

User's Manual

NEC

Phase-out/Discontinued

V852™

32/16-bit Single-Chip Microcontrollers

Hardware

μPD703002

μPD70P3002

Document No. U10038EJ3V1UM00 (3rd edition)
Date Published December 1997 N CP(K)

© NEC Corporation 1995
Printed in Japan

Phase-out/Discontinued

[MEMO]

NOTES FOR CMOS DEVICES**① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS**

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

V851, V852, and V850 Family are trademarks of NEC Corporation.

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

UNIX is a registered trademark in the United States and other countries, licenced exclusively through X/Open Company, Limited.

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

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

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:

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

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

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

Specific: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

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

Anti-radioactive design is not implemented in this product.

Regional Information

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

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

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

NEC Electronics Inc. (U.S.)

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

NEC Electronics (Germany) GmbH

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

NEC Electronics (UK) Ltd.

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

NEC Electronics Italiana s.r.l.

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

NEC Electronics (Germany) GmbH

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

NEC Electronics (France) S.A.

Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.

Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH

Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.

Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.

Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.

United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.

Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.

Cumbica-Guarulhos-SP, Brasil
Tel: 011-6465-6810
Fax: 011-6465-6829

Major Revisions in This Edition (1/2)

| Pages | Description |
|-------|---|
| p.3 | 1.5 Differences between μPD70P3002 and μPD703002 were added. |
| p.6 | Connection of pins 39 to 41 was changed in 1.6.2 PROM programming mode . |
| p.7 | V_{DD} , V_{SS} , CV_{DD} , and CV_{SS} pins were added to 1.7.1 Internal block diagram . |
| p.14 | Function in normal operation mode of V_{DD} was modified in 2.1.2 PROM programming mode (μPD70P3002 only) . |
| p.17 | Description was added to 2.3.1 (4) (b) (vii) TXD . |
| p.19 | Description was added to 2.3.1 (8) (b) (ii) \overline{UBEN} . |
| p.24 | Description was added to 2.4 Each Pin's I/O Circuit Type and Connection when Unused . |
| p.52 | 4.3.2 Bus width was added. |
| p.57 | Description was added to 4.7.1 Outline of function . |
| p.64 | Illustration was modified in 4.8 (7) Bus hold timing . |
| p.65 | Description was added to 4.10.2 Data space . |
| p.73 | 5.2.4 Noise elimination from NMI pin was added. |
| p.85 | Description of ID bit function was added to 5.3.8 Maskable interrupt status flag . |
| p.93 | Figure 5-13. Pipeline Operation upon Reception of Interrupt Request (Outline) was modified. |
| p.128 | Description was modified in 7.4.3 Overflow . |
| p.138 | Value in expression was modified in Figure 7-14. Pulse Width Measurement Timing (timer 1) . |
| p.140 | Description was modified in 7.6 (3) (a) Using timer 1 . |
| p.140 | Value in expression in Remark was modified in Figure 7-17. PWM Output Timing (TM1) . |
| p.142 | Contents were modified in Figure 7-19. Interrupt Request Processing Routine, Modifying Compare Value (timer 1) . |
| p.143 | Value in expression was modified in Figure 7-20. Cycle Measurement Timing (TM1) . |
| p.164 | Description of CRXEn bit function was added to 8.3.3 (1) Clocked serial interface mode register n (CSIMn) . |
| p.166 | Chart was modified in 8.3.4 (1) Transfer format . |
| p.168 | Chart was modified in Figure 8-6. Timing of 3-Wire Serial I/O Mode (transmission) . |
| p.169 | Chart was modified in Figure 8-7. Timing of 3-Wire Serial I/O Mode (reception) . |
| p.170 | Description was modified in 8.3.7 (1) Starting transmission/reception . |
| p.171 | Chart was modified in Figure 8-8. Timing of 3-Wire Serial I/O Mode (transmission/reception) . |
| p.187 | Description of P20 bit function was added in 9.3.3 Port 2 . |

Major Revisions in This Edition (2/2)

| Pages | Description |
|-------|---|
| p.211 | Port Output latch was modified to Port Input/output latch in Table 10-2. Initial Values of Each Register at Reset. |
| p.214 | Description was added to 11.2 (2) Output disable mode. |
| p.215 | 11.3 Page programming mode flowchart was modified partially. |
| p.216 | Description of V_{PP} and V_{DD} was modified in 11.3 Page programming mode timing. |
| p.217 | Part of 11.3 Byte programming mode flowchart was modified. |
| p.219 | Description was modified in 11.4 PROM Read Procedure (1). |
| p.221 | CSIC1 and CSIC2 were added to APPENDIX A REGISTER INDEX. |
| p.225 | APPENDIX B Legend (2) Symbols used for code was added. |
| p.230 | Code of SATSUBI was modified in APPENDIX B INSTRUCTION SET LIST. |

The mark ★ shows major revised points.

Phase-out/Discontinued

[MEMO]

INTRODUCTION

Readers This manual is intended for users who wish to understand the functions of the V852 (μ PD703002, 70P3002) and design application systems using the V852.

Purpose This manual presents information on the hardware functions of the V852.

Organization Two volumes of the V852 User's Manual are available: hardware (this manual) and architecture (V850 Family™ User's Manual - Architecture) manuals. The organization of each manual is as follows:

| |
|----------|
| Hardware |
|----------|

- Pin function
- CPU function
- Internal peripheral function
- PROM mode

| |
|--------------|
| Architecture |
|--------------|

- Data type
- Register set
- Instruction format and instruction set
- Interrupt and exception
- Pipeline operation

How to Read This Manual It is assumed that the readers of this manual have general knowledge on electric engineering, logic circuits, and microcontrollers.

- To find the details of a register where the name is known
-> Refer to **APPENDIX A REGISTER INDEX**.
- To confirm the details of the function where the name is known
-> Refer to **APPENDIX C INDEX**.
- To understand the details of an instruction function
-> Refer to the **V850 Family User's Manual - Architecture**.
- To understand the overall functions of the V852
-> Read this manual according to the Table of Contents.

Legend

| | |
|---|---|
| Data representation weight | : High digits on the left and low digits on the right |
| Active low representation | : \overline{xxx} (top bar over pin or signal name) |
| Memory map address | : Top: highest, bottom: lowest |
| Note | : Description of "Note" in the text |
| Caution | : Information requiring particular attention |
| Remark | : Additionally explanatory material |
| Numeric representations | : Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH |
| Prefix indicating power of 2 (address space, memory capacity) | : K (kilo): $2^{10} = 1024$ M (mega): $2^{20} = 1024^2$ G (giga): $2^{30} = 1024^3$ |

Related documents

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

○ Documents related to device

| Document Name | Document Number | |
|--|-----------------|-------------|
| | Japanese | English |
| V850 Family User's Manual-Architecture | U10243J | U10243E |
| V850 Family Instruction List | U10229J | U10229E |
| μPD703002 Data Sheet | U11826J | U11826E |
| μPD70P3002 Data Sheet | U11827J | U11827E |
| V852 User's Manual-Hardware | U10038J | This manual |
| V852 Register Instruction List | U10513J | — |

○ Documents related to development tools

| Document Name | | Document Number | |
|-------------------------------------|-------------------------------|-----------------|---------|
| | | Japanese | English |
| IE-703002-MC (In-circuit Emulator) | | U11595J | — |
| CA850 (C Compiler Package) | Operation UNIX™ based | U11013J | U11013E |
| | Operation Windows™ based | U11068J | U11068E |
| | C Language | U11010J | U11010E |
| | Assembly Language | U10543J | U10543E |
| | Project Manager Windows based | U11991J | U11991E |
| RX850 (Real-time OS) | Basic | U11037J | U11037E |
| | Technical | U11117J | U11117E |
| | Nucleus Installation | U11038J | U11038E |
| | Debugger Windows based | U11158J | U11158E |
| AZ850 (System Performance Analyzer) | | U11181J | U11181E |
| ID850 (C source debugger) | Instruction UNIX based | U12209J | — |
| | Installation UNIX based | U12210J | U12210E |
| | Instruction Windows based | U11196J | U11196E |

CONTENTS

| | |
|---|---------------|
| CHAPTER 1 INTRODUCTION | 1 |
| 1.1 General | 1 |
| 1.2 Features | 2 |
| 1.3 Application Fields | 3 |
| 1.4 Ordering Information | 3 |
| 1.5 Differences between μPD70P3002 and μPD703002 | 3 |
| 1.6 Pin Configuration (Top View)..... | 4 |
| 1.6.1 Normal operation mode | 4 |
| 1.6.2 PROM programming mode | 6 |
| 1.7 Function Block Configuration | 7 |
| 1.7.1 Internal block diagram | 7 |
| 1.7.2 Internal units | 8 |
| 1.8 Differences between V851 and V852..... | 10 |
| CHAPTER 2 PIN FUNCTIONS..... | 11 |
| 2.1 Pin Function List..... | 11 |
| 2.1.1 Normal operation mode | 11 |
| 2.1.2 PROM programming mode (μ PD70P3002 only) | 14 |
| 2.2 Pin Status | 15 |
| 2.3 Pin Function | 16 |
| 2.3.1 Normal operation mode | 16 |
| 2.3.2 PROM programming mode (μ PD70P3002 only) | 23 |
| 2.4 I/O Circuit Type and Connection of Unused Pins..... | 24 |
| 2.5 Pin I/O Circuits | 25 |
| CHAPTER 3 CPU FUNCTIONS | 27 |
| 3.1 Features | 27 |
| 3.2 CPU Register Set | 28 |
| 3.2.1 Program register set | 29 |
| 3.2.2 System register set | 30 |
| 3.3 Operation Modes | 32 |
| 3.3.1 Operation modes | 32 |
| 3.3.2 Specifying operation mode | 33 |
| 3.4 Address Space | 34 |
| 3.4.1 CPU address space | 34 |
| 3.4.2 Image (Virtual Address Space) | 35 |
| 3.4.3 Wrap-around of CPU address space | 36 |
| 3.4.4 Memory map | 37 |
| 3.4.5 Area | 38 |
| 3.4.6 External expansion mode | 45 |
| 3.4.7 Recommended use of address space | 47 |
| 3.4.8 Peripheral I/O registers | 49 |
| CHAPTER 4 BUS CONTROL FUNCTION | 51 |
| 4.1 Features | 51 |

| | | |
|--|---|-----------|
| 4.2 | Bus Control Pins | 51 |
| 4.3 | Bus Access | 52 |
| | 4.3.1 Number of access clocks | 52 |
| ★ | 4.3.2 Bus width | 52 |
| 4.4 | Memory Block Function | 53 |
| 4.5 | Wait Function | 54 |
| | 4.5.1 Programmable wait function | 54 |
| | 4.5.2 External wait function | 55 |
| | 4.5.3 Relationships between programmable wait and external wait | 55 |
| 4.6 | Idle State Insertion Function | 56 |
| 4.7 | Bus Hold Function | 57 |
| | 4.7.1 Outline of function | 57 |
| | 4.7.2 Bus hold procedure | 57 |
| | 4.7.3 Operation in power save mode | 57 |
| 4.8 | Bus Timing | 58 |
| 4.9 | Bus Priority | 65 |
| 4.10 | Memory Boundary Operation Condition | 65 |
| | 4.10.1 Program space | 65 |
| | 4.10.2 Data space | 65 |
| 4.11 | Internal Peripheral I/O Interface | 66 |
| CHAPTER 5 INTERRUPT/EXCEPTION PROCESSING FUNCTION | | 67 |
| 5.1 | Features | 67 |
| 5.2 | Non-Maskable Interrupt | 69 |
| | 5.2.1 Accepting operation | 70 |
| | 5.2.2 Restore operation | 72 |
| | 5.2.3 NP flag | 73 |
| ★ | 5.2.4 Noise elimination for NMI pin | 73 |
| | 5.2.5 External interrupt mode register 0 (INTM0) | 73 |
| 5.3 | Maskable Interrupts | 74 |
| | 5.3.1 Block diagram | 75 |
| | 5.3.2 Operation | 75 |
| | 5.3.3 Restore | 77 |
| | 5.3.4 Priorities of maskable interrupts | 78 |
| | 5.3.5 Interrupt control register (xxICn) | 82 |
| | 5.3.6 External interrupt mode registers 1 and 2 (INTM1 and INTM2) | 84 |
| | 5.3.7 In-service priority register (ISPR) | 85 |
| | 5.3.8 Maskable interrupt status flag | 85 |
| 5.4 | Software Exception | 86 |
| | 5.4.1 Operation | 86 |
| | 5.4.2 Restore | 87 |
| | 5.4.3 EP flag | 88 |
| 5.5 | Exception Trap | 88 |
| | 5.5.1 Illegal op code definition | 88 |
| | 5.5.2 Operation | 89 |
| | 5.5.3 Restore | 90 |
| 5.6 | Priority Control | 91 |
| | 5.6.1 Priorities of interrupts and exceptions | 91 |
| | 5.6.2 Multiple interrupt processing | 91 |

| | | |
|---|--|------------|
| 5.7 | Interrupt Latency Time | 93 |
| 5.8 | Periods Where Interrupt Is Not Acknowledged | 93 |
| CHAPTER 6 CLOCK GENERATION FUNCTION..... | | 95 |
| 6.1 | Features | 95 |
| 6.2 | Configuration | 95 |
| 6.3 | Selecting Input Clock | 96 |
| 6.3.1 | Direct mode | 96 |
| 6.3.2 | PLL mode | 96 |
| 6.4 | PLL Stabilization | 98 |
| 6.5 | Power Save Control | 99 |
| 6.5.1 | General..... | 99 |
| 6.5.2 | Control registers | 101 |
| 6.5.3 | HALT mode | 104 |
| 6.5.4 | IDLE mode | 106 |
| 6.5.5 | Software STOP mode..... | 108 |
| 6.6 | Specifying Oscillation Stabilization Time | 110 |
| 6.7 | Clock Output Control | 113 |
| CHAPTER 7 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)..... | | 115 |
| 7.1 | Features | 115 |
| 7.2 | Basic Configuration | 116 |
| 7.2.1 | Timer 1 | 118 |
| 7.2.2 | Timer 4 | 120 |
| 7.3 | Control Registers | 121 |
| 7.4 | Timer 1 Operation | 127 |
| 7.4.1 | Count operation | 127 |
| 7.4.2 | Selecting count clock frequency | 127 |
| 7.4.3 | Overflow | 128 |
| 7.4.4 | Clearing/starting timer by TCLR1 input | 129 |
| 7.4.5 | Capture operation | 130 |
| 7.4.6 | Compare operation | 132 |
| 7.5 | Timer 4 Operation | 134 |
| 7.5.1 | Count operation | 134 |
| 7.5.2 | Selecting the count clock frequency | 134 |
| 7.5.3 | Overflow | 134 |
| 7.5.4 | Compare operation | 135 |
| 7.6 | Application Examples | 137 |
| 7.7 | Note | 145 |
| CHAPTER 8 SERIAL INTERFACE FUNCTION..... | | 147 |
| 8.1 | Features | 147 |
| 8.2 | Asynchronous Serial Interface (UART) | 148 |
| 8.2.1 | Features | 148 |
| 8.2.2 | Configuration of asynchronous serial interface | 149 |
| 8.2.3 | Mode registers and control registers | 151 |
| 8.2.4 | Interrupt request | 157 |
| 8.2.5 | Operation..... | 158 |

| | | |
|--|---|------------|
| 8.3 | Clocked Serial Interface 0 to 2 (CSI0 to CSI2) | 162 |
| 8.3.1 | Features | 162 |
| 8.3.2 | Configuration | 163 |
| 8.3.3 | Mode registers and control registers | 164 |
| 8.3.4 | Basic operation | 166 |
| 8.3.5 | Transmission in 3-wire serial I/O mode | 168 |
| 8.3.6 | Reception in 3-wire serial I/O mode | 169 |
| 8.3.7 | Transmission/reception in 3-wire serial I/O mode | 170 |
| 8.3.8 | System Configuration Example | 172 |
| 8.4 | Baud Rate Generator 0, 1 (BRG0, BRG1) | 173 |
| 8.4.1 | Configuration and function | 173 |
| 8.4.2 | Baud rate generator register 0, 1 (BRG0, BRG1) | 176 |
| 8.4.3 | Baud rate generator prescaler mode register 0, 1 (BPRM0, BPRM1) | 176 |
| CHAPTER 9 PORT FUNCTION | | 177 |
| 9.1 | Features | 177 |
| 9.2 | Basic Configuration of Port | 178 |
| 9.3 | Port Pin Function | 182 |
| 9.3.1 | Port 0 | 182 |
| 9.3.2 | Port 1 | 186 |
| 9.3.3 | Port 2 | 187 |
| 9.3.4 | Port 3 | 192 |
| 9.3.5 | Port 4 | 197 |
| 9.3.6 | Port 5 | 199 |
| 9.3.7 | Port 6 | 201 |
| 9.3.8 | Port 9 | 202 |
| 9.3.9 | Port 10 | 205 |
| 9.4 | Noise Elimination Circuit | 208 |
| CHAPTER 10 RESET FUNCTION | | 209 |
| 10.1 | Features | 209 |
| 10.2 | Pin Function | 209 |
| 10.3 | Initialize | 210 |
| CHAPTER 11 PROM MODE | | 213 |
| 11.1 | PROM Mode | 213 |
| 11.2 | Operation Mode | 213 |
| 11.3 | PROM Write Procedure | 215 |
| 11.4 | PROM Read Procedure | 219 |
| 11.5 | Screening of OTPROM Version | 220 |
| 11.6 | Caution on STOP Mode Release when Using External Clock | 220 |
| APPENDIX A REGISTER INDEX | | 221 |
| APPENDIX B INSTRUCTION SET LIST | | 225 |
| APPENDIX C INDEX | | 233 |

LIST OF FIGURES (1/2)

| Figure No. | Title | Page |
|------------|--|------|
| 3-1. | Program Counter (PC) | 29 |
| 3-2. | Interrupt Source Register (ECR) | 30 |
| 3-3. | Program Status Word (PSW) | 31 |
| 3-4. | CPU Address Space | 34 |
| 3-5. | Image on Address Space | 35 |
| 3-6. | Interrupt/Exception Table | 39 |
| 3-7. | External Memory Area (when expanded to 64 KB, 256 KB, or 1 MB) | 42 |
| 3-8. | External Memory Area (when expanded to 4 MB) | 43 |
| 3-9. | External Memory Area (when fully expanded) | 44 |
| 3-10. | Recommended Memory Map | 48 |
| 4-1. | Example of Inserting Wait States | 55 |
| 5-1. | Non-Maskable Interrupt Processing | 70 |
| 5-2. | Accepting Non-Maskable Interrupt Request | 71 |
| 5-3. | RETI Instruction Processing | 72 |
| 5-4. | Maskable Interrupt Block Diagram | 75 |
| 5-5. | Maskable Interrupt Processing | 76 |
| 5-6. | RETI Instruction Processing | 77 |
| 5-7. | Example of Interrupt Nesting Process | 79 |
| 5-8. | Example of Processing Interrupt Requests Simultaneously Generated | 81 |
| 5-9. | Software Exception Processing | 86 |
| 5-10. | RETI Instruction Processing | 87 |
| 5-11. | Exception Trap Processing | 89 |
| 5-12. | RETI Instruction Processing | 90 |
| 5-13. | Pipeline Operation upon Reception of Interrupt Request (Outline) | 93 |
| 6-1. | Block Configuration | 112 |
| 7-1. | Basic Operation of Timer 1 | 127 |
| 7-2. | Operation after Occurrence of Overflow (when ECLR1 = 0, OST = 1) | 128 |
| 7-3. | Clearing/Starting Timer by TCLR1 Input (when ECLR1 = 1, OST = 0) | 129 |
| 7-4. | Relationships between Clear/Start by TCLR1 Input and Overflow (when ECLR1 = 1, OST = 1) | 129 |
| 7-5. | Example of TM1 Capture Operation (when both edges are specified) | 130 |
| 7-6. | Example of TM1 Capture Operation | 131 |
| 7-7. | Example of Compare Operation | 132 |
| 7-8. | Example of TM1 Compare Operation (set/reset output mode) | 133 |
| 7-9. | Basic Operation of Timer 4 | 134 |
| 7-10. | Operation with CM4 at 1 to FFFFH | 135 |
| 7-11. | When CM4 Is Set to 0 | 136 |
| 7-12. | Example of Timing of Interval Timer Operation (timer 4) | 137 |
| 7-13. | Setting Procedure of Interval Timer Operation (timer 4) | 137 |

LIST OF FIGURES (2/2)

| Figure No. | Title | Page |
|------------|---|------|
| 7-14. | Pulse Width Measurement Timing (timer 1) | 138 |
| 7-15. | Setting Procedure for Pulse Width Measurement (timer 1) | 139 |
| 7-16. | Interrupt Request Processing Routine Calculating Pulse Width (timer 1) | 139 |
| 7-17. | PWM Output Timing (TM1) | 140 |
| 7-18. | Programming Procedure of PWM Output (timer 1) | 141 |
| 7-19. | Interrupt Request Processing Routine, Modifying Compare Value (timer 1) | 142 |
| 7-20. | Cycle Measurement Timing (TM1) | 143 |
| 7-21. | Set-up Procedure for Cycle Measurement (timer 1) | 144 |
| 7-22. | Interrupt Request Processing Routine Calculating Cycle (timer 1) | 144 |
| 8-1. | Block Diagram of Asynchronous Serial Interface | 150 |
| 8-2. | Format of Transmit/Receive Data of Asynchronous Serial Interface | 158 |
| 8-3. | Asynchronous Serial Interface Transmission Completion Interrupt Timing | 159 |
| 8-4. | Asynchronous Serial Interface Reception Completion Interrupt Timing | 161 |
| 8-5. | Receive Error Timing | 161 |
| 8-6. | Timing of 3-Wire Serial I/O Mode (transmission) | 168 |
| 8-7. | Timing of 3-Wire Serial I/O Mode (reception) | 169 |
| 8-8. | Timing of 3-Wire Serial I/O Mode (transmission/reception) | 171 |
| 8-9. | Example of CSI System Configuration | 171 |
| 8-10. | Block Diagram of Baud Rate Generator | 172 |
| 9-1. | Block Diagram of P00, P01 (Port 0) | 183 |
| 9-2. | Block Diagram of P02 to P07 (Port 0) | 183 |
| 9-3. | Block Diagram of P10 to P17 (Port 1) | 186 |
| 9-4. | Block Diagram of P20 (Port 2) | 188 |
| 9-5. | Block Diagram of P21 to P24 (Port 2) | 188 |
| 9-6. | Block Diagram of P25 (Port 2) | 189 |
| 9-7. | Block Diagram of P26 (Port 2) | 189 |
| 9-8. | Block Diagram of P27 (Port 2) | 190 |
| 9-9. | Block Diagram of P30, P33, P35 (Port 3) | 193 |
| 9-10. | Block Diagram of P31, P36 (Port 3) | 194 |
| 9-11. | Block Diagram of P32, P37 (Port 3) | 194 |
| 9-12. | Block Diagram of P34 (Port 3) | 195 |
| 9-13. | Block Diagram of P40 to P47 (Port 4) | 197 |
| 9-14. | Block Diagram of P50 to P57 (Port 5) | 199 |
| 9-15. | Block Diagram of P60 to 67 (Port 6) | 202 |
| 9-16. | Block Diagram of P90 to P97 (Port 9) | 203 |
| 9-17. | Block Diagram of P100, P103 (Port 10) | 205 |
| 9-18. | Block Diagram of P101 (Port 10) | 206 |
| 9-19. | Block Diagram of P102 (Port 10) | 206 |
| 9-20. | Example of Noise Elimination Timing | 208 |
| 11-1. | PROM Read Timing | 219 |

LIST OF TABLES

| Table No. | Title | Page |
|-----------|---|------|
| ★ 1-1. | Differences between μ PD70P3002 and μ PD70P3002 | 3 |
| 1-2. | Differences between V851 and V852 | 10 |
| 3-1. | Program Registers | 29 |
| 3-2. | System Register Numbers | 30 |
| 4-1. | Bus Priority | 65 |
| 5-1. | Interrupt List | 68 |
| 5-2. | Addresses and Bits of Interrupt Control Register | 83 |
| 6-1. | Operation of Clock Generator by Power Save Control | 100 |
| 6-2. | Operating Status in HALT Mode | 104 |
| 6-3. | Operating Status in IDLE Mode | 106 |
| 6-4. | Operating Status in Software STOP Mode | 108 |
| 6-5. | Example of Count Time | 112 |
| 7-1. | Configuration of RPU | 116 |
| 7-2. | Capture Trigger Signal to 16-Bit Capture Register (TM1) | 130 |
| 7-3. | Interrupt Request Signal from 16-Bit Compare Register (TM1) | 132 |
| 8-1. | Default Priority of Interrupts | 157 |
| 8-2. | BRG Set-up Values | 174 |
| 10-1. | Operating Status of Each Pin During Reset Period | 209 |
| 10-2. | Initial Values of Each Register at Reset | 211 |

Phase-out/Discontinued

[MEMO]

CHAPTER 1 INTRODUCTION

The V852 is a product of NEC's V850 family of single-chip microcontrollers for real-time control applications. This chapter briefly outlines the V852.

1.1 General

The V852 is a 32-/16-bit single-chip microcontroller that employs the CPU core of the V850 family of high-performance 32-bit single-chip microcontrollers for real-time control applications, and integrates peripheral functions such as ROM/RAM, real-time pulse unit, and serial interface.

The V852 is provided with multiplication instructions that are executed with a hardware multiplier, saturated operation instructions, and bit manipulation instructions that are ideal for digital servo control applications, in addition to basic instructions that have a high real-time response speed and can be executed in 1 clock cycle. This microcontroller can be applied as a real-time control system to numerous fields such as the AV field (camcorders, VCRs, etc.), OA field (PPCs, LBPs, printers, etc.), industrial field (motor control, NC machine tools, etc.), and communication field (cellular phones, etc.). In any of these applications, the V852 demonstrates an extremely high cost effectiveness.

1.2 Features

- Number of instructions : 74
- Minimum instruction execution time : 40 ns (at 25 MHz)
- General register : 32 bits x 32
- Instruction set : Signed multiply (16 bits x 16 bits -> 32 bits): 1 to 2 clocks
Saturated operation instructions (with overflow/underflow detection function)
32-bit shift instructions: 1 clock
Bit manipulation instructions
Load/store instructions with long/short format
- Memory space : 16 MB linear (common to program/data)
Memory is divided in 1 MB unit blocks and wait states can be inserted into a bus cycle for every two blocks.
Programmable wait function
Idle state insertion function
- External bus interface : 16-bit data bus (address/data multiplexed)
Bus hold function
External wait function
- Internal memory : ROM/PROM : 90 KB
RAM : 3 KB
- Interrupt/exception : Non-maskable: 1 source
Maskable: 16 sources (Eight levels of priorities can be set.)
Illegal instruction code exception
- I/O line : I/O port : 67
- Real-time pulse unit : 16-bit timer/event counter: 1 ch
16-bit capture/compare register: 4
16-bit interval timer: 1 ch
16-bit compare register: 1
- Serial interface : Asynchronous serial interface (UART): 1 ch
Clocked serial interface (CSI): 3 ch
Dedicated baud rate generator
- Clock generator : Multiplication function by PLL clock synthesizer
- Power save function : HALT/IDLE/STOP mode
Clock output stop function
- CMOS technology

1.3 Application Fields

AV field : Camcorders, VCRs, etc.
 OA field : PPCs, LBPs, printers, etc.
 Industrial field : Motor control, NC machine tools, etc.
 Communication field : Cellular phones, etc.

1.4 Ordering Information

| Part number | Package | Internal ROM |
|-----------------------------|---|---------------|
| μ PD703002GC-25-xxx-7EA | 100-pin plastic QFP (fine pitch) (14 x 14 mm) | Mask ROM |
| μ PD70P3002GC-25-7EA | 100-pin plastic QFP (fine pitch) (14 x 14 mm) | One-time PROM |

Remark xxx indicates a ROM code suffix.

★ 1.5 Differences between μ PD70P3002 and μ PD703002

The μ PD70P3002 is a PROM version of the μ PD703002. Therefore, these two versions are identical except for differences because of the ROM specifications (for example, specifications concerning writing and verifying). Table 1-1 and 1-2 show the differences between the two.

Table 1-1. Differences between μ PD70P3002 and μ PD703002

| Item | Part Number | μ PD70P3002 | μ PD703002 |
|--|-------------|--|--|
| Internal program memory (electrical writing) | | One-time PROM (can be written only once) | Mask ROM |
| PROM programming pin | | Provided | None |
| Setting of MODE0 and MODE1 pins | | <ul style="list-style-type: none"> In normal operation mode MODE0, 1 = LH In PROM programming mode MODE0, 1 = HH | <ul style="list-style-type: none"> In normal operation mode MODE0, 1 = LH In ROM-less mode MODE0, 1 = LL |
| Electrical characteristics | | Supply current, recommended oscillation circuit, etc. are different. | |
| Internal ROM empty area | | When programming the internal ROM, write the same instruction code for the empty area of the PROM and mask ROM versions. | |
| Others | | Noise immunity and noise radiation differ because circuit scale and mask layout differ. | |

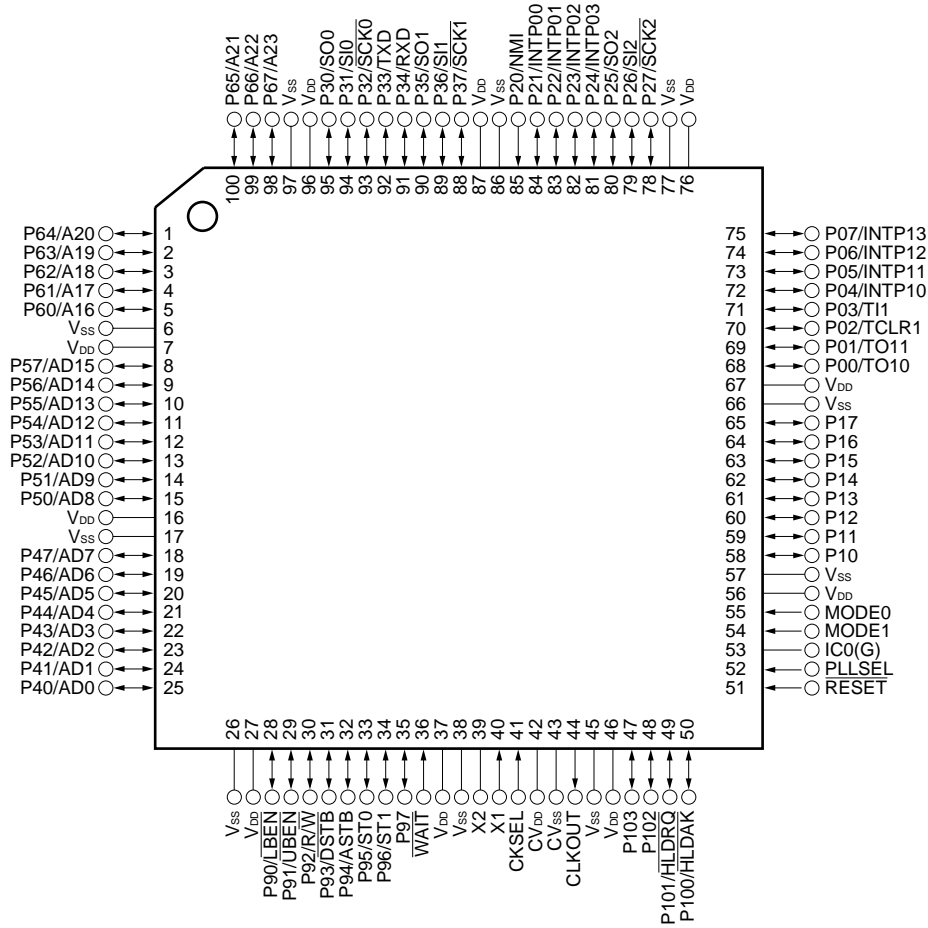
- Cautions**
- The PROM and mask ROM versions differ from each other in terms of noise immunity and noise emission. When replacing the PROM version with the mask ROM version in the course of switching from experimental production to mass production, perform thorough evaluation with the CS model (not ES model) of the mask ROM version.**
 - Directly connect the MODE0 and MODE1 pins to V_{DD} or V_{SS}.**

Remark L : low level
 H : high level

1.6 Pin Configuration (Top View)

1.6.1 Normal operation mode

- μ PD703002GC-25-xxx-7EA
- μ PD70P3002GC-25-7EA



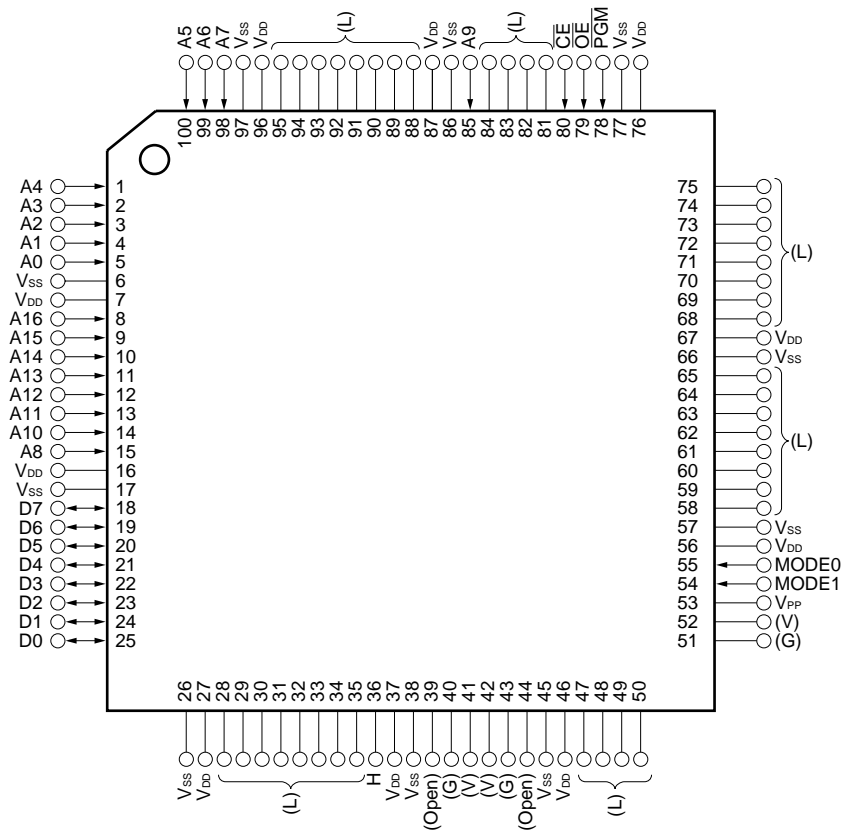
Caution The content in parentheses indicates the processing of the pin not used in the normal operation mode.

