these registers twice and compare the values, taking into consideration that the reading occurs during a count change.

After reset: 00H R Address: TM50 FFFFF5C0H, TM51 FFFFF5C1H



TM5n

(n = 0, 1)

The count value is reset to 00H in the following cases.

<1> Reset

<2> When the TMC5n.TCE5n bit is cleared (0)

<3> The TM5n register and CR5n register match in the mode in which clear & start occurs on a match between the TM5n register and the CR5n register

Caution When connected in cascade, these registers become 0000H even when the TCE50 bit in the lowest timer (TM50) is cleared.

Remark n = 0, 1

Downloaded from Arrow.com.

(2) 8-bit timer compare register 5n (CR5n)

The CR5n register can be read and written in 8-bit units.

In a mode other than the PWM mode, the value set to the CR5n register is always compared to the count value of the TM5n register, and if the two values match, an interrupt request signal (INTTM5n) is generated.

In the PWM mode, TM5n register overflow causes the TO5n pin output to change to the active level, and when the values of the TM5n register and the CR5n register match, the TO5n pin output changes to the inactive level.

The value of the CR5n register can be set in the range of 00H to FFH.

When using the TM50 register and TM51 register in cascade as a 16-bit timer, the CR50 register and CR51 register operate as 16-bit timer compare register 5 (CR5). The counter value and register value are compared in 16-bit lengths, and if they match, an interrupt request signal (INTTM50) is generated.

After reset: 00H R/W Address: CR50 FFFFF5C2H, CR51 FFFFF5C3H

	7	6	5	4	3	2	1	0
CR5n								

(n = 0, 1)

Cautions 1. In the mode in which clear & start occurs upon a match of the TM5n register and CR5n register (TMC5n.TMC5n6 bit = 0), do not write a different value to the CR5n register during the count operation.

2. In the PWM mode, set the CR5n register rewrite interval to three or more count clocks (clock selected with the TCL5n register).

3. Before changing the value of the CR5n register when using a cascade connection, be sure to stop the timer operation.

Remark n = 0, 1

Downloaded from Arrow.com.

8.3 Registers

The following two registers are used to control 8-bit timer/event counter 5n.

- Timer clock selection register 5n (TCL5n)
- 8-bit timer mode control register 5n (TMC5n)

Remark To use the functions of the TI5n and TO5n pins, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

(1) Timer clock selection register 5n (TCL5n)

The TCL5n register sets the count clock of 8-bit timer/event counter 5n and the valid edge of the TI5n pin input.

The TCL5n register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TCL50 FFFFF5C4H, TCL51 FFFFF5C5H

	7	6	5	4	3	2	1	0
TCL5n	0	0	0	0	0	TCL5n2	TCL5n1	TCL5n0

(n = 0, 1)

TCL5n2	TCL5n1	TCL5n0	Count clock selection[Note]		
			Clock	fxx	
				20 MHz	10 MHz
0	0	0	Falling edge of TI5n	–	–
0	0	1	Rising edge of TI5n	–	–
0	1	0	fxx	Setting prohibited	100 ns
0	1	1	fxx/2	100 ns	200 ns
1	0	0	fxx/4	200 ns	0.4 μs
1	0	1	fxx/64	3.2 μs	6.4 μs
1	1	0	fxx/256	12.8 μs	25.6 μs
1	1	1	INTTM010	–	–

Note When the internal clock is selected, set so as to satisfy the following conditions.

V_{DD} = 4.0 to 5.5 V: Count clock ≤ 10 MHz

V_{DD} = 2.7 to 4.0 V: Count clock ≤ 5 MHz

Caution Before overwriting the TCL5n register with different data, stop the timer operation.

Remark When the TM5n register is connected in cascade, the TCL51 register settings are invalid.

Downloaded from Arrow.com.

(2) 8-bit timer mode control register 5n (TMC5n)

The TMC5n register performs the following six settings.

- Controls counting by the TM5n register
- Selects the operation mode of the TM5n register
- Selects the individual mode or cascade connection mode
- Sets the status of the timer output flip-flop
- Controls the timer output flip-flop or selects the active level in the PWM (free-running timer) mode
- Controls timer output

The TMC5n register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Downloaded from Arrow.com.

After reset: 00H R/W Address: TMC50 FFFFF5C6H, TMC51 FFFFF5C7H

	<7>	6	5	4	<3>	<2>	1	<0>
TMC5n	TCE5n	TMC5n6	0	TMC514[Note]	LVS5n	LVR5n	TMC5n1	TOE5n

(n = 0, 1)

TCE5n	Control of count operation of 8-bit timer/event counter 5n
0	Counting is disabled after the counter is cleared to 0 (counter disabled)
1	Start count operation

TMC5n6	Selection of operation mode of 8-bit timer/event counter 5n
0	Mode in which clear & start occurs on match between TM5n register and CR5n register
1	PWM (free-running timer) mode

TMC514	Selection of individual mode or cascade connection mode for 8-bit timer/event counter 51
0	Individual mode
1	Cascade connection mode (connected with 8-bit timer/event counter 50)

LVS5n	LVR5n	Setting of status of timer output F/F
0	0	Unchanged
0	1	Reset timer output F/F to 0
1	0	Set timer output F/F to 1
1	1	Setting prohibited

TMC5n1	Other than PWM (free-running timer) mode (TMC5n6 bit = 0)	PWM (free-running timer) mode (TMC5n6 bit = 1)
	Controls timer F/F	Selects active level
0	Disable inversion operation	High active
1	Enable inversion operation	Low active

TOE5n	Timer output control
0	Disable output (TO5n pin is low level)
1	Enable output

Note Bit 4 of the TMC50 register is fixed to 0.

Cautions 1. Because the TO51 and TI51 pins are alternate functions of the same pin, only one can be used at one time.

2. The LVS5n and LVR5n bit settings are valid in modes other than the PWM mode.

3. Do not set <1> to <4> below at the same time. Set as follows.

<1> Set the TMC5n1, TMC5n6, and TMC514[Note] bits: Setting of operation mode

<2> Set the TOE5n bit for timer output enable: Timer output enable

<3> Set the LVS5n and LVR5n bits (Caution 2): Setting of timer output F/F

<4> Set the TCE5n bit

Remarks 1. In the PWM mode, the PWM output is set to the inactive level by the TCE5n bit = 0.

2. When the LVS5n and LVR5n bits are read, 0 is read.

3. The values of the TMC5n6, LVS5n, LVR5n, TMC5n1, and TOE5n bits are reflected to the TO5n output regardless of the TCE5n bit value.

Downloaded from Arrow.com.

8.4 Operation

8.4.1 Operation as interval timer

8-bit timer/event counter 5n operates as an interval timer that repeatedly generates interrupts at the interval of the count value preset in the CR5n register. If the count value in the TM5n register matches the value set in the CR5n register, the value of the TM5n register is cleared to 00H and counting is continued, and at the same time, an interrupt request signal (INTTM5n) is generated.

Setting method

<1> Set each register.
- TCL5n register: Selects the count clock (t).
- CR5n register: Compare value (N)
- TMC5n register: Stops count operation and selects the mode in which clear & start occurs on a match between the TM5n register and CR5n register (TMC5n register = 0000xx00B, ×: don't care).

<2> When the TMC5n.TCE5n bit is set to 1, the count operation starts.

<3> When the values of the TM5n register and CR5n register match, the INTTM5n signal is generated (TM5n register is cleared to 00H).

<4> Then, the INTTM5n signal is repeatedly generated at the same interval. To stop counting, set the TCE5n bit = 0.

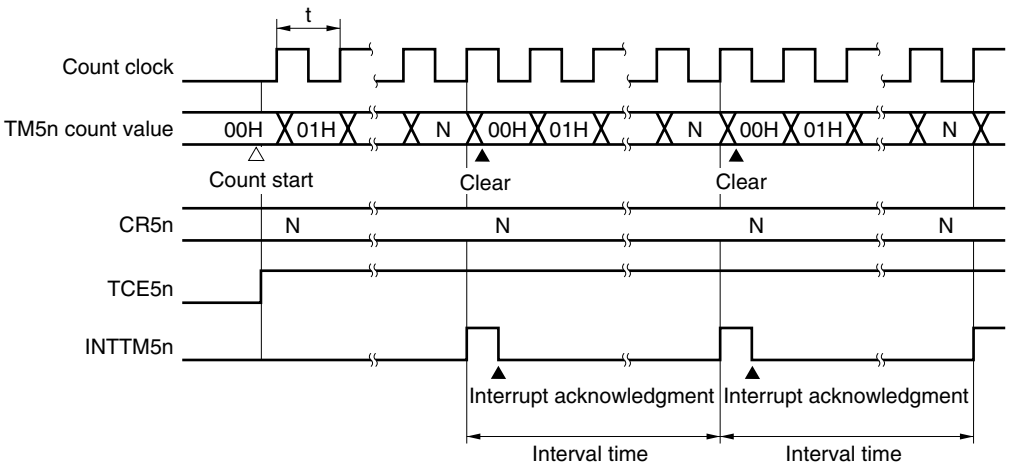
Interval time = (N + 1) × t: N = 00H to FFH

Caution During interval timer operation, do not rewrite the value of the CR5n register.

Remark n = 0, 1

Figure 8-2. Timing of Interval Timer Operation (1/2)

Basic operation

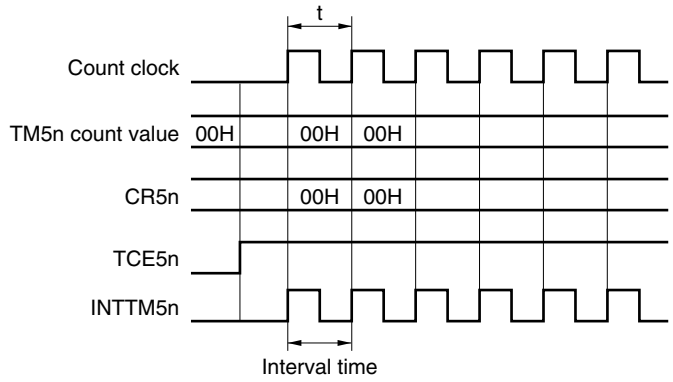


Remark n = 0, 1

Downloaded from Arrow.com.

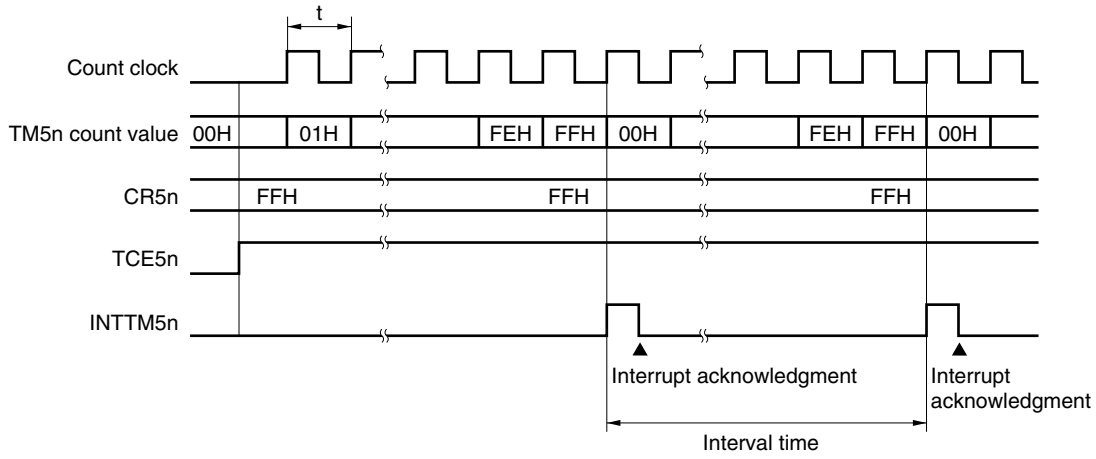
Figure 8-2. Timing of Interval Timer Operation (2/2)

When CR5n register = 00H



Remark n = 0, 1

When CR5n register = FFH



Remark n = 0, 1

Downloaded from Arrow.com.

8.4.2 Operation as external event counter

The external event counter counts the number of clock pulses input to the TI5n pin from an external source by using the TM5n register.

Each time the valid edge specified by the TCL5n register is input to the TI5n pin, the TM5n register is incremented. Either the rising edge or the falling edge can be specified as the valid edge.

When the count value of the TM5n register matches the value of the CR5n register, the TM5n register is cleared to 00H and an interrupt request signal (INTTM5n) is generated.

Setting method

<1> Set each register.

- TCL5n register: Selects the TI5n pin input edge.
  Falling edge of TI5n pin → TLC5n register = 00H
  Rising edge of TI5n pin → TCL5n register = 01H
- CR5n register: Compare value (N)
- TMC5n register: Stops count operation, selects the mode in which clear & start occurs on a match between the TM5n register and CR5n register, disables timer output F/F inversion operation, and disables timer output.
  (TMC5n register = 0000xx00B, ×: don't care)
- For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions.**

<2> When the TMC5n.TCE5n bit is set to 1, the counter counts the number of pulses input from the TI5n pin.

<3> When the values of the TM5n register and CR5n register match, the INTTM5n signal is generated (TM5n register is cleared to 00H).

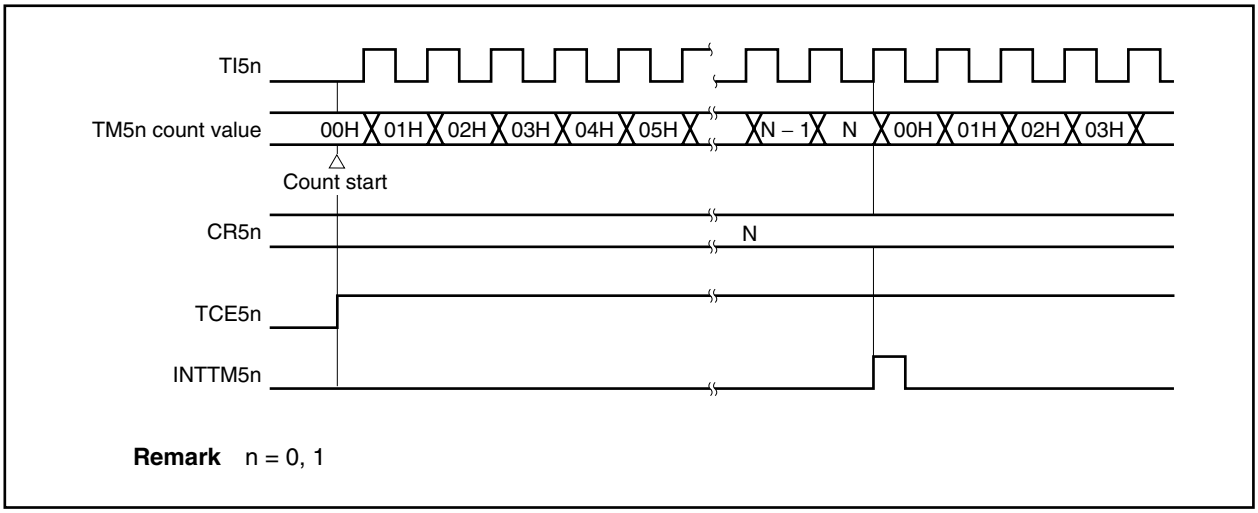
<4> Then, the INTTM5n signal is generated each time the values of the TM5n register and CR5n register match.

INTTM5n signal is generated when the valid edge of TI5n pin is input N + 1 times: N = 00H to FFH

Caution During external event counter operation, do not rewrite the value of the CR5n register.

Remark n = 0, 1

Figure 8-3. Timing of External Event Counter Operation (with Rising Edge Specified)



Remark n = 0, 1

Downloaded from Arrow.com.

8.4.3 Square-wave output operation

A square wave with any frequency can be output at an interval determined by the value preset in the CR5n register.

By setting the TMC5n.TOE5n bit to 1, the output status of the TO5n pin is inverted at an interval determined by the count value preset in the CR5n register. In this way, a square wave of any frequency can be output (duty = 50%) (n = 0, 1).

Setting method

<1> Set each register.

- TCL5n register: Selects the count clock (t).
- CR5n register: Compare value (N)
- TMC5n register: Stops count operation, selects the mode in which clear & start occurs on a match between the TM5n register and CR5n register, sets initial value of timer output, enables timer output F/F inversion operation, and enables timer output.
  (TMC5n register = 00001011B or 00000111B)
- For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions.**

<2> When the TMC5n.TCE5n bit is set to 1, counting starts.

<3> When the values of the TM5n register and CR5n register match, the timer output F/F is inverted. Moreover, the INTTM5n signal is generated and the TM5n register is cleared to 00H.

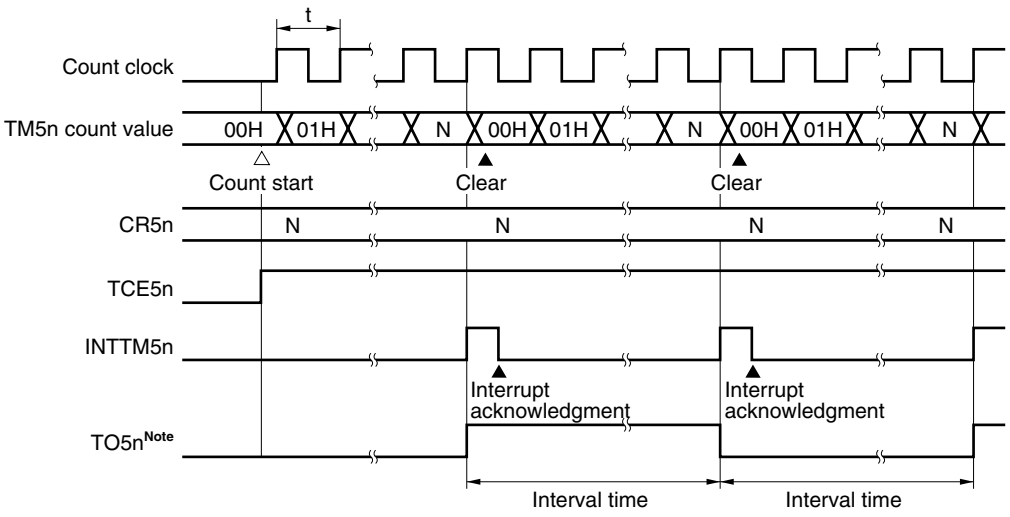
<4> Then, the timer output F/F is inverted during the same interval and a square wave is output from the TO5n pin.

Frequency = 1/2t(N + 1): N = 00H to FFH

Caution Do not rewrite the value of the CR5n register during square-wave output.

Downloaded from Arrow.com.

Figure 8-4. Timing of Square-Wave Output Operation



Note The initial value of the TO5n pin output can be set using the TMC5n.LVS5n and TMC5n.LVR5n bits.

Remark n = 0, 1

Downloaded from Arrow.com.

8.4.4 8-bit PWM output operation

By setting the TMC5n.TMC5n6 bit to 1, 8-bit timer/event counter 5n performs PWM output.

Pulses with a duty factor determined by the value set in the CR5n register are output from the TO5n pin.

Set the width of the active level of the PWM pulse in the CR5n register. The active level can be selected using the TMC5n.TMC5n1 bit.

The count clock can be selected using the TCL5n register.

PWM output can be enabled/disabled by the TMC5n.TOE5n bit.

Caution The CR5n register rewrite interval must be three or more operation clocks (set by the TCL5n register).

Use method

<1> Set each register.

- TCL5n register: Selects the count clock (t).
- CR5n register: Compare value (N)
- TMC5n register: Stops count operation, selects PWM mode, and leave timer output F/F unchanged, sets active level, and enables timer output.
  (TMC5n register = 01000001B or 01000011B)
- For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions.**

<2> When the TMC5n.TCE5n bit is set to 1, counting starts.

PWM output operation

<1> When counting starts, PWM output (output from the TO5n pin) outputs the inactive level until an overflow occurs.

<2> When an overflow occurs, the active level set by setting method <1> is output. The active level is output until the value of the CR5n register and the count value of the TM5n register match. An interrupt request signal (INTTM5n) is generated.

<3> When the value of the CR5n register and the count value of the TM5n register match, the inactive level is output and continues to be output until an overflow occurs again.

<4> Then, steps <2> and <3> are repeated until counting is stopped.

<5> When counting is stopped by clearing TCE5n bit to 0, PWM output becomes inactive.

Cycle = 256t, active level width = Nt, duty = N/256: N = 00H to FFH

Remarks 1. n = 0, 1

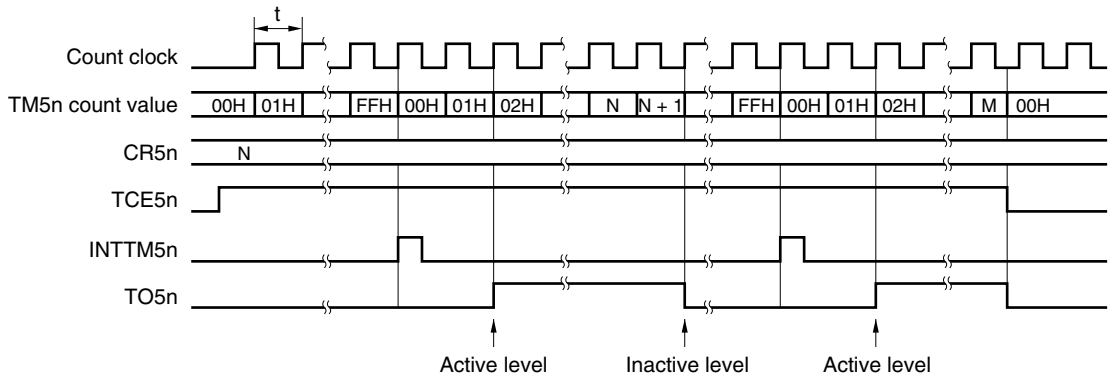
2. For the detailed timing, refer to **Figure 8-5 Timing of PWM Output Operation** and **Figure 8-6 Timing of Operation Based on CR5n Register Transitions**.

Downloaded from Arrow.com.

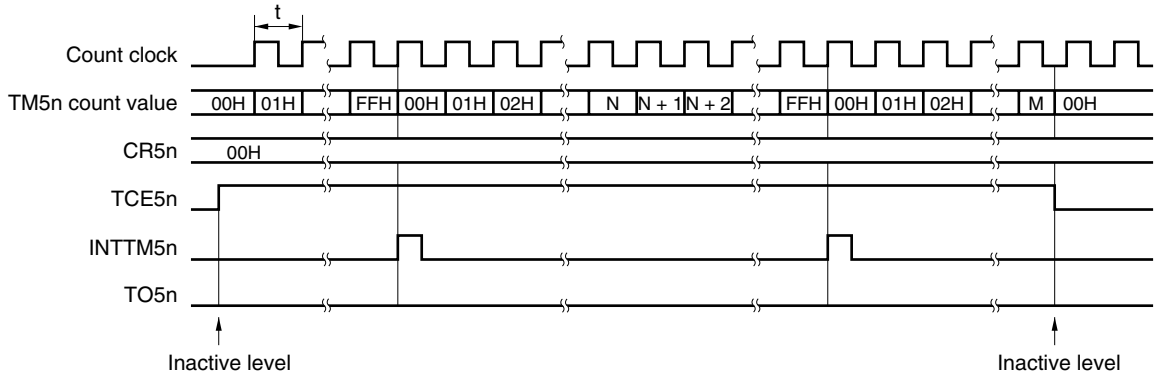
(a) Basic operation of PWM output

Figure 8-5. Timing of PWM Output Operation

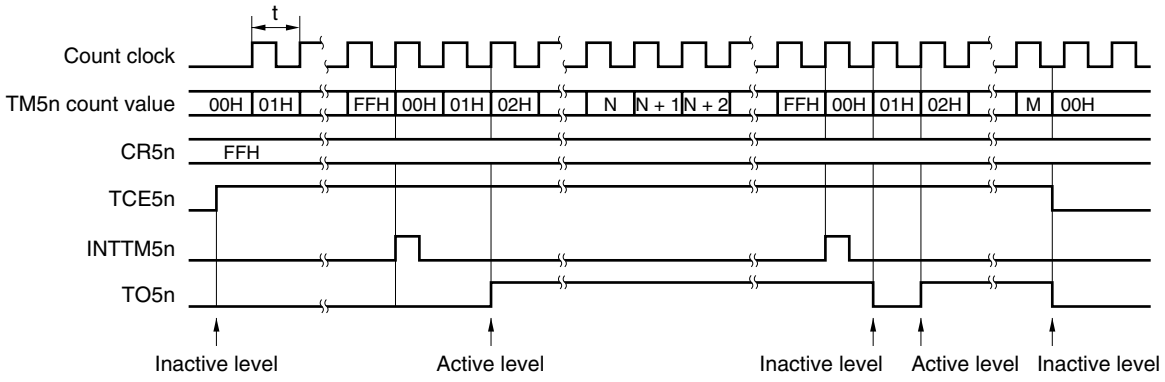
Basic operation (active level = H)



When CR5n register = 00H



When CR5n register = FFH



Remark n = 0, 1

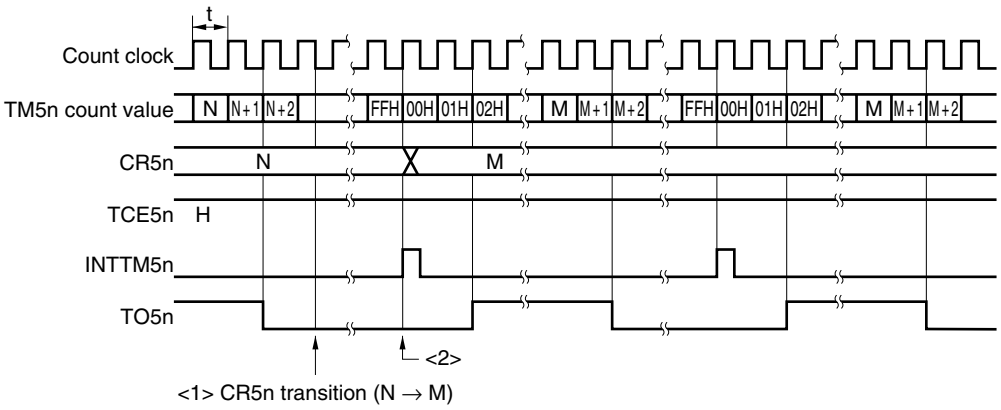
Downloaded from Arrow.com.

(b) Operation based on CR5n register transitions

Figure 8-6. Timing of Operation Based on CR5n Register Transitions

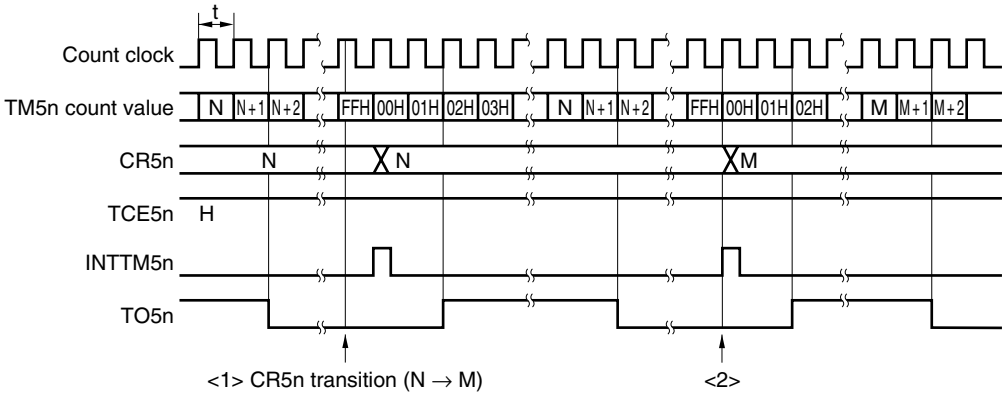
When the value of the CR5n register changes from N to M before the rising edge of the FFH clock

→ The value of the CR5n register is transferred at the overflow that occurs immediately after.



When the value of the CR5n register changes from N to M after the rising edge of the FFH clock

→ The value of the CR5n register is transferred at the second overflow.



Caution In the case of reload from the CR5n register between <1> and <2>, the value that is actually used differs (Read value: M; Actual value of CR5n register: N).

Remark n = 0, 1

Downloaded from Arrow.com.

8.4.5 Operation as interval timer (16 bits)

The 16-bit resolution timer/event counter mode is selected by setting the TMC51.TMC514 bit to 1.

8-bit timer/event counter 5n operates as an interval timer by repeatedly generating interrupts using the count value preset in 16-bit timer compare register 5 (CR5) as the interval.

Setting method

<1> Set each register.

- TCL50 register: Selects the count clock (t)
  (The TCL51 register does not need to be set in cascade connection)
- CR50 register: Compare value (N) ... Lower 8 bits (settable from 00H to FFH)
- CR51 register: Compare value (N) ... Higher 8 bits (settable from 00H to FFH)
- TMC50, TMC51 register: Selects the mode in which clear & start occurs on a match between TM5 register and CR5 register (×: don't care)

  (TMC50 register = 0000xx00B
  TMC51 register = 0001xx00B)

<2> Set the TMC51.TCE51 bit to 1. Then set the TMC50.TCE50 bit to 1 to start the count operation.

<3> When the values of the TM5 register and CR5 register connected in cascade match, the INTTM50 signal is generated (the TM5 register is cleared to 0000H).

<4> The INTTM50 signal is then generated repeatedly at the same interval.

Interval time = (N + 1) × t: N = 0000H to FFFFH

Cautions 1. To write using 8-bit access during cascade connection, set the TCE51 bit to 1 at operation start and then set the TCE50 bit to 1. When operation is stopped, clear the TCE50 bit to 0 and then clear the TCE51 bit to 0.

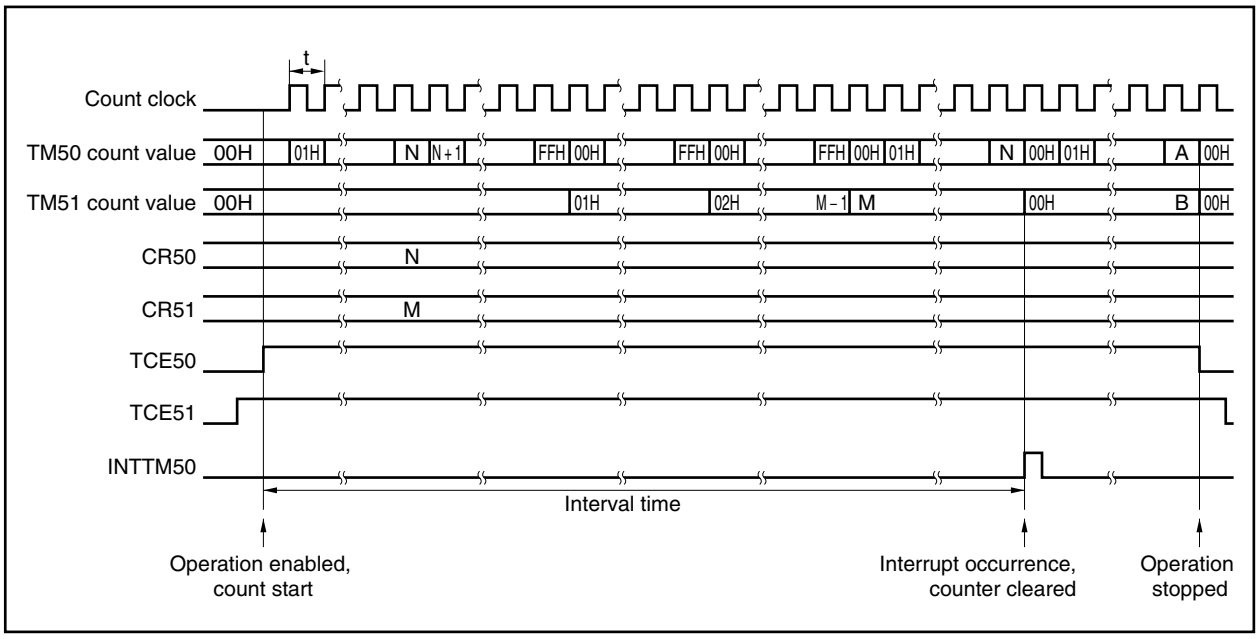
2. During cascade connection, TI50 input, TO50 output, and the INTTM50 signal are used. Do not use TI51 input, TO51 output, and the INTTM51 signal; mask them instead (for details, refer to CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION). Clear the LVS51, LVR51, TMC511, and TOE51 bits to 0.

3. Do not change the value of the CR5 register during timer operation.

Downloaded from Arrow.com.

Figure 8-7 shows a timing example of the cascade connection mode with 16-bit resolution.

Figure 8-7. Cascade Connection Mode with 16-bit Resolution



Downloaded from Arrow.com.

8.4.6 Operation as external event counter (16 bits)

The 16-bit resolution timer/event counter mode is selected by setting the TMC51.TMC514 bit to 1.

The external event counter counts the number of clock pulses input to the TI50 pin from an external source using 16-bit timer counter 5 (TM5).

Setting method

<1> Set each register.

- TCL50 register: Selects the TI50 pin input edge.
  (The TCL51 register does not have to be set during cascade connection.)
  Falling edge of TI50 pin → TCL50 register = 00H
  Rising edge of TI50 pin → TCL50 register = 01H
- CR50 register: Compare value (N) ... Lower 8 bits (settable from 00H to FFH)
- CR51 register: Compare value (N) ... Higher 8 bits (settable from 00H to FFH)
- TMC50, TMC51 registers: Stops count operation, selects the clear & stop mode entered on a match between the TM5 register and CR5 register, disables timer output F/F inversion, and disables timer output.
  (×: don't care)
  (TMC50 register = 0000xx00B
  TMC51 register = 0001xx00B)
- For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions.**

<2> Set the TMC51.TCE51 bit to 1. Then set the TMC50.TCE50 bit to 1 and count the number of pulses input from the TI50 pin.

<3> When the values of the TM5 register and CR5 register connected in cascade match, the INTTM50 signal is generated (the TM5 register is cleared to 0000H).

<4> The INTTM50 signal is then generated each time the values of the TM5 register and CR5 register match.

INTTM50 signal is generated when the valid edge of TI50 pin is input N + 1 times: N = 0000H to FFFFH

Cautions 1. During external event counter operation, do not rewrite the value of the CR5n register.

2. To write using 8-bit access during cascade connection, set the TCE51 bit to 1 and then set the TCE50 bit to 1. When operation is stopped, clear the TCE50 bit to 0 and then clear the TCE51 bit to 0 (n = 0, 1).

3. During cascade connection, TI50 input and the INTTM50 signal are used. Do not use TI51 input, TO51 output, and the INTTM51 signal; mask them instead (for details, refer to CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION). Clear the LVS51, LVR51, TMC511, and TOE51 bits to 0.

4. Do not change the value of the CR5 register during external event counter operation.

Downloaded from Arrow.com.

8.4.7 Square-wave output operation (16-bit resolution)

The 16-bit resolution timer/event counter mode is selected by setting the TMC51.TMC514 bit to 1.

8-bit timer/event counter 5n outputs a square wave of any frequency using the interval preset in 16-bit timer compare register 5 (CR5).

Setting method

<1> Set each register.

- TCL50 register: Selects the count clock (t)
  (The TCL51 register does not have to be set in cascade connection)
- CR50 register: Compare value (N) ... Lower 8 bits (settable from 00H to FFH)
- CR51 register: Compare value (N) ... Higher 8 bits (settable from 00H to FFH)
- TMC50, TCM51 registers: Stops count operation, selects the mode in which clear & start occurs on a match between the TM5 register and CR5 register.

LVS50	LVR50	Timer Output F/F Status Settings
1	0	High-level output
0	1	Low-level output

Enables timer output F/F inversion, and enables timer output.
(TMC50 register = 00001011B or 00000111B
TMC51 register = 00010000B)

- For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions.**

<2> Set the TMC51.TCE51 bit to 1. Then set the TMC50.TCE50 bit to 1 to start the count operation.

<3> When the values of the TM5 register and the CR5 register connected in cascade match, the TO50 timer output F/F is inverted. Moreover, the INTTM50 signal is generated and the TM5 register is cleared to 0000H.

<4> Then, the timer output F/F is inverted during the same interval and a square wave is output from the TO50 pin.

Frequency = 1/2t(N + 1): N = 0000H to FFFFH

Caution Do not write a different value to the CR5 register during operation.

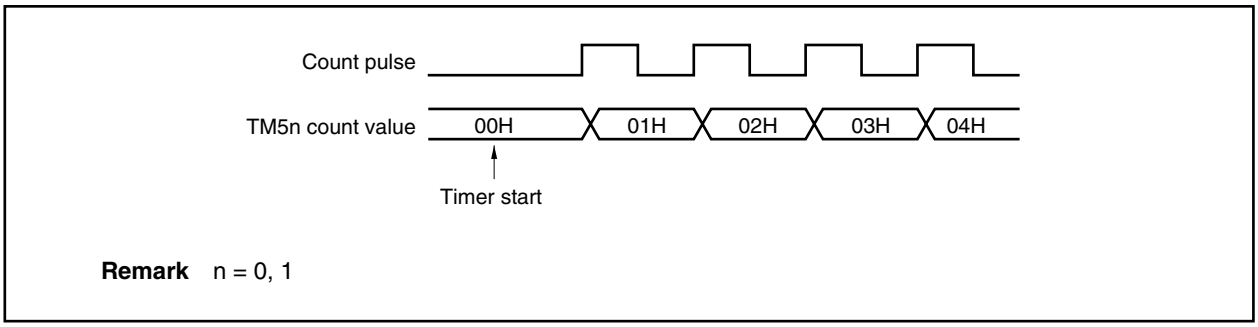
Downloaded from Arrow.com.

8.4.8 Cautions

(1) Error on starting timer

An error of up to 1 clock occurs before the match signal is generated after the timer has been started. This is because the TM5n register is started asynchronously to the count pulse.

Figure 8-8. Count Start Timing of TM5n Register



Remark n = 0, 1

Downloaded from Arrow.com.

CHAPTER 9 8-BIT TIMER H

In the V850ES/KE1, two channels of 8-bit timer H are provided.

9.1 Functions

8-bit timer Hn has the following functions (n = 0, 1).

- Interval timer
- Square wave output
- PWM output
- Carrier generator

9.2 Configuration

8-bit timer Hn includes the following hardware.

Table 9-1. Configuration of 8-bit Timer Hn

Item	Configuration
Timer registers	8-bit timer counter Hn: 1 each
Register	8-bit timer H compare register n0 (CMPn0): 1 each 8-bit timer H compare register n1 (CMPn1): 1 each
Timer outputs	TOHn, output controller
Control registers[Note]	8-bit timer H mode register n (TMHMDn) 8-bit timer H carrier control register n (TMCYCn)

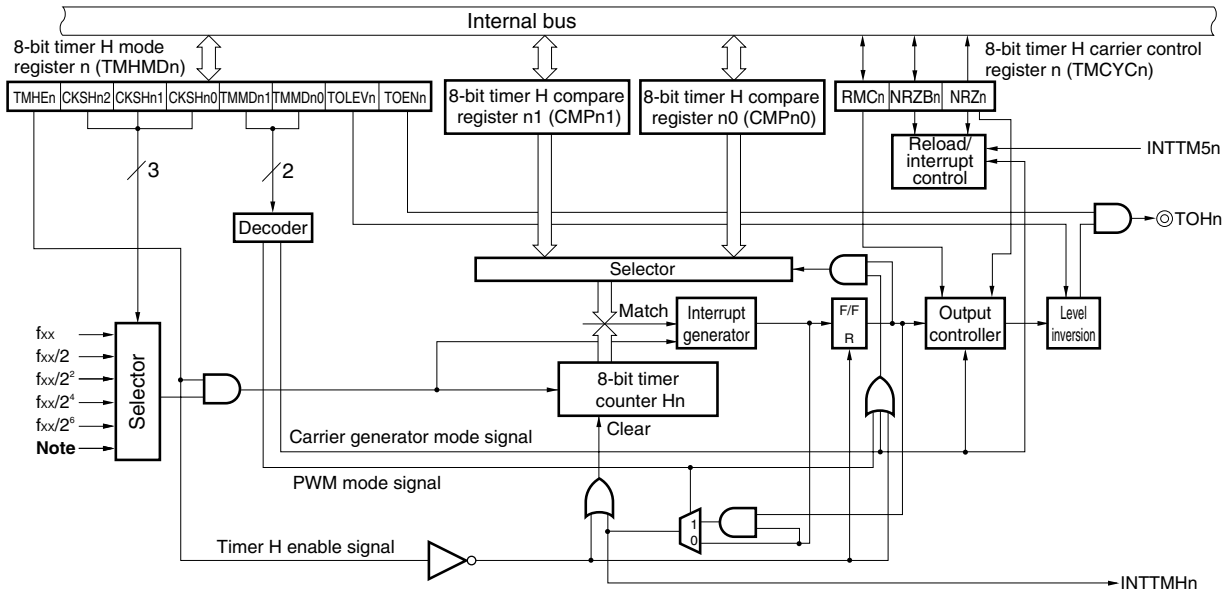
Note To use the TOHn pin function, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

Remark n = 0, 1

Downloaded from Arrow.com.

The block diagram is shown below.

Figure 9-1. Block Diagram of 8-bit Timer Hn



Note $f_{XX}/2^{10}$ when n = 0, f_{XT} when n = 1

Remark n = 0, 1

(1) 8-bit timer H compare register n0 (CMPn0)

This register can be read or written in 8-bit units. This register is used in all of the timer operation modes.

<R> This register constantly compares the value set to the CMPn0 register with the count value of 8-bit timer counter Hn and, when the two values match, generates an interrupt request signal (INTTMHn) and inverts the output level of the TOHn pin.

Rewrite the value of the CMPn0 register while the timer is stopped (TMHMDn.TMHEn bit = 0).

Reset sets this register to 00H.

After reset: 00H R/W Address: CMP00 FFFFF582H, CMP10 FFFFF592H

	7	6	5	4	3	2	1	0
CMPn0								

(n = 0, 1)

Caution Rewriting the CMPn0 register during timer count operation is prohibited.

Downloaded from Arrow.com.

(2) 8-bit timer H compare register n1 (CMPn1)

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

<R> This register is used in the PWM output mode and carrier generator mode.

In the PWM output mode, this register constantly compares the value set to the CMPn1 register with the count value of 8-bit timer counter Hn and, when the two values match, inverts the output level of the TOHn pin. No interrupt request signal is generated.

In the carrier generator mode, the CMPn1 register always compares the value set to the CMPn1 register with the count value of 8-bit timer counter Hn and, when the two values match, generates an interrupt request signal (INTTMHn). At the same time, the count value is cleared.

The CMPn1 register can be rewritten during timer count operation.

If the value of the CMPn1 register is rewritten while the timer is operating, the new value is latched and transferred to the CMPn1 register when the count value of the timer matches the old value of the CMPn1 register, and then the value of the CMPn1 register is changed to the new value. If matching of the count value and the CMPn1 register value and writing a value to the CMPn1 register conflict, the value of the CMPn1 register is not changed.

Reset sets this register to 00H.

After reset: 00H R/W Address: CMP01 FFFFF583H, CMP11 FFFFF593H

	7	6	5	4	3	2	1	0
CMPn1 (n = 0, 1)								

The CMPn1 register can be rewritten during timer count operation.

In the carrier generator mode, after the CMPn1 register is set, if the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, an interrupt request signal (INTTMHn) is generated. At the same time, the value of 8-bit timer counter Hn is cleared to 00H.

If the set value of the CMPn1 register is rewritten during timer operation, the reload timing is when the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match. If the transfer timing and write to the CMPn1 register from the CPU conflict, transfer is not performed.

Caution In the PWM output mode and carrier generator mode, be sure to set the CMPn1 register when starting the timer count operation (TMHMDn.TMHEn bit = 1) after the timer count operation was stopped (TMHEn bit = 0) (be sure to set again even if setting the same value to the CMPn1 register).

Downloaded from Arrow.com.

9.3 Registers

The registers that control 8-bit timer Hn are as follows.

- 8-bit timer H mode register n (TMHMDn)
- 8-bit timer H carrier control register n (TMCYCn)

Remarks 1. To use the TOHn pin function, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. n = 0, 1

(1) 8-bit timer H mode register n (TMHMDn)

The TMHMDn register controls the mode of 8-bit timer Hn.

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

Reset sets TMHMDn to 00H.

Remark n = 0, 1

Downloaded from Arrow.com.

(a) 8-bit timer H mode register 0 (TMHMD0)

After reset: 00H R/W Address: FFFFF580H

	<7>	6	5	4	3	2	<1>	<0>
TMHMD0	TMHE0	CKSH02	CKSH01	CKSH00	TMMD01	TMMD00	TOLEV0	TOEN0

TMHE0	8-bit timer H0 operation enable
0	Stop timer count operation (8-bit timer counter H0 = 00H)
1	Enable timer count operation (Counting starts when clock is input)

CKSH02	CKSH01	CKSH00	Selection of count clock			
			Count clock[Note]	f_{XX} = 20 MHz	f_{XX} = 16.0 MHz	f_{XX} = 10.0 MHz
0	0	0	f_{XX}	Setting prohibited	Setting prohibited	100 ns
0	0	1	$f_{XX}/2$	100 ns	125 ns	200 ns
0	1	0	$f_{XX}/4$	200 ns	250 ns	400 ns
0	1	1	$f_{XX}/16$	800 ns	1 μs	1.6 μs
1	0	0	$f_{XX}/64$	3.2 μs	4 μs	6.4 μs
1	0	1	$f_{XX}/1024$	51.2 μs	64 μs	102.4 μs
Other than above			Setting prohibited			

TMMD01	TMMD00	8-bit timer H0 operation mode
0	0	Interval timer mode
0	1	Carrier generator mode
1	0	PWM output mode
1	1	Setting prohibited

TOLEV0	Timer output level control (default)
0	Low level
1	High level

TOEN0	Timer output control
0	Disable output
1	Enable output

Note Set so as to satisfy the following conditions.

V_{DD} = 4.0 to 5.5 V: Count clock ≤ 10 MHz

V_{DD} = 2.7 to 4.0 V: Count clock ≤ 5 MHz

Cautions 1. When the TMHE0 bit = 1, setting bits other than those of the TMHMD0 register is prohibited.

2. In the PWM output mode and carrier generator mode, be sure to set the CMP01 register when starting the timer count operation (TMHE0 bit = 1) after the timer count operation was stopped (TMHE0 bit = 0) (be sure to set again even if setting the same value to the CMP01 register).

3. When using the carrier generator mode, set 8-bit timer H0 count clock frequency to six times 8-bit timer/event counter 50 count clock frequency or higher.

Downloaded from Arrow.com.

(b) 8-bit timer H mode register 1 (TMHMD1)

After reset: 00H R/W Address: FFFFF590H

	<7>	6	5	4	3	2	<1>	<0>
TMHMD1	TMHE1	CKSH12	CKSH11	CKSH10	TMMD11	TMMD10	TOLEV1	TOEN1

TMHE1	8-bit timer H1 operation enable
0	Stop timer count operation (8-bit timer counter H1 = 00H)
1	Enable timer count operation (Counting starts when clock is input)

CKSH12	CKSH11	CKSH10	Selection of count clock			
			Count clock[Note]	f_{XX} = 20.0 MHz	f_{XX} = 16.0 MHz	f_{XX} = 10.0 MHz
0	0	0	f_{XX}	Setting prohibited	Setting prohibited	100 ns
0	0	1	$f_{XX}/2$	100 ns	125 ns	200 ns
0	1	0	$f_{XX}/4$	200 ns	250 ns	400 ns
0	1	1	$f_{XX}/16$	800 ns	1 μs	1.6 μs
1	0	0	$f_{XX}/64$	3.2 μs	4 μs	6.4 μs
1	0	1	f_{XT} (subclock)			
Other than above			Setting prohibited			

TMMD11	TMMD10	8-bit timer H1 operation mode
0	0	Interval timer mode
0	1	Carrier generator mode
1	0	PWM output mode
1	1	Setting prohibited

TOLEV1	Timer output level control (default)
0	Low level
1	High level

TOEN1	Timer output control
0	Disable output
1	Enable output

Note Set so as to satisfy the following conditions.

V_{DD} = 4.0 to 5.5 V: Count clock ≤ 10 MHz

V_{DD} = 2.7 to 4.0 V: Count clock ≤ 5 MHz

Cautions 1. When the TMHE1 bit = 1, setting bits other than those of the TMHMD1 register is prohibited.

2. In the PWM output mode and carrier generator mode, be sure to set the CMP11 register when starting the timer count operation (TMHE1 bit = 1) after the timer count operation was stopped (TMHE1 bit = 0) (be sure to set again even if setting the same value to the CMP11 register).

3. When using the carrier generator mode, set 8-bit timer H1 count clock frequency to six times 8-bit timer/event counter 51 count clock frequency or higher.

Downloaded from Arrow.com.

(2) 8-bit timer H carrier control register n (TMCYCn)

This register controls the 8-bit timer Hn remote control output and carrier pulse output status.

TMCYCn register can be read or written in 8-bit or 1-bit units. The NRZn bit is a read-only bit.

Reset sets TMCYCn to 00H.

Remark n = 0, 1

After reset: 00H R/W Address: TMCYC0 FFFFF581H, TMCYC1 FFFFF591H

	7	6	5	4	3	2	1	<0>
TMCYCn	0	0	0	0	0	RMCn	NRZBn	NRZn

(n = 0, 1)

RMCn	NRZBn	Remote control output
0	0	Low-level output
0	1	High-level output
1	0	Low-level output
1	1	Carrier pulse output

NRZn	Carrier pulse output status flag
0	Carrier output disabled status (low-level status)
1	Carrier output enable status

Downloaded from Arrow.com.

9.4 Operation

9.4.1 Operation as interval timer/square wave output

When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, an interrupt request signal (INTTMHn) is generated and 8-bit timer counter Hn is cleared to 00H.

The CMPn1 register cannot be used in the interval timer mode. Even if the CMPn1 register is set, this has no effect on the timer output because matches between 8-bit timer counter Hn and the CMPn1 register are not detected.

A square wave of the desired frequency (duty = 50%) is output from the TOHn pin, by setting the TMHMDn.TOENn bit to 1.

Remarks 1. For the alternate-function pin (TOHn) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

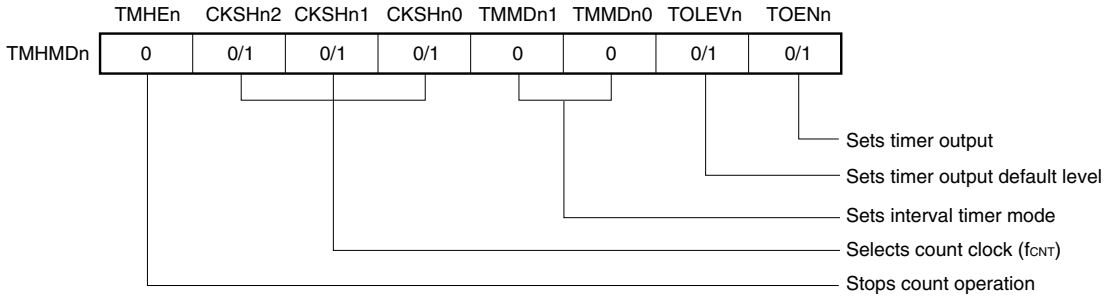
2. For INTTMHn interrupt enable, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

Setting

<1> Set each register.

Figure 9-2. Register Settings in Interval Timer Mode

(i) 8-bit timer H mode register n (TMHMDn) settings



(ii) CMPn0 register settings

The interval time is as follows if N is set as a comparison value.

- Interval time = (N + 1)/f_{CNT}

<2> When the TMHEn bit is set to 1, counting starts.

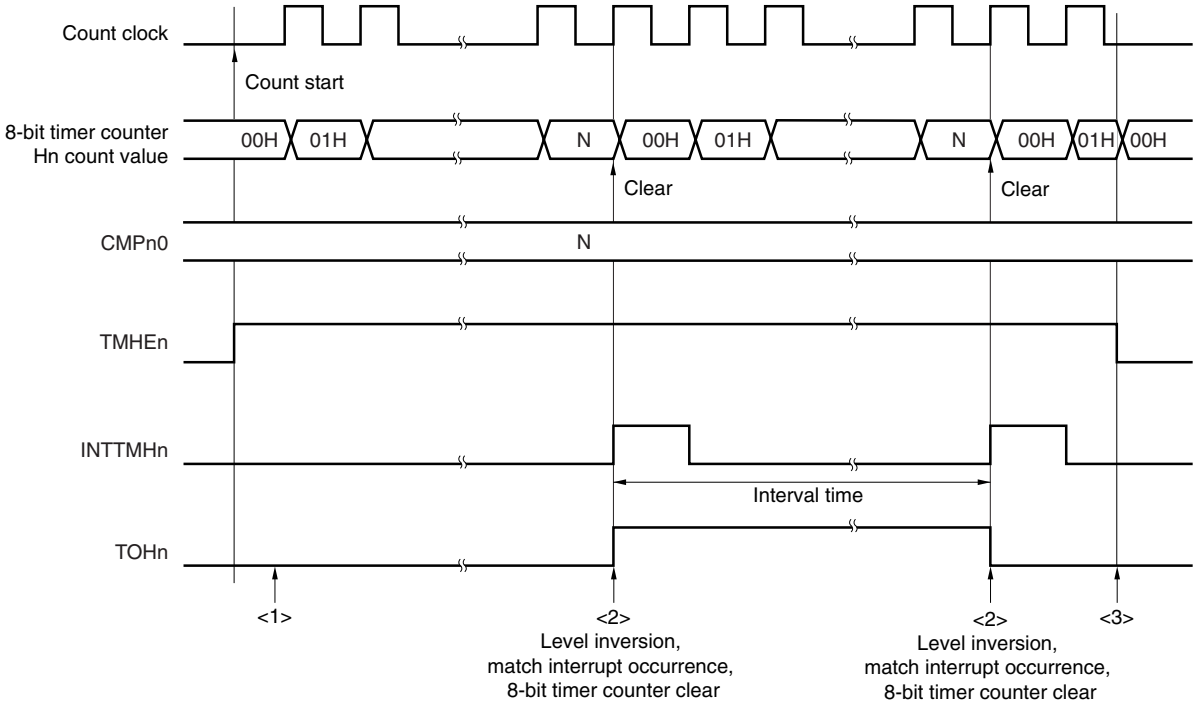
<3> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the INTTMHn signal is generated and 8-bit timer counter Hn is cleared to 00H.

<4> Then, the INTTMHn signal is generated in the same interval. To stop the count operation, clear the TMHEn bit to 0.

Downloaded from Arrow.com.

Figure 9-3. Timing of Interval Timer/Square Wave Output Operation (1/2)

Basic operation (operation when 01H ≤ CMPn0 ≤ FEH)



<1> When the TMHEn bit is set to 1, the count operation is enabled. The count clock starts counting no more than one clock after operation has been enabled.

<2> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the value of 8-bit timer counter Hn is cleared, the TOHn output level is inverted, and the INTTMHn signal is output at the rising edge of the count clock.

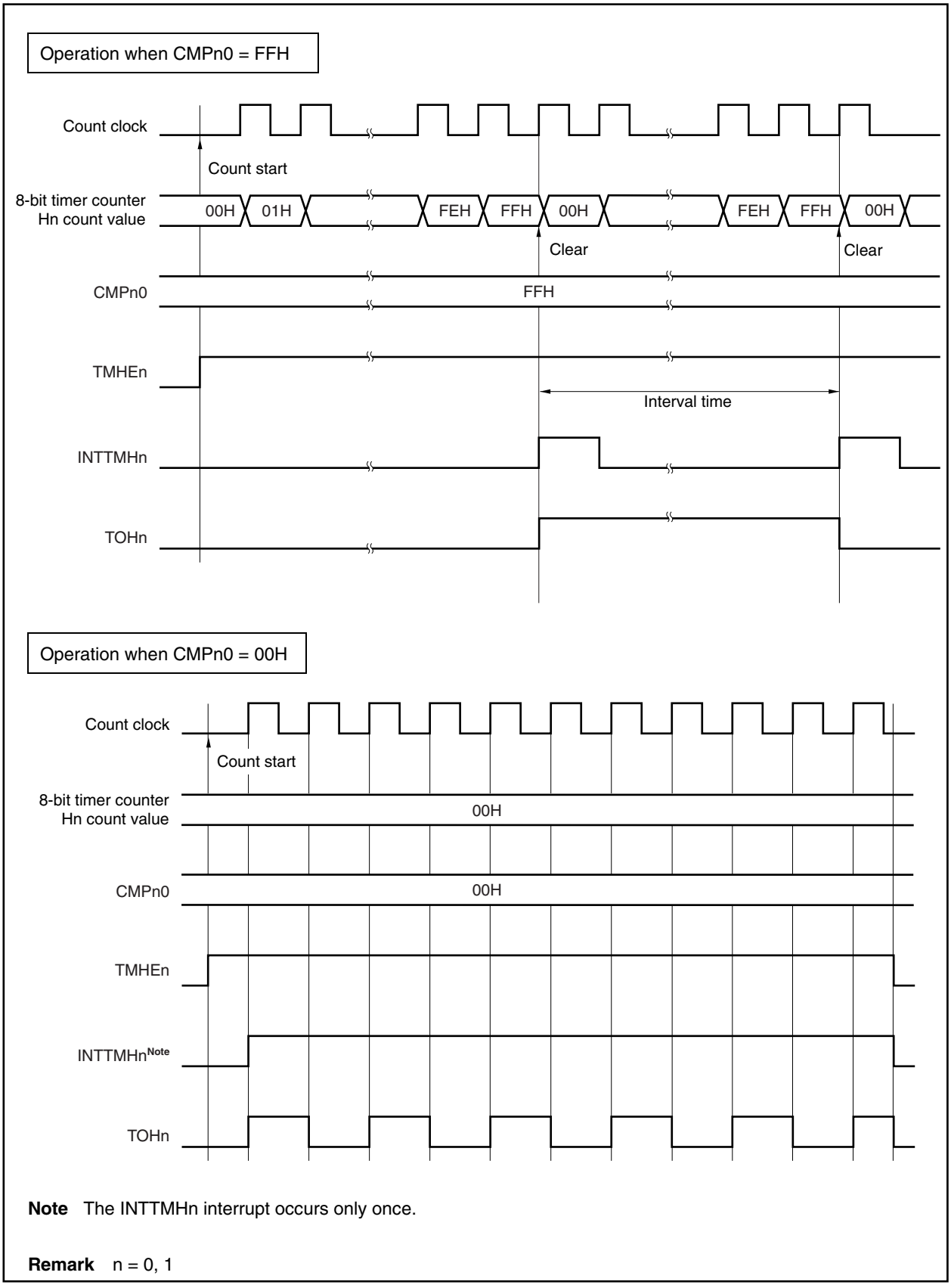
<3> The INTTMHn signal and TOHn output are set to the default level when the TMHEn bit is cleared to 0 during 8-bit timer Hn operation. If the level is already at the default level before the TMHMDn.TMHEn bit is cleared to 0, that level is maintained.

Remarks 1. n = 0, 1

2. 01H ≤ N ≤ FEH

Downloaded from Arrow.com.

Figure 9-3. Timing of Interval Timer/Square Wave Output Operation (2/2)



<R> **Note** The INTTMHn interrupt occurs only once.

Remark n = 0, 1

Downloaded from Arrow.com.

9.4.2 PWM output mode operation

In the PWM output mode, a pulse of any duty and cycle can be output.

The CMPn0 register controls the timer output (TOHn) cycle. Rewriting the CMPn0 register during timer operation is prohibited.

The CMPn1 register controls the timer output (TOHn) duty. The CMPn1 register can be rewritten during timer operation.

The operation in the PWM output mode is as follows.

After timer counting starts, when the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the TOHn output level is inverted and 8-bit timer counter Hn is cleared to 00H. When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the TOHn output level is inverted.

Remarks 1. For the alternate-function pin (TOHn) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

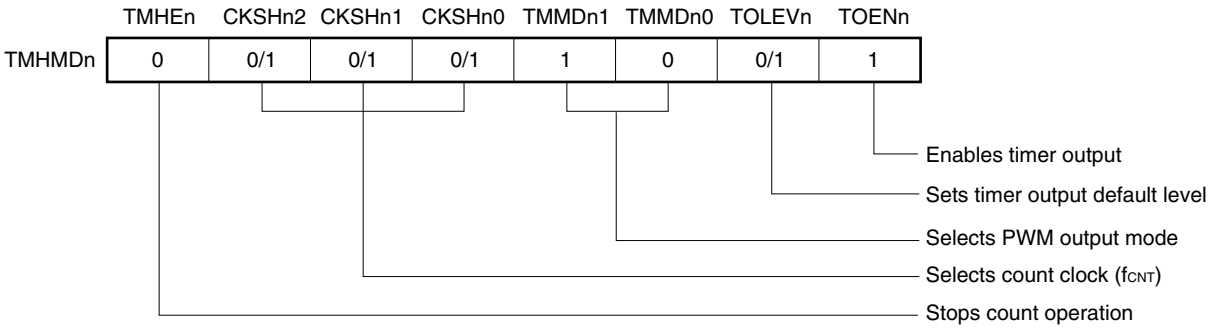
2. For INTTMHn interrupt enable, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

Setting

<1> Set each register.

Figure 9-4. Register Settings in PWM Output Mode

(i) 8-bit timer H mode register n (TMHMDn) settings



(ii) CMPn0 register setting

- Compare value (N): Sets cycle

(ii) CMPn1 register setting

- Compare value (M): Sets duty

Remarks 1. n = 0, 1

2. 00H ≤ CMPn1 (M) < CMPn0 (N) ≤ FFH

<2> When the TMHEn bit is set to 1, counting starts.

Downloaded from Arrow.com.

<3> After the count operation is enabled, the first compare register to be compared is the CMPn0 register. When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, 8-bit timer counter Hn is cleared, an interrupt request signal (INTTMHn) is generated, and the TOHn output level is inverted. At the same time, the register that is compared with 8-bit timer counter Hn changes from the CMPn0 register to the CMPn1 register.

<4> When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the TOHn output level is inverted, and at the same time the register that is compared with 8-bit timer counter Hn changes from the CMPn1 register to the CMPn0 register. At this time, 8-bit timer counter Hn is not cleared and the INTTMHn signal is not generated.

<5> A pulse of any duty can be obtained through the repetition of steps <3> and <4> above.

<6> To stop the count operation, clear the TMHEn bit to 0.

Designating the set value of the CMPn0 register as (N), the set value of the CMPn1 register as (M), and the count clock frequency as f_{CNT}, the PWM pulse output cycle and duty are as follows.

PWM pulse output cycle = (N + 1)/f_{CNT}
Duty = inactive width: Active width = (M + 1) : (N + 1)

<R> **Cautions 1. The set value of the CMPn1 register can be changed while the timer counter is operating. However, this takes a duration of at least three operating clocks (signal selected by the CKSHn2 to CKSHn0 bits of the TMHMDn register) from when the value of the CMPn1 register is changed until the value is transferred to the register.**

2. Be sure to set the CMPn1 register when starting the timer count operation (TMHEn bit = 1) after the timer count operation was stopped (TMHEn bit = 0) (be sure to set again even if setting the same value to the CMPn1 register).

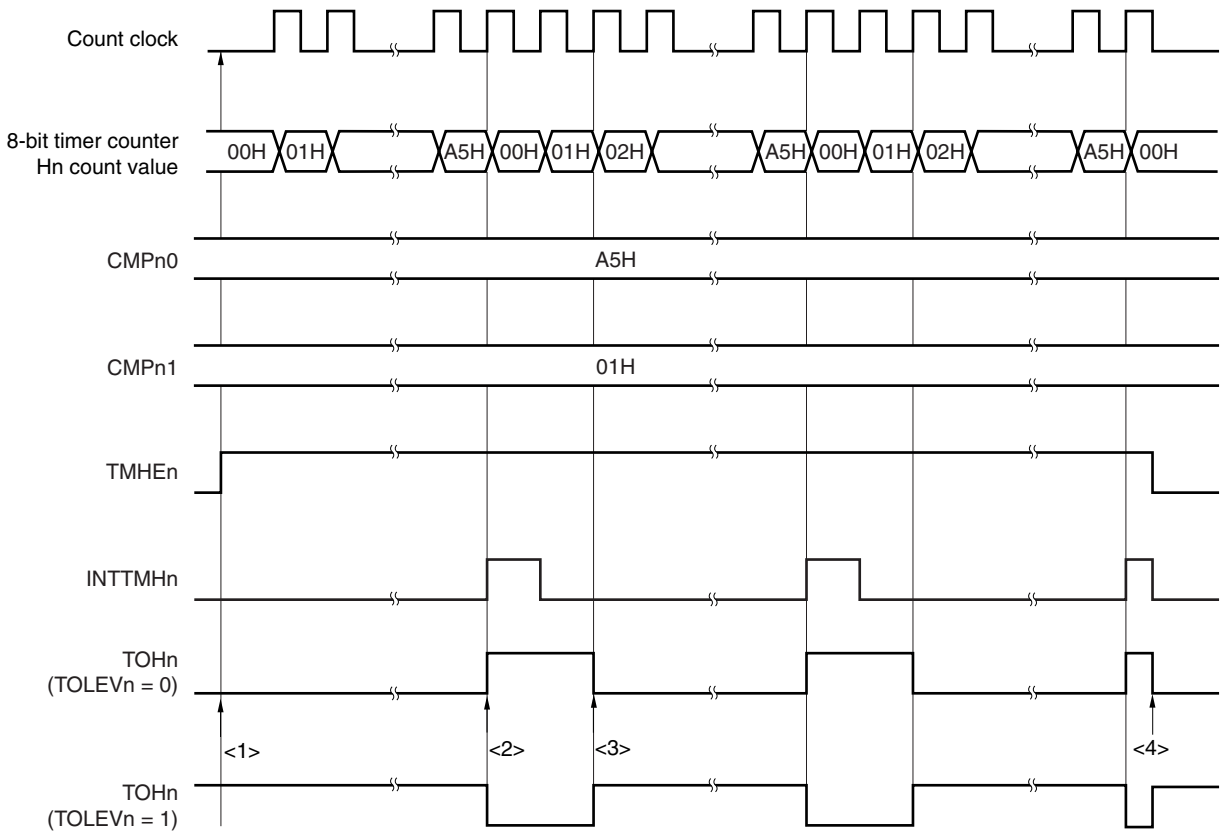
<R> **3. Make sure that the CMPn1 register set value (M) and CMPn0 register set value (N) are within the following range.**

00H ≤ CMPn1 (M) < CMPn0 (N) ≤ FFH

Downloaded from Arrow.com.

Figure 9-5. Operation Timing in PWM Output Mode (1/4)

Basic operation



<1> When the TMHEn bit is set to 1, counting starts. At this time TOHn output remains the default level.

<2> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the TOHn output level is inverted, 8-bit timer counter Hn is cleared, and the INTTMHn signal is output.

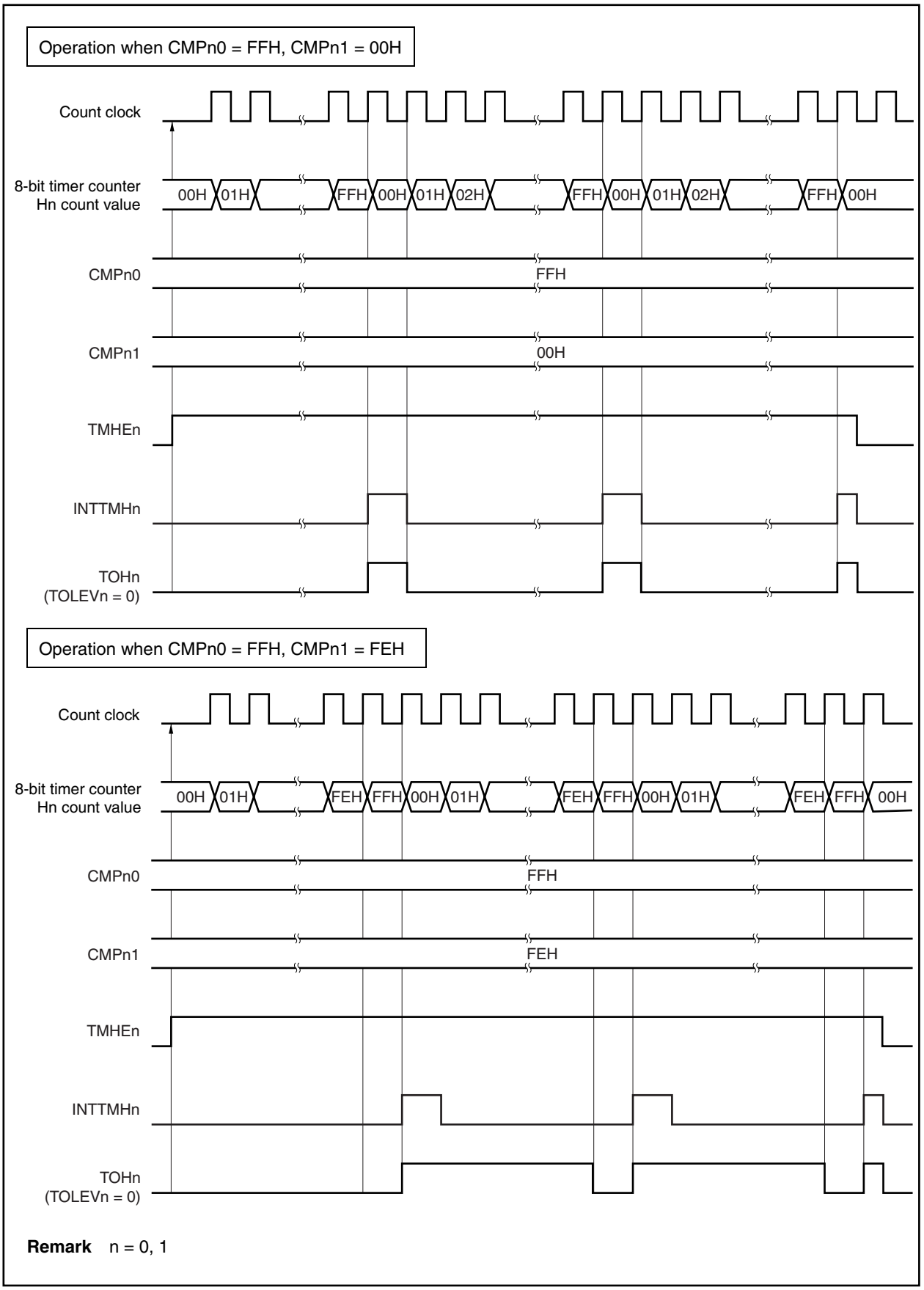
<3> When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the TOHn output level is inverted. At this time, the value of 8-bit timer counter Hn is not cleared and the INTTMHn signal is not output.

<4> When the TMHEn bit is cleared to 0 during 8-bit timer Hn operation, the INTTMHn signal and TOHn output are set to the default level.

Remark n = 0, 1

Downloaded from Arrow.com.

Figure 9-5. Operation Timing in PWM Output Mode (2/4)

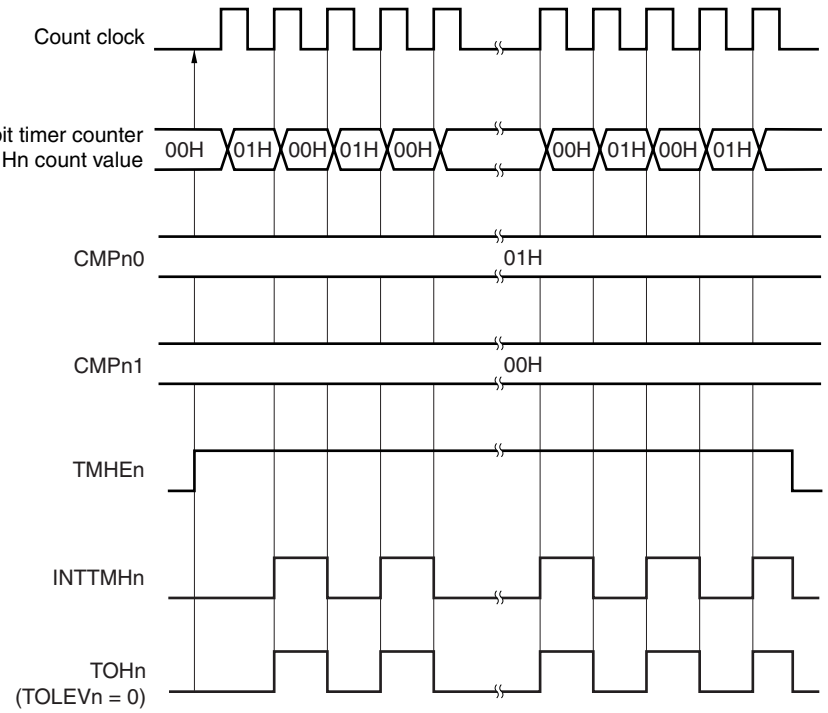


Remark n = 0, 1

Downloaded from Arrow.com.

Figure 9-5. Operation Timing in PWM Output Mode (3/4)

Operation when CMPn0 = 01H, CMPn1 = 00H

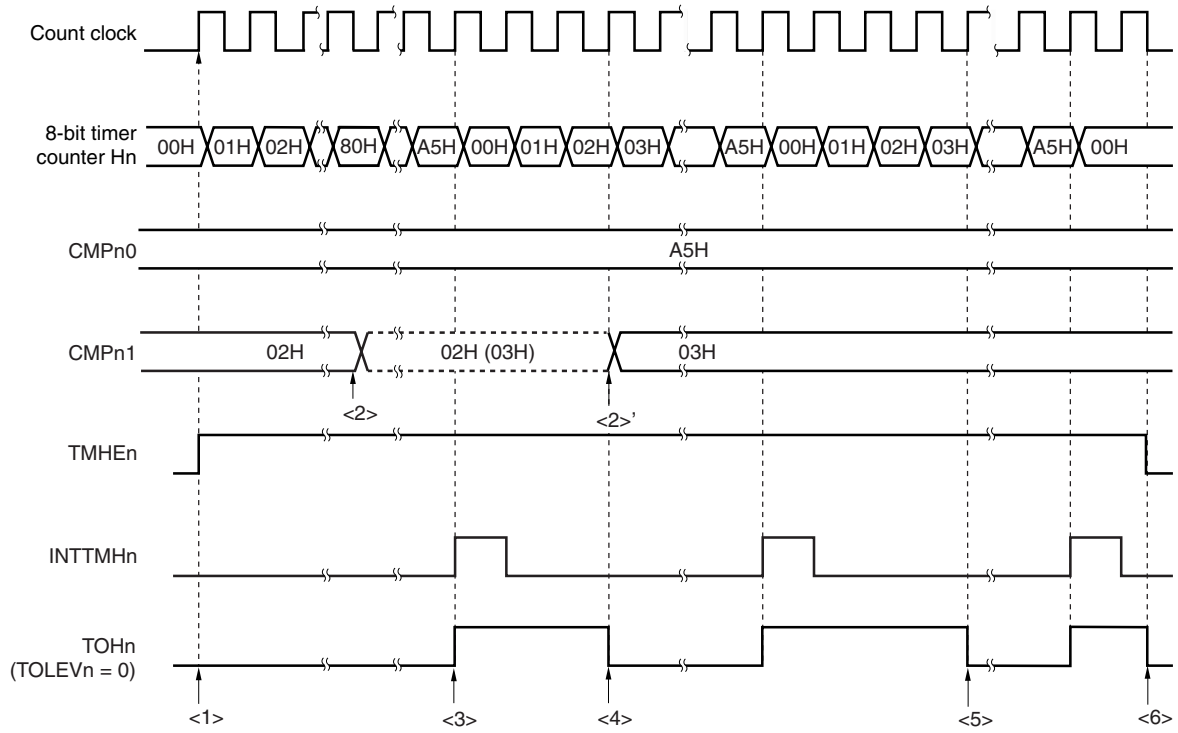


Remark n = 0, 1

Downloaded from Arrow.com.

<R> **Figure 9-5. Operation Timing in PWM Output Mode (4/4)**

Operation by changing CMPn1 (CMPn1 = 02H → 03H, CMPn0 = A5H)



<1> When the TMHEn bit is set to 1, counting starts. At this time, the TOHn output remains the default level.

<2> The set value of the CMPn1 register can be changed during count operation. This operation is asynchronous to the count clock.

<3> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, 8-bit timer counter Hn is cleared, the TOHn output level is inverted, and the INTTMHn signal is generated.

<4> Even if the value of the CMPn1 register is changed, that value is latched and not transferred to the register. When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register prior to the change match, the changed value is transferred to the CMPn1 register and the value of the CMPn1 register is changed (<2>').

However, three or more count clocks are required from the time the value of the CMPn1 register is changed until it is transferred to the register. Even if a match signal is generated within three count clocks, the changed value cannot be transferred to the register.

<5> When the count value of 8-bit timer counter Hn matches the changed set value of the CMPn1 register, the TOHn output level is inverted. 8-bit timer counter Hn is not cleared and the INTTMHn signal is not generated.

<6> When the TMHEn bit is cleared to 0 during 8-bit timer Hn operation, the INTTMHn signal and TOHn output are set to the default level.

Downloaded from Arrow.com.

9.4.3 Carrier generator mode operation

The carrier clock generated by 8-bit timer Hn is output using the cycle set with 8-bit timer/event counter 5n.

In the carrier generator mode, 8-bit timer/event counter 5n is used to control the extent to which the carrier pulse of 8-bit timer Hn is output, and the carrier pulse is output from the TOHn output.

Remarks 1. For the alternate-function pin (TOHn) settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

2. For INTTMHn interrupt enable, refer to **CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

(1) Carrier generation

In the carrier generator mode, the CMPn0 register generates a waveform with the low-level width of the carrier pulse and the CMPn1 register generates a waveform with the high-level width of the carrier pulse.

During 8-bit timer Hn operation, the CMPn1 register can be rewritten, but rewriting of the CMPn0 register is prohibited.

(2) Carrier output control

Carrier output control is performed with the interrupt request signal (INTTM5n) of 8-bit timer/event counter 5n and the TMCYCn.NRZBn and TMCYCn.RMCn bits. The output relationships are as follows.

RMCn Bit	NRZBn Bit	Output
0	0	Low level output
0	1	High level output
1	0	Low level output
1	1	Carrier pulse output

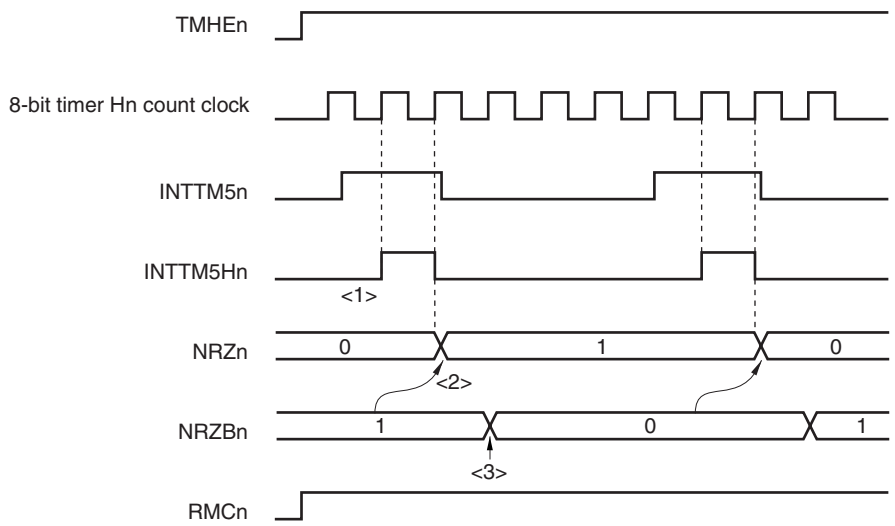
Remark n = 0, 1

Downloaded from Arrow.com.

To control carrier pulse output during count operation, the TMCYCn.NRZn and TMCYCn.NRZBn bits have a master and slave bit configuration. The NRZn bit is read-only while the NRZBn bit can be read and written. The INTTM5n signal is synchronized with the 8-bit timer Hn clock and output as the INTTM5Hn signal. The INTTM5Hn signal becomes the data transfer signal of the NRZn bit and the value of the NRZBn bit is transferred to the NRZn bit. The transfer timing from the NRZBn bit to the NRZn bit is as follows.

<R>

Figure 9-6. Transfer Timing



<1> The INTTM5n signal is synchronized with the count clock of 8-bit timer Hn and is output as the INTTM5Hn signal.

<2> The value of the NRZBn bit is transferred to the NRZn bit at the second clock from the rising edge of the INTTM5Hn signal.

<3> Write the next value to the NRZBn bit in the interrupt servicing programming that has been started by the INTTM5Hn interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR5n register.

Cautions 1. Do not rewrite the NRZBn bit again until at least the second clock after it has been rewritten, or else transfer from the NRZBn bit to the NRZn bit is not guaranteed.

2. When using 8-bit timer/event counter 5n in the carrier generator mode, an interrupt occurs at the timing of <1>. An interrupt occurs at a different timing when it is used in other than the carrier generator mode.

Remark n = 0, 1

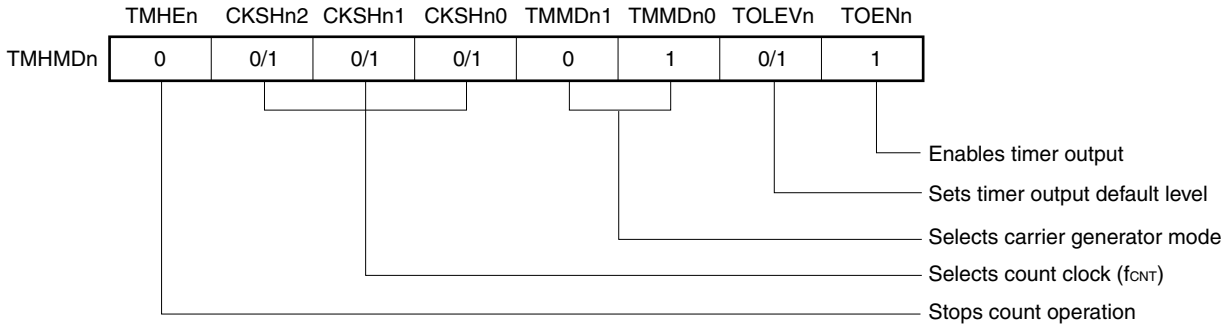
Downloaded from Arrow.com.

Setting

<1> Set each register.

Figure 9-7. Register Settings in Carrier Generator Mode

- 8-bit timer H mode register n (TMHMDn)



- CMPn0 register: Compare value
- CMPn1 register: Compare value
- TMCYCn register: RMCn = 1 ... Remote control output enable bit
  NRZBn = 0/1 ... Carrier output enable bit
- TCL5n, TMC5n registers: Refer to **8.3 Registers**.

Remark n = 0, 1

<2> When the TMHEn bit is set to 1, 8-bit timer Hn count operation starts.

<3> When the TMC5n.TCE5n bit is set to 1, 8-bit timer/event counter 5n count operation starts.

<4> After the count operation is enabled, the first compare register to be compared is the CMPn0 register. When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the INTTMHn signal is generated, 8-bit timer counter Hn is cleared, and at the same time, the register that is compared with 8-bit timer counter Hn changes from the CMPn0 register to the CMPn1 register.

<5> When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the INTTMHn signal is generated, 8-bit timer counter Hn is cleared, and at the same time, the register that is compared with 8-bit timer counter Hn changes from the CMPn1 register to the CMPn0 register.

<6> The carrier clock is obtained through the repetition of steps <4> and <5> above.

<7> The INTTM5n signal is synchronized with 8-bit timer Hn and output as the INTTM5Hn signal. This signal becomes the data transfer signal of the NRZBn bit and the value of the NRZBn bit is transferred to the NRZn bit.

<R> <8> Write the next value to the NRZBn bit in the interrupt servicing programming that has been started by the INTTM5Hn interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR5n register.

<9> When the NRZn bit becomes high level, the carrier clock is output from the TOHn pin.

<10> Any carrier clock can be obtained through the repetition of the above steps. To stop the count operation, clear the TMHEn bit to 0.

Downloaded from Arrow.com.

Designating the set value of the CMPn0 register as (N), the set value of the CMPn1 register as (M), and the count clock frequency as f_{CNT}, the carrier clock output cycle and duty are as follows.

Carrier clock output cycle = (N + M + 2)/f_{CNT}
Duty = High level width: Carrier clock output width = (M + 1) : (N + M + 2)

Cautions 1. Be sure to set the CMPn1 register when starting the timer count operation (TMHEn bit = 1) after the timer count operation was stopped (TMHEn bit = 0) (be sure to set again even if setting the same value to the CMPn1 register).

2. Set the values of the CMPn0 and CMPn1 registers in the range of 01H to FFH.

3. In the carrier generator mode, three operating clocks (signal selected by the TMHMDn.CKSHn0 to TMHMDn.CKSHn2 bits) are required for actual transfer of the new value to the register after the CMPn1 register has been rewritten.

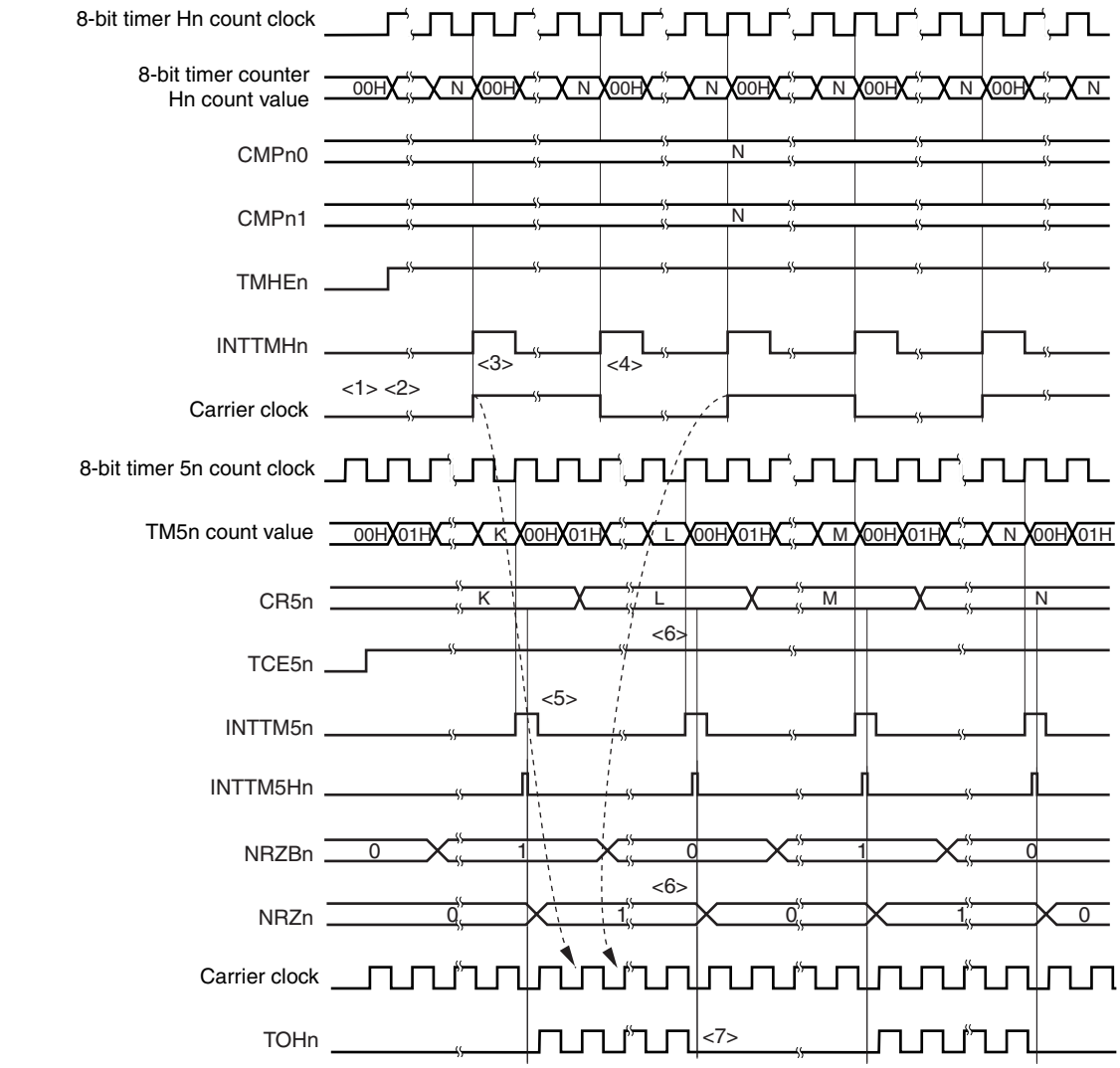
4. Be sure to perform the TMCYCn.RMCn bit setting before the start of the count operation.

5. When using the carrier generator mode, set the 8-bit timer Hn count clock frequency to six times the 8-bit timer/event counter 5n count clock frequency or higher.

Downloaded from Arrow.com.

Figure 9-8. Carrier Generator Mode (1/3)

Operation when the CMPn0 register = N, the CMPn1 register = N is set



<1> When the TMHEn bit = 0 and the TCE5n bit = 0, the operation of 8-bit timer Hn is stopped.

<2> When the TMHEn bit is set to 1, 8-bit timer Hn starts counting. The carrier clock remains the default level.

<3> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the first INTTMHn signal is generated, the carrier clock signal is inverted, and the register that is compared with 8-bit timer counter Hn changes from the CMPn0 register to the CMPn1 register. 8-bit timer counter Hn is cleared to 00H.

<4> When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the INTTMHn signal is generated, the carrier clock signal is inverted, and the register that is compared with 8-bit timer counter Hn changes from the CMPn1 register to the CMPn0 register. 8-bit timer counter Hn is cleared to 00H. A carrier clock with a duty of 50% is generated through the repetition of steps <3> and <4>.

<5> The INTTM5n signal is synchronized with 8-bit timer Hn and output as the INTTM5Hn signal.

<6> The INTTM5Hn signal becomes the data transfer signal of the NRZBn bit, and the value of the NRZBn bit is transferred to the NRZn bit.

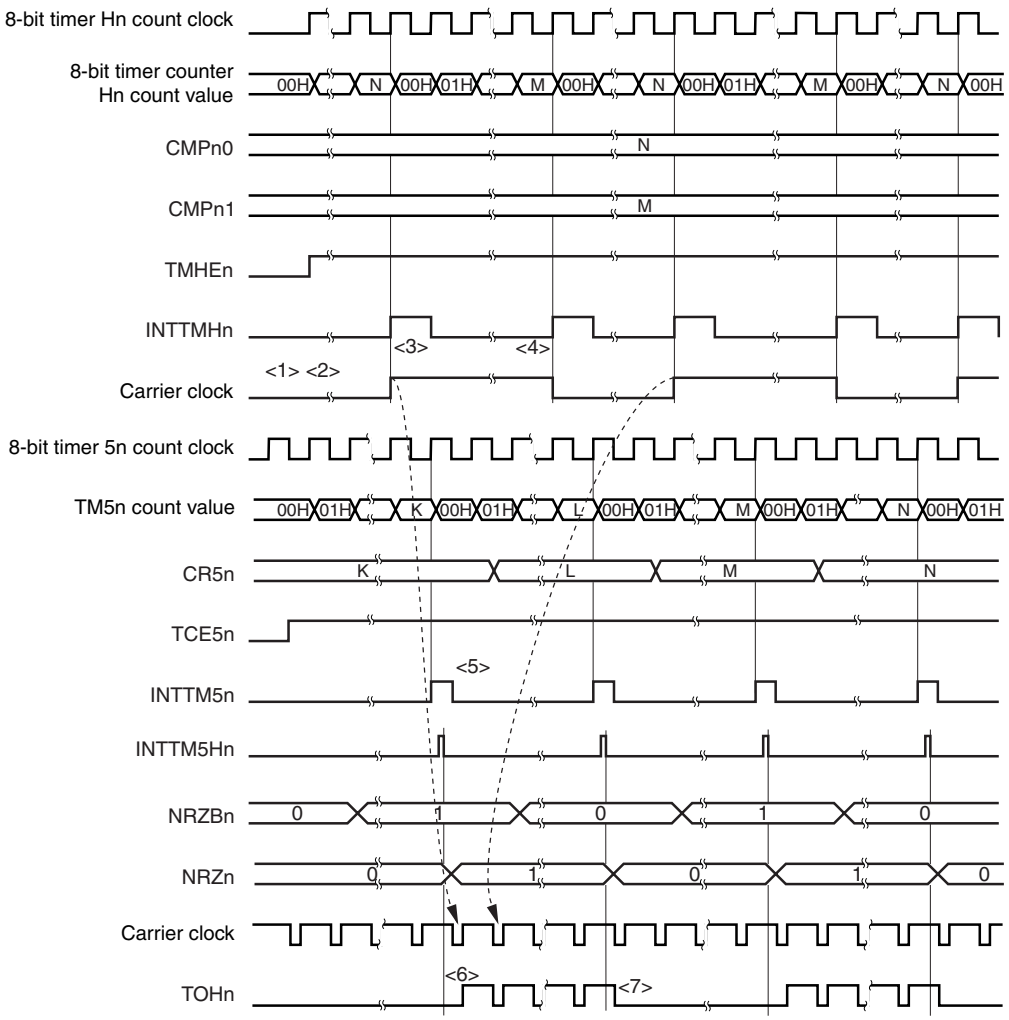
<7> The TOHn output is made low level by clearing the NRZn bit = 0.

Remark n = 0, 1

Downloaded from Arrow.com.

Figure 9-8. Carrier Generator Mode (2/3)

Operation when the CMPn0 register = N, the CMPn1 register = M is set



<1> When the TMHEn bit = 0 and the TCE5n bit = 0, the operation of 8-bit timer Hn is stopped.

<2> When the TMHEn bit is set to 1, 8-bit timer Hn starts counting. The carrier clock remains the default level at this time.

<3> When the count value of 8-bit timer counter Hn and the set value of the CMPn0 register match, the first INTTMHn signal is generated, the carrier clock signal is inverted, and the register that is compared with 8-bit timer counter Hn changes from the CMPn0 register to the CMPn1 register. 8-bit timer counter Hn is cleared to 00H.

<4> When the count value of 8-bit timer counter Hn and the set value of the CMPn1 register match, the INTTMHn signal is generated, the carrier clock signal is inverted, and the register that is compared with 8-bit timer counter Hn changes from the CMPn1 register to the CMPn0 register. 8-bit timer counter Hn is cleared to 00H. A carrier clock with a fixed duty (other than 50%) is generated through the repetition of steps <3> and <4>.

<5> The INTTM5n signal is generated. This signal is synchronized with 8-bit timer Hn and output as the INTTM5Hn signal.

<6> The carrier is output from the rising edge of the first carrier clock by setting the NRZn bit = 1.

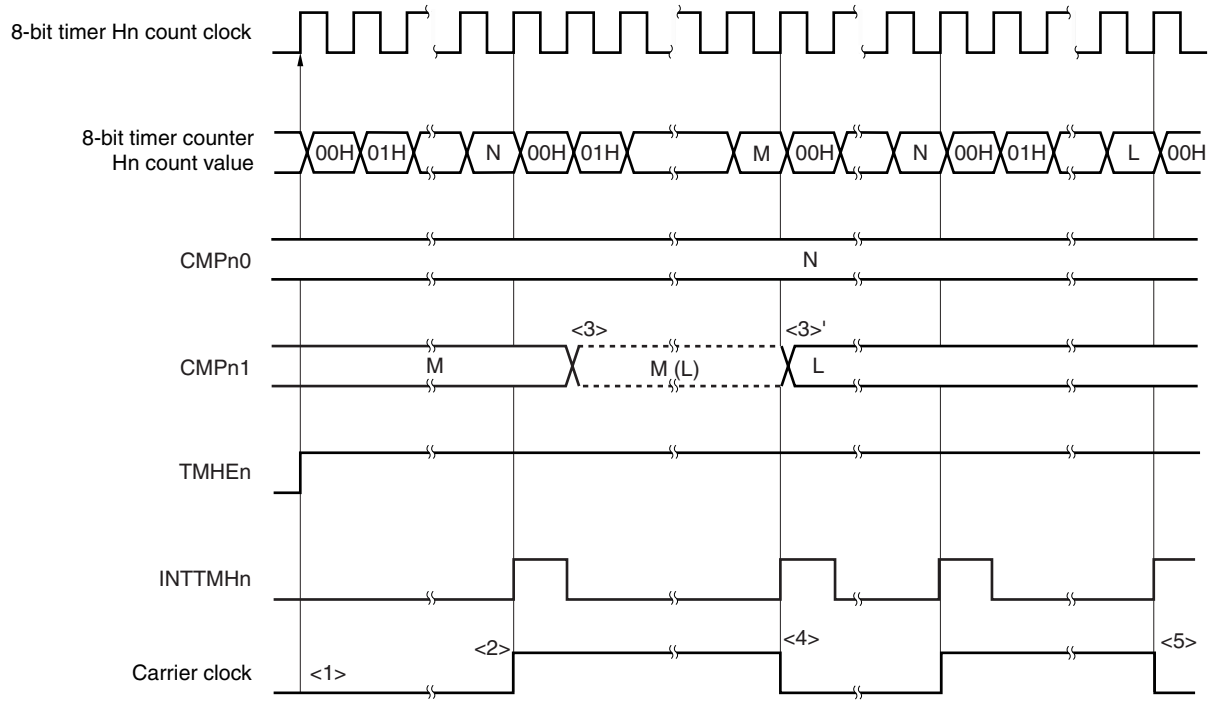
<7> By setting the NRZn bit = 0, the TOHn output is also maintained high level while the carrier clock is high level, and does not change to low level (the high level width of the carrier waveform is guaranteed through steps <6> and <7>).

Remark n = 0, 1

Downloaded from Arrow.com.

Figure 9-8. Carrier Generator Mode (3/3)

Operation based on the CMPn1 register transitions



<1> When the TMHEn bit is set to 1, counting starts. The carrier clock remains the default level at this time.

<R> <2> When the count value of the 8-bit timer counter Hn matches the value of the CMPn0 register, the INTTMHn signal is output, the carrier signal is inverted, and the 8-bit timer counter is cleared to 00H. At the same time, the compare register whose value is to be compared with that of the 8-bit timer counter Hn is changed from the CMPn0 register to the CMPn1 register.

<R> <3> The CMPn1 register is asynchronous to the count clock, and its value can be changed while the 8-bit timer Hn is operating. The new value (L) to which the value of the register is to be changed is latched. When the count value of the 8-bit timer counter Hn matches the value (M) of the CMPn1 register before the change, the CMPn1 register is changed (<3>').

However, it takes three count clocks or more since the value of the CMPn1 register has been changed until the value is transferred to the register. Even if a match signal is generated before the duration of three count clocks elapses, the new value is not transferred to the register.

<4> When the count value of 8-bit timer counter Hn and the value (M) of the CMPn1 register match, the INTTMHn signal is output, the carrier signal is inverted, and 8-bit timer counter Hn is cleared to 00H. At the same time, the compare register whose value is to be compared with that of the 8-bit timer counter Hn is changed from the CMPn1 register to the CMPn0 register.

<5> The timing at which the count value of 8-bit timer counter Hn and the value of the CMPn1 register match again is the changed value (L).

Remark n = 0, 1

Downloaded from Arrow.com.

CHAPTER 10 INTERVAL TIMER, WATCH TIMER

The V850ES/KE1 includes interval timer BRG and a watch timer. Interval timer BRG can also be used as the source clock of the watch timer. The watch timer can also be used as interval timer WT.

Two interval timer channels and one watch timer channel can be used at the same time.

10.1 Interval Timer BRG

10.1.1 Functions

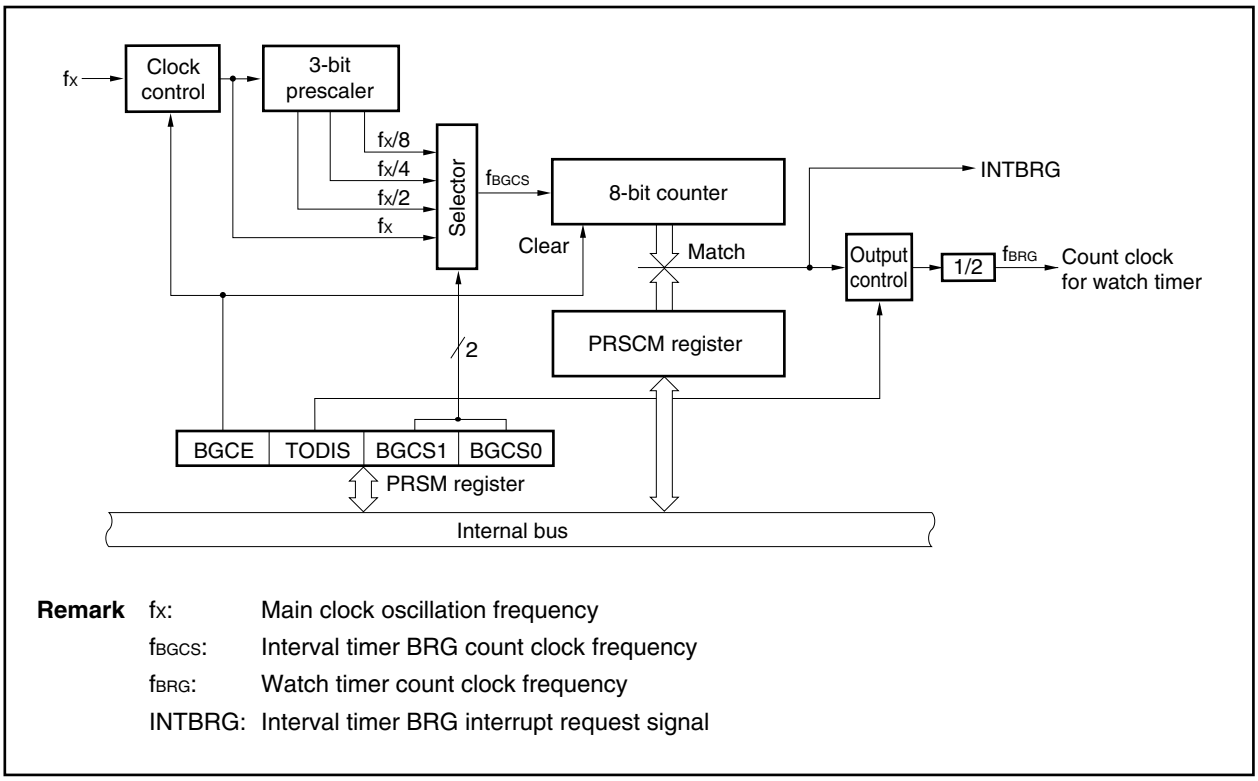
Interval timer BRG has the following functions.