G: Connect this pin to V_{SS}.

| | | | |
|--|--------------------------------------|----------------------------|--------------------------------|
| P00 to P07 | : Port0 | A16 to A23 | : Address Bus |
| P10 to P17 | : Port1 | $\overline{\text{LBEN}}$ | : Lower Byte Enable |
| P20 to P27 | : Port2 | $\overline{\text{UBEN}}$ | : Upper Byte Enable |
| P30 to P37 | : Port3 | $\overline{\text{R/W}}$ | : Read/Write Status |
| P40 to P47 | : Port4 | $\overline{\text{DSTB}}$ | : Data Strobe |
| P50 to P57 | : Port5 | ASTB | : Address Strobe |
| P60 to P67 | : Port6 | ST0, ST1 | : Status |
| P90 to P97 | : Port9 | $\overline{\text{HLDK}}$ | : Hold Acknowledge |
| P100 to P103 | : Port10 | $\overline{\text{HLDRQ}}$ | : Hold Request |
| TO10, TO11 | : Timer Output | CLKOUT | : Clock Output |
| TCLR1 | : Timer Clear | CKSEL | : Clock Select |
| TI1 | : Timer Input | $\overline{\text{PLLSEL}}$ | : PLL Select |
| INTP00 to INTP03, | | $\overline{\text{WAIT}}$ | : Wait |
| INTP10 to INTP13 | : Interrupt Request From Peripherals | MODE0, MODE1 | : Mode |
| NMI | : Non-maskable Interrupt Request | $\overline{\text{RESET}}$ | : Reset |
| SO0 to SO2 | : Serial Output | X1, X2 | : Crystal |
| SI0 to SI2 | : Serial Input | CV_{DD} | : Clock Generator Power Supply |
| $\overline{\text{SCK0}}$ to $\overline{\text{SCK2}}$ | : Serial Clock | CV_{SS} | : Clock Generator Ground |
| TXD | : Transmit Data | V_{DD} | : Power Supply |
| RXD | : Receive Data | V_{SS} | : Ground |
| AD0 to AD15 | : Address/Data Bus | IC0 | : Internally Connected |

★ 1.6.2 PROM programming mode

• μ PD70P3002GC-25-7EA



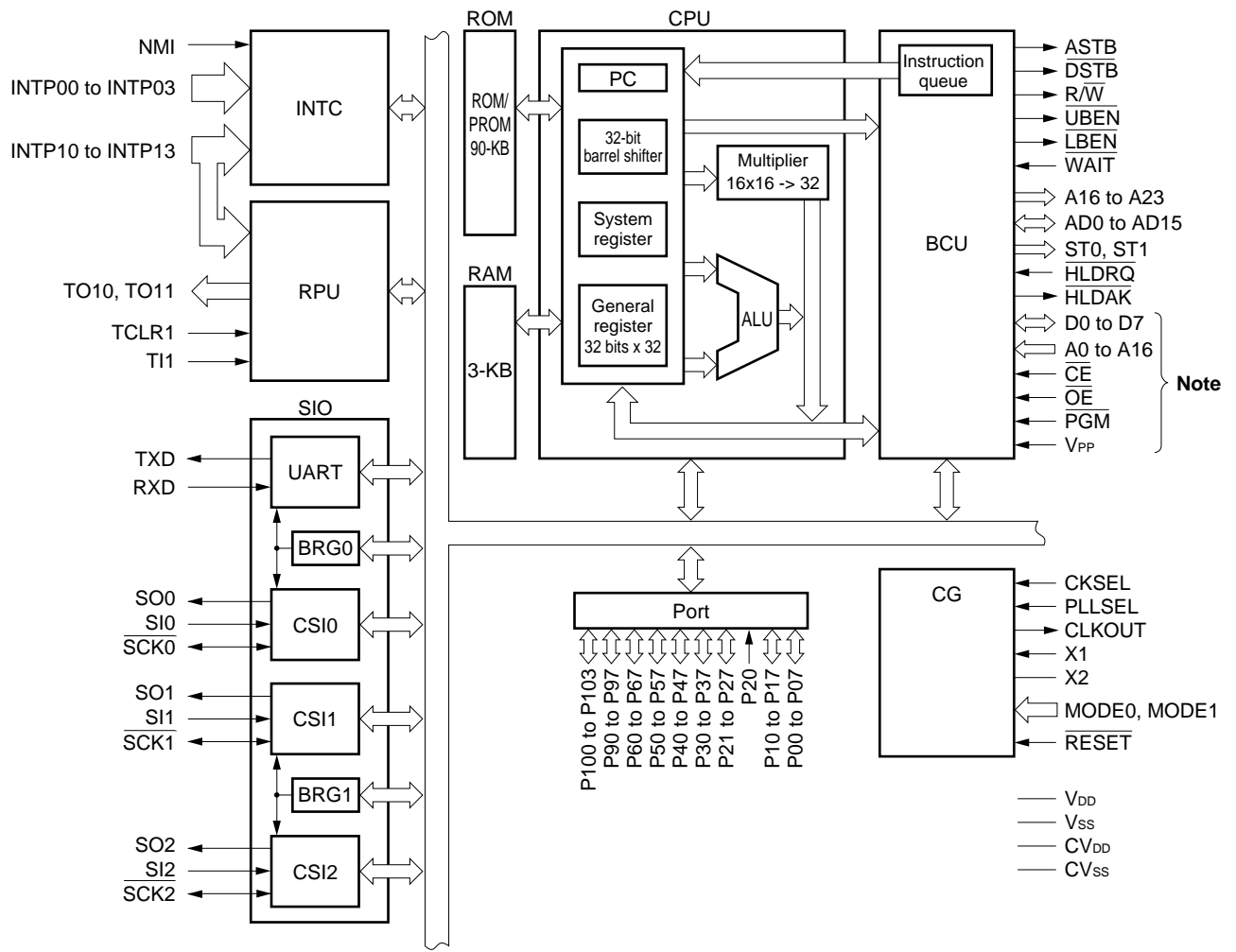
Caution The content in parentheses indicates the connection of the pin not used in the PROM programming mode.

- L** : Individually connect this pin to V_{SS} via a resistor.
- H** : Connect this pin to V_{DD} via a resistor.
- G** : Connect this pin to V_{SS}.
- V** : Connect this pin to V_{DD}.
- Open** : Connect nothing to this pin.

| | | | |
|-------------------------|--------------------|-----------------|----------------------------|
| A0 to A16 | : Address Bus | MODE0, MODE1 | : Programming Mode Set |
| D0 to D7 | : Data Bus | V _{DD} | : Power Supply |
| $\overline{\text{CE}}$ | : Chip Enable | V _{SS} | : Ground |
| $\overline{\text{OE}}$ | : Output Enable | V _{PP} | : Programming Power Supply |
| $\overline{\text{PGM}}$ | : Programming Mode | | |

1.7 Function Block Configuration

★ 1.7.1 Internal block diagram



Note Pins used in the PROM programming mode

1.7.2 Internal units

(1) CPU

Executes almost all the instruction processing such as address calculation, arithmetic/logic operation, and data transfer in 1 clock by using a 5-stage pipeline.

Dedicated hardware devices such as a multiplier (16 bits x 16 bits -> 32 bits) and a barrel shifter (32 bits) are provided to increase the speed of processing complicated instructions.

(2) Bus control unit (BCU)

Initiates the necessary number of external bus cycles based on the physical address obtained by the CPU. If the CPU does not issue a request to start a bus cycle when fetching an instruction from the external memory area, generates a prefetch address to prefetch an instruction code. The prefetched instruction code is loaded to the internal instruction queue.

(3) ROM/PROM

ROM or PROM of 90 Kbytes mapped starting from address 00000000H. Access is enabled/disabled by the MODE0 and MODE1 pins. With the PROM version, the programming mode is specified by these two pins. This ROM/PROM is accessed in 1 clock by the CPU when an instruction is fetched.

(4) RAM

3-KB RAM mapped starting from address FFFFE000H. This RAM can be accessed in 1 clock by the CPU when data is accessed.

(5) Interrupt controller (INTC)

Processes interrupt requests (NMI, INTPO0 to INTPO3, INTP10 to INTP13) from the internal peripheral hardware and external sources. Eight levels of priorities can be specified for these interrupt requests, and multiplexed processing control can be performed on an interrupt source.

(6) Clock generator (CG)

Supplies the CPU clock whose frequency is one or five times (when the internal PLL is used) or 1/2 times (when the PLL is not used) the frequency of the oscillator connected across the X1 and X2 pins. Input from an external clock source can also be referenced instead of using the oscillator.

(7) Real-time pulse unit (RPU)

Provides a 16-bit timer/event counter, a 16-bit interval timer, and capabilities for measuring pulse width and frequency, and generation of programmable pulse outputs.

(8) Serial interface (SIO)

The serial interface of the V852 consists of 1-channel asynchronous serial interface (UART) and 3-channel synchronous or clocked serial interface (CSI).

UART transfers data by using the TXD and RXD pins, and the CSI transfers data by using the SO0 to SO2, SI0 to SI2, and $\overline{\text{SCK0}}$ to $\overline{\text{SCK2}}$ pins.

The output of the baud rate generator and system clock can be selected as the serial interface clock source.

(9) Ports

The V852 is provided with a total of 68 I/O port pins (one of them is input only) that constitute ports 0 to 10. These port pins also function as various control pins.

| Port | I/O | Function | |
|--------|-----------|--------------|---|
| Port0 | 8-bit I/O | General port | Timer I/O, external interrupt |
| Port1 | | | – |
| Port2 | | | External interrupt, serial interface |
| Port3 | | | Serial interface |
| Port4 | | | External address/data bus |
| Port5 | | | |
| Port6 | | | External address bus |
| Port9 | | | External bus interface control signal I/O |
| Port10 | | | |

1.8 Differences between V851™ and V852

The V852 is provided with increased internal ROM/RAM capacity and more CSI channels of the serial interface. The number of PLL multiplication can be selected from either one or five.

Table 1-2. Differences between V851 and V852

| Item | | V851 | V852 |
|--|----------------|---|---|
| Internal ROM capacity | | 32 KB | 90 KB |
| Internal RAM capacity | | 1 KB | 3 KB |
| Serial interface | | UART : 1 ch CSI : 1 ch Baud rate generator : 1 | UART : 1 ch CSI : 3 ch Baud rate generator : 2 |
| Interrupt source | | External : 9 sources (Including NMI) Internal : 10 sources | External : 9 sources (Including NMI) Internal : 12 sources |
| Interrupt/ exception table | 00000160H | — | INTCSI1 |
| | 00000170H | — | INTCSI2 |
| Number of multiplication when using PLL | | Multiplication by 5 | Multiplication by 1 or 5 |
| I/O port (total 68) | | Dedicated pins : 17 Shared with control pins : 51 | Dedicated pins : 11 Shared with control pins : 57 |
| Pin name (at normal operation mode) | | | |
| | QFP pin number | | |
| | 52 | IC1 | PLLSEL |
| | 78 | P27 | P27/ $\overline{\text{SCK2}}$ |
| | 79 | P26 | P26/SI2 |
| | 80 | P25 | P25/SO2 |
| | 88 | P37 | P37/ $\overline{\text{SCK1}}$ |
| | 89 | P36 | P36/SI1 |
| | 90 | P35 | P35/SO1 |
| | 93 | P32/ $\overline{\text{SCK}}$ | P32/ $\overline{\text{SCK0}}$ |
| | 94 | P31/SI | P31/SI0 |
| | 95 | P30/SO | P30/SO0 |
| Peripheral I/O register | | | |
| | I/O address | | |
| | FFFFF094H | — | BRG1 |
| | FFFFF096H | — | BPRM1 |
| | FFFFF098H | — | CSIM1 |
| | FFFFF09AH | — | SIO1 |
| | FFFFF0A8H | — | CSIM2 |
| | FFFFF0AAH | — | SIO2 |
| | FFFFF11CH | — | CSIC1 |
| | FFFFF11EH | — | CSIC2 |

CHAPTER 2 PIN FUNCTIONS

The following table shows the names and functions of the V852's pins. These pins can be divided by function into port pins and other pins.

2.1 Pin Function List

2.1.1 Normal operation mode

(1) Port pins

(1/2)

| Pin Name | I/O | Function | Alternate Function |
|------------|-------|--|--------------------|
| P00 | I/O | Port 0. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | TO10 |
| P01 | | | TO11 |
| P02 | | | TCLR1 |
| P03 | | | T11 |
| P04 | | | INTP10 |
| P05 | | | INTP11 |
| P06 | | | INTP12 |
| P07 | | | INTP13 |
| P10 to P17 | I/O | Port 1. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | – |
| P20 | Input | Port 2. P20 is an input-only port. Operates as an NMI input when a valid edge is input. Indicates NMI input status with bit 0 of P2 register. P21 to P27 are 7-bit I/O ports. Input/output can be specified in 1-bit units. | NMI |
| P21 | I/O | | INTP00 |
| P22 | | | INTP01 |
| P23 | | | INTP02 |
| P24 | | | INTP03 |
| P25 | | | SO2 |
| P26 | | | SI2 |
| P27 | | | SCK2 |

(2/2)

| Pin Name | I/O | Function | Alternate Function |
|------------|-----|--|---------------------------|
| P30 | I/O | Port 3. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | SO0 |
| P31 | | | SI0 |
| P32 | | | $\overline{\text{SCK0}}$ |
| P33 | | | TXD |
| P34 | | | RXD |
| P35 | | | SO1 |
| P36 | | | SI1 |
| P37 | | | $\overline{\text{SCK1}}$ |
| P40 to 47 | I/O | Port 4. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | AD0 to AD7 |
| P50 to P57 | I/O | Port 5. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | AD8 to AD15 |
| P60 to P67 | I/O | Port 6. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | A16 to A23 |
| P90 | I/O | Port 9. 8-bit I/O port. Can be specified in input/output mode in 1-bit units. | $\overline{\text{LBEN}}$ |
| P91 | | | $\overline{\text{UBEN}}$ |
| P92 | | | $\overline{\text{R/W}}$ |
| P93 | | | $\overline{\text{DSTB}}$ |
| P94 | | | ASTB |
| P95 | | | ST0 |
| P96 | | | ST1 |
| P97 | | | - |
| P100 | I/O | Port 10. 4-bit I/O port. Can be specified in input/output mode in 1-bit units. | $\overline{\text{HLDK}}$ |
| P101 | | | $\overline{\text{HLDRQ}}$ |
| P102 | | | - |
| P103 | | | - |

(2) Non-port pins

(1/2)

| Pin Name | I/O | Function | Alternate Function |
|---------------------------|--------|--|------------------------------|
| TO10 | Output | Pulse signal output from timer 1. | P00 |
| TO11 | | | P01 |
| TCLR1 | Input | External clear signal input to timer 1. | P02 |
| TI1 | | External count clock input to timer 1. | P03 |
| INTP10 | Input | External capture trigger input to timer 1. Also used to input external maskable interrupt request. | P04 |
| INTP11 | | | P05 |
| INTP12 | | | P06 |
| INTP13 | | | P07 |
| NMI | Input | Non-maskable interrupt request input. | P20 |
| INTP00 | Input | External maskable interrupt request input. | P21 |
| INTP01 | | | P22 |
| INTP02 | | | P23 |
| INTP03 | | | P24 |
| SO0 | Output | Serial transmit data output from CS10. | P30 |
| SI0 | Input | Serial receive data input to CS10. | P31 |
| $\overline{\text{SCK0}}$ | I/O | Serial clock I/O from/to CS10. | P32 |
| SO1 | Output | Serial transmit data output from CS11. | P35 |
| SI1 | Input | Serial receive data input to CS11. | P36 |
| $\overline{\text{SCK1}}$ | I/O | Serial clock I/O from/to CS11. | P37 |
| SO2 | Output | Serial transmit data output from CS12. | P25 |
| SI2 | Input | Serial receive data input to CS12. | P26 |
| $\overline{\text{SCK2}}$ | I/O | Serial clock I/O from/to CS12. | P27 |
| TXD | Output | Serial transmit data output from UART. | P33 |
| RXD | Input | Serial receive data input to UART. | P34 |
| AD0 to AD7 | I/O | 16-bit multiplexed address/data bus when external memory is used. | P40 to P47 |
| AD8 to AD15 | | | P50 to P57 |
| A16 to A23 | O | Higher address bus when external memory is used. | P60 to P67 |
| $\overline{\text{LBEN}}$ | Output | Lower byte enable signal output of external data bus. | P90 |
| $\overline{\text{UBEN}}$ | | Higher byte enable signal output of external data bus. | P91 |
| $\overline{\text{R/W}}$ | | External read/write status output. | P92 |
| $\overline{\text{DSTB}}$ | | External data strobe signal output. | P93 |
| $\overline{\text{ASTB}}$ | | External address strobe signal output. | P94 |
| ST0 | | External bus cycle status output. | P95 |
| ST1 | | | P96 |
| $\overline{\text{HLDAK}}$ | | Output | Bus hold acknowledge output. |
| $\overline{\text{HLDRQ}}$ | Input | Bus hold request input. | P101 |
| CLKOUT | Output | System clock output. | — |

(2/2)

| Pin Name | I/O | Function | Alternate Function |
|---------------------------|-------|---|--------------------|
| CKSEL | Input | Input specifying operation mode of clock generator. | – |
| PLLSEL | Input | Input specifying the number of PLL multiplication. | – |
| $\overline{\text{WAIT}}$ | Input | Control signal input inserting wait state to bus cycle. | – |
| MODE0, MODE1 | Input | Specifies operation mode of the V852. | – |
| $\overline{\text{RESET}}$ | Input | System reset input. | – |
| X1 | Input | System clock oscillator connecting pins. Supply external clock to X1. | – |
| X2 | – | | – |
| CV _{DD} | – | Positive power supply for internal clock generator. | – |
| CV _{SS} | – | Ground for internal clock generator. | – |
| V _{DD} | – | Positive power supply | – |
| V _{SS} | – | Ground | – |
| IC0 | – | Internally connected | – |

2.1.2 PROM programming mode (μ PD70P3002 only)

Control and timing of the V852 in the PROM mode are compatible with those of the μ PD27C1001A. The functions of the pins of the V852 in the PROM mode are as follows:

| Pin Name | Function in PROM Mode | Function in Normal Operation Mode |
|-------------------------|--|-----------------------------------|
| A0 to A7 | Address input, low (A0 to A7) | P60 to P67 |
| A8, A9, A10 to A16 | Address input, high (A8 to A16) | P50, P20, P51 to P57 |
| D0 to D7 | Data input/output | P40 to P47 |
| $\overline{\text{CE}}$ | $\overline{\text{CE}}$ (chip enable) input | P25 |
| $\overline{\text{OE}}$ | $\overline{\text{OE}}$ (output enable) input | P26 |
| $\overline{\text{PGM}}$ | $\overline{\text{PGM}}$ (program) input | P27 |
| V _{PP} | Power supply for program write | IC0 |
| MODE0, MODE1 | Operation mode specification | MODE0, MODE1 |

★

2.2 Pin Status

The operating status of each pin in each operation mode is as follows:

| Operating Status Pin | Reset | STOP Mode | IDLE Mode | Bus Hold | Idle State | HALT Mode |
|---|----------|--------------|--------------|----------------------------|----------------------------|----------------------------|
| AD0 to AD15 | Hi-Z | Hi-Z | Hi-Z | Hi-Z | Hi-Z | Hi-Z |
| A16 to A23 | Hi-Z | Hi-Z | Hi-Z | Hi-Z | Retained ^{Note 1} | Retained |
| $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$ | Hi-Z | Hi-Z | Hi-Z | Hi-Z | Retained ^{Note 1} | Retained |
| $\text{R}/\overline{\text{W}}$ | Hi-Z | Hi-Z | Hi-Z | Hi-Z | H | H |
| $\overline{\text{DSTB}}$ | Hi-Z | Hi-Z | Hi-Z | Hi-Z | H | H |
| ASTB | Hi-Z | Hi-Z | Hi-Z | Hi-Z | H | H |
| ST0, ST1 | Hi-Z | Hi-Z | Hi-Z | Hi-Z | Idle status | Idle status |
| $\overline{\text{HLDRQ}}$ | – | – | – | Operates | Operates | Operates |
| $\overline{\text{HLDAK}}$ | Hi-Z | Hi-Z | Hi-Z | L | Operates | Operates |
| $\overline{\text{WAIT}}$ | – | – | – | – | – | – |
| CLKOUT | Operates | L | L | Operates ^{Note 2} | Operates ^{Note 2} | Operates ^{Note 2} |

Hi-Z : high-impedance

Retained : Retains status in external bus cycle immediately before

L : low-level output

H : high-level output

– : input not sampled

- Notes**
1. Undefined immediately after bus hold ends.
 2. "L" during clock output inhibit mode.

2.3 Pin Function

2.3.1 Normal operation mode

(1) P00 to P07 (Port0) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 0. They also serve as control signal pins.

P00 to P07 function not only as I/O port pins, but also as the I/O pins of the real-time pulse unit (RPU) and external interrupt request input pins. Each bit of port 0 can be specified in the port or control mode, by using port mode control register 0 (PMC0).

(a) Port mode

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

(b) Control mode

P00 to P07 can be set in the port or control mode in 1-bit units by the PMC0 register.

(i) TO10, TO11 (Timer Output) ... output

These pins output pulse signals from timer 1.

(ii) TCLR1 (Timer Clear) ... input

This pin inputs an external clear signal to timer 1.

(iii) TI1 (Timer Input) ... input

This pin inputs an external count clock to timer 1.

(iv) INTP10 to INTP13 (Interrupt Request From Peripherals) ... input

These pins are the external interrupt request input pins of timer 1.

(2) P10 to P17 (Port 1) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 1, which can be set in the input or output mode in 1-bit units by using port mode register 1 (PM1).

The pins of port 1 function only as I/O pins and are not multiplexed with control pins.

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

These pins constitute an I/O port, port 2, which can be set in the input or output mode in 1-bit units except P20 which is fixed to the input mode. They function not only as port pins but also as external interrupt input and serial interface (CSI) pins.

Each bit of this port can be specified in the port or control mode by using port mode control register 2 (PMC2).

(a) Port mode

P21 to P27 can be set in the input or output mode in 1-bit units by port mode register 2 (PM2).

PM20 is an input-only port, and operates as an NMI input when a valid edge is input.

(b) Control mode

P21 to P27 can be set in the port or control mode in 1-bit units by port mode control register 2 (PMC2).

P20 is dedicated to the control mode.

(i) NMI (Non-Maskable Interrupt Request) ... input

This pin inputs a non-maskable interrupt request.

(ii) INTP00 to INTP03 (Interrupt Request From Peripherals) ... input

These pins input external maskable interrupt requests.

(iii) SO2 (Serial Output) ... output

This pin outputs the serial transmit data of CSI2.

(iv) SI2 (Serial Input) ... input

This pin inputs the serial receive data of CSI2.

(v) $\overline{\text{SCK2}}$ (Serial Clock) ... 3-state I/O

This pin inputs/outputs the serial clock of CSI2.

(4) P30 to P37 (Port 3) ... 3-state input

These pins constitute an 8-bit I/O port, port 3. They also function as control signal pins. P30 to P37 function not only as I/O port pins but also as serial interface (UART, CSI) I/O pins in the control mode.

(a) Port mode

P30 to P37 can be set in the input or output mode in 1-bit units by port mode register 3 (PM3).

(b) Control mode

P30 to P37 can be set in the port or control mode in 1-bit units by the PMC3 register.

(i) SO0 (Serial Output) ... output

This pin outputs the serial transmit data of CSIO.

(ii) SI0 (Serial Input) ... input

This pin inputs the serial receive data of CSIO.

(iii) $\overline{\text{SCK0}}$ (Serial Clock) ... 3-state I/O

This pin inputs/outputs the serial clock of CSIO.

(iv) SO1 (Serial Output) ... output

This pin outputs the serial transmit data of CSI1.

(v) SI1 (Serial Input) ... input

This pin inputs the serial receive data of CSI1.

(vi) $\overline{\text{SCK1}}$ (Serial Clock) ... 3-state I/O

This pin inputs/outputs the serial clock of CSI1.

(vii) TXD (Transmit Data) ... output

This pin outputs the serial transmit data of UART.

Transmit disabled : high-impedance

Transmit enabled : high-level

★

(viii) RXD (Receive Data) ... input

This pin inputs the serial receive data of UART.

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

These pins constitute an 8-bit I/O port, port 4. They also form a portion of the address/data bus connected to external memory.

P40 to P47 function not only as I/O port pins but also as multiplexed address/data bus pins (AD0 to AD7) in the control mode (external expansion mode) when an external memory is connected.

Each bit of this port can be set in the port or control mode, in 1-bit units, by using mode specification pins (MODE0 and MODE1), and memory expansion mode register (MM).

(a) Port mode

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

(b) Control mode (external expansion mode)

P40 to P47 can be specified as AD0 to AD7 by using the MODE0 and MODE1 pins and MM register.

(i) AD0 to AD7 (Address/Data0 to 7) ... 3-state I/O

These pins constitute a multiplexed address/data bus when the external memory is accessed. They function as the A0 to A7 output pins of a 24-bit address in the address timing (T1 state), and as the lower 8-bit data I/O bus pins of 16-bit data in the data timing (T2, TW, T3).

The output status of these pins changes in synchronization with the rising edge of the clock in each state of the bus cycle. AD0 to AD7 go into a high-impedance state in the idle state (TI).

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

These pins constitute an 8-bit I/O port, port 5. They also form a portion of the address/data bus connected to external memory.

P50 to P57 function not only as I/O port pins but also as multiplexed address/data bus pins (AD8 to AD15) in the control mode (external expansion mode) when an external memory is connected.

Each bit of this port can be set in the port or control mode in 1-bit units by using mode specification pins (MODE0 and MODE1), and memory expansion mode register (MM).

(a) Port mode

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

(b) Control mode (external expansion mode)

P50 to P57 can be specified as AD8 to AD15 by using the MODE0 and MODE1 pins and MM register.

(i) AD8 to AD15 (Address/Data8 to 15) ... 3-state I/O

These pins constitute a multiplexed address/data bus when the external memory is accessed. They function as the A8 to A15 output pins of a 24-bit address in the address timing (T1 state), and as the higher 8-bit data I/O bus pins of 16-bit data in the data timing (T2, TW, T3).

The output status of these pins changes in synchronization with the rising edge of the clock in each state of the bus cycle. AD8 to AD15 go into a high-impedance state in the idle state (TI).

(7) P60 to P67 (Port 6) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 6. They also form a portion of the address/data bus connected to external memory.

P60 to P67 function not only as I/O port pins but also as address bus pins (A16 to A23) in the control mode (external expansion mode) when an external memory is connected. This port can be set in the port or control mode in 2-bit units by using mode specification pins (MODE0 and MODE1), and memory expansion mode register (MM).

(a) Port mode

P60 to P67 can be set in the input or output port mode in 1-bit units by using port mode register 6 (PM6).

(b) Control mode (external expansion mode)

P60 to P67 can be specified as A16 to A23 by using the MODE0 and MODE1 pins and MM register.

(i) A16 to A23 (Address/Data16 to 23) ... output

These pins constitute the higher 8 bits of a 24-bit address bus when the external memory is accessed. The output status of these pins changes in synchronization with the rising edge of the clock in the T1 state. During the idle state (TI), the address of the bus cycle immediately before entering the idle state is retained.

(8) P90 to P97 (Port 9) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 9, and are also used to output control signals.

P90 to P96 function not only as I/O port pins but also as control signal output pins in the control mode (external expansion mode) when an external memory is used.

This port can be set in the port or control mode in 5-, 2-, or 1-bit units by using mode specification pins (MODE0 and MODE1), and memory expansion mode register (MM).

P97 functions only as I/O pin and is not multiplexed with control pin.

(a) Port mode

P90 to P97 can be set in the input or output port mode in 1-bit units by using port mode register 9 (PM9).

(b) Control mode (external expansion mode)

P90 to P96 can be used to output control signals when so specified by the MODE0 and MODE1 pins and MM register when an external memory is used.

(i) $\overline{\text{LBEN}}$ (Lower Byte Enable) ... output

This is the lower byte enable signal of the 16-bit external data bus.

This signal changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. During the idle state (TI), the address of the bus cycle immediately before entering the idle state is retained.

★

(ii) $\overline{\text{UBEN}}$ (Upper Byte Enable) ... output

This is the upper byte enable signal of the 16-bit external data bus. It becomes active (low) in the odd-number byte access mode, and becomes inactive (high) in the even-number byte access mode.

This signal changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. During the idle state (TI), the address of the bus cycle immediately before entering the idle state is retained.

| Access | | \overline{UBEN} | \overline{LBEN} | A0 |
|------------------|--------------|-------------------|-------------------|----|
| Word Access | | 0 | 0 | 0 |
| Half-word Access | | 0 | 0 | 0 |
| Byte Access | Even address | 1 | 0 | 0 |
| | Odd address | 0 | 1 | 1 |

(iii) R/\overline{W} (Read/Write Status) ... output

This is a status signal that indicates whether the bus cycle for external access is a read or write cycle. It goes high in the read cycle and low in the write cycle.

This signal changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. It goes high in the idle state (TI).

(iv) \overline{DSTB} (Data Strobe) ... output

This is the access strobe signal of the external data bus.

It becomes active (low) in the T2 or TW state of the bus cycle, and becomes inactive (high) in the idle state (TI).

(v) $ASTB$ (Address Strobe) ... output

This is the latch strobe signal of the external address bus.

It becomes active (low) in synchronization with the falling edge of the clock in the T1 state of the bus cycle, and becomes inactive (high) in synchronization with the falling edge of the clock in the T3 state. It goes high in the idle state (TI).

(vi) $ST0, ST1$ (Status0, 1) ... output

These are status signals which indicate the access type of the current bus cycle when external memory is referenced. The status changes in synchronization with the rising edge of the clock in the T1 and TI states of the bus cycle.

| ST1 | ST0 | Bus Cycle Status |
|-----|-----|--------------------------------|
| 0 | 0 | Idle cycle |
| 0 | 1 | Instruction fetch (branch) |
| 1 | 0 | Operand data access |
| 1 | 1 | Instruction fetch (continuous) |

In the following cases, the "Instruction fetch (branch)" is output in the first bus cycle branched (T1 to T3 states).

- Instruction fetch of the branched destination by the branch instructions (JMP, JR, JARL, Bcond)
- Instruction fetch of the source by the RETI instruction
- Instruction fetch of the jumped destination (interrupt/exception table) by reset, TRAP instruction, and interrupt

Instruction fetches other than the above output the status of the instruction fetch (continuous).

(9) P100 to P103 (Port 10) ... 3-state I/O

Port 10 is a 4-bit I/O port that can be set in the input or output mode in 1-bit units. In addition to the function as a port, the pins constituting port 10 are used to input/output control signals, in the control mode, when an external bus master or ASIC is connected.

If port 10 is accessed in 8-bit units, the higher 4 bits are ignored if the access is write, and undefined if the access is read.

P102 and P103 function only as I/O pins and are not multiplexed with control pins.

(a) Port mode

P100 to P103 can be set in the input or output mode, in 1-bit units, by port mode register (PM10).

(b) Control mode

P100 and P101 function as input and output pins for bus hold control signals when the function is enabled by mode control register 10 (PMC10).

(i) $\overline{\text{HLDAK}}$ (Hold Acknowledge) ... output

This is an acknowledge signal that indicates that the V852 has set the address bus, data bus, and control bus in the high-impedance state in response to a bus hold request.

As long as this signal is active, the address/data bus, and control signals remain in a high-impedance state.

(ii) $\overline{\text{HLDRQ}}$ (Hold Request) ... input

This input signal is used by an external device to request that the V852 relinquish control of the address, data bus, and control signals. This signal can be activated asynchronously with CLKOUT. When this signal becomes active, the V852 sets the address/data bus and control signals in the high-impedance state, after the current bus cycle completes. If there is no current bus activity, the address/data bus and control signals are immediately set to high-impedance. $\overline{\text{HLDAK}}$ is then made active and the bus and control lines are released.

(10) CLKOUT (Clock Output) ... output

This pin outputs the system clock, even during reset. The output of this pin can be fixed to low level when the clock output inhibit mode is set by the PSC register.

(11) CKSEL (Clock Select) ... input

This pin specifies the operation mode of the clock generation circuit. Once set, the input value of this pin cannot be changed during operation.

| CKSEL | Operation Mode |
|-------|----------------|
| 0 | PLL mode |
| 1 | Direct mode |

(12) PLLSEL (PLL Select) ... input

This input pin selects whether the system clock has a frequency one ($1 \times f_{xx}$) or five ($5 \times f_{xx}$) times the frequency (f_{xx}) of the external oscillator or external clock in the PLL mode (CKSEL = 0).

Once set, the input value of this pin cannot be changed during operations.

This pin has no function in the direct mode (CKSEL = 1). Leave it as an unused pin.

| | |
|--------|-------------------|
| PLLSEL | ϕ |
| 0 | f_{xx} |
| 1 | $5 \times f_{xx}$ |

(13) $\overline{\text{WAIT}}$ (Wait) ... input

This control signal input pin inserts a data wait state to the bus cycle, and can be activated asynchronously to CLKOUT. This pin is sampled at the falling edge of the clock in the T2 and TW states of the bus cycle. If the set/hold time for the sampling timing is not satisfied, the wait state may not be inserted.

(14) MODE0, MODE1 (Mode0, 1) ... input

These pins specify the operation mode of the V852. Three operation modes can be selectable: single-chip mode, ROM-less mode, and PROM programming mode. The input value of these pins cannot be changed during normal operation.

| MODE1 | MODE0 | Operation Mode | |
|-------|-------|------------------|--|
| 0 | 0 | ROM-less mode | |
| 0 | 1 | RFU (reserved) | |
| 1 | 0 | Single-chip mode | |
| 1 | 1 | PROM mode | $V_{PP} = 5 \text{ V}$: read mode |
| | | | $V_{PP} = 12.5 \text{ V}$: programming mode |

(15) $\overline{\text{RESET}}$ (Reset) ... input

The $\overline{\text{RESET}}$ signal is an asynchronous input signal. A valid low-level signal on the $\overline{\text{RESET}}$ pin initiates a system reset, regardless of the clock operation. In addition to normal system initialization/start functions, the $\overline{\text{RESET}}$ signal is also used for exiting processor power-save modes (HALT, IDLE, or STOP).

(16) X1, X2 (Crystal) ... input

An oscillator for system clock generation is connected across these pins.

An external clock source can also be referenced by connecting the external clock input to the X1 pin and leaving the X2 pin open.

(17) CV_{DD} (Power Supply for Clock Generator)

This pin supplies positive power to the internal clock generator.

(18) CV_{SS} (Ground for Clock Generator)

This is the ground pin of the internal clock generator.

(19) V_{DD} (Power Supply)

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

(20) V_{SS} (Ground)

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

(21) IC0 (Internally Connected)

This pin is internally connected and must be connected to V_{SS} .

2.3.2 PROM programming mode (μ PD70P3002 only)**(1) A0 to A16 ... input**

These pins constitute an address bus that selects an address of the internal PROM (00000H to 167FFH).

(2) D0 to D7 ... I/O

These pins constitute a data bus through which the internal PROM is written/read.

(3) $\overline{\text{PGM}}$... input

This pin inputs a program pulse and is activated when $V_{PP} = 12.5 \text{ V}$, $\overline{\text{CE}} = 0$, and $\overline{\text{OE}} = 1$. Upon activation, the program on D0 to D7 is written to an internal PROM cell selected by A0 to A16.

(4) $\overline{\text{CE}}$... input

This is a chip enable input pin. When this signal is active, the program in PROM can be written/read.

(5) $\overline{\text{OE}}$... input

This is an output enable signal input pin and inputs a read strobe signal to the internal PROM. When the signal is activated while $\overline{\text{CE}} = 0$, the program (1 byte) of the internal PROM cell selected by A0 to A16, will appear at the outputs, D0 to D7.

(6) V_{PP} ... input

This pin inputs a program pulse. When this pin is activated while $V_{PP} = 12.5 \text{ V}$, $\overline{\text{CE}} = 0$, and $\overline{\text{OE}} = 1$, the program byte on D0 to D7 can be written to the internal PROM cell selected by A0 to A16.

(7) V_{DD}

Positive power supply pin

(8) V_{SS}

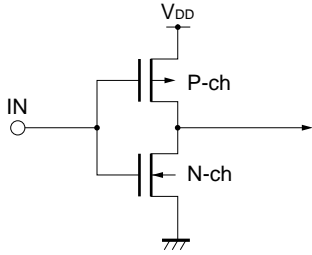
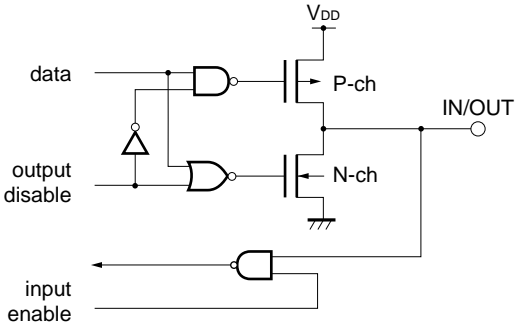
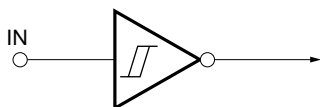
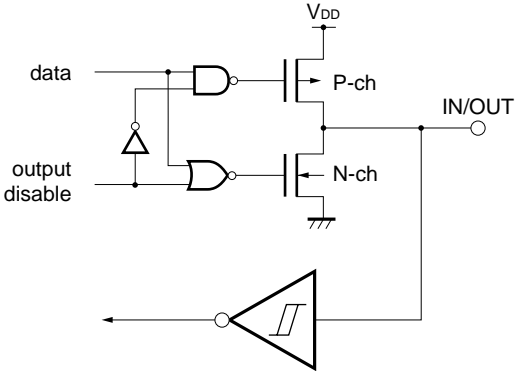
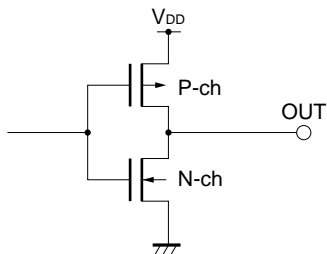
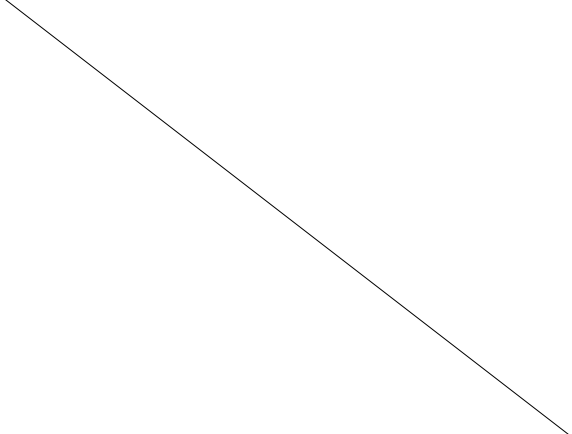
GND pin

2.4 I/O Circuit Type and Connection of Unused Pins

- ★ When connecting to V_{DD} or V_{SS} via a resistor, it is recommended to connect a resistor with a resistance of 1 to 10 k Ω .

| Pin | I/O Circuit Type | Recommended Connection |
|--|------------------|---|
| P00/TO10, P01/TO11 | 5 | Input status : Individually connect to V_{DD} or V_{SS} via resistor Output status : Open |
| P02/TCLR1, P03/TI1, P04/INTP10 to P07/INTP13 | 8 | |
| P10 to P17 | 5 | |
| ★ P20/NMI | 2 | Directly connect to V_{SS} |
| P21/INTP00 to P24/INTP03 | 8 | Input status : Individually connect to V_{DD} or V_{SS} via resistor Output status : Open |
| P25/SO2 | 5 | |
| P26/SI2, P27/SCK2 | 8 | |
| P30/SO0 | 5 | |
| P31/SI0, P32/SCK0 | 8 | |
| P33/TXD, P34/RXD, P35/SO1 | 5 | |
| P36/SI1, P37/SCK1 | 8 | |
| P40/AD0 to P47/AD7 | 5 | |
| P50/AD8 to P57/AD15 | | |
| P60/A16 to P67/A23 | | |
| P90/LBEN | | |
| P91/UBEN | | |
| P92/R/W | | |
| P93/DSTB | | |
| P94/ASTB | | |
| P95/ST0, P96/ST1 | | |
| P97 | | |
| P100/HLDAK | | |
| P101/HLDRQ | | |
| P102 | | |
| P103 | | |
| CLKOUT | 3 | Open |
| CKSEL | 2 | – |
| PLLSEL | 2 | – |
| ★ WAIT | 1 | Directly connect to V_{DD} |
| MODE0, MODE1 | 2 | – |
| RESET | | |
| IC0 | – | Directly connect to V_{SS} |
| ★ CV_{DD} | – | Directly connect to V_{DD} |
| ★ CV_{SS} | – | Directly connect to V_{SS} |

2.5 Pin I/O Circuits

| | |
|--|---|
| <p>Type 1</p>  | <p>Type 5</p>  |
| <p>Type 2</p>  <p>Schmitt trigger input with hysteresis characteristics</p> | <p>Type 8</p>  |
| <p>Type 3</p>  |  |

[MEMO]

CHAPTER 3 CPU FUNCTIONS

The CPU of the V852 is based on a RISC architecture and executes most instructions in one clock cycle by using a 5-stage pipeline.

3.1 Features

- Minimum instruction execution time: 40 ns (at 25 MHz)
- Address space: 16 MB linear
- Thirty-two 32-bit general registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- Single-cycle 32-bit shift instruction
- Long/short instruction format
- Internal memory
 - ROM/PROM : 90 KB
 - RAM : 3 KB
- Four types of bit manipulation instructions
 - Set
 - Clear
 - Not
 - Test

3.2 CPU Register Set

The registers of the V852 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers are 32 bits wide.

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

Program register set

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

| | | |
|----|-----------------|---|
| 31 | | 0 |
| PC | Program Counter | |

System register set

| | | |
|-------|-------------------------|---|
| 31 | | 0 |
| EIPC | Exception/Interrupt PC | |
| EIPSW | Exception/Interrupt PSW | |

| | | |
|-------|-----------------|---|
| 31 | | 0 |
| FEPC | Fatal Error PC | |
| FEPSW | Fatal Error PSW | |

| | | |
|-----|--------------------------|---|
| 31 | | 0 |
| ECR | Exception Cause Register | |

| | | |
|-----|---------------------|---|
| 31 | | 0 |
| PSW | Program Status Word | |

3.2.1 Program register set

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

(1) General registers

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

However, r0 and r30 are implicitly used by instructions, and care must be exercised when using these registers. Also, r1 to r5 and r31 are implicitly used by the assembler and C compiler. Therefore, before using these registers, their contents must be saved so that they are not lost. The contents must be restored to the registers after the registers have been used.

Also, for the details of r1 to r5 and r31, refer to **CA850 User's Manual**.

Table 3-1. Program Registers

| Name | Usage | Operation |
|-----------|-----------------------------|--|
| r0 | Zero register | Always holds 0 |
| r1 | Assembler-reserved register | Used as a working register for creating 32-bit immediate |
| r2 | Interrupt stack pointer | Stack pointer for interrupt handler |
| 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 | Used as a register specifying the start of the text area ^{Note} |
| r6 to r29 | – | Address/data variable registers |
| r30 | Element pointer | Base pointer register when memory is accessed |
| r31 | Link pointer | Used by compiler when calling function |
| PC | Program counter | Holds instruction address during program execution |

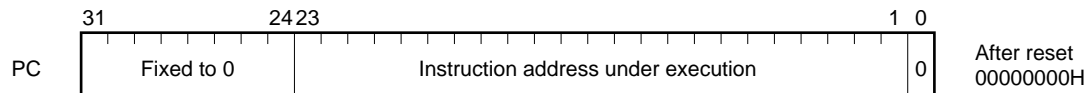
Note Area to allocate the program code.

(2) Program counter

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

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

Figure 3-1. Program Counter (PC)



3.2.2 System register set

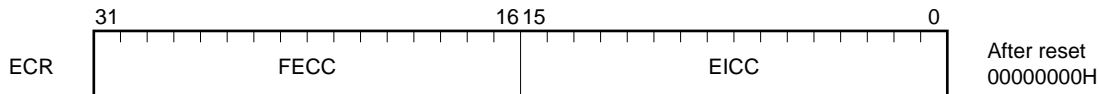
System registers control the status of the CPU and hold interrupt information.

Table 3-2. System Register Numbers

| No. | System Register Name | Usage | Operation |
|---------|----------------------|--|--|
| 0 | EIPC | Status saving registers during interrupt | These registers save the PC and PSW when an exception or interrupt occurs. Because only one set of these registers is available, their contents must be saved when multiple interrupts are enabled. The high 8 bits of the EIPC and the high 24 bits of the EIPSW are fixed to 0. |
| 1 | EIPSW | | |
| 2 | FEPC | Status saving registers for NMI | These registers save PC and PSW when NMI occurs. The high 8 bits of the FEPC and the high 24 bits of the FEPSW are fixed to 0. |
| 3 | FEPSW | | |
| 4 | ECR | Interrupt source register | If exception, maskable interrupt, or NMI occurs, this register will contain information referencing the interrupt source. The high 16 bits of this register are called FECC, to which exception code of NMI is set. The low 16 bits are called EICC to which exception code of exception/interrupt is set (refer to Figure 3-2). |
| 5 | PSW | Program status word | Program status word is a collection of flags that indicate program status (instruction execution result) and CPU status (refer to Figure 3-3). |
| 6 to 31 | Reserved | | |

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

Figure 3-2. Interrupt Source Register (ECR)



| Bit Position | Bit Name | Meaning |
|--------------|----------|---|
| 31 to 16 | FECC | Fatal Error Cause Code Exception code of NMI (Refer to Table 5-1 Interrupt List) |
| 15 to 0 | EICC | Exception/Interrupt Cause Code Exception code of exception/interrupt (Refer to Table 5-1 Interrupt List) |

Figure 3-3. Program Status Word (PSW)

| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 31 to 8 | RFU | Reserved field (fixed to 0) |
| 7 | NP | NMI Pending Indicates that NMI processing is in progress. This flag is set when NMI is accepted, and disables multiple interrupts. |
| 6 | EP | Exception Pending Indicates that exception processing is in progress. This flag is set when exception is generated and does not accept maskable interrupt request. |
| 5 | ID | Interrupt Disable Indicates that accepting external interrupt request is disabled. |
| 4 | SAT | Saturated Math This flag is set if result of executing saturated operation instruction overflows (if overflow does not occur, value of previous operation is held). |
| 3 | CY | Carry This flag is set if carry or borrow occurs as result of operation (if carry or borrow does not occur, it is reset). |
| 2 | OV | Overflow This flag is set if overflow occurs during operation (if overflow does not occur, it is reset). |
| 1 | S | Sign This flag is set if result of operation is negative. It is reset if result is positive. |
| 0 | Z | Zero This flag is set if result of operation is zero (if result is not zero, it is reset). |

3.3 Operation Modes

3.3.1 Operation modes

The V852 has the following operations modes. These modes are selected by the MODE0 and MODE1 pins.

(1) Single-chip mode

In single-chip mode, after the system has been released from the reset status, the pins related to the bus interface are set for I/O port mode, execution branches to the reset entry address of the internal ROM/PROM, and instruction processing is started. The external expansion mode can be set in which external devices can be connected to the external memory area by setting the memory expansion mode register (MM) using instructions (Refer to **3.4.6 (1) External expansion mode register (MM)**).

(2) ROM-less mode

After the system reset has been released from the reset status, the pins related to the bus interface are set for the control mode, execution branches to the external device (memory) reset entry addresses, and instruction processing is started. Instruction fetch and data access from internal ROM/PROM are disabled.

(3) PROM programming mode

This mode is provided only for the PROM version. In PROM programming mode, the appropriate pins function to provide a μ PD27C1001A compatible interface. By using a PROM programmer, the internal PROM of the V852 can be programmed.

(4) PROM read mode

This mode is provided only for the PROM version. In PROM read mode, the appropriate pins function to provide a μ PD27C1001A compatible interface. By using a PROM programmer, the internal PROM of the V852 can be read.

3.3.2 Specifying operation mode

The operation mode of the V852 is specified by using the MODE0 and MODE1 pins. Set these pins in the application system. Do not change the setting of these pins during operation.

If the setting is changed during operation, the functionality is not guaranteed.

(1) In normal mode

| MODE1 | MODE0 | Operation Mode |
|-------|-------|------------------|
| 0 | 0 | ROM-less mode |
| 0 | 1 | RFU (reserved) |
| 1 | 0 | Single-chip mode |
| 1 | 1 | RFU (reserved) |

(2) In PROM mode

| V _{PP} | Pin Status | | Operation Mode |
|-----------------|------------|-------|------------------------------|
| | MODE1 | MODE0 | |
| 5 V | 0 | 0 | RFU (reserved) |
| | 0 | 1 | |
| | 1 | 0 | |
| | 1 | 1 | PROM mode (read mode) |
| 12.5 V | 1 | 1 | PROM mode (programming mode) |

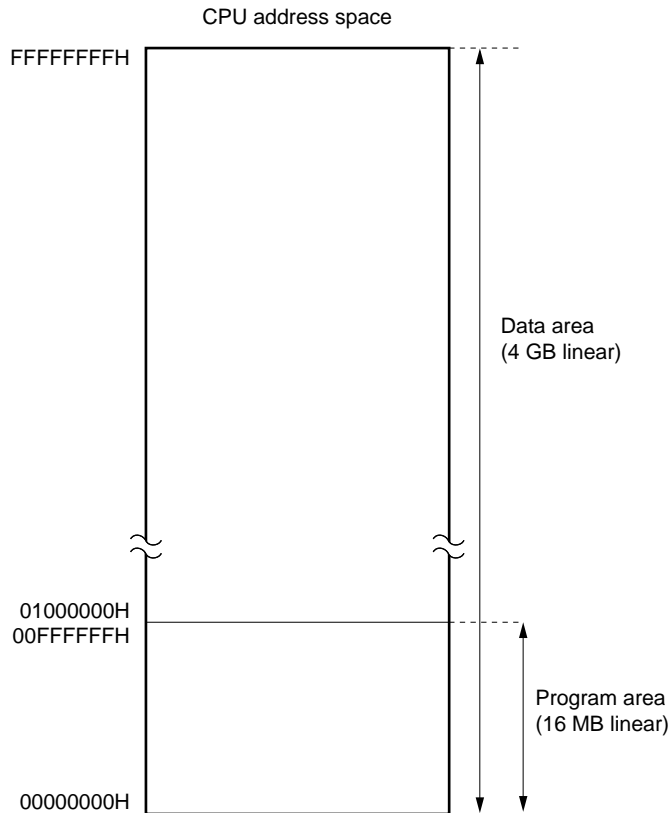
3.4 Address Space

3.4.1 CPU address space

The CPU of the V852 is of 32-bit architecture and supports up to 4 GB of linear address space (data space) during operand addressing (data access). When referencing instruction addresses, a linear address space (program space) of up to 16 MB is supported.

Figure 3-4 shows the CPU address space.

Figure 3-4. CPU Address Space

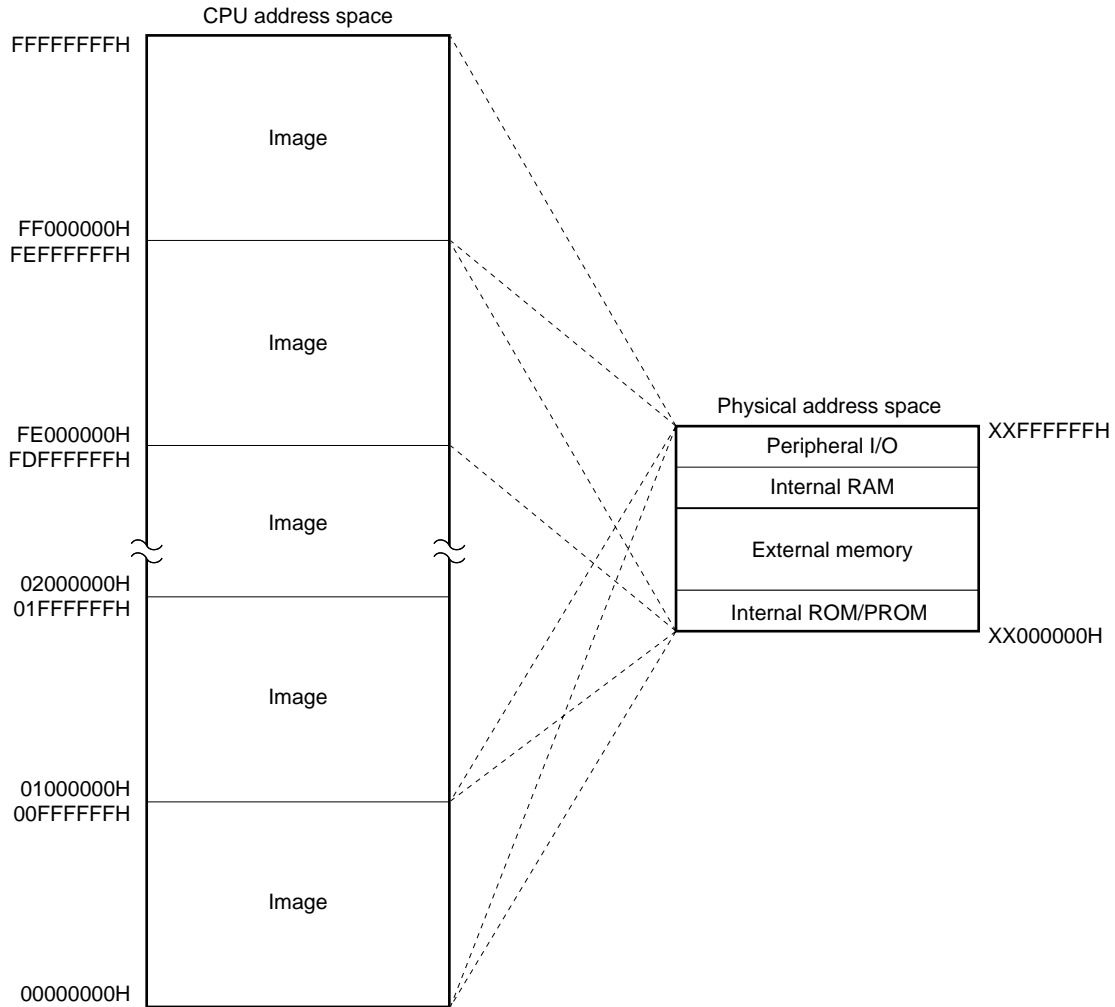


3.4.2 Image (Virtual Address Space)

The core CPU supports 4 GB of "virtual" addressing space, or 256 memory blocks, each containing 16-MB memory locations. In actuality, the same 16-MB block is accessed regardless of the values of bits 31 to 24 of the CPU address. Figure 3-5 shows the image of the virtual addressing space.

Because the higher 8 bits of a 32-bit CPU address are ignored and the CPU address is only seen as a 24-bit external physical address, the physical location XX000000H is equally referenced by multiple address values 00000000H, 01000000H, 02000000H... through FE000000H, FF000000H.

Figure 3-5. Image on Address Space



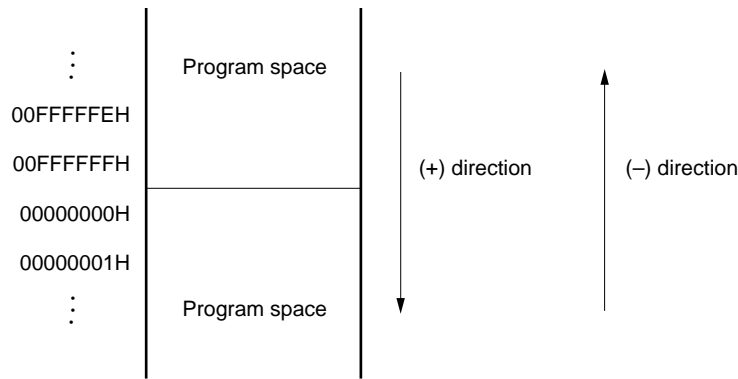
3.4.3 Wrap-around of CPU address space

(1) Program space

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

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address 00FFFFFFH are contiguous addresses. The condition in which the lower-limit address and upper-limit address of the program space are contiguous is called wrap-around.

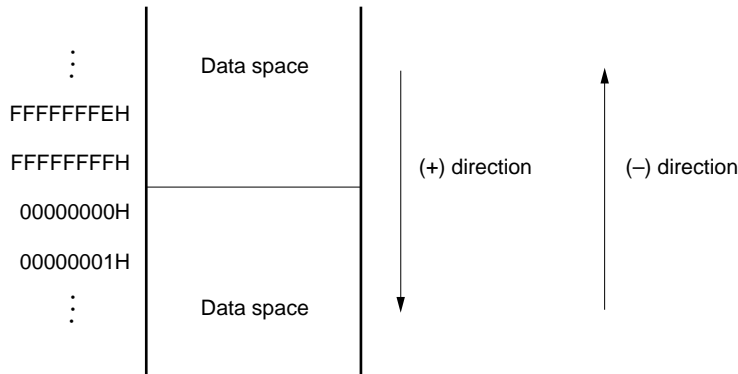
Caution No instruction can be fetched from the 4-KB area of 00FFF000H to 00FFFFFFH because this area is defined as peripheral I/O area. Therefore, do not execute any branch operation instructions in which the destination address will reside in any part of this area.



(2) Data space

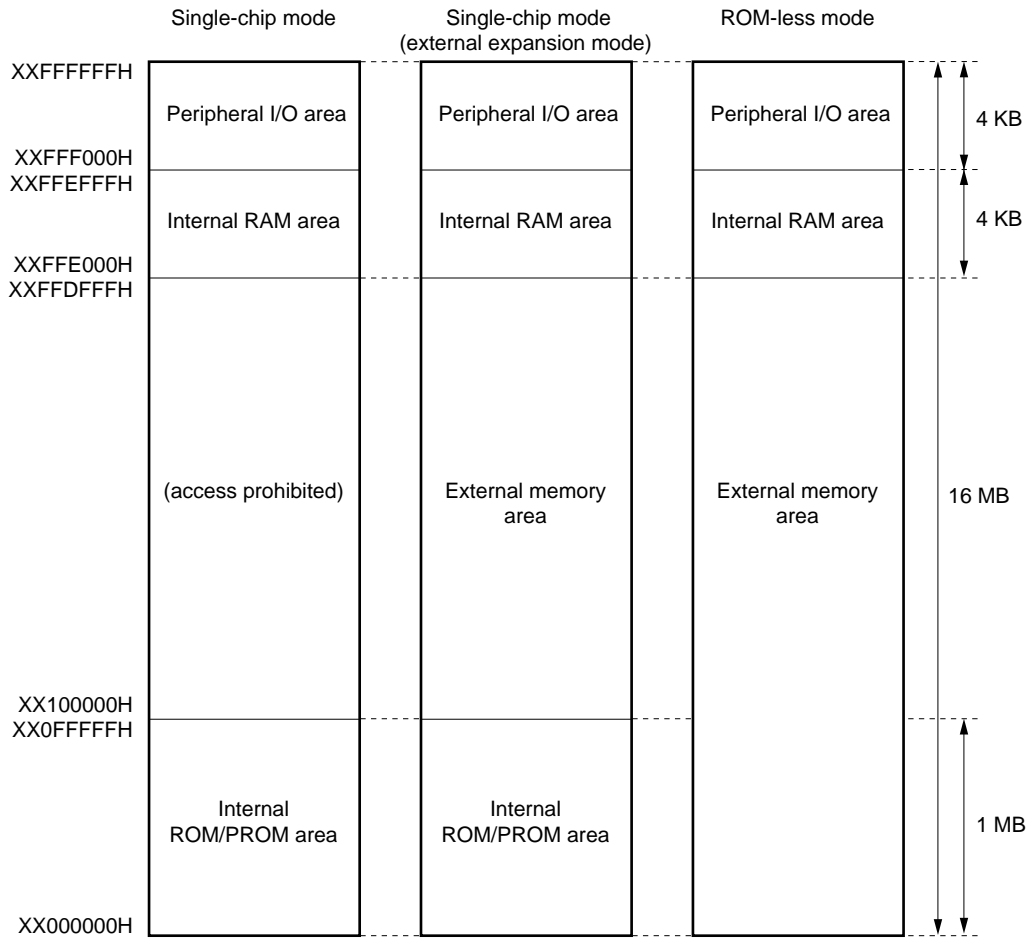
The result of operand address calculation that exceeds 32 bits is ignored.

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



3.4.4 Memory map

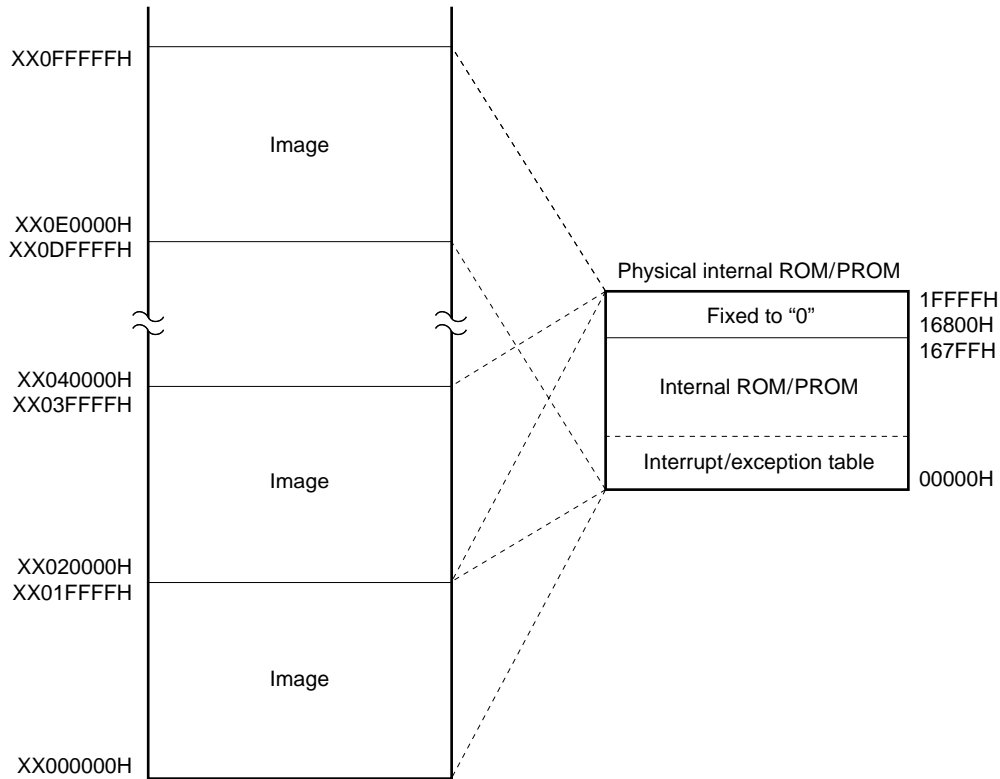
The V852 reserves areas as shown below. Each mode is specified by using the MODE0 and MODE1 pins (refer to **3.3 Operation Modes**).



3.4.5 Area

(1) Internal ROM/PROM area

A 1-MB area corresponding to addresses 000000H to 0FFFFFFH is reserved for the internal ROM/PROM area. The V852 is provided with a 90-KB area of addresses 000000H to 0167FFH as a physical internal ROM/PROM. The 38-KB area of addresses 016800H to 01FFFFH are fixed to "0". The image of 000000H to 01FFFFH is seen in the rest of the area (020000H to 0FFFFFFH)



Interrupt/exception table

The V852 increases the interrupt response speed by assigning destination addresses corresponding to interrupts/exceptions.

The collection of these destination addresses is called an interrupt/exception table, which is located in the internal ROM/PROM area. When an interrupt/exception request is granted, execution jumps to the corresponding destination address, and the program written at that memory address is executed. Figure 3-6 shows the names of interrupts/exceptions, and the corresponding addresses.