- Interval timer BRG: An interrupt request signal (INTBRG) is generated at a specified interval.
- Generation of count clock for watch timer: When the main clock is used as the count clock for the watch timer, a count clock (f_{BRG}) is generated.

10.1.2 Configuration

The following shows the block diagram of interval timer BRG.

Figure 10-1. Block Diagram of Interval Timer BRG



Remark f_X: Main clock oscillation frequency
f_{BGCS}: Interval timer BRG count clock frequency
f_{BRG}: Watch timer count clock frequency
INTBRG: Interval timer BRG interrupt request signal

Downloaded from Arrow.com.

(1) Clock control

The clock control controls supply/stop of the operation clock of interval timer BRG.

(2) 3-bit prescaler

The 3-bit prescaler divides f_X to generate $f_X/2$, $f_X/4$, and $f_X/8$.

(3) Selector

The selector selects the count clock (f_{BGCS}) for interval timer BRG from f_X, $f_X/2$, $f_X/4$, and $f_X/8$.

(4) 8-bit counter

The 8-bit counter counts the count clock (f_{BGCS}).

(5) Output control

The output control controls supply of the count clock (f_{BRG}) for the watch timer.

(6) PRSCM register

The PRSCM register is an 8-bit compare register that sets the interval time.

(7) PRSM register

The PRSM register controls the operation of interval timer BRG, the selector, and clock supply to the watch timer.

Downloaded from Arrow.com.

10.1.3 Registers

Interval timer BRG includes the following registers.

(1) Interval timer BRG mode register (PRSM)

PRSM controls the operation of interval timer BRG, selection of count clock, and clock supply to the watch timer.

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

Reset sets PRSM to 00H.

After reset: 00H R/W Address: FFFFF8B0H

	7	6	5	<4>	3	2	1	0
PRSM	0	0	0	BGCE	0	TODIS	BGCS1	BGCS0

BGCE	Control of interval timer operation
0	Operation stopped, 8-bit counter cleared to 01H
1	Operate

TODIS	Control of clock supply for watch timer
0	Clock for watch timer supplied
1	Clock for watch timer not supplied

BGCS1	BGCS0	Selection of input clock (f_{BGCS})[Note]			
			10 MHz	5 MHz	4 MHz
0	0	fx	100 ns	200 ns	250 ns
0	1	fx/2	200 ns	400 ns	500 ns
1	0	fx/4	400 ns	800 ns	1 μs
1	1	fx/8	800 ns	1.6 μs	2 μs

Note Set these bits so that the following conditions are satisfied.

V_{DD} = 4.0 to 5.5 V: $f_{BGCS} \leq$ 10 MHz

V_{DD} = 2.7 to 4.0 V: $f_{BGCS} \leq$ 5 MHz

Cautions 1. Do not change the values of the TODIS, BGCS1, and BGCS0 bits while interval timer BRG is operating (BGCE bit = 1). Set the TODIS, BGCS1, and BGCS0 bits before setting (1) the BGCE bit.

2. When the BGCE bit is cleared (to 0), the 8-bit counter is cleared.

Downloaded from Arrow.com.

(2) Interval timer BRG compare register (PRSCM)

PRSCM is an 8-bit compare register.

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

Reset sets PRSCM to 00H.

After reset: 00H R/W Address: FFFFF8B1H

	7	6	5	4	3	2	1	0
PRSCM	PRSCM7	PRSCM6	PRSCM5	PRSCM4	PRSCM3	PRSCM2	PRSCM1	PRSCM0

Caution Do not rewrite the PRSCM register while interval timer BRG is operating (PRSM.BGCE bit = 1). Set the PRSCM register before setting (1) the BGCE bit.

Downloaded from Arrow.com.

10.1.4 Operation

(1) Operation of interval timer BRG

Set the count clock by using the BGCS1 and BGCS0 bits of PRSM and the 8-bit compare value by using the PRSCM register.

When the PRSM.BGCE bit is set (1), interval timer BRG starts operating.

Each time the count value of the 8-bit counter and the set value in the PRSCM register match, an interrupt <R> request signal (INTBRG) is generated. At the same time, the 8-bit counter is cleared to 01H and counting is continued.

The interval time can be obtained from the following equation.

Interval time = $2^m \times N/fx$

Remark m: Division value (set values of BGCS1 and BGCS0[Note] bits) = 0 to 3
N: Set value in PRSCM register = 1 to 256 (when the set value in the PRSCM register is 00H, N = 256)
fx: Main clock oscillation frequency

<R> **Note** When the set value of the PRSCM register = 01H, the INTBRG interrupt occurs only once.

(2) Count clock supply for watch timer

Set the count clock by using the BGCS1 and BGCS0 bits of PRSM and the 8-bit compare value by using the PRSCM register, so that the count clock frequency (f_{BRG}) of the watch timer is 32.768 kHz. Clear (0) the PRSM.TODIS bit at the same time.

When the PRSM.BGCE bit is set (1), f_{BRG} is supplied to the watch timer.

f_{BRG} is obtained from the following equation.

$f_{BRG} = fx/(2^{m+1} \times N)$

To set f_{BRG} to 32.768 kHz, perform the following calculation to set the BGCS1 and BGCS0 bits and the PRSCM register.

<1> Set N = fx/65,536 (round off the decimal) to set m = 0.
<2> If N is even, N = N/2 and m = m + 1
<3> Repeat step <2> until N is even or m = 3
<4> Set N to the PRSCM register and m to the BGCS1 and BGCS0 bits.

Example: When fx = 4.00 MHz

<1> N = 4,000,000/65,536 = 61 (round off the decimal), m = 0
<2>, <3> Since N is odd, the values remain as N = 61, m = 0
<4> The set value in the PRSCM register: 3DH (61), the set values in the BGCS1 and BGCS0 bits: 00

Remark m: Divided value (set value in the BGCS1 and BGCS0 bits) = 0 to 3
N: Set value in PRSCM register = 1 to 256 (when the set value in the PRSCM register is 00H, N = 256)
fx: Main clock oscillation frequency

Downloaded from Arrow.com.

10.2 Watch Timer

10.2.1 Functions

The watch timer has the following functions.

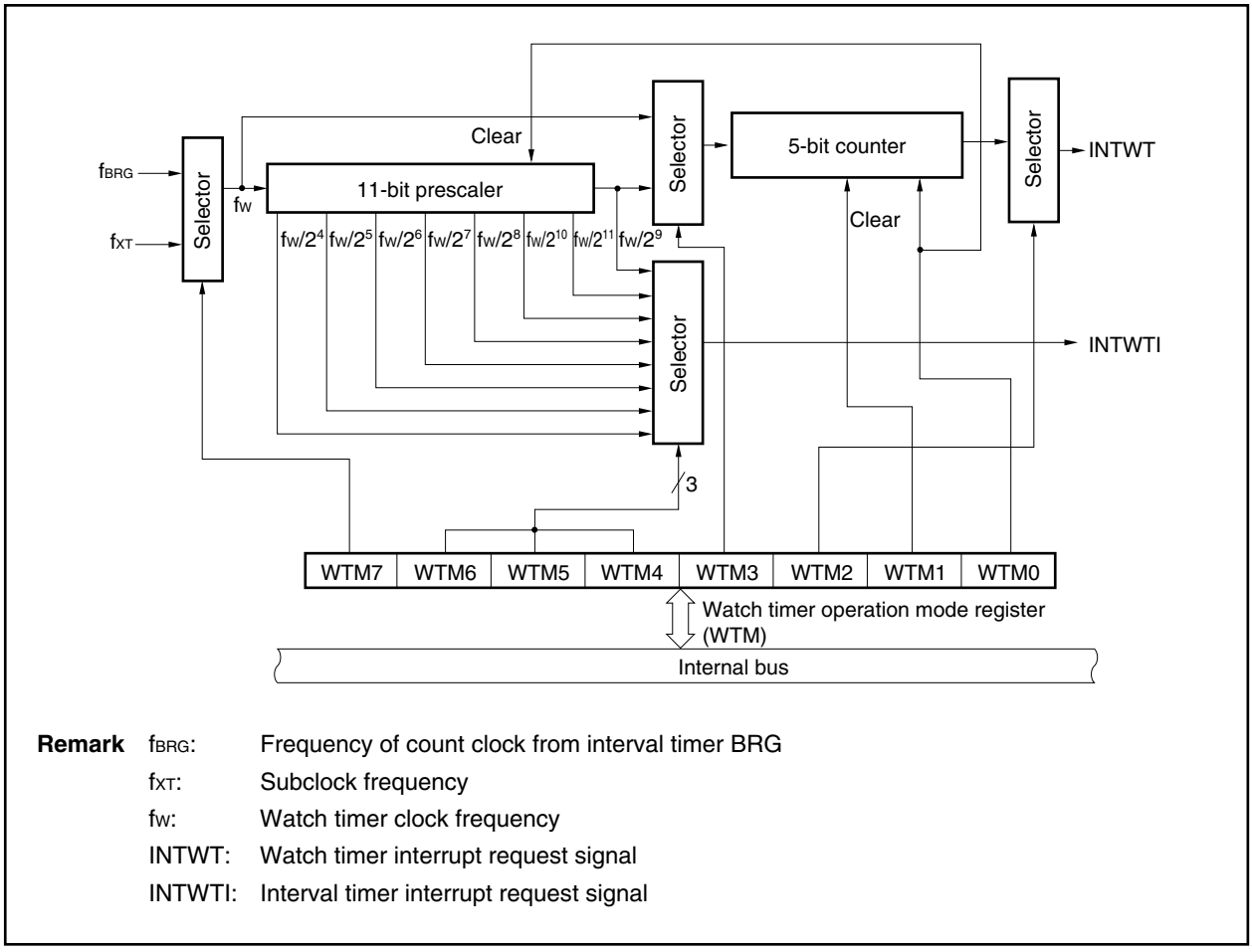
- Watch timer: An interrupt request signal (INTWT) is generated at time intervals of 0.5 or 0.25 seconds by using the main clock or subclock.
- Interval timer: An interrupt request signal (INTWTI) is generated at the preset time interval.

The watch timer and interval timer functions can be used at the same time.

10.2.2 Configuration

The following shows the block diagram of the watch timer.

Figure 10-2. Block Diagram of Watch Timer



Remark f_{BRG}: Frequency of count clock from interval timer BRG
f_{XT}: Subclock frequency
f_W: Watch timer clock frequency
INTWT: Watch timer interrupt request signal
INTWTI: Interval timer interrupt request signal

Downloaded from Arrow.com.

(1) 11-bit prescaler

The 11-bit prescaler generates a clock of $f_W/2^4$ to $f_W/2^{11}$ by dividing f_W.

(2) 5-bit counter

The 5-bit counter generates the watch timer interrupt request signal (INTWT) at intervals of $2^4/f_W$, $2^5/f_W$, $2^{13}/f_W$, or $2^{14}/f_W$ by counting f_W or $f_W/2^9$.

(3) Selectors

The watch timer has the following four selectors.

- Selector that selects the main clock (the clock from interval timer BRG (f_{BRG})) or the subclock (f_{XT}) as the clock for the watch timer.
- Selector that selects f_W or $f_W/2^9$ as the count clock frequency of the 5-bit counter
- Selector that selects $2^4/f_W$ or $2^{13}/f_W$, or $2^5/f_W$ or $2^{14}/f_W$ as the INTWT signal generation time interval.
- Selector that selects the generation time interval of the interval timer WT interrupt request signal (INTWTI) from $2^4/f_W$ to $2^{11}/f_W$.

(4) 8-bit counter

The 8-bit counter counts the count clock (f_{BGCS}).

(5) WTM register

The WTM register is an 8-bit register that controls the operation of the watch timer/interval timer WT and sets the interval of interrupt request signal generation.

10.2.3 Registers

The watch timer includes the following register.

(1) Watch timer operation mode register (WTM)

This register enables or disables the count clock and operation of the watch timer, sets the interval time of the 11-bit prescaler, controls the operation of the 5-bit counter, and sets the timer of watch timer interrupt request signal (INTWT) generation.

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

Reset sets WTM to 00H.

Downloaded from Arrow.com.

After reset: 00H R/W Address: FFFFF680H

	7	6	5	4	3	2	<1>	<0>
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0

WTM7	WTM6	WTM5	WTM4	Selection of interval timer interrupt (INTWTI) time
0	0	0	0	2^4/fw (488 μs: fw = fXT)
0	0	0	1	2^5/fw (977 μs: fw = fXT)
0	0	1	0	2^6/fw (1.95 ms: fw = fXT)
0	0	1	1	2^7/fw (3.91 ms: fw = fXT)
0	1	0	0	2^8/fw (7.81 ms: fw = fXT)
0	1	0	1	2^9/fw (15.6 ms: fw = fXT)
0	1	1	0	2^{10}/fw (31.3 ms: fw = fXT)
0	1	1	1	2^{11}/fw (62.5 ms: fw = fXT)
1	0	0	0	2^4/fw (488 μs: fw = fBRG)
1	0	0	1	2^5/fw (977 μs: fw = fBRG)
1	0	1	0	2^6/fw (1.95 ms: fw = fBRG)
1	0	1	1	2^7/fw (3.91 ms: fw = fBRG)
1	1	0	0	2^8/fw (7.81 ms: fw = fBRG)
1	1	0	1	2^9/fw (15.6 ms: fw = fBRG)
1	1	1	0	2^{10}/fw (31.3 ms: fw = fBRG)
1	1	1	1	2^{11}/fw (62.5 ms: fw = fBRG)

WTM7	WTM3	WTM2	Selection of watch timer interrupt (INTWT) time
0	0	0	2^{14}/fw (0.5 s: fw = fXT)
0	0	1	2^{13}/fw (0.25 s: fw = fXT)
0	1	0	2^5/fw (977 μs: fw = fXT)
0	1	1	2^4/fw (488 μs: fw = fXT)
1	0	0	2^{14}/fw (0.5 s: fw = fBRG)
1	0	1	2^{13}/fw (0.25 s: fw = fBRG)
1	1	0	2^5/fw (977 μs: fw = fBRG)
1	1	1	2^4/fw (488 μs: fw = fBRG)

WTM1	Control of 5-bit counter operation
0	Clear after operation stops
1	Start

WTM0	Watch timer operation enable
0	Stop operation (clear both prescaler and 5-bit counter)
1	Enable operation

Caution Rewrite the WTM2 to WTM7 bits while both the WTM0 and WTM1 bits are 0.

Remarks 1. fw: Watch timer clock frequency

2. Values in parentheses apply when fw = 32.768 kHz

Downloaded from Arrow.com.

10.2.4 Operation

(1) Operation as watch timer

The watch timer generates an interrupt request at fixed time intervals. The watch timer operates using time intervals of 0.25 or 0.5 seconds with the subclock (32.768 kHz).

The count operation starts when the WTM.WTM0 and WTM.WTM1 bits are set to 11. When these bits are cleared to 00, the 10-bit prescaler and 5-bit counter are cleared and the count operation stops.

The 5-bit counter can be cleared to synchronize the time by clearing the WTM1 bit to 0 when the watch timer and interval timer WT operate simultaneously. At this time, an error of up to 15.6 ms may occur in the watch timer, but interval timer WT is not affected.

(2) Operation as interval timer

The watch timer can also be used as an interval timer that repeatedly generates an interrupt request signal (INTWTI) at intervals specified by a count value set in advance.

The interval time can be selected by the WTM.WTM4 to WTM.WTM7 bits.

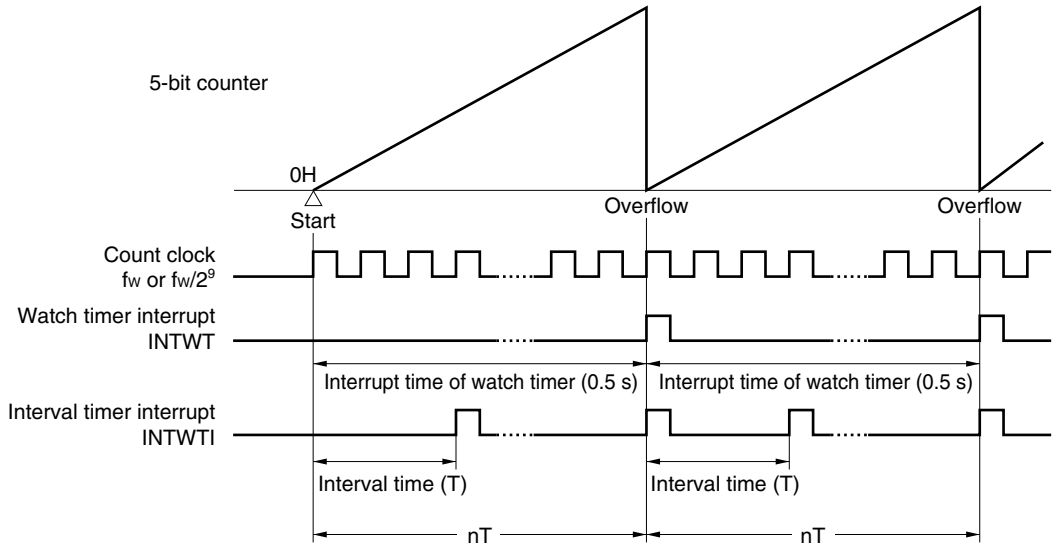
Table 10-1. Interval Time of Interval Timer

WTM7	WTM6	WTM5	WTM4	Interval Time	
0	0	0	0	$2^4 \times 1/f_W$	488 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	0	1	$2^5 \times 1/f_W$	977 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	0	$2^6 \times 1/f_W$	1.95 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	1	$2^7 \times 1/f_W$	3.91 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	0	$2^8 \times 1/f_W$	7.81 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	1	$2^9 \times 1/f_W$	15.6 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	0	$2^{10} \times 1/f_W$	31.3 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	1	$2^{11} \times 1/f_W$	62.5 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
1	0	0	0	$2^4 \times 1/f_W$	488 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	0	1	$2^5 \times 1/f_W$	977 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	0	$2^6 \times 1/f_W$	1.95 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	1	$2^7 \times 1/f_W$	3.91 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	0	$2^8 \times 1/f_W$	7.81 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	1	$2^9 \times 1/f_W$	15.6 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	0	$2^{10} \times 1/f_W$	31.3 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	1	$2^{11} \times 1/f_W$	62.5 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)

Remark f_W: Watch timer clock frequency

Downloaded from Arrow.com.

Figure 10-3. Operation Timing of Watch Timer/Interval Timer



Remarks 1. Assuming that the interrupt time of the watch timer is set to 0.5 seconds.

2. fw: Watch timer clock frequency

Values in parentheses apply when count clock fw = 32.768 kHz.

n: Number of interval timer WT operations

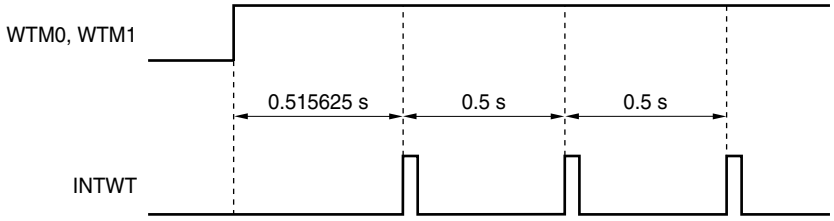
10.3 Cautions

(1) Operation as watch timer

Some time is required before the first watch timer interrupt request (INTWT) is generated after operation is enabled (WTM.WTM1 and WTM.WTM0 bits = 11).

Figure 10-4. Example of Generation of Watch Timer Interrupt Request (INTWT) (When Interrupt Period = 0.5 s)

It takes 0.515625 (max.) seconds for the first INTWT to be generated ($2^9 \times 1/32768 = 0.015625$ (max.) seconds longer). INTWT is then generated every 0.5 seconds.



Downloaded from Arrow.com.

(2) When watch timer and interval timer BRG operate simultaneously

When using the subclock as the count clock for the watch timer, the interval time of interval timer BRG can be set to any value. Changing the interval time does not affect the watch timer (before changing the interval time, stop operation).

When using the main clock as the count clock for the watch timer, set the interval time of interval timer BRG to approximately 65.536 kHz. Do not change this value.

(3) When interval timer BRG and interval timer WT operate simultaneously

When using the subclock as the count clock for interval timer WT, the interval times of interval timers BRG and WT can be set to any values. They can also be changed later (before changing the value, stop operation).

When using the main clock as the count clock for interval timer WT, the interval time of interval timer BRG can be set to any value, but cannot be changed later (it can be changed only when interval timer WT stops operation). The interval time of interval timer WT can be set to $\times 2^5$ to $\times 2^{12}$ of the set value of interval timer BRG. It can also be changed later.

(4) When watch timer and interval timer WT operate simultaneously

The interval time of interval timer WT can be set to a value between 488 μs and 62.5 ms. It cannot be changed later.

Do not stop interval timer WT (clear (0) the WTM.WTM0 bit) while the watch timer is operating. If the WTM0 bit is set (1) after it had been cleared (0), the watch timer will have a discrepancy of up to 0.5 or 0.25 seconds.

(5) When watch timer, interval timer BRG, and interval timer WT operate simultaneously

When using the subclock as the count clock for the watch timer, the interval times of interval timers BRG and WT can be set to any values. The interval time of interval timer BRG can be changed later (before changing the value, stop operation).

When using the main clock as the count clock for the watch timer, set the interval time of interval timer BRG to approximately 65.536 kHz. It cannot be changed later. The interval time of interval timer WT can be set to a value between 488 μs and 62.5 ms. It cannot be changed later.

Do not stop interval timer BRG (clear (0) the PRSM.BGCE bit) or interval timer WT (clear (0) the WTM.WTM0 bit) while the watch timer is operating.

Downloaded from Arrow.com.

CHAPTER 11 WATCHDOG TIMER FUNCTIONS

11.1 Watchdog Timer 1

11.1.1 Functions

Watchdog timer 1 has the following operation modes.

- Watchdog timer
- Interval timer

The following functions are realized from the above-listed operation modes.

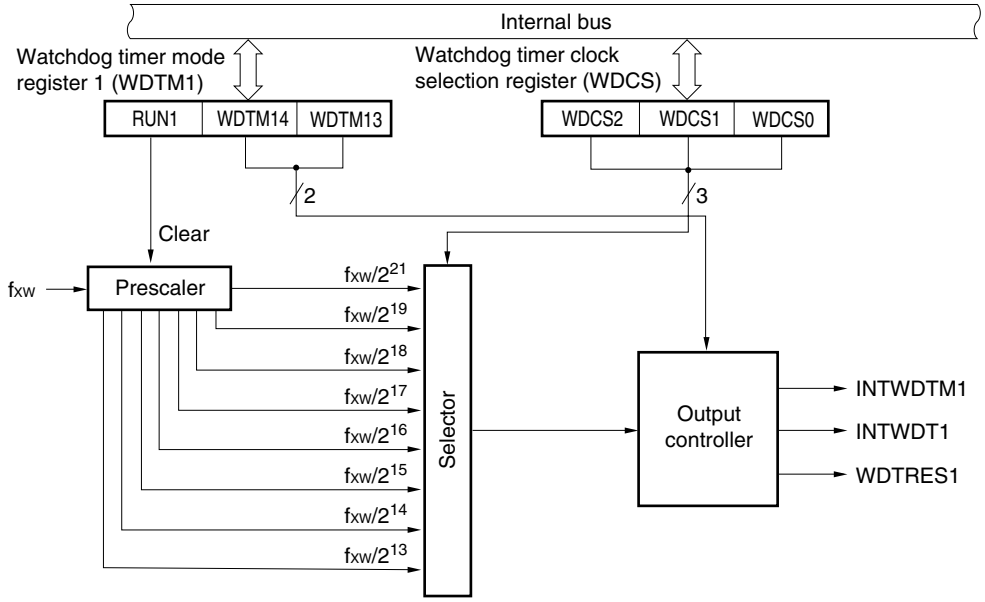
- Generation of non-maskable interrupt request signal (INTWDT1) upon overflow of watchdog timer 1[Note]
- Generation of system reset signal (WDTRES1) upon overflow of watchdog timer 1
- Generation of maskable interrupt request signal (INTWDTM1) upon overflow of interval timer

Note For non-maskable interrupt servicing due to non-maskable interrupt request signal (INTWDT1, INTWDT2), refer to **17.10 Cautions**.

Remark Select whether to use watchdog timer 1 in the watchdog timer 1 mode or the interval timer mode with the WDTM1 register.

Downloaded from Arrow.com.

Figure 11-1. Block Diagram of Watchdog Timer 1



Remark INTWDTM1: Request signal for maskable interrupt through watchdog timer 1 overflow
INTWDT1: Request signal for non-maskable interrupt through watchdog timer 1 overflow
WDTRES1: Reset signal through watchdog timer 1 overflow
$f_{XW} = f_X$: Watchdog timer 1 clock frequency

Downloaded from Arrow.com.

11.1.2 Configuration

Watchdog timer 1 includes the following hardware.

Table 11-1. Configuration of Watchdog Timer 1

Item	Configuration
Control register	Watchdog timer clock selection register (WDCS) Watchdog timer mode register 1 (WDTM1)

11.1.3 Registers

The registers that control watchdog timer 1 are as follows.

- Watchdog timer clock selection register (WDCS)
- Watchdog timer mode register 1 (WDTM1)

(1) Watchdog timer clock selection register (WDCS)

This register sets the overflow time of watchdog timer 1 and the interval timer.

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

Reset sets WDCS to 00H.

After reset: 00H R/W Address: FFFFF6C1H

	7	6	5	4	3	2	1	0
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer 1/interval timer			
				f_{XW}		
				4 MHz	5 MHz	10 MHz
0	0	0	$2^{13}/f_{XW}$	2.048 ms	1.638 ms	0.819 ms
0	0	1	$2^{14}/f_{XW}$	4.096 ms	3.277 ms	1.638 ms
0	1	0	$2^{15}/f_{XW}$	8.192 ms	6.554 ms	3.277 ms
0	1	1	$2^{16}/f_{XW}$	16.38 ms	13.11 ms	6.554 ms
1	0	0	$2^{17}/f_{XW}$	32.77 ms	26.21 ms	13.11 ms
1	0	1	$2^{18}/f_{XW}$	65.54 ms	52.43 ms	26.2 ms
1	1	0	$2^{19}/f_{XW}$	131.1 ms	104.9 ms	52.43 ms
1	1	1	$2^{21}/f_{XW}$	524.3 ms	419.4 ms	209.7 ms

Remark $f_{XW} = f_X$: Watchdog timer 1 clock frequency

Downloaded from Arrow.com.

(2) Watchdog timer mode register 1 (WDTM1)

This register sets the watchdog timer 1 operation mode and enables/disables count operations.

This register is a special register that can be written only in a special sequence (refer to **3.4.7 Special registers**).

The WDTM1 register can be read or written in 8-bit or 1-bit units.

Reset sets WDTM1 to 00H.

Caution When the main clock is stopped and the CPU is operating on the subclock, do not access the WDTM1 register.
For details, refer to 3.4.8 (1) (b).

After reset: 00H R/W Address: FFFFF6C2H

	<7>	6	5	4	3	2	1	0
WDTM1	RUN1	0	0	WDTM14	WDTM13	0	0	0

RUN1	Selection of operation mode of watchdog timer 1[Note 1]
0	Stop counting
1	Clear counter and start counting

WDTM14	WDTM13	Selection of operation mode of watchdog timer 1[Note 2]
0	0	Interval timer mode (Upon overflow, maskable interrupt INTWDTM1 is generated.)
0	1	
1	0	Watchdog timer mode 1[Note 3] (Upon overflow, non-maskable interrupt INTWDT1 is generated.)
1	1	Watchdog timer mode 2 (Upon overflow, reset operation WDTRES1 is started.)

Notes 1. Once the RUN1 bit is set (to 1), it cannot be cleared (to 0) by software.
Therefore, when counting is started, it cannot be stopped except reset.

2. Once the WDTM13 and WDTM14 bits are set (to 1), they cannot be cleared (to 0) by software and can be cleared only by reset.

3. For non-maskable interrupt servicing due to non-maskable interrupt request signal (INTWDT1), refer to **17.10 Cautions**.

Downloaded from Arrow.com.

11.1.4 Operation

(1) Operation as watchdog timer 1

Watchdog timer 1 operation to detect a program loop is selected by setting the WDTM1.WDTM14 bit to 1.

The count clock (program loop detection time interval) of watchdog timer 1 can be selected using the WDCS.WDCS0 to WDCS.WDCS2 bits. The count operation is started by setting the WDTM1.RUN1 bit to 1. When, after the count operation is started, the RUN1 bit is again set to 1 within the set program loop detection time interval, watchdog timer 1 is cleared and the count operation starts again.

If the program loop detection time is exceeded without RUN1 bit being set to 1, reset signal (WDTRES1) through the value of the WDTM1.WDTM13 bit or a non-maskable interrupt request signal (INTWDT1) is generated.

The count operation of watchdog timer 1 stops in the STOP mode and IDLE mode. Set the RUN1 bit to 1 before the STOP mode or IDLE mode is entered in order to clear watchdog timer 1.

Because watchdog timer 1 operates in the HALT mode, make sure that an overflow will not occur during HALT.

Cautions 1. When the subclock is selected for the CPU clock, the count operation of watchdog timer 1 is stopped (the value of watchdog timer 1 is maintained).

2. For non-maskable interrupt servicing due to the INTWDT1 signal, refer to 17.10 Cautions.

Table 11-2. Program Loop Detection Time of Watchdog Timer 1

Clock	Program Loop Detection Time		
	f_{XW} = 4 MHz	f_{XW} = 5 MHz	f_{XW} = 10 MHz
$2^{13}/f_{XW}$	2.048 ms	1.638 ms	0.819 ms
$2^{14}/f_{XW}$	4.096 ms	3.277 ms	1.683 ms
$2^{15}/f_{XW}$	8.192 ms	6.554 ms	3.277 ms
$2^{16}/f_{XW}$	16.38 ms	13.11 ms	6.554 ms
$2^{17}/f_{XW}$	32.77 ms	26.21 ms	13.11 ms
$2^{18}/f_{XW}$	65.54 ms	52.43 ms	26.21 ms
$2^{19}/f_{XW}$	131.1 ms	104.9 ms	52.43 ms
$2^{21}/f_{XW}$	524.3 ms	419.4 ms	209.7 ms

Remark $f_{XW} = f_X$: Watchdog timer 1 clock frequency

Downloaded from Arrow.com.

(2) Operation as interval timer

Watchdog timer 1 can be made to operate as an interval timer that repeatedly generates interrupts using the count value set in advance as the interval, by clearing the WDTM1.WDTM14 bit to 0.

When watchdog timer 1 operates as an interval timer, the interrupt mask flag (WDTMK) and priority specification flags (WDTPR0 to WDTPR2) of the WDTIC register are valid and maskable interrupt request signals (INTWDTM1) can be generated. The default priority of the INTWDTM1 signal is set to the highest level among the maskable interrupt request signals.

The interval timer continues to operate in the HALT mode, but it stops operating in the STOP mode and the IDLE mode.

Cautions 1. Once the WDTM14 bit is set to 1 (thereby selecting the watchdog timer 1 mode), the interval timer mode is not entered as long as reset is not performed.

2. When the subclock is selected for the CPU clock, the count operation of the watchdog timer 1 stops (the value of the watchdog timer is maintained).

Table 11-3. Interval Time of Interval Timer

Clock	Interval Time		
	f_{XW} = 4 MHz	f_{XW} = 5 MHz	f_{XW} = 10 MHz
$2^{13}/f_{XW}$	2.048 ms	1.638 ms	0.819 ms
$2^{14}/f_{XW}$	4.096 ms	3.277 ms	1.638 ms
$2^{15}/f_{XW}$	8.192 ms	6.554 ms	3.277 ms
$2^{16}/f_{XW}$	16.38 ms	13.11 ms	6.554 ms
$2^{17}/f_{XW}$	32.77 ms	26.21 ms	13.11 ms
$2^{18}/f_{XW}$	65.54 ms	52.43 ms	26.21 ms
$2^{19}/f_{XW}$	131.1 ms	104.9 ms	52.43 ms
$2^{21}/f_{XW}$	524.3 ms	419.4 ms	209.7 ms

Remark $f_{XW} = f_X$: Watchdog timer 1 clock frequency

Downloaded from Arrow.com.

11.2 Watchdog Timer 2

11.2.1 Functions

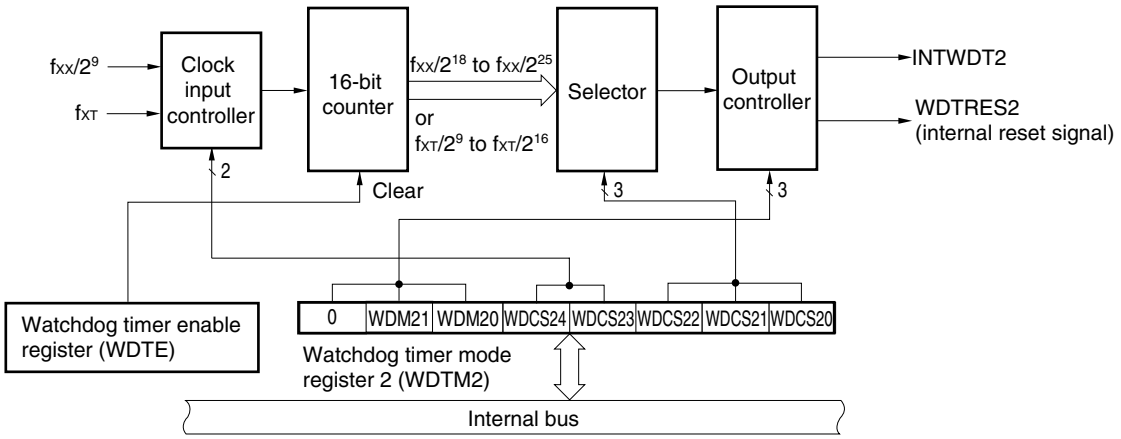
Watchdog timer 2 has the following functions.

- Default start watchdog timer[Note 1]
  → Reset mode: Reset operation upon overflow of watchdog timer 2 (generation of WDTRES2 signal)
  → Non-maskable interrupt request mode: NMI operation upon overflow of watchdog timer 2 (generation of INTWDT2 signal)[Note 2]
- Input selectable from main clock and subclock as the source clock

Notes 1. Watchdog timer 2 automatically starts in the reset mode following reset release.
When watchdog timer 2 is not used, either stop its operation before reset is executed through this function, or clear once watchdog timer 2 and stop it within the next interval time.
Also, write to the WDTM2 register for verification purposes only once, even if the default settings (reset mode, interval time: $f_{XX}/2^{25}$) need not be changed.

2. For non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT2), refer to **17.10 Cautions**.

Figure 11-2. Block Diagram of Watchdog Timer 2



Remark f_{XX}: Main clock frequency
f_{XT}: Subclock frequency
INTWDT2: Non-maskable interrupt request signal through watchdog timer 2
WDTRES2: Watchdog timer 2 reset signal

Downloaded from Arrow.com.

11.2.2 Configuration

Watchdog timer 2 includes the following hardware.

Table 11-4. Configuration of Watchdog Timer 2

Item	Configuration
Control register	Watchdog timer mode register 2 (WDTM2) Watchdog timer enable register (WDTE)

11.2.3 Registers

(1) Watchdog timer mode register 2 (WDTM2)

This register sets the overflow time and operation clock of watchdog timer 2.

The WDTM2 register can be read or written in 8-bit units. This register can be read any number of times, but it can be written only once following reset release.

Reset sets WDTM2 to 67H.

Caution When the main clock is stopped and the CPU is operating on the subclock, do not access the WDTM2 register.
For details, refer to 3.4.8 (1) (b).

After reset: 67H R/W Address: FFFFF6D0H

	7	6	5	4	3	2	1	0
WDTM2	0	WDM21	WDM20	WDCS24	WDCS23	WDCS22	WDCS21	WDCS20

WDM21	WDM20	Selection of operation mode of watchdog timer 2
0	0	Stops operation
0	1	Non-maskable interrupt request mode (generation of INTWDT2)
1	–	Reset mode (generation of WDTRES2)

Cautions 1. To stop the operation of watchdog timer 2, write "1FH" to the WDTM2 register.

2. For details about bits WDCS0 to WDCS4, refer to Table 11-5 Watchdog Timer 2 Clock Selection.

3. If the WDTM2 register is written twice after a reset, an overflow signal is forcibly output.

<R> **4. To intentionally generate an overflow signal, write data to the WDTM2 register only twice, or write a value other than "ACH" to the WDTE register only once.**

Downloaded from Arrow.com.

Table 11-5. Watchdog Timer 2 Clock Selection

WDCS24	WDCS23	WDCS22	WDCS21	WDCS20	Selected Clock	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 10 MHz
0	0	0	0	0	$2^{18}/f_{XX}$	13.1 ms	16.4 ms	26.2 ms
0	0	0	0	1	$2^{19}/f_{XX}$	26.2 ms	32.8 ms	52.4 ms
0	0	0	1	0	$2^{20}/f_{XX}$	52.4 ms	65.5 ms	104.9 ms
0	0	0	1	1	$2^{21}/f_{XX}$	104.9 ms	131.1 ms	209.7 ms
0	0	1	0	0	$2^{22}/f_{XX}$	209.7 ms	262.1 ms	419.4 ms
0	0	1	0	1	$2^{23}/f_{XX}$	419.4 ms	524.3 ms	838.9 ms
0	0	1	1	0	$2^{24}/f_{XX}$	838.9 ms	1048.6 ms	1677.7 ms
0	0	1	1	1	$2^{25}/f_{XX}$	1677.7 ms	2097.2 ms	3355.4 ms
0	1	0	0	0	$2^{9}/f_{XT}$	15.625 ms (f_{XT} = 32.768 kHz)		
0	1	0	0	1	$2^{10}/f_{XT}$	31.25 ms (f_{XT} = 32.768 kHz)		
0	1	0	1	0	$2^{11}/f_{XT}$	62.5 ms (f_{XT} = 32.768 kHz)		
0	1	0	1	1	$2^{12}/f_{XT}$	125 ms (f_{XT} = 32.768 kHz)		
0	1	1	0	0	$2^{13}/f_{XT}$	250 ms (f_{XT} = 32.768 kHz)		
0	1	1	0	1	$2^{14}/f_{XT}$	500 ms (f_{XT} = 32.768 kHz)		
0	1	1	1	0	$2^{15}/f_{XT}$	1000 ms (f_{XT} = 32.768 kHz)		
0	1	1	1	1	$2^{16}/f_{XT}$	2000 ms (f_{XT} = 32.768 kHz)		
1	×	×	×	×	Operation stopped			

(2) Watchdog timer enable register (WDTE)

The counter of watchdog timer 2 is cleared and counting restarted by writing “ACH” to the WDTE register.

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

Reset sets WDTE to 9AH.

After reset: 9AH R/W Address: FFFFF6D1H

	7	6	5	4	3	2	1	0
WDTE								

Cautions 1. When a value other than “ACH” is written to the WDTE register, an overflow signal is forcibly output.

2. When a 1-bit memory manipulation instruction is executed for the WDTE register, an overflow signal is forcibly output.

3. The read value of the WDTE register is always “9AH” (value that differs from written value “ACH”).

<R> **4. To intentionally generate an overflow signal, write a value other than “ACH” to the WDTE register only once, or write data to the WDTM2 register only twice.**

Downloaded from Arrow.com.

11.2.4 Operation

Watchdog timer 2 automatically starts in the reset mode following reset release.

The WDTM2 register can be written to only once following reset through byte access. To use watchdog timer 2, write the operation mode and the interval time to the WDTM2 register using 8-bit memory manipulation instructions. After this is done, the operation of watchdog timer 2 cannot be stopped.

The watchdog timer 2 program loop detection time interval can be selected by the WDTM2.WDCS24 to WDTM2.WDCS20 bits. Writing ACH to the WDTE register clears the counter of watchdog timer 2 and starts the count operation again. After the count operation starts, write ACH to the WDTE register within the set program loop detection time interval.

If the program loop detection time is exceeded without ACH being written to the WDTE register, a reset signal (WDTRES2) or non-maskable interrupt request signal (INTWDT2) is generated depending on the set value of the WDTM2.WDM21 and WDTM2.WDM20 bits.

To not use watchdog timer 2, write 1FH to the WDTM2 register.

For non-maskable interrupt servicing when the non-maskable interrupt request mode is set, refer to **17.10 Cautions**.

If the main clock is selected as the source clock of watchdog timer 2, the watchdog timer stops operation in the IDLE/STOP mode. Therefore, clear watchdog timer 2 by writing ACH to the WDTE register before the IDLE/STOP mode is set.

Because watchdog timer 2 operates in the HALT mode or when the subclock is selected as its source clock in the IDLE/STOP mode, exercise care that the timer does not overflow in the HALT mode.

Downloaded from Arrow.com.

CHAPTER 12 REAL-TIME OUTPUT FUNCTION (RTO)

12.1 Function

The real-time output function (RTO) transfers preset data to the RTBL0 and RTBH0 registers, and then transfers this data with hardware to an external device via the real-time output latches, upon occurrence of a timer interrupt. The pins through which the data is output to an external device constitute a port called a real-time output port.

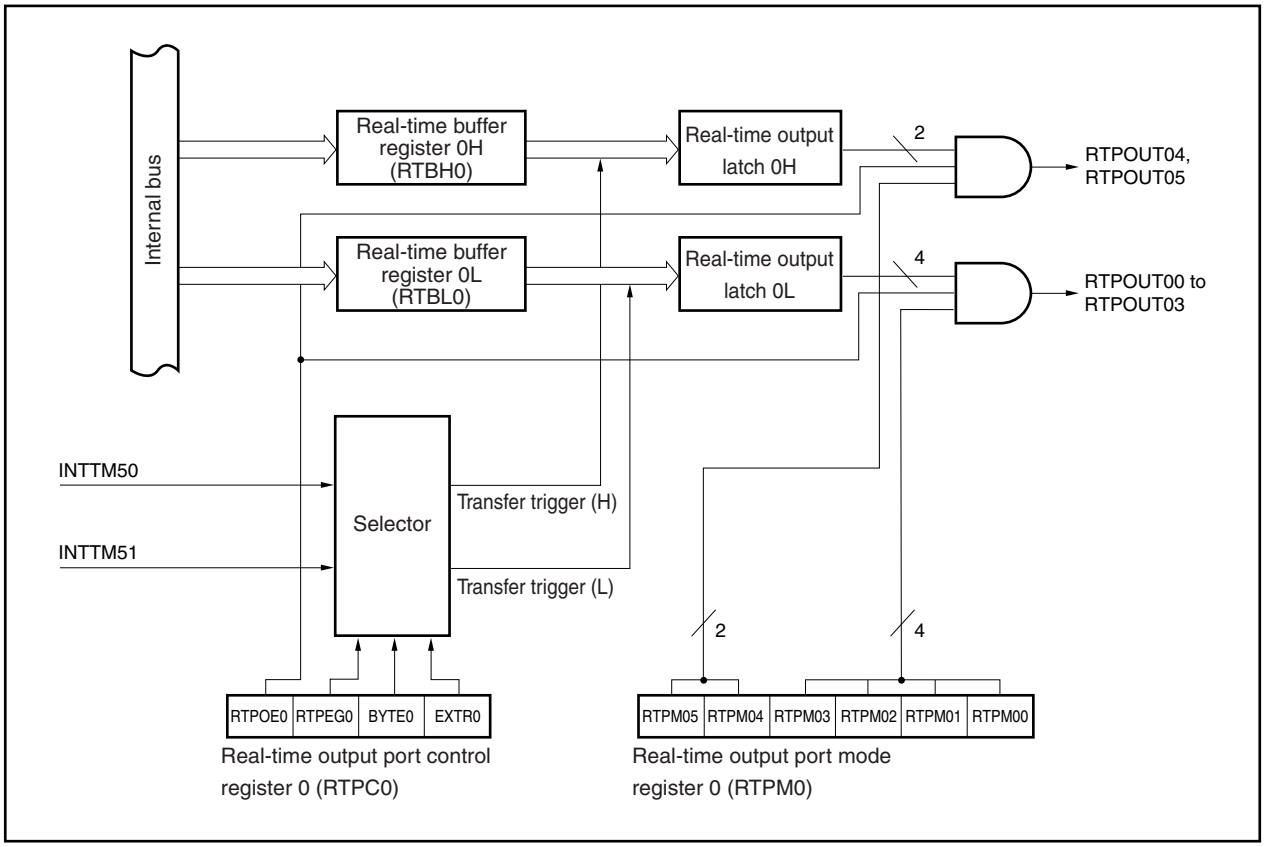
Because RTO can output signal without jitter, it is suitable for controlling a stepping motor.

In the V850ES/KE1, a 6-bit real-time output port channel is provided.

The real-time output port can be set in the port mode or real-time output port mode in 1-bit units.

The block diagram of RTO is shown below.

Figure 12-1. Block Diagram of RTO



Downloaded from Arrow.com.

12.2 Configuration

RTO includes the following hardware.

Table 12-1. Configuration of RTO

Item	Configuration
Registers	Real-time output buffer register 0 (RTBL0, RTBH0)
Control registers	Real-time output port mode register 0 (RTPM0) Real-time output port control register 0 (RTPC0)

(1) Real-time output buffer register 0 (RTBL0, RTBH0)

RTBL0 and RTBH0 are 4-bit registers that hold output data in advance.

These registers are mapped to independent addresses in the peripheral I/O register area.

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

If an operation mode of 4 bits × 1 channel or 2 bits × 1 channel is specified (RTPC0.BYTE0 bit = 0), data can be individually set to the RTBL0 and RTBH0 registers. The data of both these registers can be read at once by specifying the address of either of these registers.

If an operation mode of 6 bits × 1 channel is specified (BYTE0 bit = 1), 8-bit data can be set to both the RTBL0 and RTBH0 registers by writing the data to either of these registers. Moreover, the data of both these registers can be read at once by specifying the address of either of these registers.

Table 12-2 shows the operation when the RTBL0 and RTBH0 registers are manipulated.

After reset: 00H R/W Address: RTBL0 FFFFF6E0H, RTBH0 FFFFF6E2H

	7	6	5	4	3	2	1	0
RTBL0					RTBL03	RTBL02	RTBL01	RTBL00
RTBH0	0	0	RTBH05	RTBH04				

Cautions 1. When writing to bits 6 and 7 of the RTBH0 register, always write 0.

2. When the main clock is stopped and the CPU is operating on the subclock, do not access the RTBL0 and RTBH0 registers using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

Table 12-2. Operation During Manipulation of RTBL0 and RTBH0 Registers

Operation Mode	Register to Be Manipulated	Read		Write[Note]	
		Higher 4 Bits	Lower 4 Bits	Higher 4 Bits	Lower 4 Bits
4 bits × 1 channel, 2 bits × 1 channel	RTBL0	RTBH0	RTBL0	Invalid	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	Invalid
6 bits × 1 channel	RTBL0	RTBH0	RTBL0	RTBH0	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	RTBL0

Note After setting the real-time output port, set output data to the RTBL0 and RTBH0 registers by the time a real-time output trigger is generated.

Downloaded from Arrow.com.

12.3 Registers

RTO is controlled using the following two types of registers.

- Real-time output port mode register 0 (RTPM0)
- Real-time output port control register 0 (RTPC0)

(1) Real-time output port mode register 0 (RTPM0)

This register selects the real-time output port mode or port mode in 1-bit units.

The RTPM0 register can be read or written in 8-bit or 1-bit units.

Reset sets RTPM0 to 00H.

After reset: 00H R/W Address: FFFFF6E4H

	7	6	5	4	3	2	1	0
RTPM0	0	0	RTPM05	RTPM04	RTPM03	RTPM02	RTPM01	RTPM00

RTPM0m	Control of real-time output port (m = 0 to 5)
0	Real-time output disabled
1	Real-time output enabled

Cautions 1. To reflect real-time output signals (RTPOUT00 to RTPOUT05) to the pins (RTP00 to RTP05), set them to the real-time output port with the PMC5 and PFC5 registers.

2. By enabling real-time output operation (RTPC0.RTPOE0 bit = 1), the bits specified as real-time output enabled perform real-time output, and the bits specified as real-time output disabled output 0.

3. If real-time output is disabled (RTPOE0 bit = 0), real-time output signals (RTPOUT00 to RTPOUT05) all output 0, regardless of the RTPM0 register setting.

Downloaded from Arrow.com.

(2) Real-time output port control register 0 (RTPC0)

This register sets the operation mode and output trigger of the real-time output port.

The relationship between the operation mode and output trigger of the real-time output port is as shown in Table 12-3.

The RTPC0 register can be read or written in 8-bit or 1-bit units.

Reset sets RTPC0 to 00H.

After reset: 00H R/W Address: FFFFF6E5H

	<7>	6	5	4	3	2	1	0
RTPC0	RTPOE0	RTPEG0Note 1	BYTE0	EXTR0Note 2	0	0	0	0

RTPOE0	Control of real-time output operation
0	Disables operationNote 3
1	Enables operation

BYTE0	Specification of channel configuration for real-time output
0	4 bits × 1 channel, 2 bits × 1 channel
1	6 bits × 1 channel

Notes 1. The value of the RTPEG0 bit does not affect the operation.

2. For the EXTR0 bit, refer to **Table 12-3**.

3. When real-time output operation is disabled (RTPOE0 bit = 0), real-time output signals (RTPOUT00 to RTPOUT05) all output 0.

Caution Perform the settings for the BYTE0 and EXTR0 bits only when the RTPOE0 bit = 0.

Table 12-3. Operation Modes and Output Triggers of Real-Time Output Port

BYTE0	EXTR0	Operation Mode	RTBH0 (RTP04, RTP05)	RTBL0 (RTP00 to RTP03)
0	0	4 bits × 1 channel, 2 bits × 1 channel	INTTM51	INTTM50
	1		INTTM50	No trigger
1	0	6 bits × 1 channel	INTTM50	
	1		Setting prohibited	

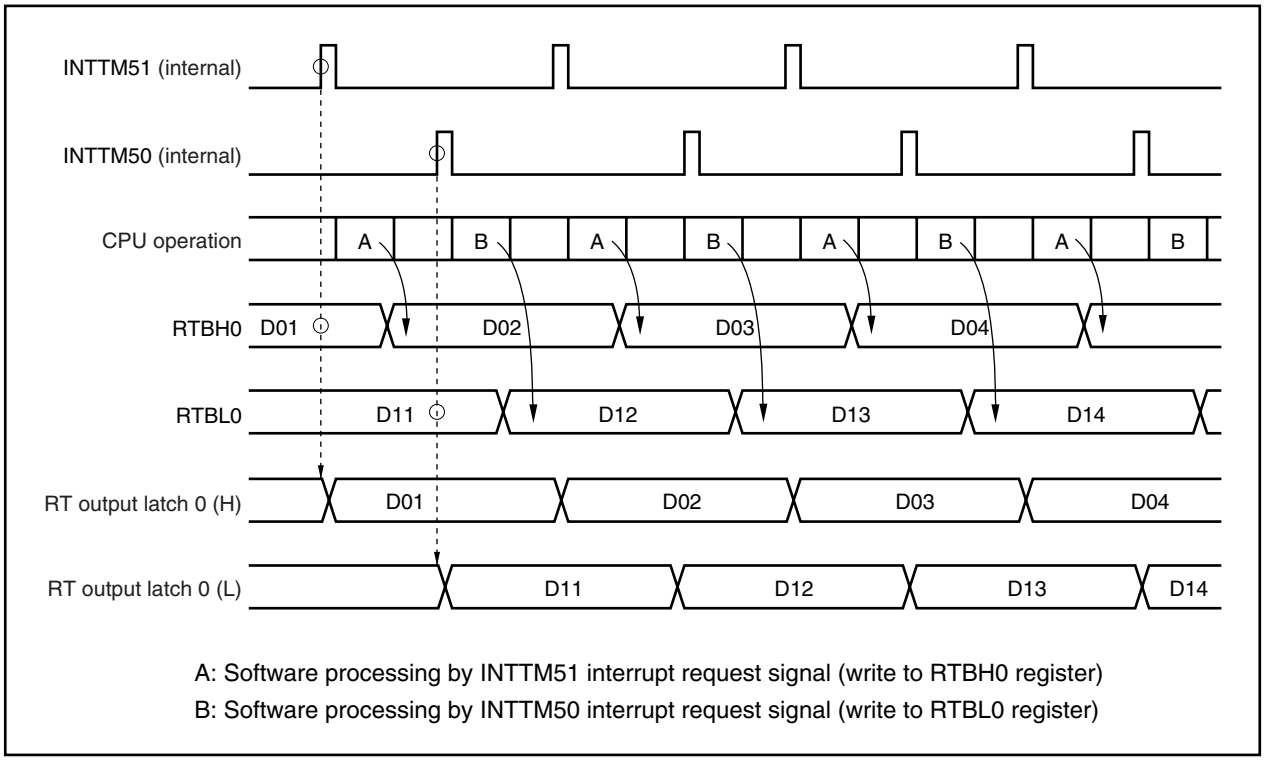
Downloaded from Arrow.com.

12.4 Operation

If the real-time output operation is enabled by setting the RTPC0.RTPOE0 bit to 1, the data of the RTBH0 and RTBL0 registers is transferred to the real-time output latch in synchronization with the generation of the selected transfer trigger (set by the RTPC0.EXTR0 and RTPC0.BYTE0 bits). Of the transferred data, only the data of the bits specified as real-time output enabled by the RTPM0 register is output from bits RTPOUT00 to RTPOUT05. The bits specified as real-time output disabled by the RTPM0 register output 0.

If the real-time output operation is disabled by clearing the RTPOE0 bit to 0, the RTPOUT00 to RTPOUT05 signals output 0 regardless of the setting of the RTPM0 register.

Figure 12-2. Example of Operation Timing of RTO0 (When EXTR0 and BYTE0 Bits = 00)



Remark For the operation during standby, refer to **CHAPTER 19 STANDBY FUNCTION**.

Downloaded from Arrow.com.

12.5 Usage

(1) Disable real-time output.
Clear the RTPC0.RTPOE0 bit to 0.

(2) Perform initialization as follows.
- Specify the real-time output port mode or port mode in 1-bit units.
  Set the RTPM0 register.
- Channel configuration: Select the trigger and valid edge.
  Set the RTPC0.EXTR0, RTPC0.BYTE0, and RTPC0.RTPEG0 bits.
- Set the initial values to the RTBH0 and RTBL0 registers[Note 1].

(3) Enable real-time output.
Set the RTPOE0 bit to 1.

(4) Set the next output value to the RTBH0 and RTBL0 registers by the time the selected transfer trigger is generated[Note 2].

(5) Set the next real-time output value to the RTBH0 and RTBL0 registers through interrupt servicing corresponding to the selected trigger.

Notes 1. If write to the RTBH0 and RTBL0 registers is performed when the RTPOE0 bit = 0, that value is transferred to real-time output latches 0H and 0L, respectively.

2. Even if write is performed to the RTBH0 and RTBL0 registers when the RTPOE0 bit = 1, data transfer to real-time output latches 0H and 0L is not performed.

Caution To reflect the real-time output signals (RTPOUT00 to RTPOUT05) to the pins, set the real-time output ports (RTP00 to RTP05) with the PMC5 and PFC5 registers.

12.6 Cautions

(1) Prevent the following conflicts by software.

- Conflict between real-time output disable/enable switching (RTPOE0 bit) and selected real-time output trigger
- Conflict between write to the RTBH0 and RTBL0 registers in the real-time output enabled status and the selected real-time output trigger.

(2) Before performing initialization, disable real-time output (RTPOE0 bit = 0).

(3) Once real-time output has been disabled (RTPOE0 bit = 0), be sure to initialize the RTBH0 and RTBL0 registers before enabling real-time output again (RTPOE0 bit = 0 → 1).

Downloaded from Arrow.com.

12.7 Security Function

A circuit that sets the pin outputs to high impedance as a security function for when malfunctions of a stepping motor controlled by RTO occur is provided on chip. It forcibly resets the pins allocated to RTP00 to RTP05 via external interrupt INTP0 pin edge detection, placing them in the high-impedance state.

The ports (P50 to P55 pins) placed in high impedance by INTP0[Note 1] pin are initialized[Note 2], so settings for these ports must be performed again.

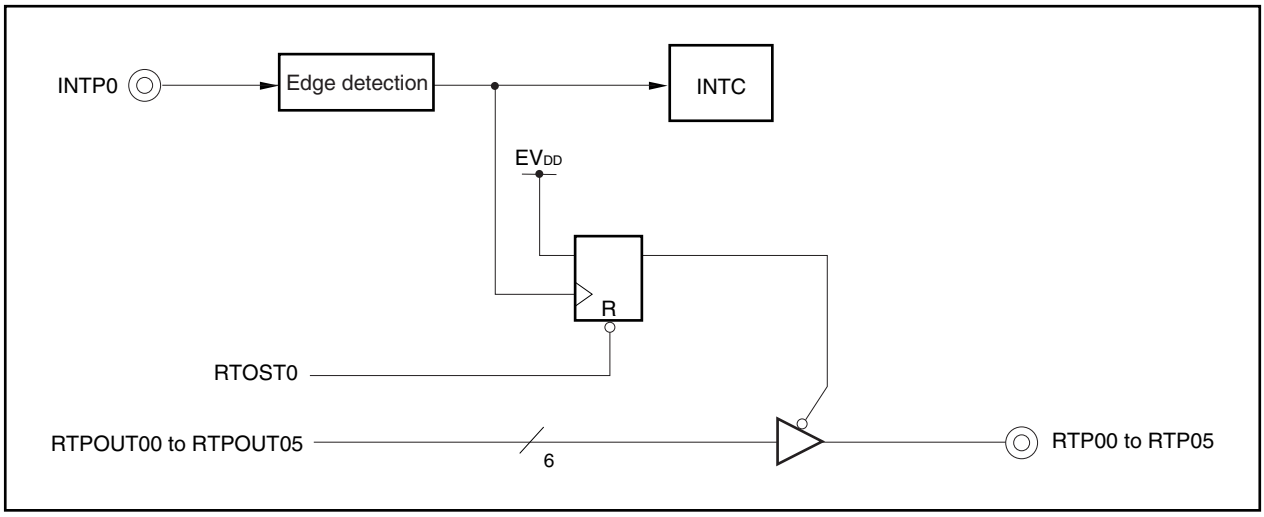
Notes 1. Regardless of the port settings, P50 to P55 pins are all placed in high impedance via the INTP0 pin.

2. The bits that are initialized are all the bits corresponding to P50 to P55 pins of the following registers.

- P5 register
- PM5 register
- PMC5 register
- PU5 register
- PFC5 register

The block diagram of the security function is shown below.

Figure 12-3. Block Diagram of Security Function



This function is set with the PLLCTL.RTOST0 bit.

Downloaded from Arrow.com.

(1) PLL control register (PLLCTL)

The PLLCTL register is an 8-bit register that controls the RTO security function and PLL.

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

Reset sets PLLCTL to 01H.

After reset: 01H R/W Address: FFFFF806H

	7	6	5	4	3	<2>	<1>	<0>
PLLCTL	0	0	0	0	0	RTOST0	SELPLL Note	PLLON Note

RTOST0	Control of RTP00 to RTP05 security function
0	INTP0 pin is not used as trigger for security function
1	INTP0 pin is used as trigger for security function

Note For details on the SELPLL and PLLON bits, refer to **CHAPTER 5 CLOCK GENERATION FUNCTION**.

Cautions 1. Before outputting a value to the real-time output ports (RTP00 to RTP05), select the INTP0 pin interrupt edge detection and then set the RTOST0 bit.

2. To set again the ports (P50 to P55 pins) as real-time output ports after placing them in high impedance via the INTP0 pin, first cancel the security function.

[Procedure to set ports again]

<1> Cancel the security function and enable port setting by clearing the RTOST0 bit to 0.

<2> Set the RTOST0 bit to 1 (only if required).

<3> Set again as real-time output port.

3. Be sure to clear bits 4 to 7 to "0". Changing bit 3 does not affect the operation.

Downloaded from Arrow.com.

CHAPTER 13 A/D CONVERTER

13.1 Function

The A/D converter converts analog input signals into digital values with a resolution of 10 bits and has an 8-channel (ANI0 to ANI7) configuration.

The A/D converter has the following functions.

(1) 10-bit resolution A/D conversion

1 analog input channel is selected from ANI0 to ANI7, and an A/D conversion operation with resolution of 10 bits is repeatedly executed. Every time A/D conversion is completed, an interrupt request signal (INTAD) is generated.

(2) Power fail detection function

This is a function to detect low voltage in a battery. The results of A/D conversion (the value in the ADCRH register) and the PFT register are compared, and INTAD signal is generated only when the comparison conditions match.

Downloaded from Arrow.com.

13.2 Configuration

The A/D converter consists of the following hardware.

Figure 13-1. Block Diagram of A/D Converter

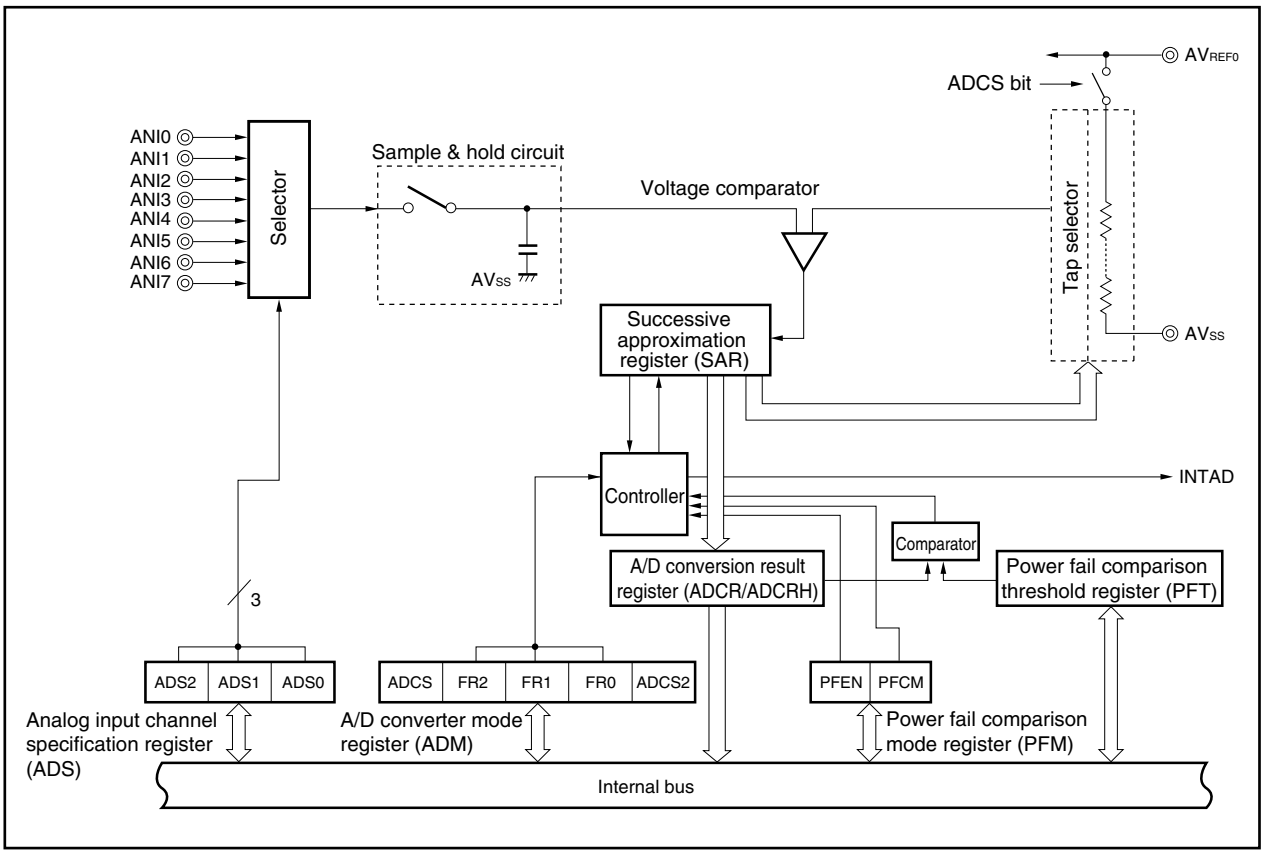


Table 13-1. Registers of A/D Converter Used by Software

Item	Configuration
Registers	A/D conversion result register (ADCR) A/D conversion result register H (ADCRH): Only higher 8 bits can be read Power fail comparison threshold register (PFT) A/D converter mode register (ADM) Analog input channel specification register (ADS) Power fail comparison mode register (PFM)

Downloaded from Arrow.com.

(1) ANI0 to ANI7 pins

These are analog input pins for the 8 channels of the A/D converter. They are used to input analog signals to be converted into digital signals. Pins other than those selected as analog input by the ADS register can be used as input ports.

(2) Sample & hold circuit

The sample & hold circuit samples the analog input signals selected by the input circuit and sends the sampled data to the voltage comparator. This circuit holds the sampled analog input voltage during A/D conversion.

(3) Series resistor string

The series resistor string is connected between AV_{REF0} and AV_{SS} and generates a voltage for comparison with the analog input signal.

(4) Voltage comparator

The voltage comparator compares the value that is sampled and held with the output voltage of the series resistor string.

(5) Successive approximation register (SAR)

This register compares the sampled analog voltage value with the voltage value from the series resistor string, and converts the comparison result starting from the most significant bit (MSB).

When the least significant bit (LSB) has been converted to a digital value (end of A/D conversion), the contents of the SAR register are transferred to the ADCR register.

The SAR register cannot be read or written directly.

(6) A/D conversion result register (ADCR), A/D conversion result register H (ADCRH)

Each time A/D conversion ends, the conversion results are loaded from the successive approximation register and the results of A/D conversion are held in the higher 10 bits of this register (the lower 6 bits are fixed to 0).

(7) Controller

The controller compares the A/D conversion results (the value of the ADCRH register) with the value of the PFT register when A/D conversion ends or the power fail detection function is used. It generates INTAD signal only when the comparison conditions match.

(8) AV_{REF0} pin

This is the analog power supply pin/reference voltage input pin of the A/D converter. Always use the same potential as the V_{DD} pin even when not using the A/D converter.

The signals input to the ANI0 to ANI7 pins are converted into digital signals based on the voltage applied across AV_{REF0} and AV_{SS}.

(9) AV_{SS} pin

This is the ground potential pin of the A/D converter. Always use the same potential as the V_{SS} pin even when not using the A/D converter.

Downloaded from Arrow.com.

(10) A/D converter mode register (ADM)

This register sets the conversion time of the analog input to be converted to a digital signal and the conversion operation start/stop.

(11) Analog input channel specification register (ADS)

This register specifies the input port for the analog voltage to be converted to a digital signal.

(12) Power fail comparison mode register (PFM)

This register sets the power fail monitoring mode.

(13) Power fail comparison threshold register (PFT)

This register sets the threshold to be compared with the ADCR register.

13.3 Registers

The A/D converter is controlled by the following registers.

- A/D converter mode register (ADM)
- Analog input channel specification register (ADS)
- Power fail comparison mode register (PFM)
- Power fail comparison threshold register (PFT)
- A/D conversion result register, A/D conversion result register H (ADCR, ADCRH)

Downloaded from Arrow.com.

<R> **(1) A/D converter mode register (ADM)**

This register sets the conversion time of the analog input signal to be converted into a digital signal as well as conversion start and stop.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF200H

	<7>	6	5	4	3	2	1	<0>
ADM	ADCS	0	FR2	FR1	FR0	0	0	ADCS2

ADCS	Control of A/D conversion operation
0	Conversion operation stopped
1	Conversion operation enabled

FR2	FR1	FR0	Selection of conversion time			
			Conversion time[Note 1]	fxx		
				20 MHz $AV_{REF0} \geq 4.5$ V	16 MHz $AV_{REF0} \geq 4.0$ V	10 MHz $AV_{REF0} \geq 2.7$ V
0	0	0	288/fxx	14.4 μs	18.0 μs	28.8 μs
0	0	1	240/fxx	Setting prohibited	15.0 μs	24.0 μs
0	1	0	192/fxx	Setting prohibited	Setting prohibited	19.2 μs
0	1	1	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
1	0	0	144/fxx	Setting prohibited	Setting prohibited	14.4 μs
1	0	1	120/fxx	Setting prohibited	Setting prohibited	Setting prohibited
1	1	0	96/fxx	Setting prohibited	Setting prohibited	Setting prohibited
1	1	1	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited

ADCS2	Control of reference voltage generator for boosting operation[Note 2]
0	Reference voltage generator operation stopped
1	Reference voltage generator operation stopped

Notes 1. Set the conversion time (time actually required for A/D conversion) as follows.

$4.0 \text{ V} \leq AV_{REF0} \leq 5.5$ V: 14 to 100 μs

$2.7 \text{ V} \leq AV_{REF0} < 4.0$ V: 17 to 100 μs

2. The operation of the reference voltage generator for boosting is controlled by the ADCS2 bit and it takes 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) after operation is started until it is stabilized. Therefore the ADCS bit is set to 1 (A/D conversion is started) at least 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) after if the ADCS2 bit was set to 1 (reference voltage generator for boosting is on), the first conversion result is valid.

Cautions 1. Be sure to clear bits 6, 2, and 1 to "0".

2. A write access to the ADM register, except the ADCS bit, is prohibited while the ADCS bit = 1.

3. When the main clock is stopped and the CPU is operating on the subclock, do not access the ADM register using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

Remark fxx: Main clock frequency

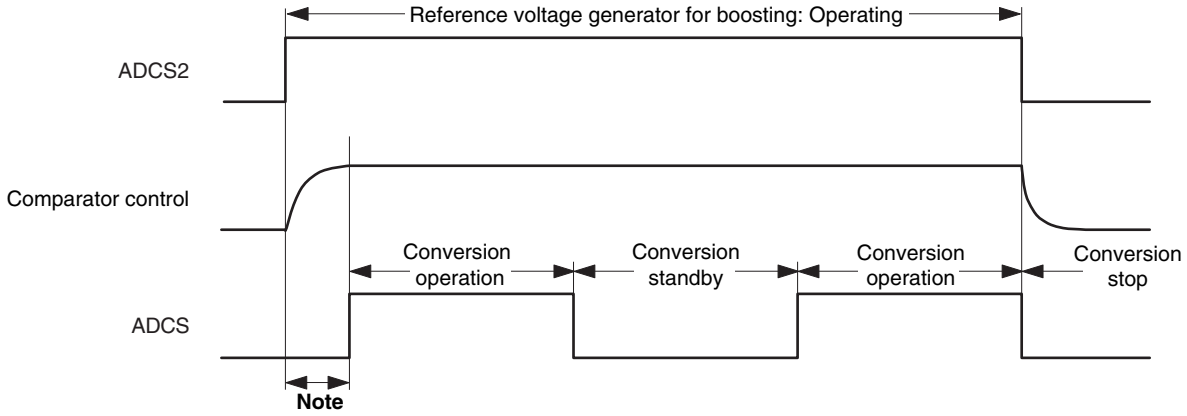
Downloaded from Arrow.com.

Table 13-2. Setting of ADCS Bit and ADCS2 Bit

ADCS	ADCS2	A/D Conversion Operation
0	0	Stopped status (DC power consumption path does not exist)
0	1	Conversion standby mode (only the reference voltage generator for boosting consumes power)
1	0	Conversion mode (reference voltage generator stops operation[Note])
1	1	Conversion mode (reference voltage generator is operating)

Note The data obtained by the first conversion must not be used.

Figure 13-2. Operation Sequence



Note 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) or more are required for the operation of the reference voltage generator for boosting between when the ADCS2 bit is set (1) and when the ADCS bit is set (1).

Downloaded from Arrow.com.

(2) Analog input channel specification register (ADS)

This register specifies the analog voltage input port for A/D conversion.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF201H

	7	6	5	4	3	2	1	0
ADS	0	0	0	0	0	ADS2	ADS1	ADS0

ADS2	ADS1	ADS0	Specification of analog input channel
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

Cautions 1. Be sure to clear bits 3 to 7 to "0".

2. When the main clock is stopped and the CPU is operating on the subclock, do not access the ADS register using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

<R> **3. Writing to the ADS register is prohibited during A/D conversion (ADM.ADCS = 1).**

Downloaded from Arrow.com.

(3) A/D conversion result register, A/D conversion result register H (ADCR, ADCRH)

The ADCR and ADCRH registers store the A/D conversion results.

These registers are read-only, in 16-bit or 8-bit units. However, specify the ADCR register for 16-bit access, and the ADCRH register for 8-bit access. In the ADCR register, the 10 bits of conversion results are read in the higher 10 bits and 0 is read in the lower 6 bits. In the ADCRH register, the higher 8 bits of the conversion results are read.

Reset makes these registers undefined.

After reset: Undefined R Address: FFFFF204H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0	0	0	0	0	0

After reset: Undefined R Address: FFFFF205H

	7	6	5	4	3	2	1	0
ADCRH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2

Caution When the main clock is stopped and the CPU is operating on the subclock, do not access the ADCR and ADCRH registers using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

Downloaded from Arrow.com.

The following shows the relationship between the analog input voltage input to the analog input pins (ANI0 to ANI7) and A/D conversion results (ADCR register).

$$SAR = INT\left(\frac{V_{IN}}{AV_{REF0}} \times 1024 + 0.5\right)$$

$$ADCR^{Note} = SAR \times 64$$

Or,

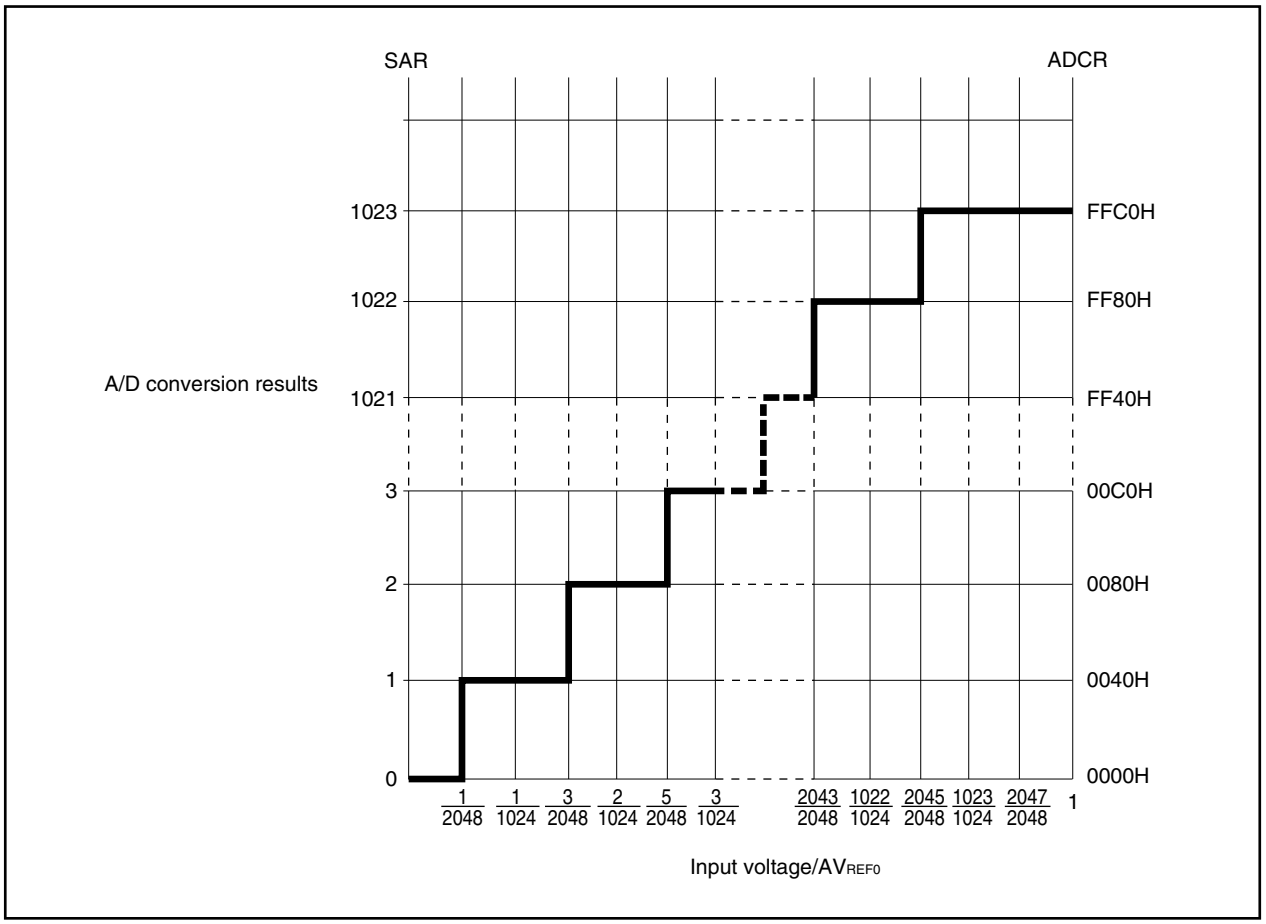
$$(SAR - 0.5) \times \frac{AV_{REF0}}{1024} \leq V_{IN} < (SAR + 0.5) \times \frac{AV_{REF0}}{1024}$$

INT ( ): Function that returns the integer part of the value in parentheses
V_{IN}: Analog input voltage
AV_{REF0}: Voltage of AV_{REF0} pin
ADCR: Value in the ADCR register

Note The lower 6 bits of the ADCR register are fixed to 0.

The following shows the relationship between the analog input voltage and A/D conversion results.

Figure 13-3. Relationship Between Analog Input Voltage and A/D Conversion Results



Downloaded from Arrow.com.

(4) Power fail comparison mode register (PFM)

This register sets the power fail monitoring mode.

The PFM register compares the value in the PFT register with the value of the ADCRH register.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF202H

	<7>	<6>	5	4	3	2	1	0
PFM	PFEN	PFCM	0	0	0	0	0	0

PFEN	Selection of power fail comparison enable/disable
0	Power fail comparison disabled
1	Power fail comparison enabled

PFCM	Selection of power fail comparison mode
0	Interrupt request signal (INTAD) generated when ADCR ≥ PFT
1	Interrupt request signal (INTAD) generated when ADCR < PFT

Cautions 1. When the main clock is stopped and the CPU is operating on the subclock, do not access the PFM register using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

<R> **2. Writing to the PFM register is prohibited during A/D conversion (ADM.ADCS = 1).**

(5) Power fail comparison threshold register (PFT)

The PFT register sets the comparison value in the power fail comparison mode.

The 8-bit data set in the PFT register is compared with the value of the ADCRH register.

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

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF203H

	7	6	5	4	3	2	1	0
PFT								

Cautions 1. When the main clock is stopped and the CPU is operating on the subclock, do not access the PFT register using an access method that causes a wait. For details, refer to 3.4.8 (1) (b).

<R> **2. Writing to the PFT register is prohibited during A/D conversion (ADM.ADCS = 1).**

Downloaded from Arrow.com.

13.4 Operation

13.4.1 Basic operation

<1> Select the channel whose analog signal is to be converted into a digital signal using the ADS register.

<2> Set (1) the ADM.ADCS2 bit and wait 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) or longer.

<3> Set the ADM.ADCS bit to 1 to start A/D conversion.
(Steps <4> to <10> are executed by hardware.)

<4> The sample & hold circuit samples the voltage input to the selected analog input channel.

<5> After sampling for a specific time, the sample & hold circuit enters the hold status and holds the input analog voltage until it has been converted into a digital signal.

<6> Set bit 9 of the successive approximation register (SAR). The tap selector sets the voltage tap of the series resistor string to $(1/2) \times AV_{REF0}$.

<7> The voltage comparator compares the voltage difference between the voltage tap of the series resistor string and the analog input voltage. If the analog input voltage is greater than $(1/2) \times AV_{REF0}$, the MSB of the SAR register remains set. If the analog input voltage is less than $(1/2) \times AV_{REF0}$, the MSB is reset.

<8> Next, bit 8 of the SAR register is automatically set and the next comparison starts. Depending on the value of bit 9 to which the result of the preceding comparison has been set, the voltage tap of the series resistor string is selected as follows.

- Bit 9 = 1: $(3/4) \times AV_{REF0}$
- Bit 9 = 0: $(1/4) \times AV_{REF0}$

The analog input voltage is compared with one of these voltage taps and bit 8 of the SAR register is manipulated as follows depending on the result of the comparison.

Analog input voltage ≥ voltage tap: Bit 8 = 1
Analog input voltage ≤ voltage tap: Bit 8 = 0

<9> The above steps are repeated until bit 0 of the SAR register has been manipulated.

<10> When comparison of all 10 bits of the SAR register has been completed, the valid digital value remains in the SAR register, and the value of the SAR register is transferred and latched to the ADCR register.
At the same time, an A/D conversion end interrupt request signal (INTAD) is generated.

<11> Repeat steps <4> to <10> until the ADCS bit is cleared to 0.
For another A/D conversion, start at <3>. However, when operating the A/D converter with the ADCS2 bit cleared to 0, start at <2>.

Downloaded from Arrow.com.

13.4.2 A/D conversion operation

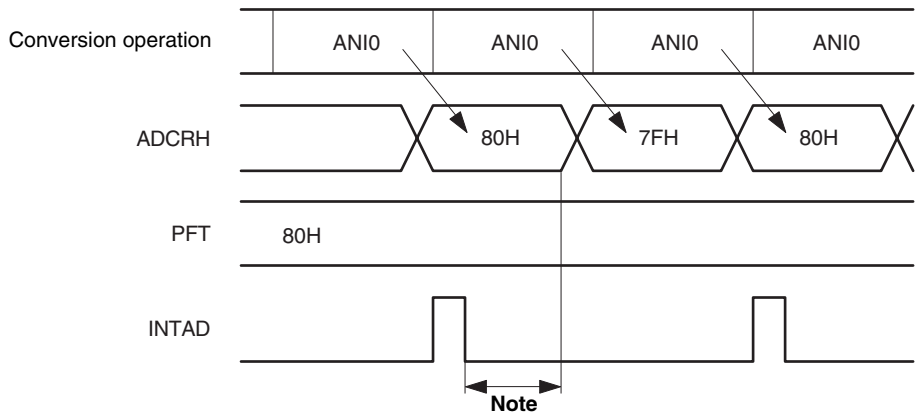
- Setting the ADM.ADCS bit to 1 starts conversion of the signal input to the channel specified by the ADS register. Upon completion of the conversion, the conversion result is stored in the ADCR register and a new conversion starts.
- <R> Writing to the ADM, ADS, PFT, and PFM registers during conversion is prohibited.
- If the ADCS bit is cleared to 0 during conversion, conversion is interrupted and the conversion operation is stopped.
- For whether or not the conversion end interrupt request signal (INTAD) is generated, refer to **13.4.3**.

13.4.3 Power fail monitoring function

The conversion end interrupt request signal (INTAD) can be controlled as follows using the PFM and PFT registers.

- If the PFM.PFEN bit = 0, the INTAD signal is generated each time conversion ends.
- If the PFEN bit = 1 and the PFM.PFCM bit = 0, the conversion result and the value of the PFT register are compared when conversion ends, and the INTAD signal is output only if ADCRH ≥ PFT.
- If the PFEN and PFCM bits = 1, the conversion result and the value of the PFT register are compared when conversion ends and the INTAD signal is output only if ADCRH < PFT.
- Because, when the PFEN bit = 1, the conversion result is overwritten after the INTAD signal has been output, unless the conversion result is read by the time the next conversion ends, in some cases it may appear as if the actual operation differs from the operation described above (refer to **Figure 13-4**).

Figure 13-4. Power Fail Monitoring Function (PFCM Bit = 0)



Note If reading is not performed during this interval, the conversion result changes to the next conversion result.

Downloaded from Arrow.com.

13.4.4 Setting method

The following describes how to set registers.

- When using the A/D converter for A/D conversion
  - <1> Set (1) the ADM.ADCS2 bit.
  - <2> Select the channel and conversion time by setting the ADS.ADS2 to ADS.ADS0 bits and the ADM.FR2 to ADM.FR0 bits.
  - <3> Set (1) the ADM.ADCS bit.
  - <4> Transfer the A/D conversion data to the ADCR register.
  - <5> An interrupt request signal (INTAD) is generated.

<Changing the channel>

  - <6> Change the channel by setting the ADS2 to ADS0 bits.
  - <7> Transfer the A/D conversion data to the ADCR register.
  - <8> An interrupt request signal (INTAD) is generated.

<Ending A/D conversion>

  - <9> Clear (0) the ADCS bit.
  - <10> Clear (0) the ADCS2 bit.

Cautions 1. The time taken from <1> to <3> must be 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) or longer.

2. Steps <1> and <2> may be reversed.

3. Step <1> may be omitted. However, if omitted, do not use the first conversion result after <3>.

4. The time taken from <4> to <7> is different from the conversion time set by the FR2 to FR0 bits.
The time taken for <6> and <7> is the conversion time set by the FR2 to FR0 bits.

- When using the A/D converter for the power fail function
  - <1> Set (1) the PFM.PFEN bit.
  - <2> Set the power fail comparison conditions by using the PFM.PFCM bit.
  - <3> Set (1) the ADM.ADCS2 bit.
  - <4> Select the channel and conversion time by setting the ADS.ADS2 to ADS.ADS0 bits and the ADM.FR2 to ADM.FR0 bits.
  - <5> Set the threshold value in the PFT register.
  - <6> Set (1) the ADM.ADCS bit.
  - <7> Transfer the A/D conversion data to the ADCR register.
  - <8> Compare the ADCR register with the PFT register. An interrupt request signal (INTAD) is generated when the conditions match.

<Changing the channel>

  - <9> Change the channel by setting the ADS2 to ADS0 bits.
  - <10> Transfer the A/D conversion data to the ADCR register.
  - <11> The ADCR register is compared with the PFT register. When the conditions match, an INTAD signal is generated.

<Ending A/D conversion>

  - <12> Clear (0) the ADCS bit.
  - <13> Clear (0) the ADCS2 bit.

Downloaded from Arrow.com.

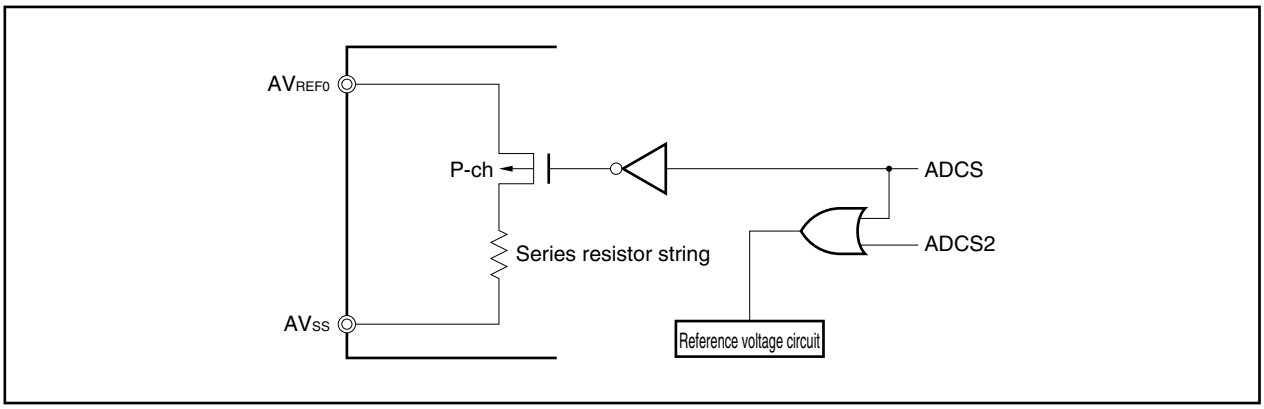
13.5 Cautions

(1) Power consumption in standby mode

The operation of the A/D converter stops in the standby mode. At this time, the power consumption can be reduced by stopping the conversion operation (ADM.ADCS bit = 0) and stopping the reference voltage circuit (ADM.ADCS2 bit = 0).

Figure 13-5 shows an example of how to reduce the power consumption in the standby mode.

<R> **Figure 13-5. Example of How to Reduce Power Consumption in Standby Mode**



(2) Input range of ANI0 to ANI7 pins

Use the A/D converter with the ANI0 to ANI7 pin input voltages within the specified range. If a voltage of AV_{REF0} or higher or AV_{SS} or lower (even if within the absolute maximum ratings) is input to these pins, the conversion value of the channel is undefined. Also, this may affect the conversion value of other channels.

(3) Conflicting operations

(a) Conflict between writing to the ADCR register and reading from ADCR register upon the end of conversion

Reading the ADCR register takes precedence. After the register has been read, a new conversion result is written to the ADCR register.

(b) Conflict between writing to the ADCR register and writing to the ADM register or writing to the ADS register upon the end of conversion

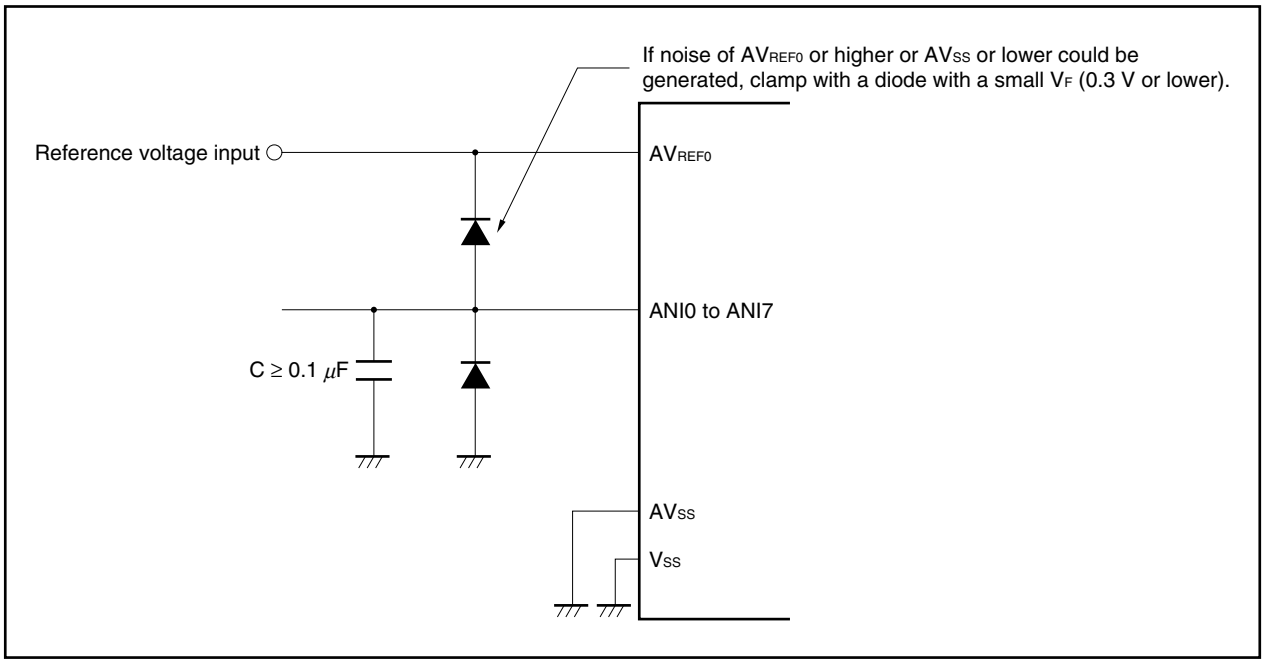
Writing to the ADM register or ADS register takes precedence. The ADCR register is not written, and neither is the conversion end interrupt request signal (INTAD) generated.

Downloaded from Arrow.com.

(4) Measures against noise

To keep a resolution of 10 bits, be aware of noise on the AV_{REF0} and ANI0 to ANI7 pins. The higher the output impedance of the analog input source, the greater the effect of noise. Therefore, it is recommended to connect external capacitors as shown in Figure 13-6 to reduce noise.

Figure 13-6. Handling of Analog Input Pins



(5) ANI0/P70 to ANI7/P77 pins

The analog input pins (ANI0 to ANI7) function alternately as input port pins (P70 to P77).

When performing A/D conversion by selecting any of the ANI0 to ANI7 pins, do not execute an input instruction to port 7 during conversion. This may decrease the conversion resolution.

If digital pulses are applied to the pin adjacent to the pin subject to A/D conversion, the value of the A/D conversion may differ from the expected value because of coupling noise. Therefore, do not apply pulses to the pin adjacent to the pin subject to A/D conversion.

(6) Input impedance of AV_{REF0} pin

A series resistor string of tens of kΩ is connected between the AV_{REF0} pin and AV_{SS} pin.

Therefore, if the output impedance of the reference voltage source is high, this will result in a series connection to the series resistor string between the AV_{REF0} pin and AV_{SS} pin, resulting in a large reference voltage error.

Downloaded from Arrow.com.

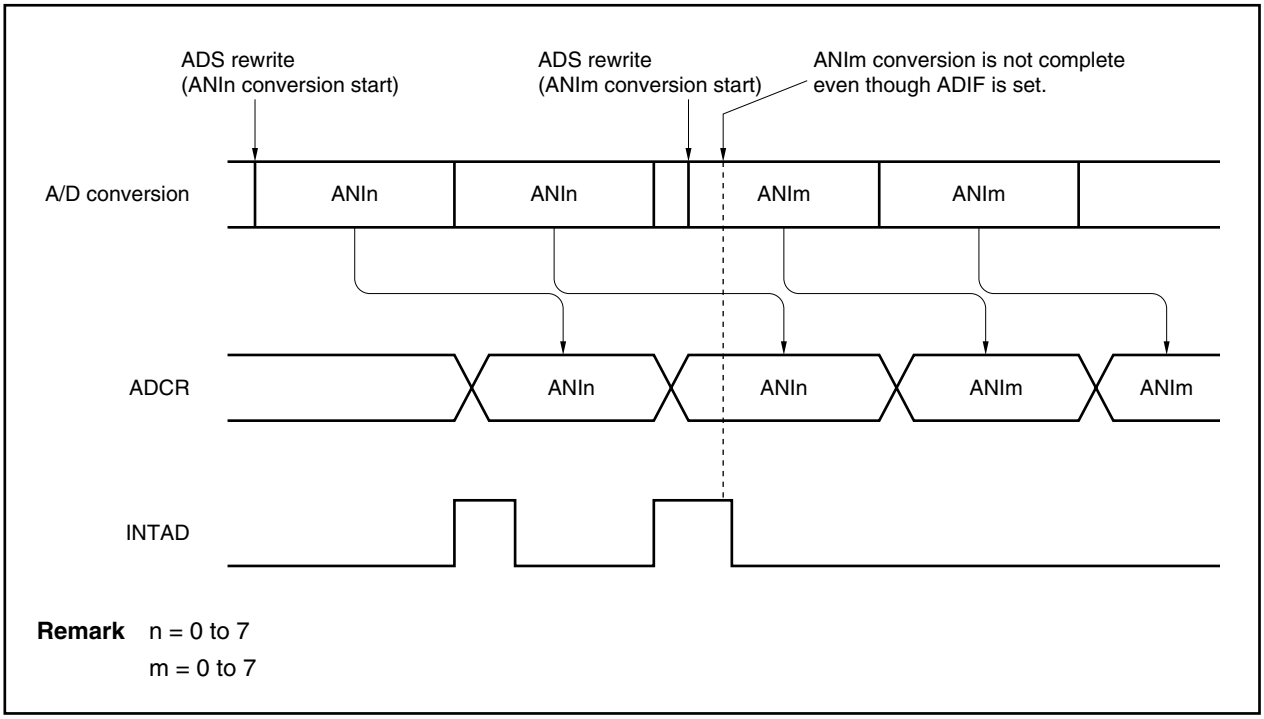
(7) Interrupt request flag (ADIC.ADIF bit)

Even when the ADS register is changed, the ADIF bit is not cleared (0).

Therefore, if the analog input pin is changed during A/D conversion, the ADIF bit may be set (1) because A/D conversion of the previous analog input pin ends immediately before the ADS register is rewritten. In a such case, note that if the ADIF bit is read immediately after the ADS register has been rewritten, the ADIF bit is set (1) even though A/D conversion of the analog input pin after the change has not been completed.

When stopping A/D conversion once and resuming it, clear the ADIF bit (0) before resuming A/D conversion.

Figure 13-7. A/D Conversion End Interrupt Request Occurrence Timing



Remark n = 0 to 7
m = 0 to 7

(8) Conversion results immediately after A/D conversion start

If the ADM.ADCS bit is set to 1 within 17 μs (14 μs when $AV_{REF0} \geq 4.0$ V) after the ADM.ADCS2 bit has been set to 1, or if the ADCS bit is set to 1 with the ADCS2 bit cleared to 0, the converted value immediately after the A/D conversion operation has started may not satisfy the rating. Take appropriate measures such as polling the A/D conversion end interrupt request signal (INTAD) and discarding the first conversion result.

(9) Reading A/D conversion result register (ADCR)

When the ADM or ADS register has been written, the contents of the ADCR register may become undefined. When the conversion operation is complete, read the conversion results before writing to the ADM or ADS register. A correct conversion result may not be able to be read at a timing other than the above.

When the CPU is operating on the subclock and main clock oscillation (fx) is stopped, do not read the ADCR register.

Downloaded from Arrow.com.

(10) A/D converter sampling time and A/D conversion start delay time

The A/D converter sampling time differs depending on the set value of the ADM register. A delay time exists until actual sampling is started after A/D converter operation is enabled.

When using a set in which the A/D conversion time must be strictly observed, care is required for the contents shown in Figure 13-8 and Table 13-3.

Figure 13-8. Timing of A/D Converter Sampling and A/D Conversion Start Delay

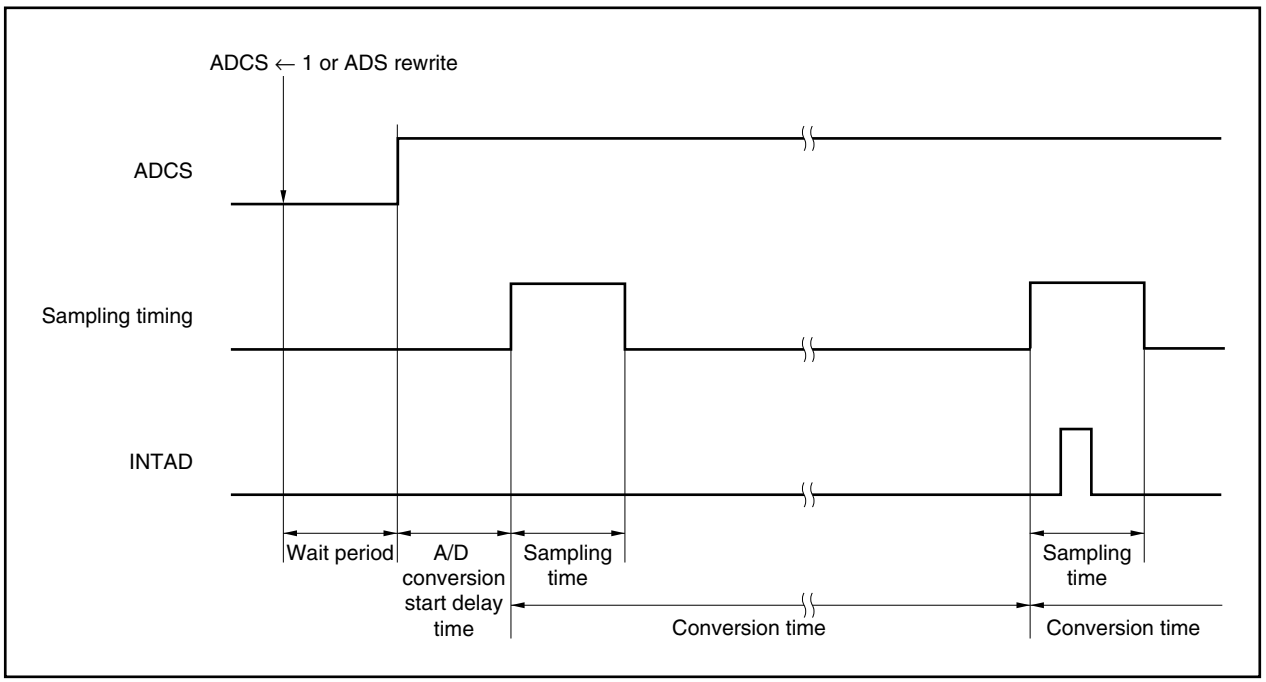


Table 13-3. A/D Converter Sampling Time and A/D Conversion Start Delay Time (ADM Register Set Value)

FR2	FR1	FR0	Conversion Time	Sampling Time	A/D Conversion Start Delay Time[Note]	
					MIN.	MAX.
0	0	0	288/fxx	40/fxx	11/fxx	12/fxx
0	0	1	240/fxx	32/fxx	11/fxx	12/fxx
0	1	0	192/fxx	24/fxx	10/fxx	11/fxx
1	0	0	144/fxx	20/fxx	9/fxx	10/fxx
1	0	1	120/fxx	16/fxx	9/fxx	10/fxx
1	1	0	96/fxx	12/fxx	11/fxx	12/fxx
Other than above			Setting prohibited	–	–	–

Note The A/D conversion start delay time is the time after the wait period. For the wait function, refer to **3.4.8 (1) (b) Access to special on-chip peripheral I/O register**.