Figure 3-6. Interrupt/Exception Table

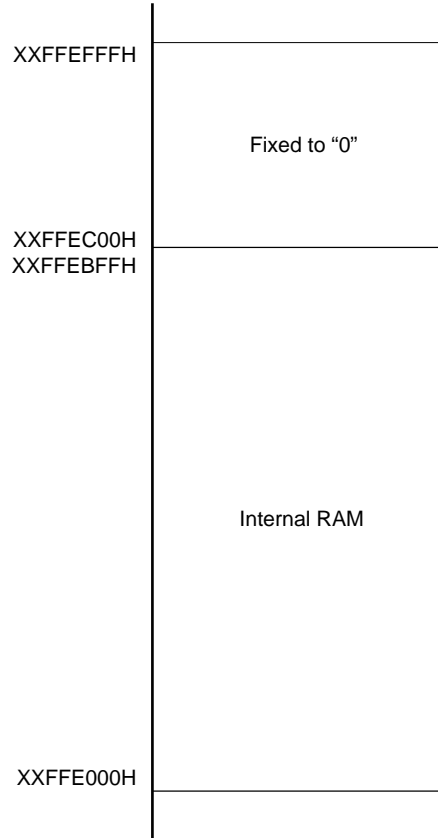
| Internal ROM/PROM area | |
|------------------------|----------------------|
| 00000170H | INTCSI2 |
| 00000160H | INTCSI1 |
| 00000150H | INTP03 |
| 00000140H | INTP02 |
| 00000130H | INTP01 |
| 00000120H | INTP00 |
| 00000110H | INTST0 |
| 00000100H | INTSR0 |
| 000000F0H | INTSER0 |
| 000000E0H | INTCSI0 |
| 000000D0H | INTCM4 |
| 000000C0H | INTP13/INTCC13 |
| 000000B0H | INTP12/INTCC12 |
| 000000A0H | INTP11/INTCC11 |
| 00000090H | INTP10/INTCC10 |
| 00000080H | INTOV1 |
| 00000060H | ILGOP |
| 00000050H | TRAP1n (n = 0 to FH) |
| 00000040H | TRAP0n (n = 0 to FH) |
| 00000010H | NMI |
| 00000000H | RESET |

← 16 bytes →

In the ROM-less mode, the internal ROM/PROM area is referenced as external memory area. To assure correct operation after reset, the destination address for the reset routine should be set to address 0 of the external memory.

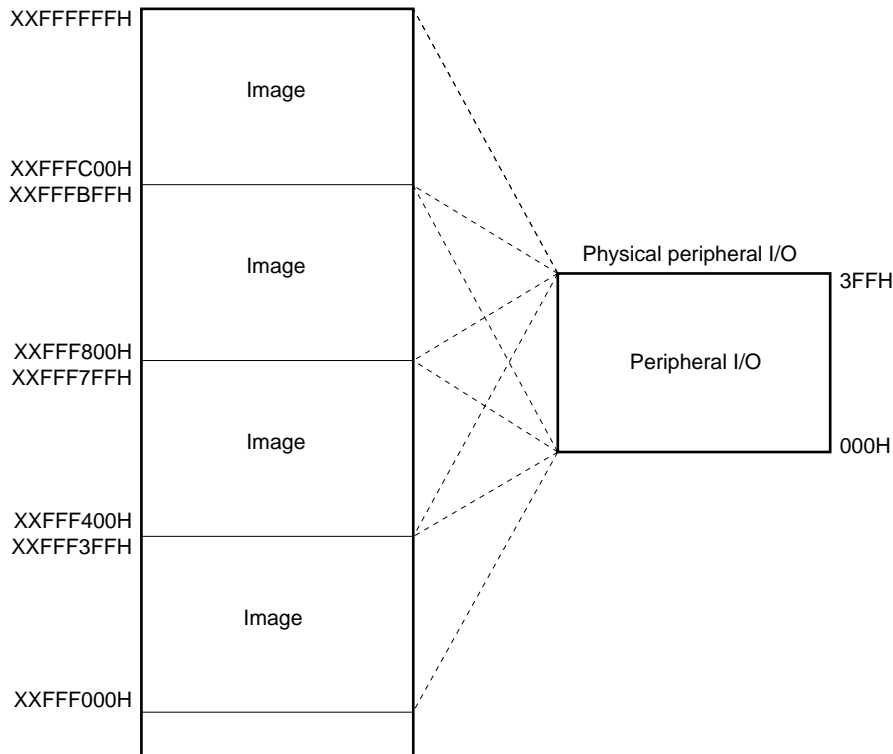
(2) Internal RAM area

A 4-KB area corresponding to addresses FFE000H through FFEFFFH is reserved as an internal RAM area. The V852 is provided with 3 KB of addresses FFE000H to FFEBFFH as a physical internal RAM area, and the rest of the area (FFEC00H to FFEFFFH) is fixed to "0".



(3) Peripheral I/O area

A 4-KB area of addresses FFF000H to FFFFFFFH is reserved as a peripheral I/O area. The V852 is provided with a 1-KB area of addresses FFF000H to FFF3FFH as a physical peripheral I/O area, and the image of FFF000H to FFF3FFH can be seen on the rest of the area (FFF400H to FFFFFFFH).



The peripheral I/O register assigned with functions such as on-chip peripheral hardware operation mode specifying function and state monitoring function are all memory-mapped to the peripheral I/O area. Instruction fetches are not allowed in this area.

- Cautions**
1. The least significant bit of an address is not decoded. If an odd address ($2n+1$) in the peripheral I/O area is referenced, the register at the next lowest even address ($2n$) will be accessed.
 2. The V852 does not have a peripheral I/O register than can be accessed in word units. If a register is accessed with a word operation, the effects will be limited to the halfword referenced by the instruction.
 3. If a register that can be accessed in byte units is accessed in half-word units, the higher 8 bits become undefined, if the access is a read operation. If a write access is made, only the data in the lower 8 bits is written to the register.
 4. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed.

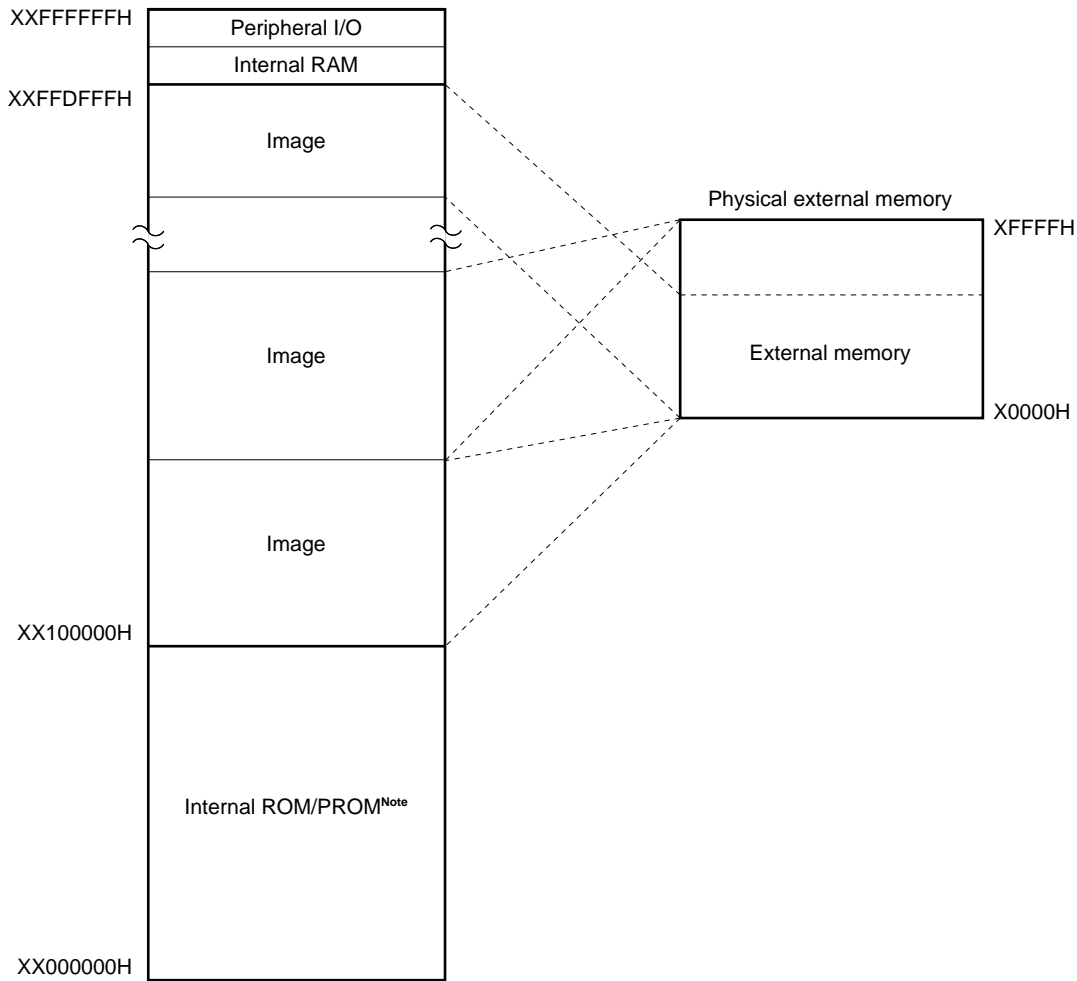
(4) External memory area

The V852 can use an area of up to xx100000H to xxFFDFFFH in the single-chip mode and an area of up to xx000000H to xxFFDFFFH in the ROM-less mode, for external memory accesses.

In the external memory area, 64 KB, 256 KB, 1 MB, 4 MB, or 16 MB of physical external memory can be allocated when the external expansion mode is specified. The same image as that of the physical external memory can be seen continuously on the external memory area, as shown in Figures 3-7 through 3-9, when the memory is not fully expanded (to 16 MB).

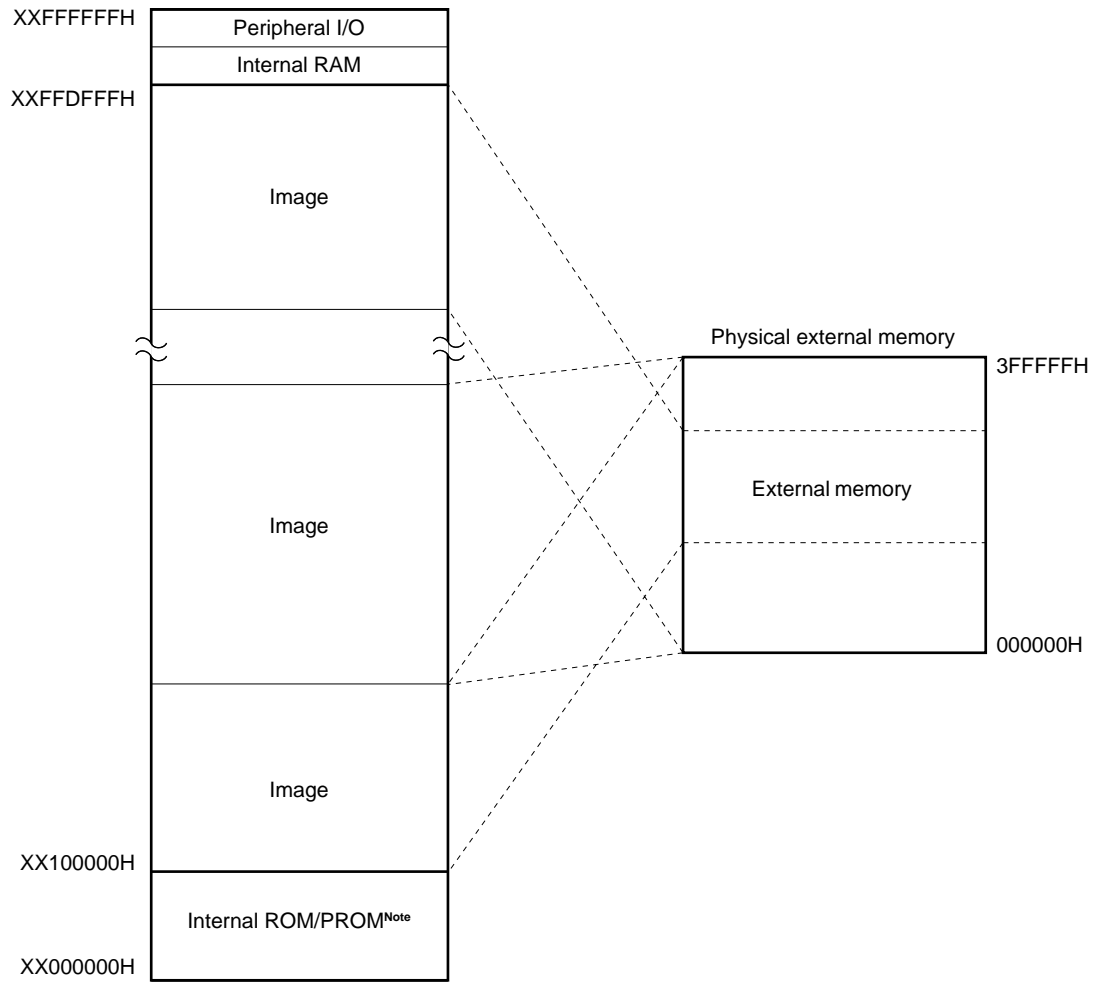
The internal RAM area, peripheral I/O area, and internal ROM/PROM area in the single-chip mode are not subject to external memory access.

Figure 3-7. External Memory Area (when expanded to 64 KB, 256 KB, or 1 MB)



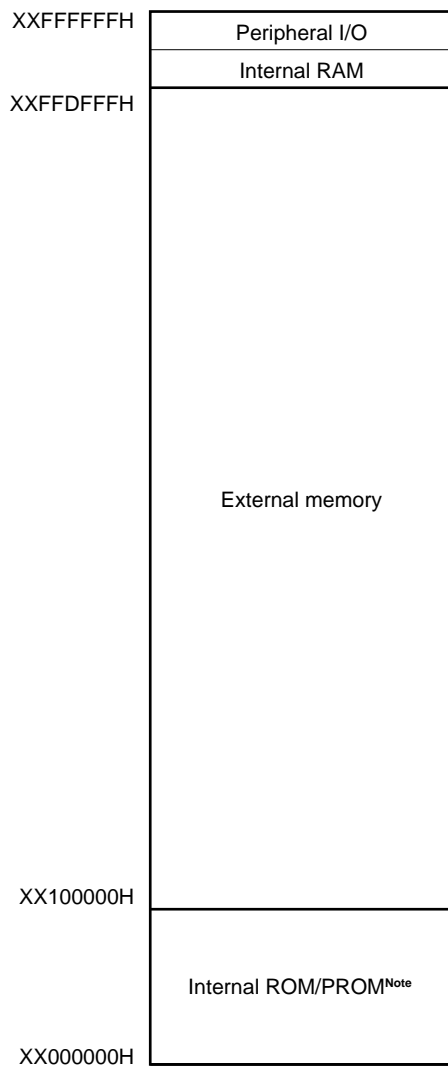
Note The image of the physical external memory can be seen continuously in the ROM-less mode.

Figure 3-8. External Memory Area (when expanded to 4 MB)



Note The image of the physical external memory can be seen continuously in the ROM-less mode.

Figure 3-9. External Memory Area (when fully expanded)



Note It becomes the external memory area in the ROM-less mode.

3.4.6 External expansion mode

The V852 allows external devices to be connected to the external memory space by using the pins of ports 4 through 10. To connect an external device, the port pins must be set in the external expansion mode by using the MODE0 and MODE1 pins and memory expansion mode register (MM). The MODE0 and MODE1 pins specify the operation mode of the V852. When MODE0 = 0 and MODE1 = 0, the V852 is set in the ROM-less mode; when MODE0 = 0 and MODE1 = 1, the single-chip mode is used.

In ROM-less mode, pins in the port4 to port6 and the P90 to P94 become the control mode after reset, thereby enabling the external memory.

In single-chip mode, port/control mode alternate function pins are in the port mode, thereby disabling the external device. When using an external device (external expansion mode), set the MM register by programming. In addition, when using the bus hold function, set the PMC10 register to control mode.

(1) Memory expansion mode register (MM)

This register sets the mode of each pin of ports 4 through 9. In the external expansion mode, an external device can be connected to the external memory area of up to 16 MB. However, the external device cannot be connected to the internal RAM area, peripheral I/O area, and internal ROM/PROM area in the single-chip mode (Not accessible even if connected physically).

The MM register can be read/written in 8- or 1-bit units. Bits 7, 5, and 4 of this register are fixed to 1.

| | | | | | | | | | | |
|----|---|---|---|---|-----|-----|-----|-----|-----------------------|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| MM | 1 | 0 | 1 | 1 | MM3 | MM2 | MM1 | MM0 | Address FFFFFF04CH | At reset B7H (in ROM-less mode) B0H (in single-chip mode) |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|-------------------------|---|------------------|--------------------------|-------------|---------------|--|-----------|--------|---------------------|---|-------------------------|-----|-----|-----------|--|--|--|---|---|---|-----------------|------------|-------------|------------|--|---|---|---|------------------|---|---|---|----------------|------------|---|---|---|----------------|---|---|---|-----------------|--------------------------|--------|--|--|--|----------------|--|--|--|
| 3 | MM3 | Memory Expansion Mode Specifies operation mode of P95 and P96 of port 9. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MM3</th> <th>Operation mode</th> <th>P95</th> <th>P96</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Port mode</td> <td colspan="2">Port</td> </tr> <tr> <td>1</td> <td>External expansion mode</td> <td>ST0</td> <td>ST1</td> </tr> </tbody> </table> | MM3 | Operation mode | P95 | P96 | 0 | Port mode | Port | | 1 | External expansion mode | ST0 | ST1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| MM3 | Operation mode | P95 | P96 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | Port mode | Port | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | External expansion mode | ST0 | ST1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 to 0 | MM2 to MM0 | Memory Expansion Mode Specifies operation mode of ports 4, 5, 6, and 9 (P90 to P94). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MM2</th> <th>MM1</th> <th>MM0</th> <th>Address space</th> <th>Port 4</th> <th>Port 5</th> <th>Port 6</th> <th>Port 9 (P90 to P94)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>–</td> <td colspan="4">Port mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>64-KB expansion</td> <td rowspan="2">AD0 to AD7</td> <td rowspan="2">AD8 to AD15</td> <td rowspan="2">A16 A17</td> <td rowspan="6"> $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, $\text{R}/\overline{\text{W}}$, $\overline{\text{DSTB}}$, ASTB </td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>256-KB expansion</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1-MB expansion</td> <td rowspan="2">A18 A19</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>4-MB expansion</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>16-MB expansion</td> <td>A20 A21 A22 A23</td> </tr> <tr> <td colspan="4">Others</td> <td colspan="4">RFU (reserved)</td> </tr> </tbody> </table> | MM2 | MM1 | MM0 | Address space | Port 4 | Port 5 | Port 6 | Port 9 (P90 to P94) | 0 | 0 | 0 | – | Port mode | | | | 0 | 1 | 1 | 64-KB expansion | AD0 to AD7 | AD8 to AD15 | A16 A17 | $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, $\text{R}/\overline{\text{W}}$, $\overline{\text{DSTB}}$, ASTB | 1 | 0 | 0 | 256-KB expansion | 1 | 0 | 1 | 1-MB expansion | A18 A19 | 1 | 1 | 0 | 4-MB expansion | 1 | 1 | 1 | 16-MB expansion | A20 A21 A22 A23 | Others | | | | RFU (reserved) | | | |
| MM2 | MM1 | MM0 | Address space | Port 4 | Port 5 | Port 6 | Port 9 (P90 to P94) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | – | Port mode | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | 64-KB expansion | AD0 to AD7 | AD8 to AD15 | A16 A17 | $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, $\text{R}/\overline{\text{W}}$, $\overline{\text{DSTB}}$, ASTB | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | 256-KB expansion | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 1 | 1-MB expansion | A18 A19 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 0 | 4-MB expansion | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | 16-MB expansion | A20 A21 A22 A23 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Others | | | | RFU (reserved) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

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

3.4.7 Recommended use of address space

The architecture of the V852 requires that a register that serves as a pointer be secured for address generation when executing operand data access in a data space. The operand data access can be performed directly from an instruction for ± 32 -Kbyte addresses in the pointer register. But general-purpose registers used as pointer registers have a limit. By minimizing performance degradation due to address calculations when changing a pointer value, the number of usable general registers for handling variables is maximized, and the program size can be saved.

To enhance the efficiency of using the pointer in connection with the memory map of the V852, the following points are recommended:

(1) Program space

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

(2) Data space

For the efficient use of resources to be performed through the wrap-around feature of the data space, the continuous 8-MB address spaces 00000000H to 007FFFFFFH and FF800000H to FFFFFFFFH of the 4-GB CPU are used as the data space. With the V852, 16-MB physical address space is seen as 256 images in the 4-GB CPU address space. The highest bit (bit 23) of this 24-bit address is assigned as address sign-extended to 32 bits.

Application of wrap-around

For example, when R = r0 (zero register) is specified for the LD/ST disp 16 [R] instruction, an addressing range of $00000000H \pm 32$ KB can be referenced with the sign-extended, 16-bit displacement value. By mapping the external memory in the 24-KB area in the figure, all resources including on-chip hardware can be accessed with one pointer.

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

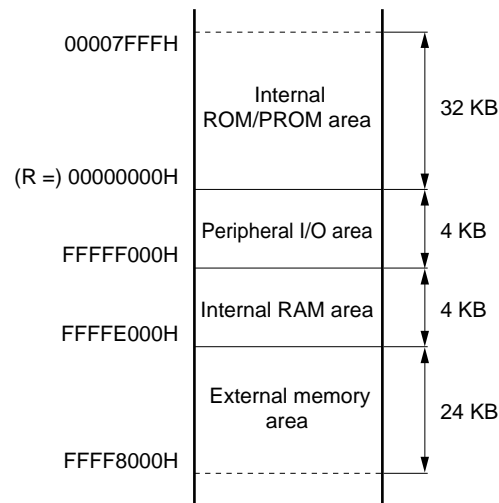
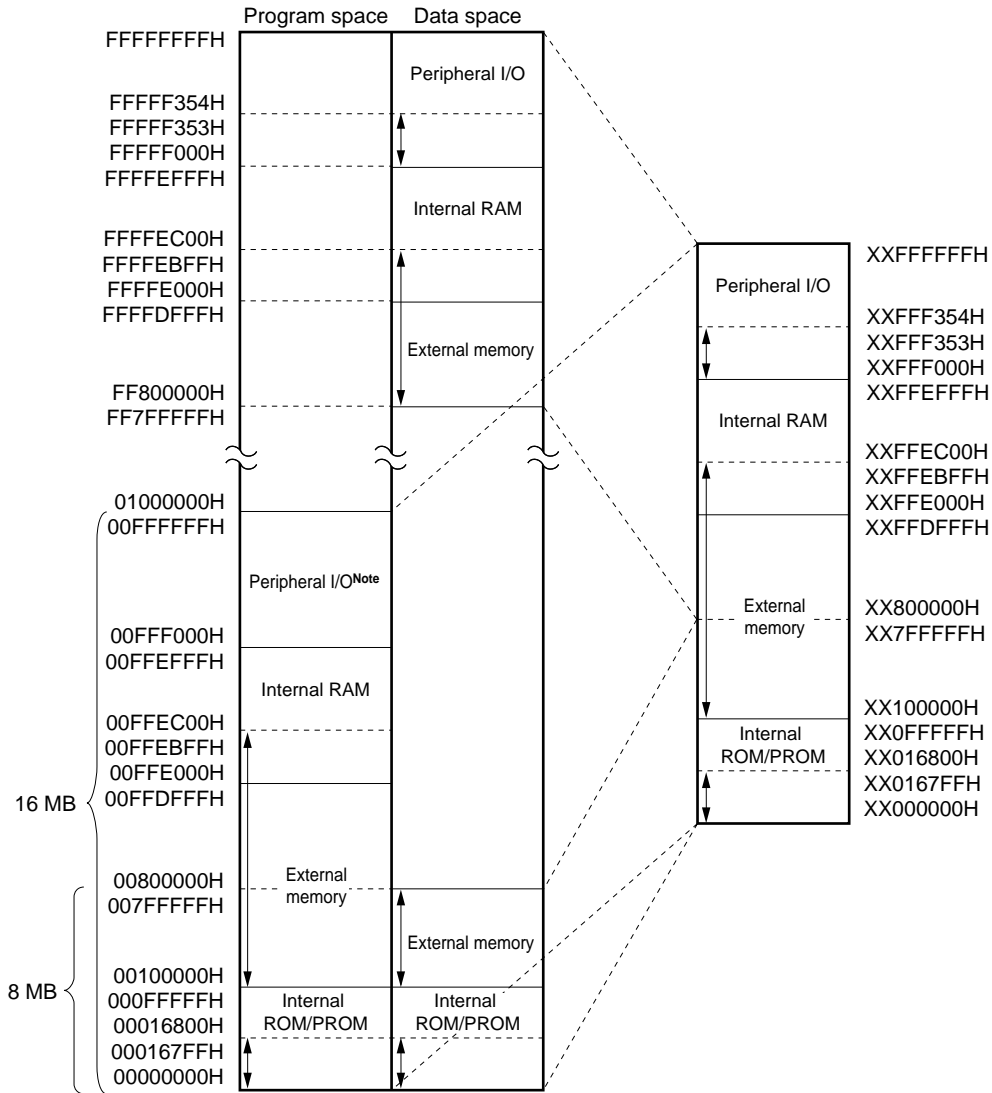


Figure 3-10. Recommended Memory Map



Note This area cannot be used as a program area.

Remark The areas shown by arrows indicate the recommended area.

3.4.8 Peripheral I/O registers

| Address | Function Register Name | Symbol | R/W | Bit Units for Manipulation | | | After reset |
|------------|--|--------|-----|----------------------------|--------|-----------|-------------|
| | | | | 1 bit | 8 bits | 16 bits | |
| FFFFFF00H | Port 0 | P0 | R/W | ○ | ○ | | Undefined |
| FFFFFF02H | Port 1 | P1 | | ○ | ○ | | |
| FFFFFF04H | Port 2 | P2 | | ○ | ○ | | |
| FFFFFF06H | Port 3 | P3 | | ○ | ○ | | |
| FFFFFF08H | Port 4 | P4 | | ○ | ○ | | |
| FFFFFF0AH | Port 5 | P5 | | ○ | ○ | | |
| FFFFFF0CH | Port 6 | P6 | | ○ | ○ | | |
| FFFFFF012H | Port 9 | P9 | | ○ | ○ | | |
| FFFFFF014H | Port 10 | P10 | | ○ | ○ | | |
| FFFFFF020H | Port 0 mode register | PM0 | | ○ | ○ | | |
| FFFFFF022H | Port 1 mode register | PM1 | | ○ | ○ | | |
| FFFFFF024H | Port 2 mode register | PM2 | | ○ | ○ | | |
| FFFFFF026H | Port 3 mode register | PM3 | | ○ | ○ | | |
| FFFFFF028H | Port 4 mode register | PM4 | | ○ | ○ | | |
| FFFFFF02AH | Port 5 mode register | PM5 | | ○ | ○ | | |
| FFFFFF02CH | Port 6 mode register | PM6 | | ○ | ○ | | |
| FFFFFF032H | Port 9 mode register | PM9 | | ○ | ○ | | |
| FFFFFF034H | Port 10 mode register | PM10 | | ○ | ○ | | |
| FFFFFF040H | Port 0 mode control register | PMC0 | | ○ | ○ | | 00H |
| FFFFFF044H | Port 2 mode control register | PMC2 | | ○ | ○ | | 01H |
| FFFFFF046H | Port 3 mode control register | PMC3 | | ○ | ○ | | 00H |
| FFFFFF04CH | Memory expansion mode register | MM | | ○ | ○ | | B0H/B7H |
| FFFFFF054H | Port 10 mode control register | PMC10 | | ○ | ○ | | 00H |
| FFFFFF060H | Data wait control register | DWC | | | | ○ | FFFFH |
| FFFFFF062H | Bus cycle control register | BCC | | | | ○ | AAAAH |
| FFFFFF070H | Power save control register | PSC | | ○ | ○ | | 00H |
| FFFFFF078H | System status register | SYS | | ○ | ○ | | 0000000XB |
| FFFFFF084H | Baud rate generator register 0 | BRG0 | | ○ | ○ | | Undefined |
| FFFFFF086H | Baud rate generator prescaler mode register 0 | BPRM0 | | ○ | ○ | | 00H |
| FFFFFF088H | Clocked serial interface mode register 0 | CSIM0 | | ○ | ○ | | |
| FFFFFF08AH | Serial I/O shift register 0 | SIO0 | | ○ | ○ | | Undefined |
| FFFFFF094H | Baud rate generator register 1 | BRG1 | | ○ | ○ | | |
| FFFFFF096H | Baud rate generator prescaler mode register 1 | BPRM1 | ○ | ○ | | 00H | |
| FFFFFF098H | Clocked serial interface mode register 1 | CSIM1 | ○ | ○ | | | |
| FFFFFF09AH | Serial I/O shift register 1 | SIO1 | ○ | ○ | | Undefined | |
| FFFFFF0A8H | Clocked serial interface mode register 2 | CSIM2 | ○ | ○ | | 00H | |
| FFFFFF0AAH | Serial I/O shift register 2 | SIO2 | ○ | ○ | | Undefined | |
| FFFFFF0C0H | Asynchronous serial interface mode register 00 | ASIM00 | ○ | ○ | | 80H | |

| Address | Function Register Name | Symbol | R/W | Bit Units for Manipulation | | | After reset | |
|-----------|---|--------|-----|----------------------------|--------|---------|-------------|-----------|
| | | | | 1 bit | 8 bits | 16 bits | | |
| FFFFF0C2H | Asynchronous serial interface mode register 01 | ASIM01 | R/W | ○ | ○ | | 00H | |
| FFFFF0C4H | Asynchronous serial interface status register 0 | ASIS0 | | ○ | ○ | | | |
| FFFFF0C8H | Receive buffer 0 (9 bits) | RXB0 | R | | | ○ | Undefined | |
| FFFFF0CAH | Receive buffer 0L (lower 8 bits) | RXB0L | | ○ | ○ | | | |
| FFFFF0CCH | Transmit shift register 0 (9 bits) | TXS0 | | | | ○ | | |
| FFFFF0CEH | Transmit shift register 0L (lower 8 bits) | TXS0L | W | | ○ | | | |
| FFFFF100H | Interrupt control register | OVIC1 | | ○ | ○ | | 47H | |
| FFFFF102H | Interrupt control register | P1IC0 | | ○ | ○ | | | |
| FFFFF104H | Interrupt control register | P1IC1 | | ○ | ○ | | | |
| FFFFF106H | Interrupt control register | P1IC2 | | ○ | ○ | | | |
| FFFFF108H | Interrupt control register | P1IC3 | | ○ | ○ | | | |
| FFFFF10AH | Interrupt control register | CMIC4 | | ○ | ○ | | | |
| FFFFF10CH | Interrupt control register | CSIC0 | | ○ | ○ | | | |
| FFFFF10EH | Interrupt control register | SEIC0 | | ○ | ○ | | | |
| FFFFF110H | Interrupt control register | SRIC0 | R/W | ○ | ○ | | | |
| FFFFF112H | Interrupt control register | STIC0 | | ○ | ○ | | | |
| FFFFF114H | Interrupt control register | P0IC0 | | ○ | ○ | | | |
| FFFFF116H | Interrupt control register | P0IC1 | | ○ | ○ | | | |
| FFFFF118H | Interrupt control register | P0IC2 | | ○ | ○ | | | |
| FFFFF11AH | Interrupt control register | P0IC3 | | ○ | ○ | | | |
| FFFFF11CH | Interrupt control register | CSIC1 | | ○ | ○ | | | |
| FFFFF11EH | Interrupt control register | CSIC2 | | ○ | ○ | | | |
| FFFFF166H | In-service priority register | ISPR | R | ○ | ○ | | | 00H |
| FFFFF170H | Command register | PRCMD | W | ○ | ○ | | | Undefined |
| FFFFF180H | External interrupt mode register 0 | INTM0 | | ○ | ○ | | 00H | |
| FFFFF182H | External interrupt mode register 1 | INTM1 | | ○ | ○ | | | |
| FFFFF184H | External interrupt mode register 2 | INTM2 | | ○ | ○ | | | |
| FFFFF230H | Timer overflow status register | TOVS | R/W | ○ | ○ | | | |
| FFFFF240H | Timer unit mode register 1 | TUM1 | | | | ○ | 0000H | |
| FFFFF242H | Timer control register 1 | TMC1 | | ○ | ○ | | 00H | |
| FFFFF244H | Timer output control register 1 | TOC1 | | ○ | ○ | | | |
| FFFFF250H | Timer 1 | TM1 | R | | | ○ | 0000H | |
| FFFFF252H | Capture/compare register 10 | CC10 | | | | ○ | Undefined | |
| FFFFF254H | Capture/compare register 11 | CC11 | | | | ○ | | |
| FFFFF256H | Capture/compare register 12 | CC12 | R/W | | | ○ | | |
| FFFFF258H | Capture/compare register 13 | CC13 | | | | ○ | | |
| FFFFF342H | Timer control register 4 | TMC4 | | ○ | ○ | | 00H | |
| FFFFF350H | Timer 4 | TM4 | R | | | ○ | 0000H | |
| FFFFF352H | Compare register 4 | CM4 | R/W | | | ○ | Undefined | |

CHAPTER 4 BUS CONTROL FUNCTION

The V852 is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

4.1 Features

- 16-bit data bus
- External devices connected through multiplexed I/O port pins
- Wait function
 - Programmable wait function, capable of inserting up to 3 wait states per 2 blocks
 - External wait control through $\overline{\text{WAIT}}$ input pin
- Idle state insertion function
- Bus mastership arbitration function
- Bus hold function

4.2 Bus Control Pins

The following pins are used for interfacing to external devices:

| External Bus Interface Function | Corresponding Port (pins) |
|--|---------------------------|
| Address/data bus (AD0 to AD7) | Port 4 (P40 to P47) |
| Address/data bus (AD8 to AD15) | Port 5 (P50 to P57) |
| Address bus (A16 to A23) | Port 6 (P60 to P67) |
| Read/write control ($\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, R/W, $\overline{\text{DSTB}}$) | Port 9 (P90 to P93) |
| Address strobe (ASTB) | Port 9 (P94) |
| External wait control ($\overline{\text{WAIT}}$) | $\overline{\text{WAIT}}$ |
| Bus cycle status (ST0, ST1) | Port 9 (P95 to P96) |
| Bus hold control ($\overline{\text{HLDRQ}}$, $\overline{\text{HLDAK}}$) | Port 10 (P100 to P101) |

The bus interface function of each pin is enabled by the memory expansion mode register (MM). In ROM-less mode, the bus interface function of each pin is unconditionally enabled by the MODE0 and MODE1 inputs. For the details of specifying an operation mode of the external bus interface, refer to **3.4.6 (1) Memory expansion mode register (MM)**.