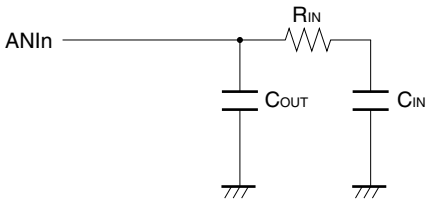
Remark fxx: Main clock frequency

Downloaded from Arrow.com.

(11) Internal equivalent circuit

The following shows the equivalent circuit of the analog input block.

Figure 13-9. Internal Equivalent Circuit of ANIn Pin



AV_{REF0}	R_{IN}	C_{OUT}	C_{IN}
4.5 V	3 kΩ	8 pF	15 pF
2.7 V	60 kΩ	8 pF	15 pF

Remarks 1. The above values are reference values.

2. n = 0 to 7

<R>

(12) Variation of A/D conversion results

The results of the A/D conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise. To reduce the variation, take counteractive measures with the program such as averaging the A/D conversion results.

<R>

(13) A/D conversion result hysteresis characteristics

The successive approximation type A/D converter holds the analog input voltage in the internal sample & hold capacitor and then performs A/D conversion. After the A/D conversion has finished, the analog input voltage remains in the internal sample & hold capacitor. As a result, the following phenomena may occur.

- When the same channel is used for A/D conversions, if the voltage is higher or lower than the previous A/D conversion, then hysteresis characteristics may appear where the conversion result is affected by the previous value. Thus, even if the conversion is performed at the same potential, the result may vary.
- When switching the analog input channel, hysteresis characteristics may appear where the conversion result is affected by the previous channel value. This is because one A/D converter is used for the A/D conversions. Thus, even if the conversion is performed at the same potential, the result may vary.

Therefore, to obtain more accurate conversion result, perform A/D conversion twice successively for the same channel, and discard the first conversion result.

<R>

(14) Writing to register during A/D conversion

Writing to the ADM, ADS, PFT, and PFM registers during A/D conversion is prohibited.

Downloaded from Arrow.com.

13.6 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

(1) Resolution

This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1 LSB (Least Significant Bit). The percentage of 1 LSB with respect to the full scale is expressed by %FSR (Full Scale Range). %FSR indicates the ratio of analog input voltage that can be converted as a percentage, and is always represented by the following formula regardless of the resolution.

1 %FSR = (Max. value of analog input voltage that can be converted – Min. value of analog input voltage that can be converted)/100
= (AV_{REF0} – 0)/100
= AV_{REF0}/100

1 LSB is as follows when the resolution is 10 bits.

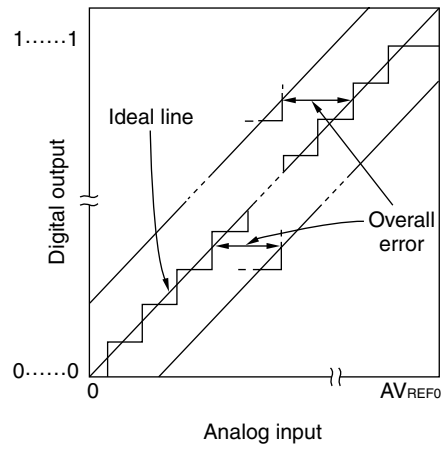
1 LSB = $1/2^{10}$ = 1/1024
= 0.098%FSR

Accuracy has no relation to resolution, but is determined by overall error.

(2) Overall error

This shows the maximum error value between the actual measured value and the theoretical value.
Zero-scale error, full-scale error, linearity error and errors that are combinations of these express the overall error.
Note that the quantization error is not included in the overall error in the characteristics table.

Figure 13-10. Overall Error



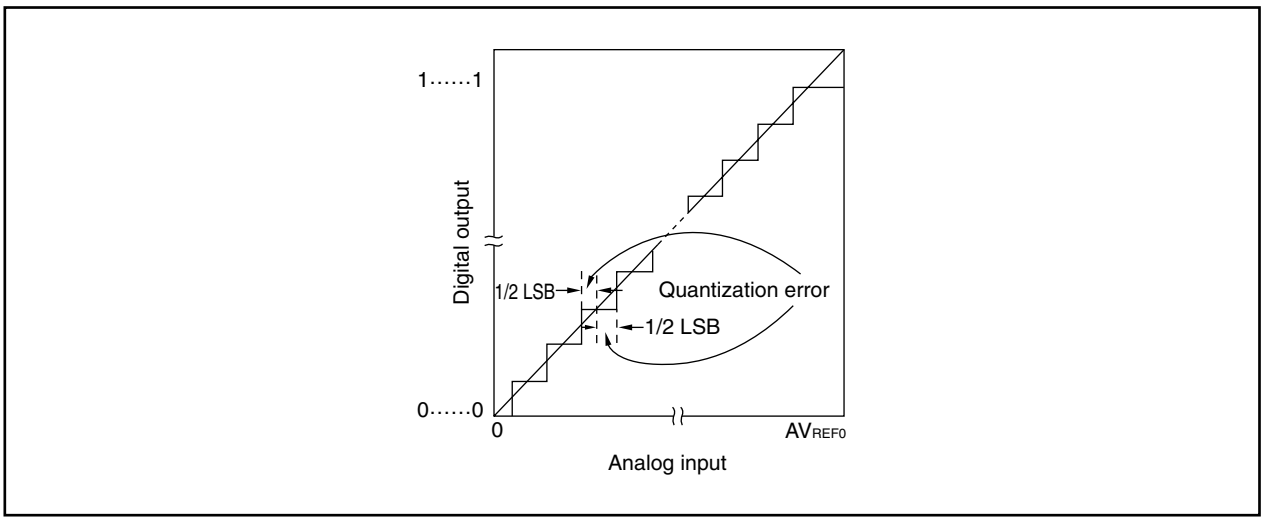
Downloaded from Arrow.com.

(3) Quantization error

When analog values are converted to digital values, a ±1/2 LSB error naturally occurs. In an A/D converter, an analog input voltage in a range of ±1/2 LSB is converted to the same digital code, so a quantization error cannot be avoided.

Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

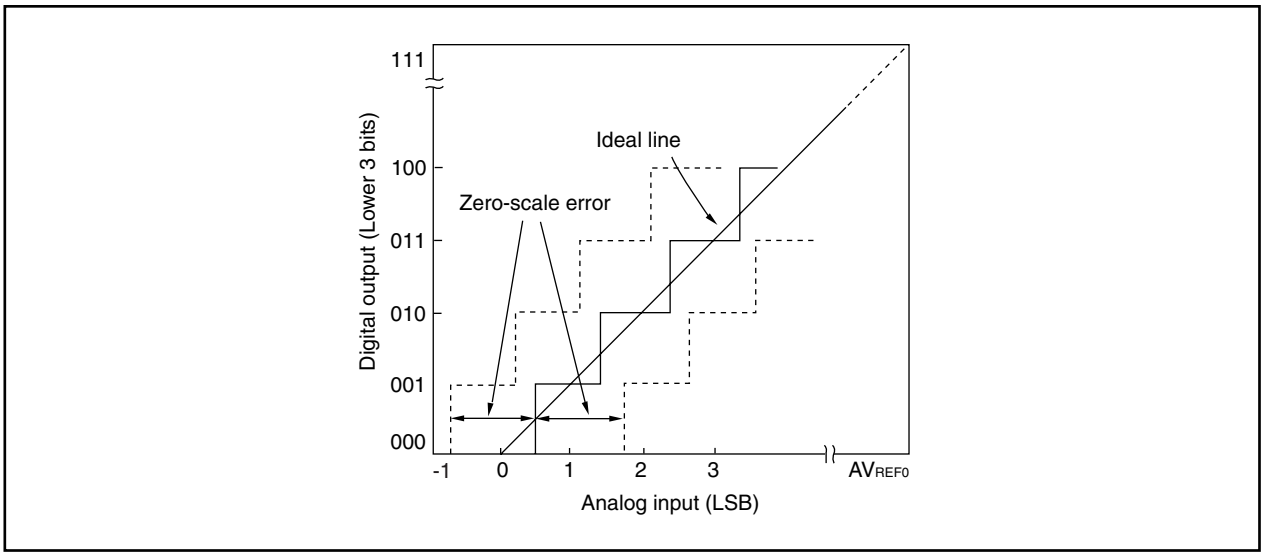
Figure 13-11. Quantization Error



(4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (1/2 LSB) when the digital output changes from 0……000 to 0……001.

Figure 13-12. Zero-Scale Error

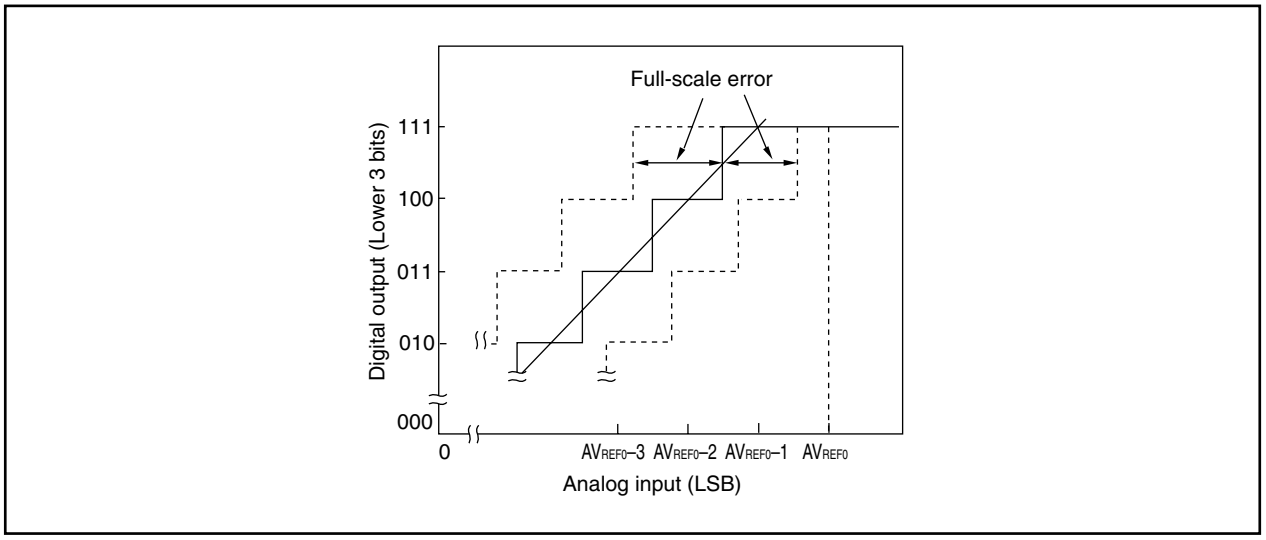


Downloaded from Arrow.com.

(5) Full-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (full scale – 3/2 LSB) when the digital output changes from 1……110 to 1……111.

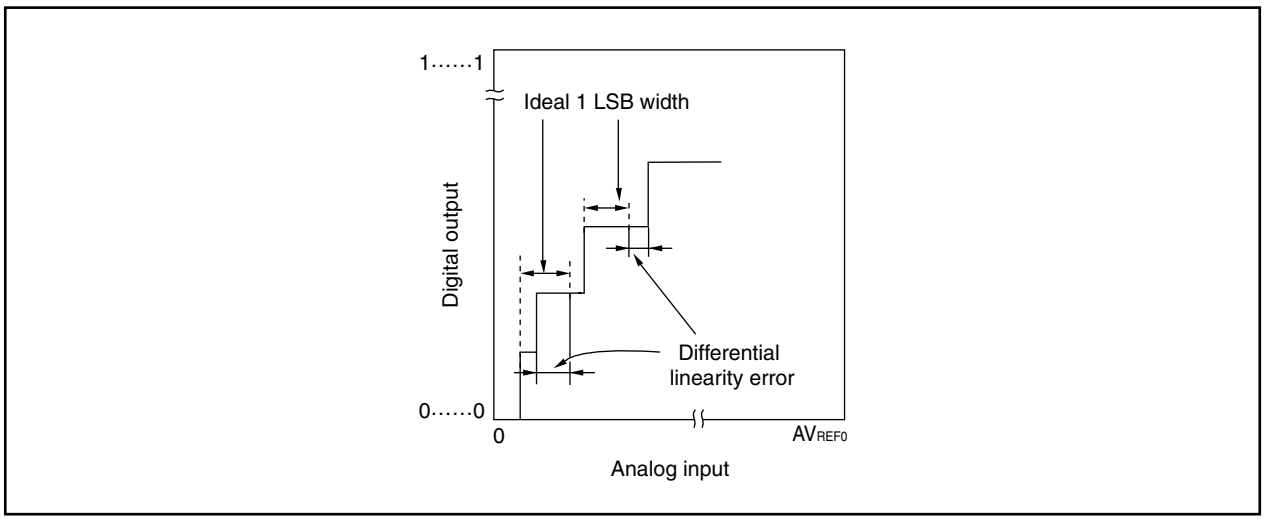
Figure 13-13. Full-Scale Error



(6) Differential linearity error

<R> While the ideal width of code output is 1 LSB, this indicates the difference between the actual measurement value and the ideal value. This indicates the basic characteristics of the A/D conversion when the voltage applied to the analog input pins of the same channel is consistently increased bit by bit from AV_{SS} to AV_{REF0}. When the input voltage is increased or decreased, or when two or more channels are used, refer to **13.6 (2) Overall error**.

Figure 13-14. Differential Linearity Error

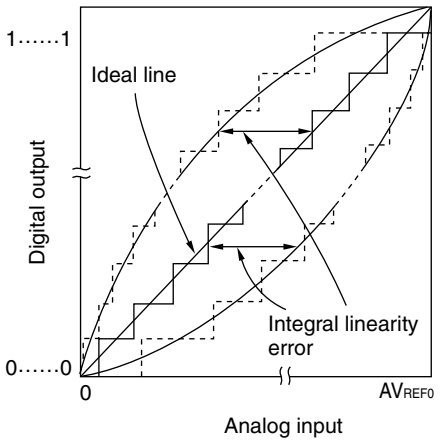


Downloaded from Arrow.com.

(7) Integral linearity error

This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

Figure 13-15. Integral Linearity Error



(8) Conversion time

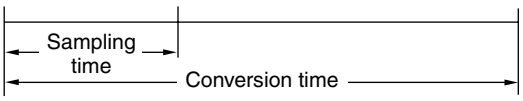
This expresses the time from when the analog input voltage was applied to the time when the digital output was obtained.

The sampling time is included in the conversion time in the characteristics table.

(9) Sampling time

This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.

Figure 13-16. Sampling Time



Downloaded from Arrow.com.

CHAPTER 14 ASYNCHRONOUS SERIAL INTERFACE (UART)

In the V850ES/KE1, two channels of asynchronous serial interface (UART) are provided.

14.1 Features

- Maximum transfer speed: 312.5 kbps
- Full-duplex communications
  On-chip RXBn register
  On-chip TXBn register
- Two-pin configuration[Note]
  TXDn: Transmit data output pin
  RXDn: Receive data input pin
- Reception error detection functions
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources: 3 types
  - Reception error interrupt request signal (INTSREn): Interrupt is generated according to the logical OR of the three types of reception errors
  - Reception completion interrupt request signal (INTSRn): Interrupt is generated when receive data is transferred from the receive shift register to the RXBn register after serial transfer is completed during a reception enabled state
  - Transmission completion interrupt request signal (INTSTn): Interrupt is generated when the serial transmission of transmit data (8 or 7 bits) from the transmit shift register is completed
- Character length: 7 or 8 bits
- Parity functions: Odd, even, 0, or none
- Transmission stop bits: 1 or 2 bits
- On-chip dedicated baud rate generator

Note The ASCK0 pin (external clock input) is available only for UART0.

Remark n = 0, 1

Downloaded from Arrow.com.

14.2 Configuration

Table 14-1. Configuration of UARTn

Item	Configuration
Registers	Receive buffer register n (RXBn) Transmit buffer register n (TXBn) Receive shift register Transmit shift register Asynchronous serial interface mode register n (ASIMn) Asynchronous serial interface status register n (ASISn) Asynchronous serial interface transmit status register n (ASIFn)
Other	Reception control parity check Addition of transmission control parity

Remark n = 0, 1

Figure 14-1 shows the configuration of UARTn.

(1) Asynchronous serial interface mode register n (ASIMn)

The ASIMn register is an 8-bit register for specifying the operation of UARTn.

(2) Asynchronous serial interface status register n (ASISn)

The ASISn register consists of a set of flags that indicate the error contents when a reception error occurs. The various reception error flags are set (1) when a reception error occurs and are cleared (0) when the ASISn register is read.

(3) Asynchronous serial interface transmit status register n (ASIFn)

The ASIFn register is an 8-bit register that indicates the status when a transmit operation is performed.

This register consists of a transmit buffer data flag, which indicates the hold status of the TXBn register data, and the transmit shift register data flag, which indicates whether transmission is in progress.

(4) Reception control parity check

The receive operation is controlled according to the contents set in the ASIMn register. A check for parity errors is also performed during a receive operation, and if an error is detected, a value corresponding to the error contents is set in the ASISn register.

(5) Receive shift register

This is a shift register that converts the serial data that was input to the RXDn pin to parallel data. One byte of data is received, and if a stop bit is detected, the receive data is transferred to the RXBn register.

This register cannot be directly manipulated.

(6) Receive buffer register n (RXBn)

The RXBn register is an 8-bit buffer register for holding receive data. When 7 characters are received, 0 is stored in the MSB.

During a reception enabled state, receive data is transferred from the receive shift register to the RXBn register, synchronized with the end of the shift-in processing of one frame.

Also, the reception completion interrupt request signal (INTSRn) is generated by the transfer of data to the RXBn register.

Downloaded from Arrow.com.

(7) Transmit shift register

This is a shift register that converts the parallel data that was transferred from the TXBn register to serial data. When one byte of data is transferred from the TXBn register, the shift register data is output from the TXDn pin.

The transmission completion interrupt request signal (INTSTn) is generated synchronized with the completion of transmission of one frame.

This register cannot be directly manipulated.

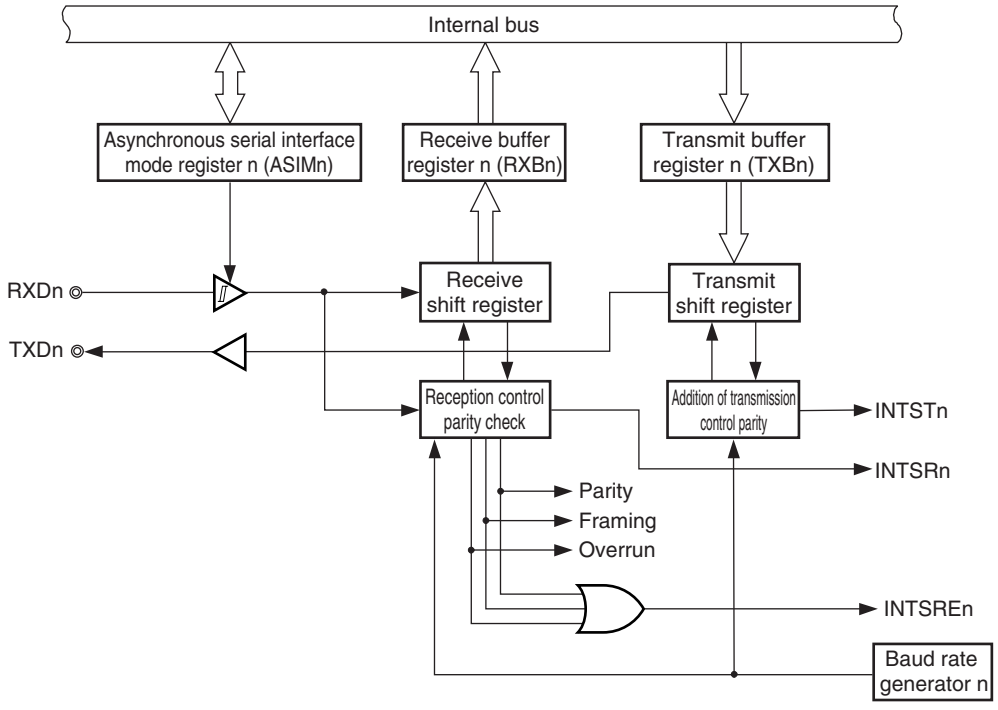
(8) Transmit buffer register n (TXBn)

The TXBn register is an 8-bit buffer for transmit data. A transmit operation is started by writing transmit data to the TXBn register.

(9) Addition of transmission control parity

A transmit operation is controlled by adding a start bit, parity bit, or stop bit to the data that is written to the TXBn register, according to the contents that were set in the ASIMn register.

Figure 14-1. Block Diagram of UARTn



Remark For the configuration of the baud rate generator, refer to **Figure 14-12**.

Downloaded from Arrow.com.

14.3 Registers

(1) Asynchronous serial interface mode register n (ASIMn)

The ASIMn register is an 8-bit register that controls the UARTn transfer operation.

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

Reset sets this register to 01H.

Cautions 1. When using UARTn, be sure to set the external pins related to UARTn functions to the control made before setting the CKSRn and BRGCn registers, and then set the UARTEn bit to 1. Then set the other bits.

2. Set the UARTEn and RXEn bits to 1 while a high level is input to the RXDn pin. If these bits are set to 1 while a low level is input to the RXDn pin, reception will be started.

(1/2)

After reset: 01H R/W Address: ASIM0 FFFFFA00H, ASIM1 FFFFFA10H

	<7>	<6>	<5>	4	3	2	1	0
ASIMn	UARTEn	TXEn	RXEn	PSn1	PSn0	CLn	SLn	ISRMn

(n = 0, 1)

UARTEn	Control of operating clock
0	Stop clock supply to UARTn.
1	Supply clock to UARTn.

- If the UARTEn bit is cleared to 0, UARTn is asynchronously reset[Note].
- If the UARTEn bit = 0, UARTn is reset. To operate UARTn, first set the UARTEn bit to 1.
- If the UARTEn bit is cleared from 1 to 0, all the registers of UARTn are initialized. To set the UARTEn bit to 1 again, be sure to re-set the registers of UARTn.

The output of the TXDn pin goes high when transmission is disabled, regardless of the setting of the UARTEn bit.

TXEn	Transmission enable/disable
0	Disable transmission
1	Enable transmission

- Set the TXEn bit to 1 after setting the UARTEn bit to 1 at startup. Clear the UARTEn bit to 0 after clearing the TXEn bit to 0 to stop.
- To initialize the transmission unit, clear (0) the TXEn bit, and after letting 2 Clock cycles (base clock) elapse, set (1) the TXEn bit again. If the TXEn bit is not set again, initialization may not be successful. (For details about the base clock, refer to **14.6.1 (1) Base clock**.)

Note The ASISn, ASIFn, and RXBn registers are reset.

Downloaded from Arrow.com.

(2/2)

RXEn	Reception enable/disable
0	Disable reception[Note]
1	Enable reception

- Set the RXEn bit to 1 after setting the UARTEn bit to 1 at startup. Clear the UARTEn bit to 0 after clearing the RXEn bit to 0 to stop.
- To initialize the reception unit status, clear (0) the RXEn bit, and after letting 2 Clock cycles (base clock) elapse, set (1) the RXEn bit again. If the RXEn bit is not set again, initialization may not be successful. (For details about the base clock, refer to **14.6.1 (1) Base clock**.)

PSn1	PSn0	Transmit operation	Receive operation
0	0	Don't output parity bit	Receive with no parity
0	1	Output 0 parity	Receive as 0 parity
1	0	Output odd parity	Judge as odd parity
1	1	Output even parity	Judge as even parity

- To overwrite the PSn1 and PSn0 bits, first clear (0) the TXEn and RXEn bits.
- If "0 parity" is selected for reception, no parity judgment is performed. Therefore, no error interrupt is generated because the ASISn.PEn bit is not set.

CLn	Specification of character length of 1 frame of transmit/receive data
0	7 bits
1	8 bits

- To overwrite the CLn bit, first clear (0) the TXEn and RXEn bits.

SLn	Specification of stop bit length of transmit data
0	1 bit
1	2 bits

- To overwrite the SLn bit, first clear (0) the TXEn bit.
- Since reception is always done with a stop bit length of 1, the SLn bit setting does not affect receive operations.

ISRMn	Enable/disable of generation of reception completion interrupt request signals when an error occurs
0	Generate a reception error interrupt request signal (INTSREn) as an interrupt when an error occurs. In this case, no reception completion interrupt request signal (INTSRn) is generated.
1	Generate a reception completion interrupt request signal (INTSRn) as an interrupt when an error occurs. In this case, no reception error interrupt request signal (INTSREn) is generated.

- To overwrite the ISRMn bit, first clear (0) the RXEn bit.

Note When reception is disabled, the receive shift register does not detect a start bit. No shift-in processing or transfer processing to the RXBn register is performed, and the contents of the RXBn register are retained.

When reception is enabled, the receive shift operation starts, synchronized with the detection of the start bit, and when the reception of one frame is completed, the contents of the receive shift register are transferred to the RXBn register. A reception completion interrupt request signal (INTSRn) is also generated in synchronization with the transfer to the RXBn register.

Downloaded from Arrow.com.

(2) Asynchronous serial interface status register n (ASISn)

The ASISn register, which consists of 3 error flag bits (PEn, FEn and OVEn), indicates the error status when UARTn reception is complete.

The ASISn register is cleared to 00H by a read operation. When a reception error occurs, the RXBn register should be read and the error flag should be cleared after the ASISn register is read.

This register is read-only in 8-bit units.

Reset sets this register to 00H.

Cautions 1. When the ASIMn.UARTEn bit or ASIMn.RXEn bit is cleared to 0, or when the ASISn register is read, the PEn, FEn, and OVEn bits are cleared (0).

2. Operation using a bit manipulation instruction is prohibited.

3. When the main clock is stopped and the CPU is operating on the subclock, do not access the ASISn register.
For details, refer to 3.4.8 (1) (b).

After reset: 00H R Address: ASIS0 FFFFFA03H, ASIS1 FFFFFA13H

	7	6	5	4	3	2	1	0
ASISn	0	0	0	0	0	PEn	FEn	OVEn

(n = 0, 1)

PEn	Status flag indicating a parity error
0	When the UARTEn or RXEn bit is cleared to 0, or after the ASISn register has been read
1	When reception was completed, the receive data parity did not match the parity bit
• The operation of the PEn bit differs according to the settings of the ASIMn.PSn1 and ASIMn.PSn0 bits.	

FEn	Status flag indicating framing error
0	When the UARTEn or RXEn bit is cleared to 0, or after the ASISn register has been read
1	When reception was completed, no stop bit was detected
• For receive data stop bits, only the first bit is checked regardless of the stop bit length.	

OVEn	Status flag indicating an overrun error
0	When the UARTEn or RXEn bit is cleared to 0, or after the ASISn register has been read.
1	UARTn completed the next receive operation before reading receive data of the RXBn register.
• When an overrun error occurs, the next receive data value is not written to the RXBn register and the data is discarded.	

Downloaded from Arrow.com.

(3) Asynchronous serial interface transmit status register n (ASIFn)

The ASIFn register, which consists of 2 status flag bits, indicates the status during transmission.

By writing the next data to the TXBn register after data is transferred from the TXBn register to the transmit shift register, transmit operations can be performed continuously without suspension even during an interrupt interval. When transmission is performed continuously, data should be written after referencing the TXBFn bit to prevent writing to the TXBn register by mistake.

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

Reset sets this register to 00H.

After reset: 00H R Address: ASIF0 FFFFFA05H, ASIF1 FFFFFA15H

	7	6	5	4	3	2	<1>	<0>
ASIFn	0	0	0	0	0	0	TXBFn	TXSFn

(n = 0, 1)

TXBFn	Transmission buffer data flag
0	Data to be transferred next to TXBn register does not exist (When the ASIMn.UARTEn or ASIMn.TXEn bit is cleared to 0, or when data has been transferred to the transmission shift register)
1	Data to be transferred next exists in TXBn register (Data exists in TXBn register when the TXBn register has been written to)
• When transmission is performed continuously, data should be written to the TXBn register after confirming that this flag is 0. If writing to TXBn register is performed when this flag is 1, transmit data cannot be guaranteed.	

TXSFn	Transmit shift register data flag (indicates the transmission status of UARTn)
0	Initial status or a waiting transmission (When the UARTEn or TXEn bit is cleared to 0, or when following transmission completion, the next data transfer from the TXBn register is not performed)
1	Transmission in progress (When data has been transferred from the TXBn register)
• When the transmission unit is initialized, initialization should be executed after confirming that this flag is 0 following the occurrence of a transmission completion interrupt request signal (INTSTn). If initialization is performed when this flag is 1, transmit data cannot be guaranteed.	

Downloaded from Arrow.com.

(4) Receive buffer register n (RXBn)

The RXBn register is an 8-bit buffer register for storing parallel data that had been converted by the receive shift register.

When reception is enabled (ASIMn.RXEn bit = 1), receive data is transferred from the receive shift register to the RXBn register, synchronized with the completion of the shift-in processing of one frame. Also, a reception completion interrupt request signal (INTSRn) is generated by the transfer to the RXBn register. For information about the timing for generating this interrupt request, refer to **14.5.4 Receive operation**.

If reception is disabled (ASIMn.RXEn bit = 0), the contents of the RXBn register are retained, and no processing is performed for transferring data to the RXBn register even when the shift-in processing of one frame is completed. Also, the INTSRn signal is not generated.

When 7 bits is specified for the data length, bits 6 to 0 of the RXBn register are transferred for the receive data and the MSB (bit 7) is always 0. However, if an overrun error (ASISn.OVEn bit = 1) occurs, the receive data at that time is not transferred to the RXBn register.

The RXBn register becomes FFH when a reset is input or ASIMn.UARTEn bit = 0.

This register is read-only in 8-bit units.

After reset: FFH R Address: RXB0 FFFFFA02H, RXB1 FFFFFA12H

	7	6	5	4	3	2	1	0
RXBn	RXBn7	RXBn6	RXBn5	RXBn4	RXBn3	RXBn2	RXBn1	RXBn0

(n = 0, 1)

Downloaded from Arrow.com.

(5) Transmit buffer register n (TXBn)

The TXBn register is an 8-bit buffer register for setting transmit data.

When transmission is enabled (ASIMn.TXEn bit = 1), the transmit operation is started by writing data to TXBn register.

When transmission is disabled (TXEn bit = 0), even if data is written to TXBn register, the value is ignored.

The TXBn register data is transferred to the transmit shift register, and a transmission completion interrupt request signal (INTSTn) is generated, synchronized with the completion of the transmission of one frame from the transmit shift register. For information about the timing for generating this interrupt request, refer to **14.5.2 Transmit operation**.

When ASIFn.TXBFn bit = 1, writing must not be performed to TXBn register.

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

Reset sets this register to FFH.

After reset: FFH R/W Address: TXB0 FFFFFA04H, TXB1 FFFFFA14H

	7	6	5	4	3	2	1	0
TXBn	TXBn7	TXBn6	TXBn5	TXBn4	TXBn3	TXBn2	TXBn1	TXBn0

(n = 0, 1)

Downloaded from Arrow.com.

14.4 Interrupt Requests

The following three types of interrupt request signals are generated from UARTn.

- Reception error interrupt request signal (INTSREn)
- Reception completion interrupt request signal (INTSRn)
- Transmission completion interrupt request signal (INTSTn)

The default priorities among these three types of interrupt request signals are, from high to low, reception error interrupt, reception completion interrupt, and transmission completion interrupt.

Table 14-2. Generated Interrupt Request Signals and Default Priorities

Interrupt Request Signal	Priority
Reception error interrupt request signal (INTSREn)	1
Reception completion interrupt request signal (INTSRn)	2
Transmission completion interrupt request signal (INTSTn)	3

(1) Reception error interrupt request signal (INTSREn)

When reception is enabled, the INTSREn signal is generated according to the logical OR of the three types of reception errors explained for the ASISn register. Whether the INTSREn signal or the INTSRn signal is generated when an error occurs can be specified according to the ASIMn.ISRMn bit.

When reception is disabled, the INTSREn signal is not generated.

(2) Reception completion interrupt request signal (INTSRn)

When reception is enabled, the INTSRn signal is generated when data is shifted in to the receive shift register and transferred to the RXBn register.

The INTSRn signal can be generated in place of the INTSREn signal according to the ASIMn.ISRMn bit even when a reception error has occurred.

When reception is disabled, the INTSRn signal is not generated.

(3) Transmission completion interrupt request signal (INTSTn)

The INTSTn signal is generated when one frame of transmit data containing 7-bit or 8-bit characters is shifted out from the transmit shift register.

Downloaded from Arrow.com.

14.5 Operation

14.5.1 Data format

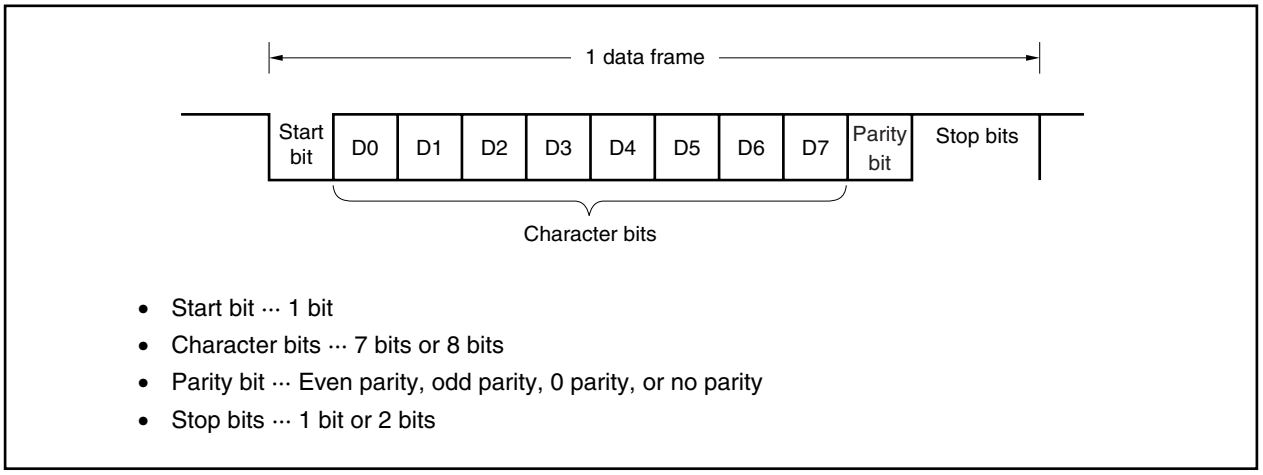
Full-duplex serial data transmission and reception can be performed.

The transmit/receive data format consists of one data frame containing a start bit, character bits, a parity bit, and stop bits as shown in Figure 14-2.

The character bit length within one data frame, the type of parity, and the stop bit length are specified according to the ASIMn register.

Also, data is transferred LSB first.

Figure 14-2. Format of UARTn Transmit/Receive Data



- Start bit ··· 1 bit
- Character bits ··· 7 bits or 8 bits
- Parity bit ··· Even parity, odd parity, 0 parity, or no parity
- Stop bits ··· 1 bit or 2 bits

Downloaded from Arrow.com.

14.5.2 Transmit operation

When the ASIMn.UARTEn bit is set to 1, a high level is output from the TXDn pin.

Then, when the ASIMn.TXEn bit is set to 1, transmission is enabled, and the transmit operation is started by writing transmit data to the TXBn register.

(1) Transmission enabled state

This state is set by the TXEn bit.

- TXEn bit = 1: Transmission enabled state
- TXEn bit = 0: Transmission disabled state

Since UARTn does not have a CTS (transmission enabled signal) input pin, a port should be used to confirm whether the destination is in a reception enabled state.

(2) Starting a transmit operation

In the transmission enabled state, a transmit operation is started by writing transmit data to the TXBn register. When a transmit operation is started, the data in the TXBn register is transferred to the transmit shift register. Then, the transmit shift register outputs data to the TXDn pin (the transmit data is transferred sequentially starting with the start bit). The start bit, parity bit, and stop bits are added automatically.

(3) Transmission interrupt

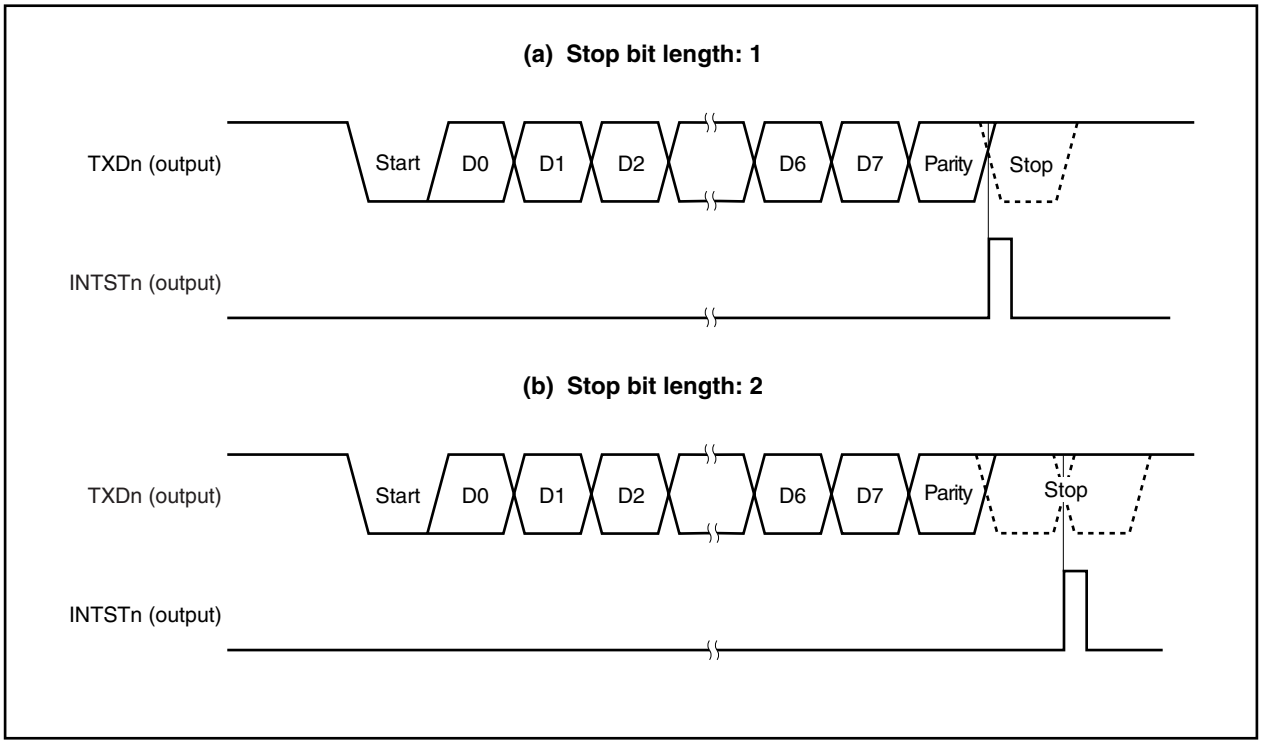
When the transmit shift register becomes empty, a transmission completion interrupt request signal (INTSTn) is generated. The timing for generating the INTSTn signal differs according to the specification of the stop bit length. The INTSTn signal is generated at the same time that the last stop bit is output.

If the data to be transmitted next has not been written to the TXBn register, the transmit operation is suspended.

Caution Normally, when the transmit shift register becomes empty, the INTSTn signal is generated. However, the INTSTn signal is not generated if the transmit shift register becomes empty due to reset.

Downloaded from Arrow.com.

Figure 14-3. UARTn Transmission Completion Interrupt Timing



Downloaded from Arrow.com.

14.5.3 Continuous transmission operation

UARTn can write the next transmit data to the TXBn register at the timing that the transmit shift register starts the shift operation. This enables an efficient transmission rate to be realized by continuously transmitting data even during the transmission completion interrupt service after the transmission of one data frame. In addition, reading the ASIFn.TXSFn bit after the occurrence of a transmission completion interrupt request signal (INTSTn) enables the TXBn register to be efficiently written twice (2 bytes) without waiting for the transmission of 1 data frame.

When continuous transmission is performed, data should be written after referencing the ASIFn register to confirm the transmission status and whether or not data can be written to the TXBn register.

Caution The values of the ASIF.TXBFn and ASIF.TXSFn bits change 10 → 11 → 01 in continuous transmission.
Therefore, do not confirm the status based on the combination of the TXBFn and TXSFn bits.
Read only the TXBFn bit during continuous transmission.

TXBFn	Whether or Not Writing to TXBn Register Is Enabled
0	Writing is enabled
1	Writing is not enabled

Caution When transmission is performed continuously, write the first transmit data (first byte) to the TXBn register and confirm that the TXBFn bit is 0, and then write the next transmit data (second byte) to TXBn register. If writing to the TXBn register is performed when the TXBFn bit is 1, transmit data cannot be guaranteed.

The communication status can be confirmed by referring to the TXSFn bit.

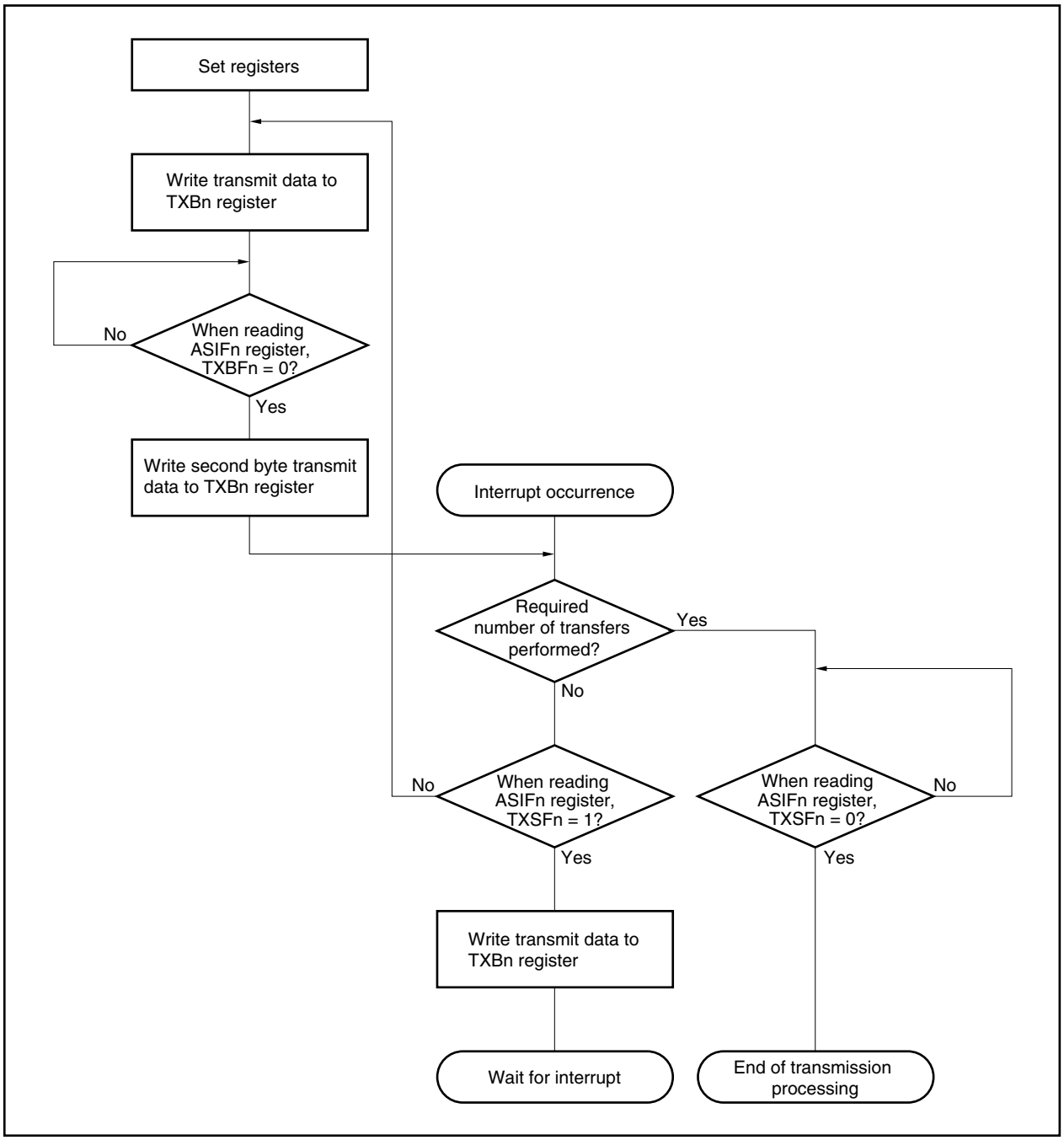
TXSFn	Transmission Status
0	Transmission is completed.
1	Under transmission.

Cautions 1. When initializing the transmission unit when continuous transmission is completed, confirm that the TXSFn bit is 0 after the occurrence of the transmission completion interrupt, and then execute initialization. If initialization is performed when the TXSFn bit is 1, transmit data cannot be guaranteed.

2. While transmission is being performed continuously, an overrun error may occur if the next transmission is completed before the INTSTn interrupt servicing following the transmission of 1 data frame is executed. An overrun error can be detected by embedding a program that can count the number of transmit data and referencing TXSFn bit.

Downloaded from Arrow.com.

Figure 14-4. Continuous Transmission Processing Flow

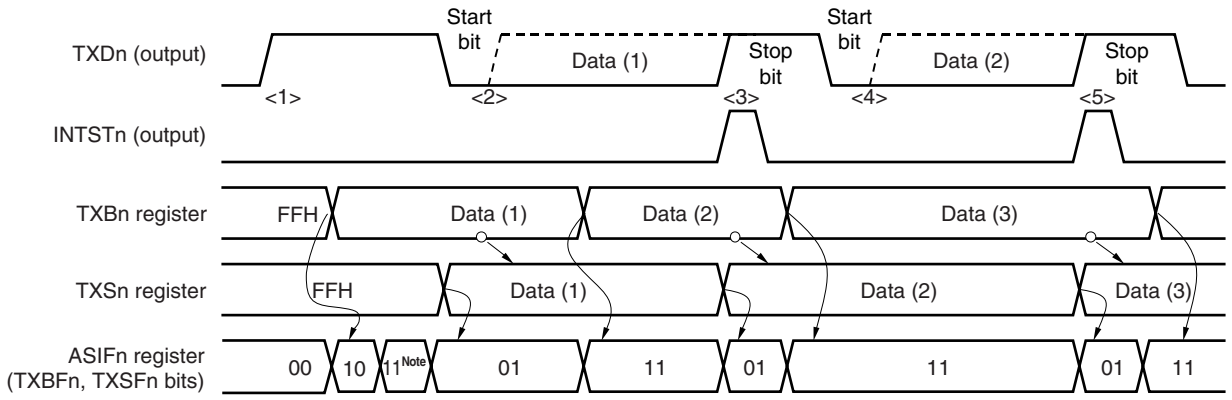


Downloaded from Arrow.com.

(1) Starting procedure

The procedure to start continuous transmission is shown below.

Figure 14-5. Continuous Transmission Starting Procedure



Note Refer to **14.7 Cautions (2)**.

Transmission Starting Procedure	Internal Operation	ASIFn Register	
		TXBFn	TXSFn
• Set transmission mode	<1> Start transmission unit	0	0
• Write data (1) →		1	0
	<2> Generate start bit	1	1Note
		0	1
	Start data (1) transmission →	0	1
• Read ASIFn register (confirm that TXBFn bit = 0) ←		0	1
• Write data (2) →		1	1
	<<Transmission in progress>>		
	<3> INTSTn interrupt occurs →	0	1
• Read ASIFn register (confirm that TXBFn bit = 0) ←		0	1
• Write data (3) →		1	1
	<4> Generate start bit		
	Start data (2) transmission		
	<<Transmission in progress>>		
	<5> INTSTn interrupt occurs →	0	1
• Read ASIFn register (confirm that TXBFn bit = 0) ←		0	1
• Write data (4) →		1	1

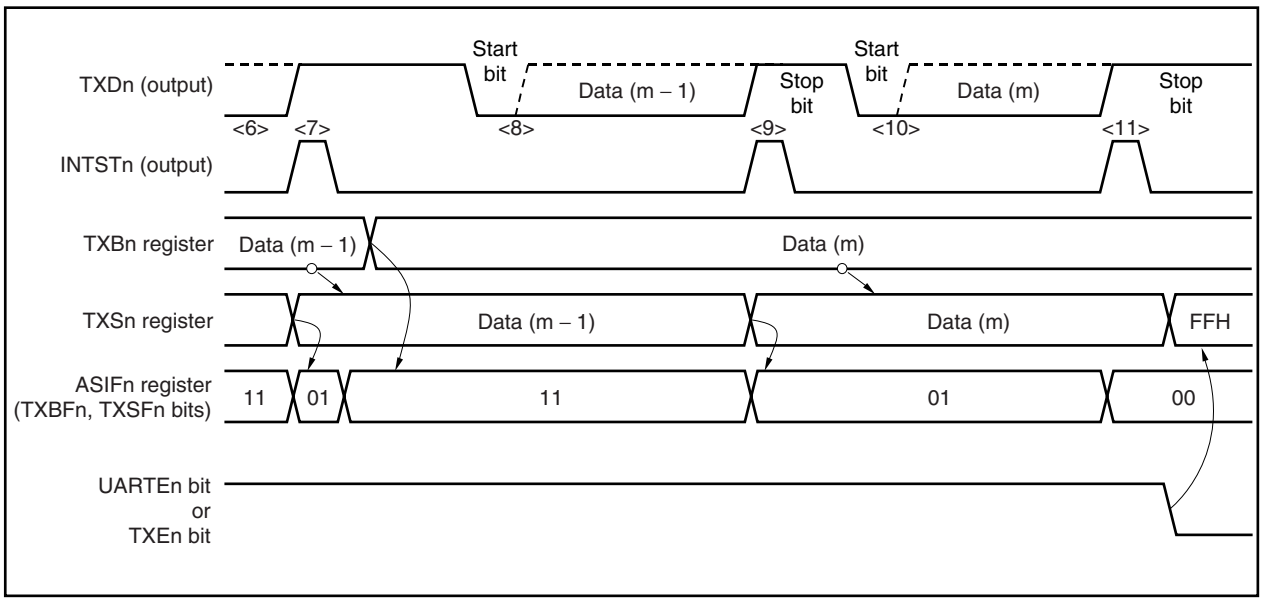
Note Refer to **14.7 Cautions (2)**.

Downloaded from Arrow.com.

(2) Ending procedure

The procedure for ending continuous transmission is shown below.

Figure 14-6. Continuous Transmission End Procedure



Transmission End Procedure	Internal Operation	ASIFn Register	
		TXBFn	TXSFn
	<6> Transmission of data (m – 2) is in progress	1	1
	<7> INTSTn interrupt occurs →	0	1
• Read ASIFn register (confirm that TXBFn bit = 0) ←		0	1
• Write data (m) →		1	1
	<8> Generate start bit Start data (m – 1) transmission <<Transmission in progress>>		
	<9> INTSTn interrupt occurs →	0	1
• Read ASIFn register (confirm that TXSFn bit = 1) ←		0	1
There is no write data			
	<10> Generate start bit Start data (m) transmission <<Transmission in progress>>		
	<11> Generate INTSTn interrupt →	0	0
• Read ASIFn register (confirm that TXSFn bit = 0) ←		0	0
• Clear (0) the UARTEn bit or TXEn bit	Initialize internal circuits		

Downloaded from Arrow.com.

14.5.4 Receive operation

The awaiting reception state is set by setting the ASIMn.UARTEn bit to 1 and then setting the ASIMn.RXEn bit to 1. To start the receive operation, start sampling at the falling edge when the falling of the RXDn pin is detected. If the RXDn pin is low level at a start bit sampling point, the start bit is recognized. When the receive operation begins, serial data is stored sequentially in the receive shift register according to the baud rate that was set. A reception completion interrupt request signal (INTSRn) is generated each time the reception of one frame of data is completed. Normally, the receive data is transferred from the RXBn register to memory by this interrupt servicing.

(1) Reception enabled state

The receive operation is set to the reception enabled state by setting the RXEn bit to 1.

- RXEn bit = 1: Reception enabled state
- RXEn bit = 0: Reception disabled state

In receive disabled state, the reception hardware stands by in the initial state. At this time, the contents of the RXBn register are retained, and no reception completion interrupt or reception error interrupt is generated.

(2) Starting a receive operation

A receive operation is started by the detection of a start bit.

The RXDn pin is sampled using the serial clock from baud rate generator n (BRGn).

(3) Reception completion interrupt

When the RXEn bit = 1 and the reception of one frame of data is completed (the stop bit is detected), the INTSRn signal is generated and the receive data within the receive shift register is transferred to the RXBn register at the same time.

Also, if an overrun error (ASISn.OVEn bit = 1) occurs, the receive data at that time is not transferred to the RXBn register, and either the INTSRn signal or a reception error interrupt request signal (INTSREn) is generated according to the ASIMn.ISRMn bit setting.

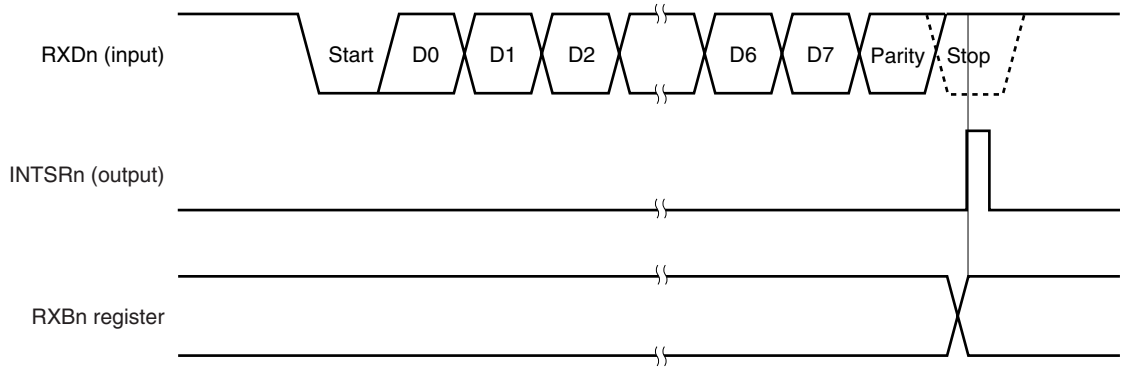
Even if a parity error (ASISn.PEn bit = 1) or framing error (ASISn.FEn bit = 1) occurs during a reception operation, the receive operation continues until stop bit is received, and after reception is completed, either the INTSRn signal or the INTSREn signal is generated according to the ISRMn bit setting (the receive data within the receive shift register is transferred to the RXBn register).

If the RXEn bit is cleared (0) during a receive operation, the receive operation is immediately stopped. The contents of the RXBn register and the ASISn register at this time do not change, and the INTSRn signal or the INTSREn signal is not generated.

The INTSRn signal or the INTSREn signal is not generated when the RXEn bit = 0 (reception is disabled).

Downloaded from Arrow.com.

Figure 14-7. UARTn Reception Completion Interrupt Timing



Cautions 1. Be sure to read the RXBn register even when a reception error occurs. If the RXBn register is not read, an overrun error will occur at the next data reception and the reception error status will continue infinitely.

2. Reception is always performed assuming a stop bit length of 1.
A second stop bit is ignored.

14.5.5 Reception error

The three types of errors that can occur during a receive operation are a parity error, framing error, and overrun error. As a result of data reception, the various flags of the ASISn register are set (1), and a reception error interrupt request signal (INTSREn) or a reception completion interrupt request signal (INTSRn) is generated at the same time. The ASIMn.ISRMn bit specifies whether the INTSREn signal or the INTSRn signal is generated.

The type of error that occurred during reception can be detected by reading the contents of the ASISn register during the INTSREn or INTSRn interrupt servicing.

The contents of the ASISn register are cleared (0) by reading the ASISn register.

Table 14-3. Reception Error Causes

Error Flag	Reception Error	Cause
PEn	Parity error	The parity specification during transmission did not match the parity of the reception data
FEn	Framing error	No stop bit was detected
OVEn	Overrun error	The reception of the next data was completed before data was read from the RXBn register

Downloaded from Arrow.com.

(1) Separation of reception error interrupt request signal

A reception error interrupt request signal can be separated from the INTSRn signal and generated as the INTSREn signal by clearing the ISRMn bit to 0.

Figure 14-8. When Reception Error Interrupt Request Signal Is Separated from INTSRn Signal (ISRMn Bit = 0)

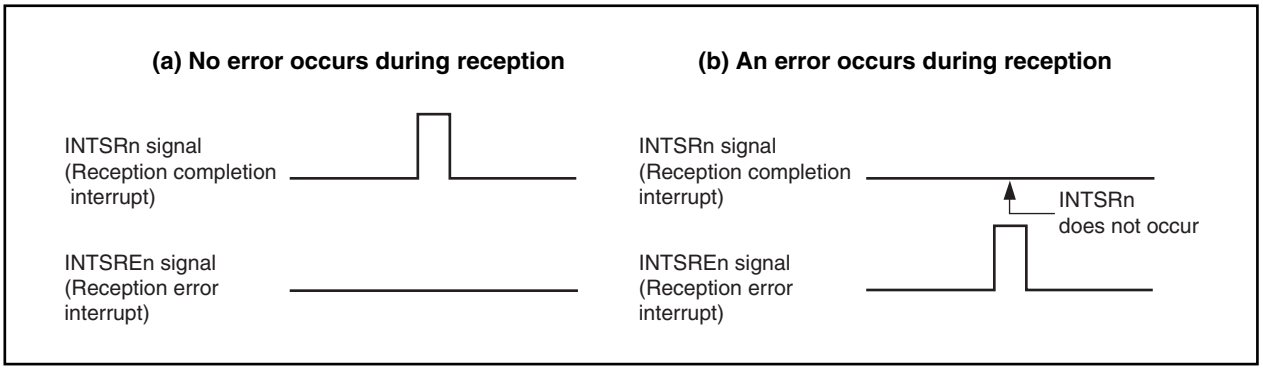
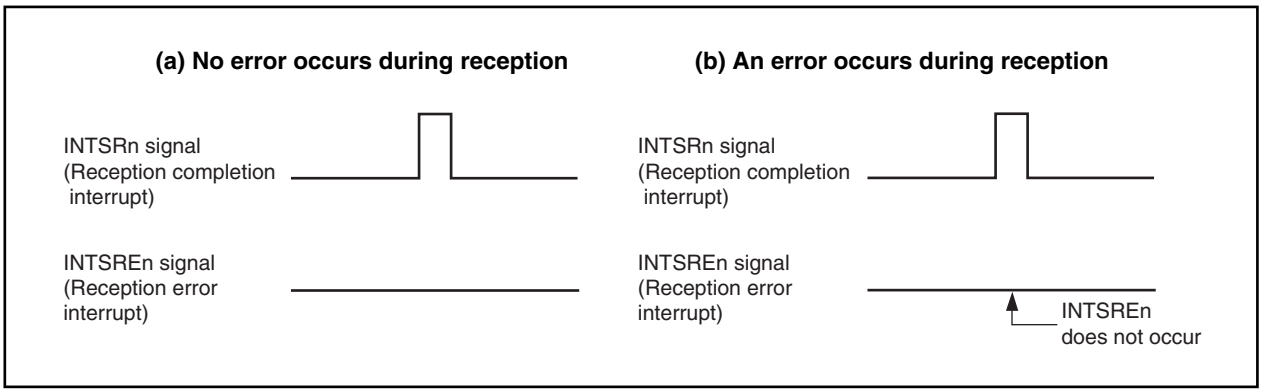


Figure 14-9. When Reception Error Interrupt Request Signal Is Included in INTSRn Signal (ISRMn Bit = 1)



Downloaded from Arrow.com.

14.5.6 Parity types and corresponding operation

A parity bit is used to detect a bit error in communication data. Normally, the same type of parity bit is used on the transmission and reception sides.

(1) Even parity

(i) During transmission

The parity bit is controlled so that the number of bits with the value “1” within the transmit data including the parity bit is even. The parity bit value is as follows.

- If the number of bits with the value “1” within the transmit data is odd: 1
- If the number of bits with the value “1” within the transmit data is even: 0

(ii) During reception

The number of bits with the value “1” within the receive data including the parity bit is counted, and a parity error is generated if this number is odd.

(2) Odd parity

(i) During transmission

In contrast to even parity, the parity bit is controlled so that the number of bits with the value “1” within the transmit data including the parity bit is odd. The parity bit value is as follows.

- If the number of bits with the value “1” within the transmit data is odd: 0
- If the number of bits with the value “1” within the transmit data is even: 1

(ii) During reception

The number of bits with the value “1” within the receive data including the parity bit is counted, and a parity error is generated if this number is even.

(3) 0 parity

During transmission the parity bit is set to “0” regardless of the transmit data.

During reception, no parity bit check is performed. Therefore, no parity error is generated regardless of whether the parity bit is “0” or “1”.

(4) No parity

No parity bit is added to the transmit data.

During reception, the receive operation is performed as if there were no parity bit. Since there is no parity bit, no parity error is generated.

Downloaded from Arrow.com.

14.5.7 Receive data noise filter

The RXDn signal is sampled at the rising edge of the prescaler output base clock (f_{UCLK}). If the same sampling value is obtained twice, the match detector output changes, and this output is sampled as input data. Therefore, data not exceeding one clock width is judged to be noise and is not delivered to the internal circuit (refer to **Figure 14-11**). Refer to **14.6.1 (1) Base clock** regarding the base clock.

Also, since the circuit is configured as shown in Figure 14-10, internal processing during a receive operation is delayed by up to 2 clocks according to the external signal status.

Figure 14-10. Noise Filter Circuit

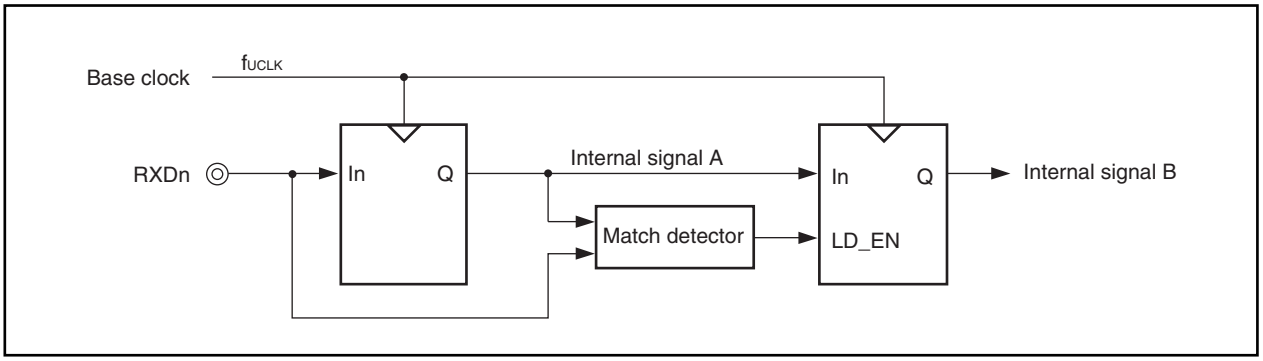
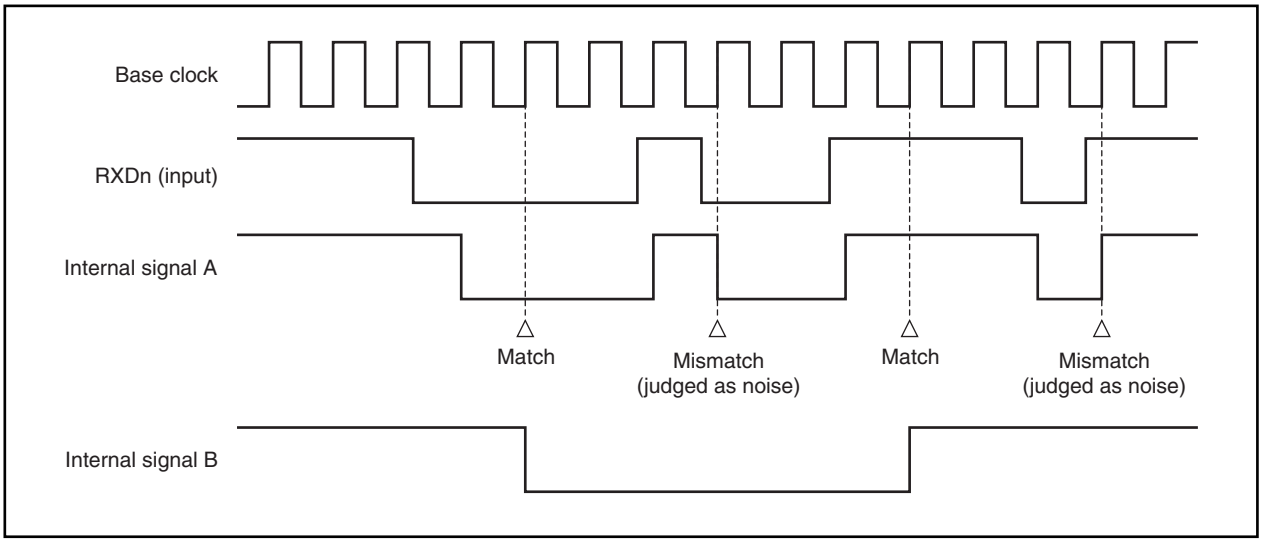


Figure 14-11. Timing of RXDn Signal Judged as Noise



Downloaded from Arrow.com.

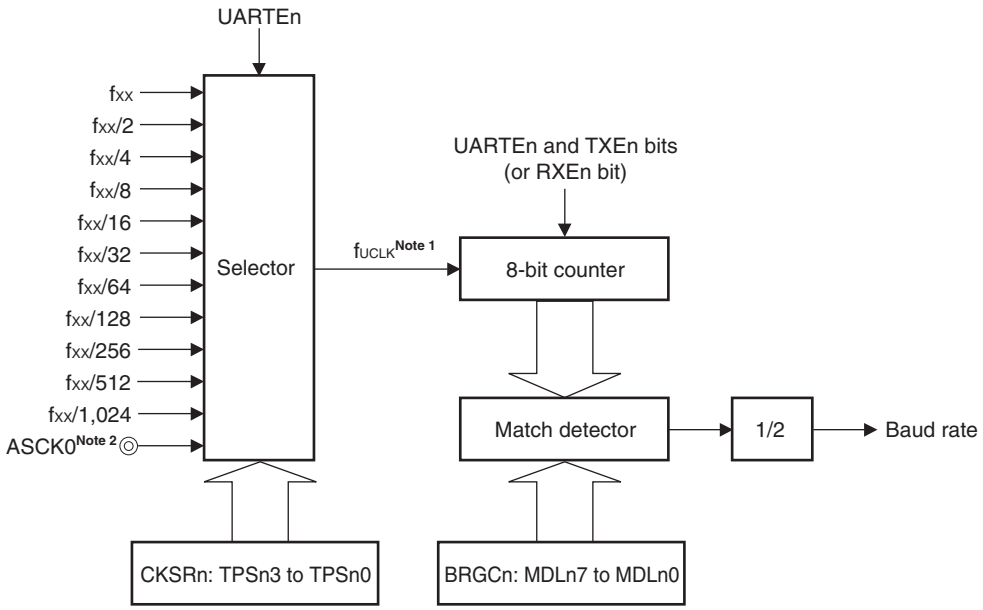
14.6 Dedicated Baud Rate Generator n (BRGn)

A dedicated baud rate generator, which consists of a source clock selector and an 8-bit programmable counter, generates serial clocks during transmission/reception by UARTn. The dedicated baud rate generator output can be selected as the serial clock for each channel.

Separate 8-bit counters exist for transmission and for reception.

14.6.1 Baud rate generator n (BRGn) configuration

Figure 14-12. Configuration of Baud Rate Generator n (BRGn)



Notes 1. Set f_{UCLK} so as to satisfy the following conditions.

- V_{DD} = 4.5 to 5.5 V: $f_{UCLK} \leq$ 12 MHz
- V_{DD} = 2.7 to 4.5 V: $f_{UCLK} \leq$ 6 MHz

2. ASCK0 pin input can be used only by UART0.

Remark f_{XX}: Main clock frequency
f_{UCLK}: Base clock

(1) Base clock

When the ASIMn.UARTEn bit = 1, the clock selected according to the CKSRn.TPSn3 to CKSRn.TPSn0 bits is supplied to the transmission/reception unit. This clock is called the base clock (f_{UCLK}). When the UARTEn bit = 0, f_{UCLK} is fixed to low level.

Downloaded from Arrow.com.

14.6.2 Serial clock generation

A serial clock can be generated according to the settings of the CKSRn and BRGCn registers.

The base clock to the 8-bit counter is selected by the CKSRn.TPSn3 to CKSRn.TPSn0 bits.

The 8-bit counter divisor value can be set by the BRGCn.MDLn7 to BRGCn.MDLn0 bits.

(1) Clock select register n (CKSRn)

The CKSRn register is an 8-bit register for selecting the basic block using the TPSn3 to TPSn0 bits. The clock selected by the TPSn3 to TPSn0 bits becomes the base clock (f_{UCLK}) of the transmission/reception module.

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

Reset sets this register to 00H.

Caution Clear the ASIMn.UARTEn bit to 0 before rewriting the TPSn3 to TPSn0 bits.

After reset: 00H R/W Address: CKSR0 FFFFFA06H, CKSR1 FFFFFA16H

	7	6	5	4	3	2	1	0
CKSRn	0	0	0	0	TPSn3	TPSn2	TPSn1	TPSn0

(n = 0, 1)

TPSn3	TPSn2	TPSn1	TPSn0	Base clock (f_{UCLK})[Note 1]
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	$f_{XX}/1,024$
1	0	1	1	External clock[Note 2] (ASCK0 pin)
Other than above				Setting prohibited

Notes 1. Set f_{UCLK} so as to satisfy the following conditions.

- V_{DD} = 4.5 to 5.5 V: $f_{UCLK} \leq$ 12 MHz
- V_{DD} = 2.7 to 4.5 V: $f_{UCLK} \leq$ 6 MHz

2. ASCK0 pin input clock can be used only by UART0.
Setting of UART1 is prohibited.

Remark f_{XX}: Main clock frequency

Downloaded from Arrow.com.

(2) Baud rate generator control register n (BRGCn)

The BRGCn register is an 8-bit register that controls the baud rate (serial transfer speed) of UARTn.

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

Reset sets this register to FFH.

Caution If the MDLn7 to MDLn0 bits are to be overwritten, the ASIMn.TXEn and ASIMn.RXEn bits should be cleared to 0 first.

After reset: FFH R/W Address: BRGC0 FFFFFA07H, BRGC1 FFFFFA17H

	7	6	5	4	3	2	1	0
BRGCn	MDLn7	MDLn6	MDLn5	MDLn4	MDLn3	MDLn2	MDLn1	MDLn0

(n = 0, 1)

MDLn7	MDLn6	MDLn5	MDLn4	MDLn3	MDLn2	MDLn1	MDLn0	Set value (k)	Serial clock
0	0	0	0	0	×	×	×	–	Setting prohibited
0	0	0	0	1	0	0	0	8	f_{UCLK}/8
0	0	0	0	1	0	0	1	9	f_{UCLK}/9
0	0	0	0	1	0	1	0	10	f_{UCLK}/10
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	0	1	0	250	f_{UCLK}/250
1	1	1	1	1	0	1	1	251	f_{UCLK}/251
1	1	1	1	1	1	0	0	252	f_{UCLK}/252
1	1	1	1	1	1	0	1	253	f_{UCLK}/253
1	1	1	1	1	1	1	0	254	f_{UCLK}/254
1	1	1	1	1	1	1	1	255	f_{UCLK}/255

Remarks 1. f_{UCLK}: Frequency [Hz] of base clock selected by CKSR0.TPSn3 to CKSR0.TPSn0 bits

2. k: Value set by MDLn7 to MDLn0 bits (k = 8, 9, 10, ..., 255)

3. The baud rate is the output clock for the 8-bit counter divided by 2.

4. ×: don't care

Downloaded from Arrow.com.

(3) Baud rate

The baud rate is the value obtained by the following formula.

$$\text{Baud rate [bps]} = \frac{f_{UCLK}}{2 \times k}$$

f_{UCLK} = Frequency [Hz] of base clock selected by CKSRn.TPSn3 to CKSRn.TPSn0 bits.
k = Value set by BRGCn.MDLn7 to BRGCn.MDLn0 bits (k = 8, 9, 10, ..., 255)

(4) Baud rate error

The baud rate error is obtained by the following formula.

$$\text{Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Target baud rate (normal baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

Cautions 1. Make sure that the baud rate error during transmission does not exceed the allowable error of the reception destination.

2. Make sure that the baud rate error during reception is within the allowable baud rate range during reception, which is described in 14.6.4 Allowable baud rate range during reception.

Example: Base clock frequency = 10 MHz = 10,000,000 Hz
Setting of BRGCn.MDLn7 to BRGCn.MDLn0 bits = 00100001B (k = 33)
Target baud rate = 153,600 bps

Baud rate = 10,000,000/(2 × 33)
= 151,515 [bps]

Error = (151,515/153,600 − 1) × 100
= −1.357 [%]

Downloaded from Arrow.com.

14.6.3 Baud rate setting example

Table 14-4. Baud Rate Generator Setting Data

Baud Rate (bps)	fxx = 20 MHz			fxx = 16 MHz			fxx = 10 MHz		
	fUCLK	k	ERR	fUCLK	k	ERR	fUCLK	k	ERR
300	fxx/512	41H (65)	0.16	fxx/1024	1AH (26)	0.16	fxx/256	41H (65)	0.16
600	fxx/256	41H (65)	0.16	fxx/1024	0DH (13)	0.16	fxx/128	41H (65)	0.16
1200	fxx/128	41H (65)	0.16	fxx/512	0DH (13)	0.16	fxx/64	41H (65)	0.16
2400	fxx/64	41H (65)	0.16	fxx/256	0DH (13)	0.16	fxx/32	41H (65)	0.16
4800	fxx/32	41H (65)	0.16	fxx/128	0DH (13)	0.16	fxx/16	41H (65)	0.16
9600	fxx/16	41H (65)	0.16	fxx/64	0DH (13)	0.16	fxx/8	41H (65)	0.16
10400	fxx/64	0FH (15)	0.16	fxx/64	0CH (12)	0.16	fxx/32	0FH (15)	0.16
19200	fxx/8	41H (65)	0.16	fxx/32	0DH (13)	0.16	fxx/4	41H (65)	0.16
24000	fxx/32	0DH (13)	0.16	fxx/2	A7H (167)	–0.20	fxx/16	0DH (13)	0.16
31250	fxx/32	0AH (10)	0.00	fxx/32	08H (8)	0.00	fxx/16	0AH (10)	0
33600	fxx/2	95H (149)	–0.13	fxx/2	77H (119)	0.04	fxx	95H (149)	–0.13
38400	fxx/4	41H (65)	0.16	fxx/16	0DH (13)	0.16	fxx/2	41H (65)	0.16
48000	fxx/16	0DH (13)	0.16	fxx/2	53H (83)	0.40	fxx/8	0DH (13)	0.16
56000	fxx/2	59H (89)	0.32	fxx/2	47H (71)	0.60	fxx	59H (89)	0.32
62500	fxx/16	0AH (10)	0.00	fxx/16	08H (8)	0.00	fxx/8	0AH (10)	0.00
76800	fxx/2	41H (65)	0.16	fxx/8	0DH (13)	0.16	fxx	41H (65)	0.16
115200	fxx/2	2BH (43)	0.94	fxx/2	23H (35)	–0.79	fxx	2BH (43)	0.94
153600	fxx/2	21H (33)	–1.36	fxx/4	0DH (13)	0.16	fxx	21H (33)	–1.36
312500	fxx/4	08H (8)	0	fxx/2	0DH (13)	–1.54	fxx/2	08H (8)	0.00

Caution The allowable frequency of the base clock (f_{UCLK}) is as follows.

- **V_{DD} = 4.5 to 5.5 V: $f_{UCLK} \leq$ 12 MHz**
- **V_{DD} = 2.7 to 4.5 V: $f_{UCLK} \leq$ 6 MHz**

Remark fxx: Main clock frequency
fUCLK: Base clock frequency
k: Set values of BRGCn.MDLn7 to BRGCn.MDLn0 bits
ERR: Baud rate error [%]
n = 0, 1

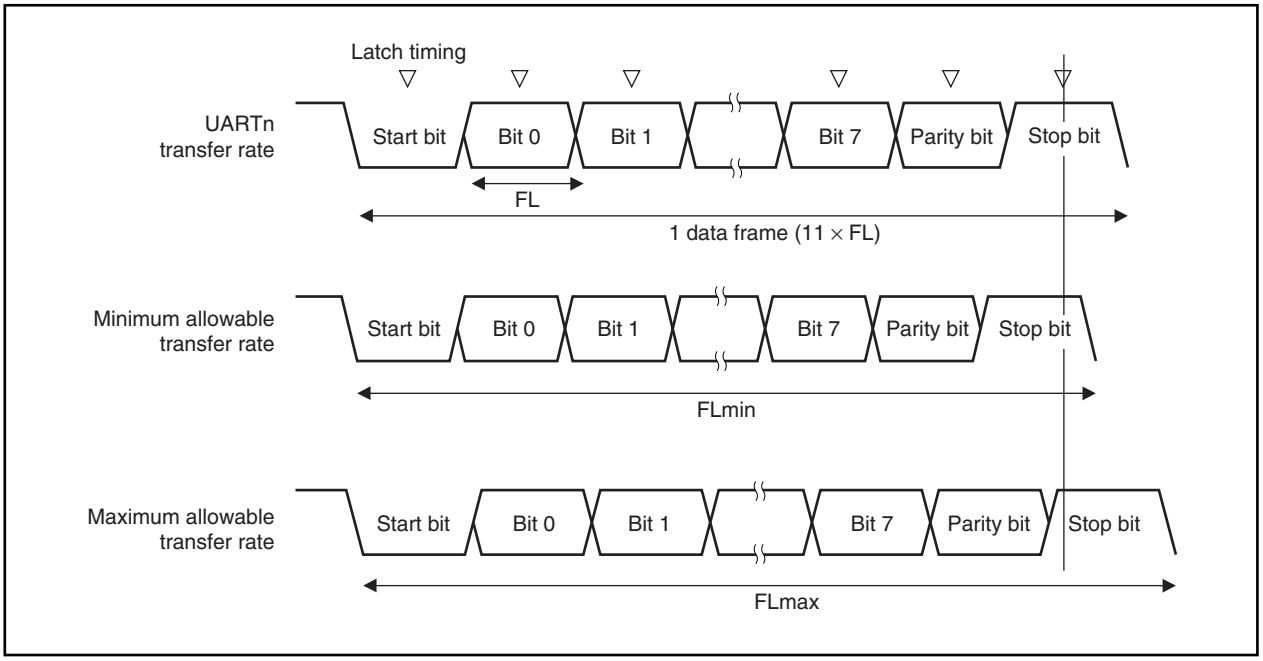
Downloaded from Arrow.com.

14.6.4 Allowable baud rate range during reception

The degree to which a discrepancy from the transmission destination's baud rate is allowed during reception is shown below.

Caution The equations described below should be used to set the baud rate error during reception so that it always is within the allowable error range.

Figure 14-13. Allowable Baud Rate Range During Reception



As shown in Figure 14-13, after the start bit is detected, the receive data latch timing is determined according to the counter that was set by the BRGCn register. If all data up to the final data (stop bit) is in time for this latch timing, the data can be received normally.
If this is applied to 11-bit reception, the following is theoretically true.

$$FL = (Brate)^{-1}$$

Brate: UARTn baud rate
k: BRGCn register set value
FL: 1-bit data length

When the latch timing margin is 2 base clocks, the minimum allowable transfer rate (FLmin) is as follows.

$$FLmin = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$$

Downloaded from Arrow.com.

Therefore, the transfer destination's maximum receivable baud rate (BRmax) is as follows.

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, the maximum allowable transfer rate (FLmax) can be obtained as follows.

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FLmax = \frac{21k - 2}{20k} FL \times 11$$

Therefore, the transfer destination's minimum receivable baud rate (BRmin) is as follows.

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

The allowable baud rate error of UARTn and the transfer destination can be obtained as follows from the expressions described above for computing the minimum and maximum baud rate values.

Table 14-5. Maximum and Minimum Allowable Baud Rate Error

Division Ratio (k)	Maximum Allowable Baud Rate Error	Minimum Allowable Baud Rate Error
8	+3.53%	–3.61%
20	+4.26%	–4.31%
50	+4.56%	–4.58%
100	+4.66%	–4.67%
255	+4.72%	–4.73%

Remarks 1. The reception precision depends on the number of bits in one frame, the base clock frequency, and the division ratio (k). The higher the base clock frequency and the larger the division ratio (k), the higher the precision.

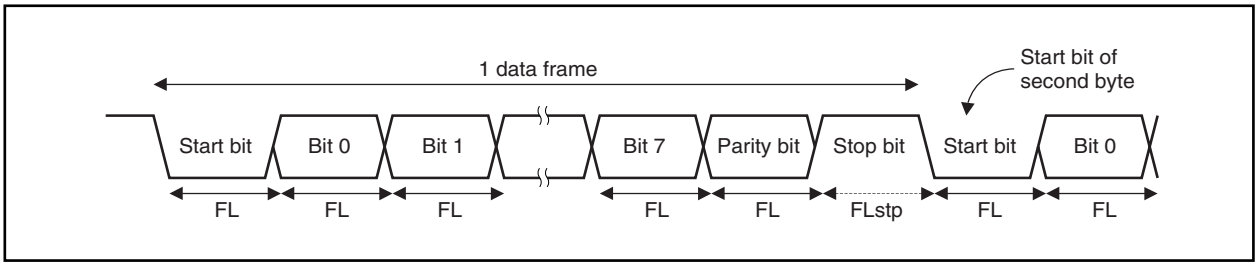
2. k: BRGCn register set value

Downloaded from Arrow.com.

14.6.5 Transfer rate during continuous transmission

During continuous transmission, the transfer rate from a stop bit to the next start bit is extended two clocks of the base clock longer than normal. However, on the reception side, the transfer result is not affected since the timing is initialized by the detection of the start bit.

Figure 14-14. Transfer Rate During Continuous Transmission



Representing the 1-bit data length by FL, the stop bit length by FLstp, and the base clock frequency by f_{UCLK} yields the following equation.

$$FLstp = FL + 2/f_{UCLK}$$

Therefore, the transfer rate during continuous transmission is as follows (when the stop bit length = 1).

$$\text{Transfer rate} = 11 \times FL + (2/f_{UCLK})$$

14.7 Cautions

Cautions to be observed when using UARTn are shown below.

(1) When the supply of clocks to UARTn is stopped (for example, in IDLE or STOP mode), operation stops with each register retaining the value it had immediately before the supply of clocks was stopped. The TXDn pin output also holds and outputs the value it had immediately before the supply of clocks was stopped. However, operation is not guaranteed after the supply of clocks is restarted. Therefore, after the supply of clocks is restarted, the circuits should be initialized by clearing the ASIMn.UARTEn, ASIMn.RXEn, and ASIMn.TXEn bits to 000.

(2) UARTn has a 2-stage buffer configuration consisting of the TXBn register and the transmission shift register, and has status flags (ASIFn.TXBFn and ASIFn.TXSFn bits) that indicate the status of each buffer. If the TXBFn and TXSFn bits are read in continuous transmission, the value changes 10 → 11 → 01. For the timing to write the next data to the TXBn register, read only the TXBFn bit during continuous transmission.

Downloaded from Arrow.com.

CHAPTER 15 CLOCKED SERIAL INTERFACE 0 (CSI0)

In the V850ES/KE1, two channels of clocked serial interface 0 (CSI0) are provided.

15.1 Features

- Maximum transfer speed: 5 Mbps
- Master mode/slave mode selectable
- Transmission data length: 8 bits or 16 bits can be set
- MSB/LSB-first selectable for transfer data
- Eight clock signals can be selected (7 master clocks and 1 slave clock)
- 3-wire type SO0n: Serial transmit data output
  SI0n: Serial receive data input
  $\overline{\text{SCK0n}}$: Serial clock I/O
- Interrupt sources: 1 type
  - Transmission/reception completion interrupt request signal (INTCSI0n)
- Transmission/reception mode or reception-only mode selectable
- Two transmission buffer registers (SOTBFn/SOTBFLn, SOTBn/SOTBLn) and two reception buffer registers (SIRBn/SIRBLn, SIRBEn/SIRBELn) are provided on chip
- Single transfer mode/continuous transfer mode selectable

Remark n = 0, 1

Downloaded from Arrow.com.

15.2 Configuration

CSI0n is controlled via the CSIM0n register.

(1) Clocked serial interface mode register 0n (CSIM0n)

The CSIM0n register is an 8-bit register that specifies the operation of CSI0n.

(2) Clocked serial interface clock selection register n (CSICn)

The CSICn register is an 8-bit register that controls the CSI0n serial transfer operation.

(3) Serial I/O shift register 0n (SIO0n)

The SIO0n register is a 16-bit shift register that converts parallel data into serial data.
The SIO0n register is used for both transmission and reception.
Data is shifted in (reception) and shifted out (transmission) from the MSB or LSB side.
The actual transmission/reception operations are started up by accessing the buffer register.

(4) Serial I/O shift register 0nL (SIO0nL)

The SIO0nL register is an 8-bit shift register that converts parallel data into serial data.
The SIO0nL register is used for both transmission and reception.
Data is shifted in (reception) and shifted out (transmission) from the MSB or LSB side.
The actual transmission/reception operations are started up by access of the buffer register .

(5) Clocked serial interface receive buffer register n (SIRBn)

The SIRBn register is a 16-bit buffer register that stores receive data.

(6) Clocked serial interface receive buffer register nL (SIRBnL)

The SIRBnL register is an 8-bit buffer register that stores receive data.

(7) Clocked serial interface read-only receive buffer register n (SIRBEn)

The SIRBEn register is a 16-bit buffer register that stores receive data.
The SIRBEn register is the same as the SIRBn register. It is used to read the contents of the SIRBn register.

(8) Clocked serial interface read-only receive buffer register nL (SIRBEnL)

The SIRBEnL register is an 8-bit buffer register that stores receive data.
The SIRBEnL register is the same as the SIRBnL register. It is used to read the contents of the SIRBnL register.

(9) Clocked serial interface transmit buffer register n (SOTBn)

The SOTBn register is a 16-bit buffer register that stores transmit data.

(10) Clocked serial interface transmit buffer register nL (SOTBLnL)

The SOTBnL register is an 8-bit buffer register that stores transmit data.

(11) Clocked serial interface initial transmit buffer register n (SOTBFn)

The SOTBFn register is a 16-bit buffer register that stores the initial transmit data in the continuous transfer mode.

Downloaded from Arrow.com.

(12) Clocked serial interface initial transmit buffer register nL (SOTBFnL)

The SOTBFnL register is an 8-bit buffer register that stores initial transmit data in the continuous transfer mode.

(13) Selector

The selector selects the serial clock to be used.

(14) Serial clock controller

Controls the serial clock supply to the shift register. Also controls the clock output to the $\overline{\text{SCK0n}}$ pin when the internal clock is used.

(15) Serial clock counter

Counts the serial clock output or input during transmission/reception, and checks whether 8-bit or 16-bit data transmission/reception has been performed.

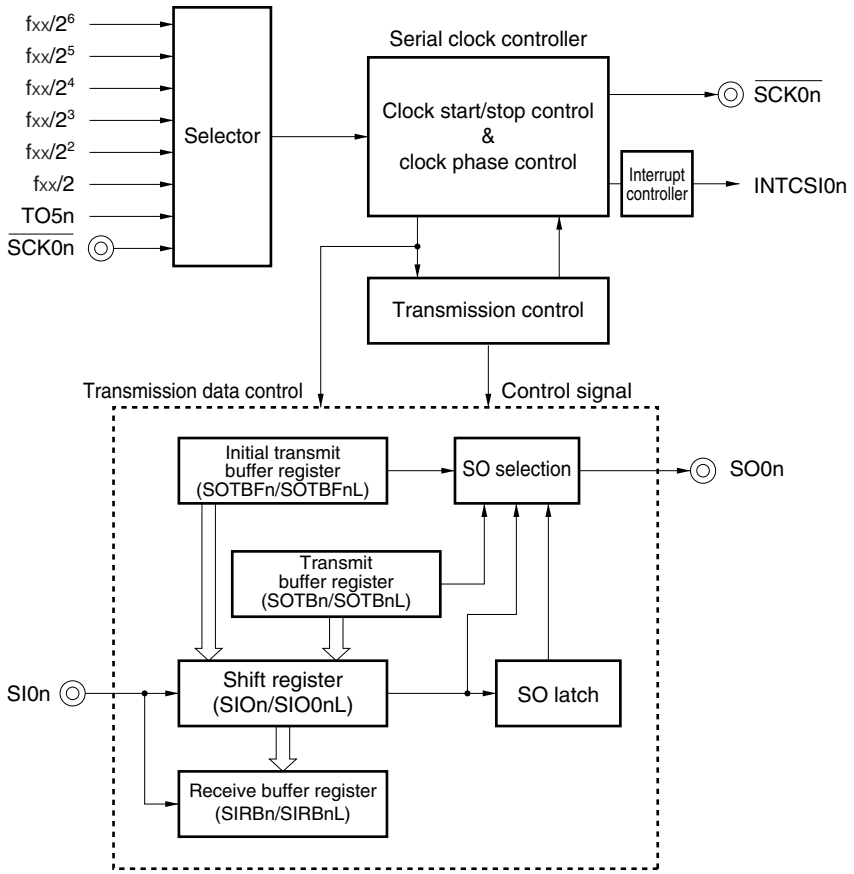
(16) Interrupt controller

Controls the interrupt request timing.

Remark n = 0, 1

Downloaded from Arrow.com.

Figure 15-1. Block Diagram of Clocked Serial Interface



Remarks 1. n = 0, 1

2. fxx: Main clock

Downloaded from Arrow.com.

15.3 Registers

(1) Clocked serial interface mode register 0n (CSIM0n)

The CSIM0n register controls the CSI0n operation.

This register can be read or written in 8-bit or 1-bit units (however, CSOTn bit is read-only).

Reset sets CSIM0n to 00H.

Caution Overwriting the CSIM0n.TRMDn, CSIM0n.CCLn, CSIM0n.DIRn, CSIM0n.CSITn, and CSIM0n.AUTOn bits can be done only when the CSOTn bit = 0. If these bits are overwritten at any other time, the operation cannot be guaranteed.

Downloaded from Arrow.com.

After reset: 00H R/W Address: CSIM00 FFFFFD00H, CSIM01 FFFFFD10H

	<7>	<6>	5	<4>	3	2	1	<0>
CSIM0n	CSI0En	TRMDn	CCLn	DIRn	CSITn	AUTOn	0	CSOTn

(n = 0, 1)

CSI0En	CSI0n operation enable/disable
0	Disable CSI0n operation.
1	Enable CSI0n operation.
The internal CSI0n circuit can be reset[Note] asynchronously by clearing the CSI0En bit to 0. For the $\overline{\text{SCK0n}}$ and SO0n pin output status when the CSI0En bit = 0, refer to **15.5 Output Pins**.	

TRMDn	Specification of transmission/reception mode
0	Receive-only mode
1	Transmission/reception mode
When the TRMDn bit = 0, reception is performed and the SO0n pin outputs a low level. Data reception is started by reading the SIRBn register. When the TRMDn bit = 1, transmission/reception is started by writing data to the SOTBn register.	

CCLn	Specification of data length
0	8 bits
1	16 bits

DIRn	Specification of transfer direction mode (MSB/LSB)
0	First bit of transfer data is MSB
1	First bit of transfer data is LSB

CSITn	Control of delay of interrupt request signal
0	No delay
1	Delay mode (interrupt request signal is delayed 1/2 cycle compared to the serial clock)
The delay mode (CSITn bit = 1) is valid only in the master mode (CSICn.CKS0n2 to CSICn.CSK0n0 bits are not 111B). In the slave mode (CKS0n2 to CKS0n0 bits are 111B), do not set the delay mode.	

AUTOn	Specification of single transfer mode or continuous transfer mode
0	Single transfer mode
1	Continuous mode

CSOTn	Communication status flag
0	Communication stopped
1	Communication in progress
The CSOTn bit is cleared (0) by writing 0 to the CSI0En bit.	

Note The CSOTn bit and the SIRBn, SIRBnL, SIRBE, SIRBEnL, SIOn, and SIOnL registers are reset.

Downloaded from Arrow.com.

(2) Clocked serial interface clock selection register n (CSICn)

The CSICn register is an 8-bit register that controls the CSI0n transfer operation.

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

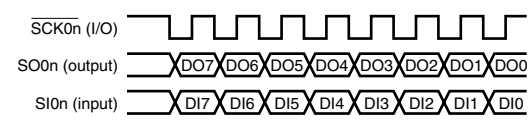
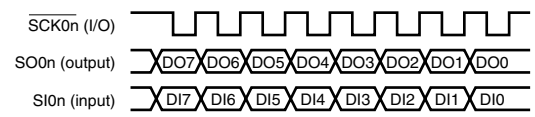
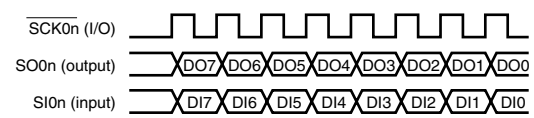
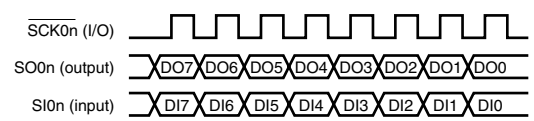
Reset sets CSICn to 00H.

Caution The CSICn register can be overwritten only when the CSIM0n.CSI0En bit = 0.

After reset: 00H R/W Address: CSIC0 FFFFFD01H, CSIC1 FFFFFD11H

	7	6	5	4	3	2	1	0
CSICn	0	0	0	CKPn	DAPn	CKS0n2	CKS0n1	CKS0n0

(n = 0, 1)

CKPn	DAPn	Specification of timing of transmitting/receiving data to/from $\overline{SCK0n}$
0	0	(Type 1) 
0	1	(Type 2) 
1	0	(Type 3) 
1	1	(Type 4) 

CKS0n2	CKS0n1	CKS0n0	Serial clock[Note]	Mode
0	0	0	$f_{XX}/2$	Master mode
0	0	1	$f_{XX}/2^2$	Master mode
0	1	0	$f_{XX}/2^3$	Master mode
0	1	1	$f_{XX}/2^4$	Master mode
1	0	0	$f_{XX}/2^5$	Master mode
1	0	1	$f_{XX}/2^6$	Master mode
1	1	0	Clock generated by TO5n	Master mode
1	1	1	External clock ($\overline{SCK0n}$ pin)	Slave mode

Note Set the serial clock so as to satisfy the following conditions.

- V_{DD} = 4.0 to 5.5 V: Serial clock ≤ 5 MHz
- V_{DD} = 2.7 to 4.0 V: Serial clock ≤ 2.5 MHz

Remark f_{XX}: Main clock frequency

Downloaded from Arrow.com.

(3) Clocked serial interface receive buffer registers n, nL (SIRBn, SIRBnL)

The SIRBn register is a 16-bit buffer register that stores receive data.

When the receive-only mode is set (CSIM0n.TRMDn bit = 0), the reception operation is started by reading data from the SIRBn register.

This register is read-only in 16-bit units. When the lower 8 bits are used as the SIRBnL register, this register is read-only in 8-bit units.

In addition to reset input, this register can also be initialized by clearing (0) the CSIM0n.CSI0En bit.

Cautions 1. Read the SIRBn register only when a 16-bit data length has been set (CSIM0n.CCLn bit = 1).

Read the SIRBnL register only when an 8-bit data length has been set (CCLn bit = 0).

2. When the single transfer mode has been set (CSIM0n.AUTOn bit = 0), perform a read operation only in the idle state (CSIM0n.CSOTn bit = 0). If the SIRBn or SIRBnL register is read during data transfer, the data cannot be guaranteed.

(a) SIRBn register

After reset: 0000H R Address: SIRB0 FFFFFD02H, SIRB1 FFFFFD12H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn	SIRBn
(n = 0, 1)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

(b) SIRBnL register

After reset: 00H R Address: SIRB0L FFFFFD02H, SIRB1L FFFFFD12H

	7	6	5	4	3	2	1	0
SIRBnL	SIRBn7	SIRBn6	SIRBn5	SIRBn4	SIRBn3	SIRBn2	SIRBn1	SIRBn0
(n = 0, 1)								

Downloaded from Arrow.com.

(4) Clocked serial interface read-only receive buffer registers n, nL (SIRBEn, SIRBEnL)

The SIRBEn register is a 16-bit buffer register that stores receive data.

The SIRBEn register is the same as the SIRBn register. Even if the SIRBEn register is read, the next operation will not start. The SIRBEn register is used to read the contents of the SIRBn register when the serial reception is not continued.

This register is read-only in 16-bit units. However, when the lower 8 bits are used as the SIRBEnL register, the register is read-only in 8-bit units.

In addition to reset input, this register can also be initialized by clearing (0) the CSIM0n.CSI0En bit.

Cautions 1. The receive operation is not started even if data is read from the SIRBEn and SIRBEnL registers.

2. The SIRBEn register can be read only if a 16-bit data length has been set (CSIM0n.CCLn bit = 1).

The SIRBEnL register can be read only if an 8-bit data length has been set (CCLn bit = 0).

(a) SIRBEn register

After reset: 0000H R Address: SIRBE0 FFFFFD06H, SIRBE1 FFFFFD16H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn	SIRBEn
(n = 0, 1)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

(B) SIRBEnL register

After reset: 00H R Address: SIRBE0L FFFFFD06H, SIRBE1L FFFFFD16H

	7	6	5	4	3	2	1	0
SIRBEnL	SIRBEn7	SIRBEn6	SIRBEn5	SIRBEn4	SIRBEn3	SIRBEn2	SIRBEn1	SIRBEn0
(n = 0, 1)								

Downloaded from Arrow.com.

(5) Clocked serial interface transmit buffer registers n, nL (SOTBn, SOTBnL)

The SOTBn register is a 16-bit buffer register that stores transmit data.

When the transmission/reception mode is set (CSIM0n.TRMDn bit = 1), the transmission operation is started by writing data to the SOTBn register.

This register can be read or written in 16-bit units. However, when the lower 8 bits are used as the SOTBnL register, the register is read-only in 8-bit units.

After reset, this register is initialized.

Cautions 1. Access the SOTBn register only when a 16-bit data length has been set (CSIM0n.CCLn bit = 1).

Access the SOTBnL register only when an 8-bit data length has been set (CCLn bit = 0).

2. When the single transfer mode is set (CSIM0n.AUTOn bit = 0), perform access only in the idle state (CSIM0n.CSOTn bit = 0). If the SOTBn and SOTBnL registers are accessed during data transfer, the data cannot be guaranteed.

(a) SOTBn register

After reset: 0000H R/W Address: SOTB0 FFFFFD04H, SOTB1 FFFFFD14H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOTBn	SOTBn 15	SOTBn 14	SOTBn 13	SOTBn 12	SOTBn 11	SOTBn 10	SOTBn 9	SOTBn 8	SOTBn 7	SOTBn 6	SOTBn 5	SOTBn 4	SOTBn 3	SOTBn 2	SOTBn 1	SOTBn 0
(n = 0, 1)																

(b) SOTBnL register

After reset: 00H R/W Address: SOTB0L FFFFFD04H, SOTB1L FFFFFD14H

	7	6	5	4	3	2	1	0
SOTBnL	SOTBn7	SOTBn6	SOTBn5	SOTBn4	SOTBn3	SOTBn2	SOTBn1	SOTBn0
(n = 0, 1)								

Downloaded from Arrow.com.

(6) Clocked serial interface initial transmit buffer registers n, nL (SOTBFn, SOTBFnL)

The SOTBFn register is a 16-bit buffer register that stores initial transmission data in the continuous transfer mode.

The transmission operation is not started even if data is written to the SOTBFn register.

This register can be read or written in 16-bit units. However, when the lower 8 bits are used as the SOTBFnL register, the register can be read or written in 8-bit units.

After reset, this register is initialized.

Caution Access the SOTBFn register and SOTBFnL register only when a 16-bit data length has been set (CSIM0n.CCLn bit = 1), and only when an 8-bit data length has been set (CCLn bit = 0), respectively, and only in the idle state (CSIM0n.CSOTn bit = 0). If the SOTBFn and SOTBFnL registers are accessed during data transfer, the data cannot be guaranteed.

(a) SOTBFn register

After reset: 0000H R/W Address: SOTBF0 FFFFFD08H, SOTBF1 FFFFFD18H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn	SOTBFn
(n = 0, 1)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

(b) SOTBFnL register

After reset: 00H R/W Address: SOTBF0L FFFFFD08H, SOTBF1L FFFFFD18H

	7	6	5	4	3	2	1	0
SOTBFnL	SOTBFn7	SOTBFn6	SOTBFn5	SOTBFn4	SOTBFn3	SOTBFn2	SOTBFn1	SOTBFn0
(n = 0, 1)								

Downloaded from Arrow.com.

(7) Serial I/O shift registers n, nL (SIO0n, SIO0nL)

The SIO0n register is a 16-bit shift register that converts parallel data into serial data.

The transfer operation is not started even if the SIO0n register is read.

This register is read-only in 16-bit units. However, when the lower 8 bits are used as the SIO0nL register, the register is read-only in 8-bit units.

In addition to reset input, this register can also be initialized by clearing (0) the CSIM0n.CSI0En bit.

Caution Read the SIO0n register and SIO0nL register only when a 16-bit data length has been set (CSIM0n.CCLn bit = 1), and only when an 8-bit data length has been set (CCLn bit = 0), respectively, and only in the idle state (CSIM0n.CSOTn bit = 0). If the SIO0n and SIO0nL registers are read during data transfer, the data cannot be guaranteed.

(a) SIO0n register

After reset: 0000H R Address: SIO00 FFFFFD0AH, SIO01 FFFFFD1AH

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIO0n	SIOn15	SIOn14	SIOn13	SIOn12	SIOn11	SIOn10	SIOn9	SIOn8	SIOn7	SIOn6	SIOn5	SIOn4	SIOn3	SIOn2	SIOn1	SIOn0

(n = 0, 1)

(b) SIO0nL register

After reset: 00H R Address: SIO00L FFFFFD0AH, SIO01L FFFFFD1AH

	7	6	5	4	3	2	1	0
SIO0nL	SIOn7	SIOn6	SIOn5	SIOn4	SIOn3	SIOn2	SIOn1	SIOn0

(n = 0, 1)

Downloaded from Arrow.com.

Downloaded from Arrow.com.

Table 15-1. Use of Each Buffer Register

Register Name	R/W		Single Transfer		Continuous Transfer[Note 1]	
			Transmission/Reception Mode	Receive-Only Mode	Transmission/Reception Mode	Receive-Only Mode
SIRBn (SIRBnL)	Read	Function	Storing received data[Note 2]	• Reading starts reception • Storing received data	Storing up to the (N – 1)th received data (other than the last)[Note 2]	• Reading starts reception • Storing up to the (N – 2)th data (other than the last two)
		Use method	When transmission and reception are complete, read the received data from this register.	• First, read dummy data and start transfer. • To perform reception of the next data after reception is complete, read the received data from this register.	When reception is complete, read the received data from this register. Repeat this operation until the (N – 1)th data has been received.	When reception is complete, read the received data from this register. Repeat this operation until the (N – 2)th data has been received. (Supplement) Do not read the (N – 1)th data from this register. If read, a reception operation starts and continuous transfer cannot be completed.
SIRBEn (SIRBEnL)	Read	Function	–	Storing the data received last[Note 2]	–	Storing the (N – 1)th received data[Note 2]
		Use method	Not used.	If reception of the next data will not be performed after reception is complete, read the received data from this register.	Not used	Read the (N – 1)th received data from this register when the (N – 1)th or Nth (last) data has been received.
SIO0n (SIO0nL)	Read	Function	–	–	Storing the Nth (last) received data[Note 2]	Storing the Nth (last) received data[Note 2]
		Use method	Not used.	Not used	When the Nth (last) transmission/reception is complete, read the Nth (last) data.	When the Nth (last) data has been received, read the Nth (last) data.
SOTBn (SOTBnL)	Write	Function	• Starting transmission/reception when written • Storing the data to be transmitted	–	• Starting transmission/reception when written • Storing the data to be transmitted second and subsequently	–
		Use method	• When transmission/reception is complete, write the data to be transmitted next.	Not used	When transmission/reception is complete, write the data to be transmitted next to this register to start the next transmission/reception.	Not used
SOTBFn (SOTBFnL)	Write	Function	–	–	Storing the data to be transmitted first[Note 2]	–
		Use method	Not used	Not used	Before starting transmission/reception (writing to SOTBn), write the data to be transmitted first.	Not used

Notes 1. It is assumed that the number of data to be transmitted is N.

2. Neither reading nor writing will start communication.

Remark In the 16-bit mode, the registers not enclosed in parentheses are used; in the 8-bit mode, the registers in parentheses are used.

15.4 Operation

15.4.1 Transmission/reception completion interrupt request signal (INTCSI0n)

The INTCSI0n signal is set (1) upon completion of data transmission/reception.

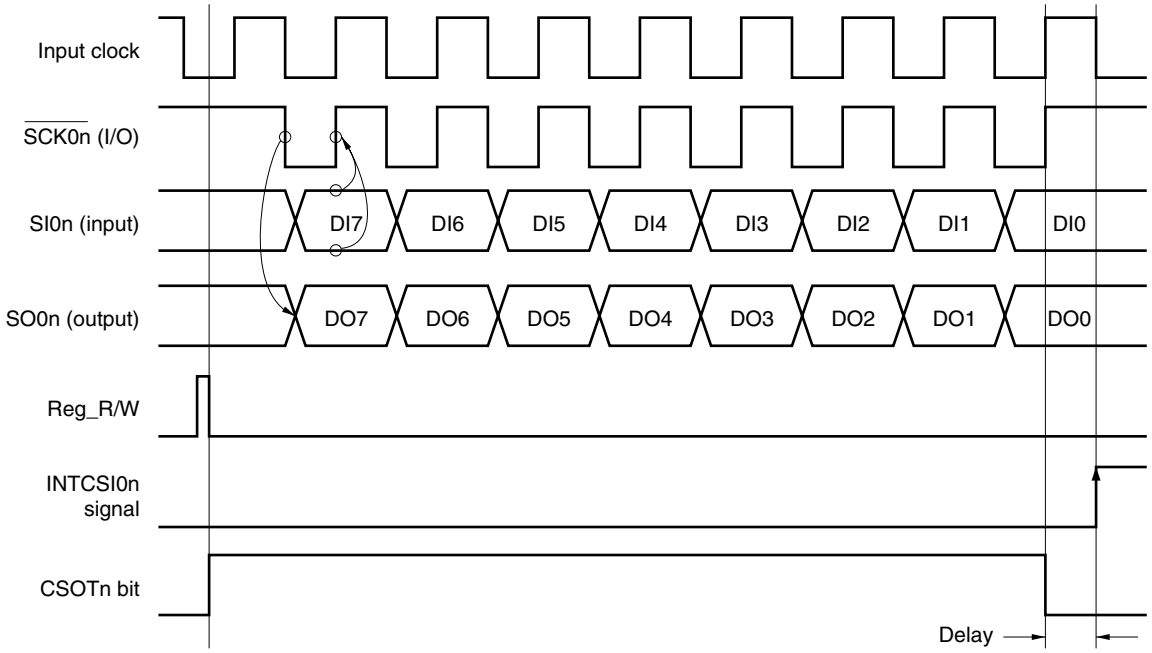
Writing to the CSIM0n register clears (0) the INTCSI0n signal.

Caution The delay mode (CSIM0n.CSITn bit = 1) is valid only in the master mode (CSICn.CKS0n2 to CSICn.CKS0n0 bits are not 111B). The delay mode cannot be set when the slave mode is set (CKS0n2 to CKS0n0 bits = 111B).

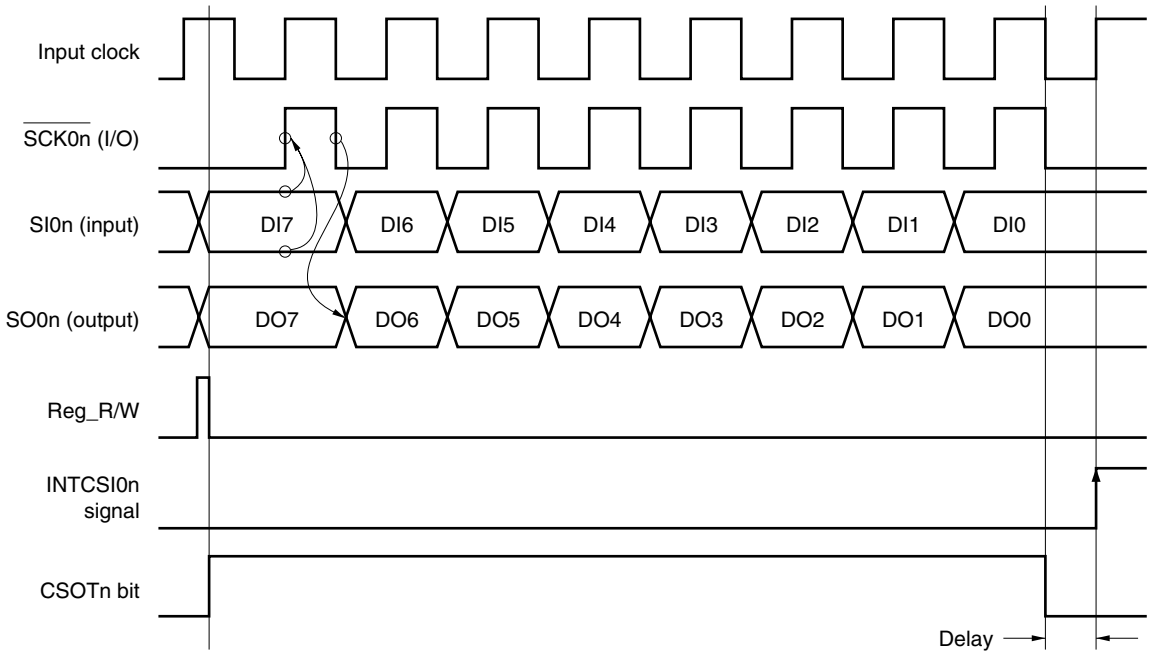
Downloaded from Arrow.com.

Figure 15-2. Timing Chart of INTCSI0n Signal Output in Delay Mode

(a) Transmit/receive type 1



(b) Transmit/receive type 4



Remarks 1. Reg_R/W: Internal signal. This signal indicates that the SIRBn/SIRBnL register read or the SOTBn/SOTBnL register write was performed.

2. n = 0, 1

Downloaded from Arrow.com.

15.4.2 Single transfer mode

(1) Usage

In the receive-only mode (CSIM0n.TRMDn bit = 0), communication is started by reading the SIRBn/SIRBnL register.

In the transmission/reception mode (TRMDn bit = 1), communication is started by writing to the SOTBn/SOTBnL register.

In the slave mode, the operation must be enabled beforehand (CSIM0n.CSI0En bit = 1).

When communication is started, the value of the CSIM0n.CSOTn bit becomes 1 (transmission execution status).

Upon communication completion, the transmission/reception completion interrupt request signal (INTCSI0n) is generated, and the CSOTn bit is cleared (0). The next data communication request is then waited for.

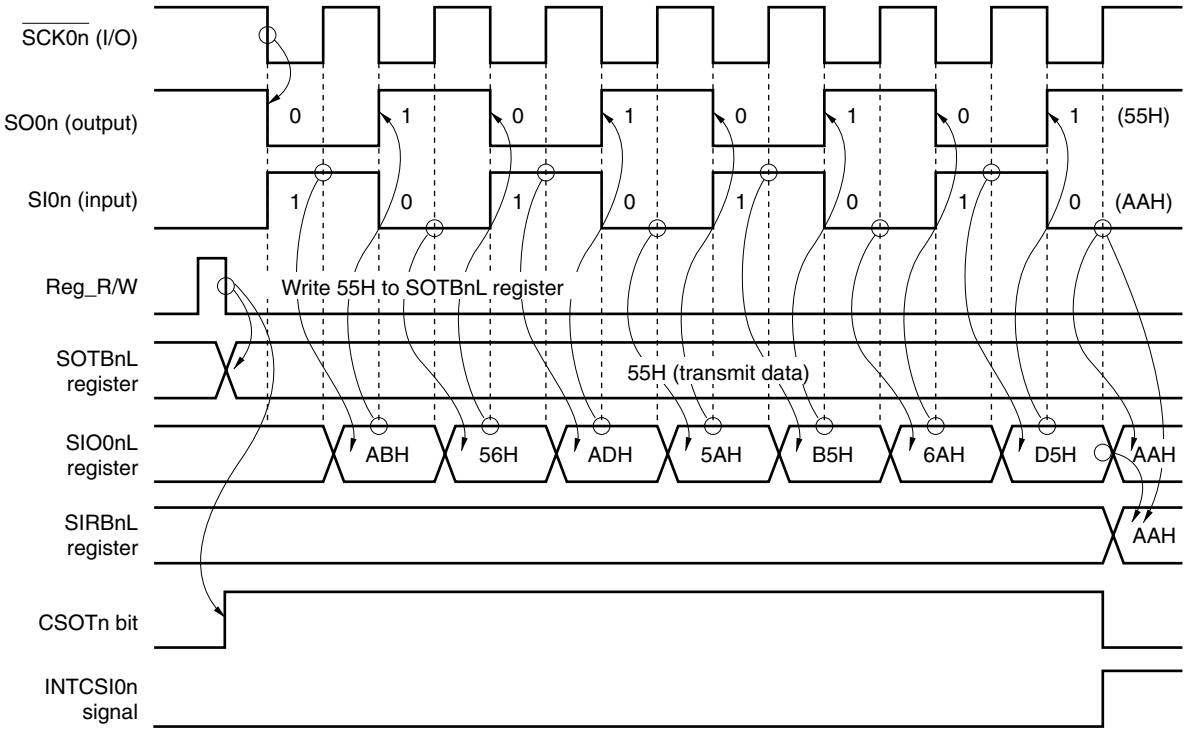
Caution When the CSOTn bit = 1, do not manipulate the CSI0n register.

Remark n = 0, 1

Downloaded from Arrow.com.

Figure 15-3. Timing Chart in Single Transfer Mode (1/2)

(a) In transmission/reception mode, data length: 8 bits, transfer direction: MSB first, no interrupt delay, single transfer mode, when AAH is received and 55H is transmitted, transmit/receive type 1



Remarks 1. Reg_R/W: Internal signal. This signal indicates that the SIRBn/SIRBnL register read or the SOTBn/SOTBnL register write was performed.

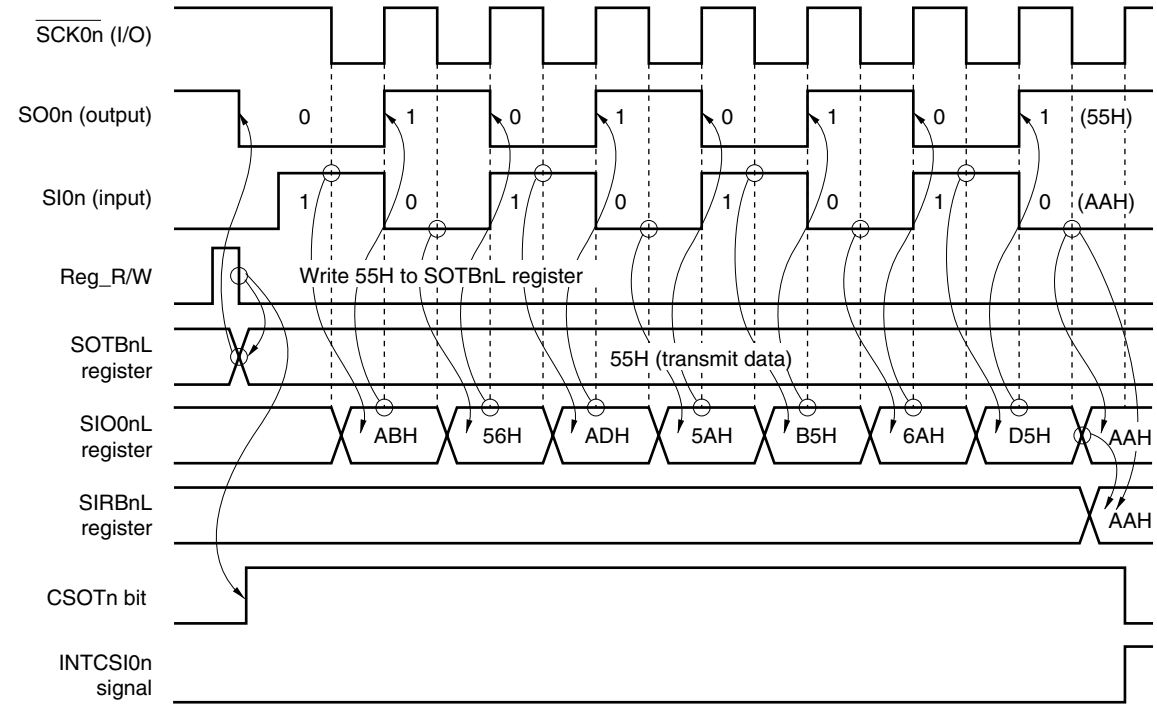
2. For the transmit/receive types, refer to **15.3 (2) Clocked serial interface clock selection register n (CSICn)**.

3. n = 0, 1

Downloaded from Arrow.com.

Figure 15-3. Timing Chart in Single Transfer Mode (2/2)

(b) In transmission/reception mode, data length: 8 bits, transfer direction: MSB first, no interrupt delay, single transfer mode, when AAH is received and 55H is transmitted, transmit/receive type 2



Remarks 1. Reg_R/W: Internal signal. This signal indicates that the SIRBn/SIRBnL register read or the SOTBn/SOTBnL register write was performed.

2. For the transmit/receive types, refer to **15.3 (2) Clocked serial interface clock selection register n (CSICn)**.

3. n = 0, 1

Downloaded from Arrow.com.

15.4.3 Continuous transfer mode

(1) Usage (receive-only: 8-bit data length)

<1> Set the continuous transfer mode (CSIM0n.AUTOn bit = 1) and the receive-only mode (CSIM0n.TRMDn bit = 0).

<2> Read the SIRBnL register (start transfer with dummy read).

<3> When the transmission/reception completion interrupt request signal (INTCSI0n) has been generated, read the SIRBnL register[Note] (reserve next transfer).

<4> Repeat step <3> (N – 2) times. (N: Number of transfer data)
Ignore the interrupt triggered by reception of the (N – 1)th data (at this time, the SIRBEnL register can be read).

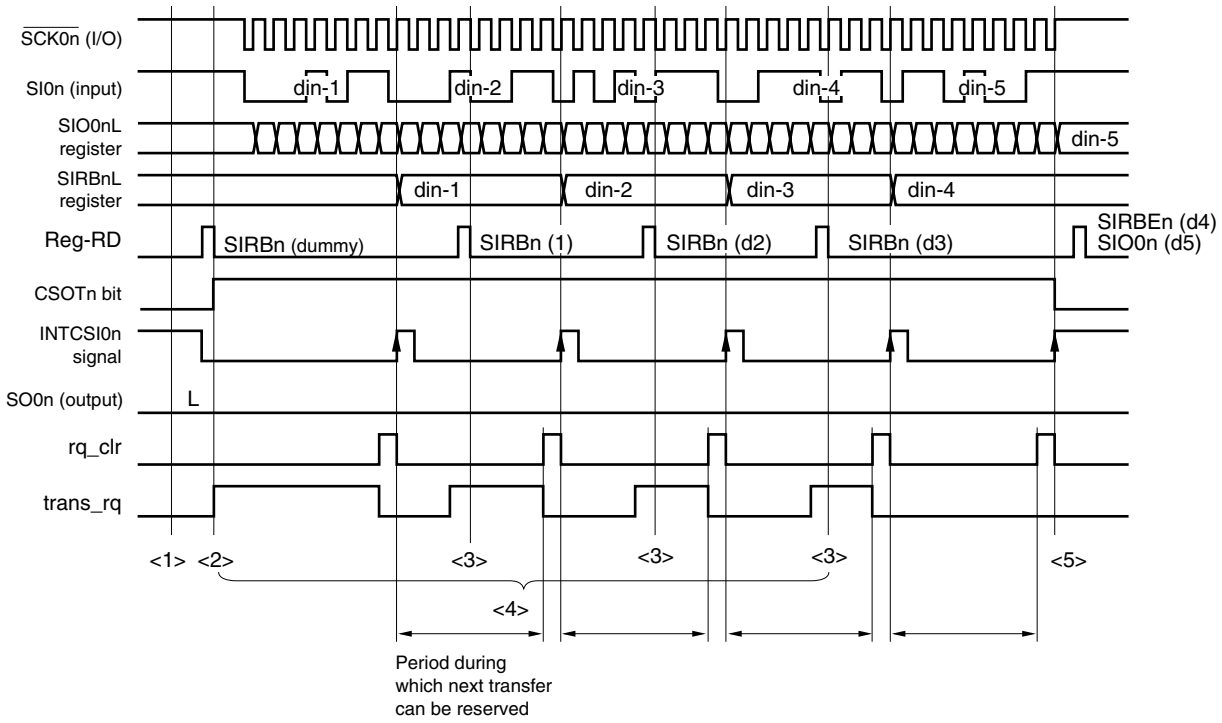
<5> Following generation of the last INTCSI0n signal, read the SIRBEnL register and the SIO0nL register[Note].

Note When transferring N number of data, receive data is loaded by reading the SIRBnL register from the first data to the (N – 2)th data. The (N – 1)th data is loaded by reading the SIRBEnL register, and the Nth (last) data is loaded by reading the SIO0nL register (refer to **Table 15-1 Use of Each Buffer Register**).

Downloaded from Arrow.com.

Figure 15-4. Continuous Transfer (Receive-Only) Timing Chart

- Transmit/receive type 1, 8-bit data length



Remarks 1. Reg_RD: Internal signal. This signal indicates that the SIRBnL register has been read.
rq_clr: Internal signal. Transfer request clear signal.
trans_rq: Internal signal. Transfer request signal.

2. n = 0, 1

In the case of the continuous transfer mode, two transfer requests are set at the start of the first transfer. Following the INTCSI0n signal, transfer is continued if the SIRBnL register can be read within the next transfer reservation period. If the SIRBnL register cannot be read, transfer ends and the SIRBnL register does not receive the new value of the SIO0nL register.

The last data can be obtained by reading the SIO0nL register following completion of the transfer.

Downloaded from Arrow.com.

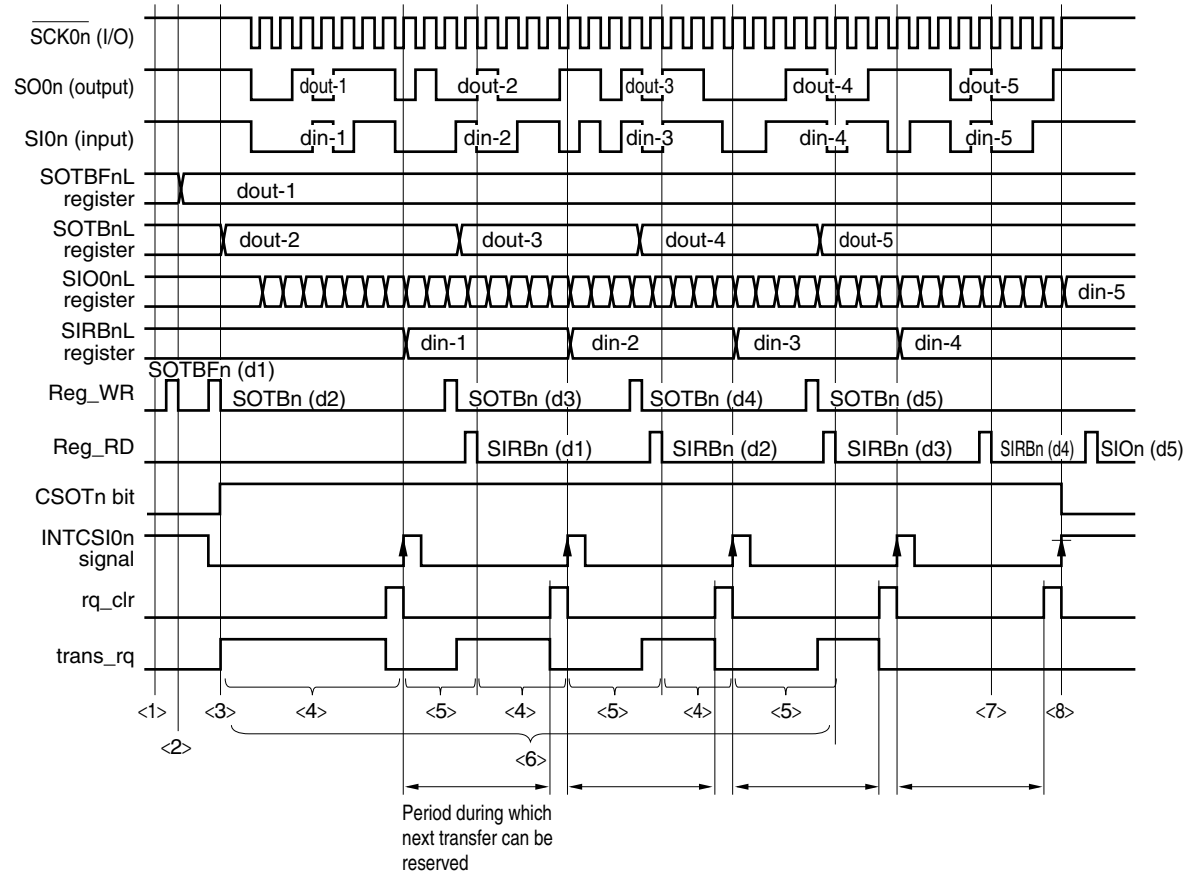
(2) Usage (transmission/reception: 8-bit data length)

<1> Set the continuous transfer mode (CSIM0n.AUTOn bit = 1) and the transmission/reception mode (CSIM0n.TRMDn bit = 1).
<2> Write the first data to the SOTBFnL register.
<3> Write the 2nd data to the SOTBnL register (start transfer).
<4> When the transmission/reception completion interrupt request signal (INTCSI0n) has been generated, write the next data to the SOTBnL register (reserve next transfer). Read the SIRBnL register to load the receive data.
<5> Repeat step <4> as long as data to be sent remains.
<6> When the INTCSI0n signal is generated, read the SIRBnL register to load the (N − 1)th receive data (N: Number of transfer data).
<7> Following the last INTCSI0n signal, read the SIO0nL register to load the Nth (last) receive data.

Downloaded from Arrow.com.

Figure 15-5. Continuous Transfer (Transmission/Reception) Timing Chart

- Transmit/receive type 1, 8-bit data length



Remarks 1. Reg_WR: Internal signal. This signal indicates that the SOTBnL register has been written.
Reg_RD: Internal signal. This signal indicates that the SIRBnL register has been read.
rq_clr: Internal signal. Transfer request clear signal.
trans_rq: Internal signal. Transfer request signal.

2. n = 0, 1

In the case of the continuous transfer mode, two transfer requests are set at the start of the first transfer. Following the INTCSI0n signal, transfer is continued if the SOTBnL register can be written within the next transfer reservation period. If the SOTBnL register cannot be written, transfer ends and the SIRBnL register does not receive the new value of the SIO0nL register.

The last receive data can be obtained by reading the SIO0nL register following completion of the transfer.

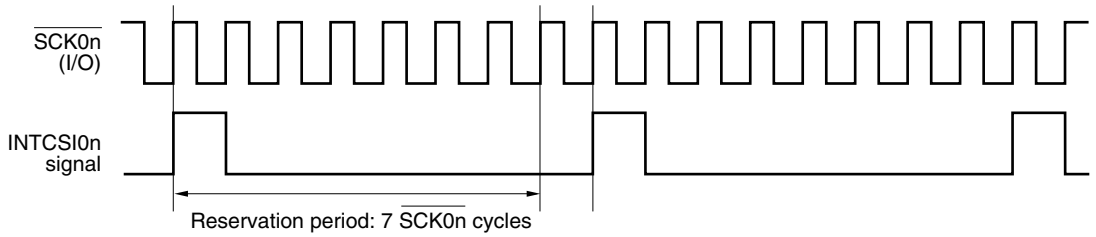
Downloaded from Arrow.com.

(3) Next transfer reservation period

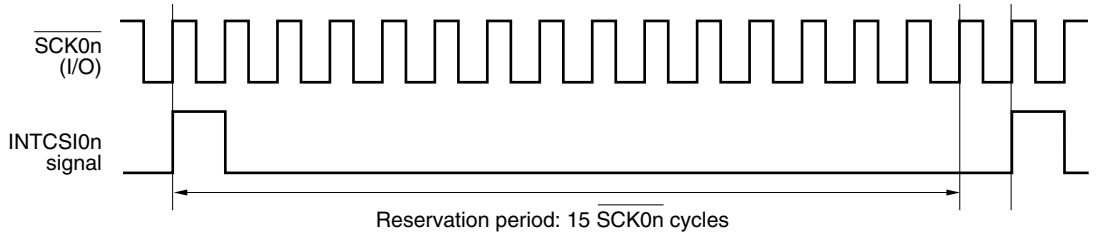
In the continuous transfer mode, the next transfer must be prepared with the period shown in Figure 15-6.

Figure 15-6. Timing Chart of Next Transfer Reservation Period (1/2)

(a) When data length: 8 bits, transmit/receive type 1



(b) When data length: 16 bits, transmit/receive type 1

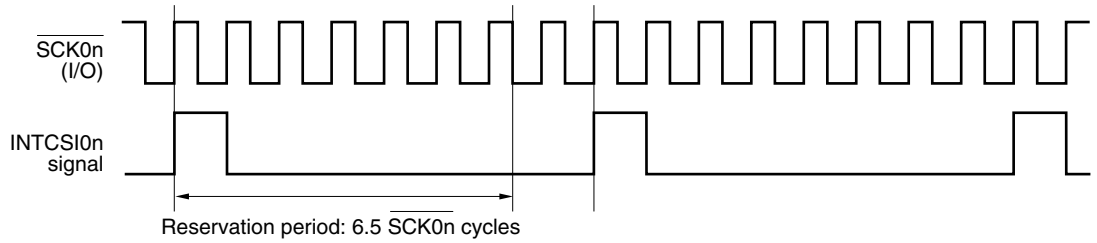


Remark n = 0, 1

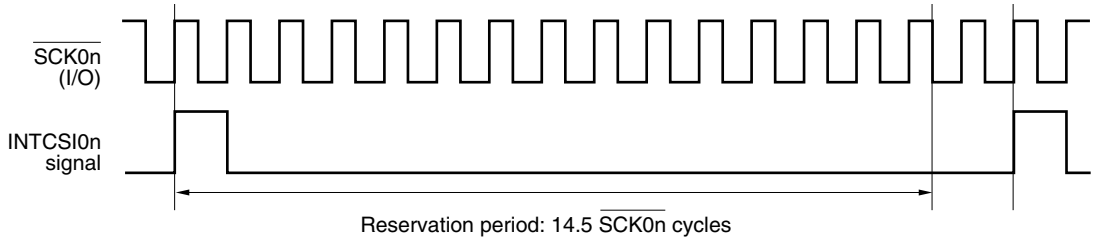
Downloaded from Arrow.com.

Figure 15-6. Timing Chart of Next Transfer Reservation Period (2/2)

(c) When data length: 8 bits, transmit/receive type 2



(d) When data length: 16 bits, transmit/receive type 2



Remark n = 0, 1

Downloaded from Arrow.com.

(4) Cautions

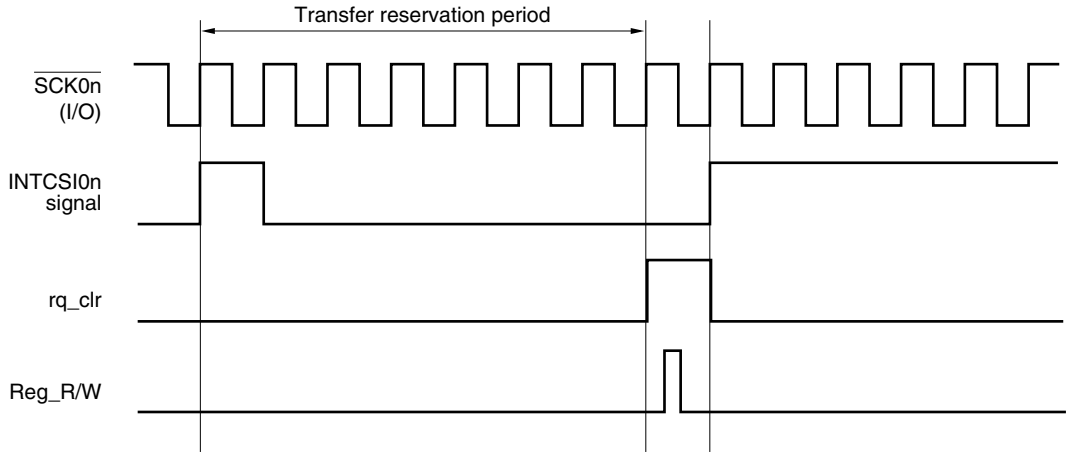
To continue continuous transfers, it is necessary to either read the SIRBn register or write to the SOTBn register during the transfer reservation period.

If access is performed to the SIRBn register or the SOTBn register when the transfer reservation period is over, the following occurs.

(i) In case of conflict between transfer request clear and register access

Since transfer request clear has higher priority, the next transfer request is ignored. Therefore, transfer is interrupted, and normal data transfer cannot be performed.

Figure 15-7. Transfer Request Clear and Register Access Conflict



Remarks 1. rq_clr: Internal signal. Transfer request clear signal.

Reg_R/W: Internal signal. This signal indicates that the SIRBn/SIRBnL register read or the SOTBn/SOTBnL register write was performed.

2. n = 0, 1

Downloaded from Arrow.com.

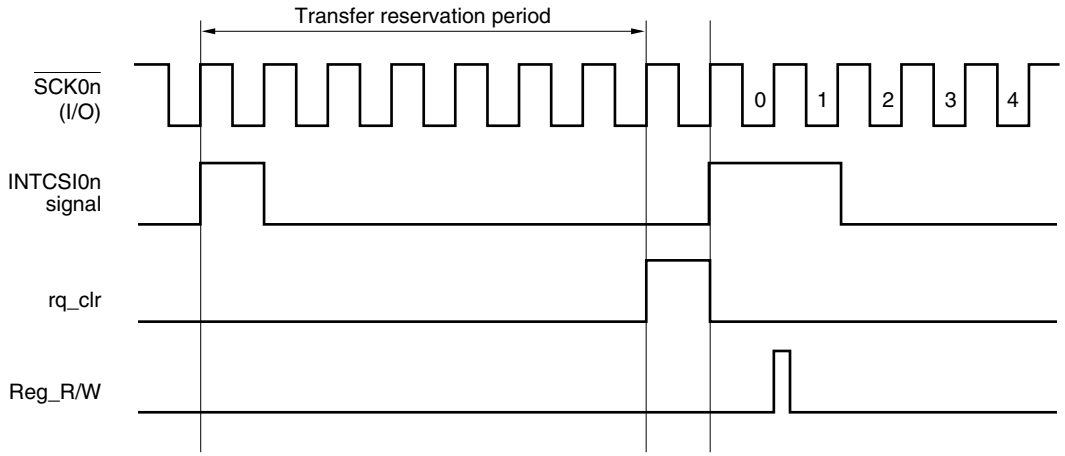
(ii) In case of conflict between transmission/reception completion interrupt request signal (INTCSI0n) generation and register access

Since continuous transfer has stopped once, executed as a new continuous transfer.

In the slave mode, a bit phase error transfer error results (refer to **Figure 15-8**).

In the transmission/reception mode, the value of the SOTBFn register is retransmitted, and illegal data is sent.

Figure 15-8. Interrupt Request and Register Access Conflict



Remarks 1. rq_clr: Internal signal. Transfer request clear signal.
Reg_R/W: Internal signal. This signal indicates that the SIRBn/SIRBnL register read or the SOTBn/SOTBnL register write was performed.

2. n = 0, 1

Downloaded from Arrow.com.

15.5 Output Pins

The following describes the output pins. For the setting of each pin, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

(1) $\overline{\text{SCK0n}}$ pin

When the CSI0n operation is disabled (CSIM0n.CSI0En bit = 0), the $\overline{\text{SCK0n}}$ pin output status is as follows.

Table 15-2. $\overline{\text{SCK0n}}$ Pin Output Status

CKPn	CKS0n2	CKS0n1	CKS0n0	$\overline{\text{SCK0n}}$ Pin Output
0	Don't care	Don't care	Don't care	Fixed to high level
1	1	1	1	High impedance
	Other than above			Fixed to low level

Remark n = 0, 1

(2) SO0n pin

When the CSI0n operation is disabled (CSI0En bit = 0), the SO0n pin output status is as follows.

Table 15-3. SO0n Pin Output Status

TRMDn	DAPn	AUTOn	CCLn	DIRn	SO0n Pin Output
0	Don't care	Don't care	Don't care	Don't care	Fixed to low level
1	0	Don't care	Don't care	Don't care	SO latch value (low level)
	1	0	0	0	SOTBn7 bit value
				1	SOTBn0 bit value
			1	0	SOTBn15 bit value
				1	SOTBn0 bit value
		1	0	0	SOTBFn7 bit value
				1	SOTBFn0 bit value
			1	0	SOTBFn15 bit value
				1	SOTBFn0 bit value

Remark n = 0, 1

Downloaded from Arrow.com.

CHAPTER 16 I²C BUS

To use the I²C bus function, use the P38/SDA0 and P39/SCL0 pins as the serial transmit/receive data I/O pin (SDA0) and the serial clock I/O pin (SCL0), respectively, and set them to N-ch open drain output.

In the V850ES/KE1, one channel of I²C bus is provided.
The μPD703207Y and 70F3207HY incorporate the I²C bus.

16.1 Features

The I²C0 has the following two modes.

- Operation stop mode
- I²C (Inter IC) bus mode (multimaster supported)

(1) Operation stop mode

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

(2) I²C bus mode (multimaster supported)

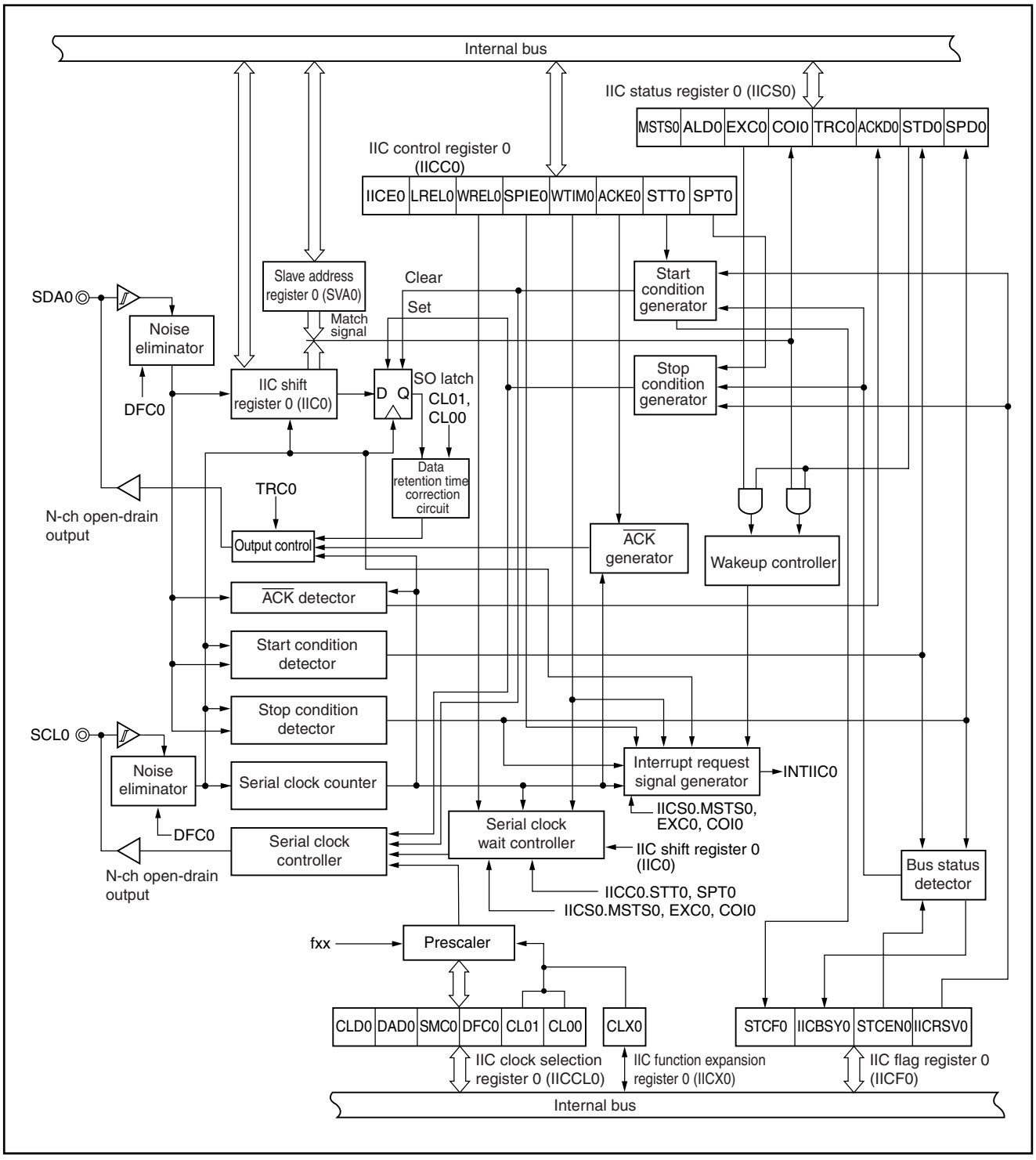
This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCL0) line and a serial data bus (SDA0) line.

This mode complies with the I²C bus format and the master device can generate “start condition”, “address”, “transfer direction specification”, “data”, and “stop condition” data to the slave device, via the serial data bus. The slave device automatically detects these received state and data by hardware. This function can simplify the part of application program that controls the I²C bus.

Since the 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.

Downloaded from Arrow.com.

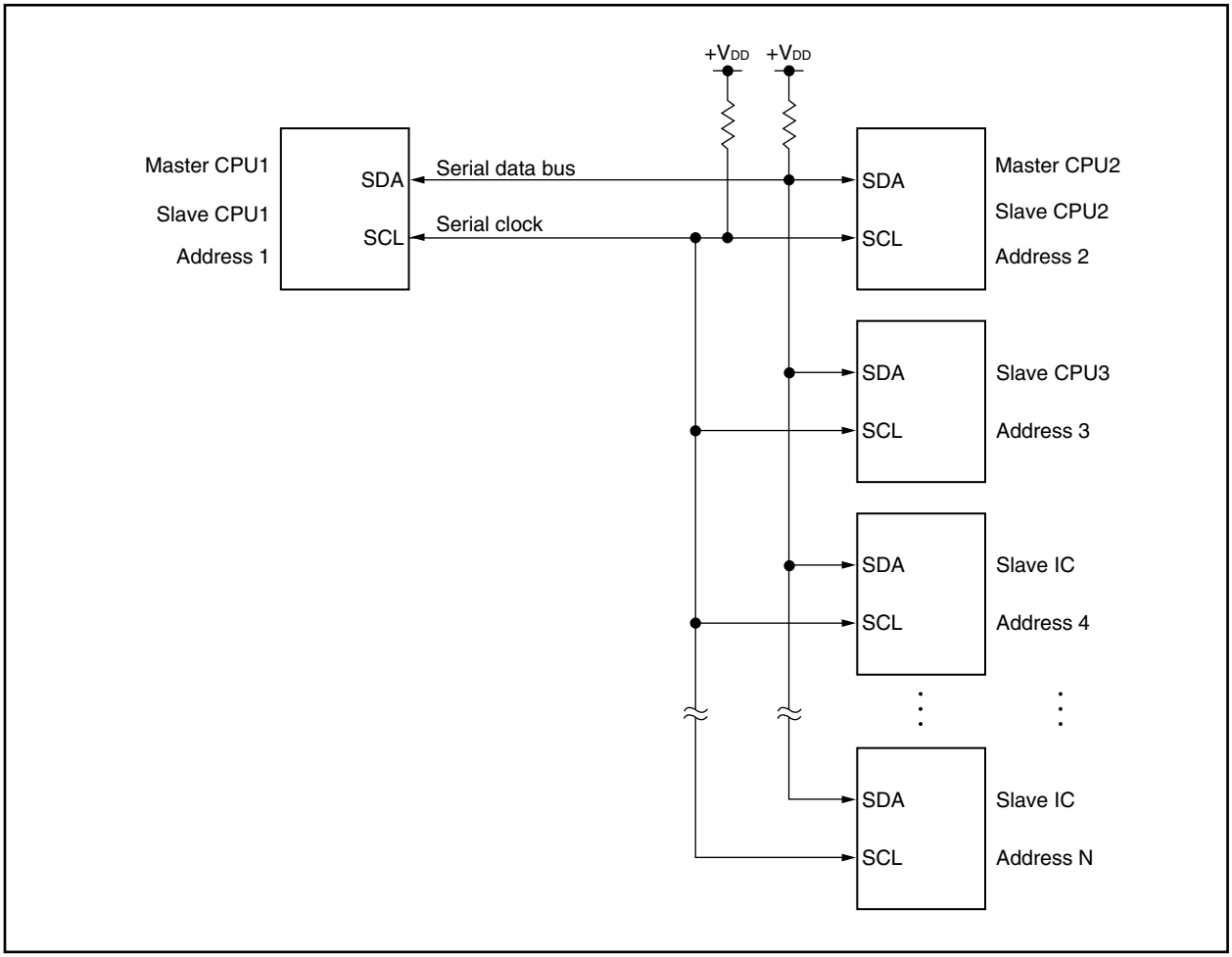
<R> **Figure 16-1. Block Diagram of I²C0**



Downloaded from Arrow.com.

A serial bus configuration example is shown below.

Figure 16-2. Serial Bus Configuration Example Using I²C Bus



Downloaded from Arrow.com.

16.2 Configuration

I²C0 includes the following hardware.

Table 16-1. Configuration of I²C0

Item	Configuration
Registers	IIC shift register 0 (IIC0) Slave address register 0 (SVA0)
Control registers	IIC control register 0 (IICC0) IIC status register 0 (IICS0) IIC flag register 0 (IICCF0) IIC clock selection register 0 (IICCL0) IIC function expansion register 0 (IICX0)

(1) IIC shift register 0 (IIC0)

The IIC0 register is used to convert 8-bit serial data to 8-bit parallel data and to convert 8-bit parallel data to 8-bit serial data. The IIC0 register 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.

The IIC0 register can be read or written in 8-bit units.

Reset sets IIC0 to 00H.

(2) Slave address register 0 (SVA0)

The SVA0 register sets local addresses when in slave mode.

The SVA0 register can be read or written in 8-bit units.

Reset sets SVA0 to 00H.

(3) SO latch

The SO latch is used to retain the SDA0 pin's output level.

(4) Wakeup controller

This circuit generates an interrupt request signal (INTIIC0) 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 sent or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).

An I²C interrupt is generated following either of two triggers.

- Falling of the eighth or ninth clock of the serial clock (set by IICC0.WTIM0 bit)
- Interrupt request generated when a stop condition is detected (set by IICC0.SPIE0 bit)

Downloaded from Arrow.com.

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) $\overline{\text{ACK}}$ generator, stop condition detector, start condition detector, and $\overline{\text{ACK}}$ detector

These circuits are used to generate and detect various statuses.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

(12) Start condition generator

This circuit generates a start condition when the IICC0.STT0 bit is set.

However, in the communication reservation disabled status (IICF0.IICRSV0 bit = 1), when the bus is not released (IICF0.IICBSY0 bit = 1), start condition requests are ignored and the IICF0.STCF0 bit is set to 1.

<R> **(13) Stop condition generator**

A stop condition is generated when the IICC0.SPT0 bit is set (1).

(14) Bus status detector

This circuit detects whether or not the bus is released by detecting start conditions and stop conditions.

However, as the bus status cannot be detected immediately following operation, the initial status is set by the IICF0.STCEN0 bit.

Downloaded from Arrow.com.

16.3 Registers

I²C0 is controlled by the following registers.

- IIC control register 0 (IICC0)
- IIC status register 0 (IICS0)
- IIC flag register 0 (IICF0)
- IIC clock selection register 0 (IICCL0)
- IIC function expansion register 0 (IICX0)

The following registers are also used.

- IIC shift register 0 (IIC0)
- Slave address register 0 (SVA0)

Remark For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

(1) IIC control register 0 (IICC0)

The IICC0 register is used to enable/stop I²C0 operations, set wait timing, and set other I²C operations.

<R> The IICC0 register can be read or written in 8-bit or 1-bit units. However, set the SPIE0, WTIM0, and ACKE0 bits when the IICE0 bit is 0 or during the wait period. When setting the IICE0 bit from “0” to “1”, these bits can also be set at the same time.

Reset sets this register to 00H.

Downloaded from Arrow.com.

(1/4)

After reset: 00H R/W Address: FFFFFD82H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICC0	IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

IICE0	I²C0 operation enable/disable specification
0	Stop operation. Reset the IICS0 register[Note 1]. Stop internal operation.
1	Enable operation.

<R> Be sure to set this bit to 1 when the SCL0 and SDA0 lines are high level.

Condition for clearing (IICE0 bit = 0)	Condition for setting (IICE0 bit = 1)
• Cleared by instruction • Reset	• Set by instruction

LREL0[Note 2]	Exit from communications
0	Normal operation
1	This exits from the current communications and sets standby mode. This setting is automatically cleared to 0 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, SPT0, IICS0.MSTS0, IICS0.EXC0, IICS0.COI0, IICS0.TRC0, IICS0.ACKD0, and IICS0.STD0 bits are cleared to 0.

The standby mode following exit from communications remains in effect until the following communications entry conditions are met.
- After a stop condition is detected, restart is in master mode.
- An address match or extension code reception occurs after the start condition.

Condition for clearing (LREL0 bit = 0)	Condition for setting (LREL0 bit = 1)
• Automatically cleared after execution • Reset	• Set by instruction

WREL0[Note 2]	Wait cancellation control
0	Do not cancel wait
1	Cancel wait. This setting is automatically cleared to 0 after wait is canceled.

Condition for clearing (WREL0 bit = 0)	Condition for setting (WREL0 bit = 1)
• Automatically cleared after execution • Reset	• Set by instruction

Notes 1. The IICS0 register, and the IICF0.STCF0, IICF0.IICBSY0, IICCL0.CLD0, and IICCL0.DAD0 bits are reset.

2. This flag's signal is invalid when the IICE0 bit = 0.

<R> **Caution If the I²C0 operation is enabled (IICE0 bit = 1) when the SCL0 line is high level and the SDA0 line is low level, the start condition is detected immediately. To avoid this, after enabling the I²C0 operation, immediately set the LREL0 bit to 1 with a bit manipulation instruction.**

Downloaded from Arrow.com.

(2/4)

SPIE0[Note]	Enable/disable generation of interrupt request when stop condition is detected
0	Disable
1	Enable

Condition for clearing (SPIE0 bit = 0)	Condition for setting (SPIE0 bit = 1)
• Cleared by instruction • Reset	• Set by instruction

WTIM0[Note]	Control of wait and interrupt request generation
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and wait is set. Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device.
1	Interrupt request is generated at the ninth clock's falling edge. Master mode: After output of nine clocks, clock output is set to low level and wait is set. Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device.

An interrupt is generated at the falling of the 9th clock during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after $\overline{ACK}$ is issued. However, when the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock.

Condition for clearing (WTIM0 bit = 0)	Condition for setting (WTIM0 bit = 1)
• Cleared by instruction • Reset	• Set by instruction

ACKE0[Note]	Acknowledgment control
0	Disable acknowledgment.
1	Enable acknowledgment. During the ninth clock period, the SDA0 line is set to low level.

The ACKE0 bit setting is invalid for address reception. In this case, $\overline{ACK}$ is generated when the addresses match. However, the ACKE0 bit setting is valid for address reception of the extension code.

Condition for clearing (ACKE0 bit = 0)	Condition for setting (ACKE0 bit = 1)
• Cleared by instruction • Reset	• Set by instruction

Note This flag's signal is invalid when the IICE0 bit = 0.

Downloaded from Arrow.com.

(3/4)

STT0	Start condition trigger
0	Do not generate a start condition.
1	When bus is released (in STOP mode): Generate a start condition (for starting as master). The SDA0 line is changed from high level to low level while the SCL0 line is high 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 (wait status). When a third party is communicating • When communication reservation function is enabled (IICF0.IICRSV0 bit = 0) Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released. • When communication reservation function is disabled (IICRSV0 bit = 1) The IICF0.STCF0 bit is set to 1 and the information set (1) to the STT0 bit is cleared. No start condition is generated. In the wait state (when master device): Generates a restart condition after releasing the wait.

Cautions concerning set timing

For master reception: Cannot be set to 1 during transfer. Can be set to 1 only when the ACKE0 bit has been cleared to 0 and slave has been notified of final reception.

For master transmission: A start condition may not be generated normally during the $\overline{\text{ACK}}$ period. Set to 1 during the wait period that follows output of the ninth clock.

- Cannot be set to 1 at the same time as the SPT0 bit.
- When the STT0 bit is set to 1, setting the STT0 bit to 1 again is disabled until the setting is cleared to 0.

Condition for clearing (STT0 bit = 0)	Condition for setting (STT0 bit = 1)
• When the STT0 bit is set to 1 in the communication reservation disabled status • Cleared when start condition is generated by master device • When the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes 1 to 0 (operation stop) • Reset	• Set by instruction

Remark The STT0 bit is 0 if it is read after data setting.

Downloaded from Arrow.com.

(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 the SCL0 pin 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 setting timing

For master reception: Cannot be set to 1 during transfer. Can be set to 1 only when the ACKE0 bit has been cleared to 0 and during the wait period after slave has been notified of final reception.

For master transmission: A stop condition may not be generated normally during the $\overline{\text{ACK}}$ period. Set to 1 during the wait period that follows output of the ninth clock.

- Cannot be set to 1 at the same time as the STT0 bit.
- The SPT0 bit can be set to 1 only when in master mode[Note].
- When the WTIM0 bit has been cleared to 0, if the SPT0 bit is set to 1 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.
  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 to 1 during the wait period that follows output of the ninth clock.
- When the SPT0 bit is set to 1, setting the SPT0 bit to 1 again is disabled until the setting is cleared to 0.

Condition for clearing (SPT0 bit = 0)	Condition for setting (SPT0 bit = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes 1 to 0 (operation stop) • Reset	• Set by instruction

Note Set the SPT0 bit to 1 only in master mode. However, the SPT0 bit must be set to 1 and a stop condition generated before the first stop condition is detected following the switch to operation enable status. For details, refer to **16.14 Cautions**.

Caution When the IICS0.TRC0 bit is set to 1, the WREL0 bit is set to 1 during the ninth clock and wait is canceled, after which the TRC0 bit is cleared to 0 and the SDA0 line is set to high impedance.

Remark The SPT0 bit is 0 if it is read after data setting.

Downloaded from Arrow.com.

(2) IIC status register 0 (IICS0)

The IICS0 register indicates the status of the I²C0 bus.

The IICS0 register is read-only, in 8-bit or 1-bit units.

<R> However, the IICS0 register can only be read when the IICC0.STT0 bit is 1 or during the wait period.

Reset sets this register to 00H.

Caution When the main clock is stopped and the CPU is operating on the subclock, do not access the IICS0 register.
For details, refer to 3.4.8 (1) (b).

(1/3)

After reset: 00H R Address: FFFFFD86H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0

MSTS0	Master device status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTS0 bit = 0)	Condition for setting (MSTS0 bit = 1)
• When a stop condition is detected • When the ALD0 bit = 1 (arbitration loss) • Cleared by the IICC0.LREL0 bit = 1 (exit from communications) • When the IICC0.IICE0 bit changes from 1 to 0 (operation stop) • Reset	• When a start condition is generated

ALD0	Detection of arbitration loss
0	This status means either that there was no arbitration or that the arbitration result was a “win”.
1	This status indicates the arbitration result was a “loss”. The MSTS0 bit is cleared to 0.

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) • Reset	• When the arbitration result is a “loss”.

Note The ALD0 bit is also cleared when a bit manipulation instruction is executed for another bit in the IICS0 register.

Downloaded from Arrow.com.

(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 the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 (operation stop) • Reset	• When the higher four bits of the received address data is either “0000” or “1111” (set at the rising edge of the eighth clock).

COI0	Detection of matching addresses
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 the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 • Reset	• When the received address matches the local address (SVA0 register) (set at the rising edge of the eighth clock).

TRC0	Detection of transmit/receive status
0	Receive status (other than transmit status). The SDA0 line is set for high impedance.
1	Transmit status. The value in the SO latch is enabled for output to the SDA0 line (valid starting at the rising edge of the first byte’s ninth clock).

<R>

Condition for clearing (TRC0 bit = 0)	Condition for setting (TRC0 bit = 1)
• When a stop condition is detected • Cleared by the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 (operation stop) • Cleared by the IICC0.WREL0 bit = 1[Note] (wait release) • When the ALD0 bit changes from 0 to 1 (arbitration loss) • 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 • When “0” is output to the first byte’s LSB (transfer direction specification bit) Slave • When “1” is input in the first byte’s LSB (transfer direction specification bit)

Note The IICS0.TRC0 bit is cleared to 0 and the SDA0 line become high impedance when the IICC0.WREL0 bit is set to 1 and wait state is released at the ninth clock with the TRC0 bit = 1.

Downloaded from Arrow.com.

(3/3)

ACKD0	Detection of $\overline{ACK}$
0	$\overline{ACK}$ was not detected.
1	$\overline{ACK}$ 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 the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 (operation stop) • Reset	• After the SDA0 pin is set to low level at the rising edge of the SCL0 pin's ninth clock

STD0	Detection of start condition
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect

Condition for clearing (STD0 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 the LREL0 bit = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 (operation stop) • Reset	• When a start condition is detected

SPD0	Detection of stop condition
0	Stop condition was not detected.
1	Stop condition was detected. The master device's communication is terminated and the bus is released.

Condition for clearing (SPD0 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) • Reset	• When a stop condition is detected

Downloaded from Arrow.com.

(3) IIC flag register 0 (IICF0)

IICF0 is a register that set the operation mode of I²C0 and indicate the status of the I²C bus.

These registers can be read or written in 8-bit or 1-bit units. However, the STCF0 and IICBSY0 bits are read-only.

The IICRSV0 bit can be used to enable/disable the communication reservation function (refer to **16.13 Communication Reservation**).

The STCEN0 bit can be used to set the initial value of the IICBSY0 bit (refer to **16.14 Cautions**).

The IICRSV0 and STCEN0 bits can be written only when the operation of I²C0 is disabled (IICC0.IICE0 bit = 0).

When operation is enabled, the IICF0 register can be read.

Reset sets this register to 00H.

Downloaded from Arrow.com.

After reset: 00H R/W[Note] Address: FFFFFD8AH

	<7>	<6>	5	4	3	2	<1>	<0>
IICF0	STCF0	IICBSY0	0	0	0	0	STCEN0	IICRSV0

STCF0	IICC0.STT0 clear flag
0	Generate start condition
1	Start condition generation unsuccessful: clear STT0 flag

Condition for clearing (STCF0 bit = 0)	Condition for setting (STCF0 bit = 1)
• Clearing by setting the STT0 bit = 1 • When the IICE0 bit changes 1 to 0 (operation stop) • Reset	• Generating start condition unsuccessful and the STT0 bit cleared to 0 when communication reservation is disabled (IICRSV0 bit = 1).

<R>

IICBSY0	I²C0 bus status flag
0	Bus release status (initial communication status when STCEN0 bit = 1)
1	Bus communication status (initial communication status when STCEN0 bit = 0)

Condition for clearing (IICBSY0 bit = 0)	Condition for setting (IICBSY0 bit = 1)
• Detection of stop condition • When the IICE0 bit changes 1 to 0 (operation stop) • Reset	• Detection of start condition • Setting of the IICE0 bit when the STCEN0 bit = 0Q

<R>

STCEN0	Initial start enable trigger
0	After operation is enabled (IICE0 bit = 1), enable generation of a start condition upon detection of a stop condition.
1	After operation is enabled (IICE0 bit = 1), enable generation of a start condition without detecting a stop condition.

Condition for clearing (STCEN0 bit = 0)	Condition for setting (STCEN0 bit = 1)
• Detection of start condition • Reset	• Setting by instruction

IICRSV0	Communication reservation function disable bit
0	Enable communication reservation
1	Disable communication reservation

Condition for clearing (IICRSV0 bit = 0)	Condition for setting (IICRSV0 bit = 1)
• Clearing by instruction • Reset	• Setting by instruction

Note Bits 6 and 7 are read-only bits.

Cautions 1. Write to the STCEN0 bit only when the operation is stopped (IICE0 bit = 0).

2. As the bus release status (IICBSY0 bit = 0) is recognized regardless of the actual bus status when the STCEN0 bit = 1, when generating the first start condition (STT0 bit = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.

3. Write to the IICRSV0 bit only when the operation is stopped (IICE0 bit = 0).

Downloaded from Arrow.com.

(4) IIC clock selection register 0 (IICCL0)

The IICCL0 register is used to set the transfer clock for the I²C0 bus.

The IICCL0 register 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 in combination with the IICX0.CLX0 bit (refer to **16.3 (6) I²C0 transfer clock setting method**).

<R> Set the IICCL0 register when the IICC0.IICE0 bit = 0.

Reset sets this register to 00H.

After reset: 00H R/W[Note] Address: IICCL0 FFFFFD84H

	7	6	<5>	<4>	3	2	1	0
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00

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 changes 1 to 0 (operation stop) • 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 IICE0 bit changes 1 to 0 (operation stop) • Reset	• When the SDA0 pin is at high level

SMC0	Operation mode switching
0	Operates in standard mode.
1	Operates in high-speed mode.

DFC0	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
Digital filter can be used only in high-speed mode. In high-speed mode, the transfer clock does not vary regardless of DFC0 bit set/clear. The digital filter is used for noise elimination in high-speed mode.	

Note Bits 4 and 5 are read-only bits.

Downloaded from Arrow.com.

(5) IIC function expansion register 0 (IICX0)

These registers set the function expansion of I²C0 (valid only in high-speed mode).

These registers can be read or written in 8-bit or 1-bit units. The CLX0 bit is set in combination with the IICCL0.SMC0, IICCL0.CL01, and IICCL0.CL00 bits (refer to **16.3 (6) I²C0 transfer clock setting method**).

<R> Set the IICX0 register when the IICC0.IICE0 bit = 0.

Reset sets this register to 00H.

After reset: 00H R/W Address: IICX0 FFFFFD85H

	7	6	5	4	3	2	1	<0>
IICX0	0	0	0	0	0	0	0	CLX0

(6) I²C0 transfer clock setting method

The I²C0 transfer clock frequency (f_{SCL}) is calculated using the following expression.

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

m = 12, 24, 48, 54, 86, 88, 172, 198 (refer to **Table 16-2 Selection Clock Setting**.)

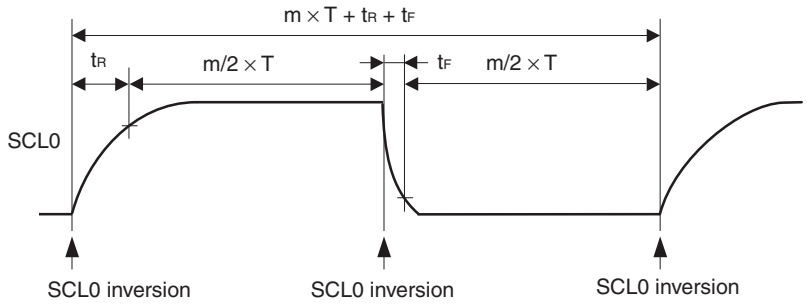
T: $1/f_{XX}$

t_R: SCL0 rise time

t_F: SCL0 fall time

For example, the I²C0 transfer clock frequency (f_{SCL}) when f_{XX} = 20 MHz, m = 54, t_R = 200 ns, and t_F = 50 ns is calculated using following expression.

$$f_{SCL} = 1/(54 \times 50 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 339 \text{ kHz}$$



The selection clock is set using a combination of the IICCL0.SMC0, IICCL0.CL01, and IICCL0.CL00 bits and the IICX0.CLX0 bit.

Downloaded from Arrow.com.

Table 16-2. Selection Clock Setting

IICX0	IICCL0			Selection Clock	Transfer Clock (f_{XX}/m)	Settable Internal System Clock Frequency (f_{XX}) Range	Operation Mode
Bit 0	Bit 3	Bit 1	Bit 0				
CLX0	SMC0	CL01	CL00				
0	0	0	0	f_{XX}/2	f_{XX}/88	4.0 MHz to 8.38 MHz	Normal mode (SMC0 bit = 0)
0	0	0	1	f_{XX}/2	f_{XX}/172	8.38 MHz to 16.76 MHz	
0	0	1	0	f_{XX}	f_{XX}/86	4.19 MHz to 8.38 MHz	
0	0	1	1	f_{XX}/3	f_{XX}/198	16.0 MHz to 19.8 MHz	
0	1	0	x	f_{XX}/2	f_{XX}/48	8 MHz to 16.76 MHz	High-speed mode (SMC0 bit = 1)
0	1	1	0	f_{XX}	f_{XX}/24	4 MHz to 8.38 MHz	
0	1	1	1	f_{XX}/3	f_{XX}/54	16 MHz to 20 MHz	
1	0	x	x	Setting prohibited			
1	1	0	x	f_{XX}/2	f_{XX}/24	8.00 MHz to 8.38 MHz	High-speed mode (SMC0 bit = 1)
1	1	1	0	f_{XX}	f_{XX}/12	4.00 MHz to 4.19 MHz	
1	1	1	1	Setting prohibited			

Remark x: don't care

Downloaded from Arrow.com.

(7) IIC shift register 0 (IIC0)

The IIC0 register is used for serial transmission/reception (shift operations) that is synchronized with the serial clock.

The IIC0 register can be read or written in 8-bit units, but data should not be written to the IIC0 register during a data transfer.

<R> Access (read/write) the IIC0 register only during the wait period. Accessing this register in communication states other than the wait period is prohibited. However, for the master device, the IIC0 register can be written once only after the transmission trigger bit (IICC0.STT0 bit) has been set to 1.

When the IIC0 register is written during wait, the wait is cancelled and data transfer is started.

Reset sets this register to 00H.

After reset: 00H R/W Address: IIC0 FFFFFD80H

	7	6	5	4	3	2	1	0
IIC0								

(8) Slave address register 0 (SVA0)

The SVA0 register holds the I²C bus's slave addresses.

The SVA0 register can be read or written in 8-bit units, but bit 0 is fixed to 0.

<R> However, rewriting this register is prohibited when the IICS0.STD0 bit = 1 (start condition detection).

Reset sets this register to 00H.

After reset: 00H R/W Address: SVA0 FFFFFD83H

	7	6	5	4	3	2	1	0
SVA0								0

Downloaded from Arrow.com.

16.4 Functions

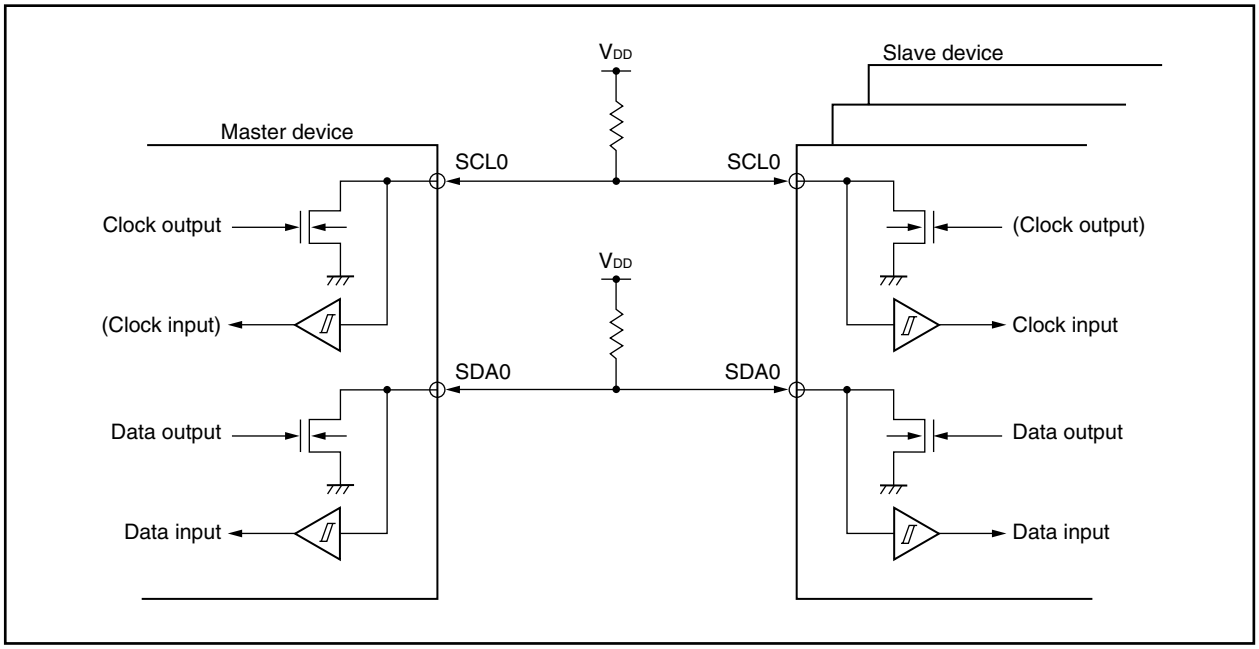
16.4.1 Pin configuration

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 16-3. Pin Configuration Diagram

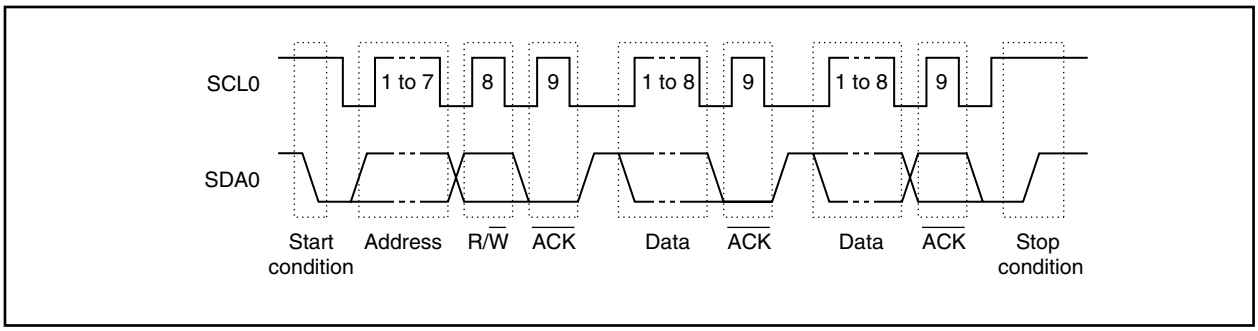


Downloaded from Arrow.com.

16.5 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the status generated by the I²C bus. The transfer timing for the "start condition", "address", "transfer direction specification", "data", and "stop condition" generated via the I²C bus's serial data bus is shown below.

Figure 16-4. I²C Bus's Serial Data Transfer Timing



The master device generates the start condition, slave address, and stop condition.

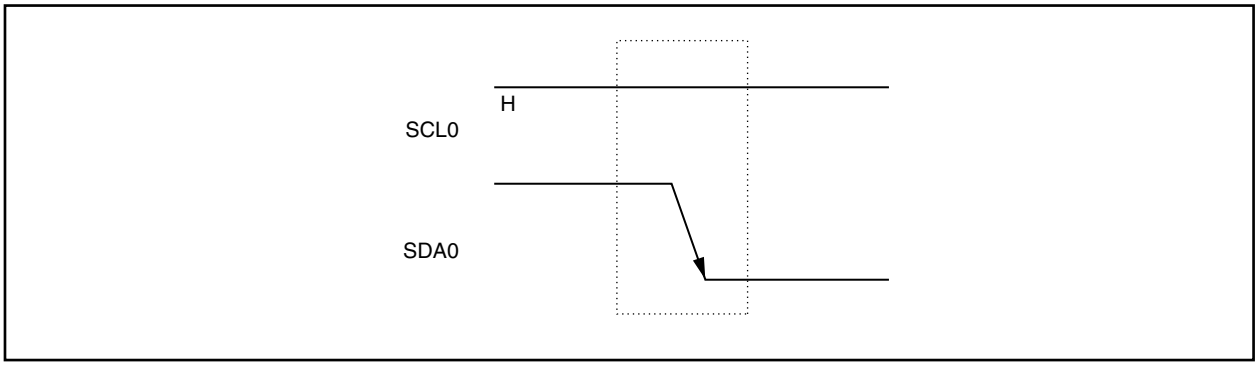
$\overline{\text{ACK}}$ can be generated by either the master or slave device (normally, it is generated by the device that receives 8-bit data).

The serial clock (SCL0) is continuously output by the master device. However, in the slave device, the SCL0's low-level period can be extended and a wait can be inserted.

16.5.1 Start condition

A start condition is met when the SCL0 pin is at high level and the SDA0 pin changes from high level to low level. The start conditions for the SCL0 pin and SDA0 pin are generated when the master device starts a serial transfer to the slave device. Start conditions can be detected when the device is used as a slave.

Figure 16-5. Start Conditions



A start condition is generated when the IICC0.STT0 bit is set to 1 after a stop condition has been detected (IICS0.SPD0 bit = 1). When a start condition is detected, IICS0.STD0 bit is set to 1.

Downloaded from Arrow.com.

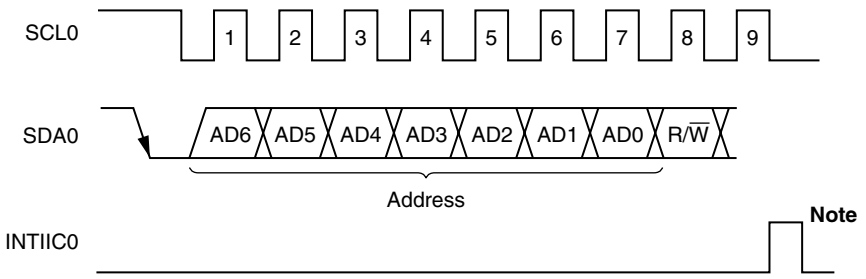
16.5.2 Addresses

The 7 bits of data that follow the start condition are defined as an address.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the SVA0 register. If the address data matches the SVA0 register values, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition.

Figure 16-6. 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 **16.5.3 Transfer direction specification** below, are together written to the IIC0 register and are then output. Received addresses are written to the IIC0 register.

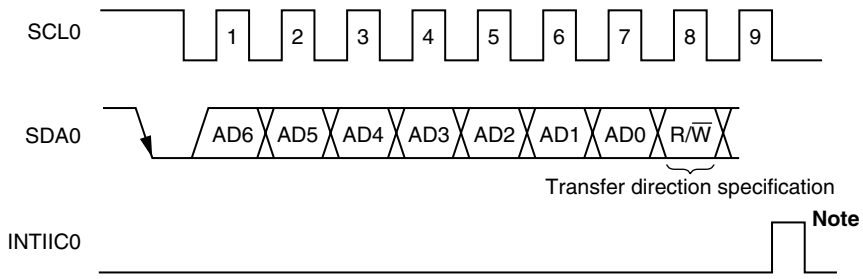
The slave address is assigned to the higher 7 bits of the IIC0 register.

Downloaded from Arrow.com.

16.5.3 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 16-7. Transfer Direction Specification



Note The interrupt request signal (INTIIC0) is generated if a local address or extension code is received during slave device operation.

Downloaded from Arrow.com.

<R> 16.5.4 $\overline{\text{ACK}}$

$\overline{\text{ACK}}$ is used to confirm the serial data status of the transmitting and receiving devices.

The receiving device returns $\overline{\text{ACK}}$ for every 8 bits of data it receives.

The transmitting device normally receives $\overline{\text{ACK}}$ after transmitting 8 bits of data. When $\overline{\text{ACK}}$ is returned from the receiving device, the reception is judged as normal and processing continues. The detection of $\overline{\text{ACK}}$ is confirmed with the IICS0.ACKD0 bit.

When the master device is the receiving device, after receiving the final data, it does not return $\overline{\text{ACK}}$ and generates the stop condition. When the slave device is the receiving device and does not return $\overline{\text{ACK}}$, the master device generates either a stop condition or a restart condition, and then stops the current transmission. Failure to return $\overline{\text{ACK}}$ may be caused by the following factors.

(a) Reception was not performed normally.
(b) The final data was received.
(c) The receiving device (slave) does not exist for the specified address.

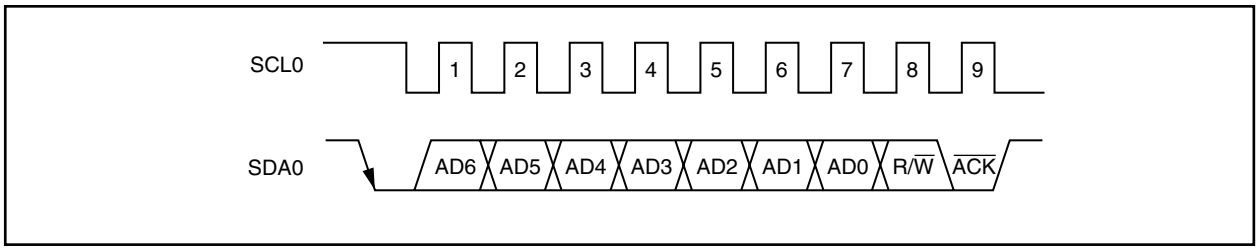
When the receiving device sets the SDA0 line to low level during the ninth clock, $\overline{\text{ACK}}$ is generated (normal reception).

When the IICC0.ACKE0 bit is set to 1, automatic $\overline{\text{ACK}}$ generation is enabled. Transmission of the eighth bit following the 7 address data bits causes the IICS0.TRC0 bit to be set. Normally, set the ACKE0 bit to 1 for reception (TRC0 bit = 0).

When the slave device is receiving (when TRC0 bit = 0), if the slave device cannot receive data or does not need to receive any more data, clear the ACKE0 bit to 0 to indicate to the master that no more data can be received.

Similarly, when the master device is receiving (when TRC0 bit = 0) and the subsequent data is not needed, clear the ACKE0 bit to 0 to prevent $\overline{\text{ACK}}$ from being generated. This notifies the slave device (transmitting device) of the end of the data transmission (transmission stopped).

Figure 16-8. $\overline{\text{ACK}}$



When the local address is received, $\overline{\text{ACK}}$ is automatically generated regardless of the value of the ACKE0 bit. No $\overline{\text{ACK}}$ is generated if the received address is not a local address (NACK).

When receiving the extension code, set the ACKE0 bit to 1 in advance to generate $\overline{\text{ACK}}$.

The $\overline{\text{ACK}}$ generation method during data reception is based on the wait timing setting, as described by the following.

- When 8-clock wait is selected (IICC0.WTIM0 bit = 0):
  $\overline{\text{ACK}}$ is generated at the falling edge of the SCL0n pin's eighth clock if the ACKE0 bit is set to 1 before the wait state cancellation.
- When 9-clock wait is selected (IICC0.WTIM0 bit = 1):
  $\overline{\text{ACK}}$ is generated if the ACKE0 bit is set to 1 in advance.

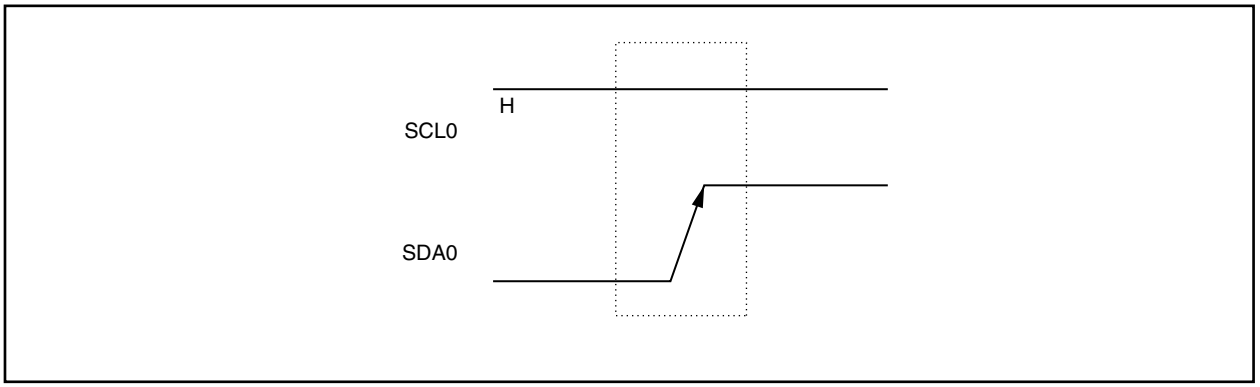
Downloaded from Arrow.com.

16.5.5 Stop condition

When the SCL0 pin is at high level, changing the SDA0 pin from low level to high level generates a stop condition.

A stop condition is generated when serial transfer from the master device to the slave device has been completed. Stop conditions can be detected when the device is used as a slave.

Figure 16-9. 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 interrupt request signal (INTIIC0) is generated when the IICC0.SPIE0 bit is set to 1.

Downloaded from Arrow.com.

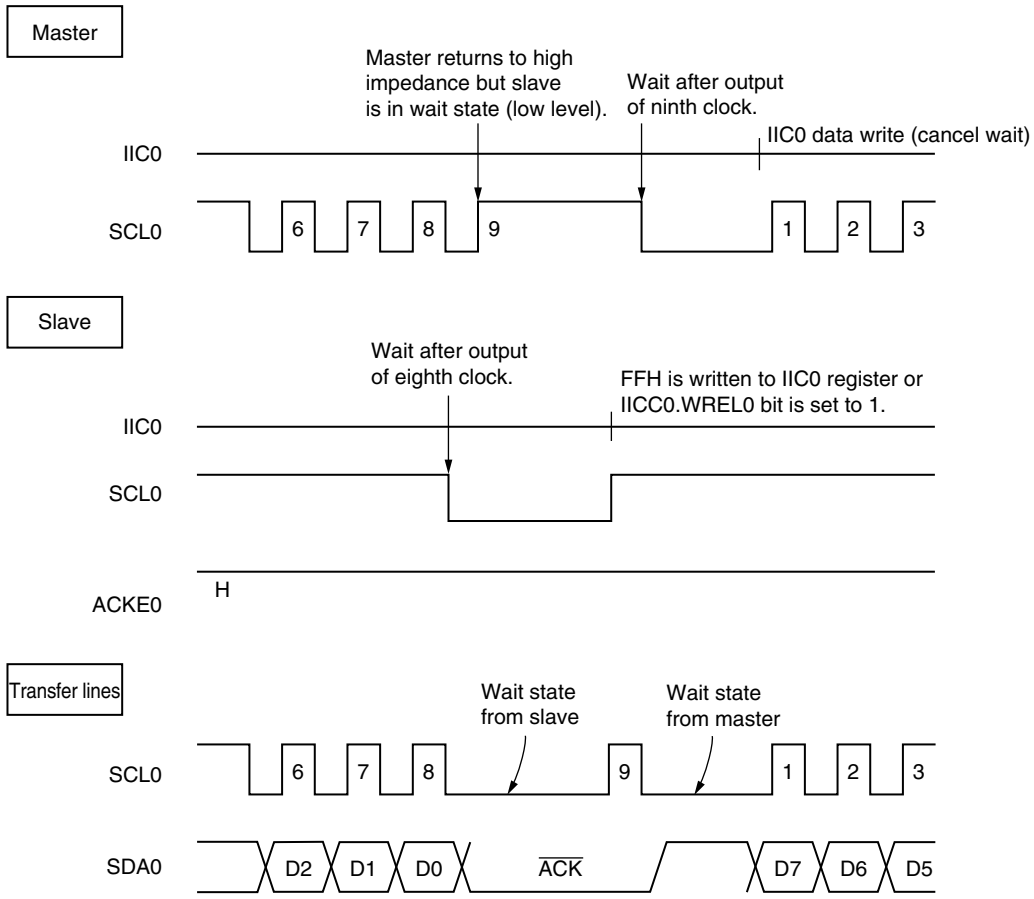
16.5.6 Wait state

The wait state 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 wait status has been canceled for both the master and slave devices, the next data transfer can begin.

Figure 16-10. Wait State (1/2)

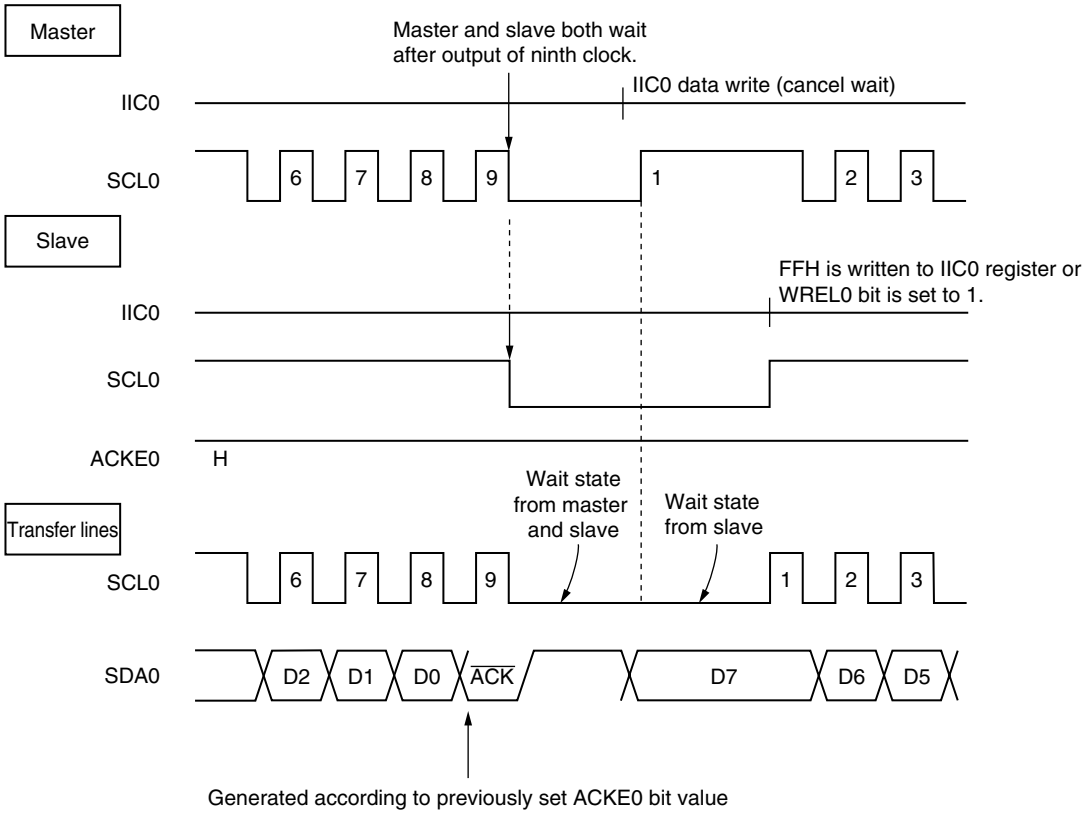
(a) When master device has a nine-clock wait and slave device has an eight-clock wait (master: transmission, slave: reception, and IICC0.ACKE0 bit = 1)



Downloaded from Arrow.com.

Figure 16-10. Wait State (2/2)

(b) When master and slave devices both have a nine-clock wait (master: transmission, slave: reception, and ACKE0 = 1)



A wait state is automatically generated after a start condition is generated. Moreover, a wait state is 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, the wait status is canceled 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

Downloaded from Arrow.com.

<R> 16.5.7 Wait state cancellation method

In the case of I²C0, wait state can be canceled normally in the following ways.

- By writing data to the IIC0 register
- By setting the IICC0.WREL0 bit to 1 (wait state cancellation)
- By setting the IICC0.STT0 bit to 1 (start condition generation)Note
- By setting the IICC0.SPT0 bit to 1 (stop condition generation)Note

Note Master only

If any of these wait state cancellation actions is performed, I²C0 will cancel wait state and restart communication.

When canceling wait state and sending data (including address), write data to the IIC0 register.

To receive data after canceling wait state, or to complete data transmission, set the WREL0 bit to 1.

To generate a restart condition after canceling wait state, set the STT0 bit to 1.

To generate a stop condition after canceling wait state, set the SPT0 bit to 1.

Execute cancellation only once for each wait state.

For example, if data is written to the IIC0 register following wait state cancellation by setting the WREL0 bit to 1, conflict between the SDA0 line change timing and IIC0 register write timing may result in the data output to the SDA0 line may be incorrect.

Even in other operations, if communication is stopped halfway, clearing the IICC0.IICE0 bit to 0 will stop communication, enabling wait state to be cancelled.

If the I²C bus dead-locks due to noise, etc., setting the IICC0.LREL0 bit to 1 causes the communication operation to be exited, enabling wait state to be cancelled.

Downloaded from Arrow.com.

<R> 16.6 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.

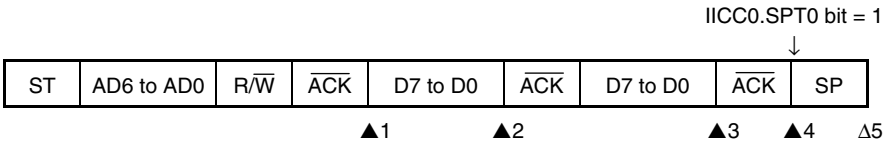
Remark ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

Downloaded from Arrow.com.

16.6.1 Master device operation

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

<1> When IICC0.WTIM0 bit = 0



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X000B

▲3: IICS0 register = 1000X000B (WTIM0 bit = 1[Note])

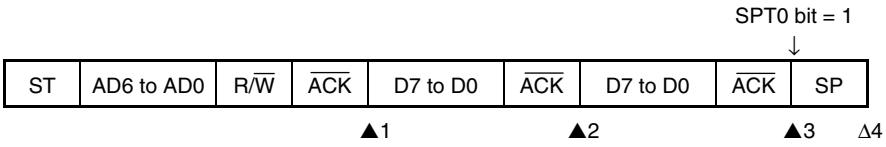
▲4: IICS0 register = 1000XX00B

Δ 5: IICS0 register = 00000001B

Note To generate a stop condition, set the WTIM0 bit to 1 and change the timing of the generation of the interrupt request signal (INTIIC0).

Remark ▲: Always generated
Δ: Generated only when IICC0.SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X100B

▲3: IICS0 register = 1000XX00B

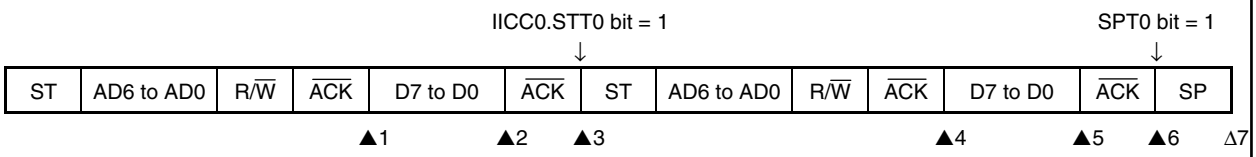
Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

<1> When WTIM0 bit = 0



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X000B (WTIM0 bit = 1[Note 1])

▲3: IICS0 register = 1000XX00B (WTIM0 bit = 0[Note 2])

▲4: IICS0 register = 1000X110B

▲5: IICS0 register = 1000X000B (WTIM0 bit = 1[Note 3])

▲6: IICS0 register = 1000XX00B

Δ 7: IICS0 register = 00000001B

Notes 1. To generate a start condition, set the WTIM0 bit to 1 and change the timing of the generation of the interrupt request signal (INTIIC0).

2. Clear the WTIM0 bit to 0 to make the settings original.

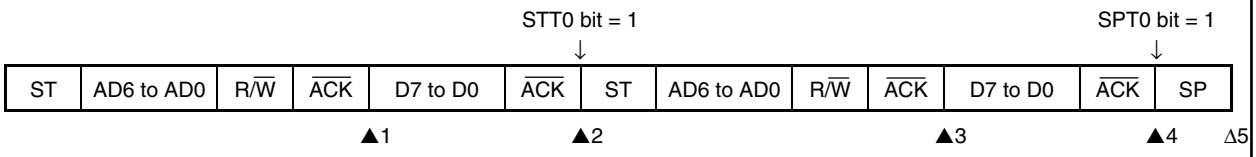
3. To generate a stop condition, set the WTIM0 bit to 1 and change the timing of the generation of the interrupt request signal (INTIIC0).

Remark ▲: Always generated

Δ: Generated only when SPIE0 bit = 1

X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000XX00B

▲3: IICS0 register = 1000X110B

▲4: IICS0 register = 1000XX00B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated

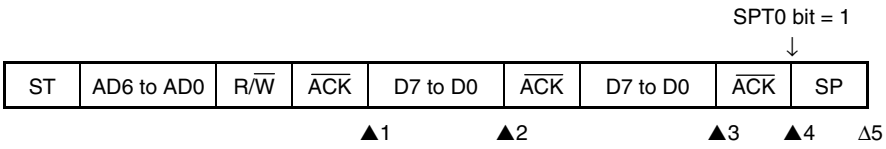
Δ: Generated only when SPIE0 bit = 1

X: don't care

Downloaded from Arrow.com.

(3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

<1> When WTIM0 bit = 0



▲1: IICS0 register = 1010X110B

▲2: IICS0 register = 1010X000B

▲3: IICS0 register = 1010X000B (WTIM0 bit = 1[Note])

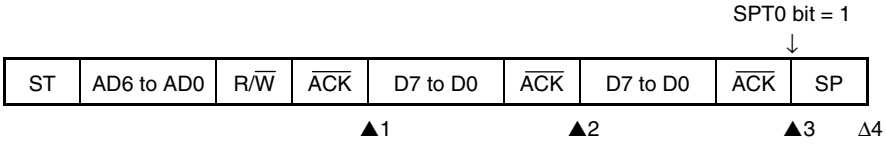
▲4: IICS0 register = 1010XX00B

Δ 5: IICS0 register = 00000001B

Note To generate a stop condition, set the WTIM0 bit to 1 and change the timing of the generation of the interrupt request signal (INTIIC0).

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1010X110B

▲2: IICS0 register = 1010X100B

▲3: IICS0 register = 1010XX00B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

16.6.2 Slave device operation (when receiving slave address data (address match))

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICC0.WTIM0 bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001X000B

▲3: IICS0 register = 0001X000B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when IICC0.SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001X100B

▲3: IICS0 register = 0001XX00B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3	▲4	Δ5

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001X000B

▲3: IICS0 register = 0001X110B

▲4: IICS0 register = 0001X000B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3	▲4	Δ5

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001XX00B

▲3: IICS0 register = 0001X110B

▲4: IICS0 register = 0001XX00B

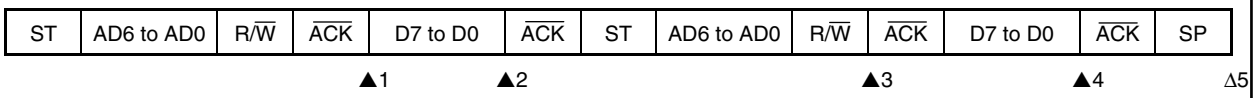
Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address mismatch (= extension code))



▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001X000B

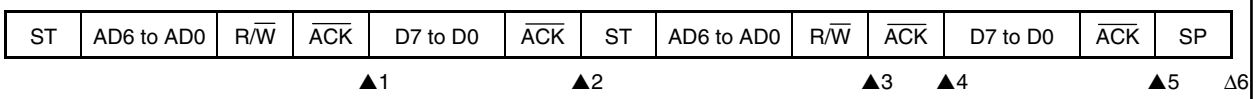
▲3: IICS0 register = 0010X010B

▲4: IICS0 register = 0010X000B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1 (after restart, address mismatch (= extension code))



▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001XX00B

▲3: IICS0 register = 0010X010B

▲4: IICS0 register = 0010X110B

▲5: IICS0 register = 0010XX00B

Δ 6: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3		Δ4

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001X000B

▲3: IICS0 register = 00000110B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3		Δ4

▲1: IICS0 register = 0001X110B

▲2: IICS0 register = 0001XX00B

▲3: IICS0 register = 00000110B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

16.6.3 Slave device operation (when receiving extension code)

Always under communication when receiving the extension code.

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICC0.WTIM0 bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X000B

▲3: IICS0 register = 0010X000B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when IICC0.SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 ▲4 Δ5

▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X110B

▲3: IICS0 register = 0010X100B

▲4: IICS0 register = 0010XX00B

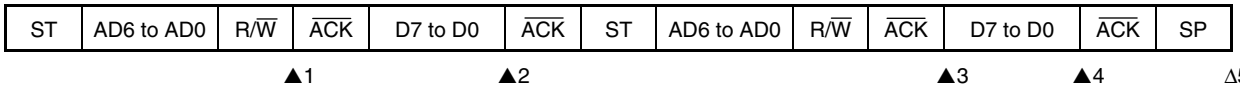
Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address match)



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X000B

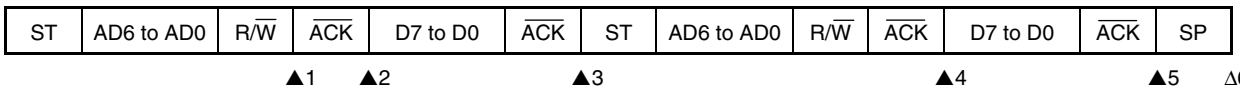
▲3: IICS0 register = 0001X110B

▲4: IICS0 register = 0001X000B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1 (after restart, address match)



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X110B

▲3: IICS0 register = 0010XX00B

▲4: IICS0 register = 0001X110B

▲5: IICS0 register = 0001XX00B

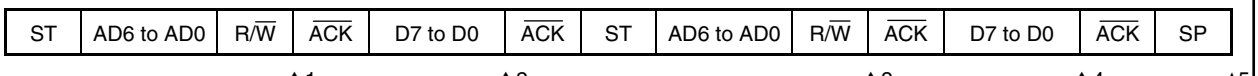
Δ 6: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, extension code reception)



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X000B

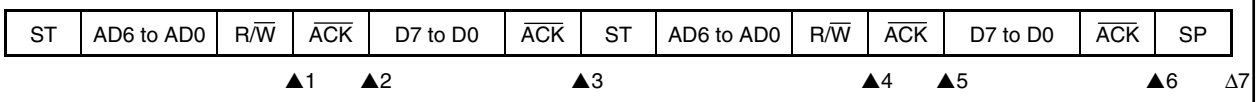
▲3: IICS0 register = 0010X010B

▲4: IICS0 register = 0010X000B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1 (after restart, extension code reception)



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X110B

▲3: IICS0 register = 0010XX00B

▲4: IICS0 register = 0010X010B

▲5: IICS0 register = 0010X110B

▲6: IICS0 register = 0010XX00B

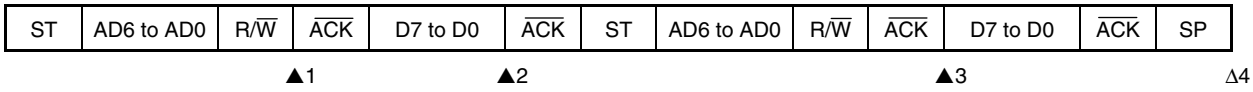
Δ 7: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIM0 bit = 0 (after restart, address mismatch (= not extension code))



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X000B

▲3: IICS0 register = 00000110B

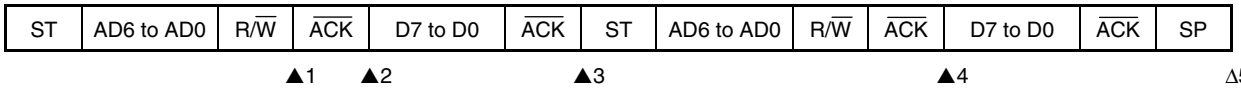
Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated

Δ: Generated only when SPIE0 bit = 1

X: don't care

<2> When WTIM0 bit = 1 (after restart, address mismatch (= not extension code))



▲1: IICS0 register = 0010X010B

▲2: IICS0 register = 0010X110B

▲3: IICS0 register = 0010XX00B

▲4: IICS0 register = 00000110B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated

Δ: Generated only when SPIE0 bit = 1

X: don't care

Downloaded from Arrow.com.

16.6.4 Operation without communication

(1) Start ~ Code ~ Data ~ Data ~ Stop

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

Δ1

Δ 1: IICS0 register = 00000001B

Remark Δ: Generated only when IICC0.SPIE0 bit = 1

Downloaded from Arrow.com.

16.6.5 Arbitration loss operation (operation as slave after arbitration loss)

When used as master in the multi-master system, check the arbitration result by reading the IICS0.MSTS0 bit for checking arbitration result by each INTIIC0 interrupt occurrence.

(1) When arbitration loss occurs during transmission of slave address data

<1> When IICC0.WTIM0 bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICS0 register = 0101X110B

▲2: IICS0 register = 0001X000B

▲3: IICS0 register = 0001X000B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when IICC0.SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICS0 register = 0101X110B

▲2: IICS0 register = 0001X100B

▲3: IICS0 register = 0001XX00B

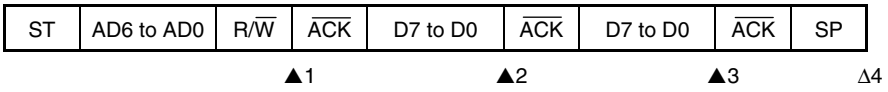
Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(2) When arbitration loss occurs during transmission of extension code

<1> When WTIM0 bit = 0



▲1: IICS0 register = 0110X010B

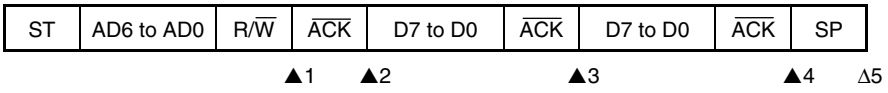
▲2: IICS0 register = 0010X000B

▲3: IICS0 register = 0010X000B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 0110X010B

▲2: IICS0 register = 0010X110B

▲3: IICS0 register = 0010X100B

▲4: IICS0 register = 0010XX00B

Δ 5: IICS0 register = 00000001B

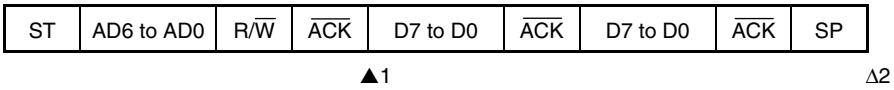
Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

16.6.6 Operation when arbitration loss occurs (no communication after arbitration loss)

When used as master in the multi-master system, check the arbitration result by reading the IICS0.MSTS0 bit for checking arbitration result by each INTIIC0 interrupt occurrence.