4.3 Bus Access

4.3.1 Number of access clocks

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

| Bus Cycle Type | Resource (bus width) | | | |
|---------------------|---------------------------|---------------------------|-----------------------------|------------------------------|
| | Internal ROM (32 bits) | Internal RAM (32 bits) | Peripheral I/O (16 bits) | External Memory (16 bits) |
| Instruction fetch | 1 | 3 | Disabled | 3 + n |
| Operand data access | 3 | 1 | 3 + n | 3 + n |

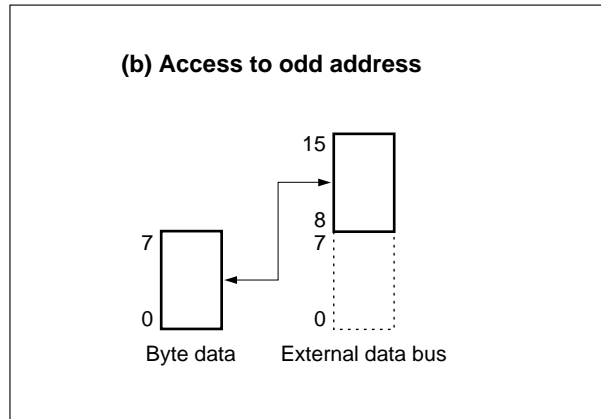
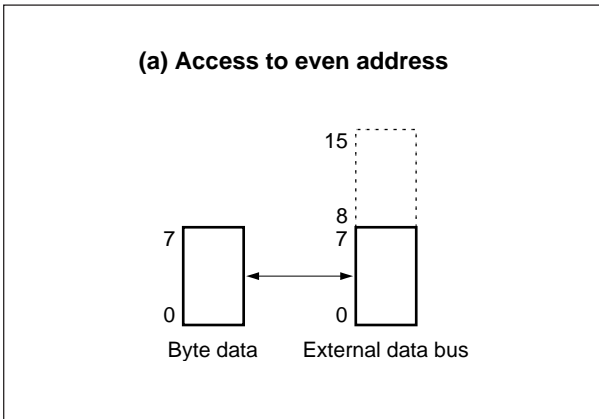
- Remarks**
1. Unit: clock/access
 2. n: number of inserted wait clock

★ **4.3.2 Bus width**

The V852 carries out peripheral I/O access and external memory access in 8-, 16-, or 32-bit units. The following shows the operation for each access.

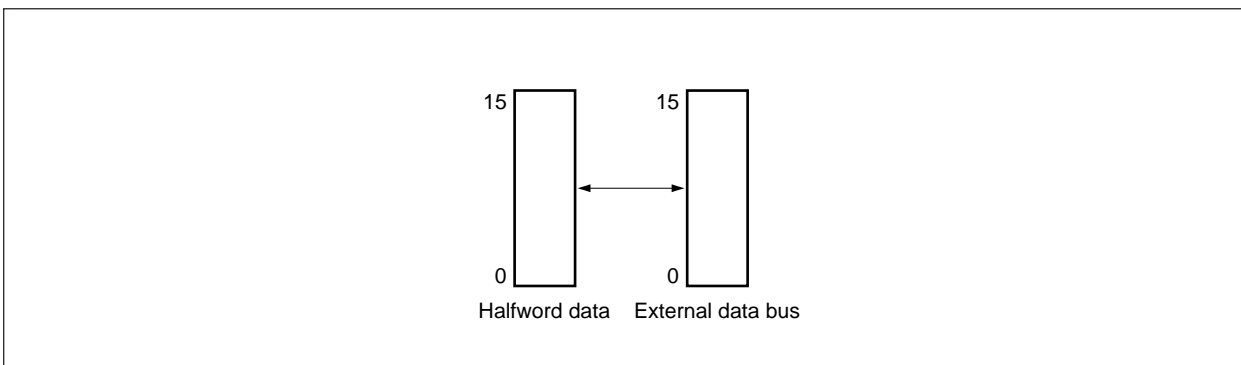
(1) Byte access (8 bits)

Byte access is divided into two types, the access to even address and the access to odd address.



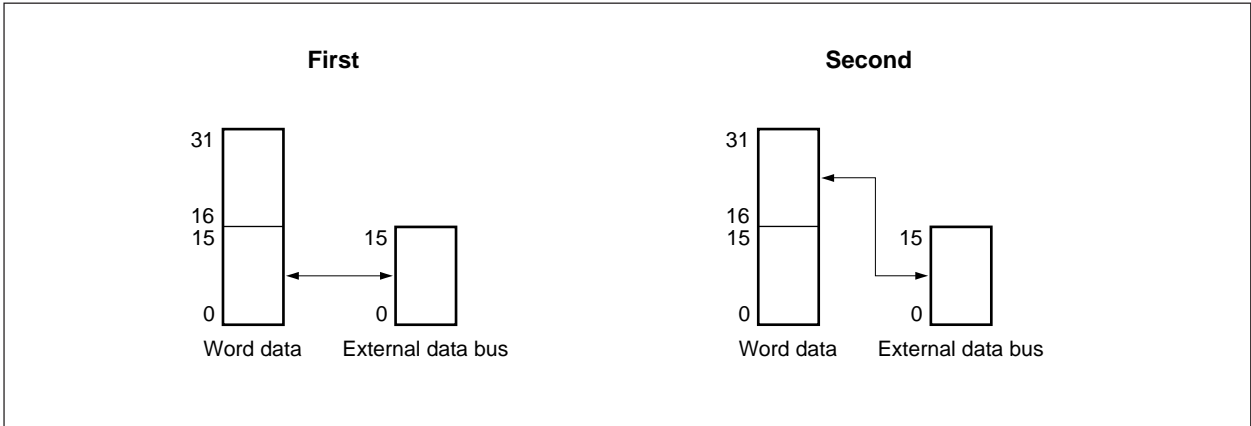
(2) Halfword access (16 bits)

In halfword access to external memory, data is dealt with as it is because the data bus is 16-bit fixed.



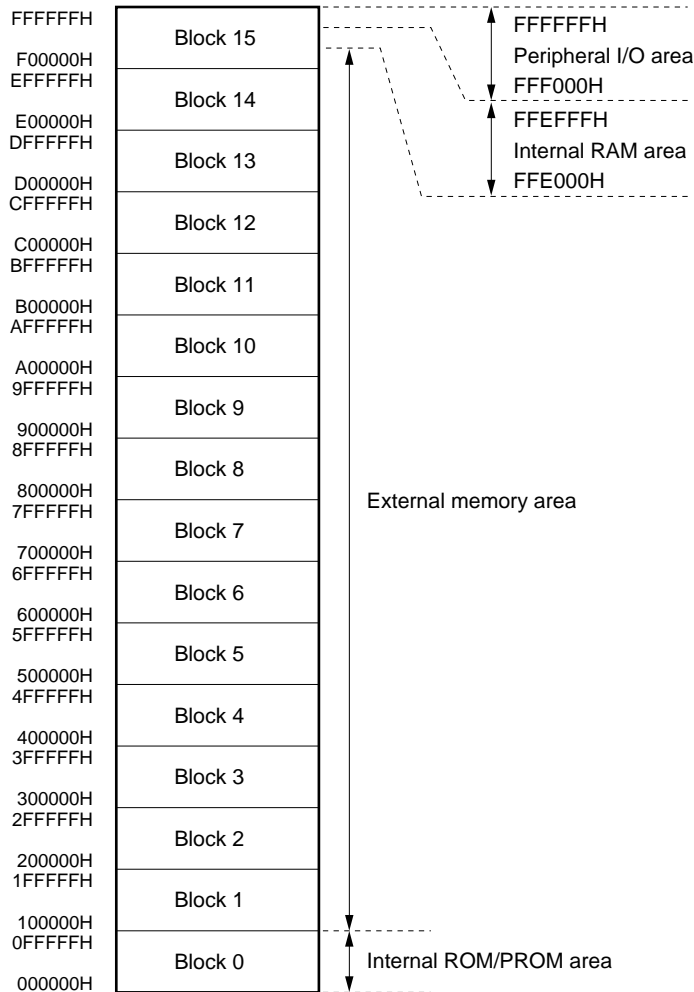
(3) Word access (32 bits)

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



4.4 Memory Block Function

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



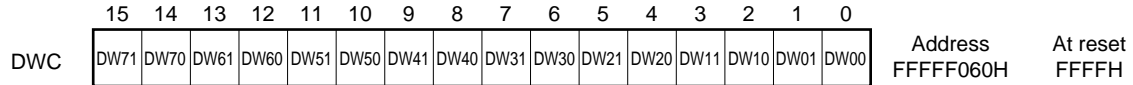
4.5 Wait Function

4.5.1 Programmable wait function

To facilitate interfacing with low-speed memories and I/O devices, up to 3 data wait states can be inserted in a bus cycle for two memory blocks. The number of wait states can be programmed by using data wait control register (DWC). Immediately after the system has been reset, three data wait states are automatically programmed for all memory blocks.

(1) Data wait control register (DWC)

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



| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|--|--|------|------|--------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|------------|---|------------|---|------------|---|------------|---|------------|---|--------------|---|--------------|---|--------------|
| 15 to 0 | DWn1 DWn0 (n = 0 to 7) | Data Wait Specifies number of wait states to be inserted <table border="1" style="border-collapse: collapse; margin: 10px auto; width: 80%;"> <thead> <tr> <th>DWn1</th> <th>DWn0</th> <th>Number of Wait States to be Inserted</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">2</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">3</td></tr> </tbody> </table> <table border="1" style="border-collapse: collapse; margin: 10px auto; width: 80%;"> <thead> <tr> <th>n</th> <th>Blocks into Which Wait States are Inserted</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td>Blocks 0/1</td></tr> <tr><td style="text-align: center;">1</td><td>Blocks 2/3</td></tr> <tr><td style="text-align: center;">2</td><td>Blocks 4/5</td></tr> <tr><td style="text-align: center;">3</td><td>Blocks 6/7</td></tr> <tr><td style="text-align: center;">4</td><td>Blocks 8/9</td></tr> <tr><td style="text-align: center;">5</td><td>Blocks 10/11</td></tr> <tr><td style="text-align: center;">6</td><td>Blocks 12/13</td></tr> <tr><td style="text-align: center;">7</td><td>Blocks 14/15</td></tr> </tbody> </table> | DWn1 | DWn0 | Number of Wait States to be Inserted | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 2 | 1 | 1 | 3 | n | Blocks into Which Wait States are Inserted | 0 | Blocks 0/1 | 1 | Blocks 2/3 | 2 | Blocks 4/5 | 3 | Blocks 6/7 | 4 | Blocks 8/9 | 5 | Blocks 10/11 | 6 | Blocks 12/13 | 7 | Blocks 14/15 |
| DWn1 | DWn0 | Number of Wait States to be Inserted | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| n | Blocks into Which Wait States are Inserted | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | Blocks 0/1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | Blocks 2/3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Blocks 4/5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Blocks 6/7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Blocks 8/9 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | Blocks 10/11 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Blocks 12/13 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | Blocks 14/15 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Cautions**
1. Block 0 is reserved for the internal ROM/PROM area in the single-chip mode. It is not subject to programmable wait control, regardless of the setting of DWC, and is always accessed without wait states.
 2. The internal RAM area of block 15 is not subject to programmable wait control and is always accessed without wait states. The peripheral I/O area of this block is not subject to programmable wait control, either. The only wait control is dependent upon the execution of each peripheral function.

4.5.2 External wait function

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

The external $\overline{\text{WAIT}}$ signal does not affect the access times of the internal ROM/PROM, internal RAM, and peripheral I/O areas. Input of the external $\overline{\text{WAIT}}$ signal can be done asynchronously to CLKOUT and is sampled at the falling edge of the clock in the T2 and TW states of a bus cycle. If the set-up and hold time of the $\overline{\text{WAIT}}$ input is not satisfied, the wait state may or may not be inserted in the next state.

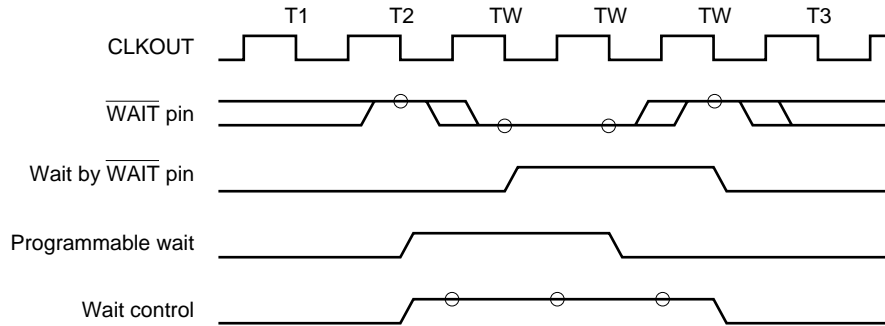
4.5.3 Relationships between programmable wait and external wait

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



For example, if the number of programmable wait states is 2 and the timing of the $\overline{\text{WAIT}}$ pin input signal is as illustrated below, three wait states will be inserted in the bus cycle.

Figure 4-1. Example of Inserting Wait States



Remark ○ : sampling timing

4.6 Idle State Insertion Function

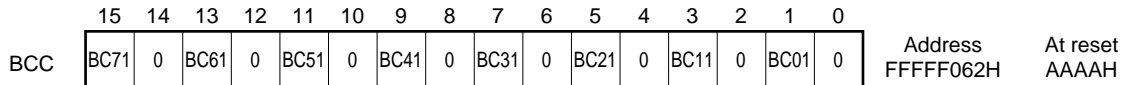
To facilitate interfacing with low-speed memory devices and meeting the data output float delay time (t_{DF}) on memory read accesses, one idle state (TI) can be inserted into the current bus cycle after the T3 state. The bus cycle following continuous bus cycles starts after one idle state.

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

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

(1) Bus cycle control register (BCC)

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



| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | |
|------------------------------|--|--|---|--|---|------------|---|------------|---|------------|---|------------|---|------------|---|--------------|---|--------------|---|--------------|
| 15, 13, 11, 9, 7, 5, 3, 1 | BCn1 (n = 0 to 7) | Bus Cycle Specifies insertion of idle state. 0: Not inserted 1: Inserted <table border="1" style="border-collapse: collapse; margin-left: 20px;"> <thead> <tr> <th style="width: 10%;">n</th> <th style="width: 90%;">Blocks into Which Idle State is Inserted</th> </tr> </thead> <tbody> <tr><td>0</td><td>Blocks 0/1</td></tr> <tr><td>1</td><td>Blocks 2/3</td></tr> <tr><td>2</td><td>Blocks 4/5</td></tr> <tr><td>3</td><td>Blocks 6/7</td></tr> <tr><td>4</td><td>Blocks 8/9</td></tr> <tr><td>5</td><td>Blocks 10/11</td></tr> <tr><td>6</td><td>Blocks 12/13</td></tr> <tr><td>7</td><td>Blocks 14/15</td></tr> </tbody> </table> | n | Blocks into Which Idle State is Inserted | 0 | Blocks 0/1 | 1 | Blocks 2/3 | 2 | Blocks 4/5 | 3 | Blocks 6/7 | 4 | Blocks 8/9 | 5 | Blocks 10/11 | 6 | Blocks 12/13 | 7 | Blocks 14/15 |
| n | Blocks into Which Idle State is Inserted | | | | | | | | | | | | | | | | | | | |
| 0 | Blocks 0/1 | | | | | | | | | | | | | | | | | | | |
| 1 | Blocks 2/3 | | | | | | | | | | | | | | | | | | | |
| 2 | Blocks 4/5 | | | | | | | | | | | | | | | | | | | |
| 3 | Blocks 6/7 | | | | | | | | | | | | | | | | | | | |
| 4 | Blocks 8/9 | | | | | | | | | | | | | | | | | | | |
| 5 | Blocks 10/11 | | | | | | | | | | | | | | | | | | | |
| 6 | Blocks 12/13 | | | | | | | | | | | | | | | | | | | |
| 7 | Blocks 14/15 | | | | | | | | | | | | | | | | | | | |

- Cautions**
1. Block 0 is reserved for the internal ROM/PROM area in the single-chip mode and therefore, no idle state is specified for this block.
 2. The internal RAM area and peripheral I/O area of block 15 are not subject to insertion of the idle state.
 3. Always set the BCC bits 0, 2, 4, 6, 8, 10, 12 and 14 to 0. Normal operation is not guaranteed when they are set to 1.

4.7 Bus Hold Function

4.7.1 Outline of function

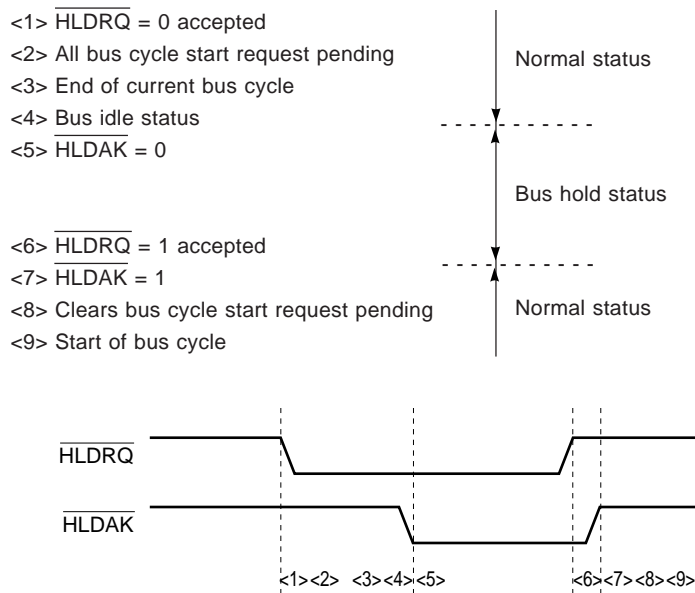
When P100 and P101 of port 10 are programmed to be in the control mode, the functions of the $\overline{\text{HLDRQ}}$ and $\overline{\text{HLDK}}$ pins become valid.

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

- ★ V852 internal operation continues until external memory is accessed during bus hold.
In the bus hold status, the $\overline{\text{HLDK}}$ pin becomes active (low).
This feature can be used to design a system where two or more bus masters exist, such as when a multi-processor configuration is used and when a DMA controller is connected.
- ★ However, bus hold requests are accepted neither between the first and the second word access, nor between read access and write access during read modify write access of bit manipulation instructions.

4.7.2 Bus hold procedure

The procedure of bus hold function is illustrated below.



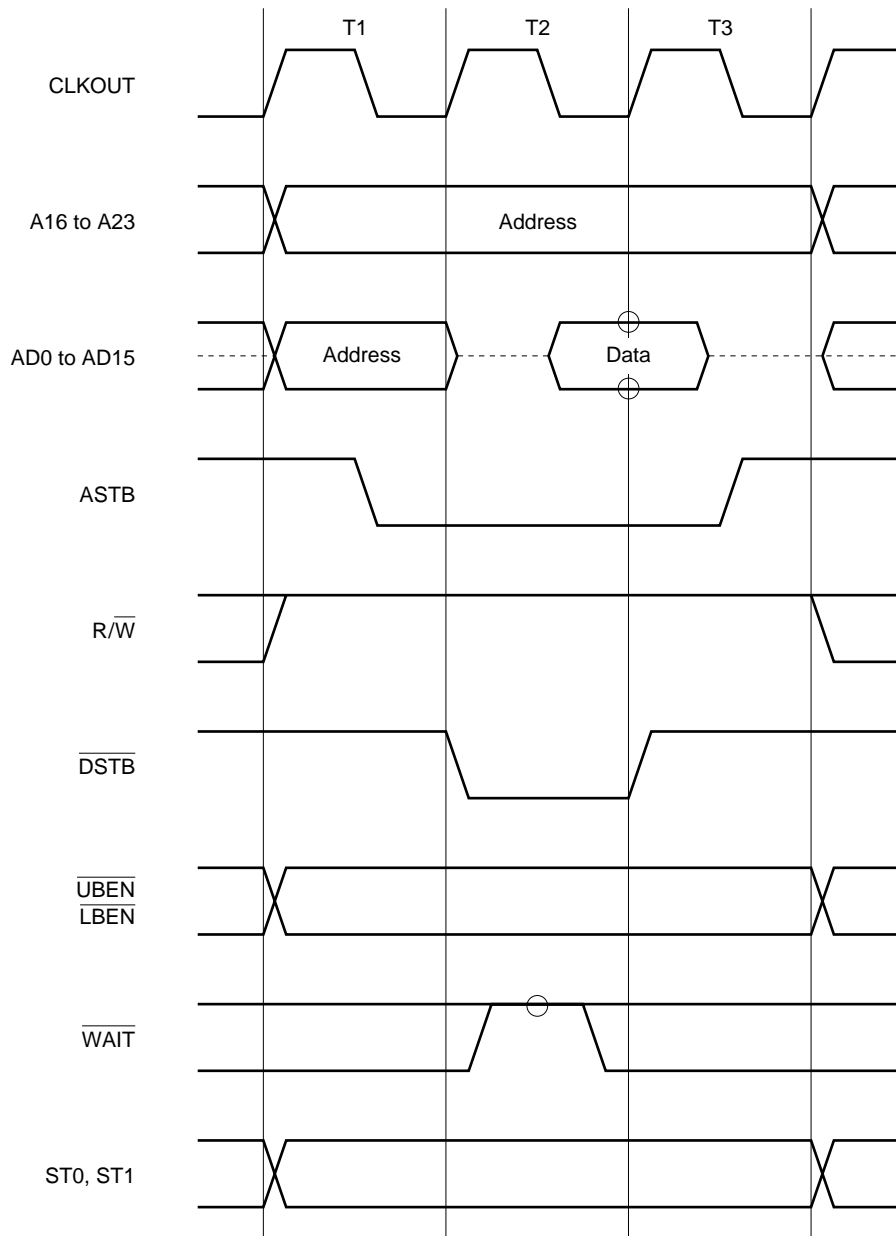
4.7.3 Operation in power save mode

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

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

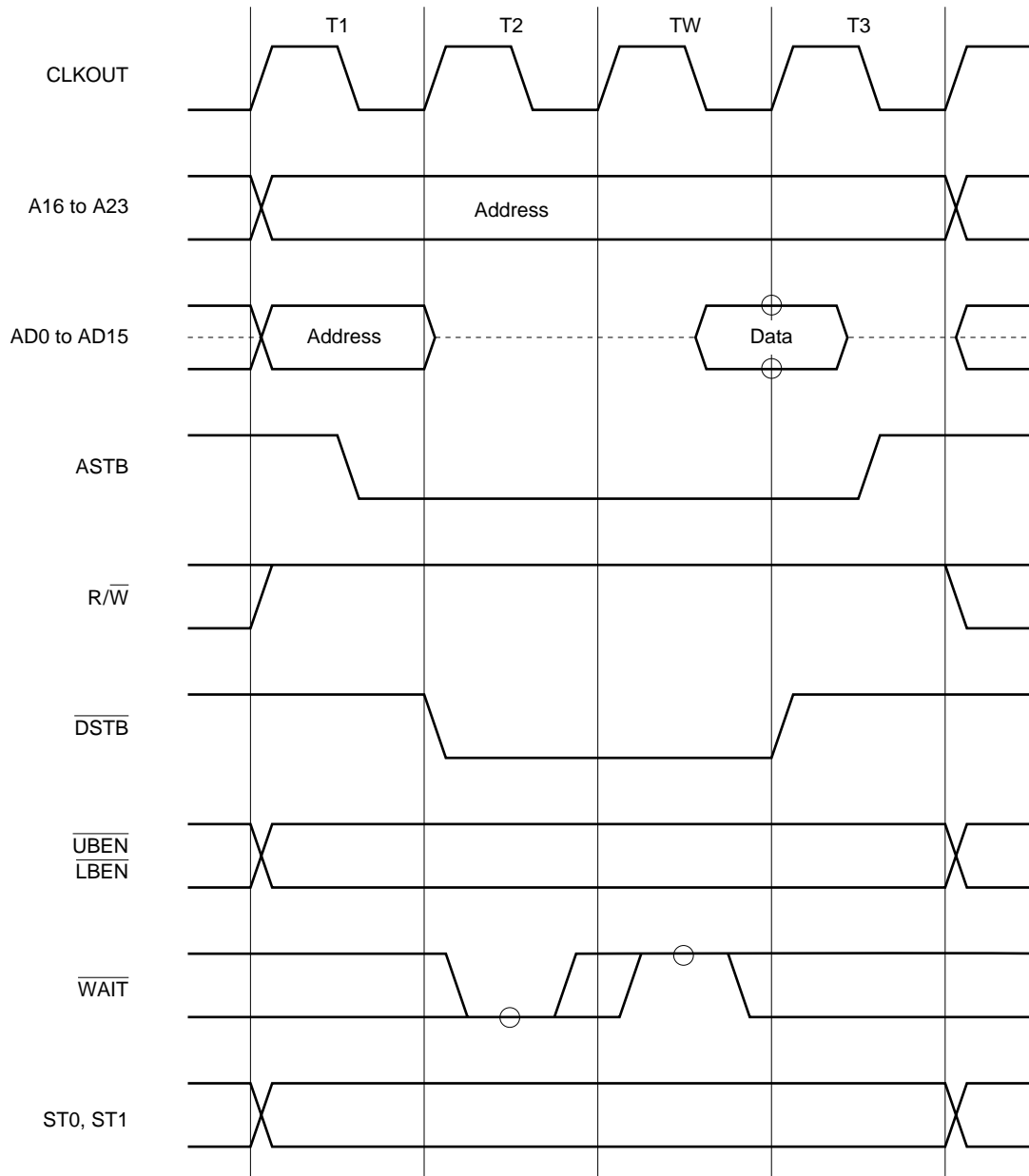
4.8 Bus Timing

(1) Memory read (0 wait)



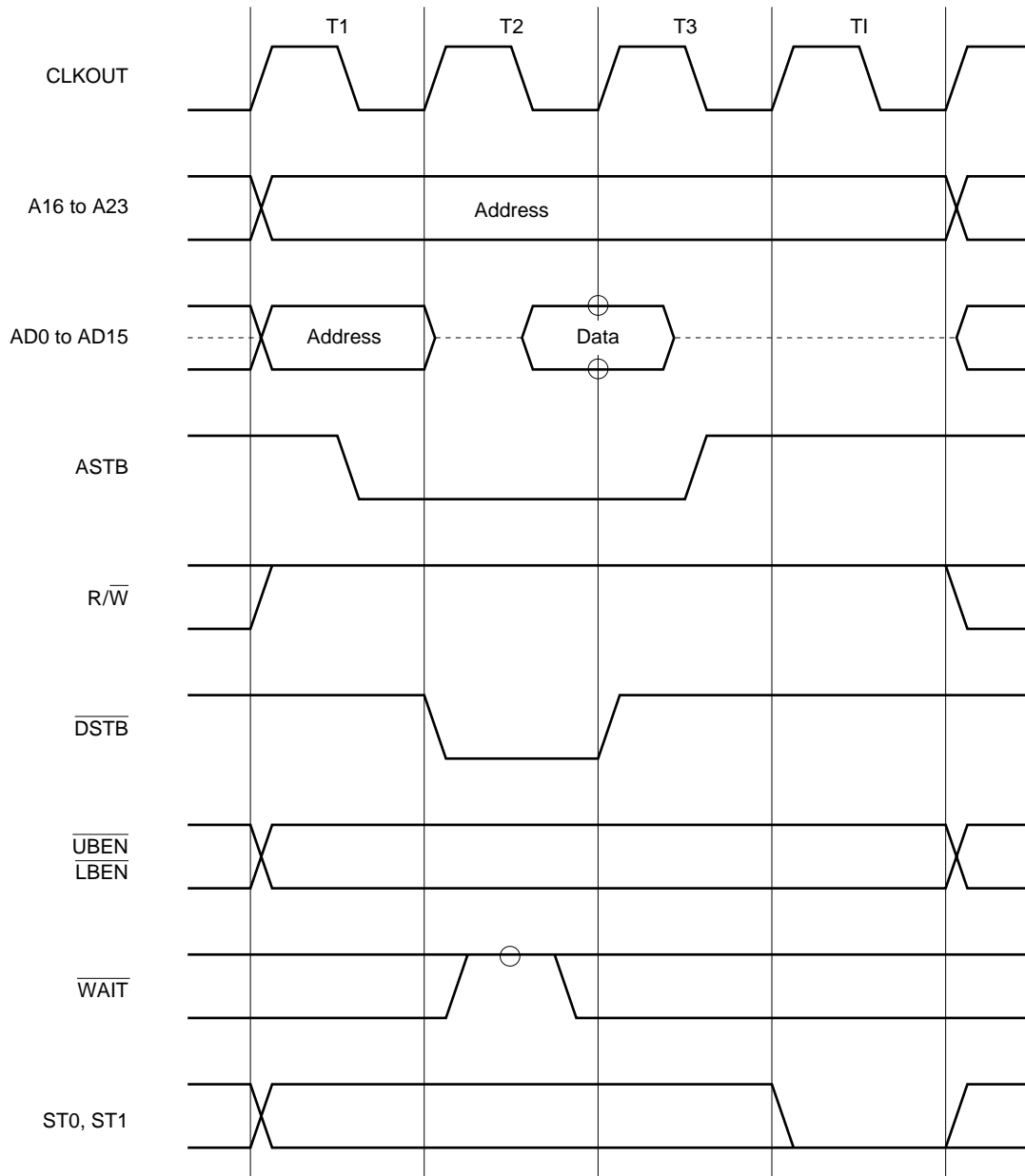
- Remarks**
- indicates the sampling timing when the number of programmable waits is set to 0.
 - The dotted line indicates the high-impedance state.

(2) Memory read (1 wait)



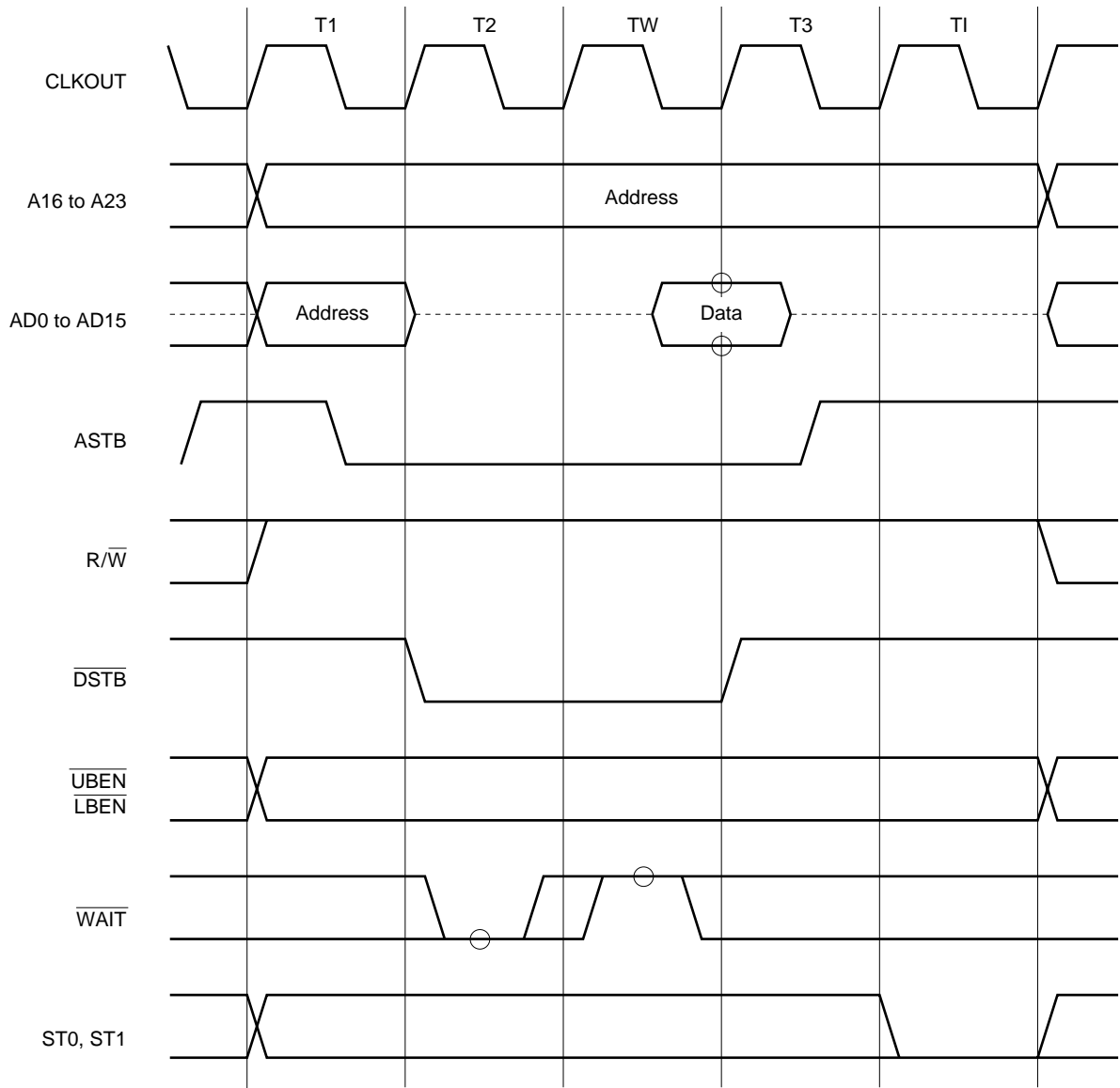
- Remarks**
1. ○ indicates the sampling timing when the number of programmable waits is set to 0.
 2. The dotted line indicates the high-impedance state.

(3) Memory read (0 wait, idle state)



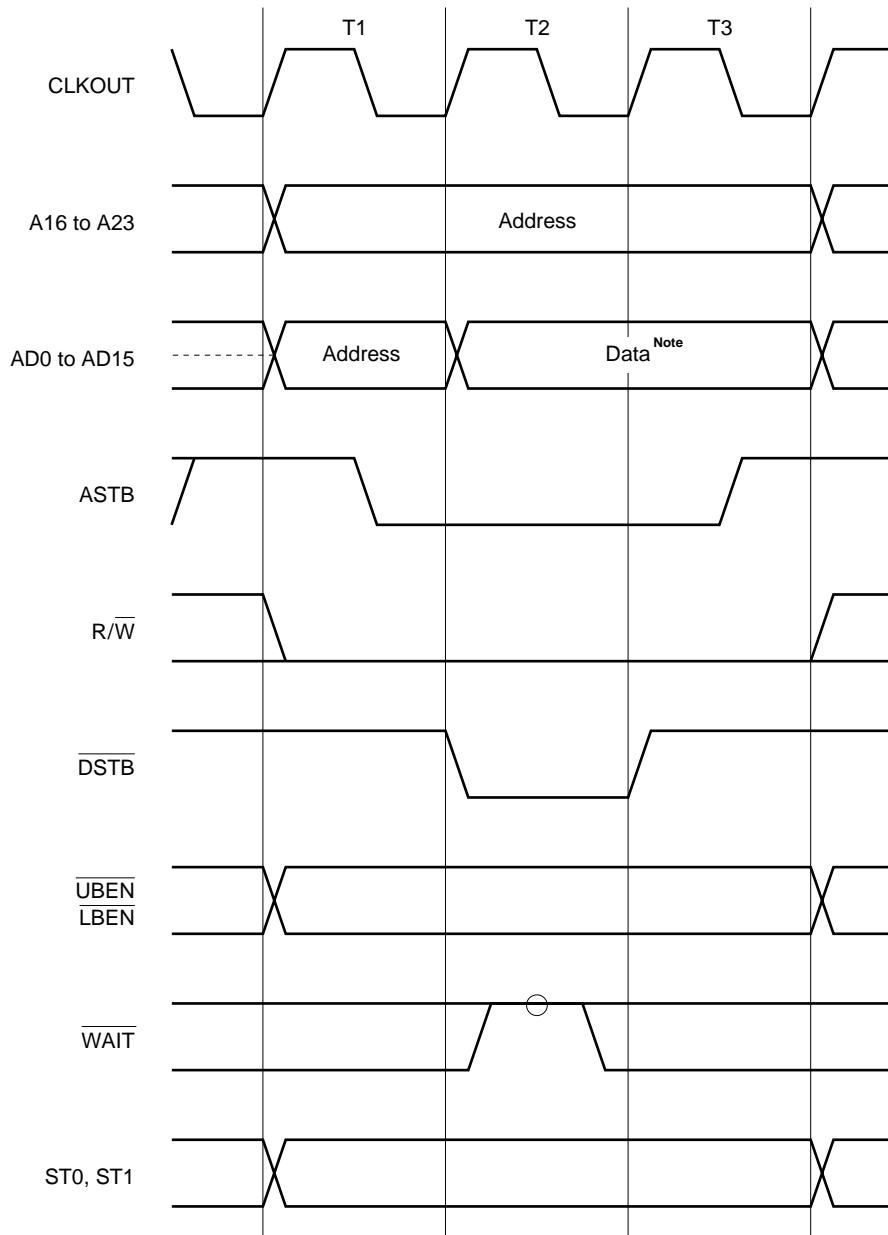
- Remarks**
1. ○ indicates the sampling timing when the number of programmable waits is set to 0.
 2. The dotted line indicates the high-impedance state.

(4) Memory read (1 wait, idle state)



- Remarks**
1. ○ indicates the sampling timing when the number of programmable waits is set to 0.
 2. The dotted line indicates the high-impedance state.

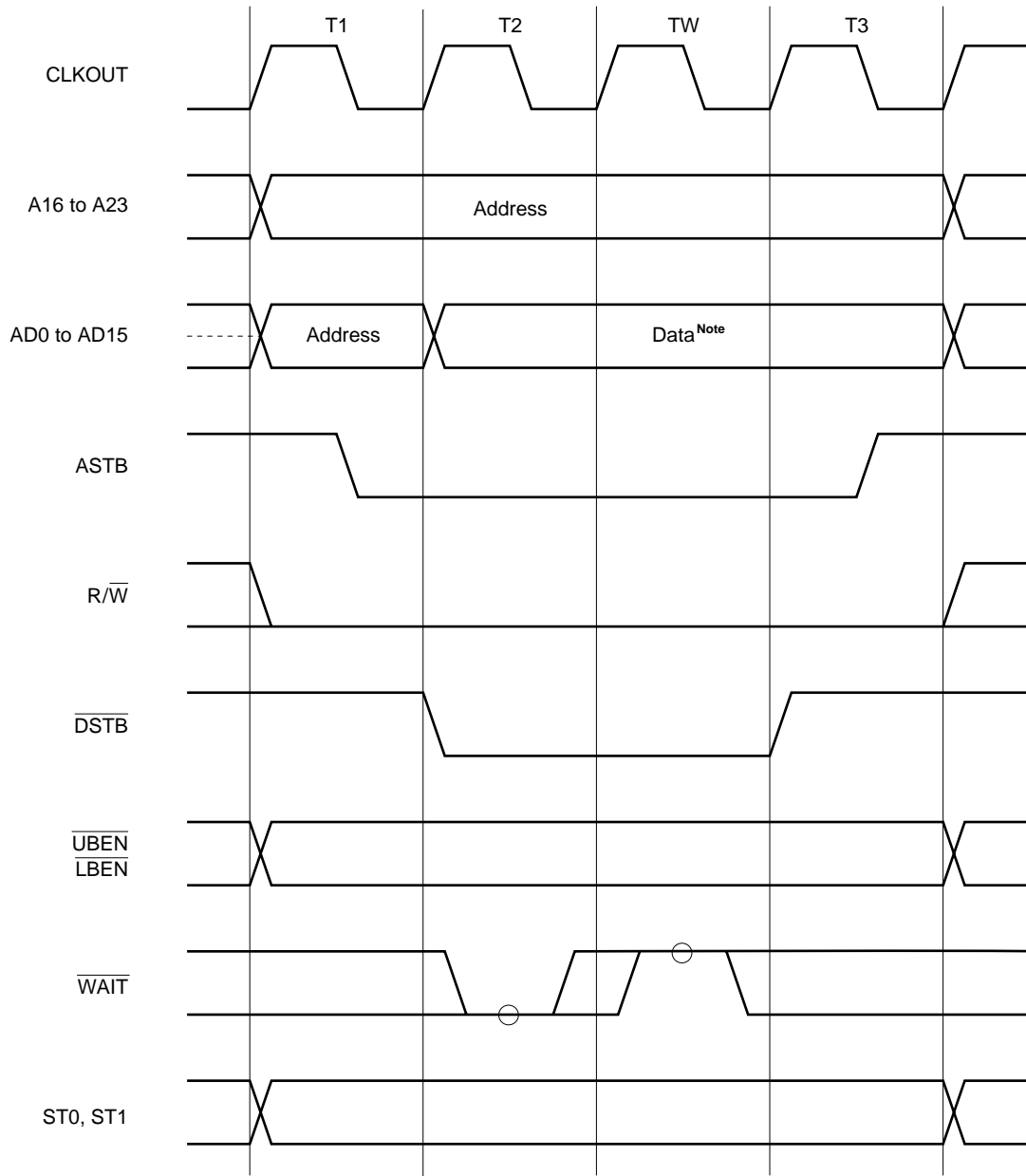
(5) Memory write (0 wait)



Note AD0 to AD7 output invalid data when odd address byte data is accessed.
AD8 to AD15 output invalid data when even address byte data is accessed.

Remarks 1. ○ indicates the sampling timing when the number of programmable waits is set to 0.
2. The dotted line indicates the high-impedance state.

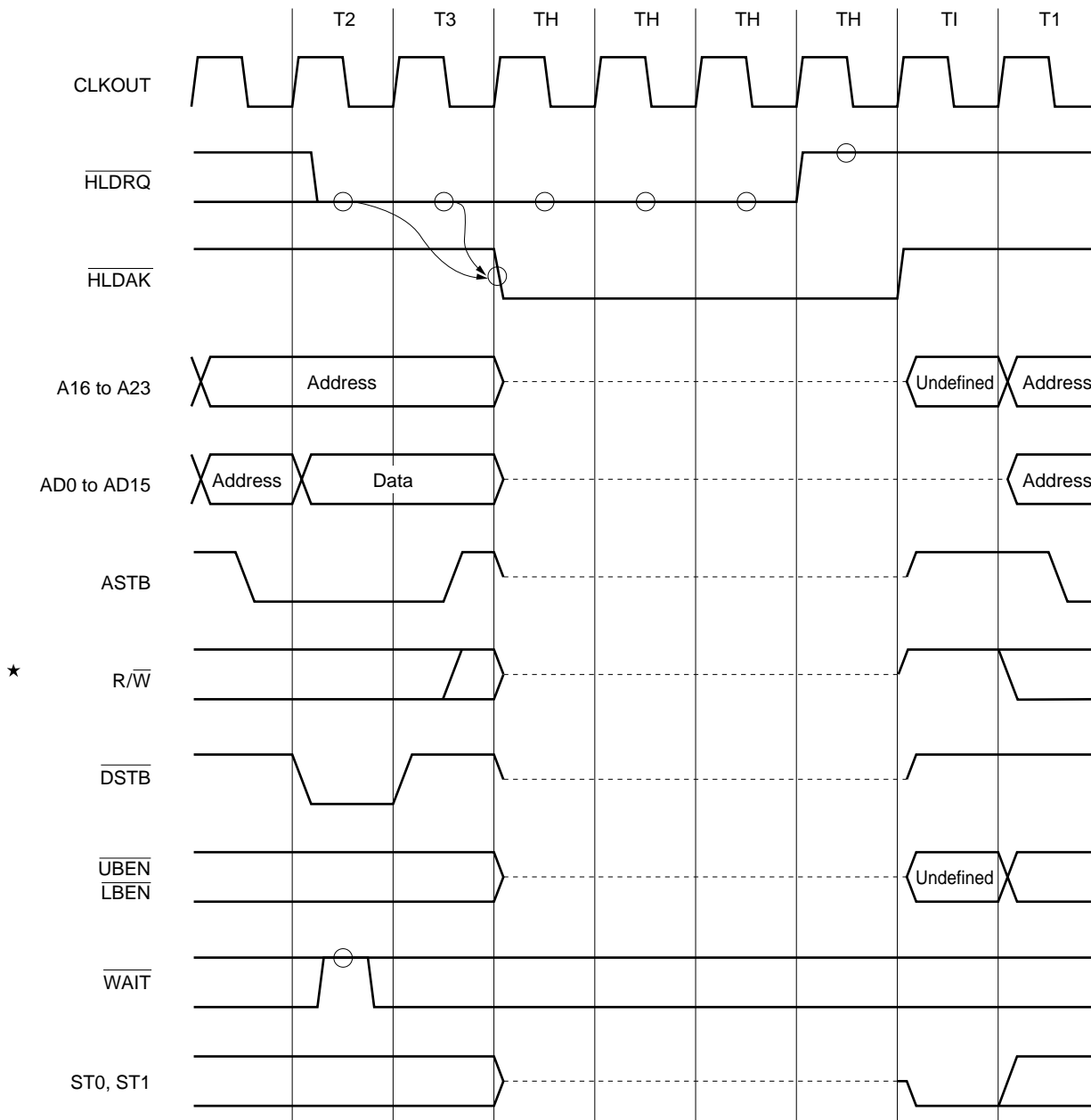
(6) Memory write (1 wait)



Note AD0 to AD7 output invalid data when odd address byte data is accessed.
 AD8 to AD15 output invalid data when even address byte data is accessed.

Remarks 1. ○ indicates the sampling timing when the number of programmable waits is set to 0.
 2. The dotted line indicates the high-impedance state.

(7) Bus hold timing



- Remarks**
1. ○ indicates the sampling timing.
 2. The dotted line indicates the high-impedance state.

★ **Caution** In case of transition to the bus hold status after a write cycle, a momentary high-level output from the $\overline{R/W}$ pin may occur just before the \overline{HLDAK} signal changes from high-level to low-level.

4.9 Bus Priority

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

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

The instruction fetch cycle and bus hold cycle are not inserted between the lower half-word access and higher half-word access of word operations.

Table 4-1. Bus Priority

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

4.10 Memory Boundary Operation Condition

4.10.1 Program space

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

4.10.2 Data space

Only the address aligned at the half-word (when the least significant bit of the address is "0")/word (when the lowest 2 bits of the address are "0") boundary is accessed for data half-word (16 bits)/word (32 bits) long.

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

- ★ The word access to the external memory is performed in the order of the lower half-word and then the higher half-word.

Refer to **V850 family User's Manual Architecture** for details.

4.11 Internal Peripheral I/O Interface

Access to the internal peripheral I/O area is not output to the external bus. Therefore, the internal peripheral I/O area can be accessed in parallel with instruction fetch access.

Accesses to the internal peripheral I/O area takes, in most cases, three clock cycles. However, accesses to the following timer/counter registers may take from 3 to 4 cycles.

| Peripheral I/O Register | Access |
|-------------------------|------------|
| TM1 | Read |
| TM4 | |
| CC10 | Read/write |
| CC11 | |
| CC12 | |
| CC13 | |
| CM4 | Write |

CHAPTER 5 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V852 is provided with a dedicated interrupt controller (INTC) for interrupt processing and can process a total of 17 interrupt requests.

An interrupt is an event that occurs independently of program execution, and an exception is an event that occurs dependently on program execution. Generally, an exception takes precedence over an interrupt.

The V852 can process interrupt requests from the internal peripheral hardware and external sources. Moreover, exception processing can be started (exception trap) by a TRAP instruction (software exception) or by generation of an exception event (fetching of an illegal op code).

5.1 Features

- Interrupt
 - Non-maskable interrupt: 1 source
 - Maskable interrupt: 16 sources
 - 8 levels programmable priorities
 - Multiple interrupt control according to priority
 - Each maskable interrupt can be individually disabled.
 - Noise elimination, edge detection, and rising and/or falling edge of external interrupt request signal can be specified.
- Exception
 - Software exception: 32 sources
 - Exception trap: 1 source (illegal op code exception)

These interrupt/exception sources are listed in Table 5-1.

Table 5-1. Interrupt List

| Type | Classification | Interrupt/Exception Source | | | | Default Priority | Exception Code | Vector Address | Restored PC |
|--------------------|----------------|----------------------------|--|--|-----------------|------------------|------------------------|----------------|-------------|
| | | Name | Control Register | Generating Source | Generating Unit | | | | |
| Reset | Interrupt | RESET | – | Reset input | – | – | 0000H | 00000000H | Undefined |
| Non-maskable | Interrupt | NMI | – | NMI input | – | – | 0010H | 00000010H | nextPC |
| Software exception | Exception | TRAP0n ^{Note} | – | TRAP instruction | – | – | 004n ^{Note} H | 00000040H | nextPC |
| | Exception | TRAP1n ^{Note} | – | TRAP instruction | – | – | 005n ^{Note} H | 00000050H | nextPC |
| Exception trap | Exception | ILGOP | – | Illegal op code | – | – | 0060H | 00000060H | nextPC |
| Maskable | Interrupt | INTOV1 | OVIC1 | Timer 1 overflow | RPU | 0 | 0080H | 00000080H | nextPC |
| | Interrupt | INTP10/INTCC10 | P1IC0 | INTP10 pin/CC10 coincidence | Pin/RPU | 1 | 0090H | 00000090H | nextPC |
| | Interrupt | INTP11/INTCC11 | P1IC1 | INTP11 pin/CC11 coincidence | Pin/RPU | 2 | 00A0H | 000000A0H | nextPC |
| | Interrupt | INTP12/INTCC12 | P1IC2 | INTP12 pin/CC12 coincidence | Pin/RPU | 3 | 00B0H | 000000B0H | nextPC |
| | Interrupt | INTP13/INTCC13 | P1IC3 | INTP13 pin/CC13 coincidence | Pin/RPU | 4 | 00C0H | 000000C0H | nextPC |
| | Interrupt | INTCM4 | CMIC4 | CM4 coincidence | RPU | 5 | 00D0H | 000000D0H | nextPC |
| | Interrupt | INTCSI0 | CSIC0 | CSI0 transmission/reception completion | SIO | 6 | 00E0H | 000000E0H | nextPC |
| | Interrupt | INTSER0 | SEIC0 | UART0 reception error | SIO | 7 | 00F0H | 000000F0H | nextPC |
| | Interrupt | INTSR0 | SRIC0 | UART0 reception completion | SIO | 8 | 0100H | 00000100H | nextPC |
| | Interrupt | INTST0 | STIC0 | UART0 transmission | SIO completion | 9 | 0110H | 00000110H | nextpc |
| | Interrupt | INTP00 | P0IC0 | INTP00 pin | Pin | 10 | 0120H | 00000120H | nextPC |
| | Interrupt | INTP01 | P0IC1 | INTP01 pin | Pin | 11 | 0130H | 00000130H | nextPC |
| | Interrupt | INTP02 | P0IC2 | INTP02 pin | Pin | 12 | 0140H | 00000140H | nextPC |
| | Interrupt | INTP03 | P0IC3 | INTP03 pin | Pin | 13 | 0150H | 00000150H | nextPC |
| | Interrupt | INTCSI1 | CSIC1 | CSI1 transmission/reception completion | SIO | 14 | 0160H | 00000160H | nextPC |
| Interrupt | INTCSI2 | CSIC2 | CSI2 transmission/reception completion | SIO | 15 | 0170H | 00000170H | nextPC | |

Note n: value of 0 to FH

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

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

2. The execution address of the illegal instruction when an illegal op code exception occurs is calculated as follows: (Restored PC–4)

5.2 Non-Maskable Interrupt

The non-maskable interrupt request is accepted unconditionally, even when interrupts are disabled (DI states) in the interrupt disabled (DI) status. The NMI is not subject to priority control and takes precedence over all the other interrupts.

The non-maskable interrupt request is input from the NMI pin. When the valid edge specified by the bit 0 (ESN0) of the external interrupt mode register 0 (INTM0) is detected on the NMI pin, the interrupt occurs.

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

5.2.1 Accepting operation

If the non-maskable interrupt is generated by NMI input, the CPU performs the following processing, and transfers control to the handler routine:

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

Figure 5-1 illustrates how the non-maskable interrupt is processed.

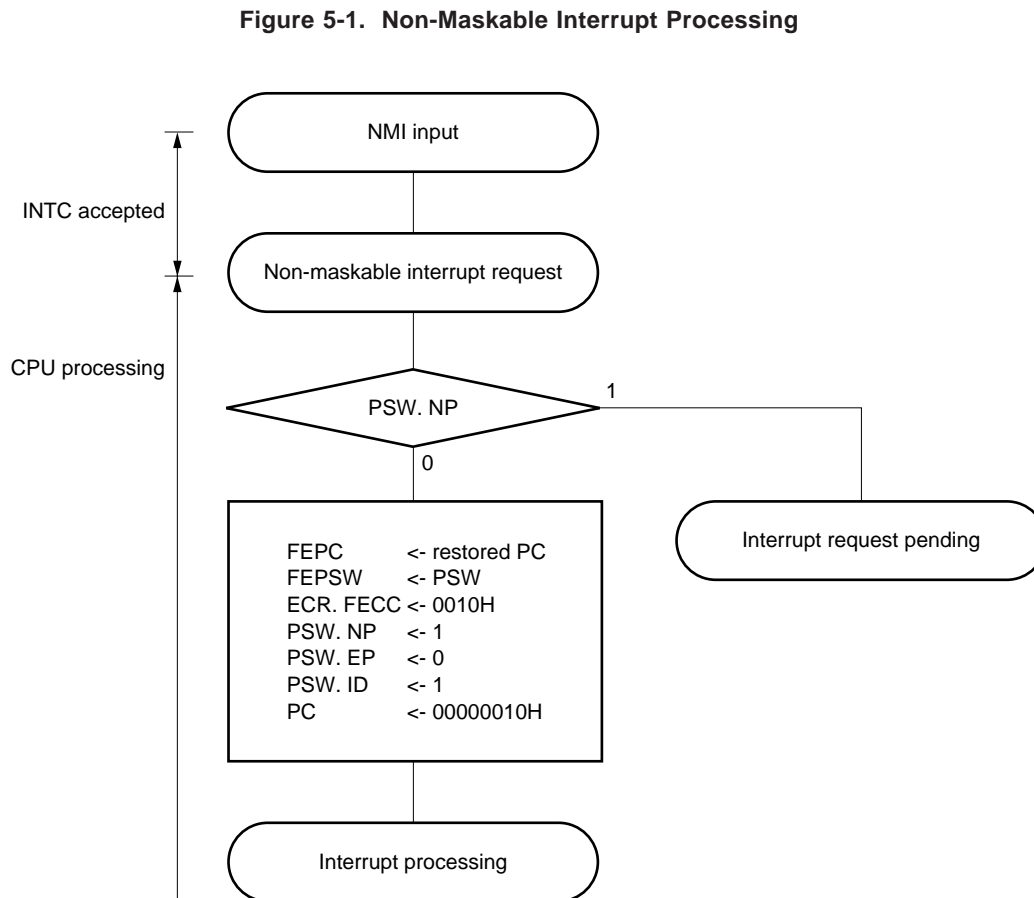
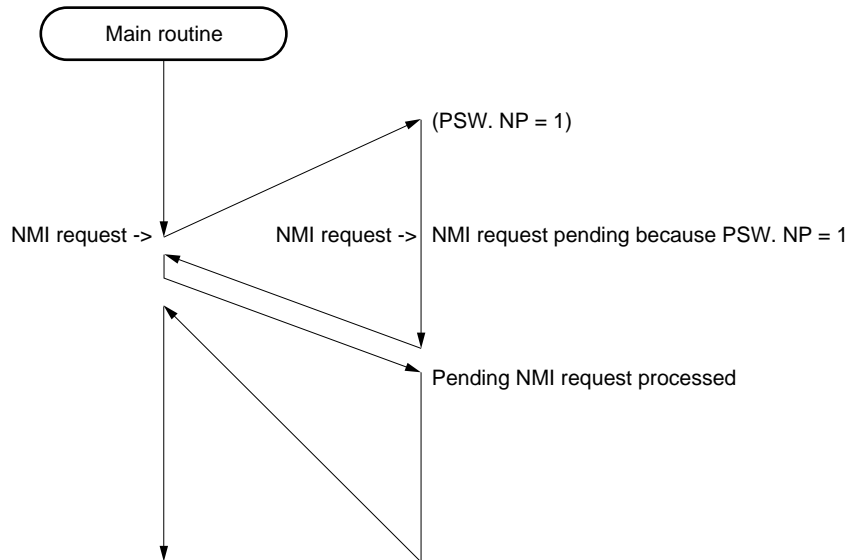
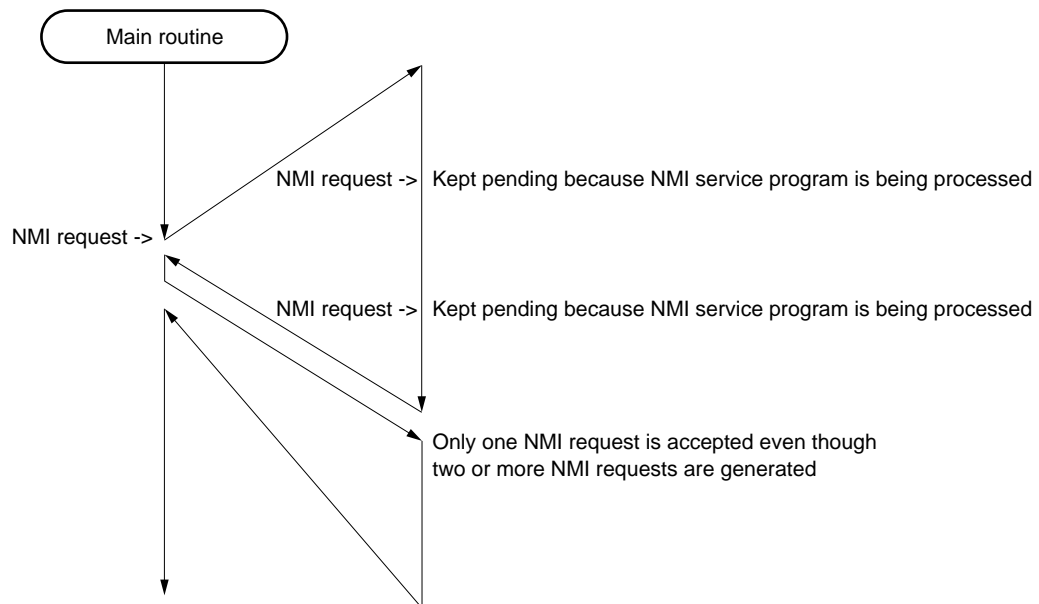


Figure 5-2. Accepting Non-Maskable Interrupt Request

(a) If a new NMI request is generated while an NMI service routine is executing:



(b) If a new NMI request is generated twice while an NMI service routine is executing:



5.2.2 Restore operation

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

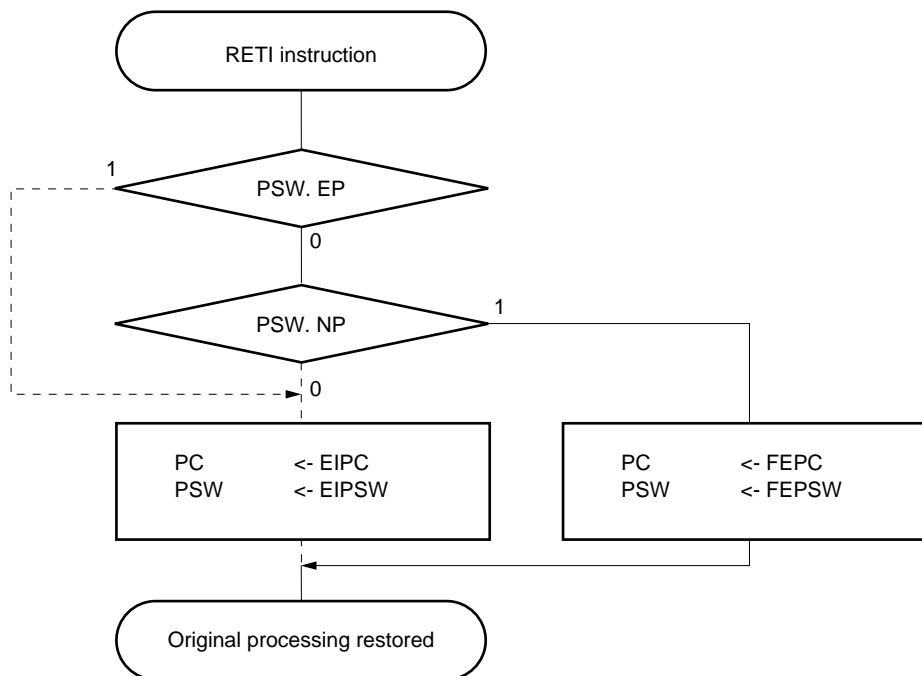
Operation of RETI instruction

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

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

Figure 5-3 illustrates how the RETI instruction is processed.

Figure 5-3. RETI Instruction Processing

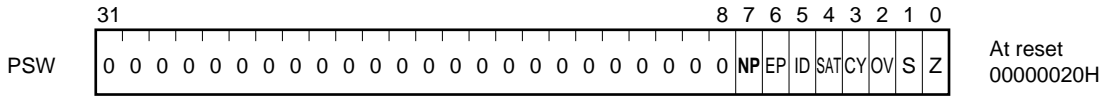


Caution When the PSW.EP bit and PSW.NP bit have been changed using the LDSR instruction during the non-maskable interrupt processing, to restore the values of PC and PSW when normal execution returns via the RETI instruction, it is necessary to set PSW.EP = 0 and PSW.NP = 1 immediately before the RETI instruction is executed using the LDSR instruction.

Remark The solid lines indicate the CPU processing.

5.2.3 NP flag

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



| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 7 | NP | NMI Pending Indicates that NMI interrupt processing is under execution 0: No NMI interrupt processing 1: NMI interrupt currently processing |

★ **5.2.4 Noise elimination for NMI pin**

Noise is eliminated from the NMI pin by analog delay.

The delay time is 60 to 220 ns. Input of a signal whose level changes within a time interval shorter than this delay time is not accepted internally.

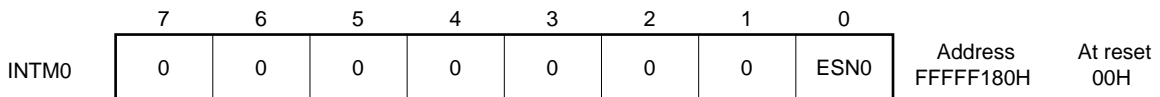
The NMI pin is used for releasing the software STOP mode.

Since the internal system clock stops in the software STOP mode, noise elimination using the system clock is not performed.

5.2.5 External interrupt mode register 0 (INTM0)

INTM0 is a register that specifies the valid edge of the non-maskable interrupt (NMI). The valid edge of NMI can be specified as the rising or falling edge by the ESN0 bit of this register.

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



| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 0 | ESN0 | Edge Select NMI Specifies valid edge of NMI pin 0: Falling edge 1: Rising edge |

5.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The V852 has 16 maskable interrupt sources.

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

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

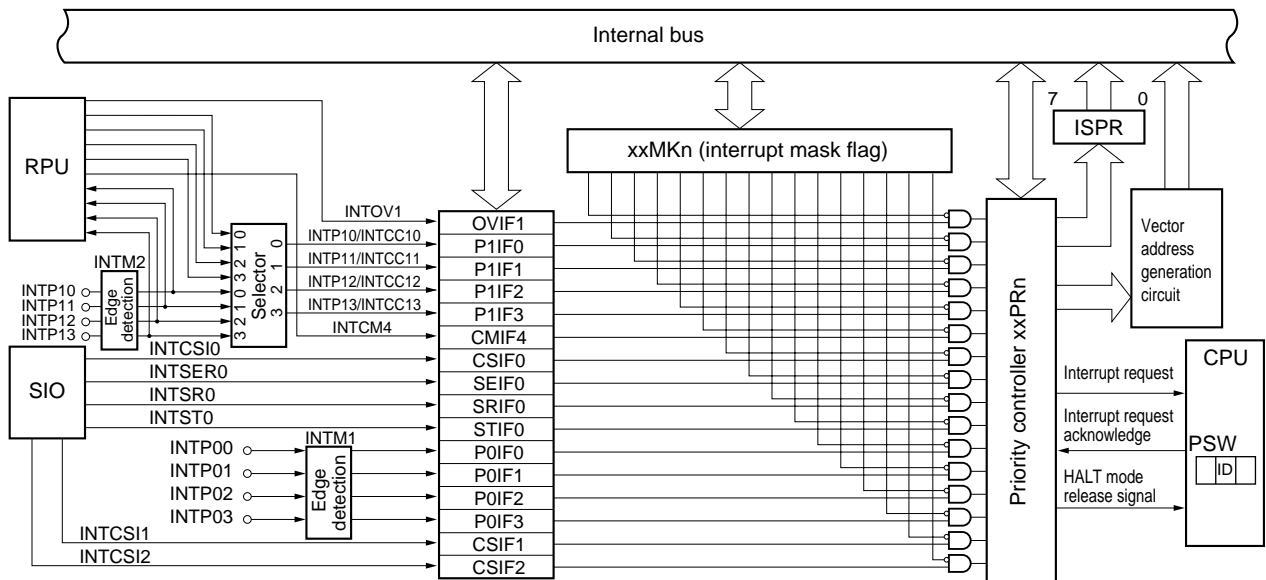
When the EI instruction is executed in an interrupt processing routine, the interrupt enabled (EI) status is set which enables interrupts having a higher priority to immediately interrupt the current service routine in progress. Note that only interrupts with a higher priority will have this capability;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

To perform multiple interrupts, the next preprocessings are necessary.