(1) When arbitration loss occurs during transmission of slave address data



▲1 Δ2

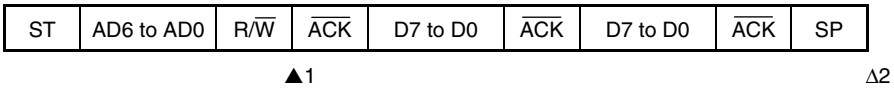
▲1: IICS0 register = 01000110B

Δ 2: IICS0 register = 00000001B

Remark ▲: Always generated

Δ: Generated only when IICC0.SPIE0 bit = 1

(2) When arbitration loss occurs during transmission of extension code



▲1 Δ2

▲1: IICS0 register = 0110X010B

IICC0.LREL0 bit is set to 1 by software

Δ 2: IICS0 register = 00000001B

Remark ▲: Always generated

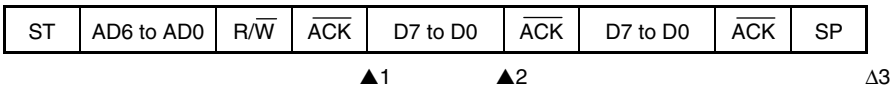
Δ: Generated only when SPIE0 bit = 1

X: don't care

Downloaded from Arrow.com.

(3) When arbitration loss occurs during data transfer

<1> When IICC0.WTIM0 bit = 0



▲1: IICS0 register = 10001110B

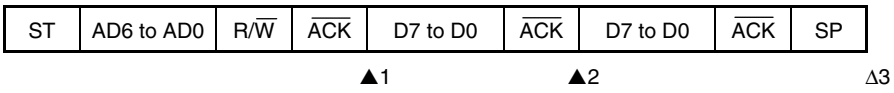
▲2: IICS0 register = 01000000B

Δ 3: IICS0 register = 00000001B

Remark ▲: Always generated

Δ: Generated only when SPIE0 bit = 1

<2> When WTIM0 bit = 1



▲1: IICS0 register = 10001110B

▲2: IICS0 register = 01000100B

Δ 3: IICS0 register = 00000001B

Remark ▲: Always generated

Δ: Generated only when SPIE0 bit = 1

Downloaded from Arrow.com.

(4) When arbitration loss occurs due to restart condition during data transfer

<1> Not extension code (Example: Address mismatch)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to Dn	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
				▲1					▲2		Δ3

▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 01000110B

Δ 3: IICS0 register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care
2. Dn = D6 to D0

<2> Extension code

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to Dn	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
				▲1					▲2		Δ3

▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 0110X010B

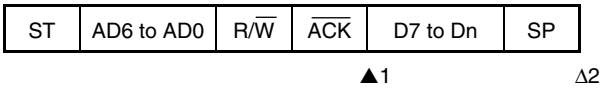
IICC0.LREL0 bit is set to 1 by software

Δ 3: IICS0 register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care
2. Dn = D6 to D0

Downloaded from Arrow.com.

(5) When arbitration loss occurs due to stop condition during data transfer



▲1 Δ2

▲1: IICS0 register = 1000X110B

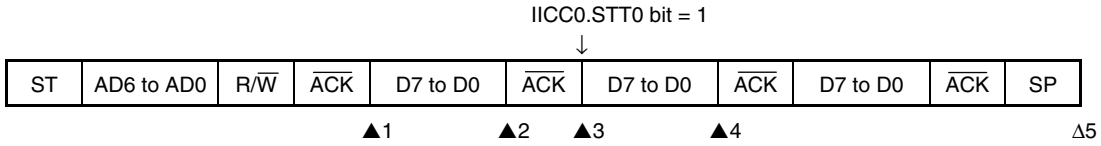
Δ 2: IICS0 register = 01000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care
2. Dn = D6 to D0

Downloaded from Arrow.com.

(6) When arbitration loss occurs due to low level of SDA0 pin when attempting to generate a restart condition

<1> When WTIM0 bit = 0



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X000B (WTIM0 bit = 1)

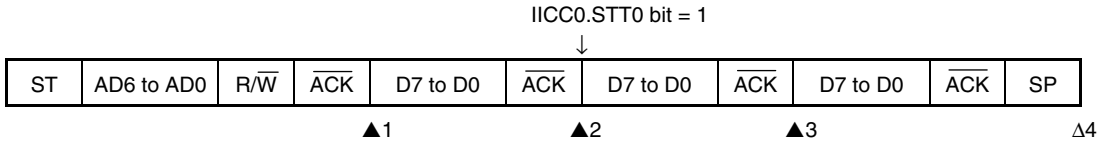
▲3: IICS0 register = 1000X100B (WTIM0 bit = 0)

▲4: IICS0 register = 01000000B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X100B

▲3: IICS0 register = 01000100B

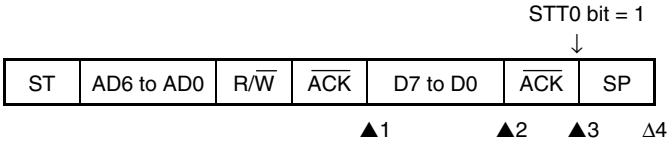
Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(7) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

<1> When WTIM0 bit = 0



▲1: IICS0 register = 1000X110B

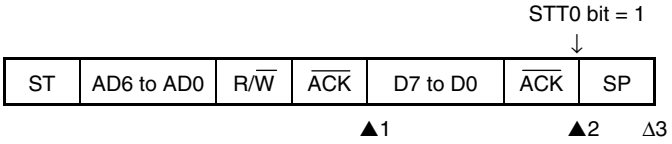
▲2: IICS0 register = 1000X000B (WTIM0 bit = 1)

▲3: IICS0 register = 1000XX00B

Δ 4: IICS0 register = 01000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000XX00B

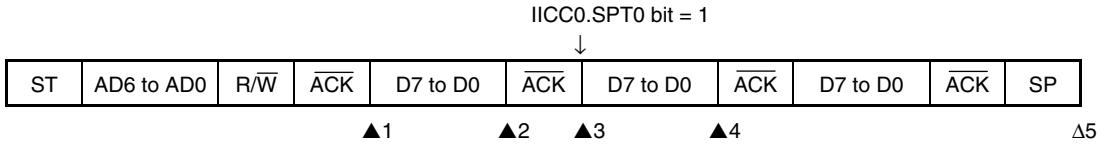
Δ 3: IICS0 register = 01000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

(8) When arbitration loss occurs due to low level of SDA0 pin when attempting to generate a stop condition

<1> When WTIM0 bit = 0



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X000B (WTIM0 bit = 1)

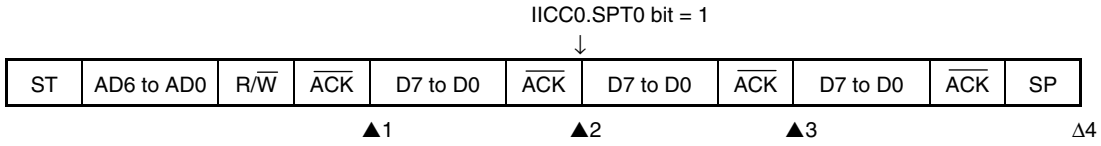
▲3: IICS0 register = 1000X100B (WTIM0 bit = 0)

▲4: IICS0 register = 01000100B

Δ 5: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

<2> When WTIM0 bit = 1



▲1: IICS0 register = 1000X110B

▲2: IICS0 register = 1000X100B

▲3: IICS0 register = 01000100B

Δ 4: IICS0 register = 00000001B

Remark ▲: Always generated
Δ: Generated only when SPIE0 bit = 1
X: don't care

Downloaded from Arrow.com.

16.7 Interrupt Request Signal (INTIIC0) Generation Timing and Wait Control

The setting of the IICC0.WTIM0 bit determines the timing by which the INTIIC0 signal is generated and the corresponding wait control, as shown below.

Table 16-3. INTIIC0 Signal 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 occurs 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, $\overline{ACK}$ is generated 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 extension codes have not been 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 depending on the conditions in Notes 1 and 2 above 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 are determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

Downloaded from Arrow.com.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By writing data to the IIC0 register
- By setting the IICC0.WREL0 bit (canceling wait state)
- By setting the IICC0.STT0 bit (generating start condition)[Note]
- By setting the IICC0.SPT0 bit (generating stop condition)[Note]

Note Master only

When an 8-clock wait has been selected (WTIM0 bit = 0), whether or not $\overline{\text{ACK}}$ has been generated must be determined prior to wait cancellation.

(5) Stop condition detection

The INTIIC0 signal is generated when a stop condition is detected.

16.8 Address Match Detection Method

When in I²C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match detection is performed automatically by hardware. An INTIIC0 interrupt request 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.

16.9 Error Detection

In I²C bus mode, the status of the serial data bus (SDA0) during data transmission is captured by the IIC0 register of the transmitting device, so the IIC0 register data prior to transmission can be compared with the transmitted IIC0 register data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

Downloaded from Arrow.com.

16.10 Extension Code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (EXC0) is set for extension code reception and an interrupt request 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 4 bits of data match: IICS0.EXC0 bit = 1
- 7 bits of data match: IICS0.COI0 bit = 1

(3) Since the processing after the INTIIC0 signal occurs differs according to the data that follows the extension code, <R> such processing is performed by software. The slave that has received an extension code is always under communication, even if the addresses mismatch.
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 16-4. Extension Code Bit Definitions

Slave Address	R/$\overline{W}$ Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	X	CBUS address
0000 010	X	Address that is reserved for different bus format
1111 0xx	X	10-bit slave address specification

Downloaded from Arrow.com.

16.11 Arbitration

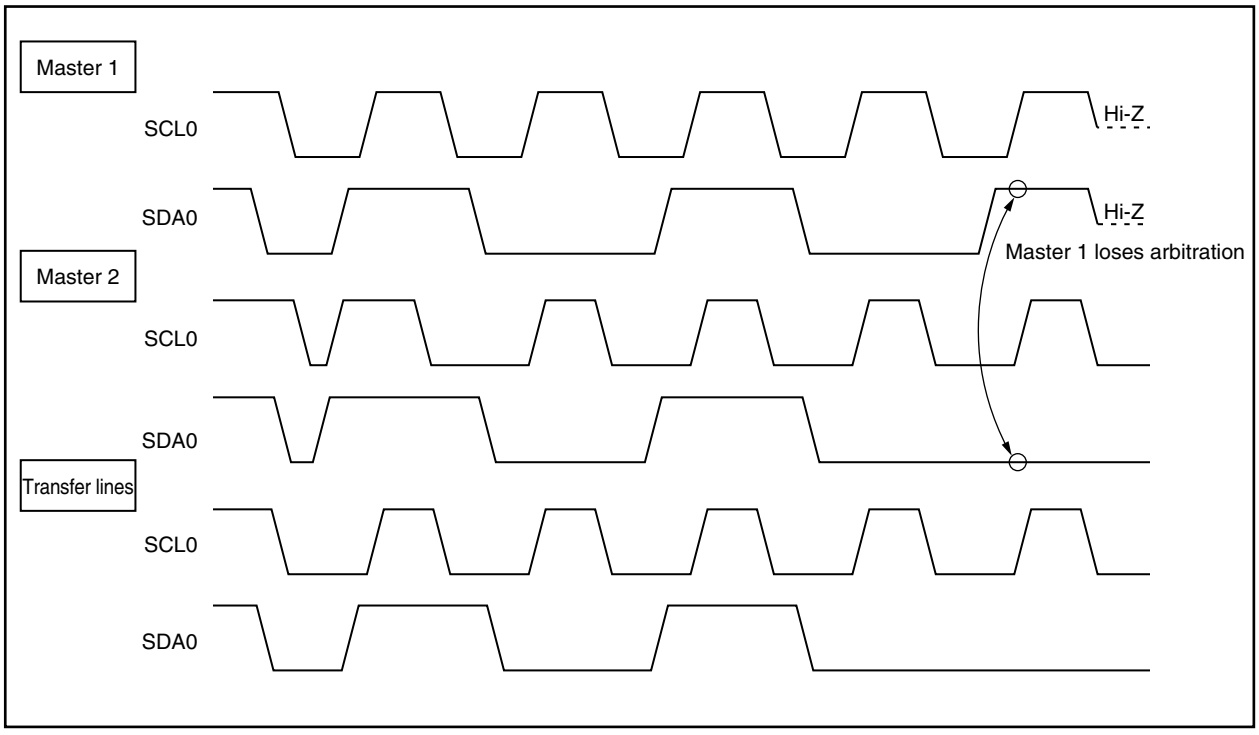
When several master devices simultaneously generate a start condition (when the IICC0.STT0 bit is set to 1 before the IICS0.STD0 bit is set to 1), communication among the master devices is performed as the number of clocks is adjusted until the data differs. This kind of operation is called arbitration.

When one of the master devices loses in arbitration, an arbitration loss flag (IICS0.ALD0 bit) is set (1) via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 lines are both set for high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request signal (INTIIC0) (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALD0 bit = 1 setting that has been made by software.

For details of interrupt request timing, refer to **16.6 I²C Interrupt Request Signals (INTIIC0)**.

Figure 16-11. Arbitration Timing Example



Downloaded from Arrow.com.

Table 16-5. Status During Arbitration and Interrupt Request Generation Timing

Status During Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer Note 1
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During $\overline{\text{ACK}}$ 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 generated (when IICC0.SPIE0 bit = 1) Note 2
When the SDA0 pin is at low level while attempting to generate a restart condition	At falling edge of eighth or ninth clock following byte transfer Note 1
When stop condition is detected while attempting to generate a restart condition	When stop condition is generated (when SPIE0 bit = 1) Note 2
When the SDA0 pin is at low level while attempting to generate a stop condition	At falling edge of eighth or ninth clock following byte transfer Note 1
When the SCL0 pin is at low level while attempting to generate a restart condition	

Notes 1. When the IICC0.WTIM0 bit = 1, an interrupt request 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 occurs at the falling edge of the eighth clock.

2. When there is a possibility that arbitration will occur, set the SPIE0 bit = 1 for master device operation.

16.12 Wakeup Function

The I²C bus slave function is a function that generates an interrupt request signal (INTIIC0) when a local address or extension code has been received.

This function makes processing more efficient by preventing unnecessary interrupt requests from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has generated 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 requests are enabled or disabled.

Downloaded from Arrow.com.

16.13 Communication Reservation

16.13.1 When communication reservation function is enabled (IICF0.IICRSV0 bit = 0)

To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when the IICC0.LREL0 bit was set to “1”).

If the IICC0.STT0 bit is set (1) while the bus is not used, a start condition is automatically generated and wait status is set after the bus is released (after a stop condition is detected).

<R> A communication is automatically started as the master by setting the IICC0.SPIE0 bit to 1, detecting the bus release due to an interrupt request (INTIIC0) occurrence (detecting a stop condition), and then writing the address to the IIC0 register. Before detecting a stop condition, data written to the IIC0 register is set to invalid.

When the STT0 bit has been set (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 (1), wait for the wait period, then check the IICS0.MSTS0 bit.

Wait periods, which should be set via software, are listed in Table 16-6. These wait periods can be set via the settings for the IICX0.CLX0, IICCL0.SMC0, IICCL0.CL01, and IICCL0.CL00 bits.

<R> **Table 16-6. Wait Periods**

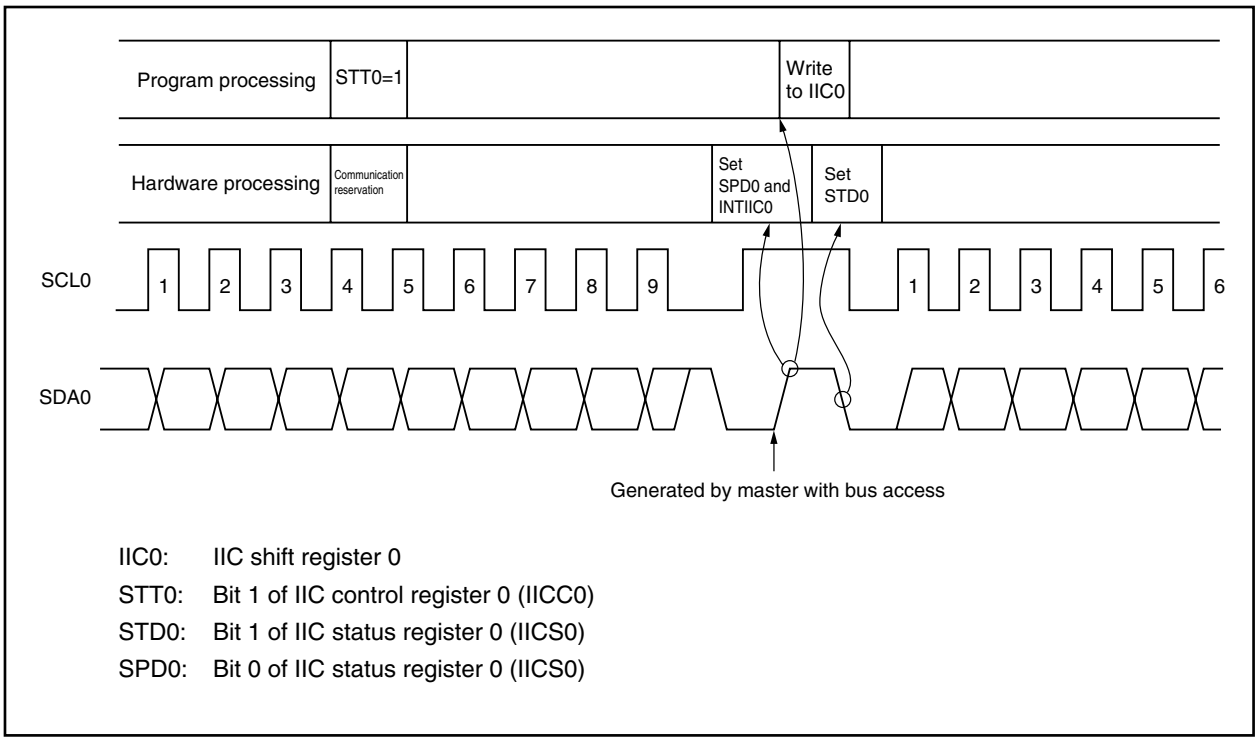
CLX0	SMC0	CL01	CL00	Selected Clock	Wait Period
0	0	0	0	$f_{XX}/2$	46 clocks
0	0	0	1	$f_{XX}/2$	86 clocks
0	0	1	0	f_{XX}	43 clocks
0	0	1	1	$f_{XX}/3$	102 clocks
0	1	0	1/0	$f_{XX}/2$	30 clocks
0	1	1	0	f_{XX}	15 clocks
0	1	1	1	$f_{XX}/3$	36 clocks
1	1	0	1/0	$f_{XX}/2$	18 clocks
1	1	1	0	f_{XX}	9 clocks

Downloaded from Arrow.com.

The communication reservation timing is shown below.

<R>

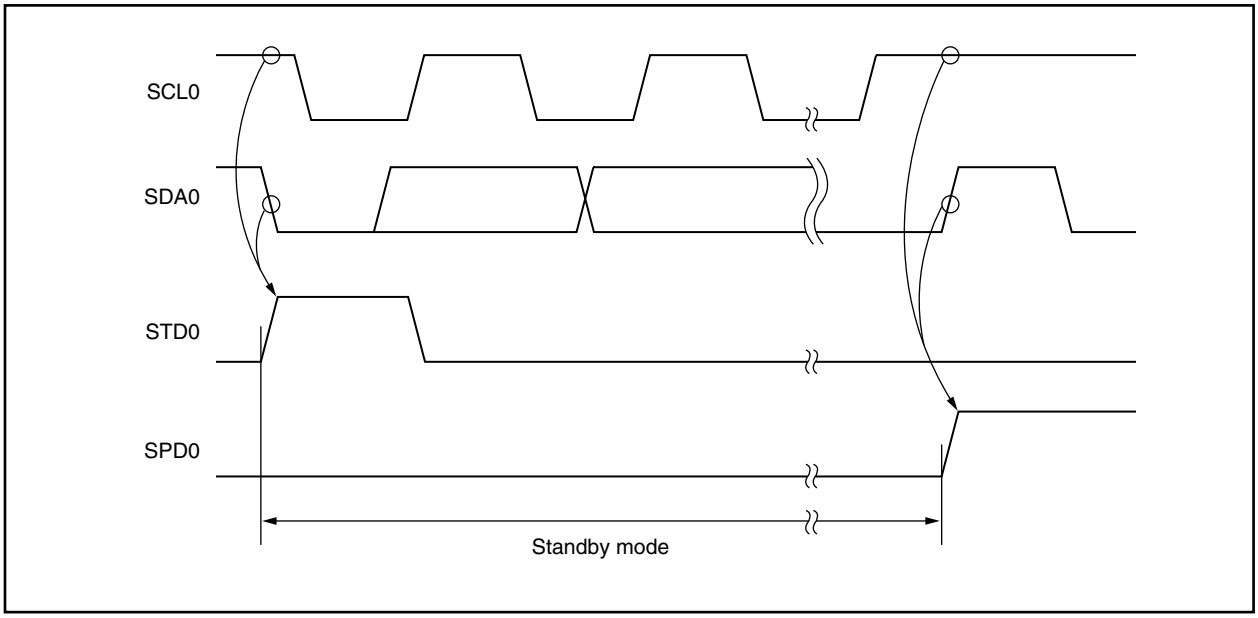
Figure 16-12. Communication Reservation Timing



IIC0: IIC shift register 0
STT0: Bit 1 of IIC control register 0 (IICC0)
STD0: Bit 1 of IIC status register 0 (IICS0)
SPD0: Bit 0 of IIC status register 0 (IICS0)

Communication reservations are accepted via the following timing. After the IICS0.STD0 bit is set to 1, a communication reservation can be made by setting the IICC0.STT0 bit to 1 before a stop condition is detected.

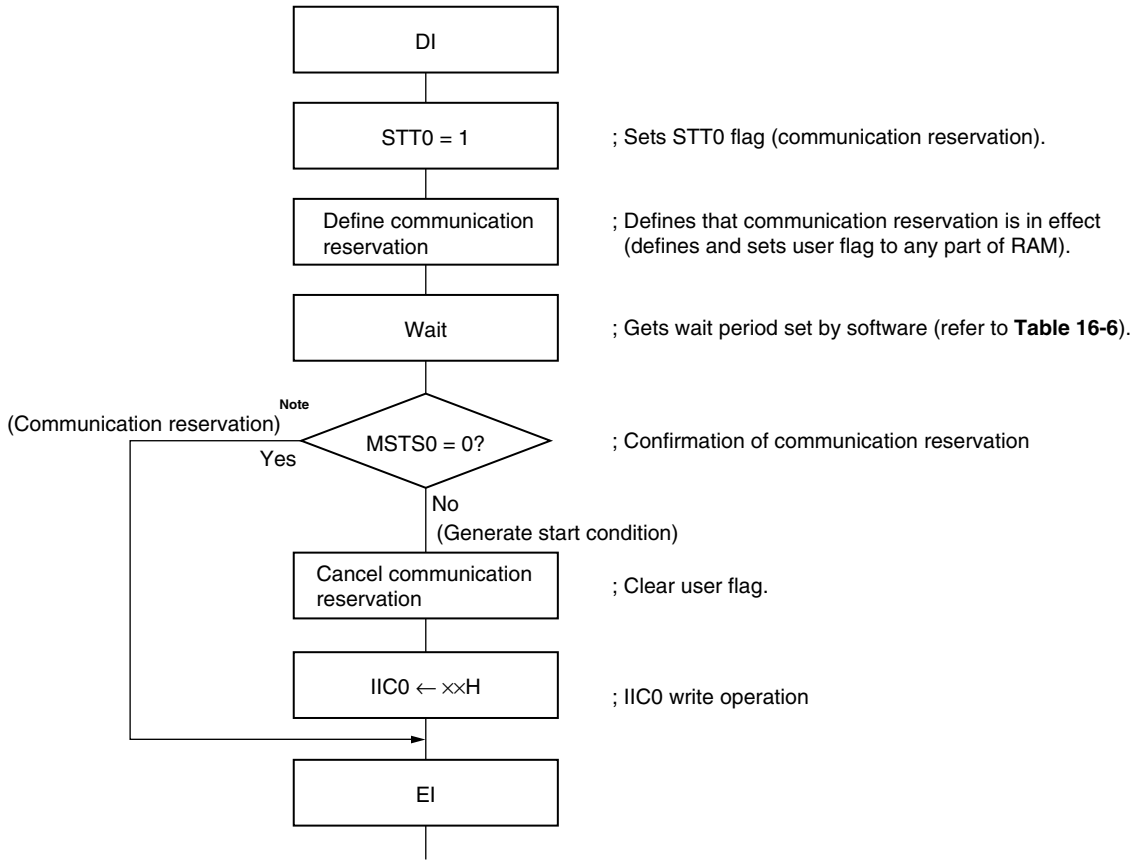
Figure 16-13. Timing for Accepting Communication Reservations



Downloaded from Arrow.com.

The communication reservation flowchart is illustrated below.

Figure 16-14. Communication Reservation Flowchart



Note The communication reservation operation executes a write to the IIC0 register when a stop condition interrupt request occurs.

Downloaded from Arrow.com.

16.13.2 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. The following two statuses are included in the status where bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ 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 16-7 is required until the STCF0 flag is set after setting the STT0 bit = 1. Therefore, secure the time by software.

<R>

Table 16-7. Wait Periods

CL01	CL00	Selected Clock	Wait Period
0	0	$f_{XX}/2$	6 clocks
0	1	$f_{XX}/2$	6 clocks
1	0	f_{XX}	3 clocks
1	1	$f_{XX}/3$	9 clocks

Downloaded from Arrow.com.

16.14 Cautions

(1) When IICF0.STCEN0 bit = 0

Immediately after 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 generate 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.

<R> (3) When the IICC0.IICE0 bit of the V850ES/KE1 is set to 1 while communications with other devices are in progress, the start condition may be detected depending on the status of the communication line. Be sure to set the IICC0.IICE0 bit to 1 when the SCL0 and SDA0 lines are high level.

<R> (4) Determine the operation clock frequency by the IICCL0 and IICX0 registers before enabling the operation (IICC0.IICE0 bit = 1). To change the operation clock frequency, clear the IICC0.IICE0 bit to 0 once.

<R> (5) After the IICC0.STT0 and IICC0.SPT0 bits have been set to 1, they must not be re-set without being cleared to 0 first.

<R> (6) If transmission has been reserved, set the IICC0.SPIE0 bit to 1 so that an interrupt request is generated by the detection of a stop condition. After an interrupt request has been generated, the wait state will be released by writing communication data to I²C0, then transferring will begin. If an interrupt is not generated by the detection of a stop condition, transmission will halt in the wait state because an interrupt request was not generated. However, it is not necessary to set the SPIE0 bit to 1 for the software to detect the IICS0.MSTS0 bit.

Downloaded from Arrow.com.

<R> ## 16.15 Communication Operations

The following shows three operation procedures with the flowchart.

(1) Master operation in single master system

The flowchart when using the V850ES/KE1 as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

(2) Master operation in multimaster system

In the I²C0 bus multimaster system, whether the bus is released or used cannot be judged by the I²C bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the V850ES/KE1 takes part in a communication with bus released state.

This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the V850ES/KE1 loses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

(3) Slave operation

An example of when the V850ES/KE1 is used as the slave is shown below.

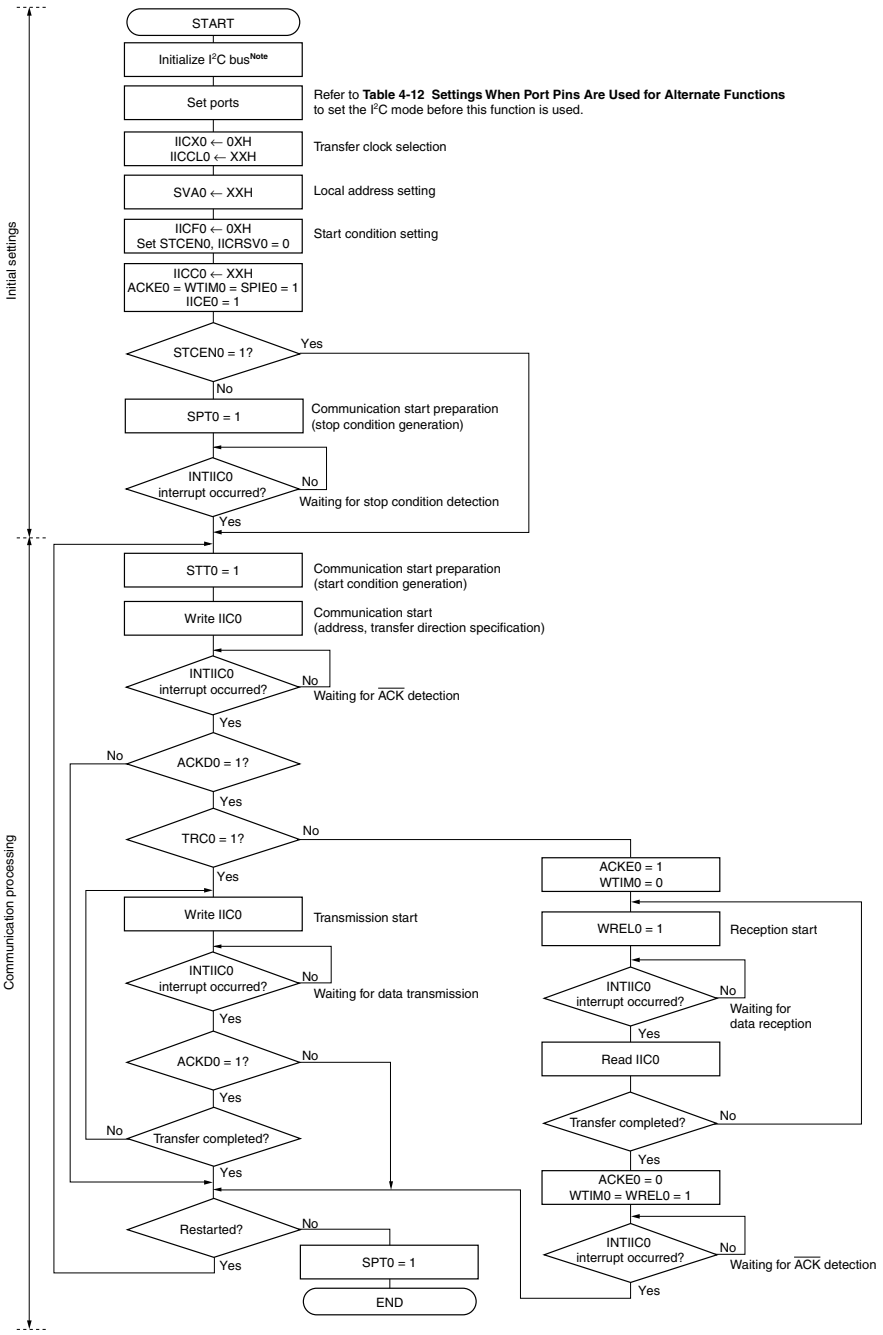
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIIC0 interrupt occurrence (communication waiting). When the INTIIC0 interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

Downloaded from Arrow.com.

16.15.1 Master operation in single master system

Figure 16-15. Master Operation in Single Master System



Note Release the I²C0 bus (SCL0, SDA0 pins = high level) in conformity with the specifications of the product in communication.

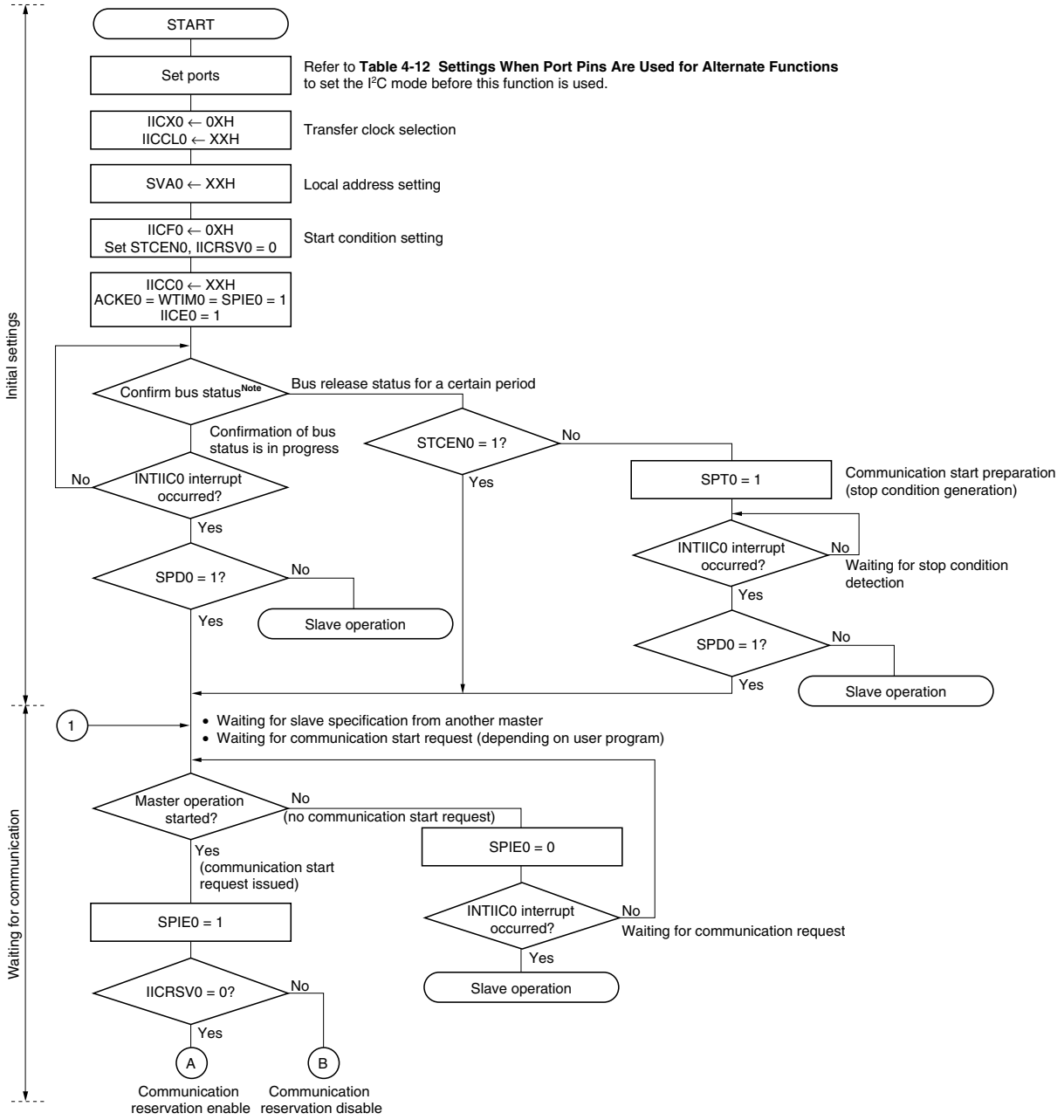
For example, when the EEPROM™ outputs a low level to the SDA0 pin, set the SCL0 pin to the output port and output clock pulses from that output port until when the SDA0 pin is constantly high level.

Remark For the transmission and reception formats, conform to the specifications of the product in communication.

Downloaded from Arrow.com.

16.15.2 Master operation in multimaster system

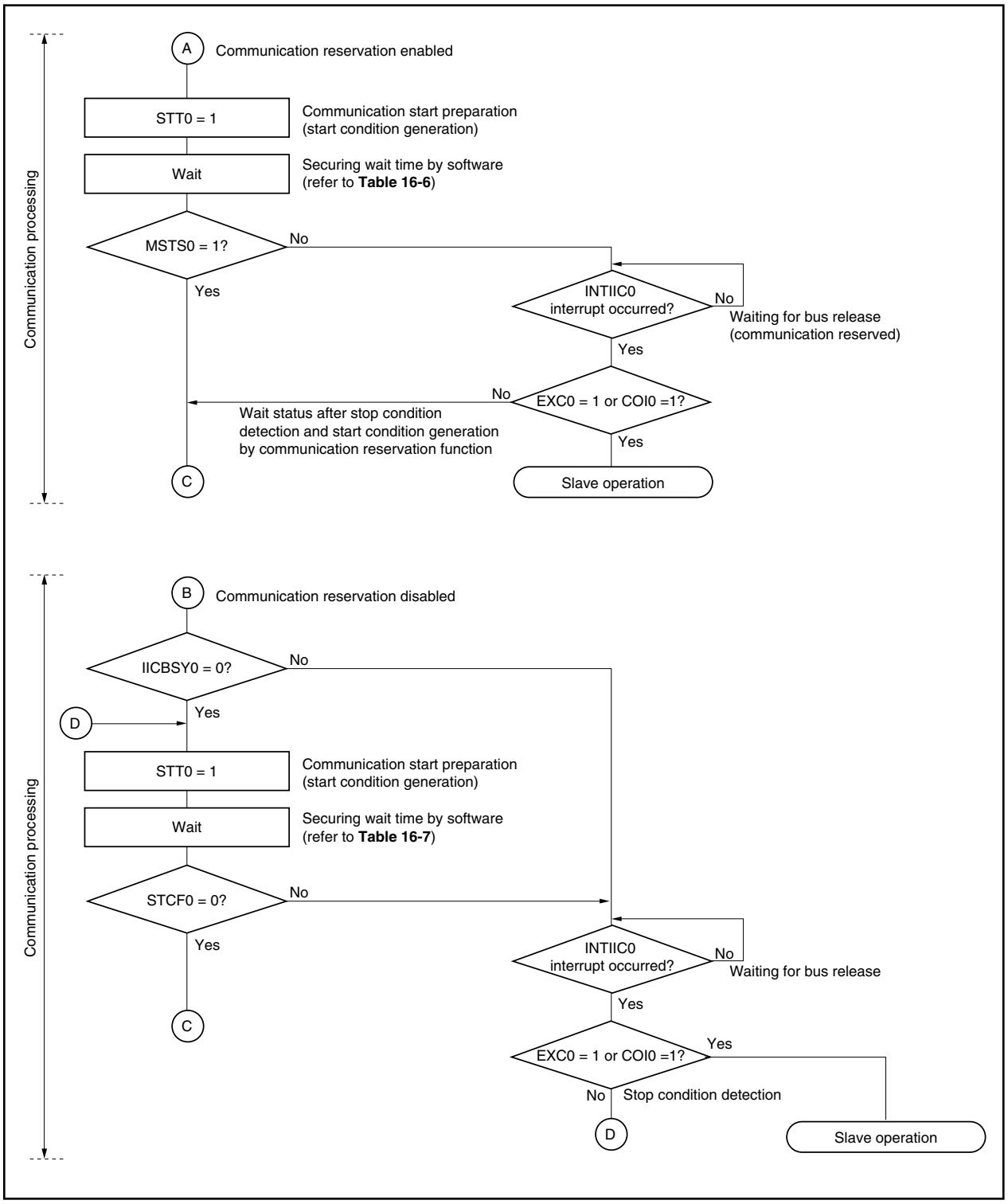
Figure 16-16. Master Operation in Multimaster System (1/3)



Note Confirm that the bus release status (IICCL0.CLD0 bit = 1, IICCL0.DAD0 bit = 1) has been maintained for a certain period (1 frame, for example). When the SDA0 pin is constantly low level, determine whether to release the I²C0 bus (SCL0, SDA0 pins = high level) by referring to the specifications of the product in communication.

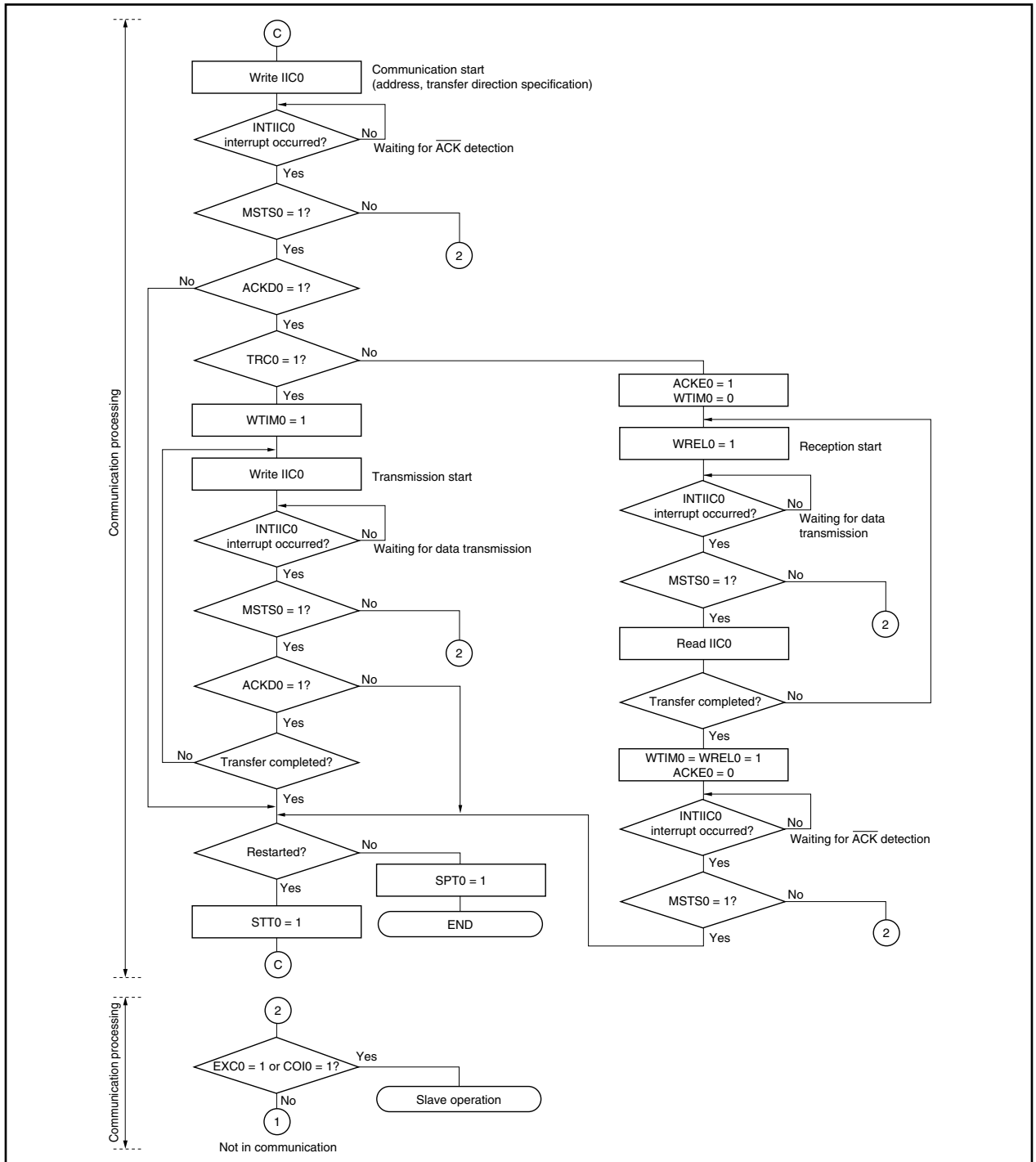
Downloaded from Arrow.com.

Figure 16-16. Master Operation in Multimaster System (2/3)



Downloaded from Arrow.com.

Figure 16-16. Master Operation in Multimaster System (3/3)



Remarks 1. Conform the transmission and reception formats to the specifications of the product in communication.

2. When using the V850ES/KE1 as the master in the multimaster system, read the IICS0.MSTS0 bit for each INTIIC0 interrupt occurrence to confirm the arbitration result.

3. When using the V850ES/KE1 as the slave in the multimaster system, confirm the status using the IICS0 and IICF0 registers for each INTIIC0 interrupt occurrence to determine the next processing.

Downloaded from Arrow.com.

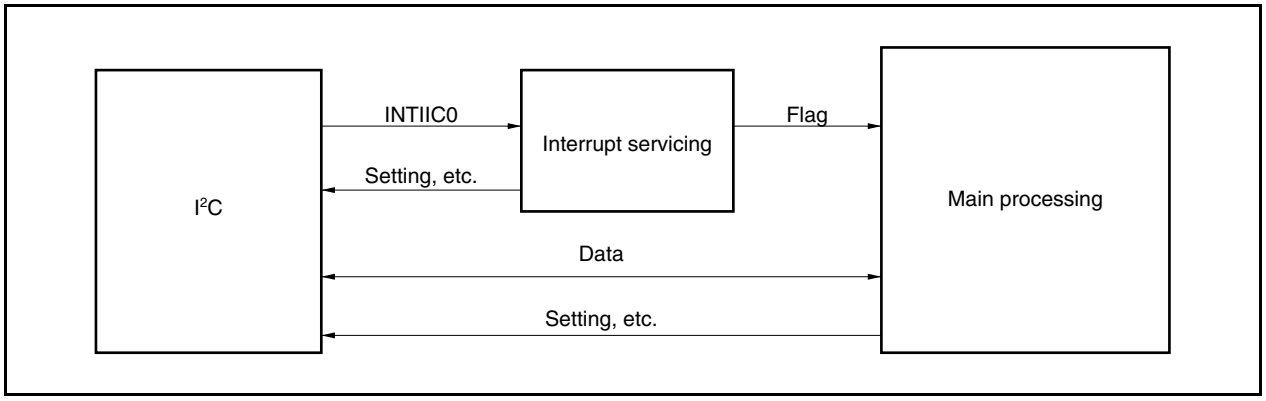
16.15.3 Slave operation

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 16-17. 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 the 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, $\overline{\text{ACK}}$ from master not detected, address mismatch)

(2) Ready flag

This flag indicates that data communication is enabled. This is the same status as an INTIIC0 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 clearance 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 the 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).

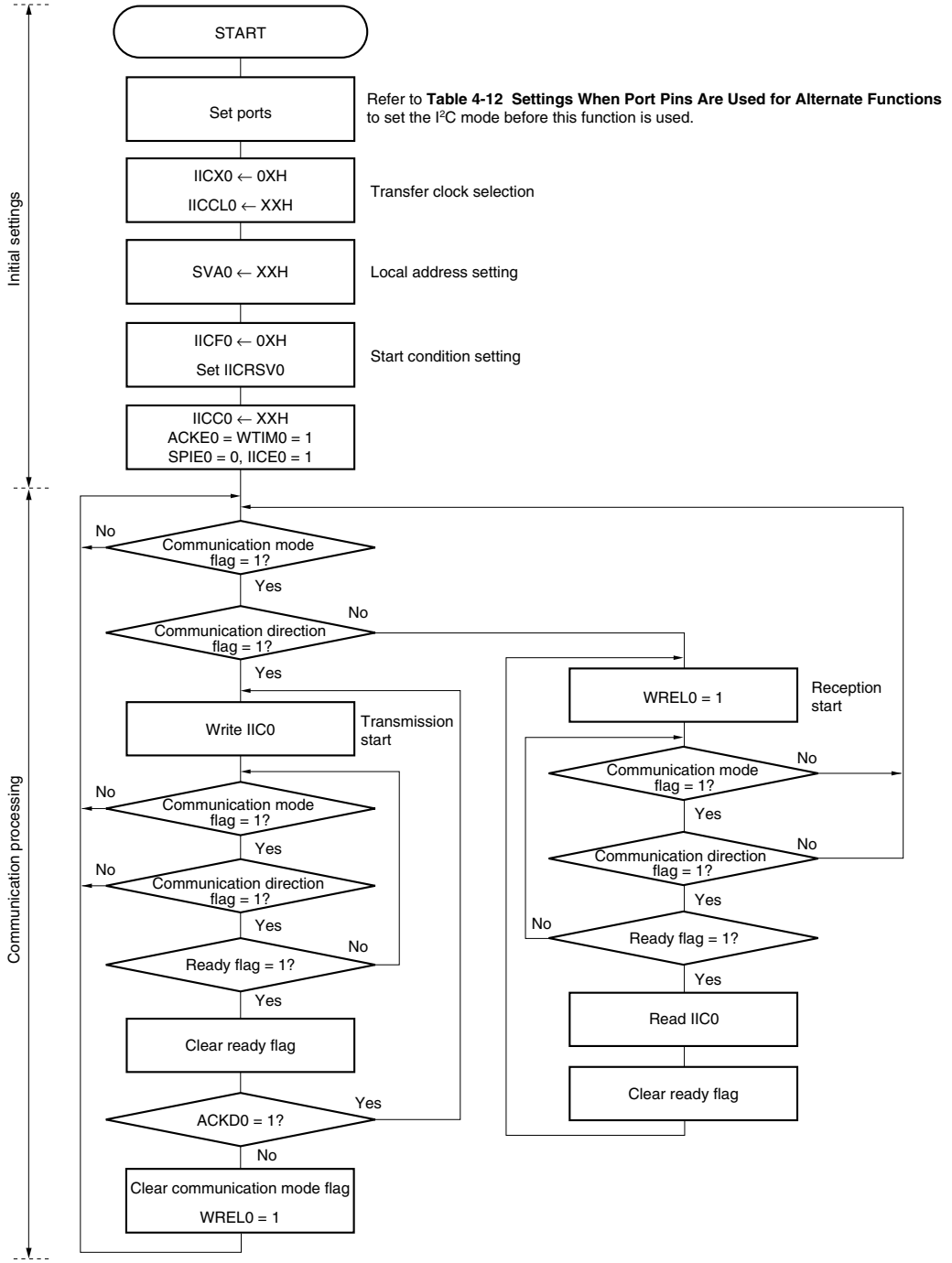
For transmission, repeat the transmission operation until the master device stops returning $\overline{\text{ACK}}$. When the master device stops returning $\overline{\text{ACK}}$, transfer is complete.

Downloaded from Arrow.com.

For reception, receive the required number of data and do not return $\overline{ACK}$ 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.

<R>

Figure 16-18. Slave Operation Flowchart (1)



Remark Transmission and reception formats are compliant with the specifications of the product in communication.

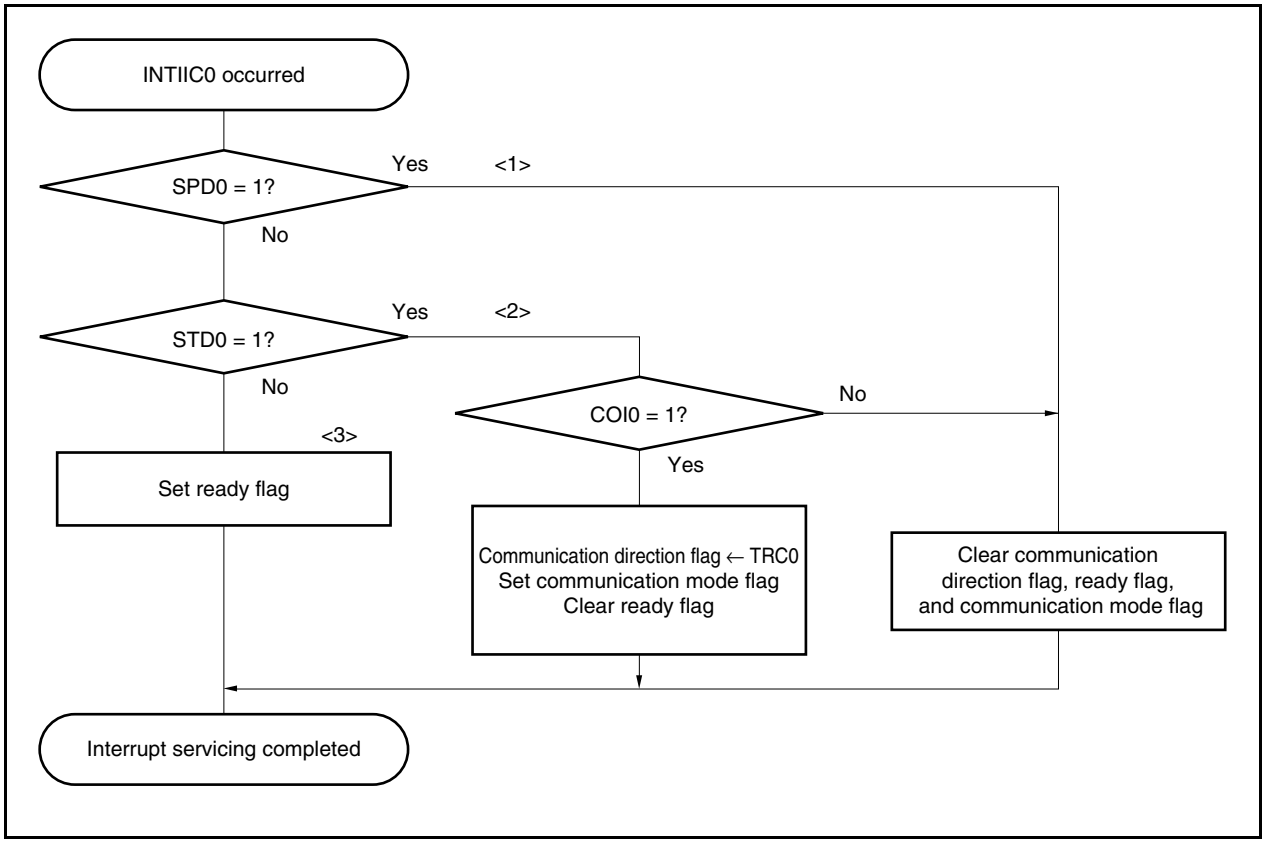
Downloaded from Arrow.com.

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 I²C0 bus remains in the wait status.

Remark <1> to <3> in the above correspond to <1> to <3> in **Figure 16-19 Slave Operation Flowchart (2)**.

<R> **Figure 16-19. Slave Operation Flowchart (2)**



Downloaded from Arrow.com.

16.16 Timing of Data Communication

When using I²C bus mode, the master device generates 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 that specifies the data transfer direction and then starts serial communication with the slave device.

The IIC0 register's shift operation is synchronized with the falling edge of the serial clock (SCL0 pin). The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0 pin.

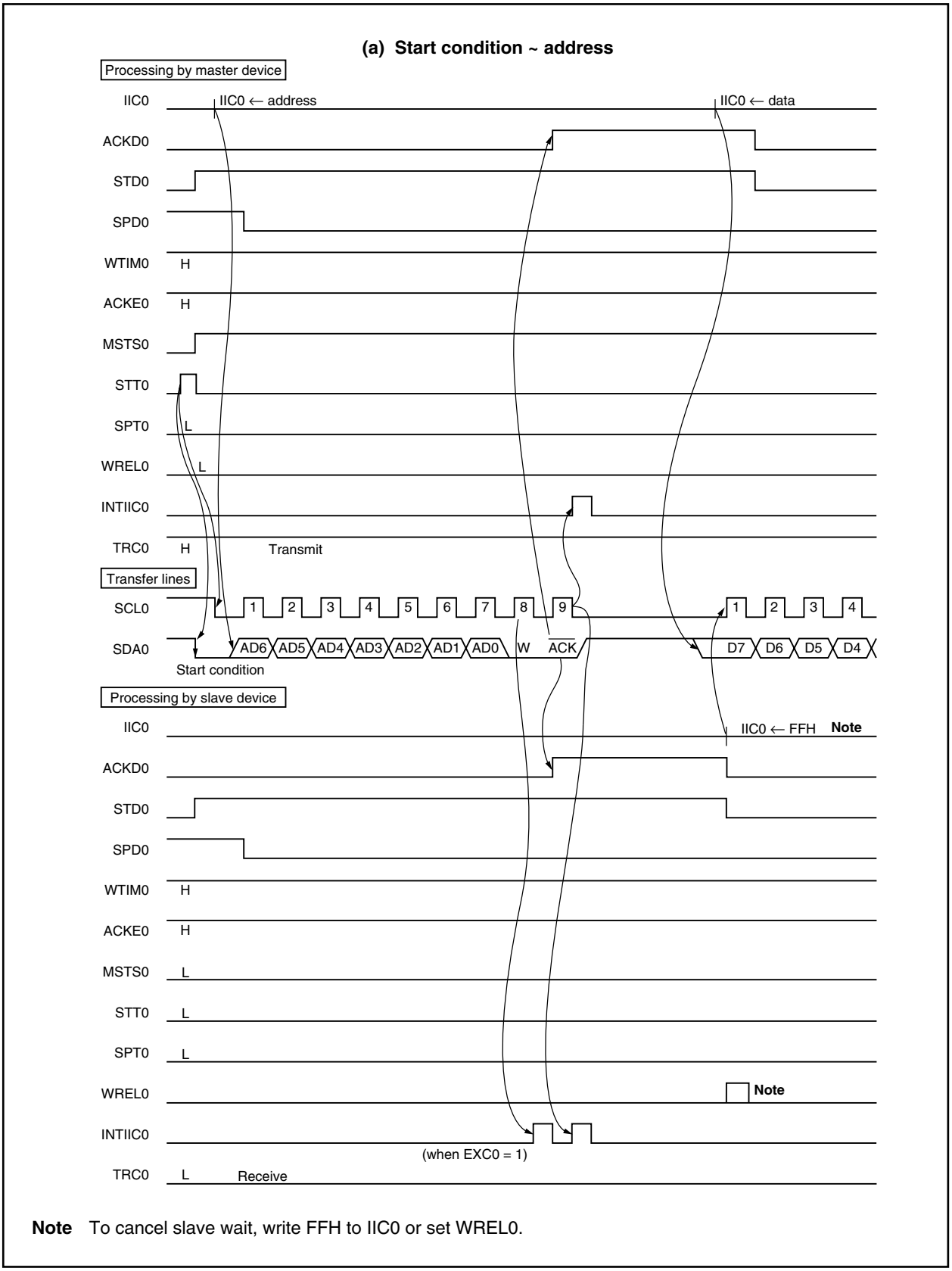
Data input via the SDA0 pin is captured by the IIC0 register at the rising edge of the SCL0 pin.

The data communication timing is shown below.

Downloaded from Arrow.com.

<R>

Figure 16-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)

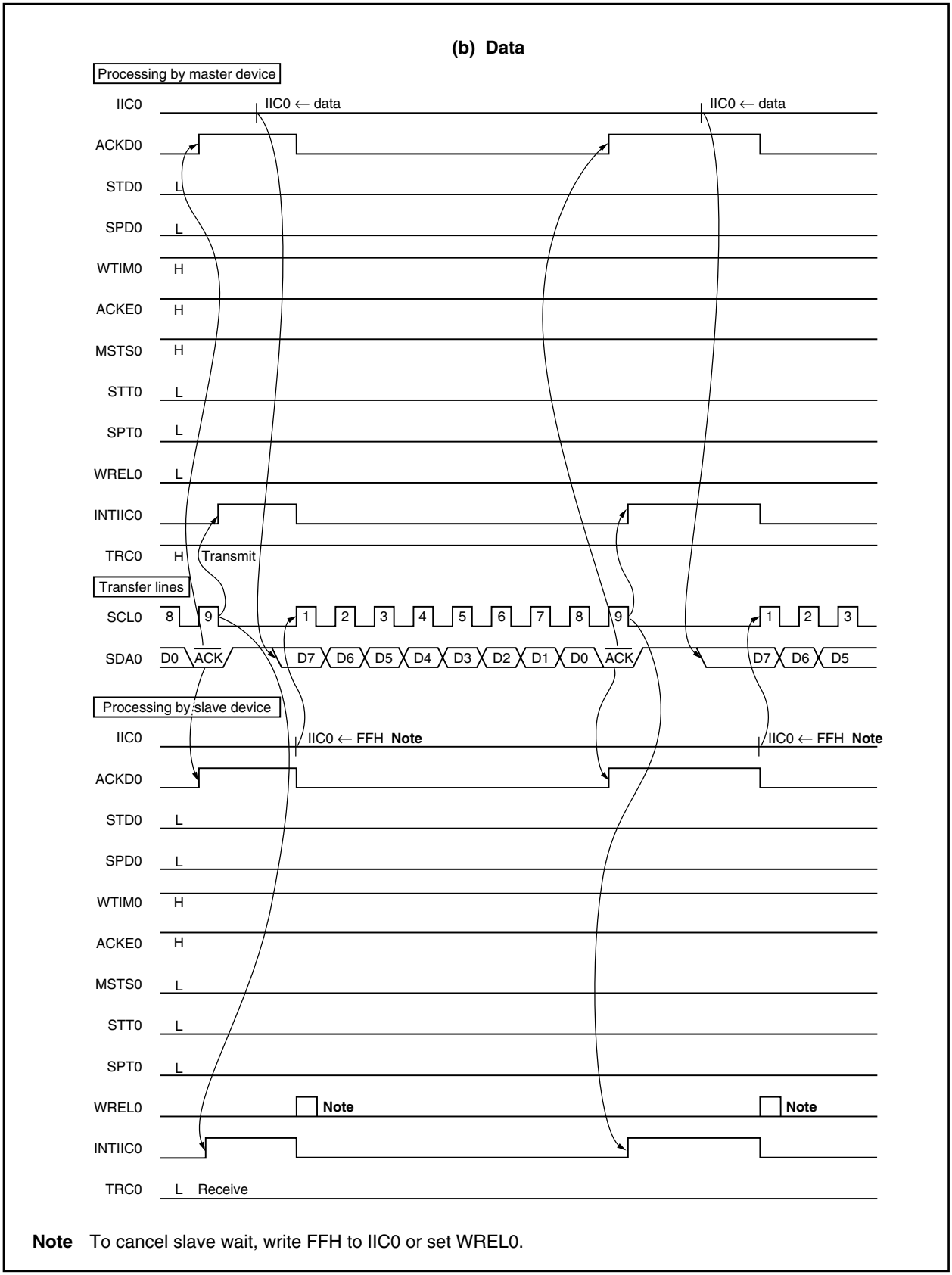


Note To cancel slave wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

<R>

Figure 16-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)



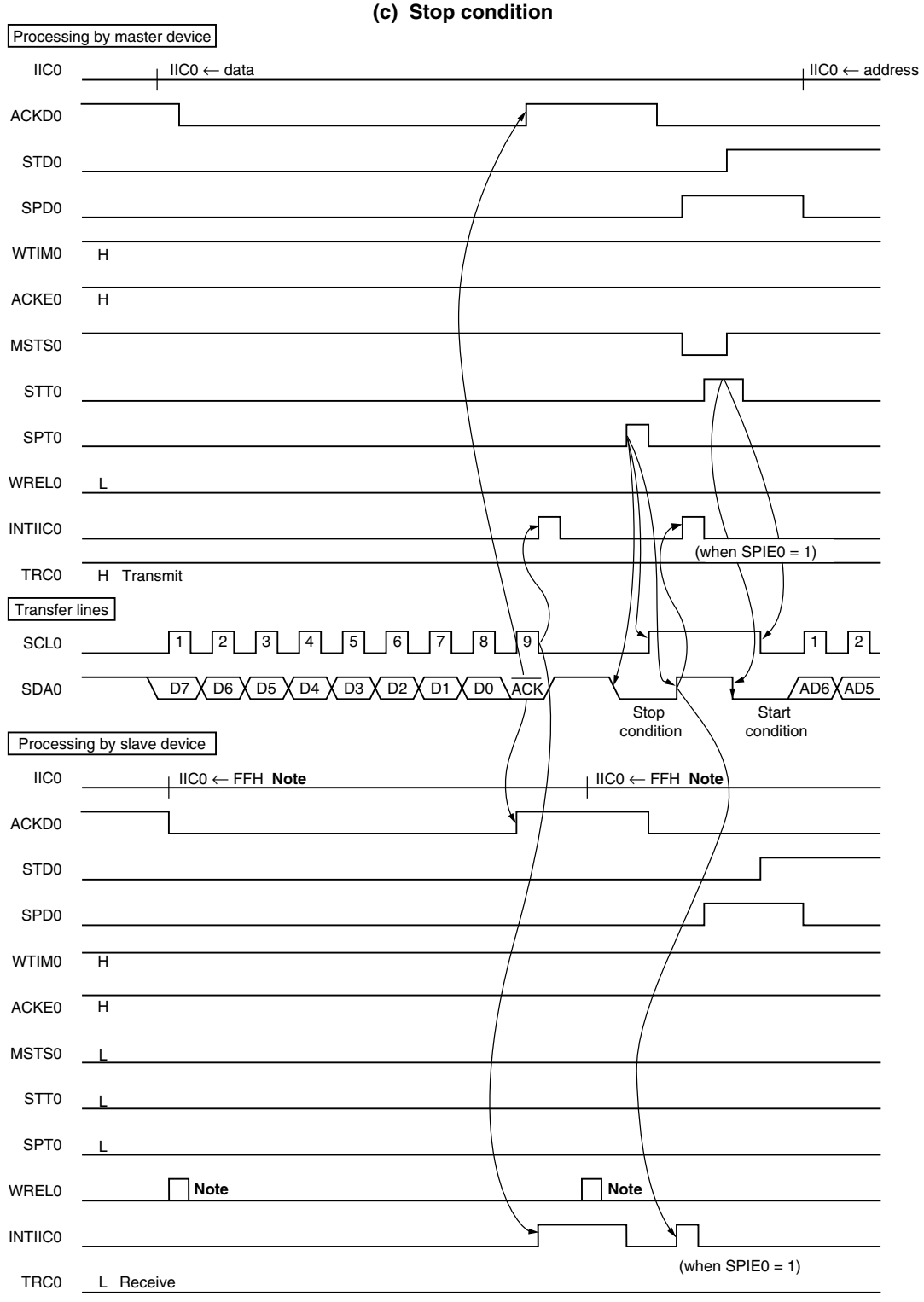
Note To cancel slave wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

<R>

Figure 16-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)

(c) Stop condition

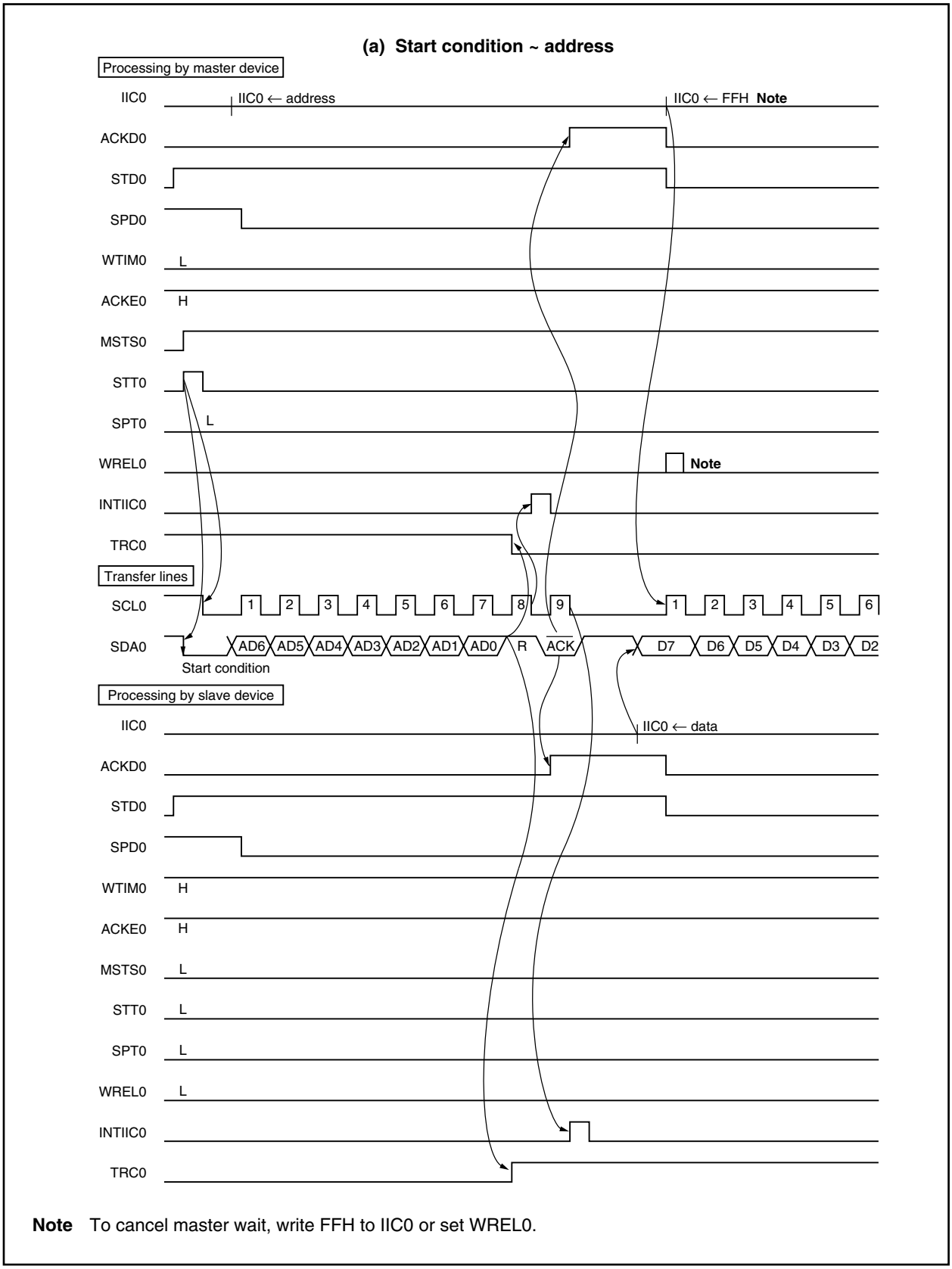


Note To cancel slave wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

<R>

Figure 16-21. Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (1/3)

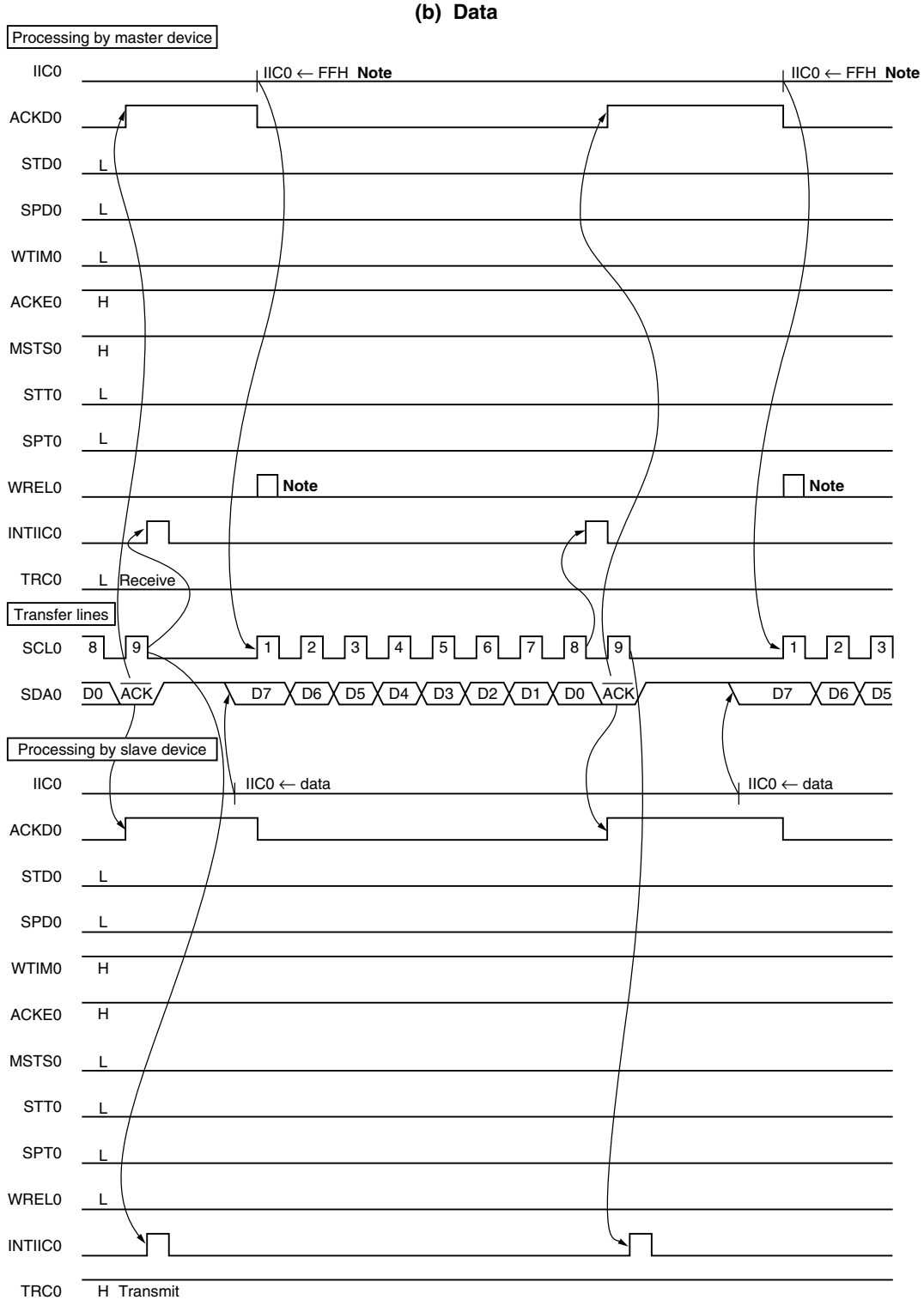


Note To cancel master wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

<R> **Figure 16-21. Example of Slave to Master Communication (When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (2/3)**

(b) Data

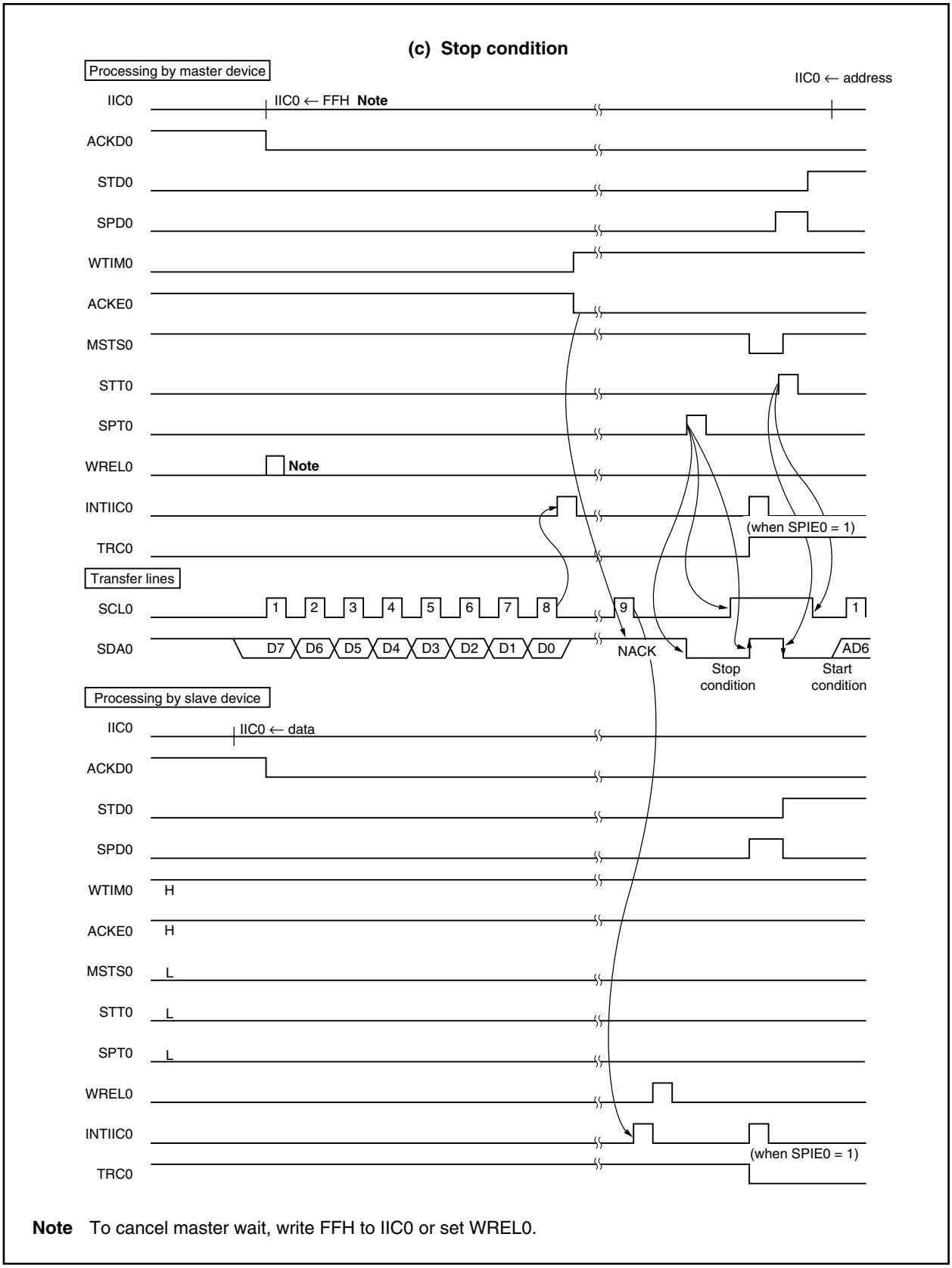


Note To cancel master wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

<R>

Figure 16-21. Example of Slave to Master Communication
(When 8-Clock → 9-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (3/3)



Note To cancel master wait, write FFH to IIC0 or set WREL0.

Downloaded from Arrow.com.

CHAPTER 17 INTERRUPT/EXCEPTION PROCESSING FUNCTION

17.1 Overview

The V850ES/KE1 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and realize an interrupt function that can service interrupt requests from a total of 33 or 34 sources.

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

The V850ES/KE1 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 (fetching of an illegal opcode) (exception trap).

17.1.1 Features

Interrupt Source				V850ES/KE1
Interrupt function	Non-maskable interrupt	External		1 channel (NMI pin)
		Internal		2 channels (WDT1, WDT2)
	Maskable interrupt	External		7 channels (all edge detection interrupts)
		Internal	WDT1	1 channel
			TMP	3 channels
			TM0	2 channels
			TMH	2 channels
			TM5	2 channels
			WT	2 channels
			BRG	1 channel
			UART	6 channels
			CSI0	2 channels
			IIC[Note]	1 channel
			KR	1 channel
			AD	1 channel
			Total	24 channels
Exception function	Software exception			16 channels (TRAP00H to TRAP0FH)
				16 channels (TRAP10H to TRAP1FH)
	Exception trap			2 channels (ILGOP/DBG0)

Note Only in the μPD703207Y, 70F3207HY

Table 17-1 lists the interrupt/exception sources.

Downloaded from Arrow.com.

Table 17-1. Interrupt Source List (1/2)

Type	Classification	Default Priority	Name	Trigger	Interrupt Source	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Reset	Interrupt	–	RESET	$\overline{\text{RESET}}$ pin input	Pin	0000H	00000000H	Undefined	–
				Internal reset input from WDT1, WDT2	WDT1 WDT2				
Non-maskable	Interrupt	–	NMI	NMI pin valid edge input	Pin	0010H	00000010H	nextPC	–
		–	INTWDT1	WDT1 overflow (when non-maskable interrupt selected)	WDT1	0020H	00000020H	**Note 1**	–
		–	INTWDT2	WDT2 overflow (when non-maskable interrupt selected)	WDT2	0030H	00000030H	**Note 1**	–
Software exception	Exception	–	TRAP0n[Note 2]	TRAP instruction	–	004nH[Note 2]	00000040H	nextPC	–
		–	TRAP1n[Note 2]	TRAP instruction	–	005nH[Note 2]	00000050H	nextPC	–
Exception trap	Exception	–	ILGOP/ DBG0	Illegal opcode/DBTRAP instruction	–	0060H	00000060H	nextPC	–
Maskable	Interrupt	0	INTWDTM1	WDT1 overflow (when interval timer selected)	WDT1	0080H	00000080H	nextPC	WDT1IC
		1	INTP0	INTP0 pin valid edge input	Pin	0090H	00000090H	nextPC	PIC0
		2	INTP1	INTP1 pin valid edge input	Pin	00A0H	000000A0H	nextPC	PIC1
		3	INTP2	INTP2 pin valid edge input	Pin	00B0H	000000B0H	nextPC	PIC2
		4	INTP3	INTP3 pin valid edge input	Pin	00C0H	000000C0H	nextPC	PIC3
		5	INTP4	INTP4 pin valid edge input	Pin	00D0H	000000D0H	nextPC	PIC4
		6	INTP5	INTP5 pin valid edge input	Pin	00E0H	000000E0H	nextPC	PIC5
		7	INTP6	INTP6 pin valid edge input	Pin	00F0H	000000F0H	nextPC	PIC6
		10	INTTM010	TM01 and CR010 match	TM01	0120H	00000120H	nextPC	TM0IC10
		11	INTTM011	TM01 and CR011 match	TM01	0130H	00000130H	nextPC	TM0IC11
		12	INTTM50	TM50 and CR50 match	TM50	0140H	00000140H	nextPC	TM5IC0
		13	INTTM51	TM51 and CR51 match	TM51	0150H	00000150H	nextPC	TM5IC1
		14	INTCSI00	CSI00 transfer completion	CSI00	0160H	00000160H	nextPC	CSI0IC0
		15	INTCSI01	CSI01 transfer completion	CSI01	0170H	00000170H	nextPC	CSI0IC1
		16	INTSRE0	UART0 reception error occurrence	UART0	0180H	00000180H	nextPC	SREIC0
		17	INTSR0	UART0 reception completion	UART0	0190H	00000190H	nextPC	SRIC0
		18	INTST0	UART0 transmission completion	UART0	01A0H	000001AH	nextPC	STIC0
		19	INTSRE1	UART1 reception error occurrence	UART1	01B0H	000001B0H	nextPC	SREIC1
		20	INTSR1	UART1 reception completion	UART1	01C0H	000001C0H	nextPC	SRIC1
		21	INTST1	UART1 transmission completion	UART1	01D0H	000001D0H	nextPC	STIC1

Notes 1. For restoration in the case of INTWDT1 and INTWDT2, refer to **17.10 Cautions**.

2. n = 0 to FH

Downloaded from Arrow.com.

Table 17-1. Interrupt Source List (2/2)

Type	Classification	Default Priority	Name	Trigger	Interrupt Source	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	22	INTTMH0	TMH0 and CMP00/CMP01 match	TMH0	01E0H	000001E0H	nextPC	TMHIC0
		23	INTTMH1	TMH1 and CMP10/CMP11 match	TMH1	01F0H	000001F0H	nextPC	TMHIC1
		25	INTIIC0Note	I^2C0 transfer completion	I^2C0	0210H	00000210H	nextPC	IICIC0
		26	INTAD	A/D conversion completion	A/D	0220H	00000220H	nextPC	ADIC
		27	INTKR	Key return interrupt	KR	0230H	00000230H	nextPC	KRIC
		28	INTWTI	Watch timer interval	WT	0240H	00000240H	nextPC	WTIIC
		29	INTWT	Watch timer reference time	WT	0250H	00000250H	nextPC	WTIC
		30	INTBRG	8-bit counter of prescaler 3 and PRSCM match	Prescaler 3	0260H	00000260H	nextPC	BRGIC
		45	INTTP0OV	TMP0 overflow	TMP0	03A0H	000003A0H	nextPC	TP0OVIC
		46	INTTP0CC0	TP0CCR0 capture/ TMP0 and TP0CCR0 match	TMP0	03B0H	000003B0H	nextPC	TP0CCIC0
		47	INTTP0CC1	TP0CCR1 capture/ TMP0 and TP0CCR1 match	TMP0	03C0H	000003C0H	nextPC	TP0CCIC1

Note Only in the μPD703207Y, 70F3207HY

Remarks 1. Default priority: The priority order when two or more maskable interrupt requests with the same priority level are generated at the same time. The highest priority is 0.
The priority of non-maskable interrupt request is as follows.
INTWDT2 > INTWDT1 > NMI

Restored PC: The value of the program counter (PC) saved to EIPC, FEPC, or DBPC when interrupt/exception processing is started. The restored PC when a non-maskable or maskable interrupt is acknowledged while either of the following instructions is being executed does not become nextPC (when an interrupt is acknowledged during the execution of an instruction, the execution of that instruction is stopped and is resumed following completion of interrupt servicing).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)
- Divide instructions (DIV, DIVH, DIVU, DIVHU)
- PREPARE, DISPOSE instructions (only when an interrupt occurs before stack pointer update)

nextPC: The PC value at which processing is started following interrupt/exception processing.

2. The execution address of the illegal opcode when an illegal opcode exception occurs is calculated with (Restored PC – 4).

Downloaded from Arrow.com.

17.2 Non-Maskable Interrupts

Non-maskable interrupt request signals are acknowledged unconditionally, even when interrupts are disabled (DI state). Non-maskable interrupts (NMI) are not subject to priority control and take precedence over all other interrupt request signals.

The following three types of non-maskable interrupt request signals are available in the V850ES/KE1.

- NMI pin input (NMI)
- Non-maskable interrupt request signal (INTWDT1) due to overflow of watchdog timer 1
- Non-maskable interrupt request signal (INTWDT2) due to overflow of watchdog timer 2

There are four choices for the valid edge of an NMI pin, namely: rising edge, falling edge, both edges, and no edge detection.

The non-maskable interrupt request signal (INTWDT1) due to overflow of watchdog timer 1 functions by setting the WDTM1.WDTM14 and WDTM1.WDTM13 bits to 10.

The non-maskable interrupt request signal (INTWDT2) due to overflow of watchdog timer 2 functions by setting the WDTM2.WDM21 and WDTM2.WDM20 bits to 01.

When two or more non-maskable interrupts occur simultaneously, they are processed in a sequence determined by the following priority order (the interrupt request signals with low priority level are ignored).

INTWDT2 > INTWDT1 > NMI

If during NMI processing, an NMI, INTWDT1, or INTWDT2 request signal newly occurs, processing is performed as follows.

(1) If an NMI request signal newly occurs during NMI processing

The new NMI request signal is held pending regardless of the value of the PSW.NP bit. The NMI request signal held pending is acknowledged upon completion of processing of the NMI currently being executed (following RETI instruction execution).

(2) If an INTWDT1 request signal newly occurs during NMI processing

If the NP bit remains set (to 1) during NMI processing, the new INTWDT1 request signal is held pending. The INTWDT1 request signal held pending is acknowledged upon completion of processing of the NMI currently being executed (following RETI instruction execution).

If the NP bit is cleared (to 0) during NMI processing, a newly generated INTWDT1 request signal is executed (NMI processing is interrupted).

(3) If an INTWDT2 request signal newly occurs during NMI processing

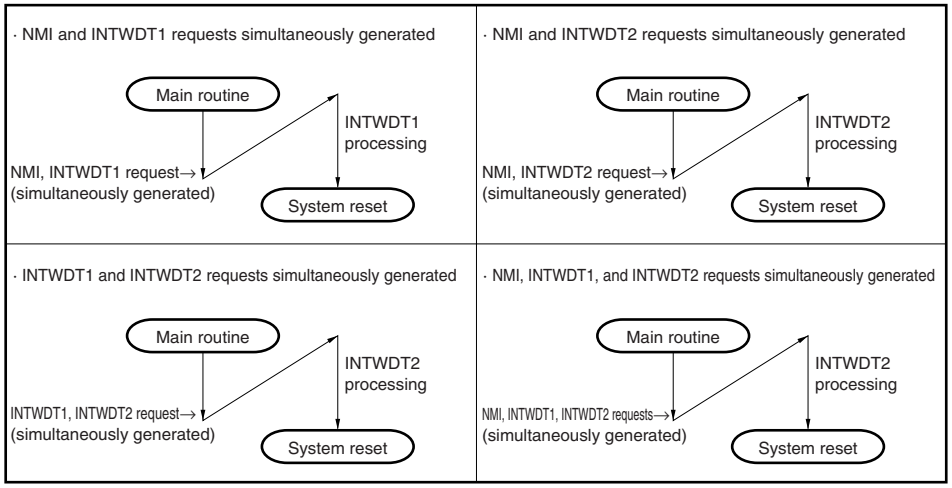
A newly generated INTWDT2 request signal is executed regardless of the value of the NP bit (NMI processing is interrupted).

Caution For non-maskable interrupt servicing from non-maskable interrupt request signals (INTWDT1, INTWDT2), refer to 17.10 Cautions.

Downloaded from Arrow.com.

Figure 17-1. Acknowledging Non-Maskable Interrupt Request Signals (1/2)

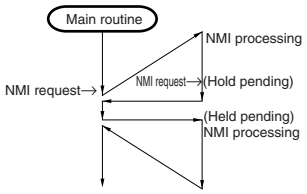
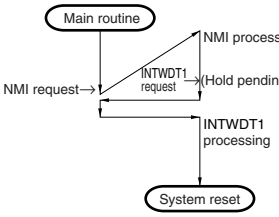
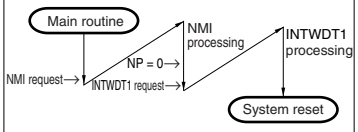
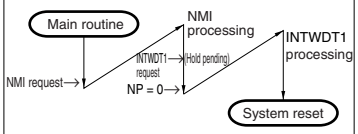
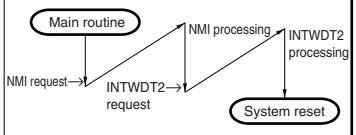
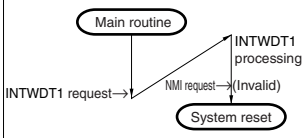
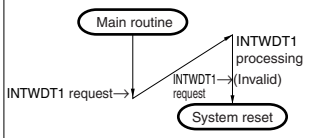
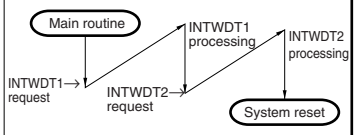
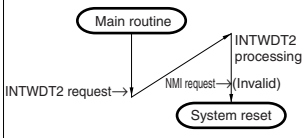
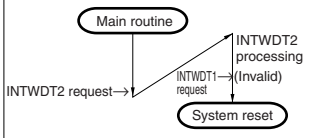
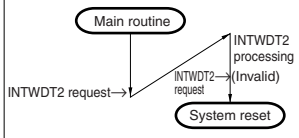
(a) If two or more NMI request signals are simultaneously generated



Downloaded from Arrow.com.

Figure 17-1. Acknowledging Non-Maskable Interrupt Request Signals (2/2)

(b) If a new non-maskable interrupt request signal is generated during a non-maskable interrupt servicing

Non-maskable interrupt currently being serviced	Non-maskable interrupt request newly generated during non-maskable interrupt servicing		
	NMI	INTWDT1	INTWDT2
NMI	· Generation of NMI request during NMI processing 	· Generation of INTWDT1 request during NMI processing (NP = 1 state prior to INTWDT1 request is maintained)  · Generation of INTWDT1 request during NMI processing (Set NP = 0 before INTWDT1 request)  · Generation of INTWDT1 request during NMI processing (Set NP = 0 after INTWDT1 request) 	· Generation of INTWDT2 request during NMI processing 
INTWDT1	· Generation of NMI request during INTWDT1 processing 	· Generation of INTWDT1 request during INTWDT1 processing 	· Generation of INTWDT2 request during INTWDT1 processing 
INTWDT2	· Generation of NMI request during INTWDT2 processing 	· Generation of INTWDT1 request during INTWDT2 processing 	· Generation of INTWDT2 request during INTWDT2 processing 

Downloaded from Arrow.com.

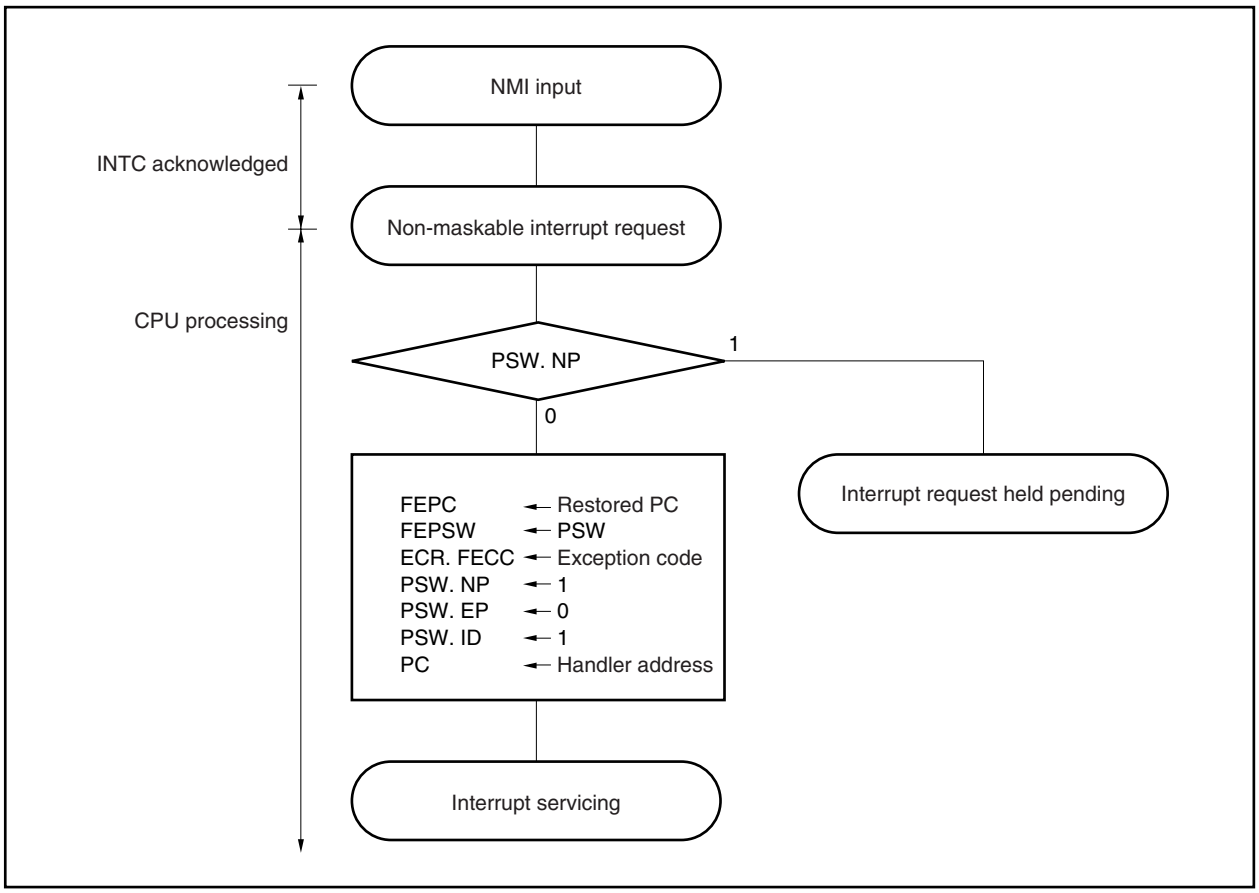
17.2.1 Operation

Upon generation of a non-maskable interrupt request signal, the CPU performs the following processing and transfers control to a handler routine.

<1> Saves the restored PC to FEPC.
<2> Saves the current PSW to FEPSW.
<3> Writes the exception code (0010H, 0020H, 0030H) to the higher halfword (FECC) of ECR.
<4> Sets the PSW.NP and PSW.ID bits to 1 and clears the PSW.EP bit to 0.
<5> Loads the handler address (00000010H, 00000020H, 00000030H) of the non-maskable interrupt to the PC and transfers control.

Figure 17-2 shows the servicing flow for non-maskable interrupts.

Figure 17-2. Non-Maskable Interrupt Servicing



Downloaded from Arrow.com.

17.2.2 Restore

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

(1) In case of NMI

Restore from NMI processing is done with the RETI instruction.

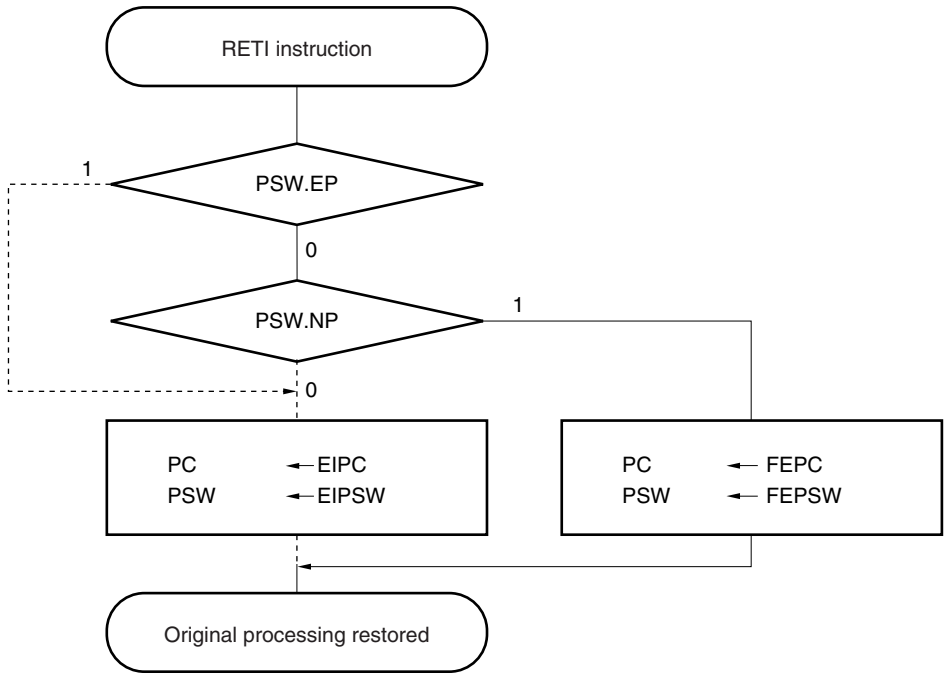
When the RETI instruction is executed, the CPU performs the following processing and transfers control to the address of the restored PC.

(i) Loads the values of the restored PC and PSW from FEPC and FEPSW, respectively, because the PSW.EP bit and the PSW.NP bit are 0 and 1, respectively.

(ii) Transfers control back to the loaded address of the restored PC and PSW.

Figure 17-3 shows the processing flow of the RETI instruction.

Figure 17-3. RETI Instruction Processing



Caution When the EP bit and the NP bit are changed by the LDSR instruction during non-maskable interrupt servicing, in order to restore the PC and PSW correctly during restoring by the RETI instruction, it is necessary to clear the EP bit back to 0 and set the NP bit back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

(2) In case of INTWDT1, INTWDT2 signals

For non-maskable interrupt servicing by the non-maskable interrupt request signals (INTWDT1, INTWDT2), refer to **17.10 Cautions**.

Downloaded from Arrow.com.

17.2.3 NP flag

The NP flag is a status flag that indicates that non-maskable interrupt servicing is in progress. This flag is set when a non-maskable interrupt request has been acknowledged, and masks all non-maskable requests to prevent multiple interrupts.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	ID	SAT	CY	OV	S	Z

NP	NMI servicing status
0	No non-maskable interrupt servicing
1	Non-maskable interrupt serving in progress

Downloaded from Arrow.com.

17.3 Maskable Interrupts

Maskable interrupt request signals can be masked by interrupt control registers. The V850ES/KE1 has 30 or 31 maskable interrupt sources (refer to **17.1.1 Features**).

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

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

When the EI instruction is executed in an interrupt servicing routine, the interrupt enabled (EI) status is set, which enables acknowledgment of interrupt request signals having a priority higher than that of the interrupt request signal currently in progress. Note that only interrupt request signals with a higher priority have this capability; interrupt request signals with the same priority level cannot be nested.

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

When the WDTM1.WDTM14 bit is cleared to 0, the watchdog timer 1 overflow interrupt functions as a maskable interrupt (INTWDTM1).

17.3.1 Operation

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

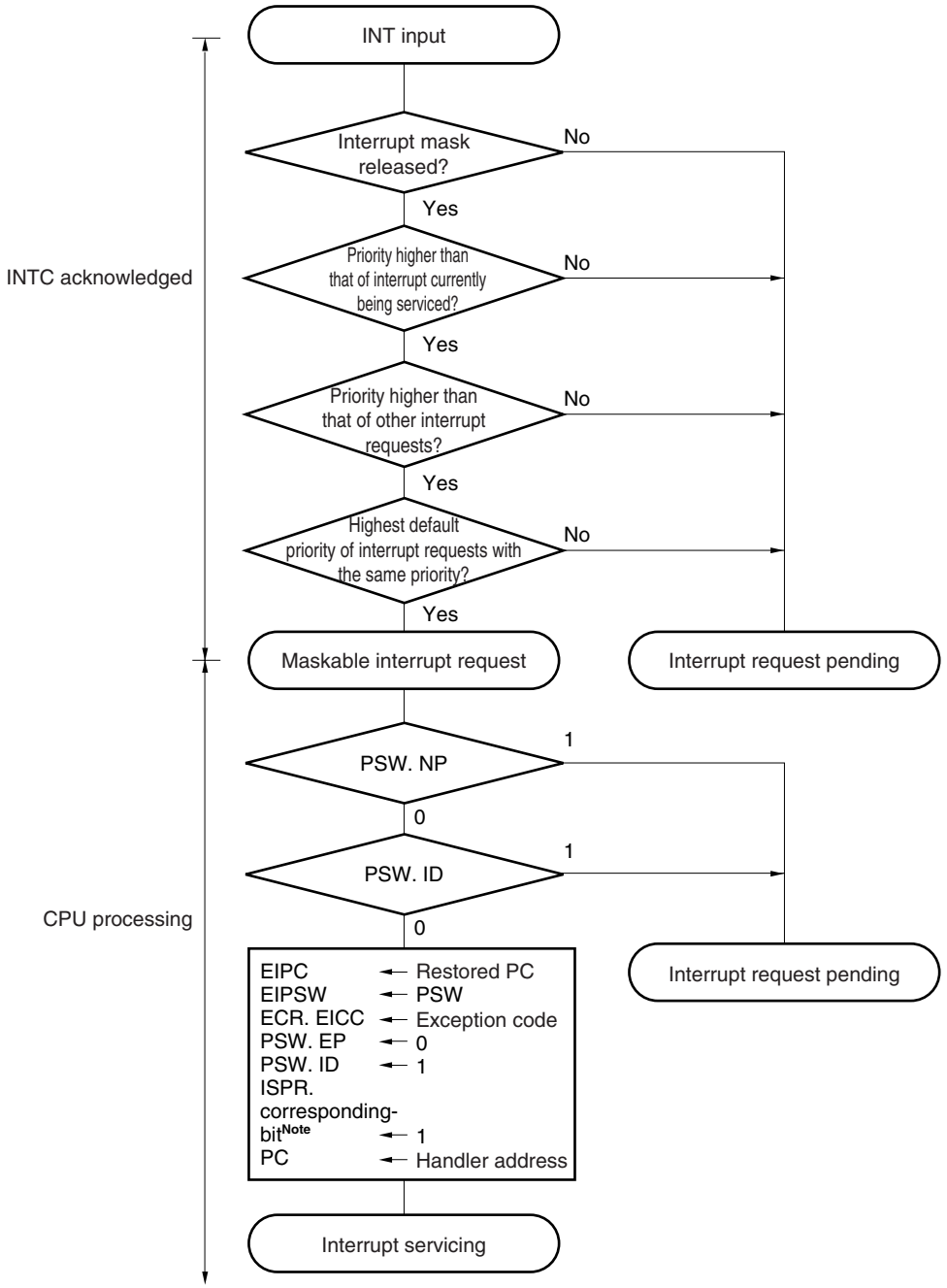
<1> Saves the restored PC to EIPC.
<2> Saves the current PSW to EIPSW.
<3> Writes an exception code to the lower halfword of ECR (EICC).
<4> Sets the PSW.ID bit to 1 and clears the PSW.EP bit to 0.
<5> Loads the corresponding handler address to the PC and transfers control.

The maskable interrupt request signal masked by INTC and the maskable interrupt request signal that occurs while another interrupt is being serviced (when PSW.NP bit = 1 or ID bit = 1) are held pending internally. When the interrupts are unmasked, or when the NP bit = 0 and the ID bit = 0 by using the RETI and LDSR instructions, a new maskable interrupt servicing is started in accordance with the priority of the pending maskable interrupt request signal.

Figure 17-4 shows the servicing flow for maskable interrupts.

Downloaded from Arrow.com.

Figure 17-4. Maskable Interrupt Servicing



Note For the ISPR register, refer to **17.3.6 In-service priority register (ISPR)**.

Downloaded from Arrow.com.

17.3.2 Restore

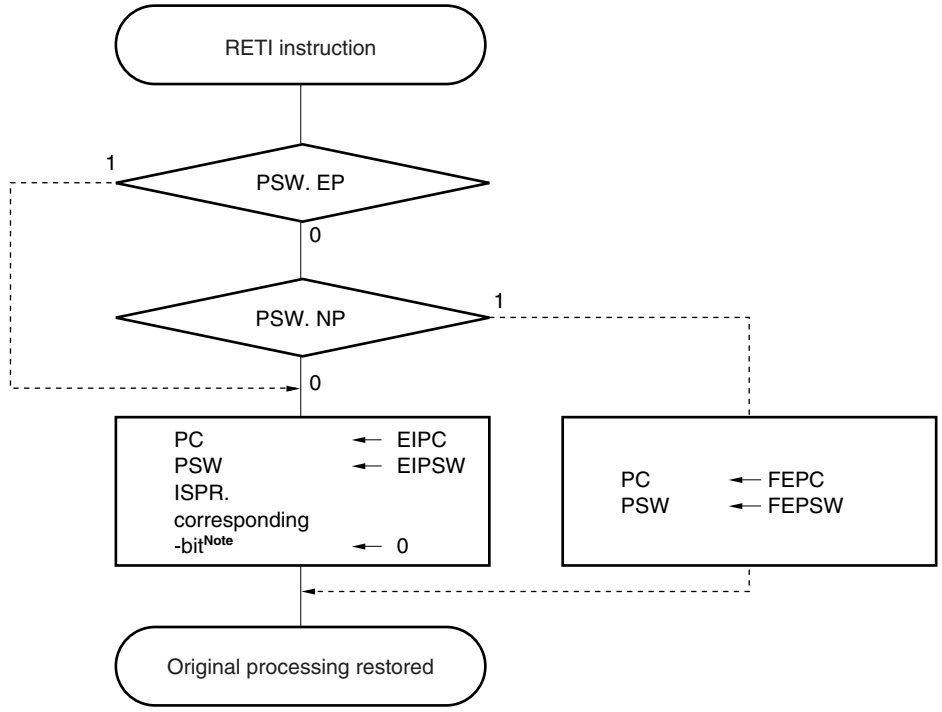
Execution is restored from maskable interrupt servicing by the RETI instruction.

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

(1) Loads the values of the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit and the PSW.NP bit are both 0.
(2) Transfers control to the loaded address of the restored PC and PSW.

Figure 17-5 shows the processing flow of the RETI instruction.

Figure 17-5. RETI Instruction Processing



Note For the ISPR register, refer to **17.3.6 In-service priority register (ISPR)**.

Caution When the EP bit and the NP bit are changed by the LDSR instruction during maskable interrupt servicing, in order to restore the PC and PSW correctly during restoring by the RETI instruction, it is necessary to clear the EP bit back to 0 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

Downloaded from Arrow.com.

17.3.3 Priorities of maskable interrupts

INTC provides a multiple interrupt servicing in which an interrupt can be acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels specified by the interrupt priority level specification bit (xxICn.xxPRn bit). When two or more interrupts having the same priority level specified by xxPRn are generated at the same time, interrupts are serviced in order depending on the priority level allocated to each interrupt request (default priority level) beforehand. For more information, refer to **Table 17-1 Interrupt Source List**. Programmable priority control divides interrupt requests into eight levels by setting the priority level specification flag.

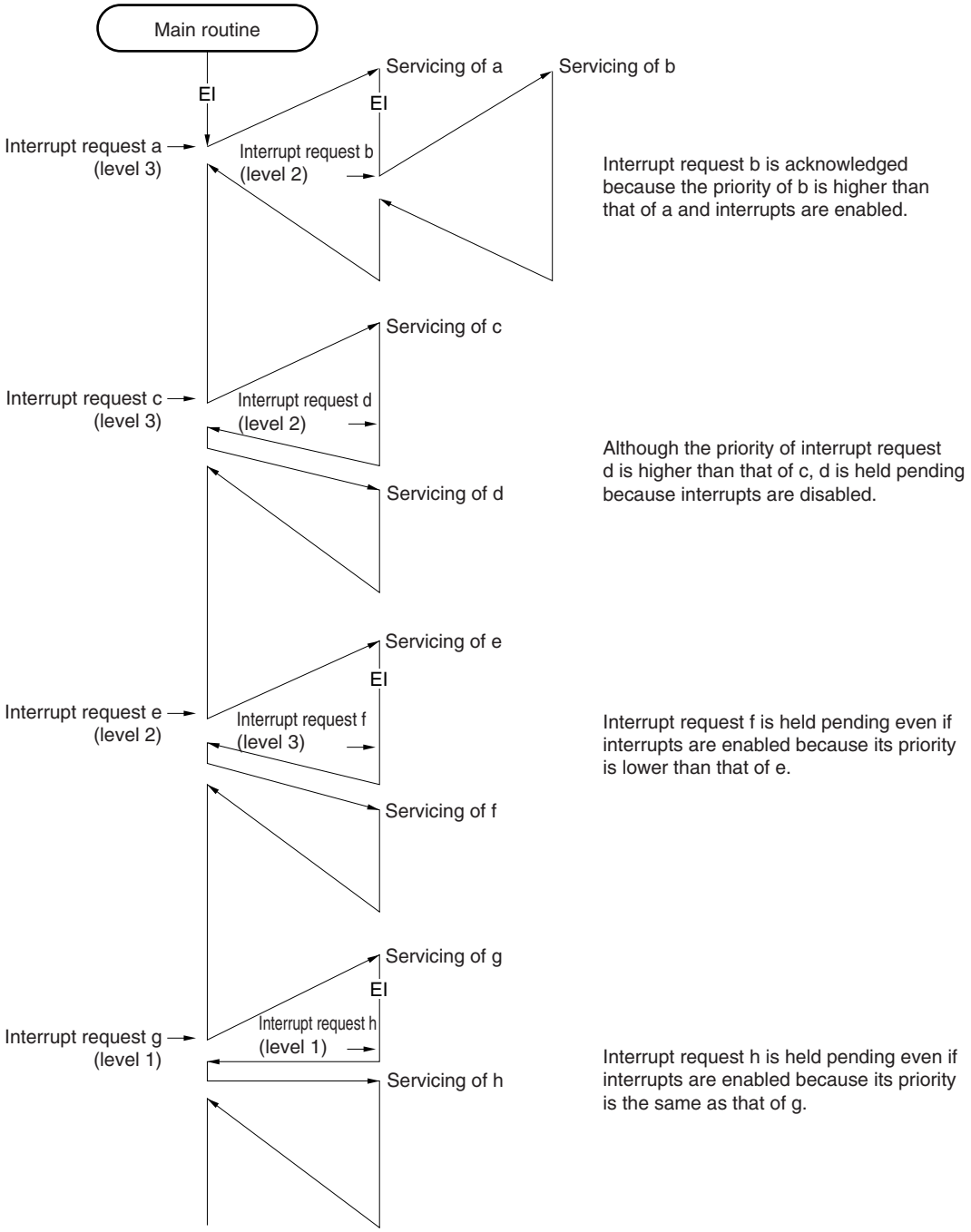
Note that when an interrupt request signal is acknowledged, the PSW.ID flag is automatically set (1). Therefore, when multiple interrupts are to be used, clear (0) the ID flag beforehand (for example, by placing the EI instruction into the interrupt service program) to enable interrupts.

Remark xx: Identifying name of each peripheral unit (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

n: Peripheral unit number (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

Downloaded from Arrow.com.

Figure 17-6. Example of Interrupt Nesting (1/2)



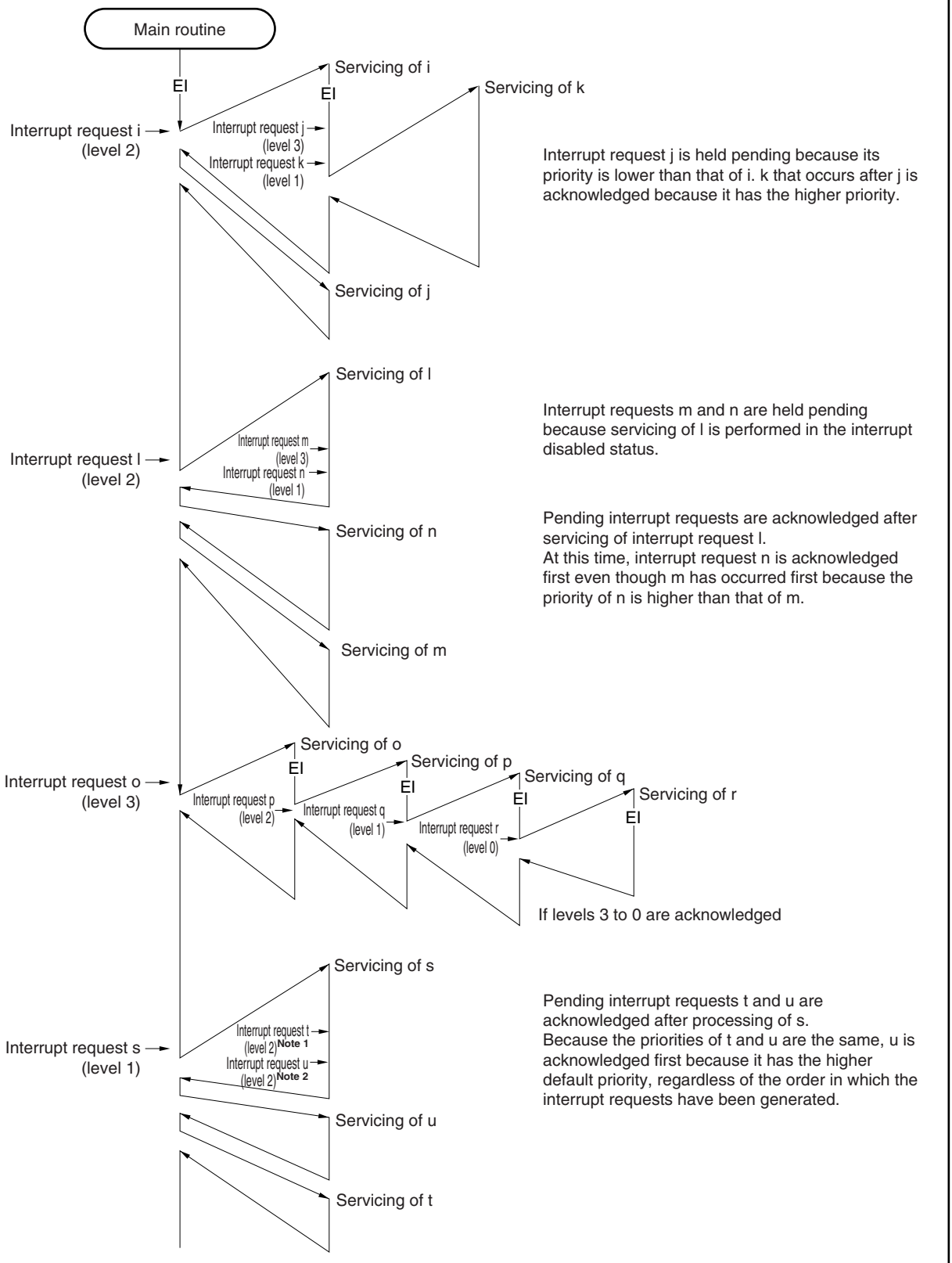
Caution The values of EIPC and EIPSW must be saved before executing multiple interrupts.

Remarks 1. a to u in the figure are the names of interrupt request signals shown for the sake of explanation.

2. The default priority in the figure indicates the relative priority between two interrupt request signals.

Downloaded from Arrow.com.

Figure 17-6. Example of Interrupt Nesting (2/2)

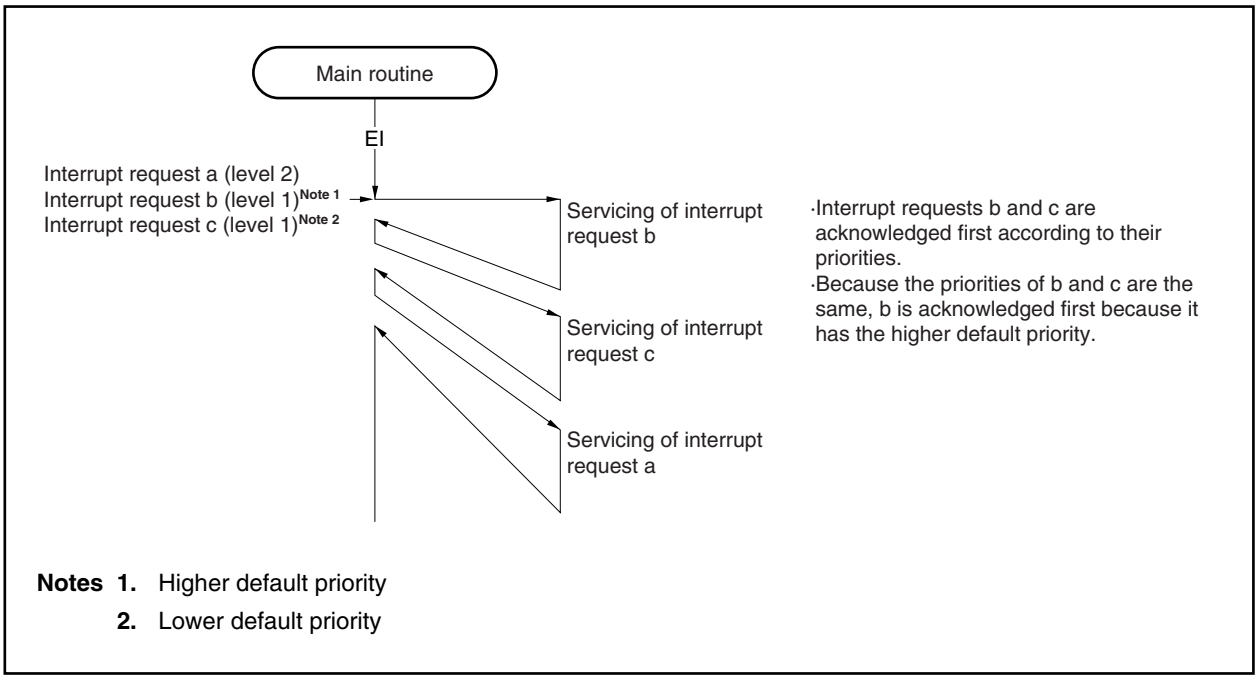


Notes 1. Lower default priority

2. Higher default priority

Downloaded from Arrow.com.

Figure 17-7. Example of Servicing Simultaneously Generated Interrupt Request Signals



Notes 1. Higher default priority

2. Lower default priority

Downloaded from Arrow.com.

17.3.4 Interrupt control register (xxICn)

An interrupt control register is assigned to each maskable interrupt and sets the control conditions for each maskable interrupt request. The interrupt control registers can be read or written in 8-bit or 1-bit units.

Reset sets xxICn to 47H.

Caution Be sure to read the xxICn.xxIFn bit while interrupts are disabled (DI). If the xxIFn bit is read while interrupts are enabled (EI), an incorrect value may be read if there is a conflict between acknowledgment of the interrupt and reading of the bit.

After reset: 47H R/W Address: FFFFF110H to FFFFF168H

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

xxIFn	Interrupt request flag[Note]
0	Interrupt request not generated
1	Interrupt request generated

xxMKn	Interrupt mask flag
0	Enables interrupt servicing
1	Disables interrupt servicing (pending)

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

Note Automatically reset by hardware when interrupt request is acknowledged.

Remark xx: Identifying name of each peripheral unit (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

n: Peripheral unit number (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

Following tables list the addresses and bits of the interrupt control registers.

Downloaded from Arrow.com.

Table 17-2. Interrupt Control Registers (xxICn)

Address	Register	Bits							
		<7>	<6>	5	4	3	2	1	0
FFFFF110H	WDT1IC	WDT1IF	WDT1MK	0	0	0	WDT1PR2	WDT1PR1	WDT1PR0
FFFFF112H	PIC0	PIF0	PMK0	0	0	0	PPR02	PPR01	PPR00
FFFFF114H	PIC1	PIF1	PMK1	0	0	0	PPR12	PPR11	PPR10
FFFFF116H	PIC2	PIF2	PMK2	0	0	0	PPR22	PPR21	PPR20
FFFFF118H	PIC3	PIF3	PMK3	0	0	0	PPR32	PPR31	PPR30
FFFFF11AH	PIC4	PIF4	PMK4	0	0	0	PPR42	PPR41	PPR40
FFFFF11CH	PIC5	PIF5	PMK5	0	0	0	PPR52	PPR51	PPR50
FFFFF11EH	PIC6	PIF6	PMK6	0	0	0	PPR62	PPR61	PPR60
FFFFF124H	TM0IC10	TM0IF10	TM0MK10	0	0	0	TM0PR102	TM0PR101	TM0PR100
FFFFF126H	TM0IC11	TM0IF11	TM0MK11	0	0	0	TM0PR112	TM0PR111	TM0PR110
FFFFF128H	TM5IC0	TM5IF0	TM5MK0	0	0	0	TM5PR02	TM5PR01	TM5PR00
FFFFF12AH	TM5IC1	TM5IF1	TM5MK1	0	0	0	TM5PR12	TM5PR11	TM5PR10
FFFFF12CH	CSI0IC0	CSI0IF0	CSI0MK0	0	0	0	CSI0PR02	CSI0PR01	CSI0PR00
FFFFF12EH	CSI0IC1	CSI0IF1	CSI0MK1	0	0	0	CSI0PR12	CSI0PR11	CSI0PR10
FFFFF130H	SREIC0	SREIF0	SREMK0	0	0	0	SREPR02	SREPR01	SREPR00
FFFFF132H	SRIC0	SRIF0	SRMK0	0	0	0	SRPR02	SRPR01	SRPR00
FFFFF134H	STIC0	STIF0	STMK0	0	0	0	STPR02	STPR01	STPR00
FFFFF136H	SREIC1	SREIF1	SREMK1	0	0	0	SREPR12	SREPR11	SREPR10
FFFFF138H	SRIC1	SRIF1	SRMK1	0	0	0	SRPR12	SRPR11	SRPR10
FFFFF13AH	STIC1	STIF1	STMK1	0	0	0	STPR12	STPR11	STPR10
FFFFF13CH	TMHIC0	TMHIF0	TMHMK0	0	0	0	TMHPR02	TMHPR01	TMHPR00
FFFFF13EH	TMHIC1	TMHIF1	TMHMK1	0	0	0	TMHPR12	TMHPR11	TMHPR10
FFFFF142H	IICIC0[Note]	IICIF0	IICMK0	0	0	0	IICPR02	IICPR01	IICPR00
FFFFF144H	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
FFFFF146H	KRIC	KRIF	KRMK	0	0	0	KRPR2	KRPR1	KRPR0
FFFFF148H	WTIIC	WTIIF	WTIMK	0	0	0	WTIPR2	WTIPR1	WTIPR0
FFFFF14AH	WTIC	WTIF	WTMK	0	0	0	WTPR2	WTPR1	WTPR0
FFFFF14CH	BRGIC	BRGIF	BRGMK	0	0	0	BRGPR2	BRGPR1	BRGPR0
FFFFF174H	TP0OVIC	TP0OVIF	TP0OVMK	0	0	0	TP0OVPR2	TP0OVPR1	TP0OVPR0
FFFFF176H	TP0CCIC0	TP0CCIF0	TP0CCMK0	0	0	0	TP0CCPR02	TP0CCPR01	TP0CCPR00
FFFFF178H	TP0CCIC1	TP0CCIF1	TP0CCMK1	0	0	0	TP0CCPR12	TP0CCPR11	TP0CCPR10

Note Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

17.3.5 Interrupt mask registers 0, 1, 3 (IMR0, IMR1, IMR3)

These registers set the interrupt mask status for maskable interrupts. The xxMKn bit of the IMR0, IMR1, and IMR3 registers and the xxMKn bit of the xxICn register are respectively linked.

The IMRm register can be read or written in 16-bit units (m = 0, 1, 3).

When the higher 8 bits of the IMRk register are treated as the IMRkH register and the lower 8 bits of the IMRk register as the IMRkL register, they can be read or written in 8-bit or 1-bit units (k = 0, 1).

Caution In the device file, the xxMKn bit of the xxICn register is defined as a reserved word. Therefore, if bit manipulation is performed using the name xxMKn, the xxICn register, not the IMRm register, is rewritten (as a result, the IMRm register is also rewritten).

After reset: FFFFH R/W Address: IMR0 FFFFF100H, IMR0L FFFFF100H, IMR0H FFFFF101H

	15	14	13	12	11	10	9	8
IMR0 (IMR0H[Note])	CSI0MK1	CSI0MK0	TM5MK1	TM5MK0	TM0MK11	TM0MK10	1	1
	7	6	5	4	3	2	1	0
(IMR0L)	PMK6	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	WDT1MK

After reset: FFFFH R/W Address: IMR1 FFFFF102H, IMR1L FFFFF102H, IMR1H FFFFF103H

	15	14	13	12	11	10	9	8
IMR1 (IMR1H[Note])	1	BRGMK	WTMK	WTIMK	KRMK	ADMK	IICMK0	1
	7	6	5	4	3	2	1	0
(IMR1L)	TMHMK1	TMHMK0	STMK1	SRMK1	SREMK1	STMK0	SRMK0	SREMK0

After reset: FFFFH R/W Address: IMR3, IMR3L FFFFF106H

	15	14	13	12	11	10	9	8
IMR3	1	1	1	1	1	1	1	1
	7	6	5	4	3	2	1	0
(IMR3L)	1	1	1	TP0CCMK1	TP0CCMK0	TP0OVMK	1	1

xxMKn	Interrupt mask flag setting
0	Enables interrupt servicing
1	Disables interrupt servicing

Note When reading from or writing to bits 8 to 15 of the IMR0 and IMR1 registers in 8-bit or 1-bit units, specify these bits as bits 0 to 7 of the IMR0H and IMR1H registers.

Caution Set bits 9 and 8 of the IMR0 register, bits 15 and 8 of the IMR1 register, and bits 15 to 5, 1, and 0 of the IMR3 register to 1. The operation is not guaranteed if their value is changed.

Remark xx: Identifying name of each peripheral unit (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

n: Peripheral unit number (refer to **Table 17-2 Interrupt Control Registers (xxICn)**)

Downloaded from Arrow.com.

17.3.6 In-service priority register (ISPR)

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

When the RETI instruction is executed, the bit among those that are set (1) in the ISPR register that corresponds to the interrupt request signal having the highest priority is automatically cleared (0) by hardware. However, it is not cleared (0) when execution is returned from non-maskable interrupt servicing or exception processing.

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

Reset sets ISPR to 00H.

Caution If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set to 1 by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI status).

After reset: 00H R Address: FFFFF1FAH

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
ISPR	ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0

ISPRn	Priority of interrupt currently being acknowledged
0	Interrupt request with priority n is not acknowledged
1	Interrupt request with priority n is being acknowledged

Remark n = 0 to 7 (priority level)

Downloaded from Arrow.com.

17.3.7 ID flag

The interrupt disable flag (ID) is allocated to the PSW and controls the maskable interrupt's operating state, and stores control information regarding enabling/disabling reception of interrupt request signals.

Reset sets this flag to 00000020H.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	ID	SAT	CY	OV	S	Z

ID	Maskable interrupt servicing specification[Note]
0	Maskable interrupt request signal acknowledgment enabled
1	Maskable interrupt request signal acknowledgment disabled

Note Interrupt disable flag (ID) function

ID is set (1) by the DI instruction and cleared (0) by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW.

Non-maskable interrupt request signals and exceptions are acknowledged regardless of this flag. When a maskable interrupt request signal is acknowledged, the ID flag is automatically set (1) by hardware.

An interrupt request signal generated during the acknowledgment disabled period (ID flag = 1) can be acknowledged when the xxICn.xxIFn bit is set (1), and the ID flag is cleared (0).

Downloaded from Arrow.com.

17.3.8 Watchdog timer mode register 1 (WDTM1)

This register is a special register that can be written to only in a special sequence. To generate a maskable interrupt (INTWDT1), clear the WDTM14 bit to 0.

This register can be read or written in 8-bit or 1-bit units (for details, refer to **CHAPTER 11 WATCHDOG TIMER FUNCTIONS**).

After reset: 00H R/W Address: FFFFF6C2H

	<7>	6	5	4	3	2	1	0
WDTM1	RUN1	0	0	WDTM14	WDTM13	0	0	0

RUN1	Watchdog timer operation mode selection[Note 1]
0	Stop count operation
1	Clear counter and start count operation

WDTM14	WDTM13	Watchdog timer operation mode selection[Note 2]
0	0	Interval timer mode (Generate maskable interrupt INTWDTM1 when overflow occurs)
0	1	
1	0	Watchdog timer mode 1[Note 3] (Generate non-maskable interrupt INTWDT1 when overflow occurs)
1	1	Watchdog timer mode 2 (Start WDTRES2 reset operation when overflow occurs)

Notes 1. Once the RUN1 bit has been set (1), it cannot be cleared (0) by software. Therefore, once counting starts, it cannot be stopped except by reset.

2. Once the WDTM14 and WDTM13 bits have been set (1), they cannot be cleared (0) by software. Reset is the only way to clear these bits.

3. For non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT1), refer to **17.10 Cautions**.

Downloaded from Arrow.com.

17.4 External Interrupt Request Input Pins (NMI, INTP0 to INTP6)

17.4.1 Noise elimination

(1) Noise elimination for NMI pin

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

The NMI pin is used for releasing the STOP mode. In the STOP mode, noise elimination using the system clock is not performed because the internal system clock is stopped.

(2) Noise elimination for INTP0 to INTP6 pins

The INTP0 to INTP6 pins include a noise eliminator that operates using analog delay. Therefore, a signal input to each pin is not detected as an edge unless it maintains its input level for a certain period. The edge is detected only after a certain period has elapsed.

17.4.2 Edge detection

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

- Rising edge
- Falling edge
- Both edges
- No edge detection

After reset, the edge detection for the NMI pin is set to “no edge detection”. Therefore, interrupt requests cannot be acknowledged (the NMI pin functions as a normal port) unless a valid edge is specified by the INTR0, INTF0, INTR9H, and INTF9H registers.

When using the P02/NMI pin as an output port, set the NMI pin valid edge to “no edge detection”.

Downloaded from Arrow.com.

(1) External interrupt rising and falling edge specification registers 0 (INTR0, INTF0)

These are 8-bit registers that specify detection of the rising and falling edges of the NMI and INTP0 to INTP3 pins.

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

Reset sets these registers to 00H.

Caution When switching to the port function from the external interrupt function (alternate function), edge detection may be performed. Therefore, set the port mode after setting the INTF0n and INTR0n bits = 00.

After reset: 00H R/W Address: INTR0 FFFFFC20H, INTF0 FFFFFC00H

	7	6	5	4	3	2	1	0
INTR0	0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0
		INTP3	INTP2	INTP1	INTP0	NMI		
INTF0	0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0
		INTP3	INTP2	INTP1	INTP0	NMI		

Remark For specification of the valid edge, refer to **Table 17-3**.

Table 17-3. NMI and INTP0 to INTP3 Pins Valid Edge Specification

INTF0n	INTR0n	Valid edge specification (n = 2 to 6)
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edges

Remark n = 2: Control of NMI pin

n = 3 to 6: Control of INTP0 to INTP3 pins

Downloaded from Arrow.com.

(2) External interrupt rising and falling edge specification registers 9H (INTR9H, INTF9H)

These are 8-bit registers that specify detection of the rising edge of the INTP4 to INTP6 pins.

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

Reset sets these registers to 00H.

Caution When switching to the port function from the external interrupt function (alternate function), edge detection may be performed. Therefore, set the port mode after setting the INTF9n and INTR9n bits = 00.

After reset: 00H R/W Address: INTR9H FFFFFC33H, INTF9H FFFFFC13H

	7	6	5	4	3	2	1	0
INTR9H	INTR915	INTR914	INTR913	0	0	0	0	0
	INTP6	INTP5	INTP4					
	7	6	5	4	3	2	1	0
INTF9H	INTF915	INTF914	INTF913	0	0	0	0	0
	INTP6	INTP5	INTP4					

Remark For specification of the valid edge, refer to **Table 17-4**.

Table 17-4. INTP4 to INTP6 Pins Valid Edge Specification

INTF9n	INTR9n	Valid edge specification (n = 13 to 15)
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edges

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

Downloaded from Arrow.com.

17.5 Software Exceptions

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

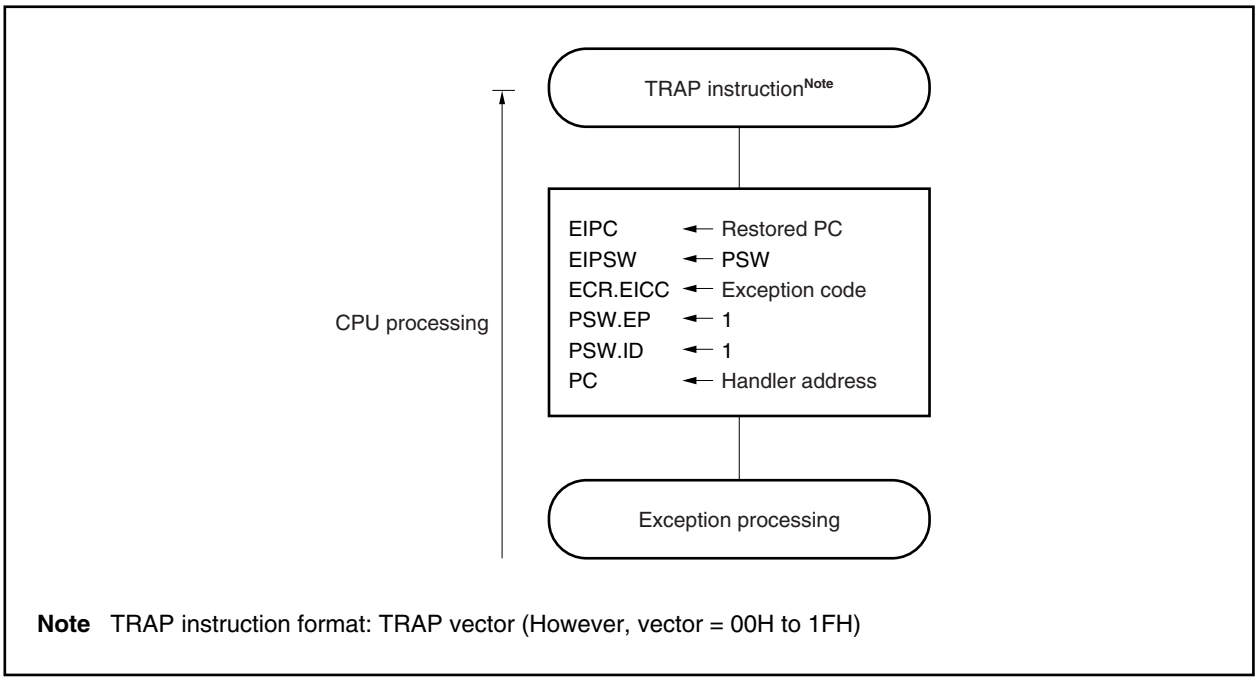
17.5.1 Operation

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

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

Figure 17-8 shows the software exception processing flow.

Figure 17-8. Software Exception Processing



Note TRAP instruction format: TRAP vector (However, vector = 00H to 1FH)

The handler address is determined by the operand (vector) of the TRAP instruction. If the vector is 00H to 1FH, the handler address is 00000040H, and if the vector is 10H to 1FH, the handler address is 00000050H.

Downloaded from Arrow.com.

17.5.2 Restore

Execution is restored from software exception processing by the RETI instruction.

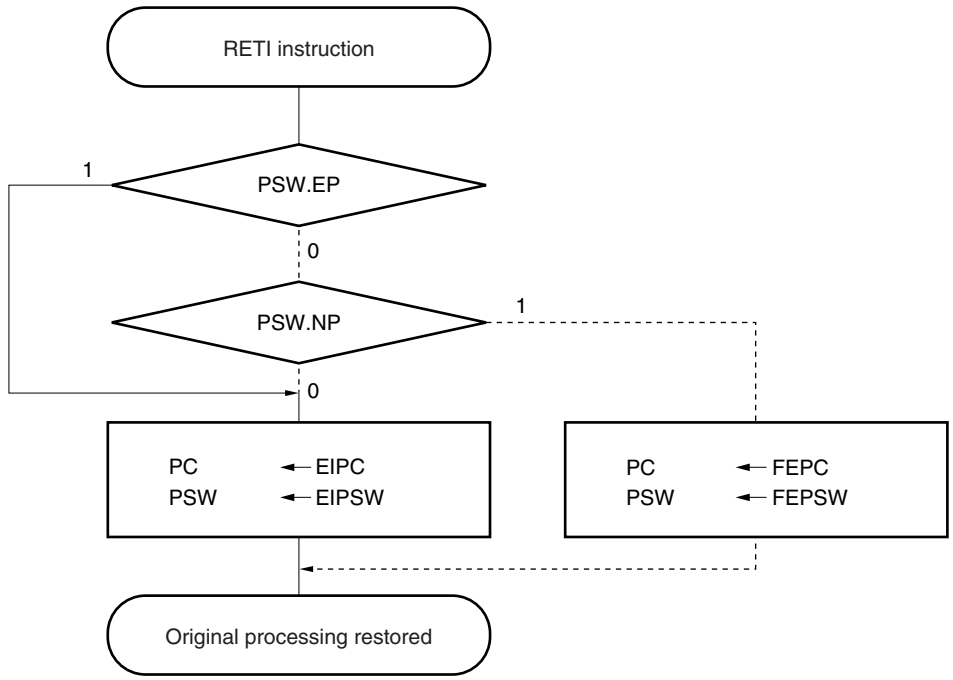
When the RETI instruction is executed, the CPU performs the following processing and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit is 1.

<2> Transfers control to the address of the restored PC and PSW.

Figure 17-9 shows the processing flow of the RETI instruction.

Figure 17-9. RETI Instruction Processing



Caution When the EP bit and the NP bit are changed by the LDSR instruction during software exception processing, in order to restore the PC and PSW correctly during restoring by the RETI instruction, it is necessary to set the PSW.EP bit back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

Downloaded from Arrow.com.

17.5.3 EP flag

The EP flag, which is bit 6 of the PSW, is a status flag that indicates that exception processing is in progress. It is set when an exception occurs.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	ID	SAT	CY	OV	S	Z

EP	Exception processing status
0	Exception processing not in progress
1	Exception processing in progress

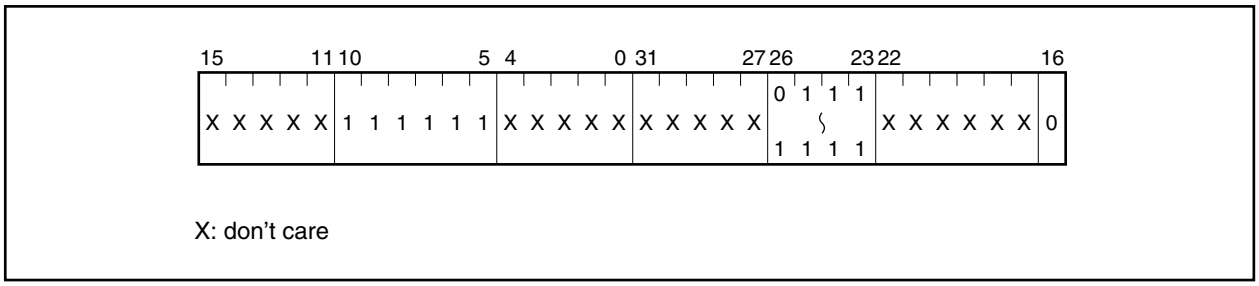
Downloaded from Arrow.com.

17.6 Exception Trap

The exception trap is an interrupt that is requested when the illegal execution of an instruction takes place. In the V850ES/KE1, an illegal opcode trap (ILGOP) is considered as an exception trap.

17.6.1 Illegal opcode

An illegal opcode is defined as an instruction with instruction opcode (bits 10 to 5) = 111111B, sub-opcode (bits 26 to 23) = 0111B to 1111B, and sub-opcode (bit 16) = 0B. When such an instruction is executed, an exception trap is generated.



X: don't care

Caution It is recommended not to use illegal opcode because instructions may newly be assigned in the future.

(1) Operation

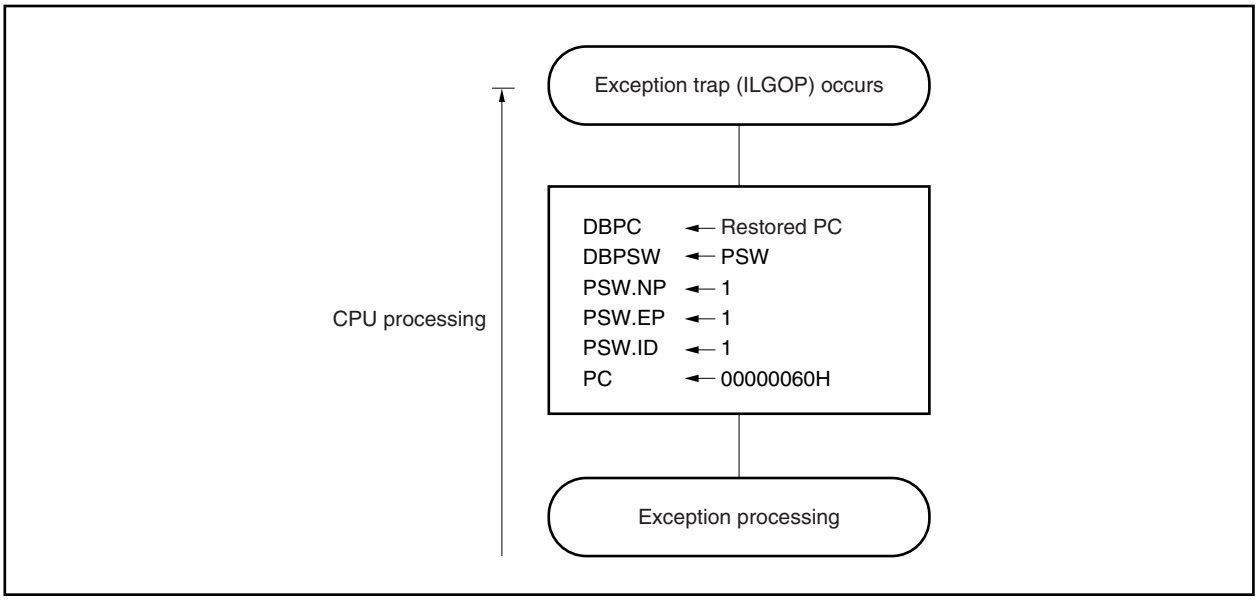
Upon generation of an exception trap, the CPU performs the following processing and transfers control to a handler routine.

<1> Saves the restored PC to DBPC.
<2> Saves the current PSW to DBPSW.
<3> Sets the PSW.NP, PSW.EP, and PSW.ID bits.
<4> Loads the handler address (00000060H) for the exception trap routine to the PC and transfers control.

Figure 17-10 shows the exception trap processing flow.

Downloaded from Arrow.com.

Figure 17-10. Exception Trap Processing



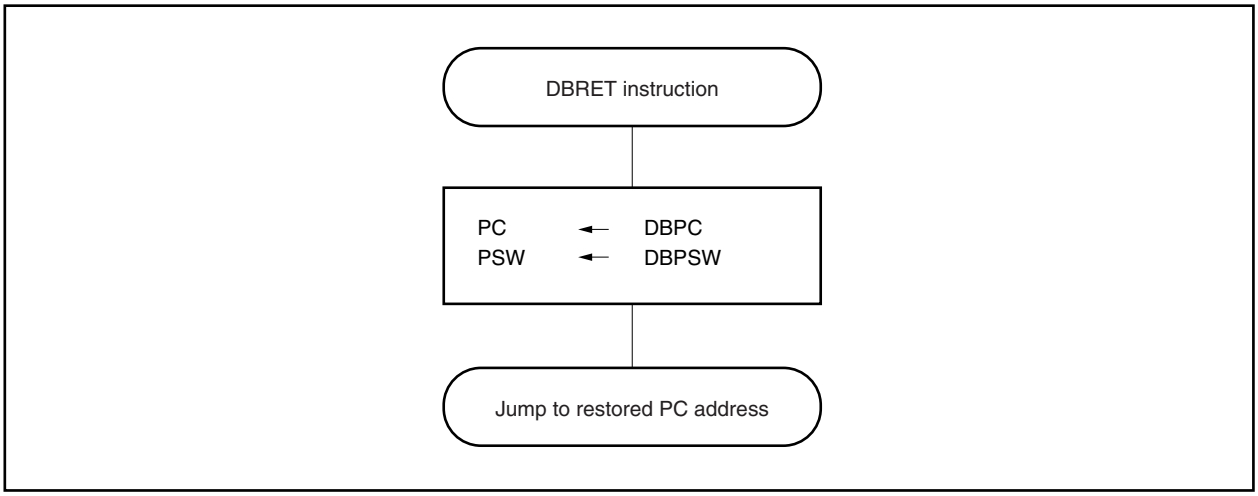
(2) Restore

Execution is restored from exception trap processing by the DBRET instruction. When the DBRET instruction is executed, the CPU performs the following processing and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.
<2> Transfers control to the loaded address of the restored PC and PSW.

Figure 17-11 shows the processing flow for restore from exception trap processing.

Figure 17-11. Processing Flow for Restore from Exception Trap



Downloaded from Arrow.com.

17.6.2 Debug trap

A debug trap is an exception that occurs upon execution of the DBTRAP instruction and that can be acknowledged at all times.

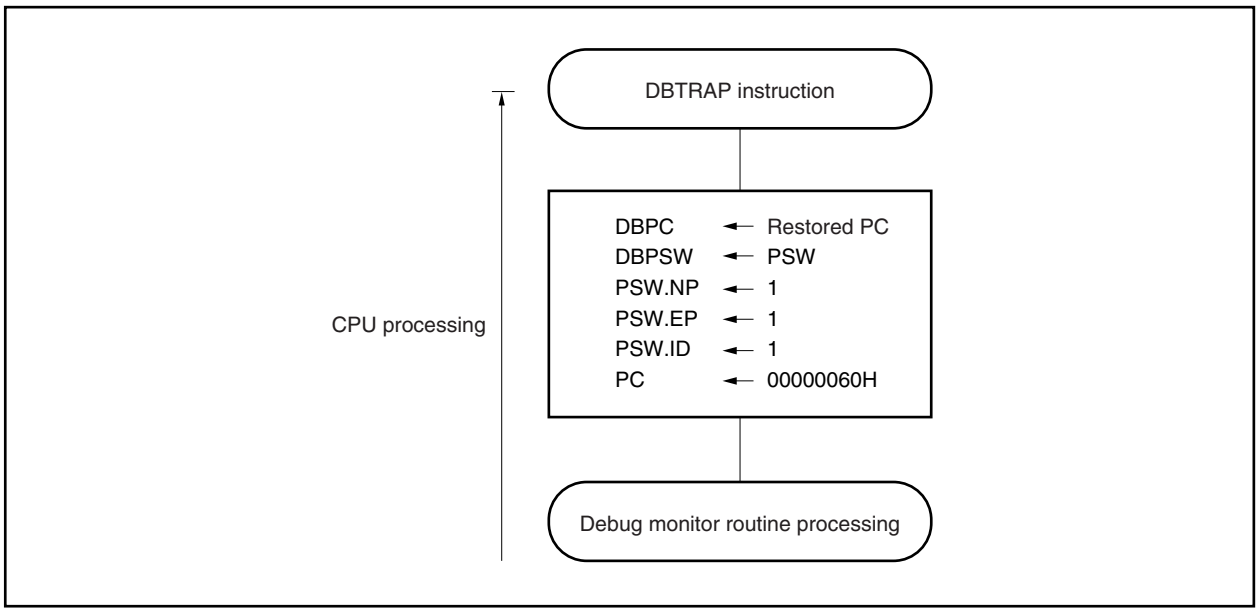
When a debug trap occurs, the CPU performs the following processing.

(1) Operation

<1> Saves the restored PC to DBPC.
<2> Saves the current PSW to DBPSW.
<3> Sets the PSW.NP, PSW.EP, and PSW.ID bits to 1.
<4> Sets the handler address (00000060H) for the debug trap routine to the PC and transfers control.

Figure 17-12 shows the debug trap processing flow.

Figure 17-12. Debug Trap Processing



Downloaded from Arrow.com.

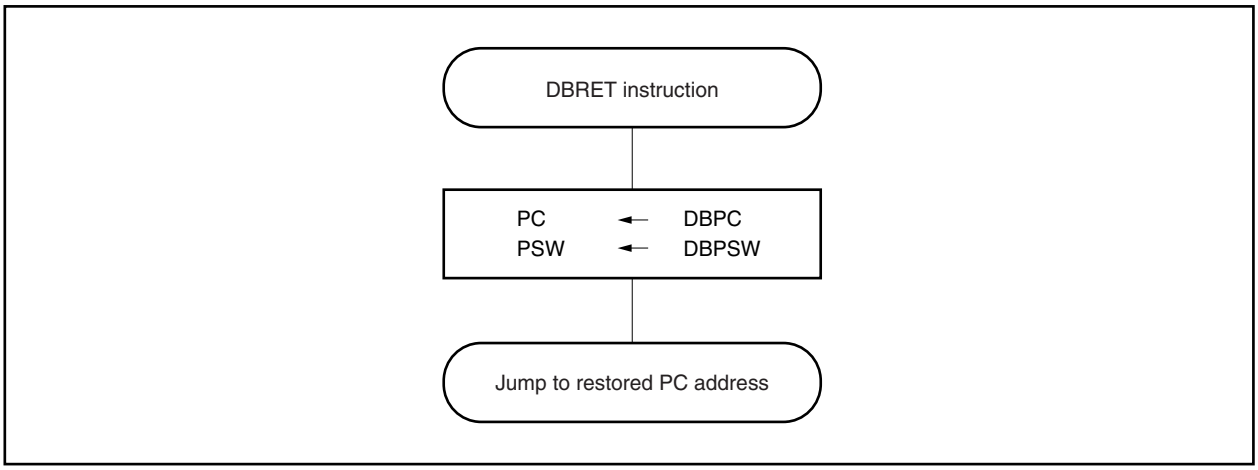
(2) Restore

Execution is restored from debug trap processing by the DBRET instruction. When the DBRET instruction is executed, the CPU performs the following processing and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.
<2> Transfers control to the loaded address of the restored PC and PSW.

Figure 17-13 shows the processing flow for restore from debug trap processing.

Figure 17-13. Processing Flow for Restore from Debug Trap



Downloaded from Arrow.com.

17.7 Multiple Interrupt Servicing Control

Multiple interrupt servicing control is a function that stops an interrupt service routine currently in progress if a higher priority interrupt request signal is generated, and processes the acknowledgment operation of the higher priority interrupt request signal.

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

Multiple interrupt servicing control is performed when interrupts are enabled (PSW.ID bit = 0). Even in an interrupt servicing routine, multiple interrupt control must be performed while interrupts are enabled (ID bit = 0).

If a maskable interrupt or software exception is generated in a maskable interrupt or software exception service program, EIPC and EIPSW must be saved.

The following example illustrates the procedure.

(1) To acknowledge maskable interrupt request signals in service program

Service program for maskable interrupt or exception

```
    …
    …
• EIPC saved to memory or register
• EIPSW saved to memory or register
• EI instruction (enables interrupt acknowledgment)
    …
    …                                  ←Acknowledges maskable interrupt
    …
    …
• DI instruction (disables interrupt acknowledgment)
• Saved value restored to EIPSW
• Saved value restored to EIPC
• RETI instruction
```


Downloaded from Arrow.com.

(2) To generate exception in service program

Service program for maskable interrupt or exception

…

…

- EIPC saved to memory or register
- EIPSW saved to memory or register

…

- TRAP instruction ←Acknowledges exceptions such as TRAP instruction.

…

- Saved value restored to EIPSW
- Saved value restored to EIPC
- RETI instruction

Priorities 0 to 7 (0 is the highest) can be set for each maskable interrupt request in multiple interrupt servicing control by software. To set a priority level, write values to the xxICn.xxPRn0 to xxICn.xxPRn2 bits corresponding to each maskable interrupt request. After reset, interrupt requests are masked by the xxICn.xxMKn bit, and the priority is set to level 7 by the xxPRn0 to xxPRn2 bits.

Priorities of maskable interrupts are as follows.

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

Interrupt servicing that has been suspended as a result of multiple interrupt servicing control is resumed after the interrupt servicing of the higher priority has been completed and the RETI instruction has been executed.
A pending interrupt request signal is acknowledged after the current interrupt servicing has been completed and the RETI instruction has been executed.

Caution In a non-maskable interrupt servicing routine (in the time until the RETI instruction is executed), maskable interrupts are not acknowledged and held pending.

Downloaded from Arrow.com.

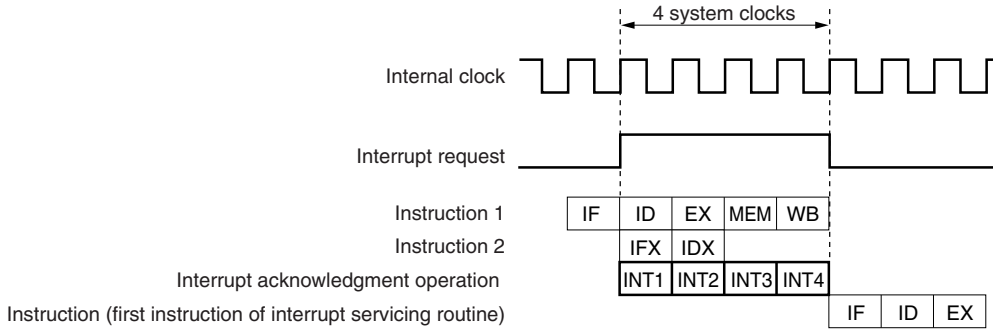
17.8 Interrupt Response Time

Except in the following cases, the CPU interrupt response time is a minimum of 4 clocks. If inputting consecutive interrupt request signals, at least 4 clocks must be placed between each interrupt request signal.

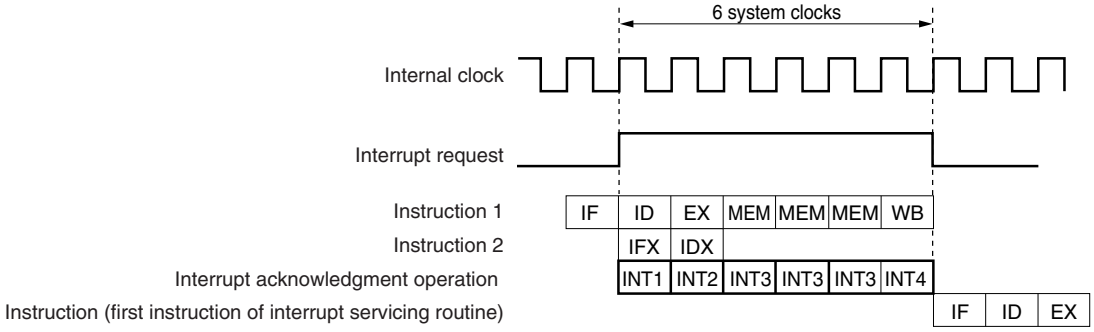
- IDLE/STOP mode
- External bus access
- Consecutive interrupt request non-sample instruction (refer to **17.9 Periods in Which Interrupts Are Not Acknowledged by CPU**)
- Access to interrupt control register
- Access to peripheral I/O register

Figure 17-14. Pipeline Operation During Interrupt Request Signal Acknowledgment (Outline)

(1) Minimum interrupt response time



(2) Maximum interrupt response time



Remark INT1 to INT4: Interrupt acknowledgment processing
IFX: Invalid instruction fetch
IDX: Invalid instruction decode

Interrupt response time (internal system clock)			Condition
	Internal interrupt	External interrupt	
Min.	4	4 + analog delay	The following cases are excluded. • IDLE/STOP mode • External bus access • Consecutive interrupt request non-sample instruction • Access to interrupt control register • Access to peripheral I/O register
Max.	6	6 + analog delay	

Downloaded from Arrow.com.

17.9 Periods in Which Interrupts Are Not Acknowledged by CPU

Interrupts are acknowledged by the CPU while an instruction is being executed. However, no interrupt is acknowledged between an interrupt request non-sample instruction and the next instruction (interrupts are held pending).

The following instructions are interrupt request non-sample instructions.

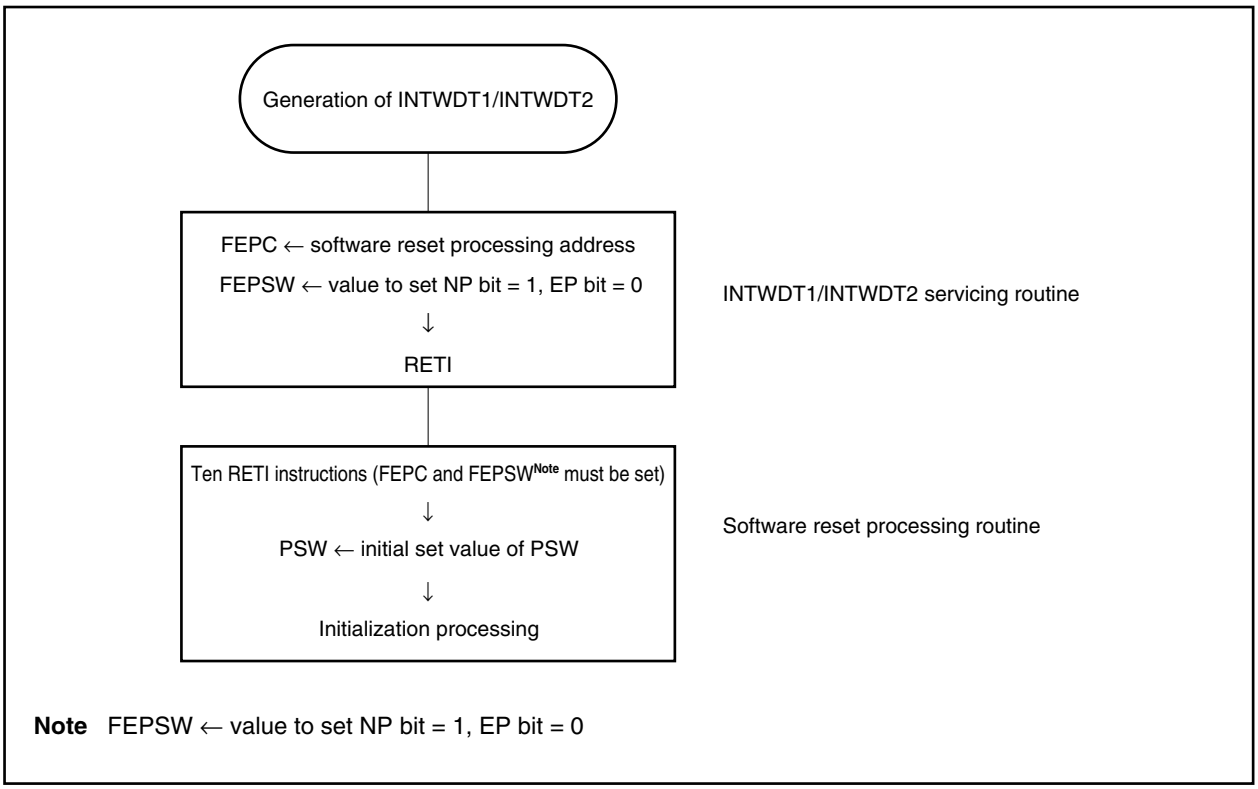
- EI instruction
- DI instruction
- LDSR reg2, 0x5 instructions (vs. PSW)
- Store instruction for the PRCMD register
- Store instruction and SET1, NOT1, and CLR1 instructions for the following registers
  - Interrupt-related registers:
    Interrupt control register (xxICn), interrupt mask registers 0, 1, 3 (IMR0, IMR1, IMR3)
  - Power save control register (PSC)

17.10 Cautions

Design the system so that restoring by the RETI instruction is as follows after a non-maskable interrupt triggered by a non-maskable interrupt request signal (INTWDT1/INTWDT2) is serviced.

Figure 17-15. Restoring by RETI Instruction

<R>



<R> **Note** FEPSW ← value to set NP bit = 1, EP bit = 0

Downloaded from Arrow.com.

CHAPTER 18 KEY INTERRUPT FUNCTION

18.1 Function

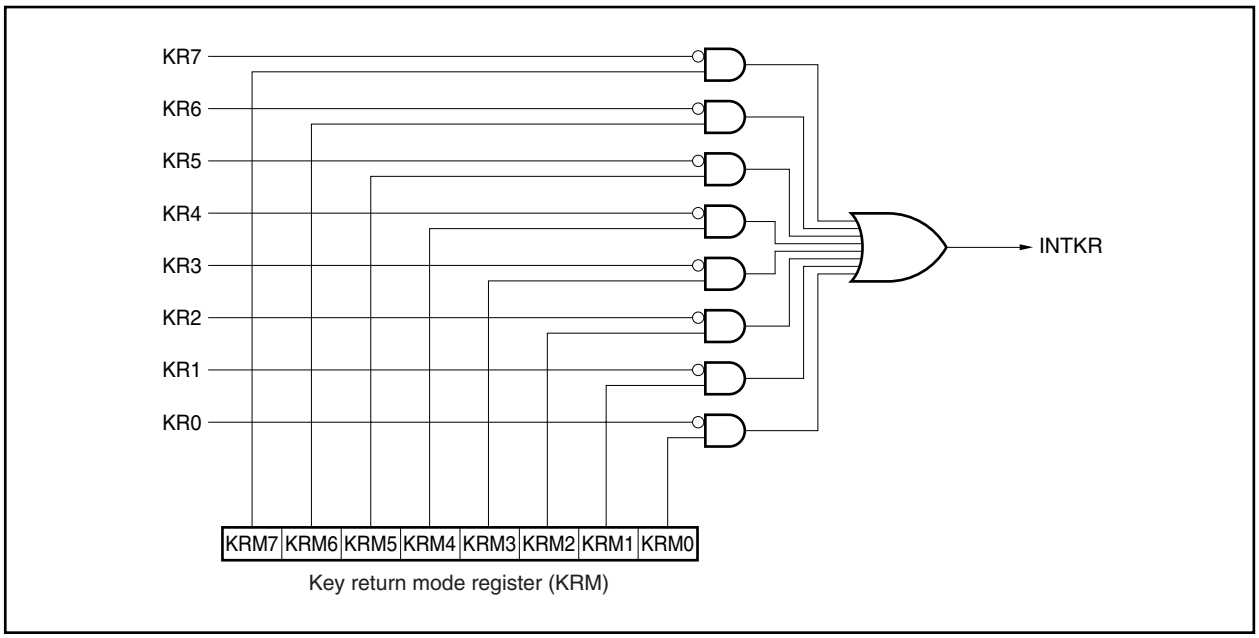
A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the eight key input pins (KR0 to KR7) by setting the KRM register.

Caution If any of the KR0 to KR7 pins is at low level, the INTKR signal is not generated even if a falling edge is input to another pin.

Table 18-1. Assignment of Key Return Detection Pins

Flag	Pin Description
KRM0	Controls KR0 signal in 1-bit units
KRM1	Controls KR1 signal in 1-bit units
KRM2	Controls KR2 signal in 1-bit units
KRM3	Controls KR3 signal in 1-bit units
KRM4	Controls KR4 signal in 1-bit units
KRM5	Controls KR5 signal in 1-bit units
KRM6	Controls KR6 signal in 1-bit units
KRM7	Controls KR7 signal in 1-bit units

Figure 18-1. Key Return Block Diagram



Downloaded from Arrow.com.

18.2 Register

(1) Key return mode register (KRM)

The KRM register controls the KRM0 to KRM7 bits using the KR0 to KR7 signals.

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

Reset sets KRM to 00H.

After reset: 00H R/W Address: FFFFF300H

	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Key return mode control
0	Does not detect key return signal
1	Detects key return signal

Caution If the KRM register is changed, an interrupt request signal (INTKR) may be generated. To prevent this, change the KRM register after disabling interrupts (DI), and then enable interrupts (EI) after clearing the interrupt request flag (KRIC.KRIF bit) to 0.

Remark For the alternate-function pin settings, refer to **Table 4-12 Settings When Port Pins Are Used for Alternate Functions**.

Downloaded from Arrow.com.

CHAPTER 19 STANDBY FUNCTION

19.1 Overview

The power consumption of the system can be effectively reduced by using the standby modes in combination and selecting the appropriate mode for the application.

The available standby modes are listed in Table 19-1.

Table 19-1. Standby Modes

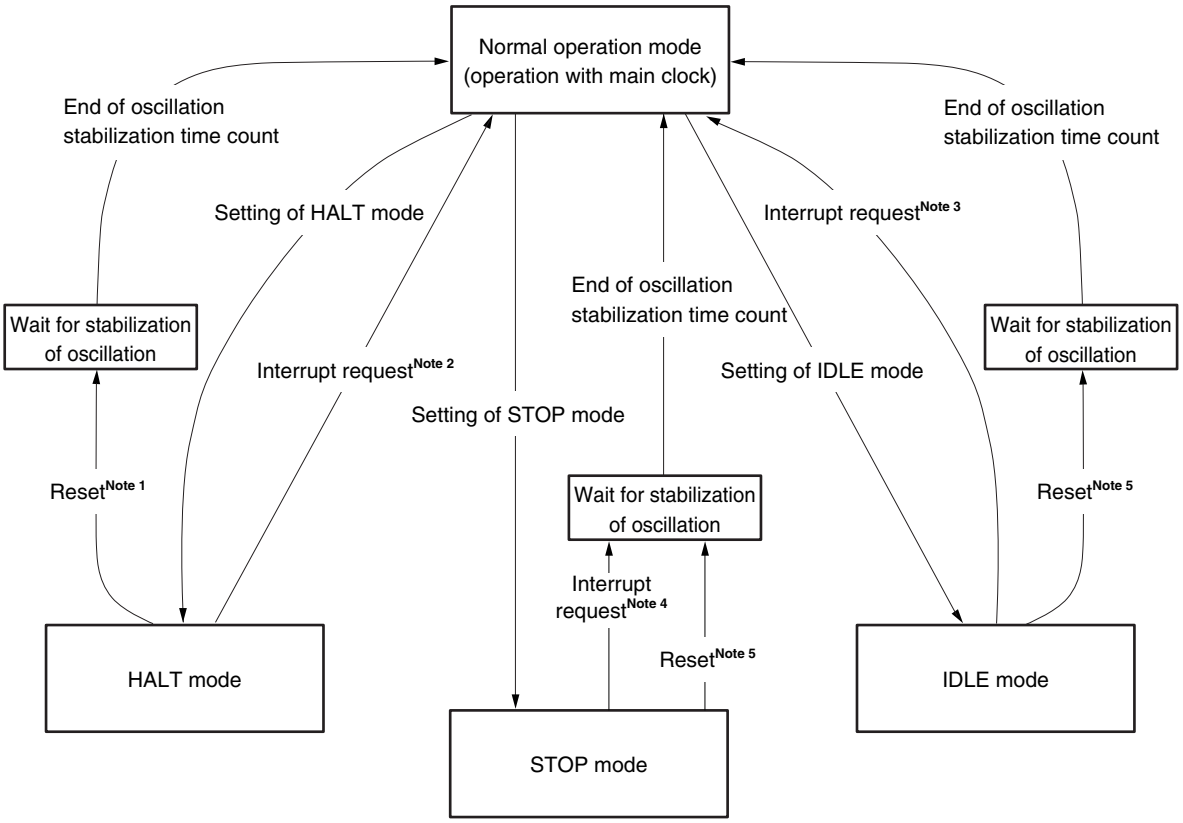
Mode	Functional Outline
HALT mode	Mode to stop only the operating clock of the CPU
IDLE mode	Mode to stop all the operations of the internal circuits except the oscillator[Note 1]
STOP mode	Mode to stop all the operations of the internal circuits except the subclock oscillator[Note 2]
Subclock operation mode	Mode to use the subclock as the internal system clock
Sub-IDLE mode	Mode to stop all the operations of the internal circuits, except the oscillator, in the subclock operation mode

Notes 1. The PLL does not stop. To realize low power consumption, stop the PLL and then shift to the IDLE mode.

2. Change to the clock-through mode, stop the PLL, then shift to the STOP mode. For details, refer to **CHAPTER 5 CLOCK GENERATION FUNCTION**.

Downloaded from Arrow.com.

Figure 19-1. Status Transition (1/2)



Notes 1. Reset by $\overline{\text{RESET}}$ pin input, watchdog timer 1 overflow (WDTRES1), or watchdog timer 2 overflow (WDTRES2).

2. Non-maskable interrupt request signal (NMI, INTWDT1, INTWDT2) or unmasked maskable interrupt request signal.

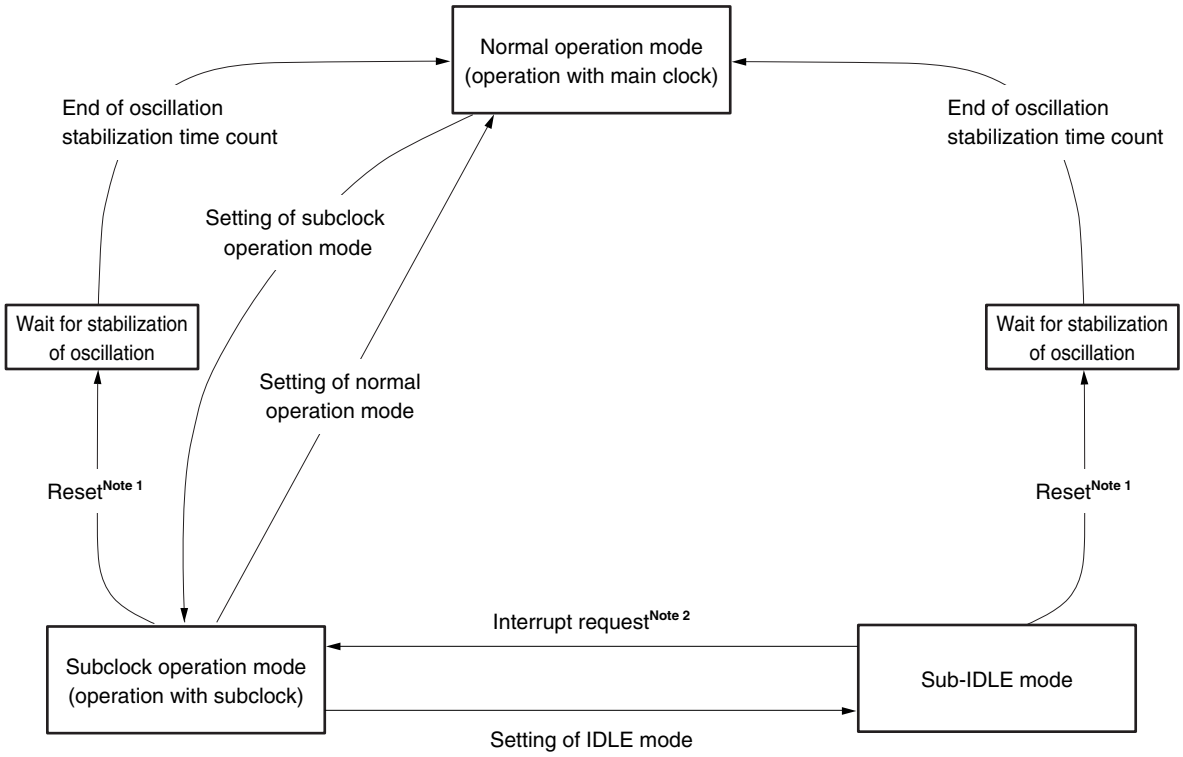
3. Non-maskable interrupt request signal (NMI pin input, INTWDT2 (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), or unmasked internal interrupt request signal from peripheral functions operable in IDLE mode.

4. Non-maskable interrupt request signal (NMI pin input, INTWDT2 (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), or unmasked internal interrupt request signal from peripheral functions operable in STOP mode.

5. Reset by $\overline{\text{RESET}}$ pin input or watchdog timer 2 (when the CPU is operating on the subclock) overflow (WDTRES2).

Downloaded from Arrow.com.

Figure 19-1. Status Transition (2/2)



Notes 1. Reset by $\overline{\text{RESET}}$ pin input or watchdog timer 2 overflow (WDTRES2).

2. Non-maskable interrupt request signal (NMI pin input, INTWDT2 (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), or unmasked internal interrupt request signal from peripheral functions operable in sub-IDLE mode.

Downloaded from Arrow.com.

19.2 Registers

(1) Power save control register (PSC)

This is an 8-bit register that controls the standby function. The STP bit of this register is used to specify the standby mode. The PSC register is a special register that can be written to only in a special sequence (refer to **3.4.7 Special registers**).

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

Reset sets PSC to 00H.

After reset: 00H R/W Address: FFFFF1FEH

	<7>	6	<5>	<4>	3	2	<1>	0
PSC	NMI2M	0	NMI0M	INTM	0	0	STP	0

NMI2M	Control of releasing standby mode[Note] by INTWDT2 signal
0	Releasing standby mode[Note] by INTWDT2 signal enabled
1	Releasing standby mode[Note] by INTWDT2 signal disabled

NMI0M	Control of releasing standby mode[Note] by NMI pin input
0	Releasing standby mode[Note] by NMI pin input enabled
1	Releasing standby mode[Note] by NMI pin input disabled

INTM	Control of releasing standby mode[Note] by maskable interrupt request signals
0	Releasing standby mode[Note] by maskable interrupt request signals enabled
1	Releasing standby mode[Note] by maskable interrupt request signals disabled

STP	Standby mode[Note] setting
0	Normal mode
1	Standby mode[Note]

Note In this case, standby mode means the IDLE/STOP mode; it does not include the HALT mode.

Cautions 1. If the NMI2M, NMI0M, and INTM bits, and the STP bit are set to 1 at the same time, the setting of NMI2M, NMI0M, and INTM bits becomes invalid. If there is an unmasked interrupt request signal being held pending when the IDLE/STOP mode is set, set the bit corresponding to the interrupt request signal (NMI2M, NMI0M, or INTM) to 1, and then set the STP bit to 1.

2. When the IDLE/STOP mode is set, set the PSMR.PSM bit and then set the STP bit.

Downloaded from Arrow.com.

(2) Power save mode register (PSMR)

This is an 8-bit register that controls the operation status in the standby mode and the clock operation.

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

Reset sets PSMR to 00H.

After reset: 00H R/W Address: FFFFF820H

	7	6	5	4	3	2	1	<0>
PSMR	XTSTP	0	0	0	0	0	0	PSM

XTSTP	Specification of subclock oscillator use
0	Subclock oscillator used
1	Subclock oscillator not used

PSM	Specification of operation in standby mode
0	IDLE mode
1	STOP mode

Cautions 1. Be sure to clear the XTSTP bit to 0 during subclock resonator connection.

2. Be sure to clear bits 1 to 6 of the PSMR register to 0.

3. The PSM bit is valid only when the PSC.STP bit is 1.

Downloaded from Arrow.com.

(3) Oscillation stabilization time selection register (OSTS)

The wait time until the oscillation stabilizes after the STOP mode is released is controlled by the OSTS register.

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

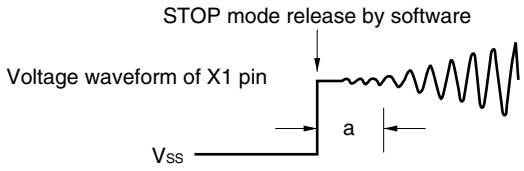
Reset sets OSTS to 01H.

After reset: 01H R/W Address: FFFFF6C0H

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

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time			
				fx		
				4 MHz	5 MHz	10 MHz
0	0	0	$2^{13}/f_x$	2.048 ms	1.638 ms	0.819 ms
0	0	1	$2^{15}/f_x$	8.192 ms	6.554 ms	3.277 ms
0	1	0	$2^{16}/f_x$	16.38 ms	13.11 ms	6.554 ms
0	1	1	$2^{17}/f_x$	32.77 ms	26.21 ms	13.11 ms
1	0	0	$2^{18}/f_x$	65.54 ms	52.43 ms	26.21 ms
1	0	1	$2^{19}/f_x$	131.1 ms	104.9 ms	52.43 ms
1	1	0	$2^{20}/f_x$	262.1 ms	209.7 ms	104.9 ms
1	1	1	$2^{21}/f_x$	524.3 ms	419.4 ms	209.7 ms

Cautions 1. The wait time following release of the STOP mode does not include the time until the clock oscillation starts (“a” in the figure below) following release of the STOP mode, regardless of whether the STOP mode is released by reset or the occurrence of an interrupt request signal.



2. Be sure to clear bits 3 to 7 to “0”.

3. The oscillation stabilization time following reset release is $2^{15}/f_x$ (because the initial value of the OSTS register = 01H).

4. The oscillation stabilization time is also inserted during external clock input.

Remark fx: Main clock oscillation frequency

Downloaded from Arrow.com.

19.3 HALT Mode

19.3.1 Setting and operation status

The HALT mode is set when a dedicated instruction (HALT) is executed in the normal operation mode.

In the HALT mode, the clock oscillator continues operating. Only clock supply to the CPU is stopped; clock supply to the other on-chip peripheral functions continues.

As a result, program execution is stopped, and the internal RAM retains the contents before the HALT mode was set. The on-chip peripheral functions that are independent of instruction processing by the CPU continue operating.

Table 19-3 shows the operation status in the HALT mode.

The average power consumption of the system can be reduced by using the HALT mode in combination with the normal operation mode for intermittent operation.

Cautions 1. Insert five or more NOP instructions after the HALT instruction.

2. If the HALT instruction is executed with an unmasked interrupt request signal held pending, the system shifts to the HALT mode, but the HALT mode is immediately released by the pending interrupt request signal.

19.3.2 Releasing HALT mode

The HALT mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT1, INTWDT2 signal), an unmasked maskable interrupt request signal, and reset signal ($\overline{\text{RESET}}$ pin input, WDTRES1, WDTRES2 signal).

After the HALT mode has been released, the normal operation mode is restored.

(1) Releasing HALT mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The HALT mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request. If the HALT mode is set in an interrupt servicing routine, however, an interrupt request that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the HALT mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the HALT mode is released and that interrupt request signal is acknowledged.

Table 19-2. Operation After Releasing HALT Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed	The next instruction is executed

(2) Releasing HALT mode by reset

The same operation as the normal reset operation is performed.

Downloaded from Arrow.com.

Table 19-3. Operation Status in HALT Mode

Setting of HALT Mode / Item		When CPU Is Operating with Main Clock	
		When Subclock Is Not Used	When Subclock Is Used
CPU		Stops operation	
ROM correction		Stops operation	
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Interrupt controller		Operable	
Timer P (TMP0)		Operable	
16-bit timer (TM01)		Operable	
8-bit timers (TM50, TM51)		Operable	
Timer H (TMH0, TMH1)		Operable	
Watch timer		Operable when main clock is selected as count clock	Operable
Watchdog timer 1		Operable	
Watchdog timer 2		Operable when main clock is selected as count clock	Operable
Serial interface	CSI00, CSI01	Operable	
	I^2C0[Note]	Operable	
	UART0, UART1	Operable	
Key interrupt function		Operable	
A/D converter		Operable	
Real-time output		Operable	
Port function		Retains status before HALT mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the HALT mode was set.	

Note Only in the μPD703207Y, 70F3207HY

Downloaded from Arrow.com.

19.4 IDLE Mode

19.4.1 Setting and operation status

The IDLE mode is set by clearing the PSMR.PSM bit to 0 and setting the PSC.STP bit to 1 in the normal operation mode.

In the IDLE mode, the clock oscillator continues operation but clock supply to the CPU and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE mode was set are retained. The CPU and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 19-5 shows the operation status in the IDLE mode.

The IDLE mode can reduce the power consumption more than the HALT mode because it stops the operation of the on-chip peripheral functions. The main clock oscillator does not stop, so the normal operation mode can be restored without waiting for the oscillation stabilization time after the IDLE mode has been released, in the same manner as when the HALT mode is released.

Caution Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE mode.

Downloaded from Arrow.com.

19.4.2 Releasing IDLE mode

The IDLE mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the IDLE mode, or reset ($\overline{\text{RESET}}$ pin input, WDTRES2 signal (when the CPU is operating on the subclock)).

After the IDLE mode has been released, the normal operation mode is restored.

(1) Releasing IDLE mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The IDLE mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request. If the IDLE mode is set in an interrupt servicing routine, however, an interrupt request that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the IDLE mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE mode is released and that interrupt request signal is acknowledged.

Table 19-4. Operation After Releasing IDLE Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed	The next instruction is executed

Caution The interrupt request signal that is disabled by setting the PSC.NMI2M, PSC.NMI0M, and PSC.INTM bits to 1 (interrupt disabled) becomes invalid and the IDLE mode is not released.

Downloaded from Arrow.com.

(2) Releasing IDLE mode by reset

The same operation as the normal reset operation is performed.

Table 19-5. Operation Status in IDLE Mode

Setting of IDLE Mode / Item		When CPU Is Operating with Main Clock	
		When Subclock Is Not Used	When Subclock Is Used
CPU		Stops operation	
ROM correction		Stops operation	
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Interrupt controller		Stops operation	
Timer P (TMP0)		Stops operation	
16-bit timer (TM01)		Operable when INTWT is selected as count clock and f_{BRG} is selected as count clock of WT	Operable when INTWT is selected as count clock
8-bit timers (TM50, TM51)		• Operable when TI5n is selected as count clock • Operable when INTTM010 is selected as count clock and TM01 is enabled in IDLE mode	
Timer H (TMH0)		Stops operation	
Timer H (TMH1)		Stops operation	Operable when f_{XT} is selected as count clock
Watch timer		Operable when main clock is selected as count clock	Operable
Watchdog timer 1		Stops operation	
Watchdog timer 2		Stops operation	Operable when f_{XT} is selected as count clock
Serial interface	CSI00, CSI01	Operable when $\overline{SCK0n}$ input clock is selected as operation clock	
	I^2C0 Note 1	Stops operation	
	UART0	Operable when ASCK0 is selected as count clock	
	UART1	Stops operation	
Key interrupt function		Operable	
A/D converter		Stops operation Note 2	
Real-time output		Operable when INTTM5n is selected as real-time output trigger and TM5n is enabled in IDLE mode. However, the RTBH0 and RTBL0 registers cannot be updated because the CPU is stopped.	
Port function		Retains status before IDLE mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE mode was set.	

Notes 1. Only in the μPD703207Y, 70F3207HY

2. Set the ADM.ADCS and ADM.ADCS2 bits to 00B.

Remark n = 0, 1

Downloaded from Arrow.com.

19.5 STOP Mode

19.5.1 Setting and operation status

The STOP mode is set when the PSMR.PSM bit is set to 1 and the PSC.STP bit is set to 1 in the normal operation mode.

In the STOP mode, the subclock oscillator continues operating but the main clock oscillator stops. Clock supply to the CPU and the on-chip peripheral functions is stopped.

As a result, program execution is stopped, and the contents of the internal RAM before the STOP mode was set are retained. The CPU and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock oscillator or an external clock continue operating.

Table 19-7 shows the operation status in the STOP mode.

Because the STOP mode stops operation of the main clock oscillator, it reduces the power consumption to a level lower than the IDLE mode. If the subclock oscillator and external clock are not used, the power consumption can be minimized with only leakage current flowing.

Caution Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the STOP mode.

Downloaded from Arrow.com.

19.5.2 Releasing STOP mode

The STOP mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the STOP mode, or reset ($\overline{\text{RESET}}$ pin input, WDTRES2 signal (when the CPU is operating on the subclock)).

After the STOP mode has been released, the normal operation mode is restored after the oscillation stabilization time has been secured.

(1) Releasing STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The STOP mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request. If the STOP mode is set in an interrupt servicing routine, however, an interrupt request that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the STOP mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the STOP mode is released and that interrupt request signal is acknowledged.

Table 19-6. Operation After Releasing STOP Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed	The next instruction is executed

Caution The interrupt request signal that is disabled by setting the PSC.NMI2M, PSC.NMI0M, and PSC.INTM bits to 1 (interrupt disabled) becomes invalid and the STOP mode is not released.

(2) Releasing STOP mode by reset

The same operation as the normal reset operation is performed.

Downloaded from Arrow.com.

Table 19-7. Operation Status in STOP Mode

Item \ Setting of STOP Mode		When CPU Is Operating with Main Clock	
		When Subclock Is Not Used	When Subclock Is Used
CPU		Stops operation	
ROM correction		Stops operation	
Main clock oscillator		Oscillation stops	
Subclock oscillator		–	Oscillation enabled
Interrupt controller		Stops operation	
Timer P (TMP0)		Stops operation	
16-bit timer (TM01)		Stops operation	Operable when INTWT is selected as count clock and f_{XT} is selected as count clock of WT
8-bit timers (TM50, TM51)		Operable when TI5n is selected as count clock	Operable when TI5n is selected as count clock or when INTTM010 is selected as count clock and TM01 is enabled in STOP mode
Timer H (TMH0)		Stops operation	
Timer H (TMH1)		Stops operation	Operable when f_{XT} is selected as count clock
Watch timer		Stops operation	Operable when f_{XT} is selected as count clock
Watchdog timer 1		Stops operation	
Watchdog timer 2		Stops operation	Operable when f_{XT} is selected as count clock
Serial interface	CSI00, CSI01	Operable when $\overline{SCK0n}$ input clock is selected as operation clock	
	I^2C0 Note 1	Stops operation	
	UART0	Operable when ASCK0 is selected as count clock	
	UART1	Stops operation	
Key interrupt function		Operable	
A/D converter		Stops operation Note 2	
Real-time output		Operable when INTTM5n is selected as real-time output trigger and TM5n is enabled in STOP mode. However, the RTBH0 and RTBL0 registers cannot be updated because the CPU is stopped.	
Port function		Retains status before STOP mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the STOP mode was set.	

Notes 1. Only in the μPD703207Y, 70F3207HY

2. Set the ADM.ADCS and ADM.ADCS2 bits to 00B.

Remark n = 0, 1

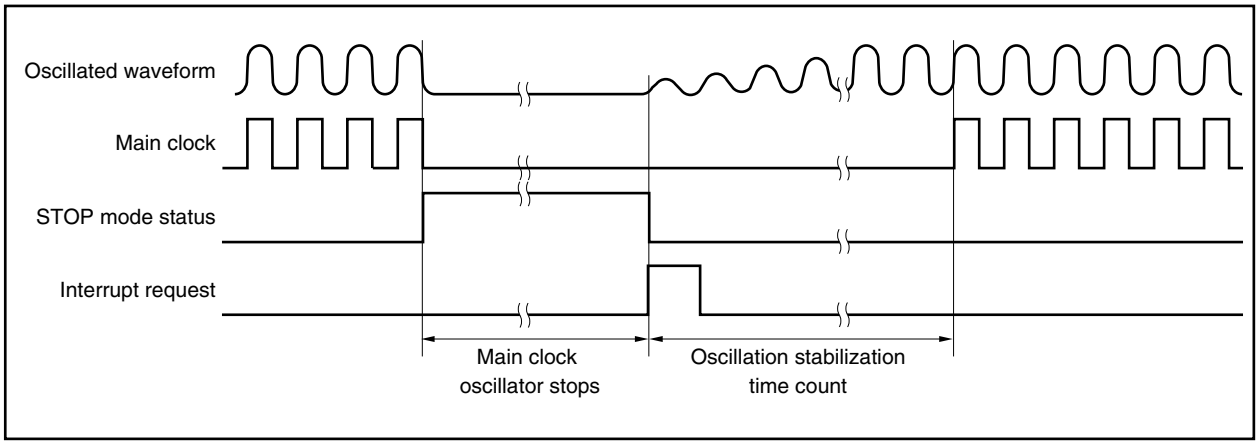
Downloaded from Arrow.com.

19.5.3 Securing oscillation stabilization time when STOP mode is released

When the STOP mode is released, only the oscillation stabilization time set by the OSTS register elapses. If the STOP mode has been released by reset, however, the reset value of the OSTS register, $2^{15}/fx$ (8.192 ms at fx = 4 MHz) elapses.

The operation performed when the STOP mode is released by an interrupt request signal is shown below.

Figure 19-2. Oscillation Stabilization Time



Caution For details of the OSTS register, refer to 19.2 (3) Oscillation stabilization time selection register (OSTS).

Downloaded from Arrow.com.

19.6 Subclock Operation Mode

19.6.1 Setting and operation status

The subclock operation mode is set when the PCC.CK3 bit is set to 1 in the normal operation mode.

When the subclock operation mode is set, the internal system clock is changed from the main clock to the subclock.

When the PCC.MCK bit is set to 1, the operation of the main clock oscillator is stopped. As a result, the system operates only with the subclock.

Table 19-8 shows the operation status in subclock operation mode.

In the subclock operation mode, the power consumption can be reduced to a level lower than in the normal operation mode because the subclock is used as the internal system clock. In addition, the power consumption can be further reduced to the level of the STOP mode by stopping the operation of the main clock oscillator.

Cautions 1. When manipulating the CK3 bit, do not change the set values of the PCC.CK2 to PCC.CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details, refer to 5.3 (1) Processor clock control register (PCC).

2. If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied and set the subclock operation mode.

Internal system clock (f_{CLK}) > Subclock (f_{XT}: 32.768 kHz) × 4

Remark Internal system clock (f_{CLK}): Clock generated from the main clock (f_{XX}) by setting the CK2 to CK0 bits

19.6.2 Releasing subclock operation mode

The subclock operation mode is released when the CK3 bit is cleared to 0 or by reset ($\overline{\text{RESET}}$ pin input, WDTRES1, WDTRES2 signal). If the main clock is stopped (MCK bit = 1), set the MCK bit to 1, secure the oscillation stabilization time of the main clock by software, and clear the CK3 bit to 0.

The normal operation mode is restored when the subclock operation mode is released.

Caution When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details, refer to 5.3 (1) Processor clock control register (PCC).

Downloaded from Arrow.com.

Table 19-8. Operation Status in Subclock Operation Mode

Item \ Setting of Subclock Operation Mode		Operation Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
CPU		Operable	
ROM correction		Operable	
Subclock oscillator		Oscillation enabled	
Interrupt controller		Operable	
Timer P (TMP0)		Operable	Stops operation
16-bit timer (TM01)		Operable	Operable when INTWT is selected as count clock and f_{XT} is selected as count clock of WT
8-bit timers (TM50, TM51)		Operable	• Operable when TI5n is selected as count clock • Operable when INTTM010 is selected as count clock and when TM01 is enabled in subclock operation mode
Timer H (TMH0)		Operable	Stops operation
Timer H (TMH1)		Operable	Operable when f_{XT} is selected as count clock
Watch timer		Operable	Operable when f_{XT} is selected as count clock
<R> Watchdog timer 1		Stops operation	
Watchdog timer 2		Operable	Operable when f_{XT} is selected as count clock
Serial interface	CSI00, CSI01	Operable	Operable when $\overline{SCK0n}$ input clock is selected as operation clock
	I^2C0[Note]	Operable	Stops operation
	UART0	Operable	Operable when ASCK0 is selected as count clock
	UART1	Operable	Stops operation
Key interrupt function		Operable	
A/D converter		Operable	Stops operation
Real-time output		Operable	Operable when INTTM5n is selected as real-time output trigger and TI5n is selected as count clock of TM5n
Port function		Settable	
Internal data		Settable	

Note Only in the μPD703207Y, 70F3207HY

Remark n = 0, 1

Downloaded from Arrow.com.

19.7 Sub-IDLE Mode

19.7.1 Setting and operation status

The sub-IDLE mode is set when the PSMR.PSM bit is cleared to 0 and the PSC.STP bit is set to 1 in the subclock operation mode.

In this mode, the clock oscillator continues operation but clock supply to the CPU and the other on-chip peripheral functions is stopped.

As a result, program execution is stopped and the contents of the internal RAM before the sub-IDLE mode was set are retained. The CPU and the other on-chip peripheral functions are stopped. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 19-10 shows the operation status in the sub-IDLE mode.

Because the sub-IDLE mode stops operation of the CPU and other on-chip peripheral functions, it can reduce the power consumption more than the subclock operation mode. If the sub-IDLE mode is set after the main clock has been stopped, the power consumption can be reduced to a level as low as that in the STOP mode.

<R> **Caution Following the store instruction to set the PSC register to the sub-IDLE mode, insert the five or more NOP instructions.**

Downloaded from Arrow.com.

19.7.2 Releasing sub-IDLE mode

The sub-IDLE mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal (when the CPU is operating on the subclock)), unmasked external interrupt request signal (INTP0 to INTP6 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the sub-IDLE mode, or reset ($\overline{\text{RESET}}$ pin input, WDTRES2 signal (when the CPU is operating on the subclock)).

When the sub-IDLE mode is released by an interrupt request signal, the subclock operation mode is set. If it is released by reset, the normal operation mode is restored.

(1) Releasing sub-IDLE mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The sub-IDLE mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request. If the sub-IDLE mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, only the sub-IDLE mode is released, and that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the sub-IDLE mode is released and that interrupt request signal is acknowledged.

Table 19-9. Operation After Releasing Sub-IDLE Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed	The next instruction is executed

Caution The interrupt request signal that is disabled by setting the PSC.NMI2M, PSC.NMI0M, and PSC.INTM bits to 1 (interrupt disabled) becomes invalid and the sub-IDLE mode is not released.

(2) Releasing sub-IDLE mode by reset

The same operation as the normal reset operation is performed.

Downloaded from Arrow.com.

Table 19-10. Operation Status in Sub-IDLE Mode

Item \ Setting of Sub-IDLE Mode		Operation Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
CPU		Stops operation	
ROM correction		Stops operation	
Subclock oscillator		Oscillation enabled	
Interrupt controller		Stops operation	
Timer P (TMP0)		Stops operation	
16-bit timer (TM01)		Operable when INTWT is selected as count clock	Operable when INTWT is selected as count clock and f_{XT} is selected as count clock of WT
<R> 8-bit timers (TM50, TM51)		• Operable when TI5n is selected as count clock • Operable when INTTM010 is selected as count clock and when TM01 is enabled in sub-IDLE mode	
Timer H (TMH0)		Stops operation	
Timer H (TMH1)		Operable when f_{XT} is selected as count clock	
<R> Watch timer		Operable	Operable when f_{XT} is selected as count clock
<R> Watchdog timer 1		Stops operation	
Watchdog timer 2		Operable when f_{XT} is selected as count clock	
<R> Serial interface	CSI00, CSI01	Operable when $\overline{SCK0n}$ input clock is selected as operation clock	
	I^2C0 Note 1	Stops operation	
	UART0	Operable when ASCK0 is selected as count clock	
	UART1	Stops operation	
Key interrupt function		Operable	
A/D converter		Stops operation Note 2	
Real-time output		Operable when INTTM5n is selected as real-time output trigger and TM5n is set to the operable condition in sub-IDLE mode	
Port function		Retains status before sub-IDLE mode was set.	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the sub-IDLE mode was set.	

Notes 1. Only in the μPD703207Y, 70F3207HY

<R> **2.** Set the ADM.ADCS and ADM.ADCS2 bits to 00B.

Remark n = 0, 1

Downloaded from Arrow.com.

CHAPTER 20 RESET FUNCTION

20.1 Overview

The following reset functions are available.

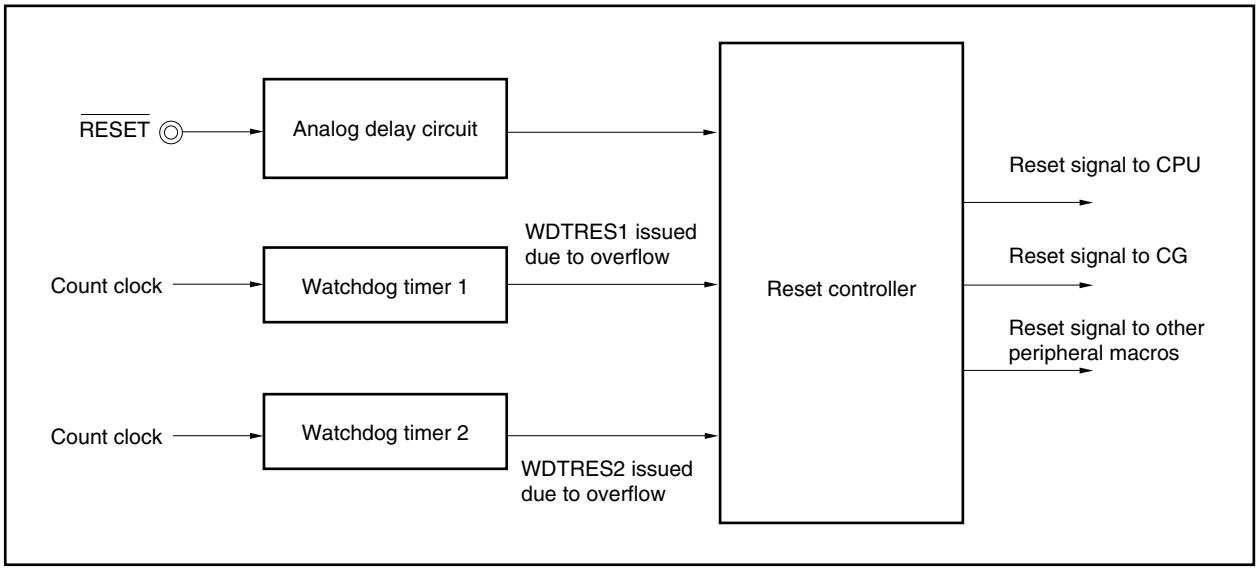
- Reset function by $\overline{\text{RESET}}$ pin input
- Reset function by overflow of watchdog timer 1 (WDTRES1)
- Reset function by overflow of watchdog timer 2 (WDTRES2)

If the $\overline{\text{RESET}}$ pin goes high, the reset status is released, and the CPU starts executing the program. Initialize the contents of each register in the program as necessary.

The $\overline{\text{RESET}}$ pin has a noise eliminator that operates by analog delay to prevent malfunction caused by noise.

20.2 Configuration

Figure 20-1. Reset Block Diagram



Downloaded from Arrow.com.

20.3 Operation

The system is reset, initializing each hardware unit, when a low level is input to the $\overline{\text{RESET}}$ pin or if watchdog timer 1 or watchdog timer 2 overflows (WDTRES1 or WDTRES2).

While a low level is being input to the $\overline{\text{RESET}}$ pin, the main clock oscillator stops. Therefore, the overall power consumption of the system can be reduced.

If the $\overline{\text{RESET}}$ pin goes high or if the WDTRES1 or WDTRES2 signal is received, the reset status is released.

If the reset status is released by $\overline{\text{RESET}}$ pin input or the WDTRES2 signal, the oscillation stabilization time elapses (reset value of OSTS register: $2^{15}/f_{XX}$) and then the CPU starts program execution.

If the reset status is released by the WDTRES1 signal, the oscillation stabilization time is not inserted because the main clock oscillator does not stop.

Downloaded from Arrow.com.

<R> **Table 20-1. Hardware Status on $\overline{\text{RESET}}$ Pin Input or Occurrence of WDTRES2 Signal**

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Peripheral clock (f_{XX} to f_{XX}/1024)	Operation stops	Operation starts after securing oscillation stabilization time
Internal system clock (f_{CLK})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
Watchdog timer 1 clock (f_{XW})	Operation stops	Operation starts
CPU	Initialized	Program execution starts after securing oscillation stabilization time
Internal RAM	Undefined if power-on reset or writing data to RAM (by CPU) and reset input conflict (data is damaged). Otherwise value immediately before reset input is retained.	
I/O lines	High impedance	
On-chip peripheral I/O registers	Initialized to specified status	
Watchdog timer 2	Operation stops	Operation starts after securing oscillation stabilization time
Other on-chip peripheral functions	Operation stops	Operation can be started after securing oscillation stabilization time

<R> **Table 20-2. Hardware Status on Occurrence of WDTRES1 Signal**

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation continues	
Subclock oscillator (f_{XT})	Oscillation continues	
Peripheral clock (f_{XX} to f_{XX}/1024)	Operation stops	Operation starts
Internal system clock (f_{CLK})	Oscillation continues (initialized to f_{XX}/8)	
CPU clock (f_{CPU})	Oscillation continues (initialized to f_{XX}/8)	
Watchdog timer 1 clock (f_{XW})	Operation continues	
Internal RAM	Undefined if writing data to RAM (by CPU) and reset input conflict (data is damaged). Otherwise value immediately before reset input is retained.	
I/O lines	High impedance	
On-chip peripheral I/O registers	Initialized to specified status	
Watchdog timer 2	Operation stops	Operation starts
Other on-chip peripheral functions	Operation stops	Operation can be started

Downloaded from Arrow.com.