- <1> Save EIPC and EIPSW to memory or general-purpose register before executing the EI instruction
- <2> Execute the DI instruction before executing the RETI instruction, and return the value that saved in the process of <1> to the EIPC and EIPSW

5.3.1 Block diagram

Figure 5-4. Maskable Interrupt Block Diagram



xx: Identification name of each peripheral unit (OV, P1, CM, CS, SE, SR, ST, P0)
 n: Peripheral unit number (0 to 4)

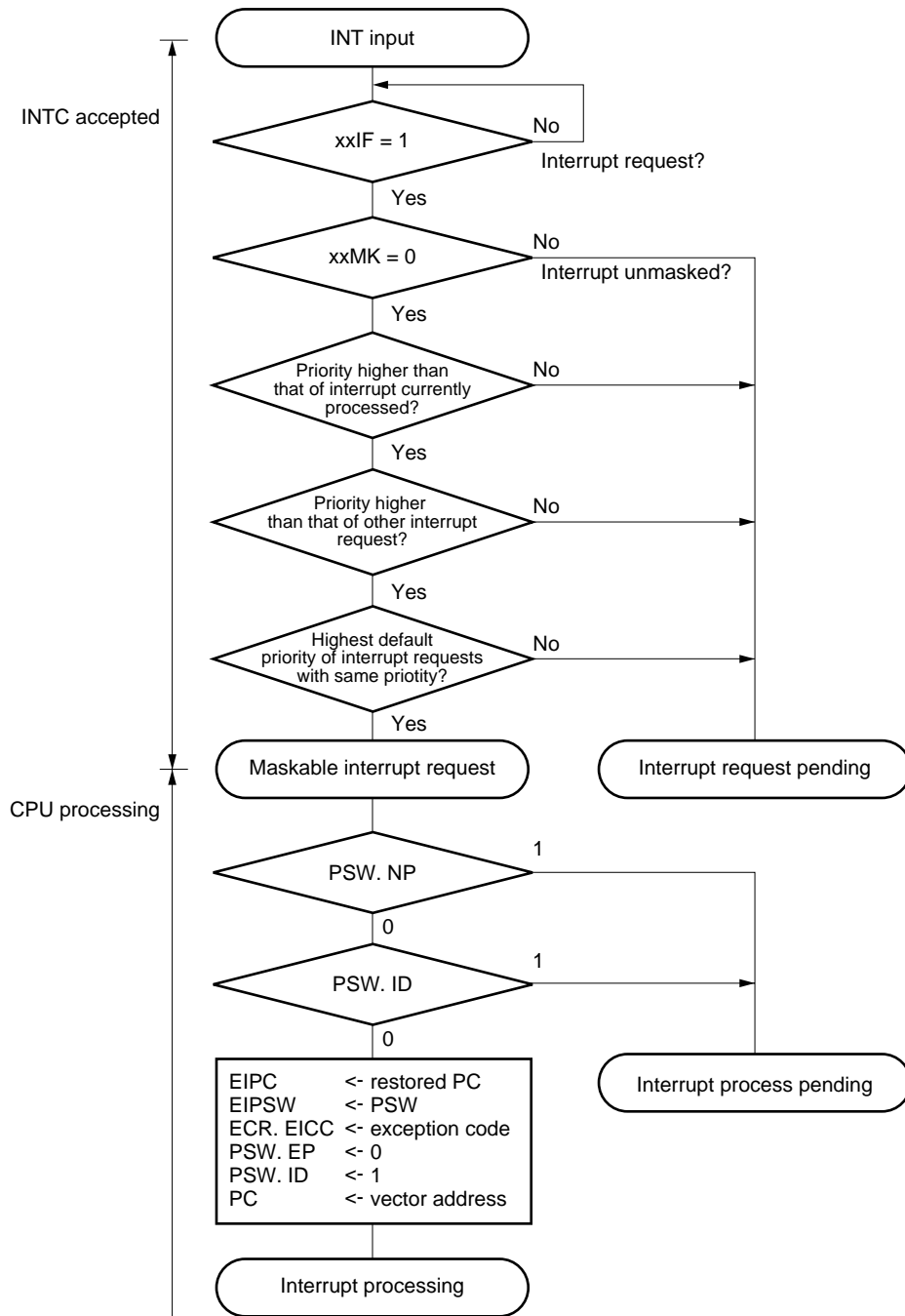
5.3.2 Operation

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

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

Figure 5-5 illustrates how the maskable interrupts are processed.

Figure 5-5. Maskable Interrupt Processing



The INT input masked by the interrupt control registers and that occurs while a previous interrupt is being processed (when PSW.NP = 1 or PSW.ID = 1) are internally monitored by the interrupt controller. When the interrupts are unmasked, or when PSW.NP = 0 and PSW.ID = 0 by using the RETI and LDSR instructions, the new maskable interrupts can then be acknowledged, by the pending INT input, and processed.

5.3.3 Restore

To restore or return execution from the maskable interrupt service routine, the RETI instruction is used.

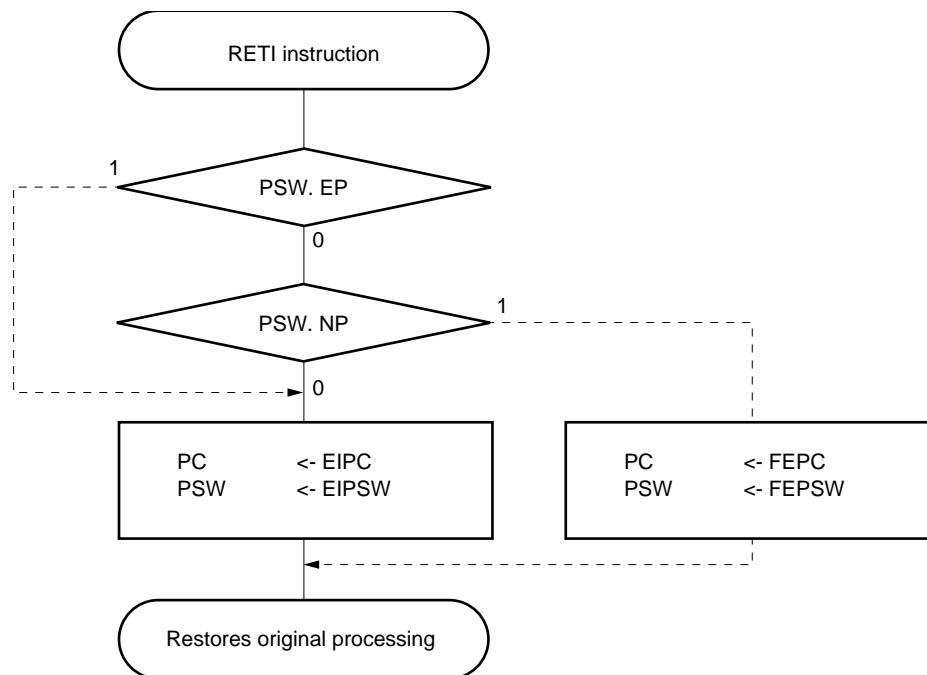
Operation of RETI instruction

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

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

Figure 5-6 illustrates the processing of the RETI instruction.

Figure 5-6. RETI Instruction Processing



Caution When the PSW.EP bit and PSW.NP bit have been changed using the LDSR instruction during the maskable interrupt processing, to restore the values of PC and PSW when normal execution returns via the RETI instruction, it is necessary to return each value of PSW.EP and PSW.NP to 0 immediately before the RETI instruction is executed using the LDSR instruction.

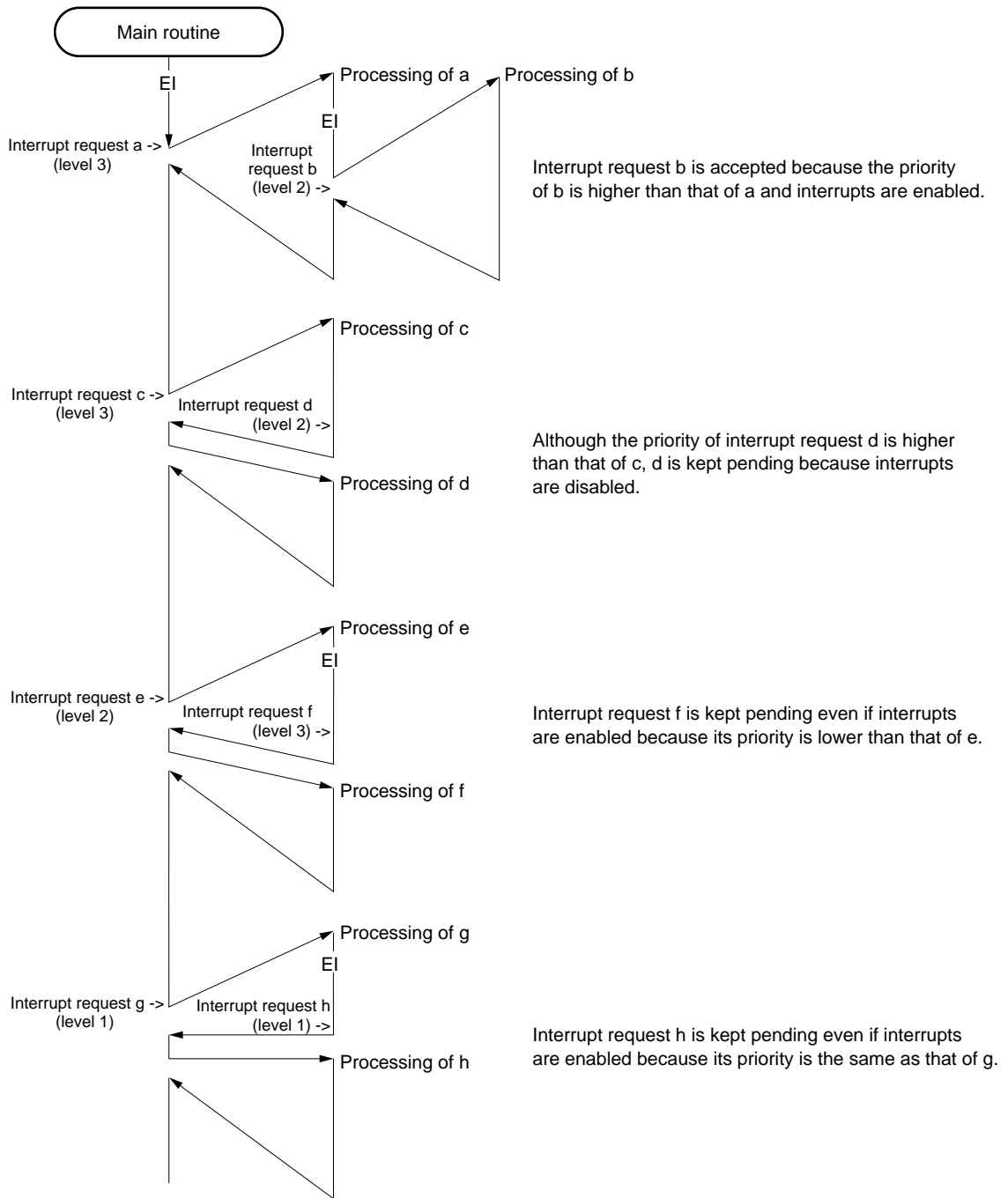
Remark The solid lines indicate the CPU processing.

5.3.4 Priorities of maskable interrupts

The V852 processes multiple interrupts which accept another interrupt during interrupt processing. Multiple interrupts can be controlled by the priority level. There are two priority control criteria in the V852: control based on the default priority levels, and programmable priority control based on interrupt priority specification bit (xxPRn). The priority level control by the default priority levels performs interrupt processing according to the priority level (default priority level) preassigned to each interrupt request when several interrupts of the same priority by xxPRn occur at the same time (Refer to **Table 5-1. List of Interrupts**). The programmable priority level control customizes the interrupt requests into eight levels according to the setting of the priority level specification flag.

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

Figure 5-7. Example of Interrupt Nesting Process (1/2)



- Remarks**
1. a to u in the figure are the names of interrupt requests shown for the sake of explanation.
 2. The default priority in the figure indicates the relative priority between two interrupt requests.

Caution The values of EIPC and EIPSW must be saved before executing multiple interrupts.

Figure 5-7. Example of Interrupt Nesting Process (2/2)

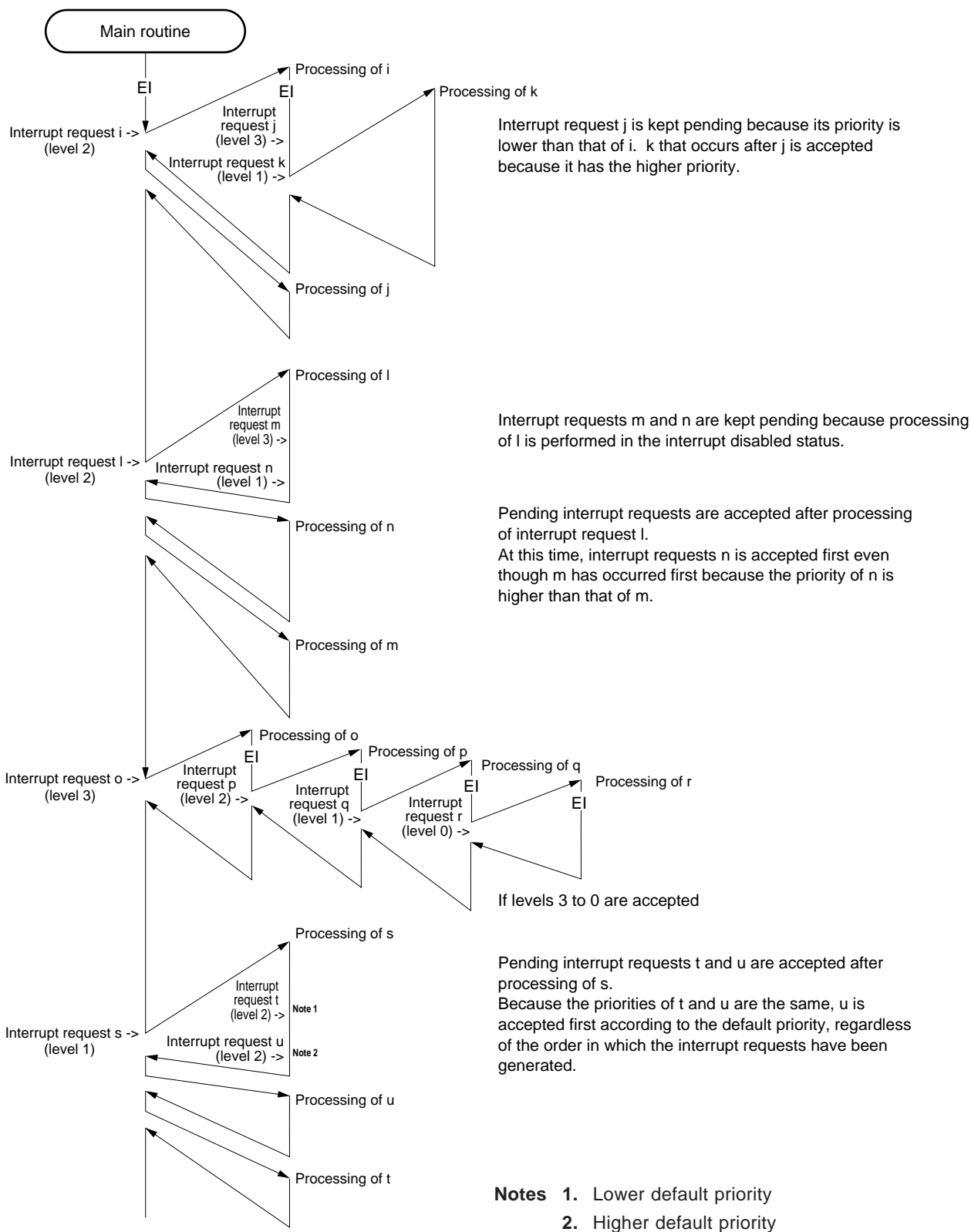
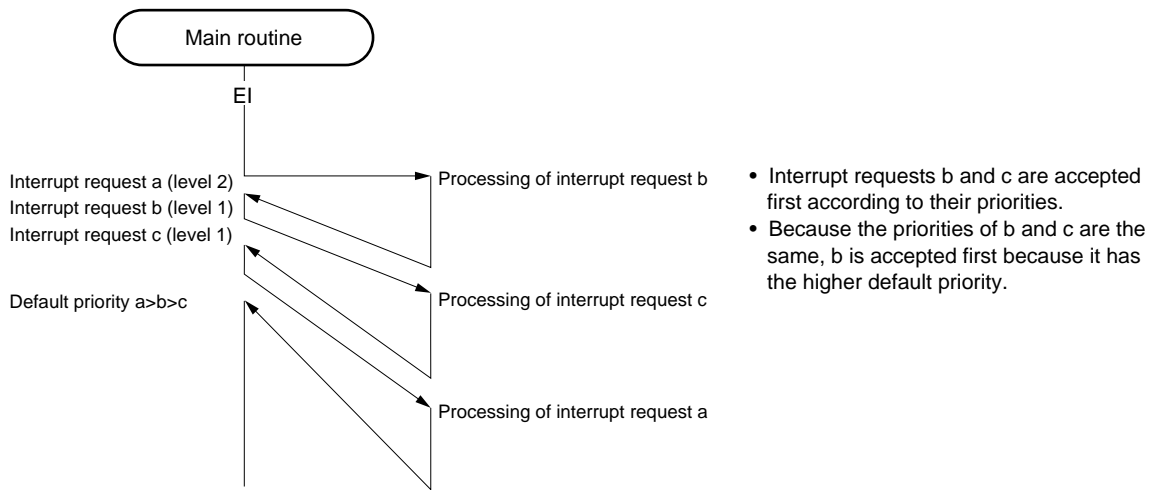
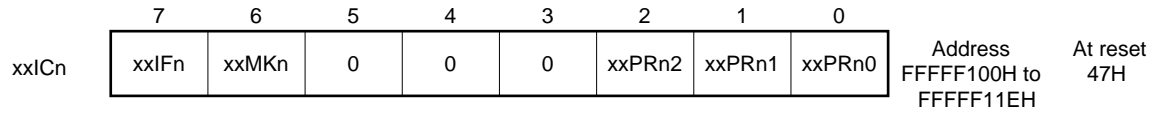


Figure 5-8. Example of Processing Interrupt Requests Simultaneously Generated

5.3.5 Interrupt control register (xxICn)

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

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



| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|------------------|---|--------------------------------------|--------|--------|--------------------------------------|---|---|---|-----------------------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|----------------------------|
| 7 | xxIFn | Interrupt Request Flag Interrupt request flag 0: Interrupt request not issued 1: Interrupt request issued xxIFn flag is automatically reset by hardware when interrupt request is accepted. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | xxMKn | Mask Flag Interrupt mask flag 0: Enables interrupt processing 1: Disables interrupt processing (pending) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 to 0 | xxPRn2 to xxPRn0 | Priority Specifies eight levels of priorities for each interrupt <table border="1"> <thead> <tr> <th>xxPRn2</th> <th>xxPRn1</th> <th>xxPRn0</th> <th>Interrupt priority specification bit</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Specifies level 0 (highest)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Specifies level 1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Specifies level 2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Specifies level 3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Specifies level 4</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Specifies level 5</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>Specifies level 6</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>Specifies level 7 (lowest)</td> </tr> </tbody> </table> | 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) |
| 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) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Remark xx: identification name of each peripheral unit (OV, P1, CM, CS, SE, SR, ST, P0)
n: peripheral unit number (0 to 4)

The address and bit of each interrupt control register is as follows.

Table 5-2. Addresses and Bits of Interrupt Control Register

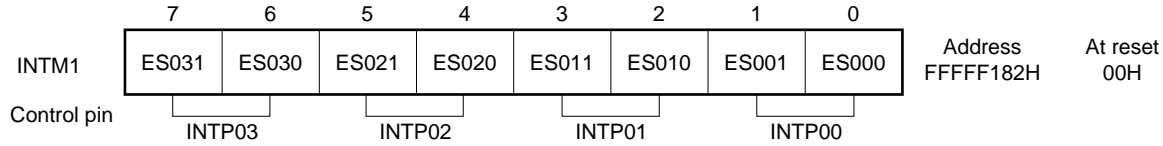
| Address | Register | Bit | | | | | | | |
|------------|----------|-------|-------|---|---|---|--------|--------|--------|
| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FFFFFF100H | OVIC1 | OVIF1 | OVMK1 | 0 | 0 | 0 | OVPR12 | OVPR11 | OVPR10 |
| FFFFFF102H | P1IC0 | P1IF0 | P1MK0 | 0 | 0 | 0 | P1PR02 | P1PR01 | P1PR00 |
| FFFFFF104H | P1IC1 | P1IF1 | P1MK1 | 0 | 0 | 0 | P1PR12 | P1PR11 | P1PR10 |
| FFFFFF106H | P1IC2 | P1IF2 | P1MK2 | 0 | 0 | 0 | P1PR22 | P1PR21 | P1PR20 |
| FFFFFF108H | P1IC3 | P1IF3 | P1MK3 | 0 | 0 | 0 | P1PR32 | P1PR31 | P1PR30 |
| FFFFFF10AH | CMIC4 | CMIF4 | CMMK4 | 0 | 0 | 0 | CMPR42 | CMPR41 | CMPR40 |
| FFFFFF10CH | CSIC0 | CSIF0 | CSMK0 | 0 | 0 | 0 | CSPR02 | CSPR01 | CSPR00 |
| FFFFFF10EH | SEIC0 | SEIF0 | SEMK0 | 0 | 0 | 0 | SEPR02 | SEPR01 | SEPR00 |
| FFFFFF110H | SRIC0 | SRIF0 | SRMK0 | 0 | 0 | 0 | SRPR02 | SRPR01 | SRPR00 |
| FFFFFF112H | STIC0 | STIF0 | STMK0 | 0 | 0 | 0 | STPR02 | STPR01 | STPR00 |
| FFFFFF114H | P0IC0 | P0IF0 | P0MK0 | 0 | 0 | 0 | P0PR02 | P0PR01 | P0PR00 |
| FFFFFF116H | P0IC1 | P0IF1 | P0MK1 | 0 | 0 | 0 | P0PR12 | P0PR11 | P0PR10 |
| FFFFFF118H | P0IC2 | P0IF2 | P0MK2 | 0 | 0 | 0 | P0PR22 | P0PR21 | P0PR20 |
| FFFFFF11AH | P0IC3 | P0IF3 | P0MK3 | 0 | 0 | 0 | P0PR32 | P0PR31 | P0PR30 |
| FFFFFF11CH | CSIC1 | CSIF1 | CSMK1 | 0 | 0 | 0 | CSPR12 | CSPR11 | CSPR10 |
| FFFFFF11EH | CSIC2 | CSIF2 | CSMK2 | 0 | 0 | 0 | CSPR22 | CSPR21 | CSPR20 |

5.3.6 External interrupt mode registers 1 and 2 (INTM1 and INTM2)

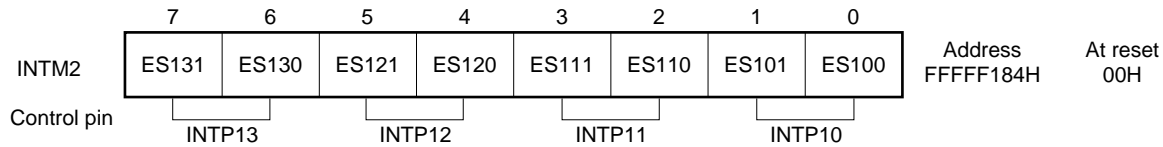
These registers specify the valid edges of external interrupt requests INTP00 to INTP03 and INTP10 to INTP13 that are input from external pins. INTM1 controls INTP00 to INTP03, and INTM2 controls INTP10 to INTP13.

The valid edge of each pin can be specified to be the rising, falling, and both rising and falling edges.

Both the registers can be read/written in 8- or 1-bit units.



| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|-----------------------|---|-------|-------|-----------|---|---|--------------|---|---|-------------|---|---|----------------|---|---|-------------------------------|
| 7, 5, 3, 1 | ES0n1 | Edge Select Specifies valid edge of INTP0n pin | | | | | | | | | | | | | | | |
| 6, 4, 2, 0 | ES0n0 (n = 3 to 0) | | | | | | | | | | | | | | | | |
| | | <table border="1"> <thead> <tr> <th>ES0n1</th> <th>ES0n0</th> <th>Operation</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table> | ES0n1 | ES0n0 | Operation | 0 | 0 | Falling edge | 0 | 1 | Rising edge | 1 | 0 | RFU (reserved) | 1 | 1 | Both rising and falling edges |
| ES0n1 | ES0n0 | Operation | | | | | | | | | | | | | | | |
| 0 | 0 | Falling edge | | | | | | | | | | | | | | | |
| 0 | 1 | Rising edge | | | | | | | | | | | | | | | |
| 1 | 0 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 1 | Both rising and falling edges | | | | | | | | | | | | | | | |



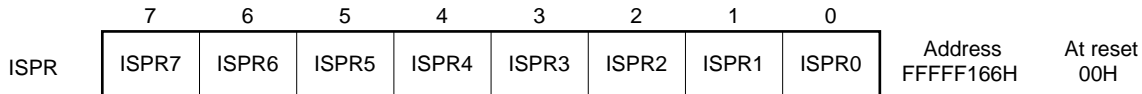
| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|-----------------------|---|-------|-------|-----------|---|---|--------------|---|---|-------------|---|---|----------------|---|---|-------------------------------|
| 7, 5, 3, 1 | ES1n1 | Edge Select Specifies valid edge of INTP1n pin | | | | | | | | | | | | | | | |
| 6, 4, 2, 0 | ES1n0 (n = 3 to 0) | | | | | | | | | | | | | | | | |
| | | <table border="1"> <thead> <tr> <th>ES1n1</th> <th>ES1n0</th> <th>Operation</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table> | ES1n1 | ES1n0 | Operation | 0 | 0 | Falling edge | 0 | 1 | Rising edge | 1 | 0 | RFU (reserved) | 1 | 1 | Both rising and falling edges |
| ES1n1 | ES1n0 | Operation | | | | | | | | | | | | | | | |
| 0 | 0 | Falling edge | | | | | | | | | | | | | | | |
| 0 | 1 | Rising edge | | | | | | | | | | | | | | | |
| 1 | 0 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 1 | Both rising and falling edges | | | | | | | | | | | | | | | |

5.3.7 In-service priority register (ISPR)

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

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

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

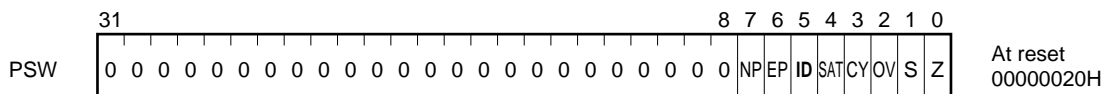


| Bit Position | Bit Name | Function |
|--------------|----------------|--|
| 7 to 0 | ISPR7 to ISPR0 | In-Service Priority Flag Indicates priority of interrupt currently accepted 0: Interrupt request with priority n not accepted 1: Interrupt request with priority n accepted |

Remark n: 0 to 7 (priority level)

5.3.8 Maskable interrupt status flag

The interrupt disable status flag (ID) of the PSW controls the enabling and disabling of maskable interrupt requests. As a status flag, it also displays the current maskable interrupt acceptance condition.



| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 5 | ID | Interrupt Disable Indicates enabling or disabling maskable interrupt processing. 0: Maskable interrupt accepting enabled 1: Maskable interrupt accepting disabled (pending) It is set to 1 by the DI instruction and reset to 0 by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW. Non-maskable interrupt and exceptions are acknowledged regardless of this flag. When a maskable interrupt is accepted, ID flag is automatically set to 1 by hardware. Interrupt requests generated during accept disabled (ID = 1) can be accepted if the $\times\times$ IFn bit of the $\times\times$ ICn register is set (1) and the ID flag is reset (0). |

★

5.4 Software Exception

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

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

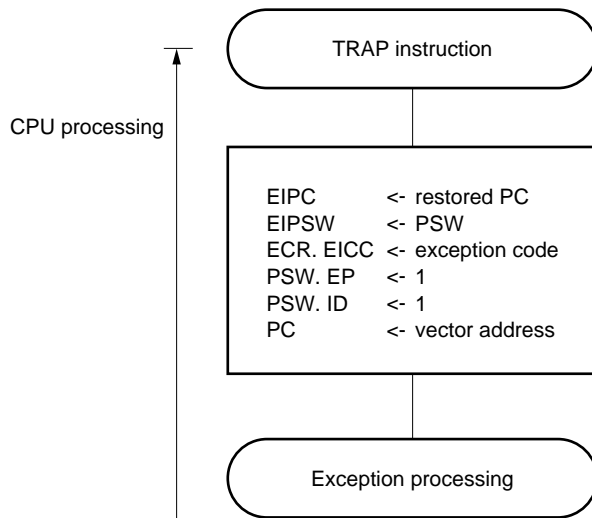
5.4.1 Operation

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

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

Figure 5-9 illustrates how the software exception is processed.

Figure 5-9. Software Exception Processing



The vector address is determined by the operand of the TRAP instruction. If the operand is 0 to 0FH, the vector address is 00000040H; if the operand is 10H to 1FH, it is 00000050H.

5.4.2 Restore

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

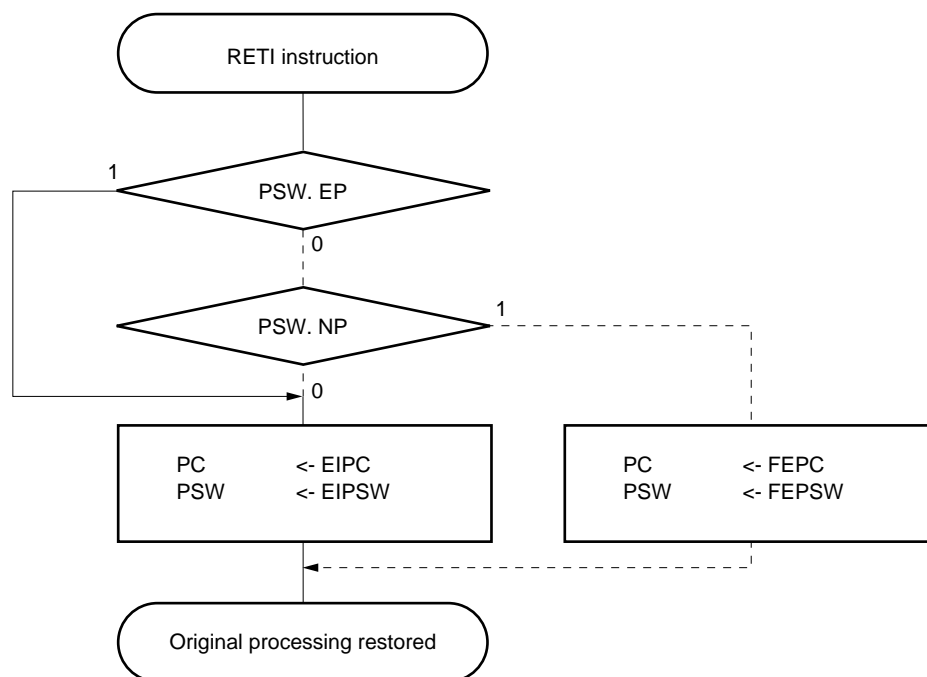
Operation of RETI instruction

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

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

Figure 5-10 illustrates the processing of the RETI instruction.

Figure 5-10. RETI Instruction Processing

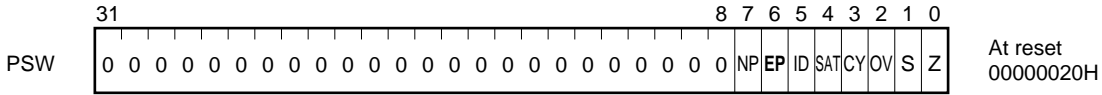


Caution When the PSW.EP bit and PSW.NP bit have been changed using the LDSR instruction during the software exception interrupt processing, to restore the values of PC and PSW when normal execution returns via the RETI instruction, it is necessary to return the value of PSW.EP to 1 immediately before the RETI instruction is executed using the LDSR instruction.

Remark The solid lines indicate the CPU processing.

5.4.3 EP flag

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



| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 6 | EP | Exception Pending Indicates that trap processing is in progress 0: Exception processing is not in progress 1: Exception processing is in progress |

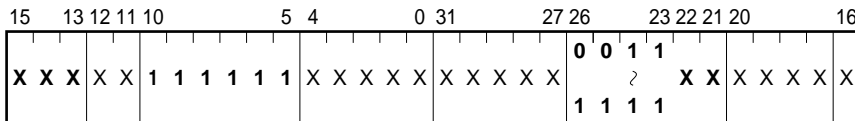
5.5 Exception Trap

The exception trap is an interrupt that is requested when illegal execution of an instruction takes place. In the V852, an illegal op code exception (ILGOP: ILLeGal OPcode trap) is considered as an exception trap.

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

5.5.1 Illegal op code definition

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



x : don't care

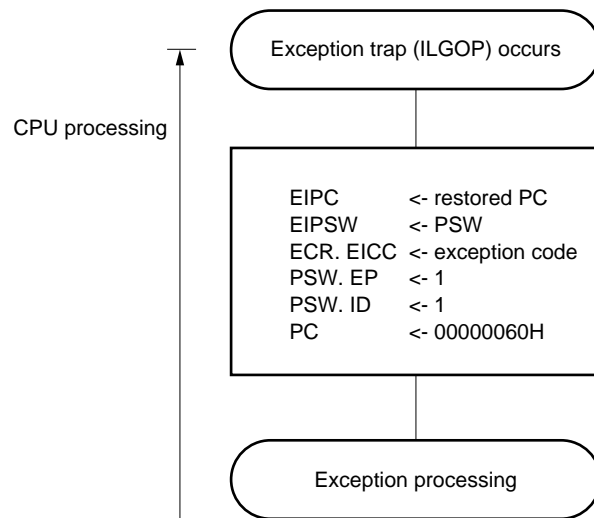
5.5.2 Operation

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

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

Figure 5-11 illustrates how the exception trap is processed.