Figure 20-2. Hardware Status on $\overline{\text{RESET}}$ Pin Input

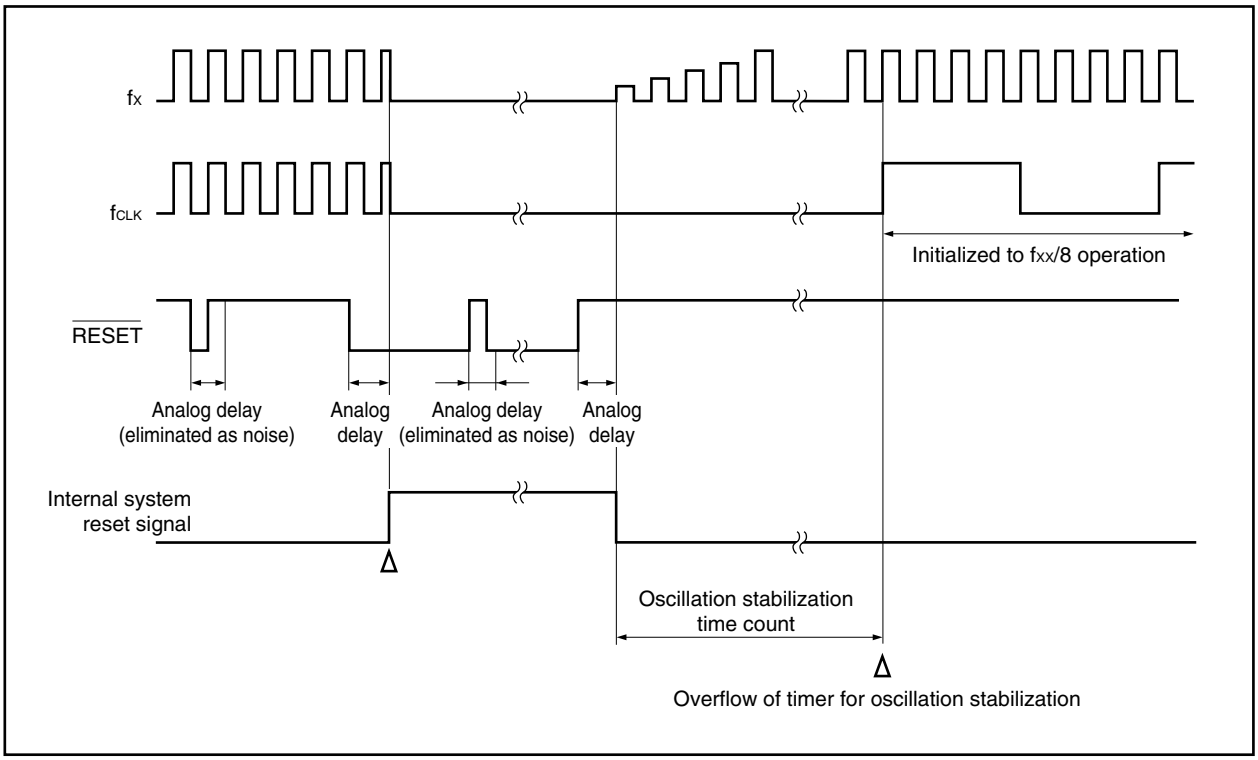
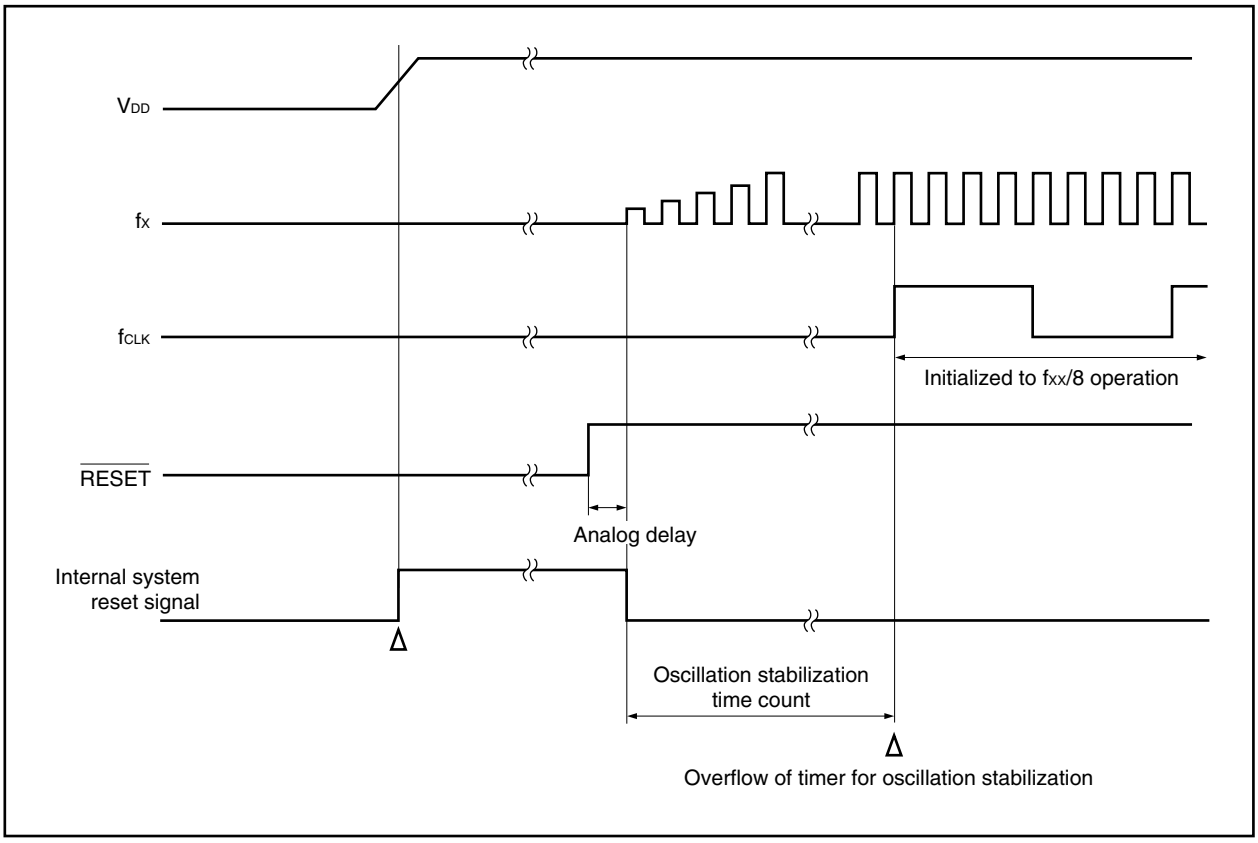
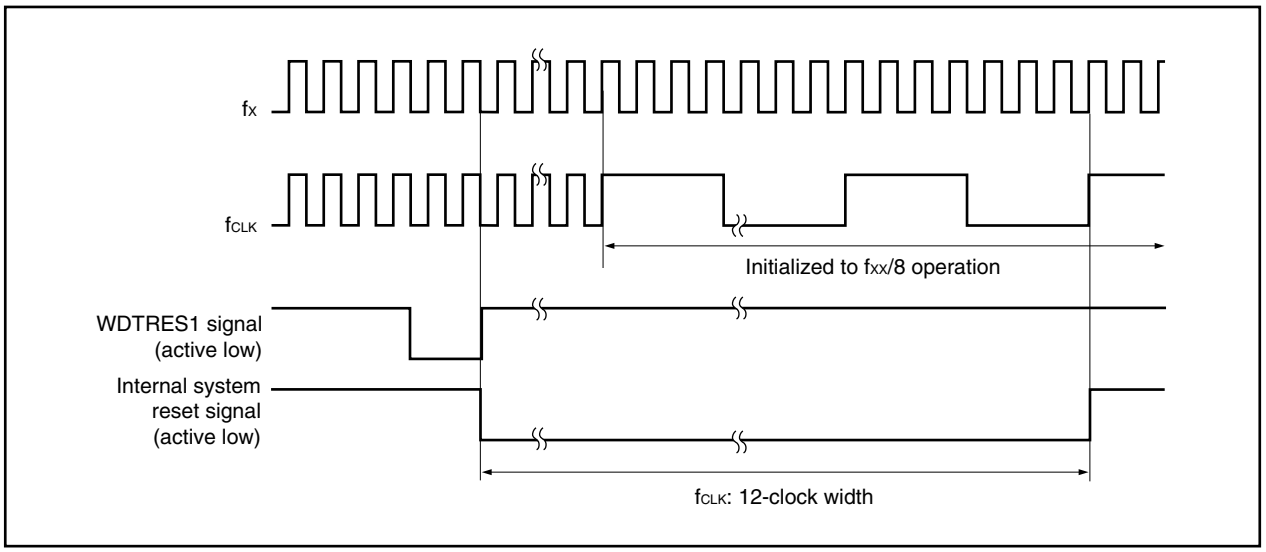


Figure 20-3. Operation on Power Application

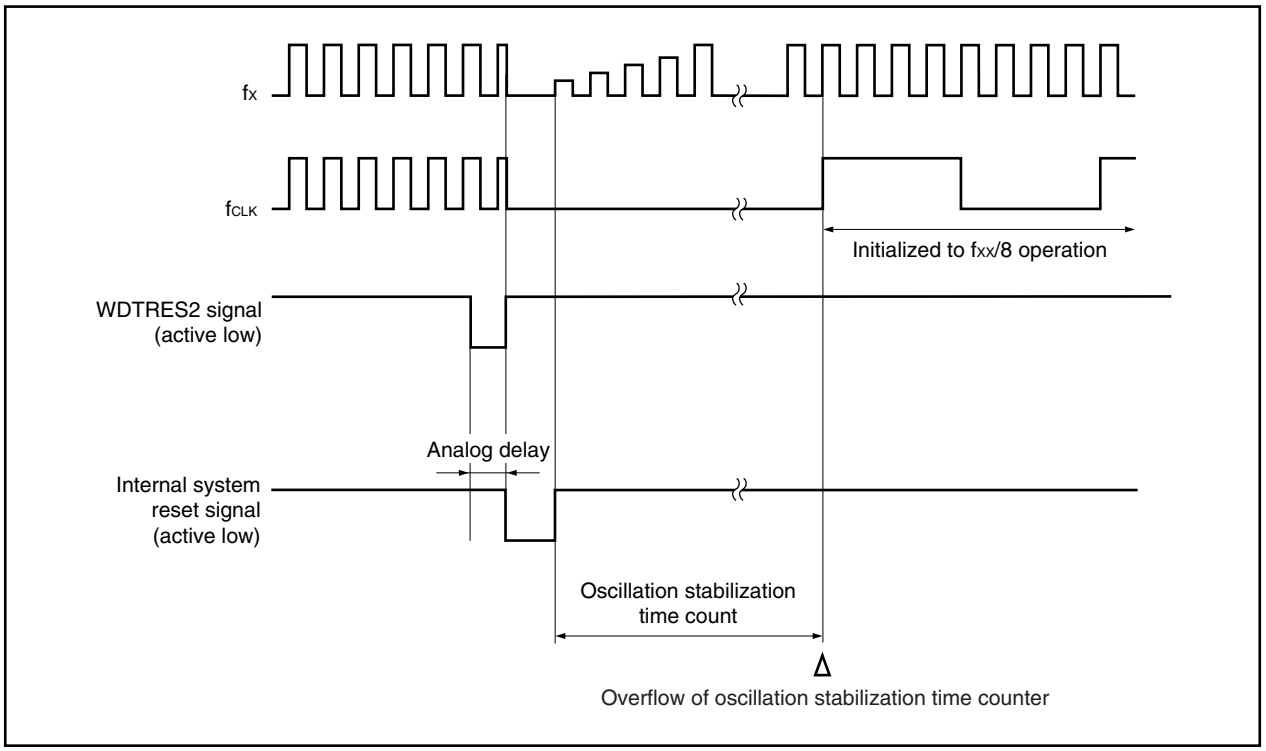


Downloaded from Arrow.com.

<R> **Figure 20-4. Timing of Reset Operation by Watchdog Timer 1**



<R> **Figure 20-5. Timing of Reset Operation by Watchdog Timer 2**



Downloaded from Arrow.com.

CHAPTER 21 ROM CORRECTION FUNCTION

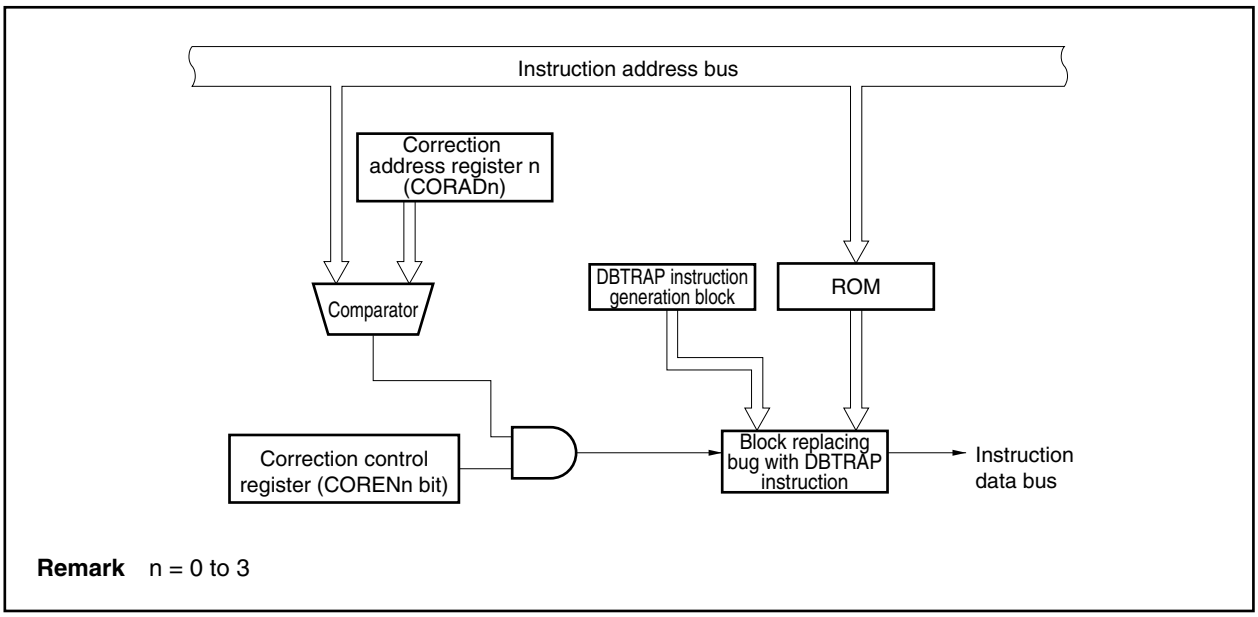
21.1 Overview

The ROM correction function is used to replace part of the program in the internal ROM with the program of an external memory or the internal RAM.

By using this function, program bugs found in the internal ROM can be corrected.

Up to four addresses can be specified for correction.

Figure 21-1. Block Diagram of ROM Correction



Remark n = 0 to 3

Downloaded from Arrow.com.

21.2 Registers

(1) Correction address registers 0 to 3 (CORAD0 to CORAD3)

These registers are used to set the first address of the program to be corrected.

The program can be corrected at up to four places because four CORADn registers are provided.

The CORADn register can be read or written in 32-bit units.

If the higher 16 bits of the CORADn register are used as the CORADnH register, and the lower 16 bits as the CORADnL register, these registers can be read or written in 16-bit units.

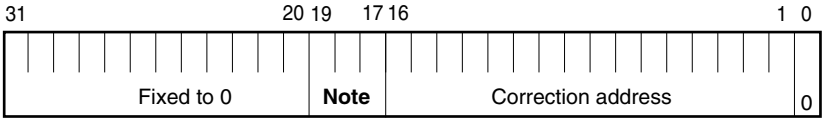
Reset sets CORADn to 00000000H.

Set correction addresses in the range of 0000000H to 001FFFEH in the V850ES/KE1.

Bits 0 and 20 to 31 are fixed to 0.

After reset: 00000000H R/W Address: CORAD0 FFFFF840H,
CORAD0L FFFFF840H, CORAD0H FFFFF842H,
CORAD1 FFFFF844H,
CORAD1L FFFFF844H, CORAD1H FFFFF846H,
CORAD2 FFFFF848H,
CORAD2L FFFFF848H, CORAD2H FFFFF84AH,
CORAD3 FFFFF84CH,
CORAD3L FFFFF84CH, CORAD3H FFFFF84EH

CORADn



(n = 0 to 3)

Note Be sure to clear these bits to 0.

Downloaded from Arrow.com.

(2) Correction control register (CORCN)

This register disables or enables the correction operation at the address specified by the CORADn register.

Each channel can be enabled or disabled by this register.

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

Reset sets CORCN to 00H.

After reset: 00H R/W Address: FFFFF880H

	7	6	5	4	<3>	<2>	<1>	<0>
CORCN	0	0	0	0	COREN3	COREN2	COREN1	COREN0

CORENn	Correction operation enable/disable
0	Disabled
1	Enabled

Remark n = 0 to 3

Table 21-1. Correspondence Between CORCN Register Bits and CORADn Registers

CORCN Register Bit	Corresponding CORADn Register
COREN3	CORAD3
COREN2	CORAD2
COREN1	CORAD1
COREN0	CORAD0

21.3 ROM Correction Operation and Program Flow

<1> If the address to be corrected and the fetch address of the internal ROM match, the fetch code is replaced by the DBTRAP instruction.

<2> When the DBTRAP instruction is executed, execution branches to address 00000060H.

<3> Software processing after branching causes the result of ROM correction to be judged (the fetch address and ROM correction operation are confirmed) and execution to branch to the correction software.

<4> After the correction software has been executed, the return address is set, and return processing is started by the DBRET instruction.

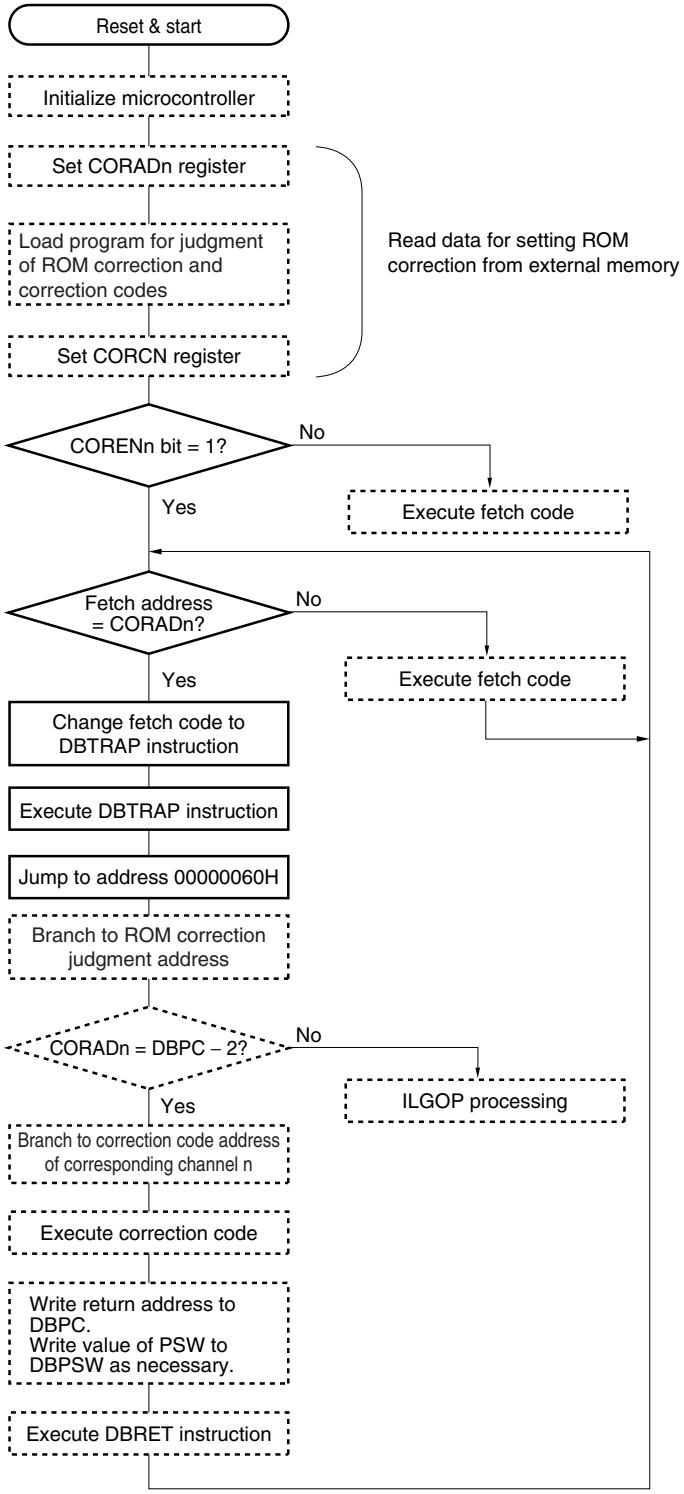
Cautions 1. The software that performs <3> and <4> must be executed in the external memory or internal RAM.

2. When setting an address to be corrected to the CORADn register, clear the higher bits to 0 in accordance with the capacity of the internal ROM.

3. The ROM correction function cannot be used to correct the data of the internal ROM. It can only be used to correct instruction codes. If ROM correction is used to correct data, that data is replaced with the DBTRAP instruction code.

Downloaded from Arrow.com.

Figure 21-2. ROM Correction Operation and Program Flow



Remarks 1. : Processing by user program (software)

: Processing by ROM correction (hardware)

2. n = 0 to 3

Downloaded from Arrow.com.

CHAPTER 22 FLASH MEMORY

The following products are the flash memory versions (single power supply) of the V850ES/KE1.

Caution There are differences in noise immunity and noise radiation between the flash memory and mask ROM versions. When pre-producing and application set with the flash memory version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluation for the commercial samples (not engineering samples) of the mask ROM version.
For the electrical specifications related to the flash memory rewriting, refer to CHAPTER 24 ELECTRICAL SPECIFICATIONS.

- μPD70F3207H, 70F3207HY: On-chip 128 KB flash memory

Flash memory versions are commonly used in the following development environments and mass production applications.

- ○ For altering software after the V850ES/KE1 is soldered onto the target system.
- ○ For data adjustment when starting mass production.
- ○ For differentiating software according to the specification in small scale production of various models.
- ○ For facilitating inventory management.
- ○ For updating software after shipment.

22.1 Features

- ○ 4-byte/1-clock access (when instruction is fetched)
- ○ Capacity: 128 KB
- ○ Write voltage: Erase/write with a single power supply
- ○ Rewriting method
  - Rewriting by communication with dedicated flash programmer via serial interface (on-board/off-board programming)
  - Rewriting flash memory by user program (self programming)
- ○ Flash memory write prohibit function supported (security function)
- ○ Safe rewriting of entire flash memory area by self programming using boot swap function
- ○ Interrupts can be acknowledged during self programming.

Caution TM50 and TM51 (including interrupts INTTM50 and INTTM51) cannot be used during self programming. For details, refer to 22.5.6 Internal resources used.

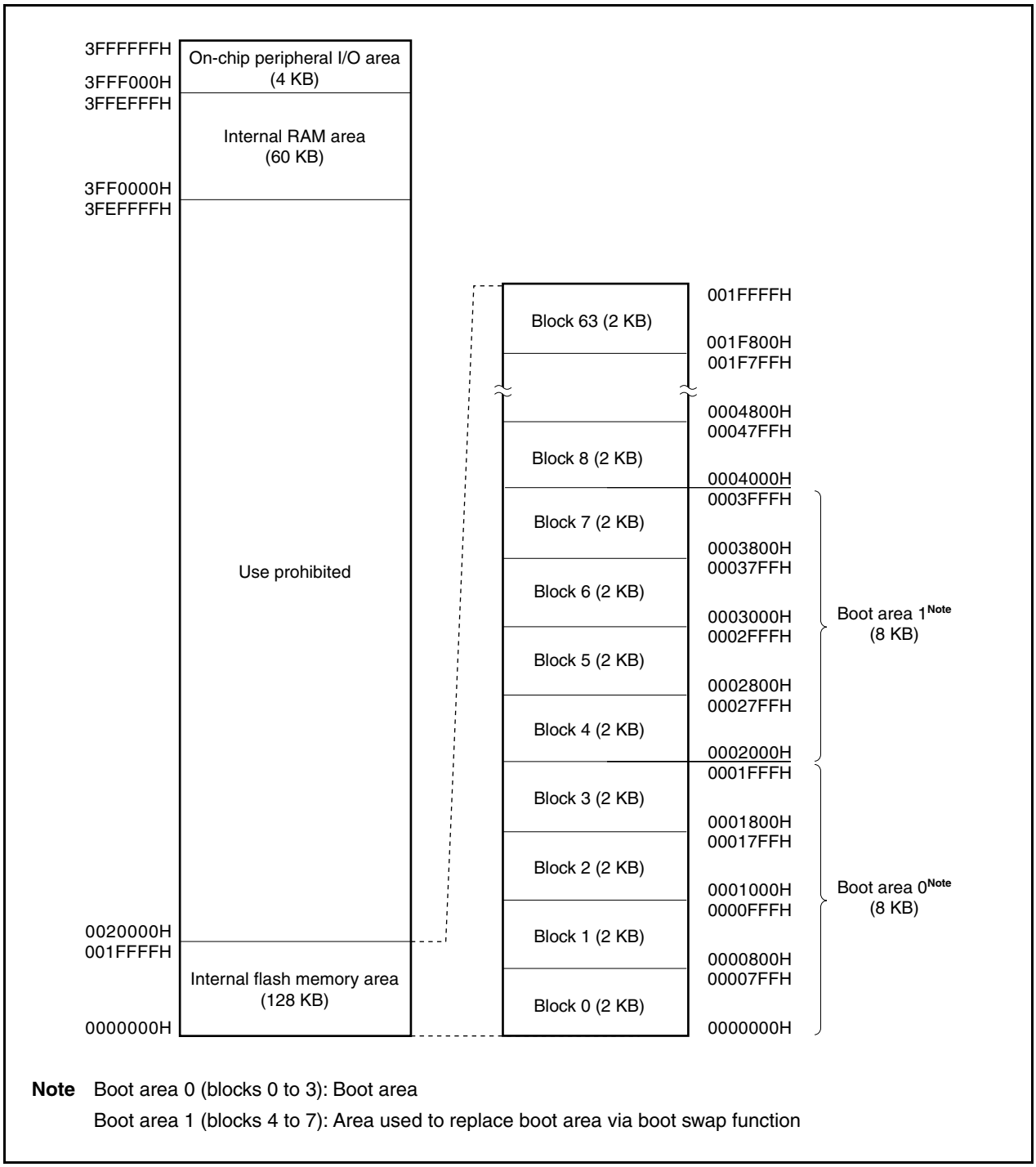
Downloaded from Arrow.com.

22.2 Memory Configuration

The 128 KB internal flash memory area is divided into 64 blocks and can be programmed/erased in block units. All the blocks can also be erased at once.

When the boot swap function is used, the physical memory (blocks 0 to 3) located at the addresses of boot area 0 is replaced by the physical memory (blocks 4 to 7) located at the addresses of boot area 1. For details of the boot swap function, refer to **22.5 Rewriting by Self Programming**.

Figure 22-1. Flash Memory Mapping



Note Boot area 0 (blocks 0 to 3): Boot area
Boot area 1 (blocks 4 to 7): Area used to replace boot area via boot swap function

Downloaded from Arrow.com.

22.3 Functional Outline

The internal flash memory of the V850ES/KE1 can be rewritten by using the rewrite function of the dedicated flash programmer, regardless of whether the V850ES/KE1 has already been mounted on the target system or not (on-board/off-board programming).

In addition, a security function that prohibits rewriting the user program written to the internal flash memory is also supported, so that the program cannot be changed by an unauthorized person.

The rewrite function using the user program (self programming) is ideal for an application where it is assumed that the program is changed after production/shipment of the target system. A boot swap function that rewrites the entire flash memory area safely is also supported. In addition, interrupt servicing is supported during self programming, so that the flash memory can be rewritten under various conditions, such as while communicating with an external device.

Table 22-1. Rewrite Method

Rewrite Method	Functional Outline	Operation Mode
On-board programming	Flash memory can be rewritten after the device is mounted on the target system, by using a dedicated flash programmer.	Flash memory programming mode
Off-board programming	Flash memory can be rewritten before the device is mounted on the target system, by using a dedicated flash programmer and a dedicated program adapter board (FA series).	
Self programming	Flash memory can be rewritten by executing a user program that has been written to the flash memory in advance by means of on-board/off-board programming. (During self-programming, instructions cannot be fetched from or data access cannot be made to the internal flash memory area. Therefore, the rewrite program must be transferred to the internal RAM or external memory in advance).	Normal operation mode

Remark The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

Downloaded from Arrow.com.

Table 22-2. Basic Functions

Function	Functional Outline	Support (○: Supported, ×: Not supported)	
		On-Board/Off-Board Programming	Self Programming
Block erasure	The contents of specified memory blocks are erased.	○	○
Chip erasure	The contents of the entire memory area are erased all at once.	○	×
Write	Writing to specified addresses, and a verify check to see if write level is secured are performed.	○	○
Verify/checksum	Data read from the flash memory is compared with data transferred from the flash programmer.	○	× (Can be read by user program)
Blank check	The erasure status of the entire memory is checked.	○	○
<R> Security setting	Use of the block erase command, chip erase command, program command, and read command can be prohibited, and rewriting boot area can be prohibited.	○	× (Supported only when setting is changed from enable to disable)

The following table lists the security functions. The block erase command prohibit, chip erase command prohibit, and program command prohibit functions are enabled by default after shipment, and security can be set by rewriting via on-board/off-board programming. Each security function can be used in combination with the others at the same time.

<R> **Table 22-3. Security Functions**

Function	Functional Outline
Block erase command prohibit	Execution of a block erase command on all blocks is prohibited. Setting of prohibition can be initialized by execution of a chip erase command.
Chip erase command prohibit	Execution of block erase and chip erase commands on all the blocks is prohibited. Once prohibition is set, setting of prohibition cannot be initialized because the chip erase command cannot be executed.
Program command prohibit	Execution of program and block erase commands on all the blocks is prohibited. Setting of prohibition can be initialized by execution of the chip erase command.
Read command prohibit	Executing of a read command on all blocks is prohibited. Setting of prohibition can be initialized by execution of the chip erase command.
Boot area rewrite prohibit	Boot areas from block 0 to the specified last block can be protected. The protected boot area cannot be rewritten (erased and written). Setting of prohibition cannot be initialized by execution of the chip erase command.

Downloaded from Arrow.com.

<R>

Table 22-4. Security Setting

Function	Erase, Write, Read Operations When Each Security Is Set (√: Executable, ×: Not Executable, –: Not Supported)		Notes on Security Setting	
	On-Board/ Off-Board Programming	Self Programming	On-Board/ Off-Board Programming	Self Programming
Block erase command prohibit	Block erase command: × Chip erase command: √ Program command: √ Read command: √	Block erasure (FlashBlockErase): √ Chip erasure: – Write (FlashWordWrite): √ Read (FlashWordRead): √	Setting of prohibition can be initialized by chip erase command.	Supported only when setting is changed from enable to prohibit
Chip erase command prohibit	Block erase command: × Chip erase command: × Program command: √ Read command: √	Block erasure (FlashBlockErase): √ Chip erasure: – Write (FlashWordWrite): √ Read (FlashWordRead): √	Setting of prohibition cannot be initialized.	
Program command prohibit	Block erase command: × Chip erase command: √Note 1 Program command: × Read command: √	Block erasure (FlashBlockErase): √ Chip erasure: – Write (FlashWordWrite): √ Read (FlashWordRead): √	Setting of prohibition can be initialized by chip erase command.	
Read command prohibit	Block erase command: √ Chip erase command: √ Program command: √ Read command: ×	Block erasure (FlashBlockErase): √ Chip erasure: – Write (FlashWordWrite): √ Read (FlashWordRead): √	Setting of prohibition can be initialized by chip erase command.	
Boot area rewrite prohibit	Block erase command: √Note 2 Chip erase command: × Program command: √Note 2 Read command: √	Block erasure (FlashBlockErase): √ Chip erasure: – Write (FlashWordWrite): √ Read (FlashWordRead): √	Setting of prohibition cannot be initialized.	

Notes 1. In this case, since the erase command is invalid, data different from the data already written in the flash memory cannot be written.

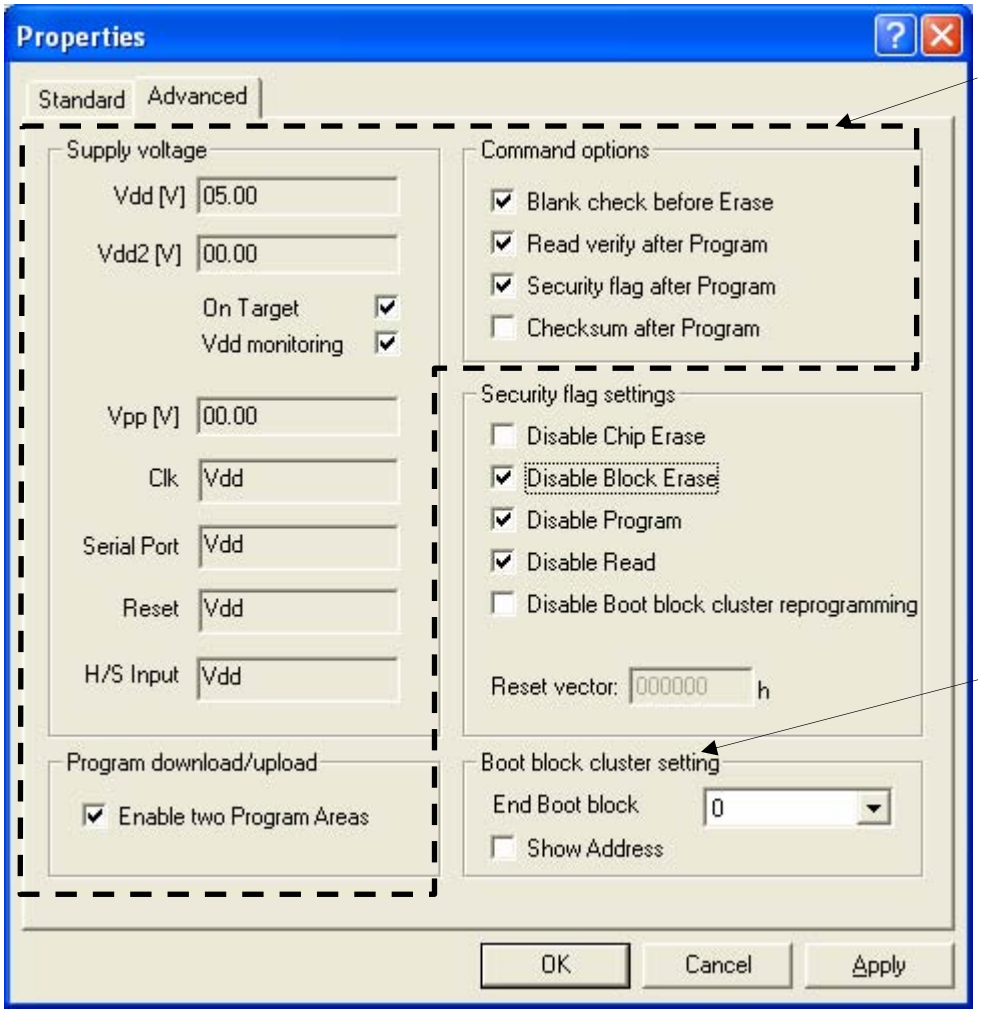
2. The boot area for which rewriting is prohibited is invalid.

Downloaded from Arrow.com.

<R> **(1) Security setting by PG-FP4 (Security flag settings)**

When disabling the read command (Disable Read), to raise the security level, it is recommended to also disable the block erase command (Disable Block Erase) and program command (Disable Program).

Furthermore, similar to the mask ROM products, when program rewriting is not necessary, additionally disable the chip erase command (Disable Chip Erase).



Notes 1. Set “Supply voltage”, “Program download/upload”, and “Command options” in broken lines in accordance with the use conditions.

2. To disable rewriting the boot area (Boot block cluster setting), select “Disable Boot block cluster reprogramming” in “Security flag settings” and select the last block of the boot area for which rewriting is to be disabled.

Downloaded from Arrow.com.

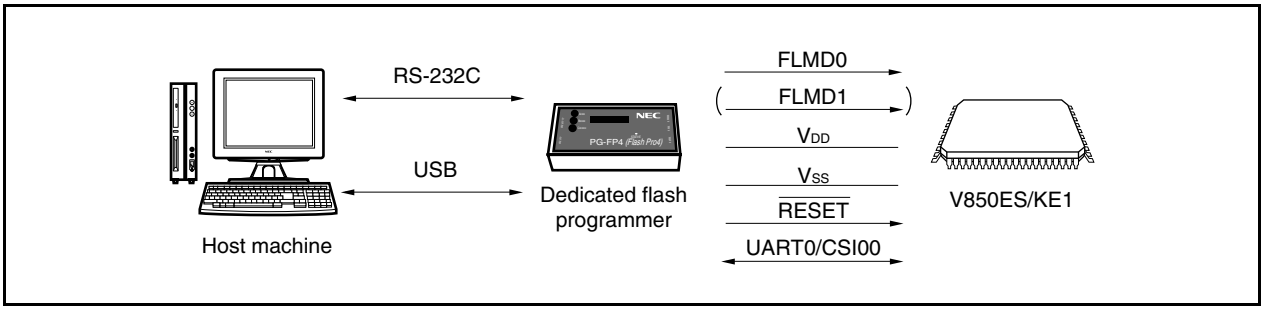
22.4 Rewriting by Dedicated Flash Programmer

The flash memory can be rewritten by using a dedicated flash programmer after the V850ES/KE1 is mounted on the target system (on-board programming). The flash memory can also be rewritten before the device is mounted on the target system (off-board programming) by using a dedicated program adapter (FA series).

22.4.1 Programming environment

The following shows the environment required for writing programs to the flash memory of the V850ES/KE1.

Figure 22-2. Environment Required for Writing Programs to Flash Memory



A host machine is required for controlling the dedicated flash programmer.

UART0 or CSI00 is used for the interface between the dedicated flash programmer and the V850ES/KE1 to perform writing, erasing, etc. A dedicated program adapter (FA series) is required for off-board writing.

Remark The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

Downloaded from Arrow.com.

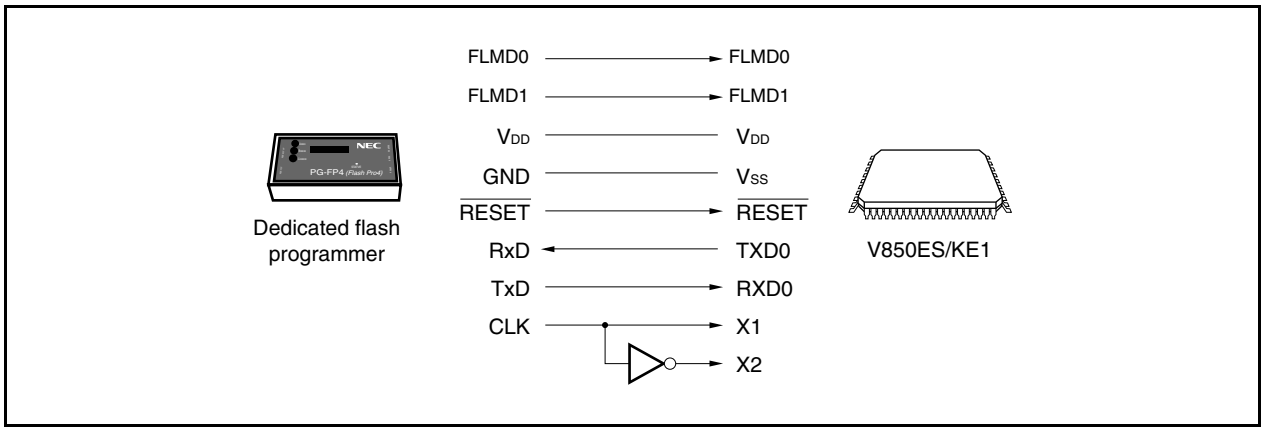
22.4.2 Communication mode

Communication between the dedicated flash programmer and the V850ES/KE1 is performed by serial communication using the UART0 or CSI00 interfaces of the V850ES/KE1.

(1) UART0

Transfer rate: 9,600 to 153,600 bps

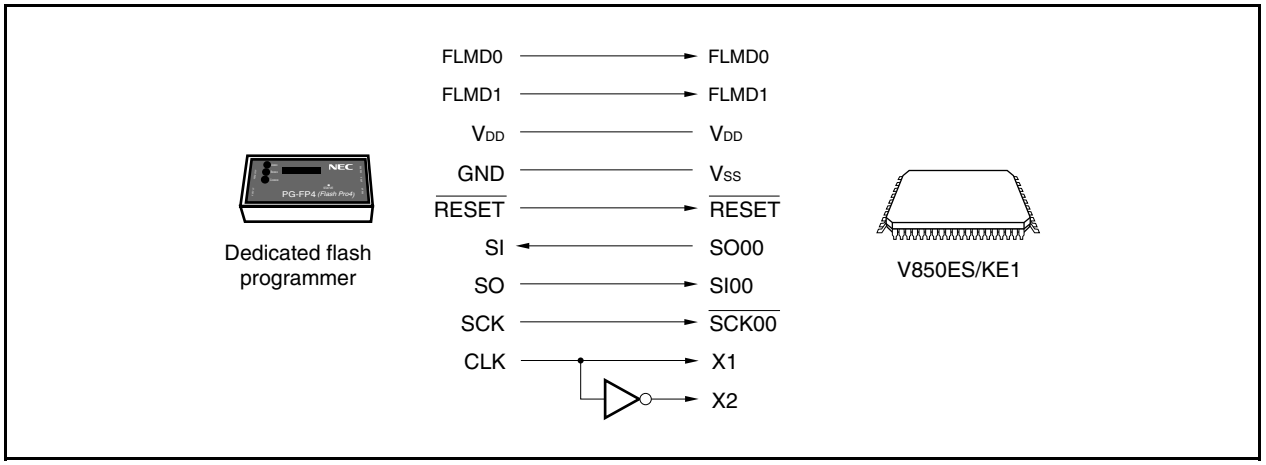
Figure 22-3. Communication with Dedicated Flash Programmer (UART0)



(2) CSI00

Serial clock: 2.4 kHz to 2.5 MHz (MSB first)

Figure 22-4. Communication with Dedicated Flash Programmer (CSI00)

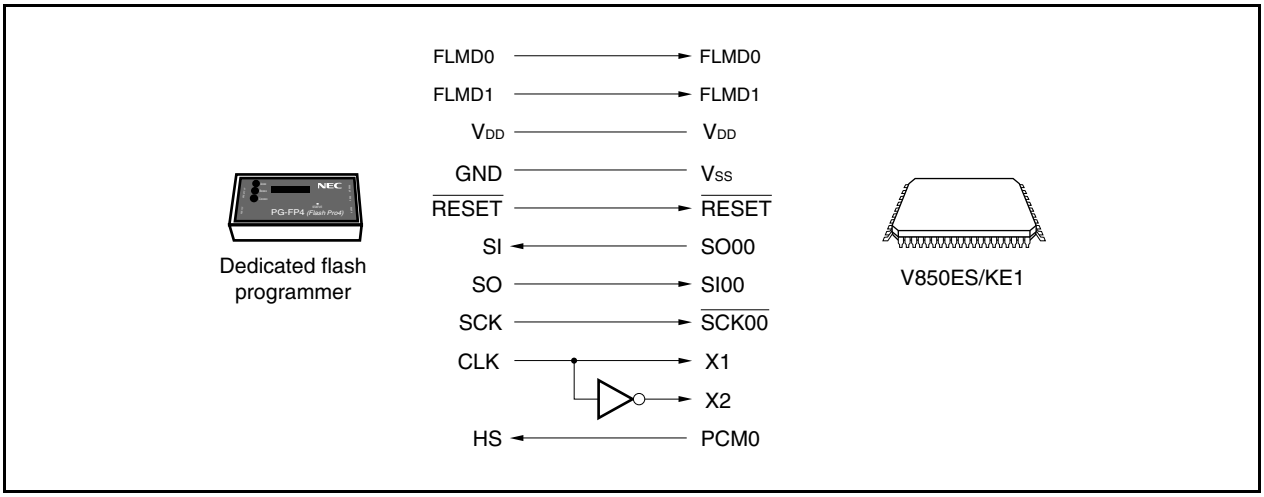


Downloaded from Arrow.com.

(3) CSI00 + HS

Serial clock: 2.4 kHz to 2.5 MHz (MSB first)

Figure 22-5. Communication with Dedicated Flash Programmer (CSI00 + HS)



The dedicated flash programmer outputs the transfer clock, and the V850ES/KE1 operates as a slave.

When the PG-FP4 is used as the dedicated flash programmer, it generates the following signals to the V850ES/KE1. For details, refer to the **PG-FP4 User's Manual (U15260E)**.

Table 22-5. Signal Connections of Dedicated Flash Programmer (PG-FP4)

PG-FP4			V850ES/KE1	Processing for Connection		
Signal Name	I/O	Pin Function	Pin Name	UART0	CSI00	CSI00 + HS
FLMD0	Output	Write enable/disable	FLMD0	◎	◎	◎
FLMD1	Output	Write enable/disable	FLMD1	◎Note 1	◎Note 1	◎Note 1
VDD	–	VDD voltage generation/voltage monitor	VDD	◎	◎	◎
GND	–	Ground	VSS	◎	◎	◎
CLK	Output	Clock output to V850ES/KE1	X1, X2	×Note 2	×Note 2	×Note 2
$\overline{\text{RESET}}$	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎	◎
SI/RxD	Input	Receive signal	SO00	◎	◎	◎
SO/TxD	Output	Transmit signal	SI00	◎	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK00}}$	×	◎	◎
HS	Input	Handshake signal for CSI00 + HS communication	PCM0	×	×	◎

Notes 1. Wire the pin as shown in Figure 22-6, or connect it to GND on board via a pull-down resistor.

2. Connect these pins to supply a clock from the PG-FP4 (wire as shown in Figure 22-6, or create an oscillator on board and supply the clock).

Remark ◎: Must be connected.

×: Does not have to be connected.

Downloaded from Arrow.com.

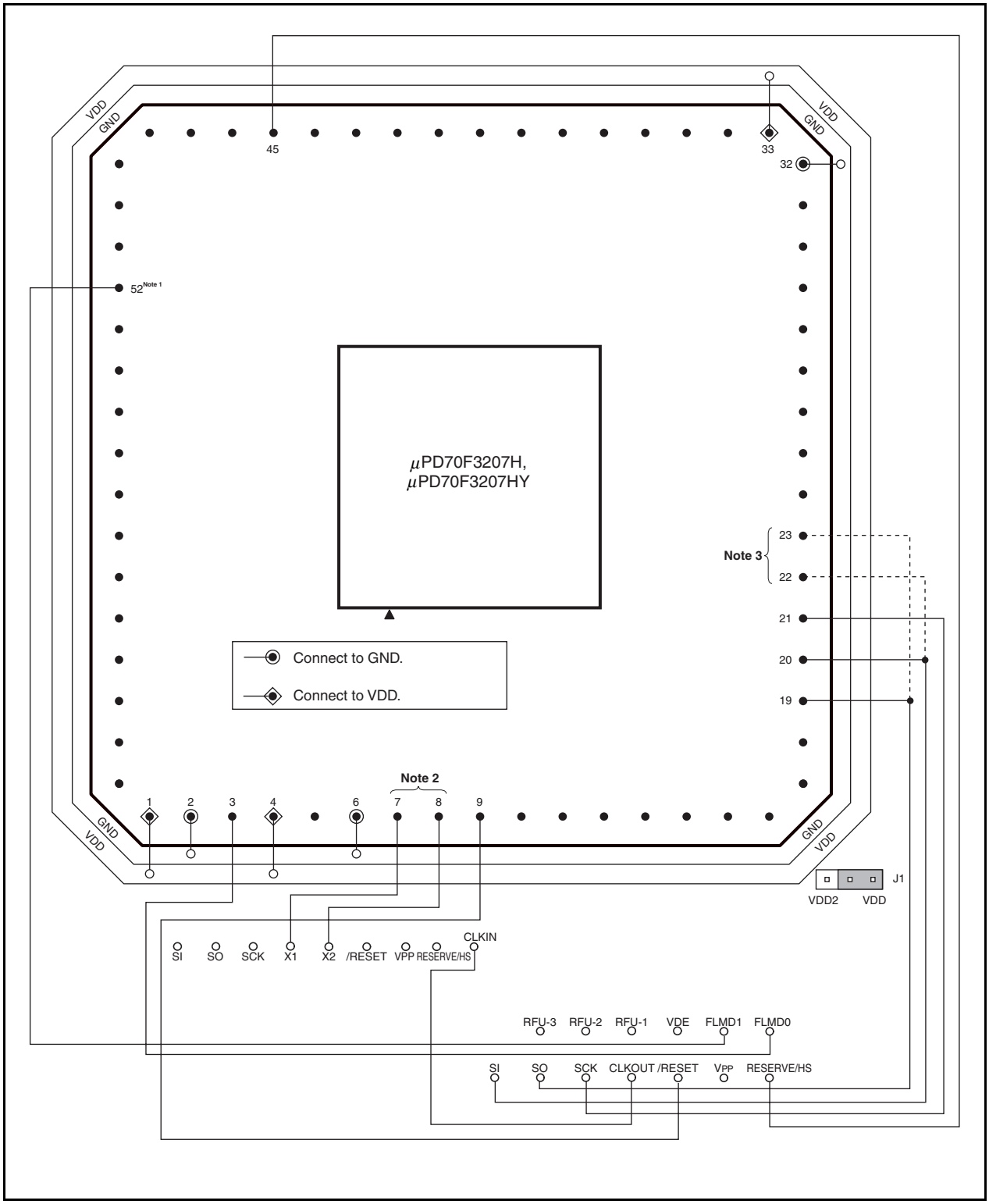
Table 22-6. Wiring Between μPD70F3207H, 70F3207HY and PG-FP4

Pin Configuration of Flash Programmer (PG-FP4)			Pin Name on FA Board	With CSI00-HS		With CSI00		With UART0	
Signal Name	I/O	Pin Function		Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.
SI/RxD	Input	Receive signal	SI	P41/SO00	20	P41/SO00	20	P30/TXD0	22
SO/TxD	Output	Transmit signal	SO	P40/SI00	19	P40/SI00	19	P31/RXD0	23
SCK	Output	Transfer clock	SCK	P42/$\overline{SCK00}$	21	P42/$\overline{SCK00}$	21	Not needed	Not needed
CLK	Output	Clock to V850ES/KE1	X1	X1	7	X1	7	X1	7
			X2	X2[Note]	8	X2[Note]	8	X2[Note]	8
/RESET	Output	Reset signal	/RESET	$\overline{RESET}$	9	$\overline{RESET}$	9	$\overline{RESET}$	9
FLMD0	Input	Write voltage	FLMD0	FLMD0	3	FLMD0	3	FLMD0	3
FLMD1	Input	Write voltage	FLMD1	PDL5/ FLMD1	52	PDL5/ FLMD1	52	PDL5/ FLMD1	52
HS	Input	Handshake signal for CSI00 + HS communication	RESERVE /HS	PCM0	45	Not needed	Not needed	Not needed	Not needed
VDD	–	V_{DD} voltage generation/voltage monitor	VDD	V_{DD}	4	V_{DD}	4	V_{DD}	4
				EV_{DD}	33	EV_{DD}	33	EV_{DD}	33
				AV_{REF0}	1	AV_{REF0}	1	AV_{REF0}	1
GND	–	Ground	GND	V_{SS}	6	V_{SS}	6	V_{SS}	6
				AV_{SS}	2	AV_{SS}	2	AV_{SS}	2
				EV_{SS}	32	EV_{SS}	32	EV_{SS}	32

Note When using the clock out of the flash programmer, connect CLK of the programmer to X1, and connect its inverse signal to X2.

Downloaded from Arrow.com.

<R> **Figure 22-6. Wiring Example of V850ES/KE1 Flash Writing Adapter (FA-64GK-9ET-A, FA-64GB-8EU-A) (1/2)**



Downloaded from Arrow.com.

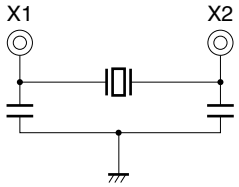
<R> **Figure 22-6. Wiring Example of V850ES/KE1 Flash Writing Adapter (FA-64GK-9ET-A, FA-64GB-8EU-A) (2/2)**

Notes 1. Wire the FLMD1 pin as shown in the figure, or connect it to GND on board via a pull-down resistor.

2. The above figure shows an example of wiring when the clock is supplied from the PG-FP4.
Be sure to set and connect as follows when the clock is supplied from the PG-FP4.

- Set J1 of the flash adapter (FA) to the VDD side.
- Connect CLKOUT of FA to CLKIN of FA.
- Connect X1 of FA to X1 of the device.
- Connect X2 of FA to X2 of the device.

If an oscillator is created on the flash adapter and a clock is supplied, the above setting and connections will not necessary. The following shows a circuit example.



3. Corresponding pin when using UART0

Remarks 1. Handle the pins not described above in accordance with the specified handling of unused pins (refer to **2.2 Pin I/O Circuits and Recommended Connection of Unused Pins).**
When connecting to V_{DD} via a resistor, use of a resistor of 1 kΩ to 10 kΩ is recommended.

2. This adapter is for a 64-pin plastic TQFP or 64-pin plastic LQFP (fine pitch) package.

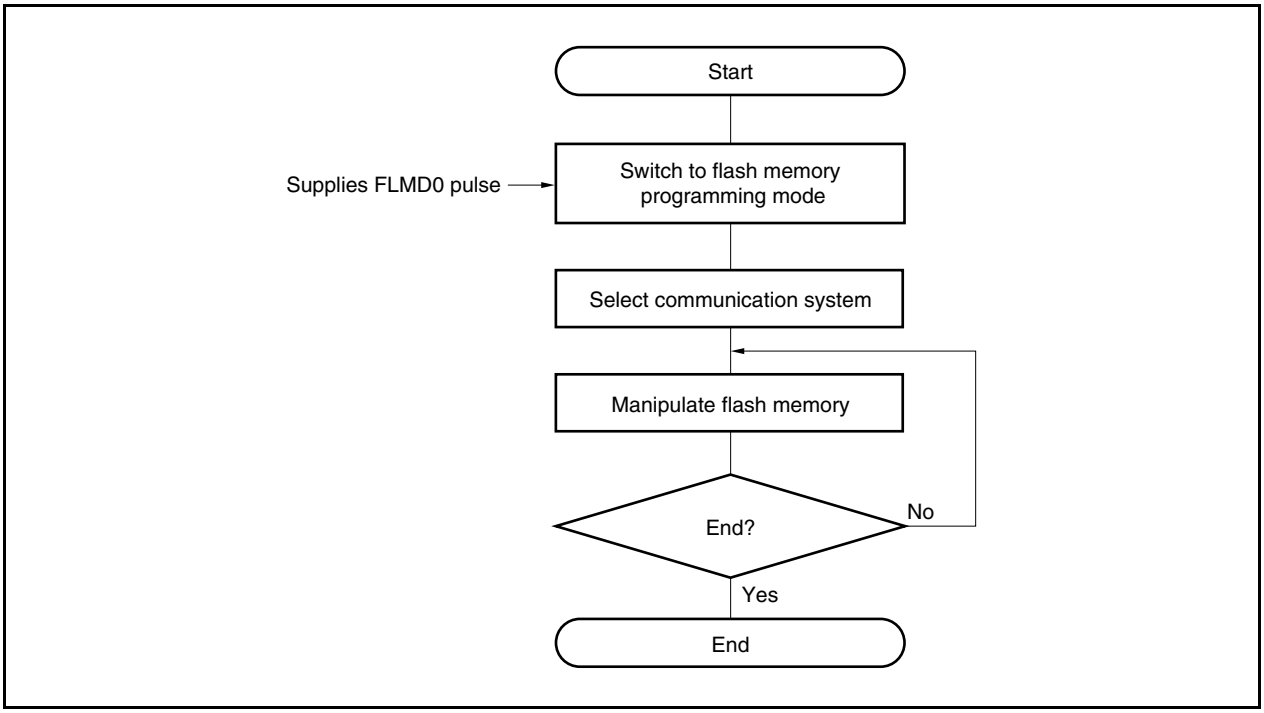
3. This diagram shows the wiring when using a handshake-supporting CSI.

Downloaded from Arrow.com.

22.4.3 Flash memory control

The following shows the procedure for manipulating the flash memory.

Figure 22-7. Procedure for Manipulating Flash Memory



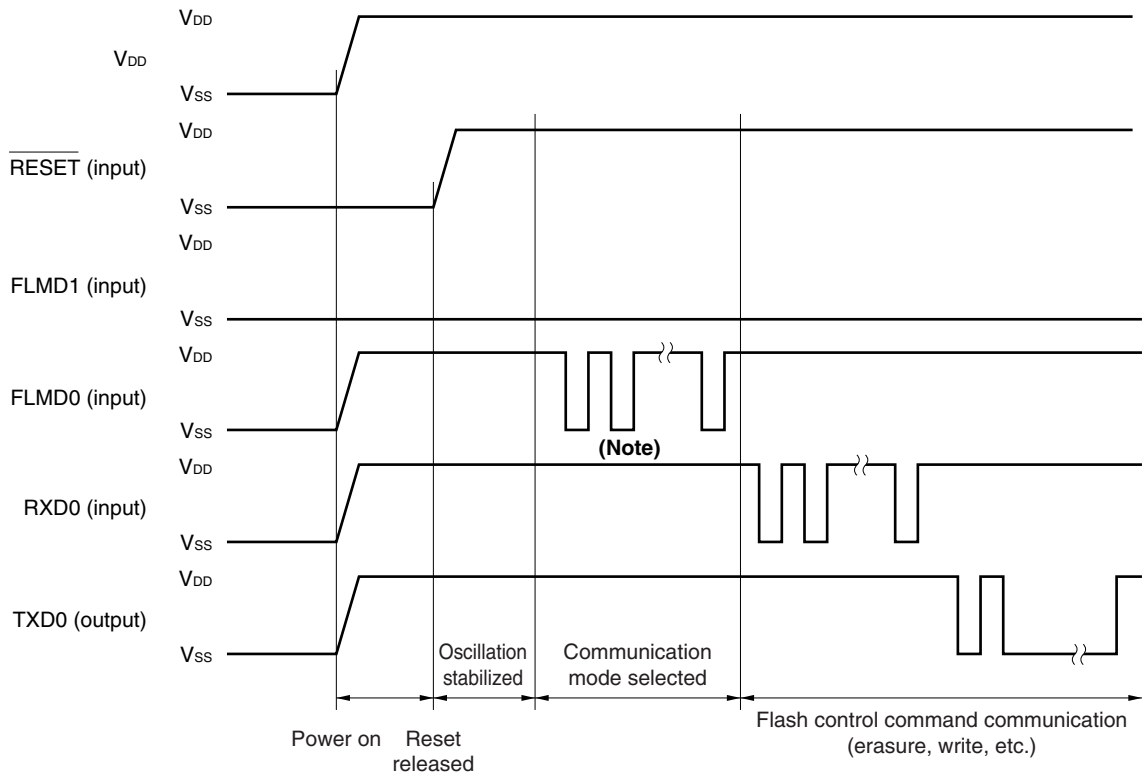
Downloaded from Arrow.com.

22.4.4 Selection of communication mode

In the V850ES/KE1, the communication mode is selected by inputting pulses (12 pulses max.) to the FLMD0 pin after switching to the flash memory programming mode. The FLMD0 pulse is generated by the dedicated flash programmer.

The following shows the relationship between the number of pulses and the communication mode.

Figure 22-8. Selection of Communication Mode



Note The number of clocks is as follows depending on the communication mode.

FLMD0 Pulse	Communication Mode	Remarks
0	UART0	Communication rate: 9600 bps (after reset), LSB first
8	CSI00	V850ES/KE1 performs slave operation, MSB first
11	CSI00 + HS	V850ES/KE1 performs slave operation, MSB first
Other	RFU	Setting prohibited

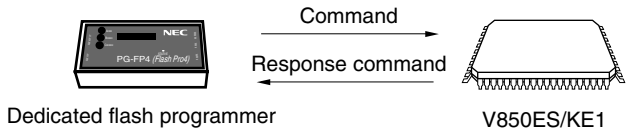
Caution When UART0 is selected, the receive clock is calculated based on the reset command sent from the dedicated flash programmer after receiving the FLMD0 pulse.

Downloaded from Arrow.com.

22.4.5 Communication commands

The V850ES/KE1 communicates with the dedicated flash programmer by means of commands. The signals sent from the dedicated flash programmer to the V850ES/KE1 are called “commands”. The response signals sent from the V850ES/KE1 to the dedicated flash programmer are called “response commands”.

Figure 22-9. Communication Commands



The following shows the commands for flash memory control in the V850ES/KE1. All of these commands are issued from the dedicated flash programmer, and the V850ES/KE1 performs the processing corresponding to the commands.

Table 22-7. Flash Memory Control Commands

Classification	Command Name	Support			Function
		CSI00	CSI00 + HS	UART0	
Blank check	Block blank check command	O	O	O	Checks if the contents of the memory in the specified block have been correctly erased.
Erase	Chip erase command	O	O	O	Erases the contents of the entire memory.
	Block erase command	O	O	O	Erases the contents of the memory of the specified block.
Write	Program command	O	O	O	Writes the specified address range, and executes a contents verify check.
Verify	Verify command	O	O	O	Compares the contents of memory in the specified address range with data transferred from the flash programmer.
	Checksum command	O	O	O	Reads the checksum in the specified address range.
System setting, control	Silicon signature command	O	O	O	Reads silicon signature information.
<R>	Security setting command	O	O	O	Disables chip erase, block erase, program, and read commands and disables rewriting the boot area.

Downloaded from Arrow.com.

22.4.6 Pin connection

When performing on-board writing, mount a connector on the target system to connect to the dedicated flash programmer. Also, incorporate a function on-board to switch from the normal operation mode to the flash memory programming mode.

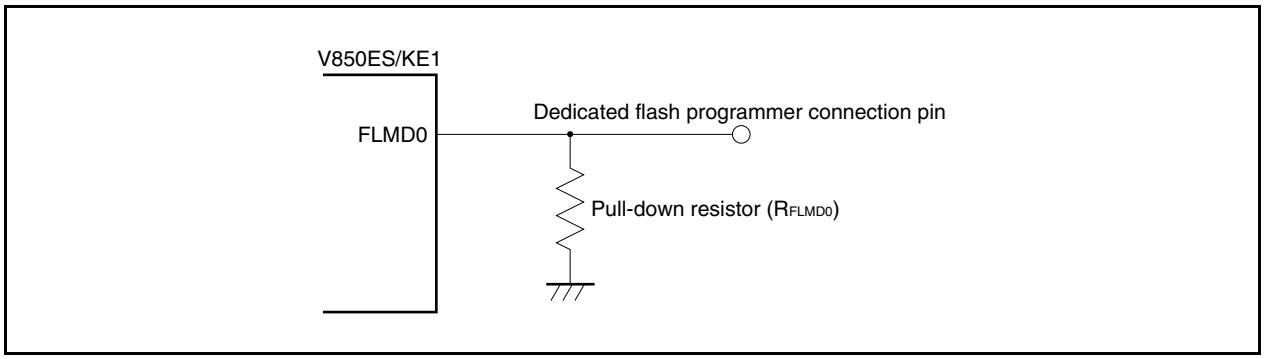
In the flash memory programming mode, all the pins not used for flash memory programming become the same status as that immediately after reset. Therefore, pin handling is required when the external device does not acknowledge the status immediately after a reset.

(1) FLMD0 pin

In the normal operation mode, input a voltage of V_{SS} level to the FLMD0 pin. In the flash memory programming mode, supply a write voltage of V_{DD} level to the FLMD0 pin.

Because the FLMD0 pin serves as a write protection pin in the self programming mode, a voltage of V_{DD} level must be supplied to the FLMD0 pin via port control, etc., before writing to the flash memory. For details, refer to **22.5.5 (1) FLMD0 pin**.

Figure 22-10. FLMD0 Pin Connection Example

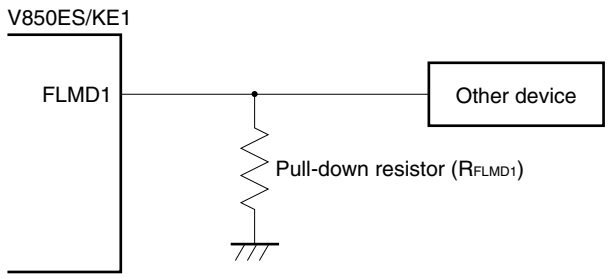


Downloaded from Arrow.com.

(2) FLMD1 pin

When 0 V is input to the FLMD0 pin, the FLMD1 pin does not function. When V_{DD} is supplied to the FLMD0 pin, the flash memory programming mode is entered, so 0 V must be input to the FLMD1 pin. The following shows an example of the connection of the FLMD1 pin.

Figure 22-11. FLMD1 Pin Connection Example



Caution If the V_{DD} signal is input to the FLMD1 pin from another device during on-board writing and immediately after reset, isolate this signal.

Table 22-8. Relationship Between FLMD0 and FLMD1 Pins and Operation Mode When Reset Is Released

FLMD0	FLMD1	Operation Mode
0	don't care	Normal operation mode
V_{DD}	0	Flash memory programming mode
V_{DD}	V_{DD}	Setting prohibited

Downloaded from Arrow.com.

(3) Serial interface pin

The following shows the pins used by each serial interface.

Table 22-9. Pins Used by Serial Interfaces

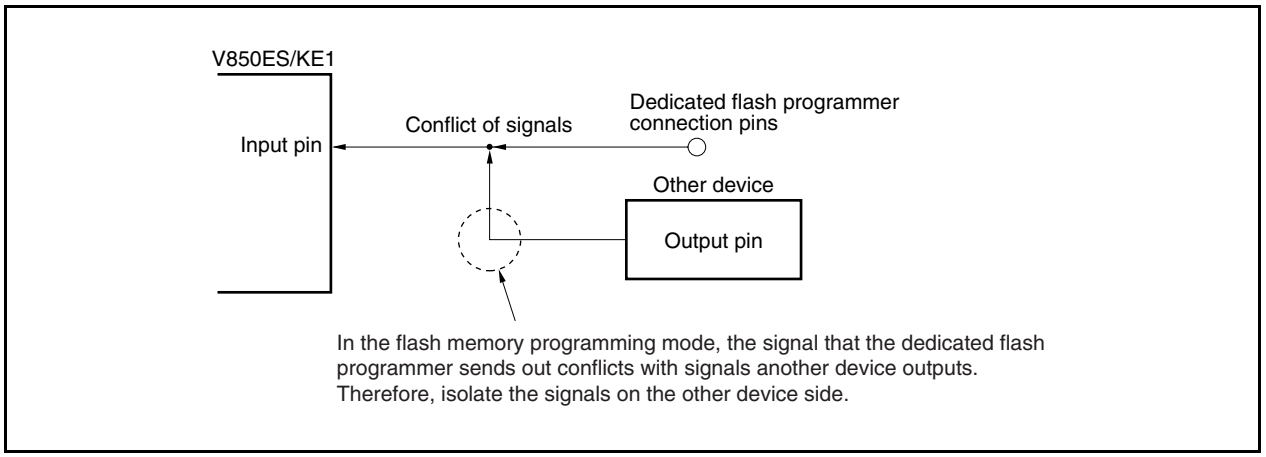
Serial Interface	Pins Used
UART0	TXD0, RXD0
CSI00	SO00, SI00, $\overline{\text{SCK00}}$
CSI00 + HS	SO00, SI00, $\overline{\text{SCK00}}$, PCM0

When connecting a dedicated flash programmer to a serial interface pin that is connected to another device on-board, care should be taken to avoid conflict of signals and malfunction of the other device.

(a) Conflict of signals

When the dedicated flash programmer (output) is connected to a serial interface pin (input) that is connected to another device (output), a conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the other device or set the other device to the output high-impedance status.

Figure 22-12. Conflict of Signals (Serial Interface Input Pin)

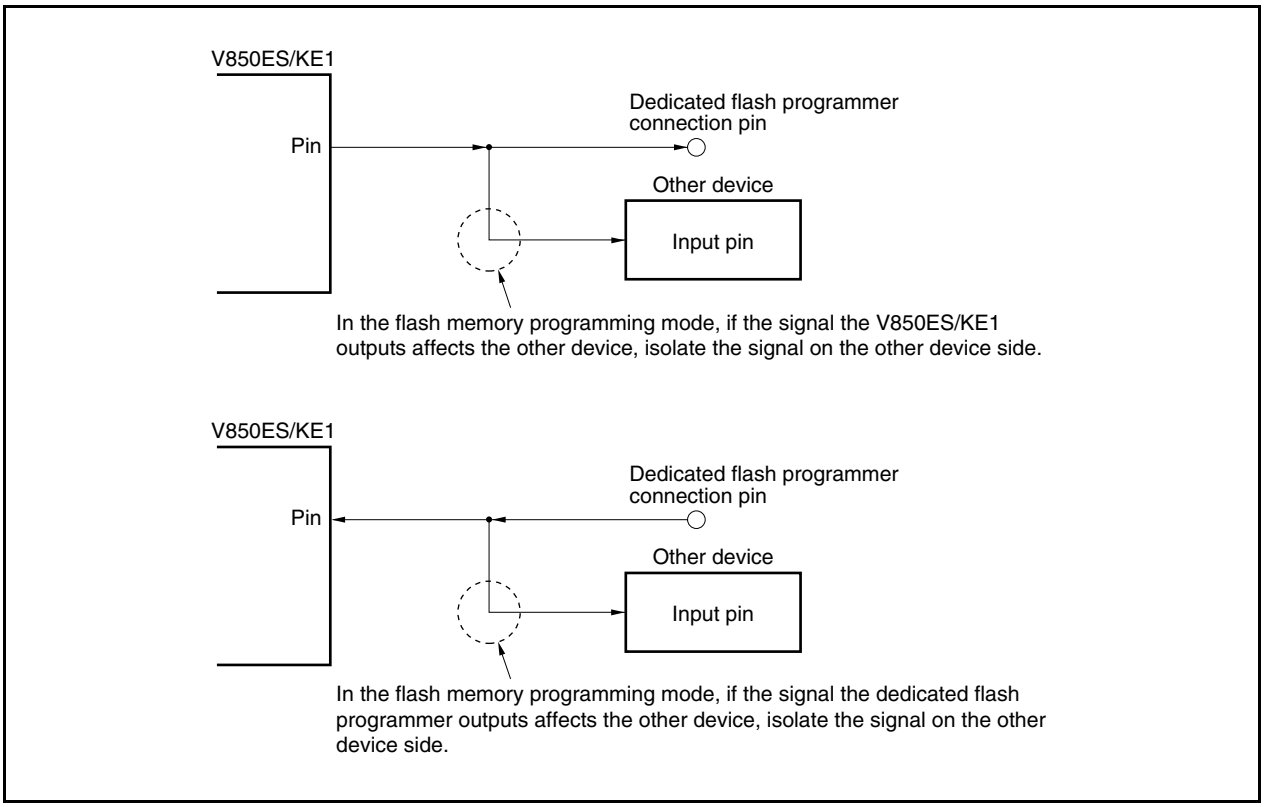


Downloaded from Arrow.com.

(b) Malfunction of other device

When the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), the signal is output to the other device, causing the device to malfunction. To avoid this, isolate the connection to the other device.

Figure 22-13. Malfunction of Other Device



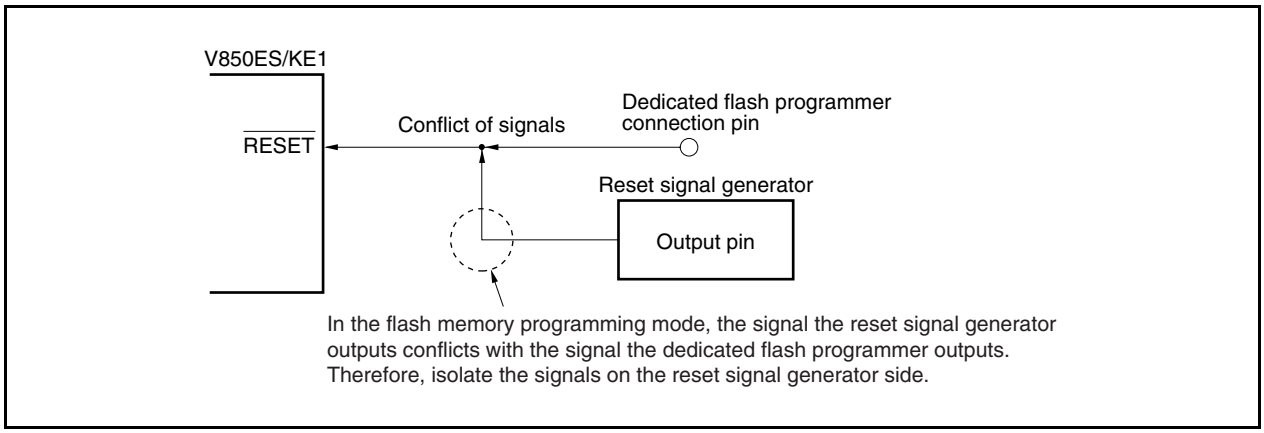
Downloaded from Arrow.com.

(4) $\overline{\text{RESET}}$ pin

When the reset signals of the dedicated flash programmer are connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on-board, a conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the reset signal generator.

When a reset signal is input from the user system in the flash memory programming mode, the programming operation will not be performed correctly. Therefore, do not input signals other than the reset signals from the dedicated flash programmer.

Figure 22-14. Conflict of Signals ($\overline{\text{RESET}}$ Pin)



(5) Port pins (including NMI)

When the system shifts to the flash memory programming mode, all the pins that are not used for flash memory programming are in the same status as that immediately after reset. If the external device connected to each port does not recognize the status of the port immediately after reset, pins require appropriate processing, such as connecting to V_{DD} via a resistor or connecting to V_{SS} via a resistor.

(6) Other signal pins

Connect X1, X2, XT1, and XT2 in the same status as that in the normal operation mode.

(7) Power supply

Supply the same power (V_{DD}, V_{SS}, EV_{DD}, EV_{SS}, AV_{SS}, AV_{REF0}) as in normal operation mode.

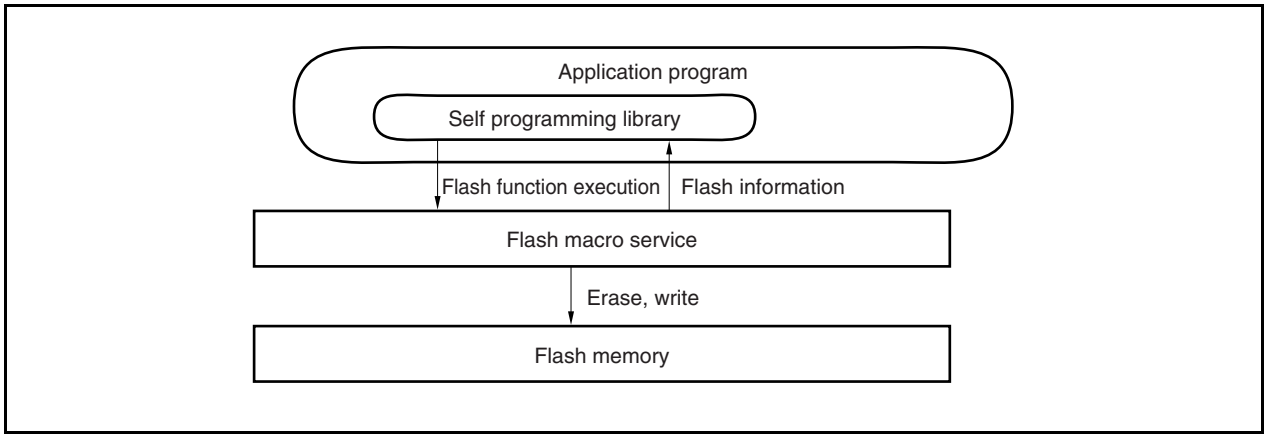
Downloaded from Arrow.com.

22.5 Rewriting by Self Programming

22.5.1 Overview

The V850ES/KE1 supports a flash macro service that allows the user program to rewrite the internal flash memory by itself. By using this interface and a self programming library that is used to rewrite the flash memory with a user application program, the flash memory can be rewritten by a user application transferred in advance to the internal RAM or external memory. Consequently, the user program can be upgraded and constant data can be rewritten in the field.

Figure 22-15. Concept of Self Programming



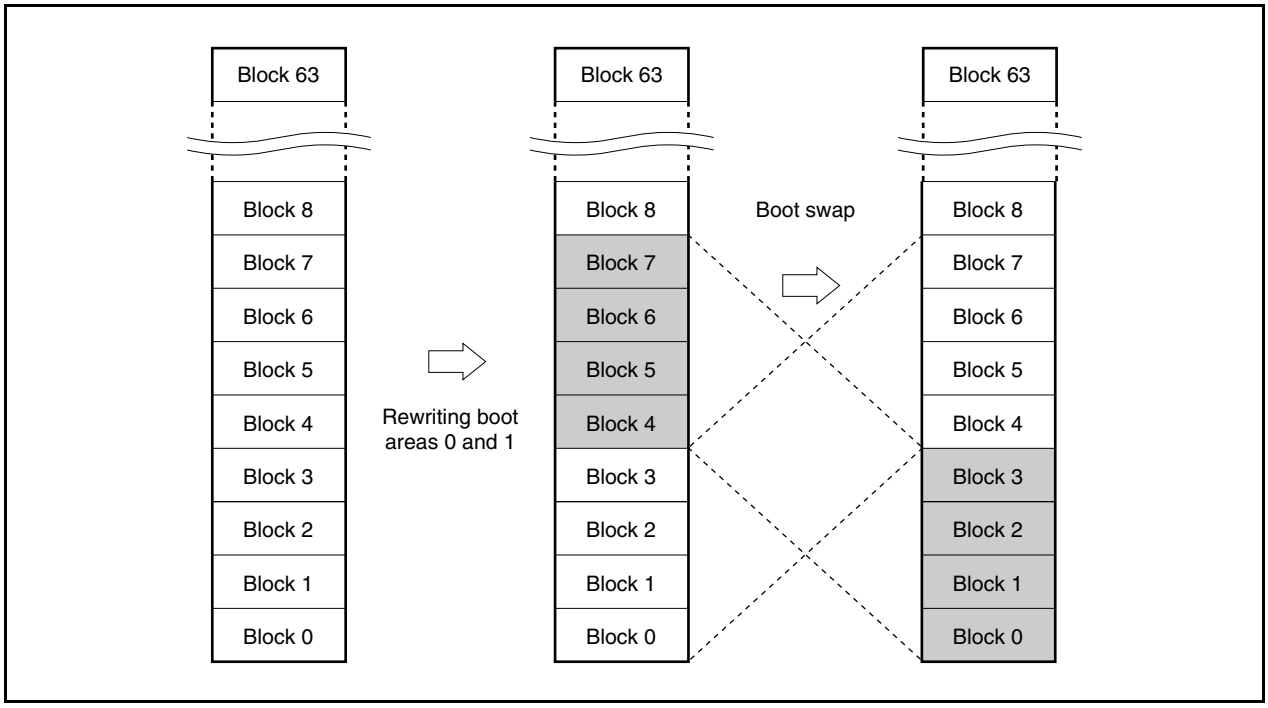
Downloaded from Arrow.com.

22.5.2 Features

(1) Secure self programming (boot swap function)

The V850ES/KE1 supports a boot swap function that can exchange the physical memory (blocks 0 to 3) of boot area 0 with the physical memory (blocks 4 to 7) of boot area 1. By writing the start program to be rewritten to boot area 1 in advance and then swapping the physical memory, the entire area can be safely rewritten even if a power failure occurs during rewriting because the correct user program always exists in boot area 0.

Figure 22-16. Rewriting Entire Memory Area (Boot Swap)



<R> **(2) Interrupt support**

Instructions cannot be fetched from the flash memory during self programming. Conventionally, therefore, a user handler written to the flash memory could not be used even if an interrupt occurred. Therefore, in the V850ES/KE1, to use an interrupt during self programming, processing transits to the specific address[Note] in the internal RAM. Allocate the jump instruction that transits processing to the user interrupt servicing at the specific address[Note] in the internal RAM.

Note NMI interrupt: Start address of internal RAM
Maskable interrupt: Start address of internal RAM + 4 addresses

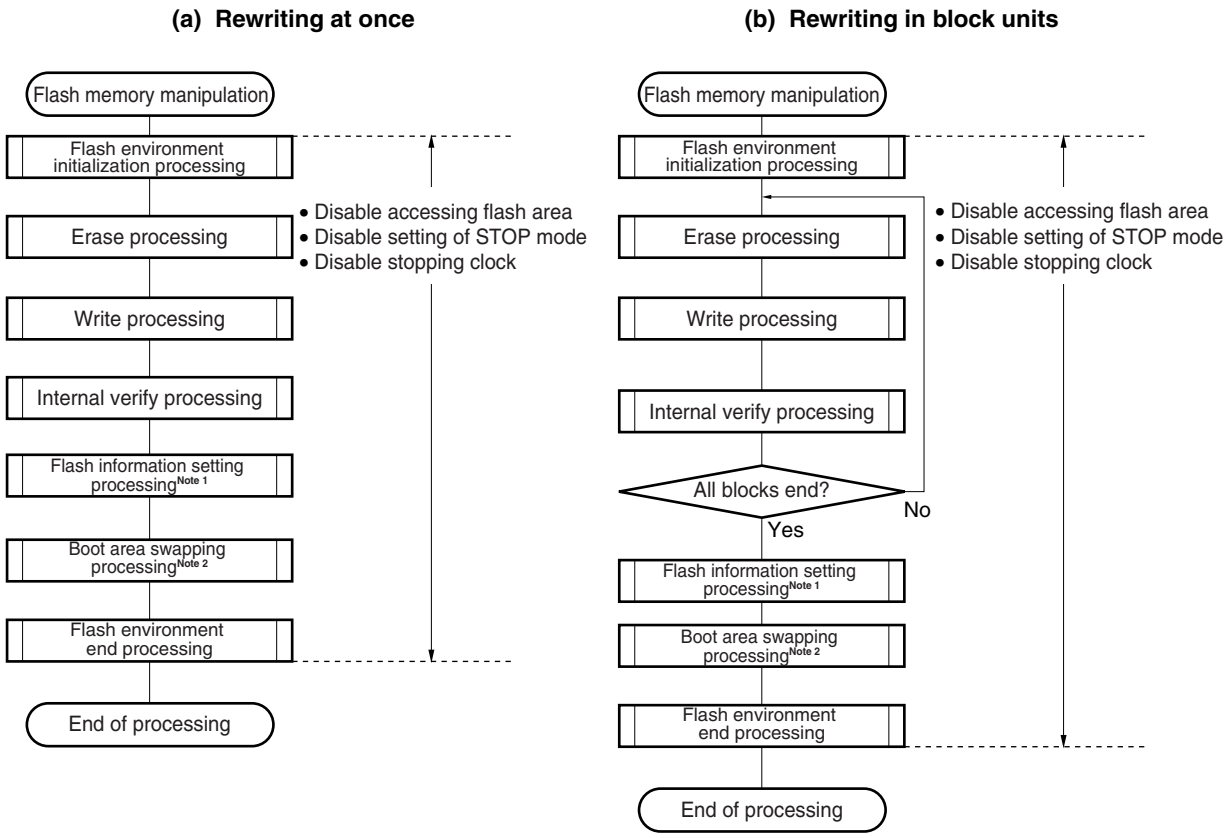
Downloaded from Arrow.com.

22.5.3 Standard self programming flow

The entire processing to rewrite the flash memory by flash self programming is illustrated below.

<R>

Figure 22-17. Standard Self Programming Flow



Notes 1. If a security setting is not performed, flash information setting processing does not have to be executed.

2. If boot swap is not used, flash information setting processing and boot area swap processing do not have to be executed.

Downloaded from Arrow.com.

22.5.4 Flash functions

Table 22-10. Main Flash Function List

Function Name	Outline	Support
FlashEnv	Initialization of flash control macro	√
FlashBlockErase	Erasure of only specified one block	√
FlashWordWrite	Writing from specified address	√
FlashBlockIVerify	Internal verification of specified block	√
FlashBlockBlankCheck	Blank check of specified block	√
FlashFLMDCheck	Check of FLMD pin	√
FlashGetInfo	Reading of flash information	√
FlashSetInfo	Setting of flash information	√
FlashBootSwap	Swapping of boot area	√
FlashWordRead	Reading data from specified address	√

Remark For details, refer to the **V850 Series Flash Memory Self Programming (Single Power Supply Flash Memory) User's Manual**.
Contact an NEC Electronics sales representative for the above manual.

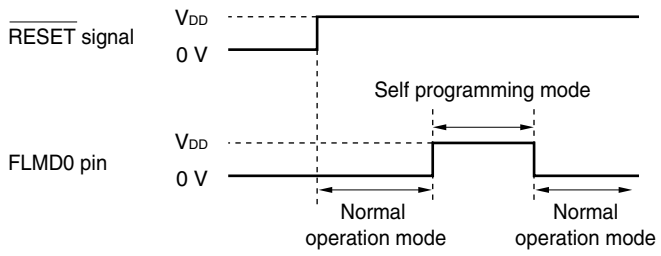
22.5.5 Pin processing

(1) FLMD0 pin

The FLMD0 pin is used to set the operation mode when reset is released and to protect the flash memory from being written during self rewriting. It is therefore necessary to keep the voltage applied to the FLMD0 pin at 0 V when reset is released and a normal operation is executed. It is also necessary to apply a voltage of V_{DD} level to the FLMD0 pin during the self programming mode period via port control before the memory is rewritten.

When self programming has been completed, the voltage on the FLMD0 pin must be returned to 0 V.

Figure 22-18. Mode Change Timing



Caution Make sure that the FLMD0 pin is at 0 V when reset is released.

Downloaded from Arrow.com.

22.5.6 Internal resources used

The following table lists the internal resources used for self programming. These internal resources can also be used freely for purposes other than self programming.

<R>

Table 22-11. Internal Resources Used

Resource Name	Description
Entry RAM area (internal RAM/external RAM size: 136 bytes)	Routines and parameters used for the flash macro service are located in this area. The entry program and default parameters are copied by calling a library initialization function.
Stack area (stack size: 600 bytes)	An extension of the stack used by the user is used by the library (can be used in both the internal RAM and external RAM).
Library code (code size: Approx. 1600 bytes)	Program entity of library (can be used anywhere other than the flash memory block to be manipulated).
Application program	Executed as user application. Calls flash functions.
Maskable interrupt	Can be used in user application execution status or self programming status. To use this interrupt in the self-programming status, since the processing transits to the address of the internal RAM start address + 4 addresses (3FFE004H), allocate the jump instruction that transits the processing to the user interrupt servicing at the address of the internal RAM start address + 4 addresses (3FFE004H) in advance.
NMI interrupt	Can be used in user application execution status or self programming status. To use this interrupt in the self-programming status, since the processing transits to the address of the internal RAM start address (3FFE000H), allocate the jump instruction that transits the processing to the user interrupt servicing at the internal RAM start address (3FFE000H) in advance.
TM50, TM51	Because TM50 and TM51 are used in the flash macro service, do not use them in the self programming status. When using TM50 and TM51 after self programming, set them again.

Remark For details, refer to the **V850 Series Flash Memory Self Programming (Single Power Supply Flash Memory) User's Manual**. Contact an NEC Electronics sales representative for the above manual.

Downloaded from Arrow.com.

<R>

CHAPTER 23 ON-CHIP DEBUG FUNCTION

The V850ES/KE1 is not provided with an on-chip debug function. However, a pseudo on-chip debug function can be realized by using the on-chip debug emulator (MINICUBE®) and debug adapter (QB-V850ESKX1-DA).
For connection examples, refer to **A.4.2 When using MINICUBE QB-V850MINI.**

23.1 ROM Security Function

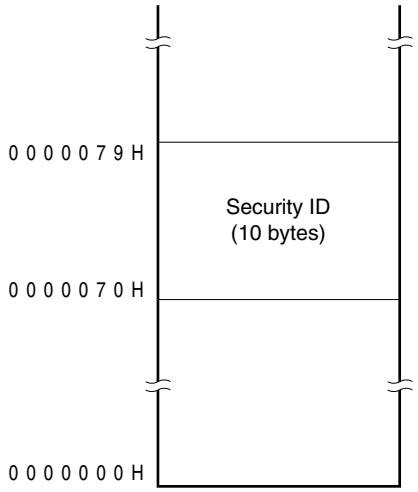
23.1.1 Security ID

The flash memory versions of the V850ES/KE1 perform authentication using a 10-byte ID code to prevent the contents of the flash memory from being read by an unauthorized person during on-chip debugging by the on-chip debug emulator.

Set the ID code in the 10-byte on-chip flash memory area from 0000070H to 0000079H to allow the debugger perform ID authentication.

If the IDs match, the security is released and reading flash memory and using the on-chip debug emulator are enabled.

- Set the 10-byte ID code to 0000070H to 0000079H.
- Bit 7 of 0000079H is the on-chip debug emulator enable flag.
  (0: Disable, 1: Enable)
- When the on-chip debug emulator is started, the debugger requests ID input. When the ID code input on the debugger and the ID code set in 0000070H to 0000079H match, the debugger starts.
- Debugging cannot be performed if the on-chip debug emulator enable flag is 0, even if the ID codes match.



Caution When the data in the flash memory has been deleted, all the bits are set to 1.

Downloaded from Arrow.com.

23.1.2 Setting

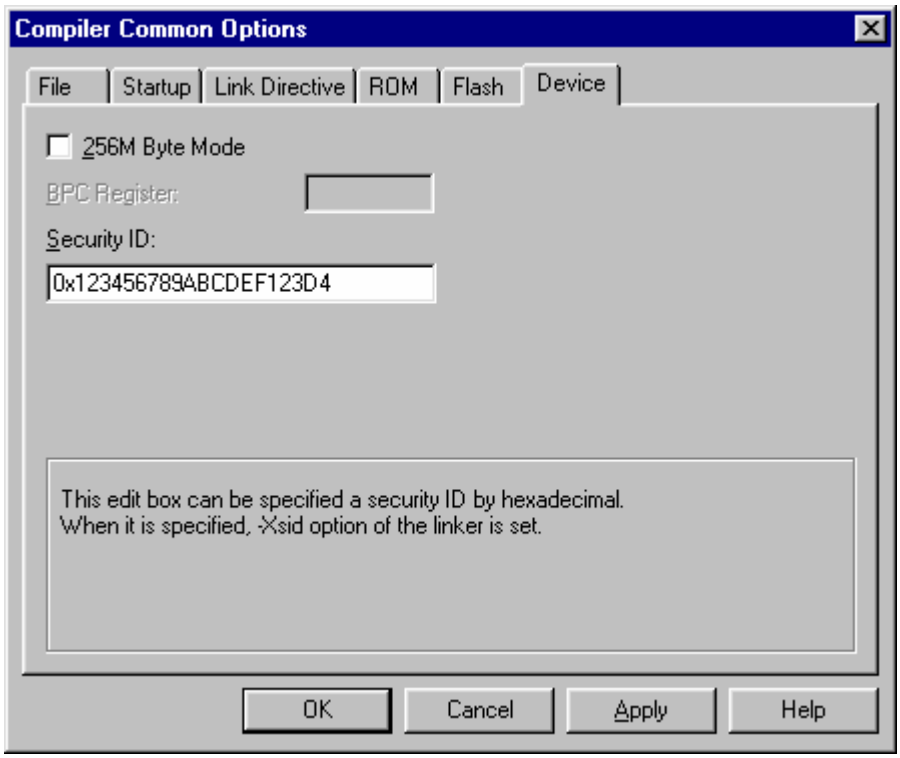
The following shows how to set the ID code as shown in Table 23-1.

When the ID code is set as shown in Table 23-1, the ID code input in the configuration dialog box of the ID850QB is “123456789ABCDEF123D4” (not case sensitive).

Table 23-1. ID Code

Address	Value
0x70	0x12
0x71	0x34
0x72	0x56
0x73	0x78
0x74	0x9A
0x75	0xBC
0x76	0xDE
0x77	0XF1
0x78	0x23
0x79	0xD4

The ID code can be specified for the device file that supports the CA850 Ver. 2.60 or later and the security ID by the PM+ linker option setting.



Downloaded from Arrow.com.

[Program example (when using CA850 Ver. 2.60 or later)]

```
#--------------------------------------
#       SECURITYID (continue ILGOP  handler)
#--------------------------------------
        .section    "SECURITY_ID"   --Interrupt handler address 0x70
        .word       0x78563412      --0-3 byte code
        .word       0xF1DEBC9A      --4-7 byte code
        .hword      0xD423          --8-9 byte code
```

Remark Add the above program example to the startup files.

23.2 Cautions

(1) If a reset signal is input (from the target system or a reset signal from an internal reset source) during RUN (program execution), the break function may malfunction.

(2) Even if the reset signal is masked by the mask function, the I/O buffer (port pin) may be reset if a reset signal is input from a pin.

(3) Because a software breakpoint set in the internal flash memory is realized by the ROM correction function, it is made temporarily invalid by target reset or internal reset generated by watchdog timer 2. The breakpoint becomes valid again when a hardware break or forced break occurs, but a software break does not occur until then.

(4) Pin reset during a break is masked and the CPU and peripheral I/O are not reset. If pin reset or internal reset is generated as soon as the flash memory is read by the RAM monitor function while the user program is being executed, the CPU and peripheral I/O may not be correctly reset.

(5) Emulation of ROM correction cannot be executed.

Downloaded from Arrow.com.

CHAPTER 24 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions		Ratings	Unit
Supply voltage	V_{DD}	$V_{DD} = EV_{DD} = AV_{REF0}$		−0.3 to +6.5	V
	EV_{DD}	$V_{DD} = EV_{DD} = AV_{REF0}$		−0.3 to +6.5	V
	AV_{REF0}	$V_{DD} = EV_{DD} = AV_{REF0}$		−0.3 to +6.5	V
	V_{SS}	$V_{SS} = EV_{SS} = AV_{SS}$		−0.3 to +0.3	V
	AV_{SS}	$V_{SS} = EV_{SS} = AV_{SS}$		−0.3 to +0.3	V
	EV_{SS}	$V_{SS} = EV_{SS} = AV_{SS}$		−0.3 to +0.3	V
Input voltage	V_{I1}	P00 to P06, P30 to P35, P38, P39, P40 to P42, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7, $\overline{\text{RESET}}$, FLMD0		−0.3 to EV_{DD} + 0.3[Note 1]	V
	V_{I2}	X1, X2, XT1, XT2		−0.3 to V_{DD} + 0.3[Note 1]	V
Analog input voltage	V_{IAN}	P70 to P77		−0.3 to AV_{REF0} + 0.3[Note 1]	V
Output current, low	I_{OL}	**Note 2**	Per pin	20	mA
		P38, P39		30	mA
		P00 to P06, P30 to P35, P38, P39, P40 to P42	Total of all pins: 70 mA	35	mA
		P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7		35	mA
Output current, high	I_{OH}	**Note 2**	Per pin	−10	mA
		P00 to P06, P30 to P35, P40 to P42	Total of all pins: −60 mA	−30	mA
		P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7		−30	mA
Operating ambient temperature	T_A	Normal operation mode		−40 to +85	°C
		Flash programming mode		−40 to +85	°C
Storage temperature	T_{stg}	Mask ROM version		−65 to +150	°C
		Flash memory version		−40 to +125	°C

<R>

Notes 1. Be sure not to exceed the absolute maximum ratings (MAX. value) of each supply voltage.

2. P00 to P06, P30 to P35, P40 to P42, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7

Cautions 1. Do not directly connect the output (or I/O) pins of IC products to each other, or to V_{DD}, V_{CC}, and GND. Open-drain pins or open-collector pins, however, can be directly connected to each other. Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.

2. Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded. The ratings and conditions indicated for DC characteristics and AC characteristics represent the quality assurance range during normal operation.

Downloaded from Arrow.com.

Capacitance (T_A = 25°C, V_{DD} = EV_{DD} = AV_{REF0} = V_{SS} = EV_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Input capacitance	C_I	f_X = 1 MHz Unmeasured pins returned to 0 V	P70 to P77			15	pF
I/O capacitance	C_{IO}		**Note**			15	pF
			P38, P39			20	pF

Note P00 to P06, P30 to P35, P40 to P42, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7

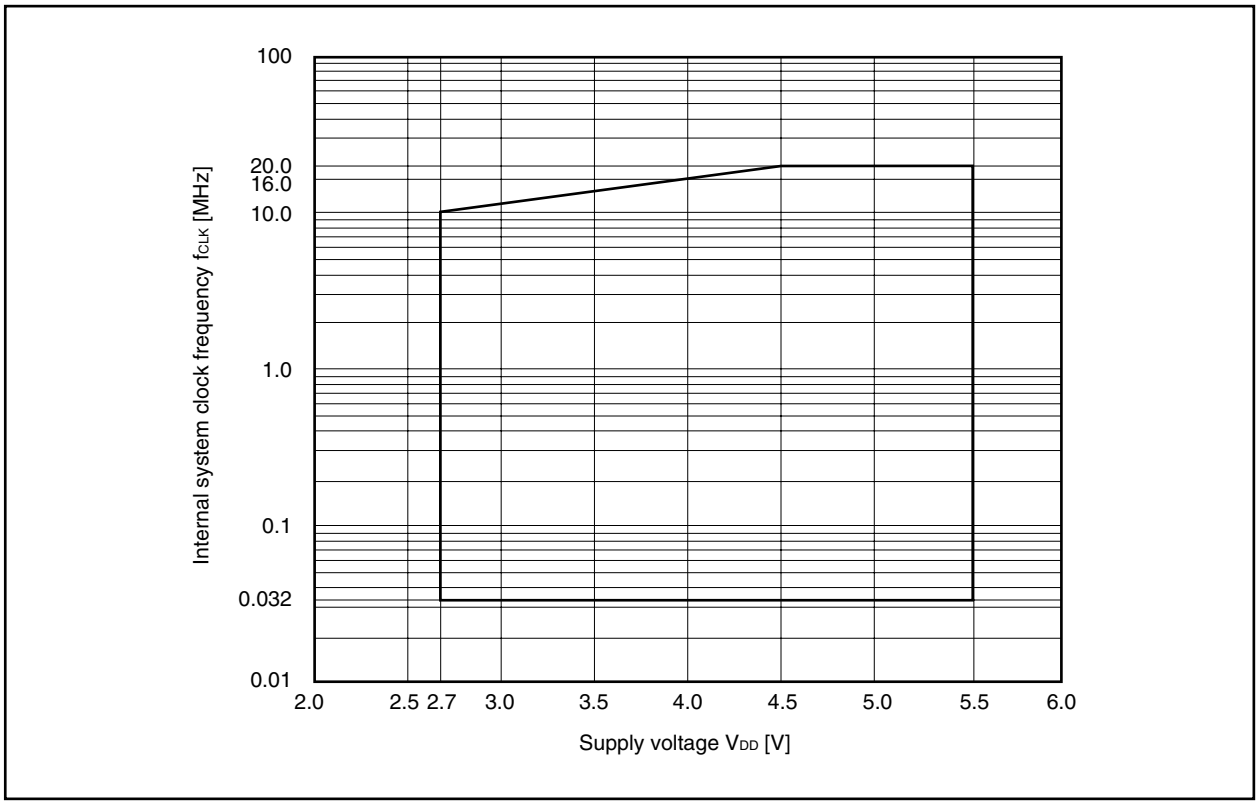
Remark f_X: Main clock oscillation frequency

Operating Conditions

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Internal system clock frequency	f_{CLK}	In PLL mode	V_{DD} = 4.5 to 5.5 V	0.25		20	MHz
			V_{DD} = 4.0 to 5.5 V	0.25		16	MHz
<R>			V_{DD} = 2.7 to 5.5 V	0.25		10	MHz
<R>		In clock-through mode	V_{DD} = 2.7 to 5.5 V	0.0625		10	MHz
		Operating with subclock	V_{DD} = 2.7 to 5.5 V		32.768		kHz

Internal System Clock Frequency vs. Supply Voltage



Downloaded from Arrow.com.

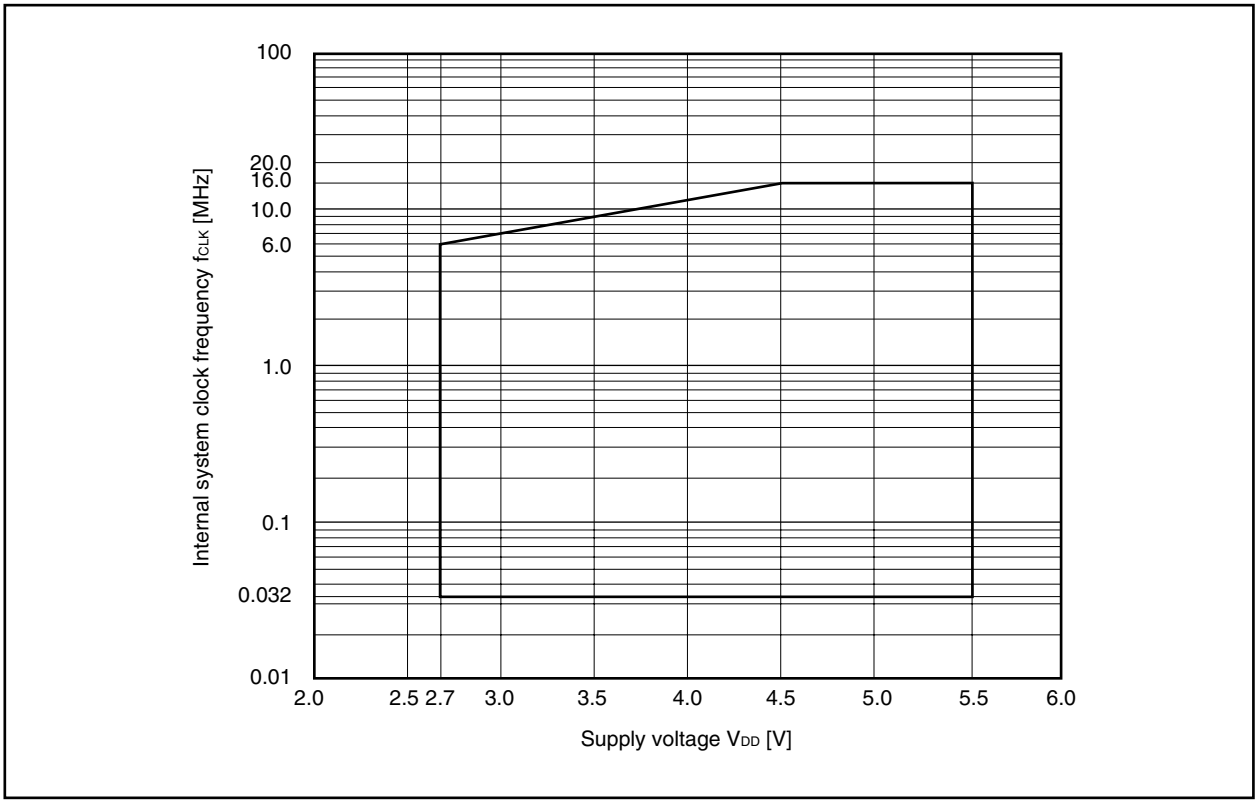
<R> **Operating Conditions for EEPROM Emulation**

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Internal system clock frequency	f_{CLK}	In PLL mode	V_{DD} = 4.5 to 5.5 V	0.25		16	MHz
			V_{DD} = 4.0 to 5.5 V	0.25		12	MHz
			V_{DD} = 2.7 to 5.5 V	0.25		6	MHz
		In clock-through mode	V_{DD} = 4.0 to 5.5 V	0.0625		10	MHz
			V_{DD} = 2.7 to 5.5 V	0.0625		6	MHz
		Operating with subclock[Note]	V_{DD} = 2.7 to 5.5 V		32.768		kHz

Note Do not stop the main clock.

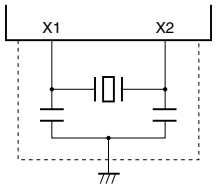
Internal System Clock Frequency vs. Supply Voltage



Downloaded from Arrow.com.

Main Clock Oscillator Characteristics

(1) Crystal resonator, ceramic resonator (T_A = –40 to +85°C, V_{DD} = 2.7 to 5.5 V, V_{SS} = 0 V)

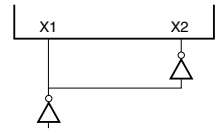
Recommended Circuit	Parameter	Conditions		MIN.	TYP.	MAX.	Unit
	Oscillation frequency (f_X)Note 1	In PLL mode	V_{DD} = 4.5 to 5.5 V	2		5	MHz
			V_{DD} = 4.0 to 5.5 V	2		4	MHz
			V_{DD} = 2.7 to 5.5 V	2		2.5	MHz
		In clock through mode	V_{DD} = 2.7 to 5.5 V	2		10	MHz
	Oscillation stabilization timeNote 2	After reset is released			$2^{15}/f_X$		s
		After STOP mode is released			**Note 3**		s

Notes 1. Indicates only oscillator characteristics.

2. Time required to stabilize the resonator after reset or STOP mode is released.

3. The value differs depending on the OSTS register settings.

(2) External clock (T_A = –40 to +85°C, V_{DD} = 2.7 to 5.5 V, V_{SS} = 0 V)

Recommended Circuit	Parameter	Conditions		MIN.	TYP.	MAX.	Unit
	X1, X2 input frequency (f_X)Note	In PLL mode	V_{DD} = 4.5 to 5.5 V	2		5	MHz
			V_{DD} = 4.0 to 5.5 V	2		4	MHz
			V_{DD} = 2.7 to 5.5 V	2		2.5	MHz
		In clock through mode	V_{DD} = 2.7 to 5.5 V	2		10	MHz

Note The duty ratio of the input waveform must be within 50% ±5%.

Cautions 1. When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

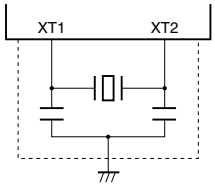
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

2. When the main clock is stopped and the device is operating on the subclock, wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.

Downloaded from Arrow.com.

Subclock Oscillator Characteristics

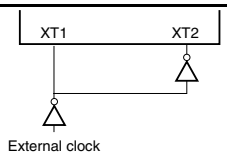
(1) Crystal resonator (T_A = –40 to +85°C, V_{DD} = 2.7 to 5.5 V, V_{SS} = 0 V)

Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
	Oscillation frequency (f_{XT})Note 1		32	32.768	35	kHz
	Oscillation stabilization timeNote 2			10		s

Notes 1. Indicates only oscillator characteristics.

2. Time required from when V_{DD} reaches oscillation voltage range (2.7 V (MIN.)) to when the crystal resonator stabilizes.

(2) External clock (T_A = –40 to +85°C, V_{DD} = 2.7 to 5.5 V, V_{SS} = 0 V)

Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
	Input frequency (f_{XT})Note		32		35	kHz

Note The duty ratio of the input waveform must be within 50% ±5%.

Cautions 1. When using the subclock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

2. The subclock oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the main clock oscillator. Particular care is therefore required with the wiring method when the subclock is used.

PLL Characteristics (T_A = –40 to +85°C, V_{DD} = 2.7 to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input frequency	f_X		2		5	MHz
Output frequency	f_{XX}		8		20	MHz
Lock time	t_{PLL}	After V_{DD} reaches 2.7 V (MIN.)			200	μs

Downloaded from Arrow.com.

DC Characteristics

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V) (1/4)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Output current, high	I_{OH1}	Per pin for P00 to P06, P30 to P35, P40 to P42, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7				–5.0	mA
		Total of P00 to P06, P30 to P35, P40 to P42	EV_{DD} = 4.0 to 5.5 V			–30	mA
			EV_{DD} = 2.7 to 5.5 V			–15	mA
		Total of P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7	EV_{DD} = 4.0 to 5.5 V			–30	mA
			EV_{DD} = 2.7 to 5.5 V			–15	mA
Output current, low	I_{OL1}	Per pin for P00 to P06, P30 to P35, P40 to P42, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7				10	mA
		Per pin for P38, P39	EV_{DD} = 4.0 to 5.5 V			15	mA
			EV_{DD} = 2.7 to 5.5 V			8	mA
		Total of P00 to P06, P30 to P35, P40 to P42				30	mA
		Total of P38, P39, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7				30	mA
Input voltage, high	V_{IH1}	**Note 1**		0.7EV_{DD}		EV_{DD}	V
	V_{IH2}	**Note 2**		0.8EV_{DD}		EV_{DD}	V
	V_{IH3}	P70 to P77		0.7AV_{REF0}		AV_{REF0}	V
	V_{IH4} Note 3	X1, X2, XT1, XT2		V_{DD} – 0.5		V_{DD}	V
Input voltage, low	V_{IL1}	**Note 1**		EV_{SS}		0.3EV_{DD}	V
	V_{IL2}	**Note 2**		EV_{SS}		0.2EV_{DD}	V
	V_{IL3}	P70 to P77		AV_{SS}		0.3AV_{REF0}	V
	V_{IL4} Note 3	X1, X2, XT1, XT2		V_{SS}		0.4	V

Notes 1. P00, P01, P30, P41, P98, PCM0, PCM1, PDL0 to PDL7 and their alternate-function pins.

2. $\overline{\text{RESET}}$, P02 to P06, P31 to P35, P38, P39, P40, P42, P50 to P55, P90, P91, P96, P97, P99, P913 to P915 including their alternate-function pins, and FLMD0 (flash memory versions only).

<R> **3.** When external clock is used

Downloaded from Arrow.com.

DC Characteristics

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V) (2/4)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Output voltage, high	V_{OH1}	**Note 1**	I_{OH} = –2.0 mA, EV_{DD} = 4.0 to 5.5 V	EV_{DD} – 1.0		EV_{DD}	V
		Note 2	I_{OH} = –0.1 mA, EV_{DD} = 2.7 to 5.5 V	EV_{DD} – 0.5		EV_{DD}	V
Output voltage, low	V_{OL1}	**Note 3**	I_{OL} = 2.0 mA[Note 4]	0		0.8	V
	V_{OL2}	P38, P39	I_{OL} = 15 mA, EV_{DD} = 4.0 to 5.5 V	0		2.0	V
			I_{OL} = 8 mA, EV_{DD} = 3.0 to 5.5 V	0		1.0	V
			I_{OL} = 5 mA, EV_{DD} = 2.7 to 5.5 V	0		1.0	V
Input leakage current, high	I_{LIH}	V_{IN} = V_{DD}				3.0	μA
Input leakage current, low	I_{LIL}	V_{IN} = 0 V				–3.0	μA
Output leakage current, high	I_{LOH}	V_O = V_{DD}				3.0	μA
Output leakage current, low	I_{LOL}	V_O = 0 V				–3.0	μA
Pull-up resistor	R_L	V_{IN} = 0 V		10	30	100	kΩ

Notes 1. Total of P00 to P06, P30 to P35, P40 to P42 and their alternate-function pins: I_{OH} = –30 mA, total of P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7 and their alternate-function pins: I_{OH} = –30 mA.

2. Total of P00 to P06, P30 to P35, P40 to P42 and their alternate-function pins: I_{OH} = –15 mA, total of P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7 and their alternate-function pins: I_{OH} = –15 mA.

3. Total of P00 to P06, P30 to P35, P40 to P42 and their alternate-function pins: I_{OL} = 30 mA, total of P38, P39, P50 to P55, P90, P91, P96 to P99, P913 to P915, PCM0, PCM1, PDL0 to PDL7 and their alternate-function pins: I_{OL} = 30 mA.

4. Refer to I_{OL1} for I_{OL} of P38 and P39.

Downloaded from Arrow.com.

<R> **DC Characteristics**

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V) (3/4)

Parameter	Symbol	Conditions	MIN.	TYP.[Note 2]	MAX.	Unit
Supply current[Note 1] (μPD70F3207H, 70F3207HY)	I_{DD1}	Normal operation mode (all peripheral functions operating)				
		f_{XX} = 20 MHz (f_X = 5 MHz) (in PLL mode) V_{DD} = 5 V ±10%		51	70	mA
		f_{XX} = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		17	34	mA
	I_{DD2}	HALT mode (all peripheral functions operating)				
		f_{XX} = 20 MHz (f_X = 5 MHz) (in PLL mode) V_{DD} = 5 V ±10%		25	38	mA
		f_{XX} = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		9	15	mA
	I_{DD3}	IDLE mode (watch timer operating)				
		f_X = 5 MHz (when PLL mode off) V_{DD} = 5 V ±10%		1.8	2.9	mA
		f_X = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		1.4	2.4	mA
	I_{DD4}	Subclock operation mode (f_{XT} = 32.768 kHz) Main oscillation stopped		240	400	μA
	I_{DD5}	Sub-IDLE mode (f_{XT} = 32.768 kHz) Watch timer operating, main oscillation stopped		20	75	μA
	I_{DD6}	STOP mode				
		Subclock oscillating		15	60	μA
		Subclock stopped (XT1 = V_{SS}, PSMR.XTSTP bit = 1)		0.1	30	μA
	I_{DD7}	Flash memory erase/write (T_A = –40 to +85°C)				
		f_{XX} = 20 MHz (f_X = 5 MHz) (in PLL mode) V_{DD} = 5 V ±10%		51	70	mA
		f_{XX} = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		17	34	mA

Notes 1. Total current of V_{DD} and EV_{DD} (all ports stopped). AV_{REF0} is not included.

2. TYP. values are formed using the following voltage.

V_{DD} = 5.0 V when V_{DD} = 5 V ±10%

V_{DD} = 3.0 V when V_{DD} = 3 V ±10%

Remark f_{XX}: Main clock frequency

f_X: Main clock oscillation frequency

f_{XT}: Subclock frequency

Downloaded from Arrow.com.

<R> **DC Characteristics**

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V) (4/4)

Parameter	Symbol	Conditions	MIN.	TYP.[Note 2]	MAX.	Unit
Supply current[Note 1] (μPD703207, 703207Y)	I_{DD1}	Normal operation mode (all peripheral functions operating)				
		f_{XX} = 20 MHz (f_X = 5 MHz) (in PLL mode) V_{DD} = 5 V ±10%		40	60	mA
		f_{XX} = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		12	24	mA
	I_{DD2}	HALT mode (all peripheral functions operating)				
		f_{XX} = 20 MHz (f_X = 5 MHz) (in PLL mode) V_{DD} = 5 V ±10%		25	35	mA
		f_{XX} = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		7	14	mA
	I_{DD3}	IDLE mode (watch timer operating, internal oscillation stopped)				
		f_X = 5 MHz (when PLL mode off) V_{DD} = 5 V ±10%		1.4	2.3	mA
		f_X = 10 MHz (in clock-through mode) V_{DD} = 3 V ±10%		1.0	1.7	mA
	I_{DD4}	Subclock operation mode (f_{XT} = 32.768 kHz) Main oscillation stopped		90	200	μA
	I_{DD5}	Sub-IDLE mode (f_{XT} = 32.768 kHz) Watch timer operating, main oscillation stopped		20	75	μA
	I_{DD6}	STOP mode				
		Subclock oscillating		15	60	μA
		Subclock stopped (XT1 = V_{SS}, PSMR.XTSTP bit = 1)		0.1	30	μA

Notes 1. Total current of V_{DD} and EV_{DD} (all ports stopped). AV_{REF0} is not included.

2. TYP. values are formed using the following voltage.
V_{DD} = 5.0 V when V_{DD} = 5 V ±10%
V_{DD} = 3.0 V when V_{DD} = 3 V ±10%

Remark f_{XX}: Main clock frequency
f_X: Main clock oscillation frequency
f_{XT}: Subclock frequency

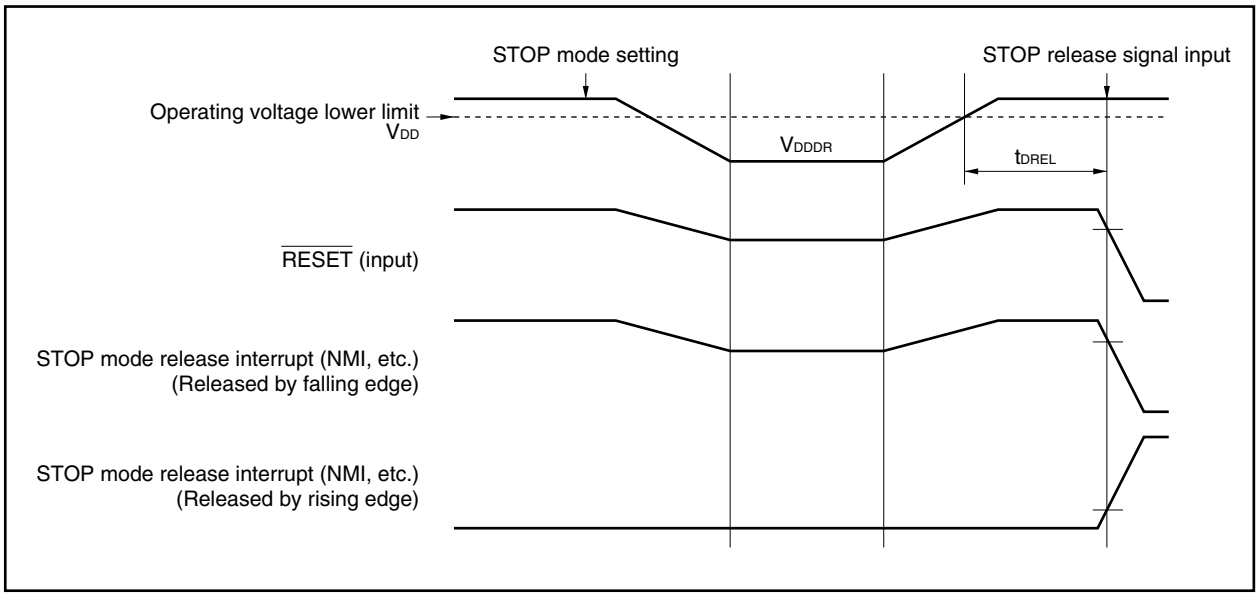
Downloaded from Arrow.com.

Data Retention Characteristics

STOP Mode (T_A = –40 to +85°C)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention voltage	V_{DDDR}	STOP mode	2.0		5.5	V
STOP release signal input time	t_{DREL}		0			μs

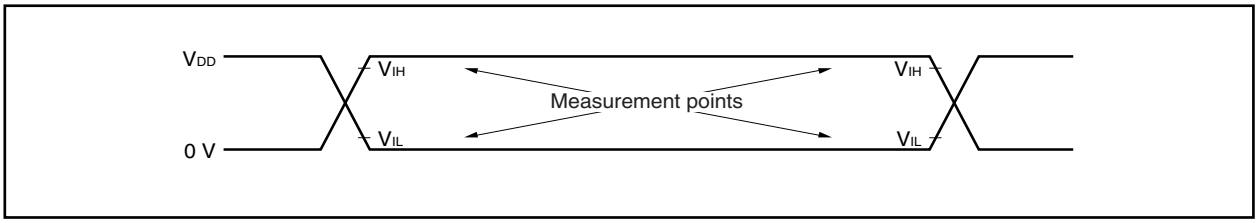
Caution Shifting to STOP mode and restoring from STOP mode must be performed within the rated operating range.



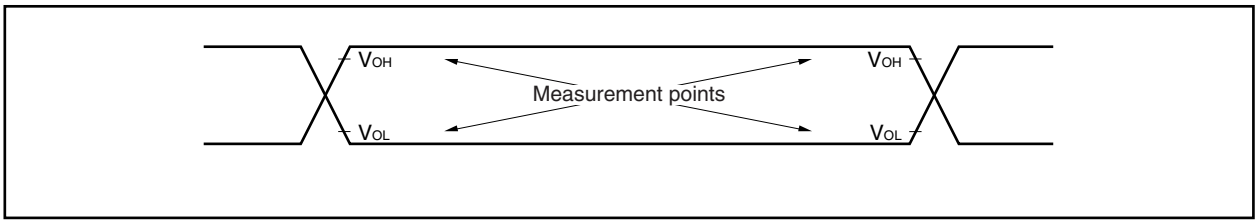
Downloaded from Arrow.com.

AC Characteristics

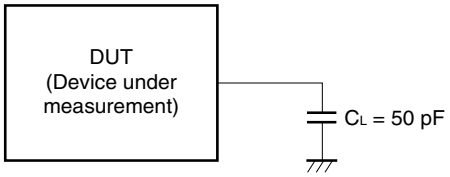
AC Test Input Measurement Points (V_{DD}, AV_{REF0}, EV_{DD})



AC Test Output Measurement Points



Load Conditions



Caution If the load capacitance exceeds 50 pF due to the circuit configuration, bring the load capacitance of the device to 50 pF or less by inserting a buffer or by some other means.

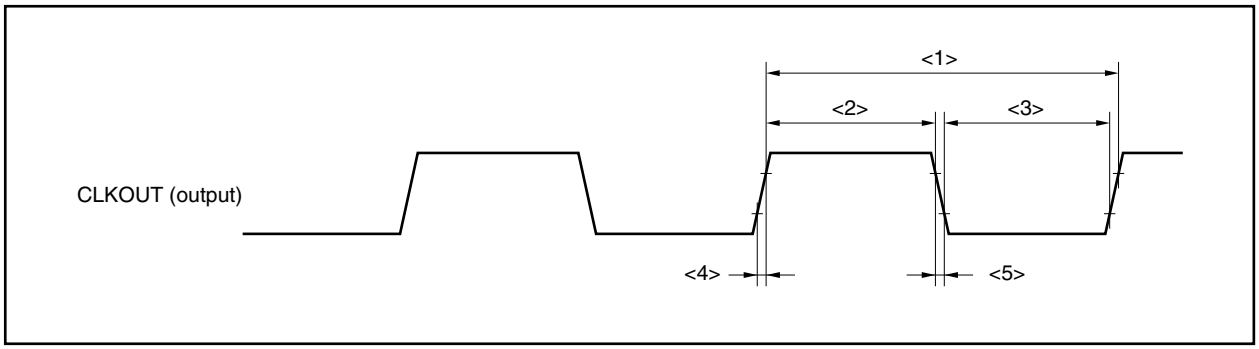
Downloaded from Arrow.com.

CLKOUT Output Timing

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Output cycle	t_{CYK}	<1>		50 ns	30.6 μs	
High-level width	t_{WKH}	<2>	V_{DD} = 4.0 to 5.5 V	t_{CYK}/2 – 17		ns
			V_{DD} = 2.7 to 5.5 V	t_{CYK}/2 – 26		ns
Low-level width	t_{WKL}	<3>	V_{DD} = 4.0 to 5.5 V	t_{CYK}/2 – 17		ns
			V_{DD} = 2.7 to 5.5 V	t_{CYK}/2 – 26		ns
Rise time	t_{KR}	<4>	V_{DD} = 4.0 to 5.5 V		17	ns
			V_{DD} = 2.7 to 5.5 V		26	ns
Fall time	t_{KF}	<5>	V_{DD} = 4.0 to 5.5 V		17	ns
			V_{DD} = 2.7 to 5.5 V		26	ns

Clock Timing



Downloaded from Arrow.com.

Basic Operation

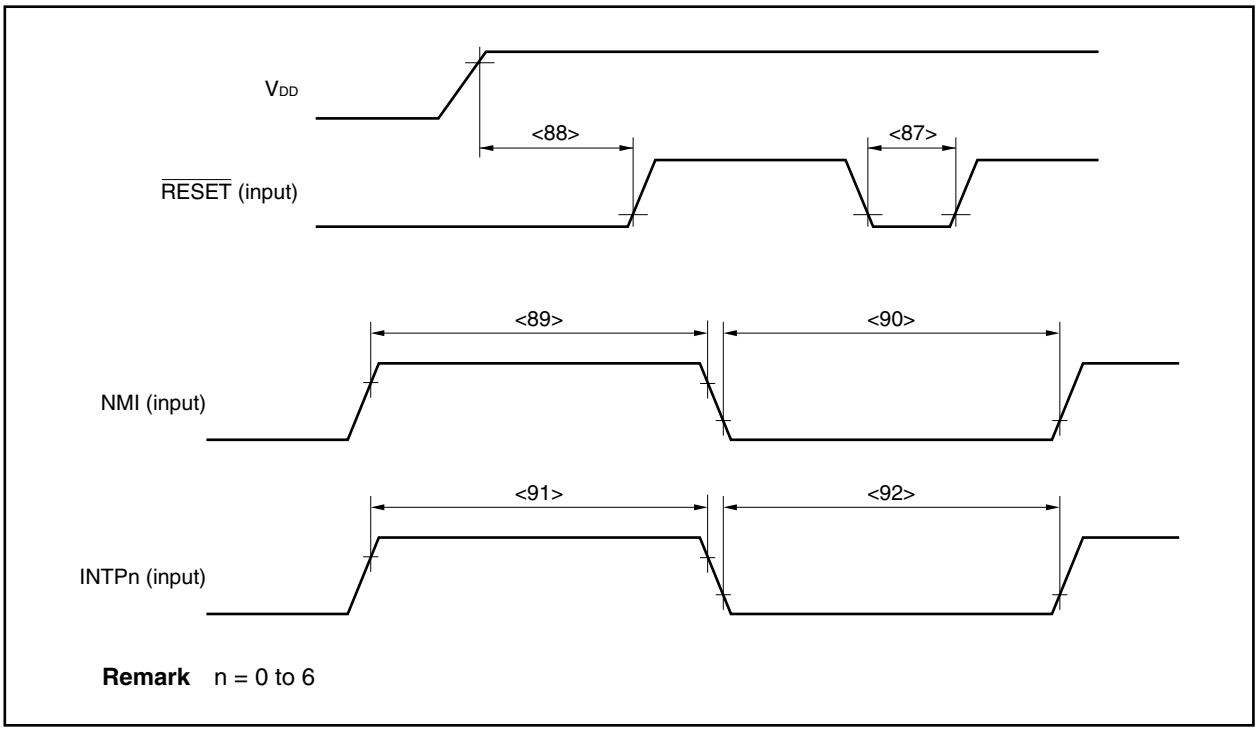
(1) Reset/external interrupt timing

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{\text{RESET}}$ low-level width	t_{WRSL1}	<87>	Reset in power-on status	2		μs
	t_{WRSL2}	<88>	Power-on-reset	2		μs
NMI high-level width	t_{WNIH}	<89>	Analog noise elimination	1		μs
NMI low-level width	t_{WNIL}	<90>	Analog noise elimination	1		μs
INTPn high-level width	t_{WITH}	<91>	n = 0 to 6 (analog noise elimination)	600		ns
INTPn low-level width	t_{WITL}	<92>	n = 0 to 6 (analog noise elimination)	600		ns

Remark The above specification shows the pulse width that is accurately detected as a valid edge. If a pulse narrower than the above specification is input, therefore, it may also be detected as a valid edge.

Reset/Interrupt



Remark n = 0 to 6

Downloaded from Arrow.com.

Timer Timing

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

	Parameter	Symbol		Conditions	MIN.	MAX.	Unit
<R>	TI01n high-level width	t_{TI0H}	<93>	V_{DD} = 4.5 to 5.5 V	$2T_{smp0}$ + 100 Note 1		ns
				V_{DD} = 2.7 to 5.5 V	$2T_{smp0}$ + 200 Note 1		ns
<R>	TI01n low-level width	t_{TI0L}	<94>	V_{DD} = 4.5 to 5.5 V	$2T_{smp0}$ + 100 Note 1		ns
				V_{DD} = 2.7 to 5.5 V	$2T_{smp0}$ + 200 Note 1		ns
<R>	TI5n high-level width	t_{TI5H}	<95>	V_{DD} = 4.5 to 5.5 V	50		ns
				V_{DD} = 2.7 to 5.5 V	100		ns
<R>	TI5n low-level width	t_{TI5L}	<96>	V_{DD} = 4.5 to 5.5 V	50		ns
				V_{DD} = 2.7 to 5.5 V	100		ns
<R>	TIP0n high-level width	t_{TIPH}	<97>	V_{DD} = 4.5 to 5.5 V	np × T_{smpp} + 100 Note 2		ns
				V_{DD} = 2.7 to 5.5 V	np × T_{smpp} + 200 Note 2		ns
<R>	TIP0n low-level width	t_{TIPL}	<98>	V_{DD} = 4.5 to 5.5 V	np × T_{smpp} + 100 Note 2		ns
				V_{DD} = 2.7 to 5.5 V	np × T_{smpp} + 200 Note 2		ns

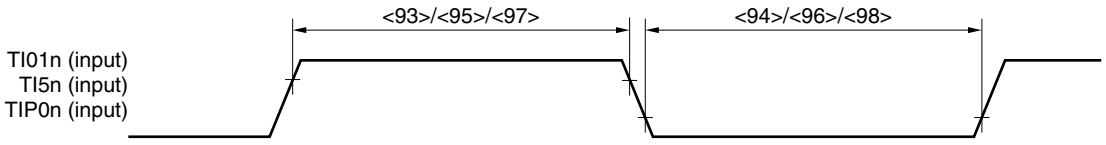
Notes 1. T_{smp0}: Timer 0 count clock cycle
However, T_{smp0} = 4/fxx when TI010 is used as an external clock.

2. np: Number of sampling clocks set by the PnNFC.PnNFSTS bit
T_{smpp}: Digital noise elimination sampling clock cycle of TIP0n pin
If TIP00 is used as an external event count input or an external trigger input, however, T_{smpp} = 0 (digital noise is not eliminated).

Remarks 1. n = 0, 1

2. The above specification shows the pulse width that is accurately detected as a valid edge. If a pulse narrower than the above specification is input, therefore, it may also be detected as a valid edge.

Timer Input Timing



Remark n = 0, 1

UART Timing

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

	Parameter	Symbol	Conditions	MIN.	MAX.	Unit
	Transmit rate				312.5	kbps
<R>	ASCK0 frequency		V_{DD} = 4.5 to 5.5 V		12	MHz
			V_{DD} = 2.7 to 5.5 V		6	MHz

Downloaded from Arrow.com.

CSI0 Timing

(1) Master mode

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{SCK0n}$ cycle time	t_{KCY1}	<99>	V_{DD} = 4.0 to 5.5 V	200		ns
			V_{DD} = 2.7 to 5.5 V	400		ns
$\overline{SCK0n}$ high-/low-level width	t_{KH1}, t_{KL1}	<100>		t_{KCY1}/2 – 30		ns
SI0n setup time (to $\overline{SCK0n}$)	t_{SIK1}	<101>	V_{DD} = 4.0 to 5.5 V	30		ns
			V_{DD} = 2.7 to 5.5 V	50		ns
SI0n hold time (from $\overline{SCK0n}$)	t_{KSI1}	<102>	V_{DD} = 4.0 to 5.5 V	30		ns
			V_{DD} = 2.7 to 5.5 V	50		ns
Delay time from $\overline{SCK0n}$ to SO0n output	t_{KSO1}	<103>	V_{DD} = 4.0 to 5.5 V		30	ns
			V_{DD} = 2.7 to 5.5 V		60	ns

Remark n = 0, 1

(2) Slave mode

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

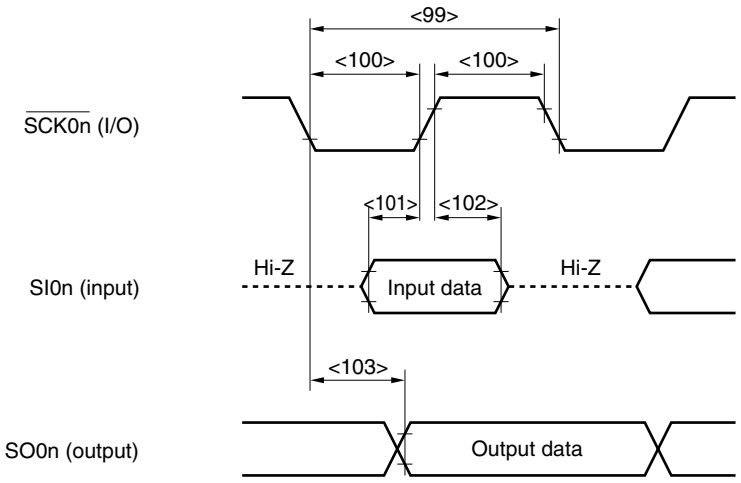
Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{SCK0n}$ cycle time	t_{KCY2}	<99>	V_{DD} = 4.0 to 5.5 V	200		ns
			V_{DD} = 2.7 to 5.5 V	400		ns
$\overline{SCK0n}$ high-/low-level width	t_{KH2}, t_{KL2}	<100>	V_{DD} = 4.0 to 5.5 V	45		ns
			V_{DD} = 2.7 to 5.5 V	90		ns
SI0n setup time (to $\overline{SCK0n}$)	t_{SIK2}	<101>	V_{DD} = 4.0 to 5.5 V	30		ns
			V_{DD} = 2.7 to 5.5 V	60		ns
SI0n hold time (from $\overline{SCK0n}$)	t_{KSI2}	<102>	V_{DD} = 4.0 to 5.5 V	30		ns
			V_{DD} = 2.7 to 5.5 V	60		ns
Delay time from $\overline{SCK0n}$ to SO0n output	t_{KSO2}	<103>	V_{DD} = 4.0 to 5.5 V		50	ns
			V_{DD} = 2.7 to 5.5 V		100	ns

Remark n = 0, 1

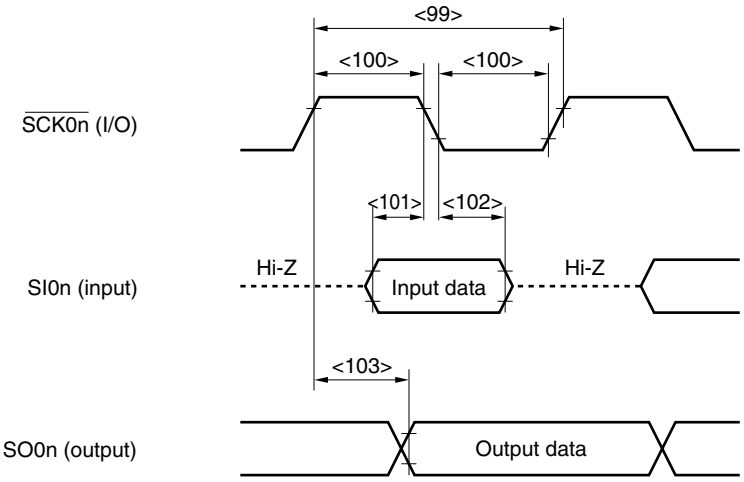
Downloaded from Arrow.com.

<R>

(a) CSICn.CKPn, DAPn bits = 00 or 11



(b) CSICn.CKPn, DAPn bits = 01 or 10



Remark n = 0, 1

Downloaded from Arrow.com.

I^2C Bus Mode (μPD703207Y, 70F3207HY Only)

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

Parameter		Symbol		Normal Mode		High-Speed Mode		Unit
				MIN.	MAX.	MIN.	MAX.	
SCL0 clock frequency		f_{CLK}		0	100	0	400	kHz
Bus free time (Between start and stop conditions)		t_{BUF}	<104>	4.7	–	1.3	–	μs
Hold time[Note 1]		$t_{HD:STA}$	<105>	4.0	–	0.6	–	μs
SCL0 clock low-level width		t_{LOW}	<106>	4.7	–	1.3	–	μs
SCL0 clock high-level width		t_{HIGH}	<107>	4.0	–	0.6	–	μs
Setup time for start/restart conditions		$t_{SU:STA}$	<108>	4.7	–	0.6	–	μs
Data hold time	CBUS compatible master	$t_{HD:DAT}$	<109>	5.0	–	–	–	μs
	I^2C mode			0[Note 2]	–	0[Note 2]	0.9[Note 3]	μs
Data setup time		$t_{SU:DAT}$	<110>	250	–	100[Note 4]	–	ns
SDA0 and SCL0 signal rise time		t_R	<111>	–	1000	20 + 0.1Cb[Note 5]	300	ns
SDA0 and SCL0 signal fall time		t_F	<112>	–	300	20 + 0.1Cb[Note 5]	300	ns
Stop condition setup time		$t_{SU:STO}$	<113>	4.0	–	0.6	–	μs
Pulse width of spike suppressed by input filter		t_{SP}	<114>	–	–	0	50	ns
Capacitance load of each bus line		Cb		–	400	–	400	pF

Notes 1. At the start condition, the first clock pulse is generated after the hold time.

2. The system requires a minimum of 300 ns hold time internally for the SDA0 signal (at $V_{IHmin.}$ of SCL0 signal) in order to occupy the undefined area at the falling edge of SCL0.

3. If the system does not extend the SCL0 signal low hold time (t_{LOW}), only the maximum data hold time ($t_{HD:DAT}$) needs to be satisfied.

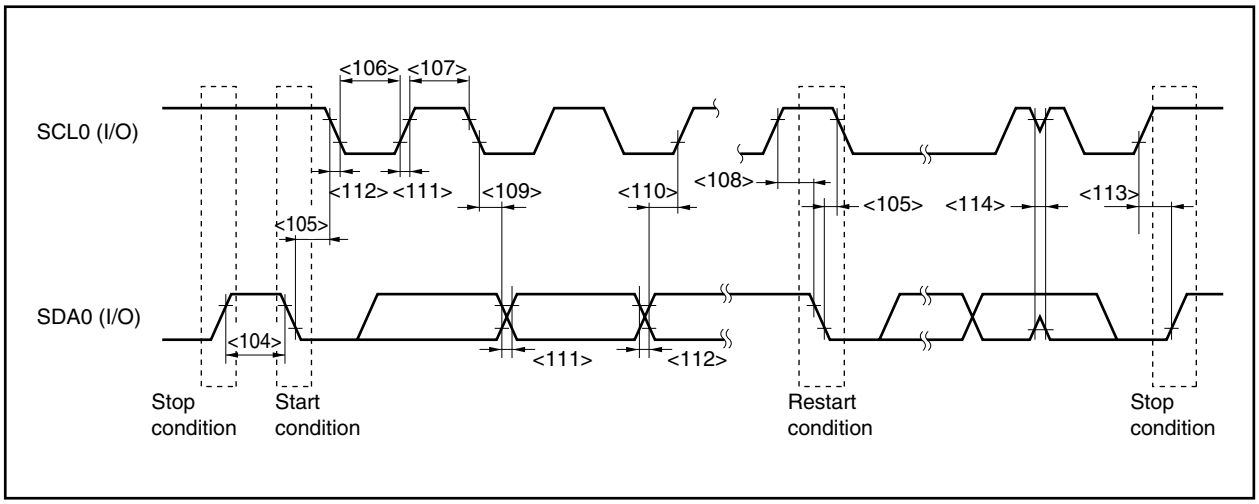
4. The high-speed mode I^2C bus can be used in the normal-mode I^2C bus system. In this case, set the high-speed mode I^2C bus so that it meets the following conditions.

- If the system does not extend the SCL0 signal's low state hold time:
  $t_{SU:DAT} \geq 250$ ns
- If the system extends the SCL0 signal's low state hold time:
  Transmit the following data bit to the SDA0 line prior to the SCL0 line release ($t_{Rmax.} + t_{SU:DAT}$ = 1000 + 250 = 1250 ns: Normal mode I^2C bus specification).

5. Cb: Total capacitance of one bus line (unit: pF)

Downloaded from Arrow.com.

I²C Bus Mode (μPD703207Y, 70F3207HY Only)



Downloaded from Arrow.com.

A/D Converter

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution			10	10	10	bit
Overall error[Note 1]	AINL	4.0 ≤ AV_{REF0} ≤ 5.5 V		±0.2	±0.4	%FSR
		2.7 ≤ AV_{REF0} ≤ 4.0 V		±0.3	±0.6	%FSR
Conversion time	t_{CONV}	4.0 ≤ AV_{REF0} ≤ 5.5 V	14		100	μs
		2.7 ≤ AV_{REF0} ≤ 4.0 V	17		100	μs
Zero-scale error[Note 1]	E_{ZS}	4.0 ≤ AV_{REF0} ≤ 5.5 V			±0.4	%FSR
		2.7 ≤ AV_{REF0} ≤ 4.0 V			±0.6	%FSR
Full-scale error[Note 1]	E_{fs}	4.0 ≤ AV_{REF0} ≤ 5.5 V			±0.4	%FSR
		2.7 ≤ AV_{REF0} ≤ 4.0 V			±0.6	%FSR
Non-linearity error[Note 2]	ILE	4.0 ≤ AV_{REF0} ≤ 5.5 V			±2.5	LSB
		2.7 ≤ AV_{REF0} ≤ 4.0 V			±4.5	LSB
Differential linearity error[Note 2]	DLE	4.0 ≤ AV_{REF0} ≤ 5.5 V			±1.5	LSB
		2.7 ≤ AV_{REF0} ≤ 4.0 V			±2.0	LSB
Analog input voltage	V_{IAN}		0		AV_{REF0}	V
AV_{REF0} current	IA_{REF0}	When using A/D converter		1.3	2.5	mA
<R>		When not using A/D converter[Note 3]		1.0	10	μA

Notes 1. Excluding quantization error (±0.05 %FSR).

2. Excluding quantization error (±0.5 LSB).

3. ADM.ADCS bit = 0, ADM.ADCS2 bit = 0

Remark LSB: Least Significant Bit
FSR: Full Scale Range

Downloaded from Arrow.com.

Flash Memory Programming Characteristics

(T_A = –40 to +85°C, V_{DD} = EV_{DD} = AV_{REF0} = 2.7 to 5.5 V, V_{SS} = EV_{SS} = AV_{SS} = 0 V, C_L = 50 pF)

(1) Basic characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Programming operation frequency		V_{DD} = 4.5 to 5.5 V	2		20	MHz
		V_{DD} = 4.0 to 5.5 V	2		16	MHz
<R>		V_{DD} = 2.7 to 5.5 V	2		10	MHz
Supply voltage	V_{DD}		2.7		5.5	V
Number of rewrites	C_{ERWR}	**Note**	100			Times
<R> Programming temperature	t_{PRG}		–40		+85	°C

Note When writing initially to shipped products, it is counted as one rewrite for both “erase to write” and “write only”.

Example (P: Write, E: Erase)

Shipped product ——→P→E→P→E→P: 3 rewrites

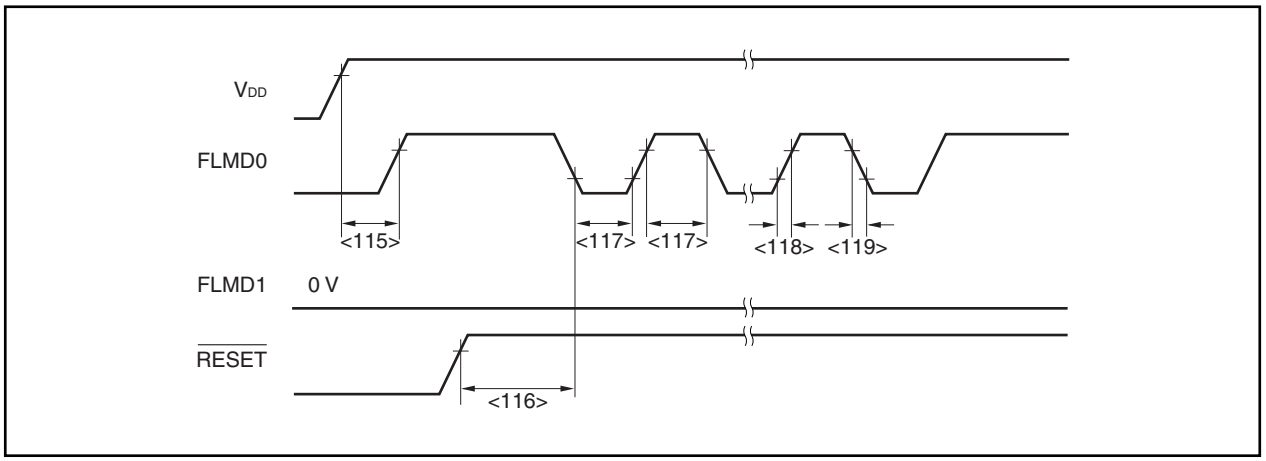
Shipped product → E →P→E→P→E→P: 3 rewrites

<R> (2) Serial write operation characteristics

Parameter	Symbol		Conditions	MIN.	TYP.	MAX.	Unit
Setup time from V_{DD}↑ to FLMD0↑	t_{DP}	<115>		10 ms		3 s	
Time from $\overline{\text{RESET}}$↑ to FLMD0 pulse input start	t_{RP}	<116>		66611.2/fx			s
FLMD0 pulse high-/low-level width	t_{PW}	<117>		10		100	μs
FLMD0 pulse rise time	t_R	<118>				50	ns
FLMD0 pulse fall time	t_F	<119>				50	ns

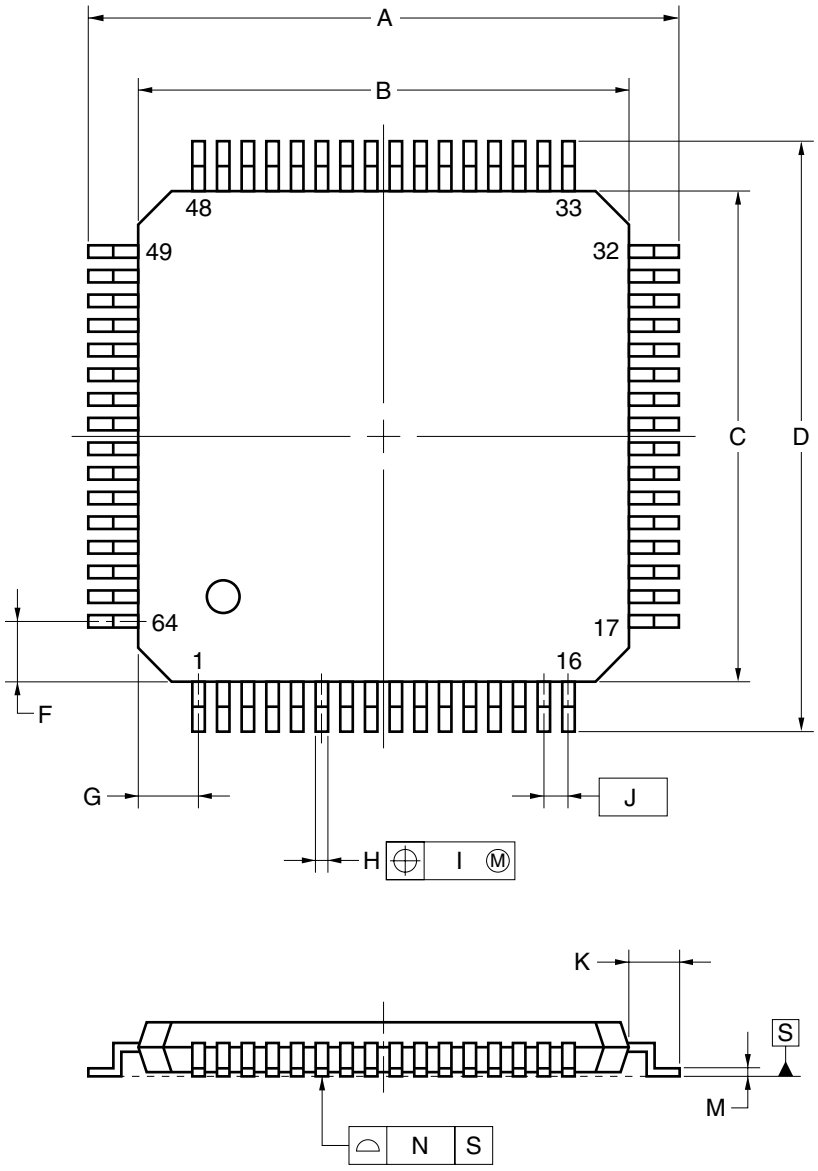
Remark fx: Main clock oscillation frequency

Serial Write Operation Timing (CSI00, CSI00-HS)

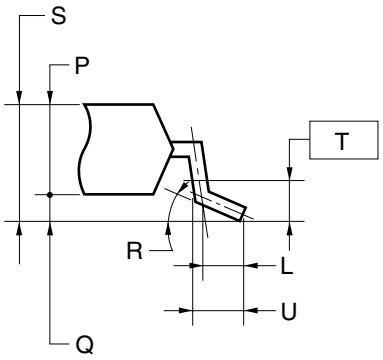


Downloaded from Arrow.com.

64-PIN PLASTIC TQFP (12x12)



detail of lead end



NOTE
Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

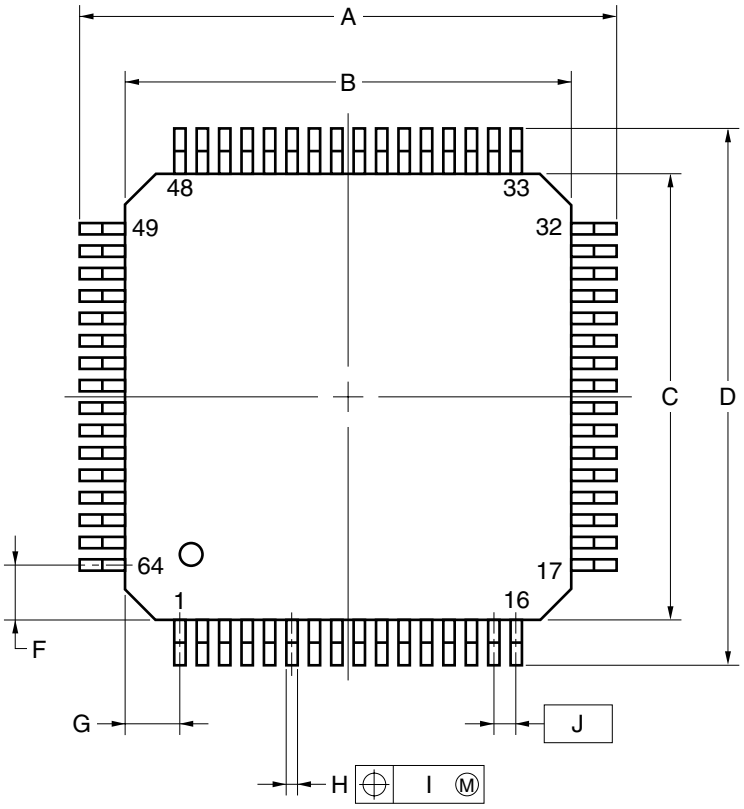
ITEM	MILLIMETERS
A	14.0±0.2
B	12.0±0.2
C	12.0±0.2
D	14.0±0.2
F	1.125
G	1.125
H	$0.32^{+0.06}_{-0.10}$
I	0.13
J	0.65 (T.P.)
K	1.0±0.2
L	0.5
M	$0.17^{+0.03}_{-0.07}$
N	0.10
P	1.0
Q	0.1±0.05
R	$3^{\circ}{}^{+4^{\circ}}_{-3^{\circ}}$
S	1.1±0.1
T	0.25
U	0.6±0.15

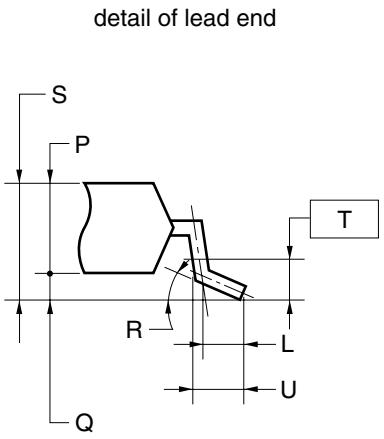
P64GK-65-9ET-3

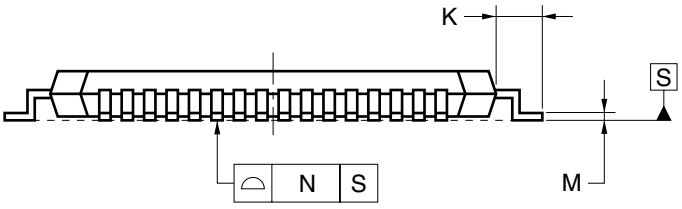
Remark The external dimensions and materials of the ES version are the same as those of the mass-produced version.

Downloaded from Arrow.com.

64-PIN PLASTIC LQFP (10x10)







NOTE

Each lead centerline is located within 0.08 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	12.0±0.2
B	10.0±0.2
C	10.0±0.2
D	12.0±0.2
F	1.25
G	1.25
H	0.22±0.05
I	0.08
J	0.5 (T.P.)
K	1.0±0.2
L	0.5
M	$0.17^{+0.03}_{-0.07}$
N	0.08
P	1.4
Q	0.1±0.05
R	$3^{\circ}{}^{+4^{\circ}}_{-3^{\circ}}$
S	1.5±0.10
T	0.25
U	0.6±0.15

S64GB-50-8EU-2

Remark The external dimensions and materials of the ES version are the same as those of the mass-produced version.

Downloaded from Arrow.com.

<R>

CHAPTER 26 RECOMMENDED SOLDERING CONDITIONS

The V850ES/KE1 should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (http://www.necel.com/pkg/en/mount/index.html)

Table 26-1. Surface Mounting Type Soldering Conditions (1/2)

(1) μPD703207GK-xxx-9ET-A: 64-pin plastic TQFP (12 × 12)
μPD703207YGK-xxx-9ET-A: 64-pin plastic TQFP (12 × 12)
μPD70F3207GK-9ET-A: 64-pin plastic TQFP (12 × 12)
μPD70F3207YGK-9ET-A: 64-pin plastic TQFP (12 × 12)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: Three times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 20 to 72 hours)	IR60-207-3
Wave soldering	Can be supported. For details, consult an NEC Electronics sales representative.	–
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

Caution Do not use different soldering methods together (except for partial heating).

Remarks 1. Products with -A at the end of the part number are lead-free products.

2. For soldering methods and conditions other than those recommended above, consult an NEC Electronics sales representative.

Downloaded from Arrow.com.

Table 26-1. Surface Mounting Type Soldering Conditions (2/2)

(2) μPD703207GB-xxx-8EU-A: 64-pin plastic LQFP (fine pitch) (10 × 10)
μPD703207YGB-xxx-8EU-A: 64-pin plastic LQFP (fine pitch) (10 × 10)
μPD70F3207GB-8EU-A: 64-pin plastic LQFP (fine pitch) (10 × 10)
μPD70F3207YGB-8EU-A: 64-pin plastic LQFP (fine pitch) (10 × 10)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: Three times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 20 to 72 hours)	IR60-207-3
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

Caution Do not use different soldering methods together (except for partial heating).

Remarks 1. Products with -A at the end of the part number are lead-free products.

2. For soldering methods and conditions other than those recommended above, consult an NEC Electronics sales representative.

Downloaded from Arrow.com.

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the V850ES/KE1. Figure A-1 shows the development tool configuration.

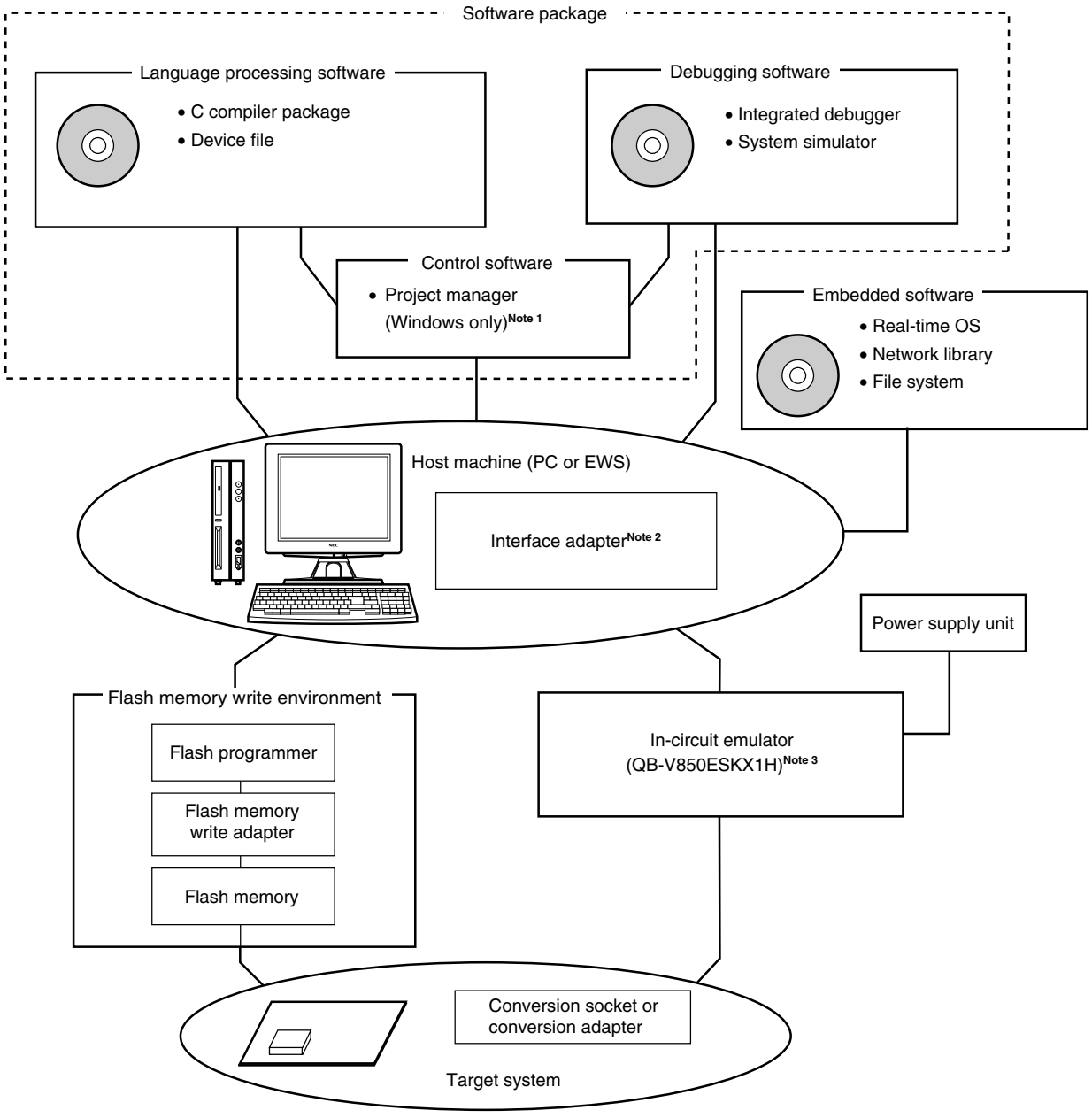
- **Support for PC98-NX series**
  Unless otherwise specified, products supported by IBM PC/AT™ compatibles are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the explanation for IBM PC/AT compatibles.

- **Windows™**
  Unless otherwise specified, “Windows” means the following OSs.
  - Windows 98, 2000
  - Windows Me
  - Windows XP
  - Windows NT™ Ver. 4.0

Downloaded from Arrow.com.

Figure A-1. Development Tool Configuration (1/2)

(1) When using IECUBE® (QB-V850ESKX1H)



Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. QB-V850ESKX1H supports USB only.

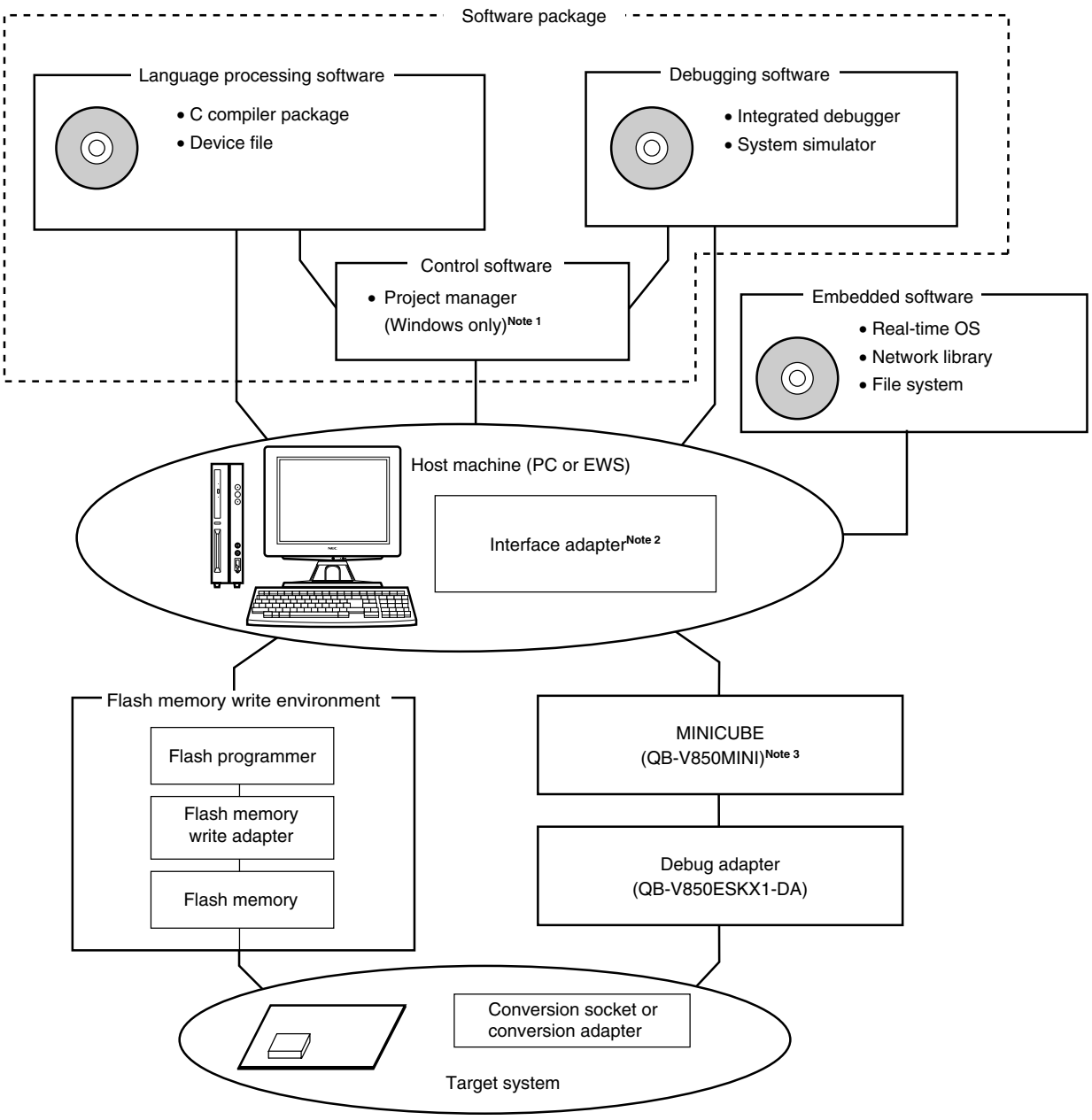
3. QB-V850ESKX1H is supplied with ID850QB, a flash memory programmer (PG-FPL), power supply unit, and a USB interface adapter. Any other products are sold separately.

Downloaded from Arrow.com.

<R>

Figure A-1. Development Tool Configuration (2/2)

(2) When using MINICUBE (QB-V850MINI)



Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. QB-V850MINI supports USB interface only.

3. QB-V850MINI is supplied with integrated debugger ID850QB, USB interface cable, OCD cable, self check board, KEL adapter, and KEL connector. Any other products are sold separately.

Downloaded from Arrow.com.

A.1 Software Package

SP850 V850 Series software package	Development tools (software) common to the V850 Series are combined in this package.
	Part number: μS××××SP850

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SP850

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

A.2 Language Processing Software

CA850 C compiler package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler is started from project manager PM+.
	Part number: μS××××CA703000
<R> DF703218 Device file	This file contains information peculiar to the device. This device file should be used in combination with a tool (CA850 and ID850QB). The corresponding OS and host machine differ depending on the tool to be used.

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××CA703000

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

A.3 Control Software

PM+ Project manager	This is control software designed to enable efficient user program development in the Windows environment. All operations used in development of a user program, such as starting the editor, building, and starting the debugger, can be performed from the PM+. **<Caution>** The PM+ is included in the C compiler package CA850. It can only be used in Windows.

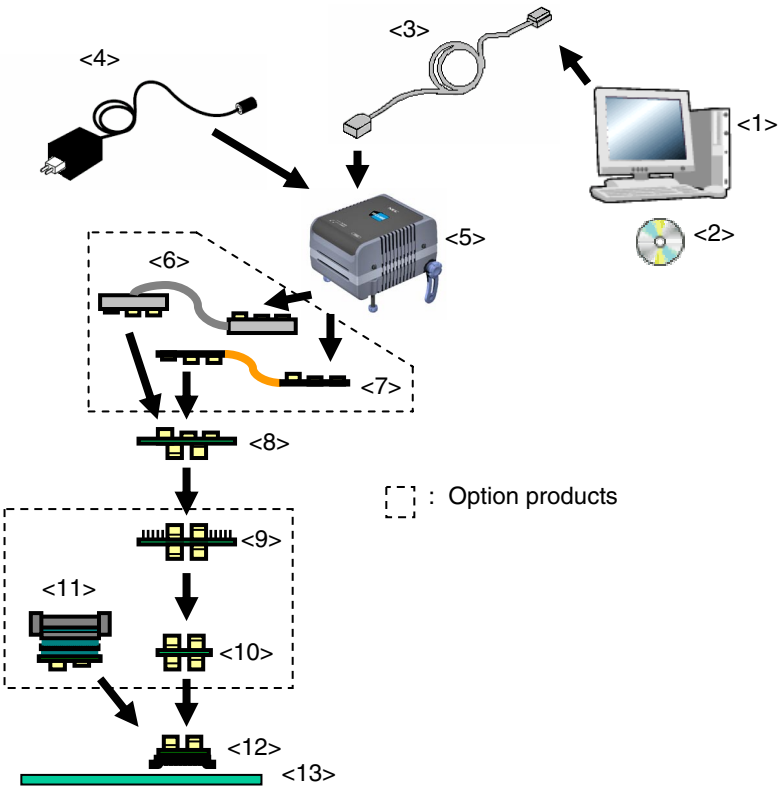
Downloaded from Arrow.com.

<R> A.4 Debugging Tools (Hardware)

A.4.1 When using IECUBE QB-V850ESKX1H

The system configuration when connecting the QB-V850ESKX1H to the host machine (PC-9821 series, PC/AT compatible) is shown below. They can still be connected without the optional products.

Figure A-2. System Configuration (QB-V850ESKX1H Used) (1/2)



<1> Host machine (PC-9821 series, IBM PC/AT compatibles)
<2> Debugger, USB driver, manuals, etc. (ID850QB Disk, Accessory Disk[Note 1])
<3> USB interface cable
<4> AC adapter
<5> In-circuit emulator (QB-V850ESKX1H)
<6> Extension probe coaxial type (QB-144-EP-01S) (option)
<7> Extension probe flexible type[Note 2] (QB-144-EP-02S) (option)
<8> Exchange adapter[Note 3] (QB-64-EA-01S)
<9> Check pin adapter[Note 4] (QB-64-CA-01S) (option)
<10> Space adapter[Note 4] (QB-64-SA-01S) (option)
<11> Mount adapter (QB-64GK-MA-01S (for GK package), QB-64GB-MA-01S (for GB package)) (option)
<12> Target connector[Note 3] (QB-64GK-TC-01S (for GK package), QB-64GB-TC-01S (for GB package))
<13> Target system

Downloaded from Arrow.com.

Figure A-2. System Configuration (QB-V850ESKX1H Used) (2/2)

Notes 1. Obtain the device file from the NEC Electronics website.
http://www.necel.com/micro/ods/eng/index.html

2. Under development

3. Depending on the ordering number, supplied with the device.
- When QB-V850ESKX1H-ZZZ is ordered
  The exchange adapter and the target connector are not supplied.
- When QB-V850ESKX1H-S64GK is ordered
  The QB-64-EA-01S and QB-64GK-TC-01S are supplied.
- When QB-V850ESKX1H-S64GB is ordered
  The QB-64-EA-01S and QB-64GB-TC-01S are supplied.

4. When using both <9> and <10>, the order between <9> and <10> is not cared.

<5> QB-V850ESKX1H[Note] In-circuit emulator	The in-circuit emulator serves to debug hardware and software when developing application systems using a V850ES/KE1 product. It corresponds to the integrated debugger ID850QB. This emulator should be used in combination with a power supply unit and emulation probe. Use USB to connect this emulator to the host machine.
<3> USB interface cable	Cable to connect the host machine and the QB-V850ESKX1H.
<4> AC adapter	100 to 240 V can be supported by replacing the AC plug.
<8> QB-64-EA-01S Exchange adapter	Adapter to perform pin conversion.
<9> QB-64-CA-01S Check pin adapter	Adapter used in waveform monitoring using the oscilloscope, etc.
<10> QB-64-SA-01S Space adapter	Adapter to adjust the height.
<11> QB-64GK-MA-01S, QB-64GB-MA-01S Mount adapter	Adapter to mount the V850ES/KE1 with socket. • QB-64GK-MA-01S: For 64-pin plastic TQFP (GK-9ET type) • QB-64GB-MA-01S: For 64-pin plastic LQFP (GB-8EU type)
<12> QB-64GK-TC-01S, QB-64GB-TC-01S Target connector	Connector to solder on the target system. • QB-64GK-TC-01S: For 64-pin plastic TQFP (GK-9ET type) • QB-64GB-TC-01S: For 64-pin plastic LQFP (GB-8EU type)

Note QB-V850ESKX1H is supplied with a power supply unit, USB interface cable, and flash memory programmer (PG-FPL). It is also supplied with integrated debugger ID850QB as control software.

Remark The numbers in the square brackets correspond to the numbers in Figure A-2.

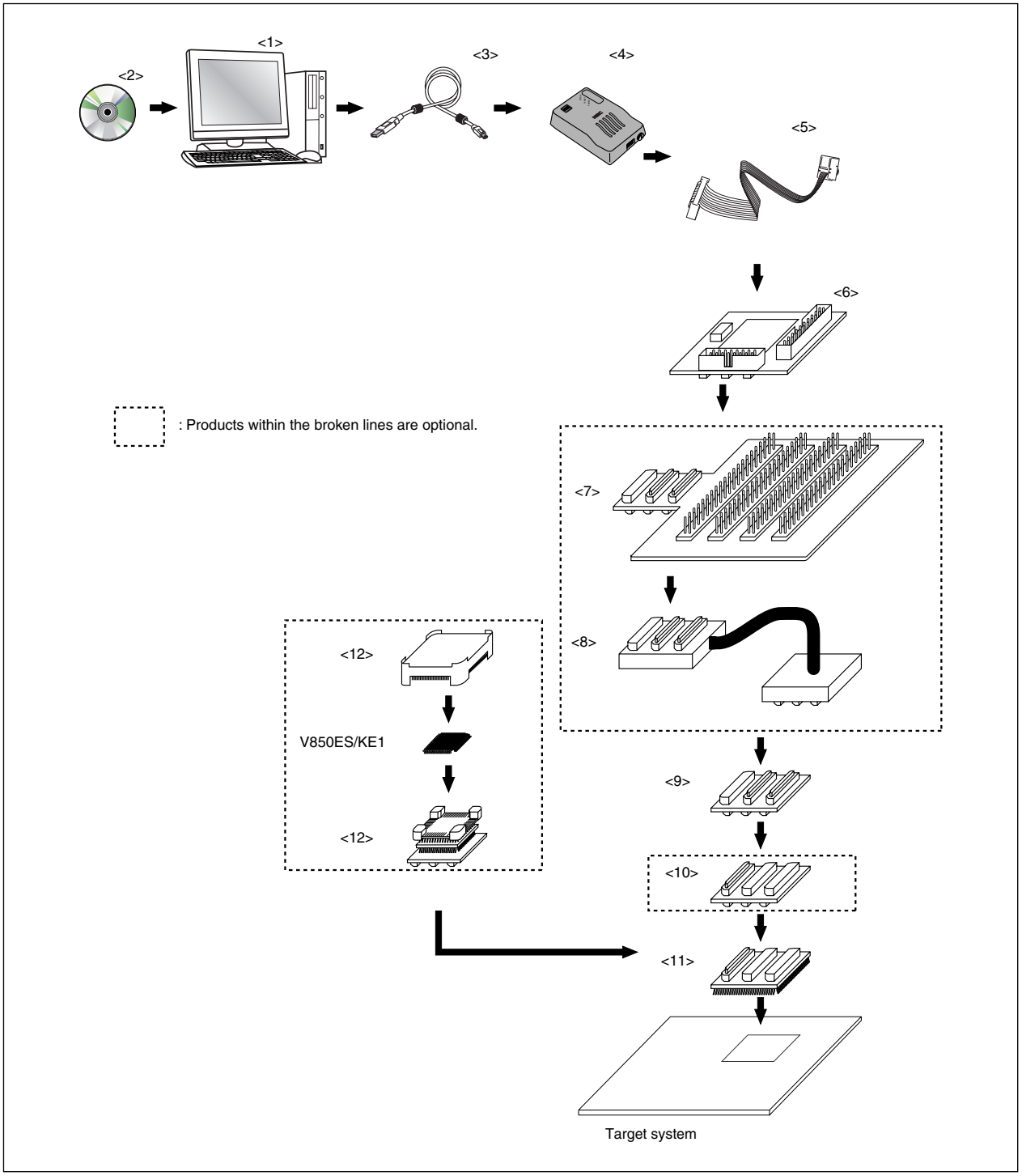
Downloaded from Arrow.com.

A.4.2 When using MINICUBE QB-V850MINI

(1) Debug emulation using MINICUBE and QB-V850ESKX1-DA

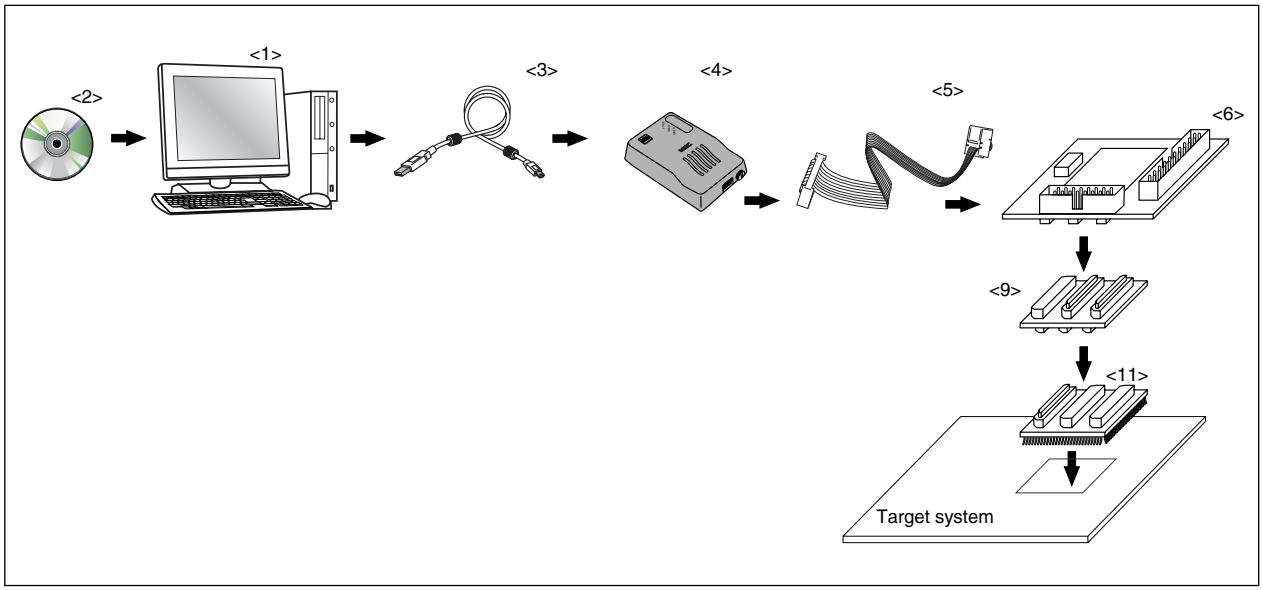
The following shows the system configuration when using MINICUBE, debug adapter QB-V850ESKX1-DA, and the host machine (PC-9821 series, PC/AT compatibles) connected together. They can still be connected without the optional products.

Figure A-3. Example of System Configuration Using QB-V850ESKX1-DA (Using Optional Products)



Downloaded from Arrow.com.

Figure A-4. Example of System Configuration Using QB-V850ESKX1-DA (Not Using Optional Products)



<1> Host machine	PC with USB ports
<2> CD-ROM[Note]	This CD-ROM is provided with MINICUBE and includes an ID850QB integrated debugger, N-Wire checker, a device driver, and documents.
<3> USB interface cable	This cable, supporting USB, is provided with MINICUBE and used to connect the host machine to MINICUBE. The length of this cable is approximately 2 m.
<4> MINICUBE On-chip debug emulator	This on-chip emulator is used to debug hardware and software when developing the application systems using the V850ES/KE1. This supports the ID850QB integrated debugger.
<5> OCD cable	This cable is used to connect MINICUBE to the target system. This is provided with MINICUBE. The length of this cable is approximately 20 cm.
<6> QB-V850ESKX1-DA Debug adapter	This operates as an in-circuit emulator by using in combination with MINICUBE. This is provided with MINICUBE.
<7> QB-144-CA-01S (optional) Check pin adapter	Adapter used when monitoring waveforms on an oscilloscope.
<8> QB-144-EP-01S (optional) Extension probe coaxial type	This probe is used to connect the QB-V850ESKX1-DA to the exchange adapter. The length of this cable is approximately 40 cm.
<9> QB-64-EA-01S Exchange adapter	Adapter that performs pin conversion
<10> QB-64-SA-01S (optional) Space adapter	Adapter for height regulation
<11> QB-64GK-TC-01S, QB-64GB-TC-01S Target connector	Connector soldered to target system • QB-64GK-TC-01S: For 64-pin plastic TQFP (GK-9ET type) • QB-64GB-TC-01S: For 64-pin plastic LQFP (GB-8EU type)
<12> QB-64GK-MA-01S (optional), QB-64GB-MA-01S (optional) Mount adapter	Adapter for socket mounting of the V850ES/KE1 • QB-64GK-MA-01S: For 64-pin plastic TQFP (GK-9ET type) • QB-64GB-MA-01S: For 64-pin plastic LQFP (GB-8EU type)

Note Obtain device files from the NEC Electronics website.
http://www.necel.com/micro/ods/eng/index_e.html

Remark The numbers shown in the above table correspond to the numbers in Figures A-3 and A-4.

Downloaded from Arrow.com.

A.5 Debugging Tools (Software)

ID850QB Integrated debugger	This debugger supports the in-circuit emulators for the V850 Series. The ID850QB is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with the device file.
	Part number: μS××××ID703000-QB

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××ID703000-QB

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

Downloaded from Arrow.com.

A.6 Embedded Software

RX850, RX850 Pro Real-time OS	The RX850 and RX850 Pro are real-time OSs conforming to μITRON 3.0 specifications. A tool (configurator) for generating multiple information tables is supplied. RX850 Pro has more functions than RX850.
	Part number: μS××××RX703000-ΔΔΔΔ (RX850) μS××××RX703100-ΔΔΔΔ (RX850 Pro)
V850mini-NET (provisional name) (Network library)	This is a network library conforming to RFC. It is a lightweight TCP/IP of compact design, requiring only a small memory. In addition to the TCP/IP standard set, an HTTP server, SMTP client, and POP client are also supported.
RX-FS850 (File system)	This is a FAT file system function. It is a file system that supports the CD-ROM file system function. This file system is used with the real-time OS RX850 Pro.

Caution To purchase the RX850 or RX850 Pro, first fill in the purchase application form and sign the user agreement.

Remark ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××RX703000-ΔΔΔΔ
μS××××RX703100-ΔΔΔΔ

ΔΔΔΔ	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Do not use for mass-produced product.
100K	Mass-production object	0.1 million units
001M		1 million units
010M		10 million units
S01	Source program	Object source program for mass production

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3K17	SPARCstation	Solaris (Rel. 2.5.1)	

A.7 Flash Memory Writing Tools

Flashpro IV (part number: PG-FP4) Flash programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-64GK-9ET-A Flash memory writing adapter	Flash memory writing adapter used connected to the Flashpro IV. • FA-64GK-9ET-A: For 64-pin plastic TQFP (GK-9ET type)
FA-64GB-8EU-A Flash memory writing adapter	Flash memory writing adapter used connected to the Flashpro IV. • FA-64GB-8EU-A: For 64-pin plastic LQFP (fine pitch) (GB-8EU type)

Remark FA-64GK-9ET-A and FA-64GB-8EU-A are products of Naito Densei Machida Mfg. Co., Ltd.
TEL: +81-45-475-4191 Naito Densei Machida Mfg. Co., Ltd.

Downloaded from Arrow.com.

APPENDIX B INSTRUCTION SET LIST

B.1 Conventions

(1) Register symbols used to describe operands

Register Symbol	Explanation
reg1	General-purpose registers: Used as source registers.
reg2	General-purpose registers: Used mainly as destination registers. Also used as source register in some instructions.
reg3	General-purpose registers: Used mainly to store the remainders of division results and the higher 32 bits of multiplication results.
bit#3	3-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 condition codes
sp	Stack pointer (r3)
ep	Element pointer (r30)
listX	X item register list

(2) 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

Downloaded from Arrow.com.

(3) Register symbols used in operations

Register Symbol	Explanation
←	Input for
GR [ ]	General-purpose 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)
Halfword	Halfword (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

(4) Register symbols used in 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).

Downloaded from Arrow.com.

(5) 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.

(6) Condition codes

Condition Code (cccc)	Condition Formula	Explanation
0 0 0 0	OV = 1	Overflow
1 0 0 0	OV = 0	No overflow
0 0 0 1	CY = 1	Carry Lower (Less than)
1 0 0 1	CY = 0	No carry Not lower (Greater than or equal)
0 0 1 0	Z = 1	Zero
1 0 1 0	Z = 0	Not zero
0 0 1 1	(CY or Z) = 1	Not higher (Less than or equal)
1 0 1 1	(CY or Z) = 0	Higher (Greater than)
0 1 0 0	S = 1	Negative
1 1 0 0	S = 0	Positive
0 1 0 1	–	Always (Unconditional)
1 1 0 1	SAT = 1	Saturated
0 1 1 0	(S xor OV) = 1	Less than signed
1 1 1 0	(S xor OV) = 0	Greater than or equal signed
0 1 1 1	((S xor OV) or Z) = 1	Less than or equal signed
1 1 1 1	((S xor OV) or Z) = 0	Greater than signed

Downloaded from Arrow.com.

B.2 Instruction Set (in Alphabetical Order)

(1/6)

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	×	×	
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,Halfword))		4	4	4					
CLR1	bit#3,disp16[reg1]	10bbb111110RRRRR dddddddddddddddd	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	rrrrr111111RRRR 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
DBRET		0000011111100000 0000000101000110	PC←DBPC PSW←DBPSW		3	3	3	R	R	R	R	R

Downloaded from Arrow.com.

(2/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
DBTRAP		1111110000100000	DBPC←PC+2 (restored PC) DBPSW←PSW PSW.NP←1 PSW.EP←1 PSW.ID←1 PC←00000060H	3	3	3					
DI		0000011111100000 0000000101100000	PSW.ID←1	1	1	1					
DISPOSE	imm5,list12	0000011001iiiiiL 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 is loaded	n+1 Note 4	n+1 Note 4	n+1 Note 4					
	imm5,list12,[reg1]	0000011001iiiiiL LLLLLLLLLLRRRRR **Note 5**	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 is 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	rrrrr111111100000 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					
JR	disp22	0000011110ddddddd ddddddddddddddd0 **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					

Downloaded from Arrow.com.

(3/6)

Mnemonic	Operand	Opcode	Operation		Execution Clock			Flags				
					i	r	l	CY	OV	S	Z	SAT
LD.H	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd0 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Halfword))		1	1	Note 11					
LDSR	reg2,regID	rrrrr111111RRRRR 0000000000100000 **Note 12**	SR[regID]←GR[reg2]	Other than regID = PSW	1	1	1					
				regID = PSW	1	1	1	×	×	×	×	×
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR ddddddddddddddd1 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Halfword)		1	1	Note 11					
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd1 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←Load-memory(adr,Word)		1	1	Note 11					
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]xGR[reg1] **Note 14**		1	4	5					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII00 **Note 13**	GR[reg3] ll GR[reg2]←GR[reg2]xsign-extend(imm9)		1	4	5					
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2]Note 6xGR[reg1]Note 6		1	1	2					
	imm5,reg2	rrrrr010111iiiii	GR[reg2]←GR[reg2]Note 6xsign-extend(imm5)		1	1	2					
MULHI	imm16,reg1,reg2	rrrrr110111RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]Note 6ximm16		1	1	2					
MULU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1] **Note 14**		1	4	5					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII10 **Note 13**	GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9)		1	4	5					
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 dddddddddddddddd	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				×	

Downloaded from Arrow.com.

(4/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
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 is 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 is 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					
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					

Downloaded from Arrow.com.

(5/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SET1	bit#3,disp16[reg1]	00bbb111110RRRRR dddddddddddddddd	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,Halfword))	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,Halfword))	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],Halfword)	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 dddddddddddddddd	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], Halfword)	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					

Downloaded from Arrow.com.

(6/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
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	×	×	×	×	
SWITCH	reg1	00000000010RRRRR	adr←(PC+2) + (GR [reg1] logically shift left by 1) PC←(PC+2) + (sign-extend (Load-memory (adr,Halfword)) 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	0 0 0 0 0 0 1 1 1 1 1 1 i i i i i 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	EIPC ←PC+4 (Restored 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 i i i i i i i i i i i i i i i i	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. ddddddddd: Higher 8 bits of disp9.

2. 3 if there is an instruction that rewrites the contents of the PSW immediately before.

3. If there is no wait state (3 + the number of read access wait states).

4. n is the total number of list12 load registers. (According to the number of wait states. Also, if there are no wait states, n is the total number of list12 registers. If n = 0, same operation as when n = 1)

5. RRRRR: other than 00000.

6. The lower halfword data only are valid.

7. ddddddddddddddddddddd: 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).

Downloaded from Arrow.com.

Notes 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: Higher 4 bits of imm9.

14. Do not specify the same register for general-purpose registers reg1 and reg3.

15. sp/imm: specified by bits 19 and 20 of the sub-opcode.

16. ff = 00: Load sp in ep.

01: Load sign expanded 16-bit immediate data (bits 47 to 32) in ep.

10: Load 16-bit logically left shifted 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.

Downloaded from Arrow.com.

APPENDIX C REGISTER INDEX

(1/5)

Symbol	Name	Unit	Page
ADCR	A/D conversion result register	ADC	365
ADIC	Interrupt control register	INTC	528
ADM	A/D converter mode register	ADC	362
ADS	Analog input channel specification register	ADC	364
ASIF0	Asynchronous serial interface transmit status register 0	UART	386
ASIF1	Asynchronous serial interface transmit status register 1	UART	386
ASIM0	Asynchronous serial interface mode register 0	UART	383
ASIM1	Asynchronous serial interface mode register 1	UART	383
ASIS0	Asynchronous serial interface status register 0	UART	385
ASIS1	Asynchronous serial interface status register 1	UART	385
BRGC0	Baud rate generator control register 0	UART	404
BRGC1	Baud rate generator control register 1	UART	404
BRGIC	Interrupt control register	INTC	528
CKSR0	Clock select register 0	UART	403
CKSR1	Clock select register 1	UART	403
CMP00	8-bit timer H compare register 00	Timer	307
CMP01	8-bit timer H compare register 01	Timer	308
CMP10	8-bit timer H compare register 10	Timer	307
CMP11	8-bit timer H compare register 11	Timer	308
CORAD0	Correction address register 0	ROMC	576
CORAD1	Correction address register 1	ROMC	576
CORAD2	Correction address register 2	ROMC	576
CORAD3	Correction address register 3	ROMC	576
CORCN	Correction control register	ROMC	577
CR010	16-bit timer capture/compare register 010	Timer	224
CR011	16-bit timer capture/compare register 011	Timer	225
CR5	16-bit timer compare register 5	Timer	289, 293, 295
CR50	8-bit timer compare register 50	Timer	289
CR51	8-bit timer compare register 51	Timer	289
CRC01	Capture/compare control register 01	Timer	230
CSI0IC0	Interrupt control register	INTC	528
CSI0IC1	Interrupt control register	INTC	528
CSIC0	Clocked serial interface clock selection register 0	CSI	416
CSIC1	Clocked serial interface clock selection register 1	CSI	416
CSIM00	Clocked serial interface mode register 00	CSI	414
CSIM01	Clocked serial interface mode register 01	CSI	414
CTBP	CALLT base pointer	CPU	48
CTPC	CALLT execution status saving register	CPU	47
CTPSW	CALLT execution status saving register	CPU	47
DBPC	Exception/debug trap status saving register	CPU	48
DBPSW	Exception/debug trap status saving register	CPU	48

Downloaded from Arrow.com.

(2/5)

Symbol	Name	Unit	Page
ECR	Interrupt source register	CPU	45
EIPC	Interrupt status saving register	CPU	44
EIPSW	Interrupt status saving register	CPU	44
FEPC	NMI status saving register	CPU	45
FEPSW	NMI status saving register	CPU	45
IIC0	IIC shift register 0	I^2C	455
IICC0	IIC control register 0	I^2C	442
IICCL0	IIC clock selection register 0	I^2C	452
IICF0	IIC flag register 0	I^2C	450
IICIC0	Interrupt control register	INTC	528
IICS0	IIC status register 0	I^2C	447
IICX0	IIC function expansion register 0	I^2C	453
IMR0	Interrupt mask register 0	INTC	530
IMR1	Interrupt mask register 1	INTC	530
IMR3	Interrupt mask register 3	INTC	530
INTF0	External interrupt falling edge specification register 0	INTC	535
INTF9H	External interrupt falling edge specification register 9H	INTC	536
INTR0	External interrupt rising edge specification register 0	INTC	535
INTR9H	External interrupt rising edge specification register 9H	INTC	536
ISPR	In-service priority register	INTC	531
KRIC	Interrupt control register	INTC	528
KRM	Key return mode register	KR	549
OSTS	Oscillation stabilization time selection register	Standby	555
P0	Port 0 register	Port	77
P0NFC	TIP00 noise elimination control register	Timer	218
P1NFC	TIP01 noise elimination control register	Timer	218
P3	Port 3 register	Port	80
P4	Port 4 register	Port	84
P5	Port 5 register	Port	86
P7	Port 7 register	Port	89
P9	Port 9 register	Port	91
PC	Program counter	CPU	42
PCC	Processor clock control register	CG	126
PCM	Port CM register	Port	96
PDL	Port DL register	Port	97
PF3H	Port 3 function register H	Port	82
PF4	Port 4 function register	Port	85
PF9H	Port 9 function register H	Port	93
PFC3	Port 3 function control register	Port	82
PFC5	Port 5 function control register	Port	88
PFC9	Port 9 function control register	Port	94
PFCE3	Port 3 function control expansion register	Port	82
PFM	Power fail comparison mode register	ADC	367
PFT	Power fail comparison threshold register	ADC	367

Downloaded from Arrow.com.

(3/5)

Symbol	Name	Unit	Page
PIC0	Interrupt control register	INTC	528
PIC1	Interrupt control register	INTC	528
PIC2	Interrupt control register	INTC	528
PIC3	Interrupt control register	INTC	528
PIC4	Interrupt control register	INTC	528
PIC5	Interrupt control register	INTC	528
PIC6	Interrupt control register	INTC	528
PLLCTL	PLL control register	CG	131, 357
PM0	Port 0 mode register	Port	77
PM3	Port 3 mode register	Port	80
PM4	Port 4 mode register	Port	84
PM5	Port 5 mode register	Port	86
PM9	Port 9 mode register	Port	91
PMC0	Port 0 mode control register	Port	78
PMC3	Port 3 mode control register	Port	81
PMC4	Port 4 mode control register	Port	85
PMC5	Port 5 mode control register	Port	87
PMC9	Port 9 mode control register	Port	91
PMCCM	Port CM mode control register	Port	96
PMCM	Port CM mode register	Port	96
PMDL	Port DL mode register	Port	97
PRCMD	Command register	CPU	65
PRM01	Prescaler mode register 01	Timer	233
PRSCM	Interval timer BRG compare register	Timer	332
PRSM	Interval timer BRG mode register	Timer	331
PSC	Power save control register	Standby	553
PSMR	Power save mode register	Standby	554
PSW	Program status word	CPU	46
PU0	Pull-up resistor option register 0	Port	78
PU3	Pull-up resistor option register 3	Port	83
PU4	Pull-up resistor option register 4	Port	85
PU5	Pull-up resistor option register 5	Port	88
PU9	Pull-up resistor option register 9	Port	95
r0 to r31	General-purpose register	CPU	42
RTBH0	Real-time output buffer register H0	RTP	351
RTBL0	Real-time output buffer register L0	RTP	351
RTPC0	Real-time output port control register 0	RTP	353
RTPM0	Real-time output port mode register 0	RTP	352
RXB0	Receive buffer register 0	UART	387
RXB1	Receive buffer register 1	UART	387
SIO00	Serial I/O shift register 0	CSI	421
SIO01	Serial I/O shift register 1	CSI	421
SIRB0	Clocked serial interface receive buffer register 0	CSI	417
SIRB0L	Clocked serial interface receive buffer register 0L	CSI	417

Downloaded from Arrow.com.

(4/5)

Symbol	Name	Unit	Page
SIRB1	Clocked serial interface receive buffer register 1	CSI	417
SIRB1L	Clocked serial interface receive buffer register 1L	CSI	417
SIRBE0	Clocked serial interface read-only receive buffer register 0	CSI	418
SIRBE0L	Clocked serial interface read-only receive buffer register 0L	CSI	418
SIRBE1	Clocked serial interface read-only receive buffer register 1	CSI	418
SIRBE1L	Clocked serial interface read-only receive buffer register 1L	CSI	418
SOTB0	Clocked serial interface transmit buffer register 0	CSI	419
SOTB0L	Clocked serial interface transmit buffer register 0L	CSI	419
SOTB1	Clocked serial interface transmit buffer register 1	CSI	419
SOTB1L	Clocked serial interface transmit buffer register 1L	CSI	419
SOTBF0	Clocked serial interface initial transmit buffer register 0	CSI	420
SOTBF0L	Clocked serial interface initial transmit buffer register 0L	CSI	420
SOTBF1	Clocked serial interface initial transmit buffer register 1	CSI	420
SOTBF1L	Clocked serial interface initial transmit buffer register 1L	CSI	420
SREIC0	Interrupt control register	INTC	528
SREIC1	Interrupt control register	INTC	528
SRIC0	Interrupt control register	INTC	528
SRIC1	Interrupt control register	INTC	528
STIC0	Interrupt control register	INTC	528
STIC1	Interrupt control register	INTC	528
SVA0	Slave address register 0	I^2C	455
SYS	System status register	CPU	66
TCL50	Timer clock selection register 50	Timer	290
TCL51	Timer clock selection register 51	Timer	290
TM01	16-bit timer counter 01	Timer	223
TM0IC10	Interrupt control register	INTC	528
TM0IC11	Interrupt control register	INTC	528
TM5	16-bit timer counter 5	Timer	288
TM50	8-bit timer counter 50	Timer	288
TM51	8-bit timer counter 51	Timer	288
TM5IC0	Interrupt control register	INTC	528
TM5IC1	Interrupt control register	INTC	528
TMC01	16-bit timer mode control register 01	Timer	228
TMC50	8-bit timer mode control register 50	Timer	291
TMC51	8-bit timer mode control register 51	Timer	291
TMCYC0	8-bit timer H carrier control register 0	Timer	312
TMCYC1	8-bit timer H carrier control register 1	Timer	312
TMHIC0	Interrupt control register	INTC	528
TMHIC1	Interrupt control register	INTC	528
TMHMD0	8-bit timer H mode register 0	Timer	310
TMHMD1	8-bit timer H mode register 1	Timer	311
TOC01	16-bit timer output control register 01	Timer	231
TP0CCIC0	Interrupt control register	INTC	528
TP0CCIC1	Interrupt control register	INTC	528

Downloaded from Arrow.com.

(5/5)

Symbol	Name	Unit	Page
TP0CCR0	TMP0 capture/compare register 0	Timer	142
TP0CCR1	TMP0 capture/compare register 1	Timer	144
TP0CNT	TMP0 counter read buffer register	Timer	146
TP0CTL0	TMP0 control register 0	Timer	136
TP0CTL1	TMP0 control register 1	Timer	137
TP0IOC0	TMP0 I/O control register 0	Timer	138
TP0IOC1	TMP0 I/O control register 1	Timer	139
TP0IOC2	TMP0 I/O control register 2	Timer	140
TP0OPT0	TMP0 option register 0	Timer	141
TP0OVIC	Interrupt control register	INTC	528
TXB0	Transmit buffer register 0	UART	388
TXB1	Transmit buffer register 1	UART	388
VSWC	System wait control register	CPU	67
WDCS	Watchdog timer clock selection register	WDT	342
WDT1IC	Interrupt control register	INTC	528
WDTE	Watchdog timer enable register	WDT	348
WDTM1	Watchdog timer mode register 1	WDT	343, 533
WDTM2	Watchdog timer mode register 2	WDT	347
WTIC	Interrupt control register	INTC	528
WTIIC	Interrupt control register	INTC	528
WTM	Watch timer operation mode register	WT	335

Downloaded from Arrow.com.

<R>

APPENDIX D REVISION HISTORY

D.1 Major Revisions in This Edition

(1/3)

Page	Description
Throughout	Modification of oscillation frequency
p. 23	Modification of **1.4 Ordering Information**
p. 36	Modification of **2.2 Pin I/O Circuits and Recommended Connection of Unused Pins**
p. 39	Addition of Type 13-AD to **2.3 Pin I/O Circuits**
p. 40	Modification of **3.1 Features**
p. 43	Modification of **Table 3-2 System Register Numbers**
p. 67	Modification of **3.4.8 (1) (a) System wait control register (VSWC)**
p. 68	Modification of **3.4.8 (1) (b) Access to special on-chip peripheral I/O register**
p. 69	Modification of **3.4.8 (2) (b) Countermeasure**
p. 72	Modification of **Table 4-3 Writing to/Reading from Pn Register**
p. 140	Addition of **Caution 3** to **6.4 (5) TMP0 I/O control register 2 (TP0IOC2)**
p. 149	Modification of **Figure 6-4 Register Setting for Interval Timer Mode Operation**
p. 158	Modification of **Figure 6-10 Basic Timing in External Event Count Mode**
p. 162	Addition of **Cautions** to **6.5.2 (2) Operation timing in external event count mode**
p. 167	Modification of **Figure 6-17 Basic Timing in External Trigger Pulse Output Mode**
p. 167	Addition of description to **6.5.3 External trigger pulse output mode (TP0MD2 to TP0MD0 bits = 010)**
p. 174	Modification of **6.5.3 (2) (b) 0%/100% output of PWM waveform**
p. 182	Modification of **Figure 6-23 Software Processing Flow in One-Shot Pulse Output Mode**
p. 183	Modification of **6.5.4 (2) (a) Note on rewriting TP0CCRa register**
p. 187	Modification of **Figure 6-26 Register Setting in PWM Output Mode**
p. 192	Modification of **6.5.5 (2) (b) 0%/100% output of PWM waveform**
p. 212	Modification of **6.5.7 Pulse width measurement mode (TP0MD2 to TP0MD0 bits = 110)**
pp. 221 to 285	Modification of description in **CHAPTER 7 16-BIT TIMER/EVENT COUNTER 0**
p. 307	Addition of description to **9.2 (1) 8-bit timer H compare register n0 (CMPn0)**
p. 308	Addition of description to **9.2 (2) 8-bit timer H compare register n1 (CMPn1)**
p. 315	Addition of **Note** to **Figure 9-3 Timing of Interval Timer/Square Wave Output Operation**
p. 317	Modification of **Caution 1** and addition of **Caution 3** to **9.4.2 PWM output mode operation**
p. 321	Modification of **Figure 9-5 Operation Timing in PWM Output Mode**
p. 323	Addition of <3> to **Figure 9-6 Transfer Timing**
p. 324	Addition of <8> to **9.4.3 Carrier generator mode operation**
pp. 326 to 328	Modification of **Figure 9-8 Carrier Generator Mode**
p. 333	Modification of **10.1.4 (1) Operation of interval timer BRG**
p. 347	Addition of **Caution 4** to **11.2.3 (1) Watchdog timer mode register 2 (WDTM2)**
p. 348	Addition of **Caution 4** to **11.2.3 (2) Watchdog timer enable register (WDTE)**
p. 362	Modification of **13.3 (1) A/D converter mode register (ADM)**
p. 364	Addition of **Caution 3** to **13.4 (2) Analog input channel specification register (ADS)**

Downloaded from Arrow.com.

(2/3)

Page	Description
p. 367	Addition of **Caution** to **13.4 (4) Power fail comparison mode register (PFM)**
p. 367	Addition of **Caution** to **13.4 (5) Power fail comparison threshold register (PFT)**
p. 369	Modification of **13.4.2 A/D conversion operation**
p. 371	Modification of **Figure 13-5 Example of How to Reduce Power Consumption in Standby Mode**
p. 375	Addition of **13.5 (12) Variation of A/D conversion results, (13) A/D conversion result hysteresis characteristics,** and **(14) Writing to register during A/D conversion**
p. 378	Addition of description to **13.6 (6) Differential linearity error**
p. 438	Modification of **Figure 16-1 Block Diagram of I^2C0**
p. 441	Addition of **16.2 (13) Stop condition generator**
pp. 442, 443	Modification of **16.3 (1) IIC control register 0 (IICC0)**
pp. 447, 448	Modification of **16.3 (2) IIC status register 0 (IICS0)**
p. 451	Addition of description to **16.3 (3) IIC flag register 0 (IICF0)**
p. 452	Addition of description to **16.3 (4) IIC clock selection register 0 (IICCL0)**
p. 453	Addition of description to **16.3 (5) IIC function expansion register 0 (IICX0)**
p. 455	Addition of description to **16.3 (7) IIC shift register 0 (IIC0)**
p. 455	Addition of description to **16.3 (8) Slave address register 0 (SVA0)**
p. 460	Modification of **16.5.4 $\overline{\text{ACK}}$**
p. 464	Addition of **16.5.7 Wait state cancellation method**
pp. 465 to 486	Modification of **16.6 I^2C Interrupt Request Signals (INTIIC0)**
p. 489	Addition of description to **16.10 Extension Code**
p. 492	Modification of **16.13.1 When communication reservation function is enabled (IICF0.IICRSV0 bit = 0)**
p. 492	Modification of **Table 16-6 Wait Periods**
p. 493	Modification of **Figure 16-12 Communication Reservation Timing**
p. 495	Modification of **Table 16-7 Wait Periods**
p. 496	Addition of **16.14 Cautions (3)** to **(6)**
pp. 497 to 501	Modification of **16.15 Communication Operations**
p. 503	Modification of **Figure 16-18 Slave Operation Flowchart (1)**
p. 504	Modification of **Figure 16-19 Slave Operation Flowchart (2)**
pp. 506 to 508	Modification of **Figure 16-20 Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave)**
pp. 509 to 511	Modification of **Figure 16-21 Example of Slave to Master Communication**
p. 547	Modification of **Figure 17-15 Restoring by RETI Instruction**
p. 560	Modification of **Table 19-5 Operation Status in IDLE Mode**
p. 563	Modification of **Table 19-7 Operation Status in STOP Mode**
p. 566	Modification of **Table 19-8 Operation Status in Subclock Operation Mode**
p. 567	Addition of **Caution** to **19.7.1 Setting and operation status**
p. 569	Modification of **Table 19-10 Operation Status in Sub-IDLE Mode**
p. 572	Modification of **Table 20-1 Hardware Status on $\overline{\text{RESET}}$ Pin Input or Occurrence of WDTRES2 Signal**
p. 572	Modification of **Table 20-2 Hardware Status on Occurrence of WDTRES1 Signal**

Downloaded from Arrow.com.

(3/3)

Page	Description
p. 574	Addition of **Figure 20-4 Timing of Reset Operation by Watchdog Timer 1**
p. 574	Addition of **Figure 20-5 Timing of Reset Operation by Watchdog Timer 2**
p. 582	Modification of **Table 22-2 Basic Functions**
p. 582	Modification of **Table 22-3 Security Functions**
p. 583	Addition of **Table 22-4 Security Setting**
p. 584	Addition of **22.3 (1) Security setting by PG-FP4 (Security flag settings)**
pp. 589, 590	Modification of **Figure 22-6 Wiring Example of V850ES/KE1 Flash Writing Adapter (FA-64GK-9ET-A, FA-64GB-8EU-A)**
p. 593	Modification of **Table 22-7 Flash Memory Control Commands**
p. 600	Modification of **22.5.2 (2) Interrupt support**
p. 601	Modification of **Figure 22-17 Standard Self Programming Flow**
p. 603	Modification of **Table 22-11 Internal Resources Used**
pp. 604 to 606	Addition of **CHAPTER 23 ON-CHIP DEBUG FUNCTION**
pp. 607 to 609, 612, 614, 615, 620, 622, 625, 626	Modification of **CHAPTER 24 ELECTRICAL SPECIFICATIONS**
pp. 629, 630	Addition of **CHAPTER 26 RECOMMENDED SOLDERING CONDITIONS**
p. 633	Addition of (2) to **Figure A-1 Development Tool Configuration**
p. 634	Modification of **A.2 Language Processing Software**
p. 635 to 638	Modification of **A.4 Debugging Tools**
p. 656	Addition of **APPENDIX D REVISION HISTORY**

Downloaded from Arrow.com.

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
Podbielskistrasse 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
Steijgerweg 6
5616 HS 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: 02-8175-9600
http://www.tw.necel.com/

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

Downloaded from Arrow.com.