Figure 5-11. Exception Trap Processing



5.5.3 Restore

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

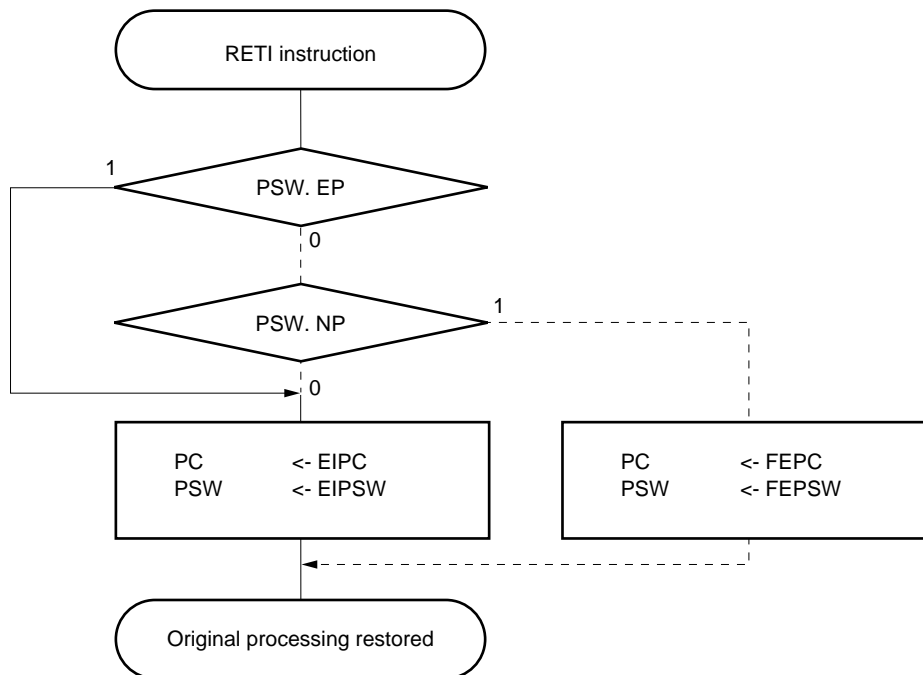
Operation of RETI instruction

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

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

Figure 5-12 illustrates the processing of the RETI instruction.

Figure 5-12. RETI Instruction Processing



Caution When the PSW.EP bit and PSW.NP bit have been changed using the LDSR instruction during the exception trap interrupt processing, to restore the values of PC and PSW when normal execution returns via the RETI instruction, it is necessary to return the value of PSW.EP to 1 immediately before the RETI instruction is executed using the LDSR instruction.

Remark The solid lines indicate the CPU processing.

5.6 Priority Control

5.6.1 Priorities of interrupts and exceptions

| | RESET | NMI | INT | TRAP | ILGOP |
|-------|-------|-----|-----|------|-------|
| RESET | | * | * | * | * |
| NMI | x | | <- | <- | <- |
| INT | x | ↑ | | <- | <- |
| TRAP | x | ↑ | ↑ | | <- |
| ILGOP | x | ↑ | ↑ | ↑ | |

RESET : reset

NMI : non-maskable interrupt

INT : maskable interrupt

TRAP : software exception

ILGOP : illegal code exception

* : Item on the left ignores the item above.

x : Item on the left is ignored by the item above.

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

<- : Item on the left is higher than the item above in priority.

5.6.2 Multiple interrupt processing

Multiple interrupt processing is a function which allows the nesting of interrupts. If a higher priority interrupt is generated and accepted, it will be allowed to stop a current interrupt service routine in progress.

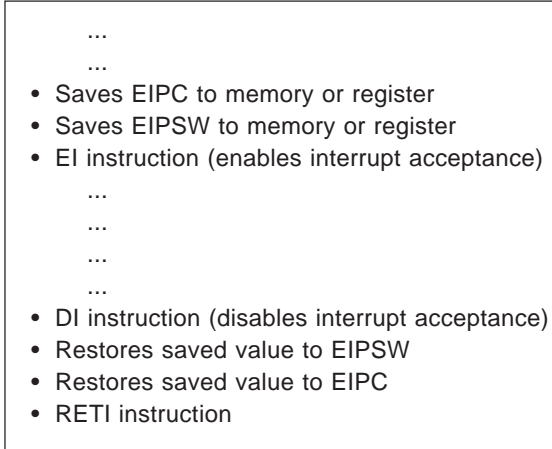
If an interrupt with a lower or equal priority is generated and a service routine is currently in progress, the later interrupt will be kept pending.

Multiple interrupt processing control is performed while an interrupt service routine is currently in progress and the interrupts are kept enabled (ID = 0). If a maskable interrupt or exception is generated and accepted while a prior interrupt routine is under progress, the higher priority interrupting routine must save the current contents of EIPC and EIPSW to allow proper restoration when the routine ends.

Programming examples used for interrupt nesting are shown in the following code fragments:

(1) To accept maskable interrupts in service routine

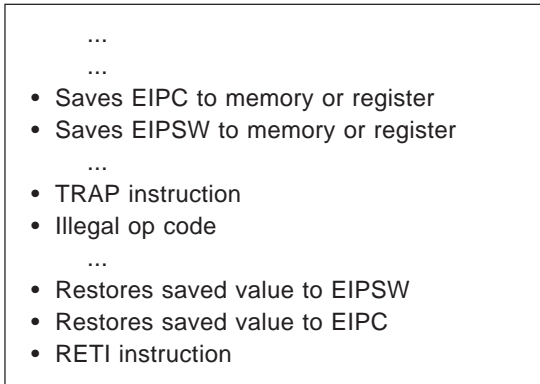
Service routine of maskable interrupt or exception



<- Accepts interrupt such as INTp input

(2) To generate exception in service program

Service program of maskable interrupt or exception



<- Accepts exception such as TRAP instruction

<- Accepts exception such as illegal op code

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

Priorities of maskable interrupts

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

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

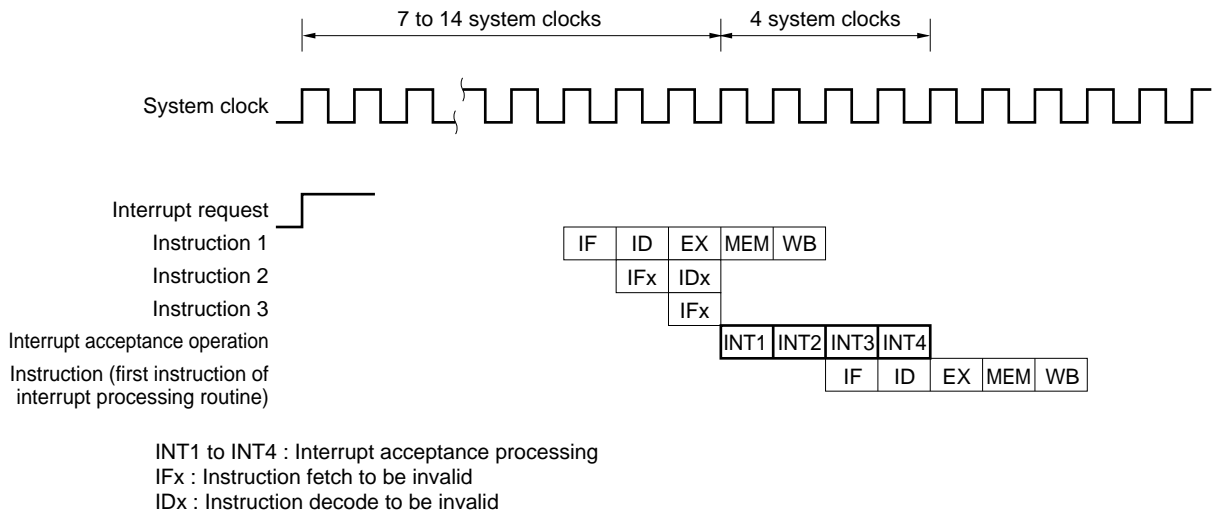
A pending interrupt request is accepted after the current interrupt processing has been completed and the RETI instruction has been executed.

Caution The maskable interrupts are not accepted but pended in non-maskable interrupt processing routine (time until the RETI instruction is executed).

5.7 Interrupt Latency Time

The interrupt latency time is defined as the time measured between the generation of the interrupt request and the execution of the first instruction in the corresponding interrupt service routine. The following describes the interrupt latency time.

★ **Figure 5-13. Pipeline Operation upon Reception of Interrupt Request (Outline)**



| Interrupt Latency Time (System clock) | Condition | | |
|---------------------------------------|--------------------|--------------------|---|
| | Internal Interrupt | External Interrupt | |
| Minimum | 11 | 13 | Except when: <ul style="list-style-type: none"> • In IDLE/STOP mode • External bus is accessed • Two or more interrupt request non-sample instructions are executed in succession • Accessed interrupt control register |
| Maximum | 18 | 20 | |

5.8 Periods Where Interrupt Is Not Acknowledged

An interrupt is acknowledged while an instruction is being executed. However, no interrupt is acknowledged between interrupt request non-sample instruction and next instruction.

Interrupt request non-sample instruction

- EI instruction
- DI instruction
- LDSR reg2, 0 x 5 instruction (vs. PSW)

[MEMO]

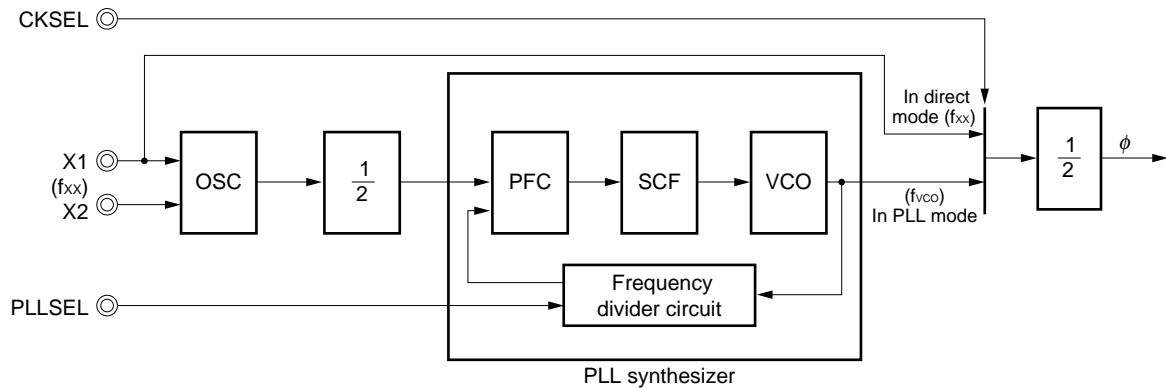
CHAPTER 6 CLOCK GENERATION FUNCTION

The clock generator produces and controls the internal system clock (ϕ) which is supplied to all the internal hardware units including the CPU.

6.1 Features

- Multiplication function by PLL (Phase Locked Loop) synthesizer
- Clock source
 - Oscillation through oscillator connection: $f_{xx} = \frac{1}{1} \times \phi$,
 $f_{xx} = \frac{1}{5} \times \phi$
 - External clock (PLL mode): $f_{xx} = \frac{1}{1} \times \phi$,
 $f_{xx} = \frac{1}{5} \times \phi$
 - External clock (direct mode): $f_{xx} = 2 \times \phi$
- Power save mode
 - HALT mode
 - IDLE mode
 - Software STOP mode
- Clock output inhibit function

6.2 Configuration



- f_{vco} : VCO oscillation frequency (= $2 \cdot f_{xx}$: PLLSEL = 0), (= $10 \cdot f_{xx}$: at PLLSEL = 1)
- ϕ : internal system clock frequency (= $1/2 \cdot f_{vco}$: in PLL mode)
internal system clock frequency (= $1/2 \cdot f_{xx}$: in direct mode)
- OSC : oscillator (PLL mode only)
- PFC : phase frequency comparator
- SCF : switched capacitor filter
- VCO : voltage-controlled oscillator
- Frequency divider circuit (=1/4: PLLSEL = 0), (=1/20: PLLSEL = 1)

6.3 Selecting Input Clock

The clock generator consists of a clock oscillator and a PLL synthesizer. It can generate, for example, at PLLSEL = 1 a 25-MHz system clock when a 5-MHz crystal resonator or ceramic resonator is connected across the X1 and X2 pins.

An external clock can be directly connected to the oscillator circuit. In this case, input the clock signal to the X1 pin, and leave the X2 pin open.

The clock generator has two operation modes: PLL and direct modes, which are selected by the CKSEL pin, as shown in the table below.

| CKSEL | Operation mode |
|-------|----------------|
| 0 | PLL mode |
| 1 | Direct mode |

Caution The CKSEL pin level should never be changed during operation. The V852 may not operate correctly.

6.3.1 Direct mode

In the direct mode, an external clock with a frequency two times higher than that of the system clock is input. Because OSC and PLL synthesizer do not operate, the power dissipation can be significantly reduced. This mode is used mainly in applications where the V852 must operate on a relatively low frequency. To minimize the influence by noise, it is recommended that the frequency of the external clock, f_{xx} , be kept to within 32 MHz (system clock ϕ = 16 MHz).

6.3.2 PLL mode

In the PLL mode, an external clock is input by connecting an external oscillator, which is multiplied by the PLL synthesizer to generate system clock (ϕ).

The system clock (ϕ) can be selected between multiplication by 1 ($1 \times f_{xx}$) or by 5 ($5 \times f_{xx}$) of the external resonator or external clock frequency (f_{xx}). (Refer to 2.3.1. (12) PLLSEL)

| PLLSEL | ϕ |
|--------|-------------------|
| 0 | f_{xx} |
| 1 | $5 \times f_{xx}$ |

Caution Fix the PLLSEL pin so that the input level of this pin cannot be changed during operations (changes in input levels during operations may lead to incorrect operations). The PLLSEL pin has no function when the direct mode (CKSEL = 1) is set using the CKSEL pin. Leave it as an unused pin.

In the PLL mode, if the external oscillator or external clock source fails, the clock generator continues to provide the internal system clock (ϕ) based on the free-running frequency of the VCO. In this mode, the internal system clock ϕ operates at about 1 MHz (target).

Example of clock in PLL mode

| PLLSEL | System clock frequency (ϕ) [MHz] | External oscillator/external clock frequency (f_{xx}) [MHz] |
|-----------------------------------|---|---|
| 0 ($\phi = f_{xx}$) | 25.000 | |
| | 20.000 | |
| | 16.384 | |
| 1 ($\phi = 5 \times f_{xx}$) | 25.000 | 5.0000 |
| | 20.000 | 4.0000 |
| | 16.384 | 3.2768 |

6.4 PLL Stabilization

Following a power-on reset or when exiting the software STOP mode, an amount of time is required for the PLL to phase lock at a fixed frequency and stabilize. This required time is PLL lock-up time. The status in which the frequency is not stable is called unlock status and the status in which it has been stabilized is called lock status.

Two system status flags are available to check with the stabilization of the PLL frequency: UNLOCK flag that indicates the stabilization status of the PLL frequency, and PRERR flag that indicates occurrence of a protection error (for the details of the PRERR flag, refer to **6.5.2 (2) Command register (PRCMD)**).

The SYS register, which contains these UNLOCK and PRERR flags, can be read/written in 8- or 1-bit units.

| | | | | | | | | | | |
|-----|---|---|---|-------|---|---|---|--------|------------|-----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| SYS | 0 | 0 | 0 | PRERR | 0 | 0 | 0 | UNLOCK | Address | At reset |
| | | | | | | | | | FFFFFF078H | 0000000XB |

| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 0 | UNLOCK | Unlock Status Flag This is read-only flag and indicates unlock status of PLL. It holds "0" as long as lock up status is maintained, and is not initialized even if system is reset. 0 : Indicates lock status 1 : Indicates unlock status |

Remark For the description of the PRERR flag, refer to **6.5.2 (2) Command register (PRCMD)**.

If the unlock status condition should arise, due to a power or clock source failure, the UNLOCK flag should be checked to verify that the PLL has stabilized before performing any execution speed dependent operations, such as real-time processing.

The static processing such as setting of the on-chip hardware units and initialization of the register data and memory data, however, can be executed before the UNLOCK flag is reset.

The following is the relationship between the oscillation stabilization time (stabilization time of input waveform after resonator start oscillating) and PLL lock-up time (a time required for frequency stabilization) if a resonator is used.

Oscillation stabilization time < PLL lock-up time

6.5 Power Save Control

6.5.1 General

The V852 has following power save modes.

(1) HALT mode

In this mode, the clock generator (oscillator and PLL synthesizer) continues operation, but the operating clock of the CPU stops. The internal peripherals continue to function in reference to the internal system clock. The total power consumption of the system can be reduced through intermittent operations between normal operation and the HALT mode.

The HALT mode is entered by a dedicated instruction (HALT instruction).

(2) IDLE mode

In this mode, both the CPU clock and the internal system clock are stopped to further reduce power consumption. However, since the clock generator continues to run, normal operation can resume without having to wait for the oscillator and PLL circuits to stabilize.

The IDLE mode is entered by programming the PSC register.

The IDLE mode stands between the STOP and HALT modes in terms of clock stabilization time and power consumption, and is used in applications where the clock stabilization time should be eliminated but low power consumption is required.

(3) Software STOP mode

In this mode, the CPU clock, the internal system clock, and the clock generator are stopped, reducing power consumption to only the leakage current. In this state, power consumption is minimized.

(a) In PLL mode

The software STOP mode is entered by programming the PSC register. As soon as the oscillator circuit stops, the clock output of the PLL synthesizer is stopped. After the software STOP mode has been released, it is necessary to allow for stabilization time of the oscillator and system clock. Moreover, the lock up or stabilization time of the PLL may also be necessary, depending on the application. However, when the processor operates on an external clock, the need for oscillation stabilization time of the oscillator will become unnecessary.

(b) In direct mode

To stop the clock, fix the X1 pin to the low level.

The PLL lock up or stabilization time is not required to allow in the direct mode.

(4) Clock output inhibit

Output of the system clock from the CLKOUT pin is disabled.

The operations of the clock generator in the normal, HALT, IDLE, and software STOP modes are shown in Table 6-1.

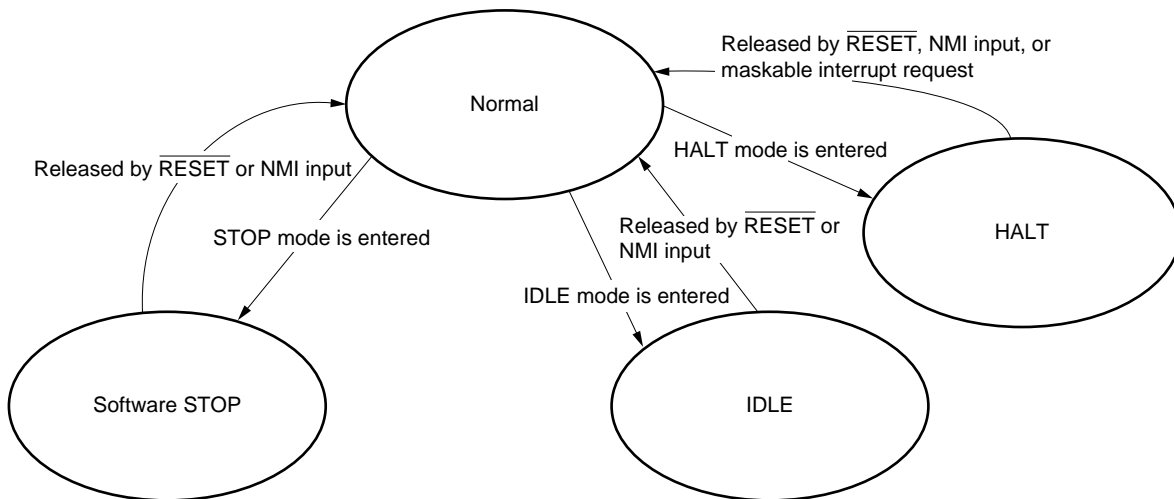
By combining and selecting the mode ideal for a specific application, the power consumption of the system can be effectively reduced.

Table 6-1. Operation of Clock Generator by Power Save Control

| Clock Source | | Standby Mode | Oscillator Circuit (OSC) | PLL Synthesizer | Clock Supply to Peripheral I/O | Clock Supply to CPU |
|--------------|--------------------------|--------------|--------------------------|-----------------|--------------------------------|---------------------|
| PLL mode | Oscillation by resonator | Normal | ○ | ○ | ○ | ○ |
| | | HALT | ○ | ○ | ○ | x |
| | | IDLE | ○ | ○ | x | x |
| | | STOP | x | x | x | x |
| | External clock | Normal | x | ○ | ○ | ○ |
| | | HALT | x | ○ | ○ | x |
| | | IDLE | x | ○ | x | x |
| | | STOP | x | x | x | x |
| Direct mode | Normal | x | x | ○ | ○ | |
| | HALT | x | x | ○ | x | |
| | IDLE | x | x | x | x | |
| | STOP | x | x | x | x | |

○ : operates
x : stops

Status Transition Diagram



6.5.2 Control registers

(1) Power save control register (PSC)

This is an 8-bit register that controls the power save mode. It can be written only by a specific combination of instruction sequences so that its contents are not written by mistake due to erroneous program execution. This register can be read/written in 8- or 1-bit units.

| | | | | | | | | | | |
|-----|-------|-------|------|-------|---|------|-----|---|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PSC | DCLK1 | DCLK0 | TBCS | CESEL | 0 | IDLE | STP | 0 | Address FFFFFF070H | At reset 00H |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|---------------------|---|-------|-------|------|---|---|--------------------|---|---|----------------|---|---|----------------|---|---|---------------------------|
| 7, 6 | DCLKn (n = 1, 0) | Disable CLKOUT Specifies operation mode of CLKOUT pin <table border="1" data-bbox="591 716 1377 926"> <thead> <tr> <th>DCLK1</th> <th>DCLK0</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Normal output mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Clock output inhibit mode</td> </tr> </tbody> </table> | DCLK1 | DCLK0 | Mode | 0 | 0 | Normal output mode | 0 | 1 | RFU (reserved) | 1 | 0 | RFU (reserved) | 1 | 1 | Clock output inhibit mode |
| DCLK1 | DCLK0 | Mode | | | | | | | | | | | | | | | |
| 0 | 0 | Normal output mode | | | | | | | | | | | | | | | |
| 0 | 1 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 0 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 1 | Clock output inhibit mode | | | | | | | | | | | | | | | |
| 5 | TBCS | Time Base Count Select Selects clock of time base counter 0: $fx/2^8$ 1: $fx/2^9$ For details, refer to explanation of “ Time Base Counter (TBC) ” in section 6.6 “ Specifying Oscillation Stabilization Time ”. | | | | | | | | | | | | | | | |
| 4 | CESEL | Crystal/External Select Specifies functions of X1 and X2 pins 0: Oscillator connected to X1 and X2 pins 1: External clock connected to X1 pin When CESEL = 1, the feedback loop of the oscillation circuit is cut so that the leakage current can be avoided during STOP mode and the oscillation stabilization time by the time base counter (TBC) after the STOP mode is released is not counted. | | | | | | | | | | | | | | | |
| 2 | IDLE | IDLE Mode Specifies IDLE mode. When “1” is written to this bit, IDLE mode is entered. When IDLE mode is released, this bit is automatically reset to “0”. | | | | | | | | | | | | | | | |
| 1 | STP | STOP Mode Specifies software STOP mode. When “1” is written to this bit, STOP mode is entered. When STOP mode is released, this bit is automatically reset to “0”. | | | | | | | | | | | | | | | |

The PSC register is programmed in the following special sequence:

- <1> The interrupt disable is set (PSW NP bit is set to 1).
- <2> Any 8-bit data is written in the command register (PRCMD).
- <3> The setting data is written in the PSC register (using the following instructions).
 - Store instruction (ST/SST instruction)
 - Bit manipulation instruction (SET1/CLR1/NOT1 instruction)
- <4> The interrupt disable is released (PSW NP bit is set to 0).
- <5> The NOP instruction (2 or 5 instructions) is inserted.

No special sequence is necessary for reading the PSC register.

Cautions 1. If interrupts are accepted in the time between the PRCMD issue (<2>) and the PSC register writing (<3>) directly after, PSC register is not written and protection error (SYS register PRERR bit is “1”) is generated in some cases. Therefore, set the PSW NP bit to 1 (<1>) and disable the INT/NMI acceptance.

The same applies when the bit manipulation instruction is used to set the PSC register. Insert the NOP instruction (<5>) as the dummy instruction so that the routine is executed correctly after the STOP/IDLE mode is released. If the PSW ID bit value is not to be changed by the execution of the instruction which returns the NP bit to 0 (<4>), insert two NOP instructions. If it is to be changed, insert five.

The following are examples.

[Example]

```

LDSR rX,5          ; NP bit = 1
ST.B r0,PRCMD [r0] ; Writing in PRCMD
ST.B rD,PSC [r0]   ; Sets PSC register
LDSR rY,5          ; NP bit = 0
NOP                ; Dummy instruction (2 or 5 instructions)
                  :
NOP
(next instruction) ; Execution routine after STOP/IDLE mode has been released
                  :

```

rX: Value written in PSW

rY: Value written back to PSW

rD: Value set to PSC

When saving the PSW value, it is necessary to transfer the PSW value before setting the NP bit to the rY register.

2. The instruction (<4> interrupt disable release, <5> NOP instruction) after the store instruction for the PSC register to be set to the software STOP mode and IDLE mode are executed before each power save mode is set.

(2) Command register (PRCMD)

The command register protects the PSC register from being illegally written so that the application system does not stop due to erroneous program execution.

Only data written first to the PSC register after the PRCMD register has been written becomes valid.

Because the register value can be rewritten only in a fixed sequence, illegal write operations are prevented.

The command register can be only written in 8-bit units (when this register is read, undefined data is read).

| | | | | | | | | | | |
|-------|------|------|------|------|------|------|------|------|-----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PRCMD | REG7 | REG6 | REG5 | REG4 | REG3 | REG2 | REG1 | REG0 | Address FFFFFF170H | At reset undefined |

| Bit Position | Bit Name | Function |
|--------------|--------------|---|
| 7 to 0 | REG7 to REG0 | Registration Code Registration code (any 8-bit data) |

Occurrence of an illegal store operation can be checked by the PRERR flag of the system status register (SYS).

| | | | | | | | | | | |
|-----|---|---|---|-------|---|---|---|--------|-----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| SYS | 0 | 0 | 0 | PRERR | 0 | 0 | 0 | UNLOCK | Address FFFFFF078H | At reset 0000000XB |

| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 4 | PRERR | Protection Error Flag Indicates that PSC register is not written in the correct sequence and that a protection error has occurred. 0: Protection error does not occur 1: Protection error occurs |

Remark For the description of the UNLOCK flag, refer to **6.4 PLL Stabilization**.

Operation conditions of PRERR flag

- ✦ Set condition (PRERR = "1") : <1> If the store instruction to the peripheral I/O most recently executed does not write data to the PRCMD register, but to the PSC register
<2> If the first store instruction executed after the write operation to the PRCMD register is to a peripheral I/O register except PSC register.
- ✦ Reset condition (PRERR = "0") : <1> When "0" is written to the PRERR flag of the SYS register.
<2> At system reset

6.5.3 HALT mode

(1) Entering and operation status

In the HALT mode, the clock generator (oscillator circuit and PLL synthesizer) operates, while the operating clock of the CPU stops. The internal peripherals continue to function in reference to the internal system clock. By entering the HALT mode during the idle time of the CPU, the total power consumption of the system can be reduced.

This mode is entered by the HALT instruction.

In the HALT mode, program execution stops, but the contents of the registers and internal RAM immediately before entering the HALT mode are retained. The on-chip peripheral functions that are not dependent on the instruction processing of the CPU continue to operate.

Table 6-2 shows the status of each hardware unit in the HALT mode.

Table 6-2. Operating Status in HALT Mode

| Function | | Operating Status | |
|------------------------------|--|--|---|
| Clock Generator | | Operates | |
| Internal System Clock | | Operates | |
| CPU | | Stops | |
| I/O Line | | Retained | |
| Peripheral Function | | Operates | |
| Internal Data | | Status of internal data before setting of HALT mode, such as CPU registers, status, data, and internal RAM contents, are retained. | |
| ★ External Expansion Mode | AD0 to AD15 | High impedance ^{Note} | |
| | A16 to A23 | Retained ^{Note} | High-impedance when $\overline{\text{HLDAK}} = 0$ |
| | $\overline{\text{LBEN}}, \overline{\text{UBEN}}$ | | |
| | $\overline{\text{R/W}}$ | High-level output ^{Note} | |
| | $\overline{\text{DSTB}}$ | | |
| | ASTB | | |
| | ST0, ST1 | Low-level output ^{Note} | |
| $\overline{\text{HLDAK}}$ | Operates | | |
| CLKOUT | | Clock output (when clock output is not inhibited) | |

Note The instruction fetch operation continues even after the HALT instruction has been executed, until the internal instruction prefetch queue becomes full. After the queue has become full, the operation stops in the status indicated in the above table.

(2) Releasing HALT mode

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

(a) Releasing by interrupt request

The HALT mode is unconditionally released by the NMI request or an unmasked maskable interrupt request, regardless of the priority. However, if the HALT mode is set in an interrupt processing routine, the operation differs as follows:

- (i) If an interrupt request with a priority lower than that of the interrupt request under execution is generated, the HALT mode is released, but the newly generated interrupt request is not accepted. The new interrupt request is kept pending.
- (ii) If an interrupt request with a priority higher (including NMI request) than the interrupt request under execution is generated, the HALT mode is released, and the interrupt request is also accepted.

Operation after HALT mode has been released by interrupt request

| Releasing Source | Interrupt Enable (EI) Status | Interrupt Disable (DI) Status |
|----------------------------|--|-------------------------------|
| NMI request | Branches to handler address | |
| Maskable interrupt request | Branches to handler address or executes next instruction | Executes next instruction |

(b) Releasing by $\overline{\text{RESET}}$ signal input

The same operation as the normal reset operation is performed.

6.5.4 IDLE mode

(1) Setting and operation status

In this mode, both the CPU clock and the internal system clock are stopped to further reduce power consumption. However, since the clock generator continues to run, normal operation can resume without having to wait for the oscillator and PLL circuit to stabilize.

The IDLE mode is entered when the PSC register is programmed by the store (ST/SST) instruction or bit manipulation (SET1/CLR1/NOT1) instruction.

Execution of the program is stopped in the IDLE mode, but the contents of the registers and internal RAM immediately before entering the IDLE mode are retained. The on-chip peripheral functions are stopped in this mode. External bus hold request (HLD \overline{RQ}) is not accepted.

Table 6-3 shows the hardware status in the IDLE mode.

Table 6-3. Operating Status in IDLE Mode

| Function | | Operating Status |
|-------------------------|---------------------------------------|--|
| Clock Generator | | Operates |
| Internal System Clock | | Stops |
| CPU | | Stops |
| I/O Line | | Retained |
| Peripheral Function | | Stops |
| Internal Data | | Status of all internal data immediately before IDLE mode is entered, such as CPU registers, status, data, and internal RAM contents, are retained. |
| External Expansion Mode | AD0 to AD15 | High-impedance |
| | A16 to A23 | |
| | \overline{LBEN} , \overline{UBEN} | |
| | R/\overline{W} | |
| | \overline{DSTB} | |
| | ASTB | |
| | ST0, ST1 | |
| | $\overline{HLD\overline{AK}}$ | |
| CLKOUT | | Low-level output |

(2) Releasing IDLE mode

The IDLE mode is released by the NMI signal input or $\overline{\text{RESET}}$ signal input.

(a) Releasing by NMI signal input

The NMI request is accepted and serviced as soon as the IDLE mode has been released.

If the IDLE mode is entered in the NMI processing routine, however, only the IDLE mode is released, and the interrupt will not be accepted. The interrupt request is retained and kept pending.

The interrupt processing that is started by the NMI signal input when the IDLE mode is released is treated in the same manner as a normal NMI interrupt that is processed (because there is only one vector address of the NMI interrupt). Therefore, if it is necessary to distinguish between the two types of NMI interrupts, a software status should be defined in advance, and the status must be set before setting the IDLE flag by the store/bit manipulation instruction. By checking this status during the NMI interrupt processing, the NMI used to release the IDLE mode can be distinguished from the normal NMI.

(b) Releasing by $\overline{\text{RESET}}$ signal input

The same operation as the normal reset operation is performed.

6.5.5 Software STOP mode

(1) Entering and operation status

In this mode, the clock generator (oscillation circuit and PLL synthesizer) is stopped, reducing power consumption to only leakage current. In this state, the whole system is stopped and power consumption is minimized.

The software STOP mode is entered by programming the PSC register using the store (ST/SST) or bit manipulation (SET1/CLR1/NOT1) instruction.

It is necessary to ensure the oscillation stabilization time of the oscillator circuit after the software STOP mode has been released, when the PLL mode (CKSEL pin = "0") and the resonator connection mode (CESEL bit = "0") are set.

In the software STOP mode, program execution is stopped, but all the contents of the registers and internal RAM immediately before entering the STOP mode are retained. The on-chip peripheral function also stops operation.

Table 6-4 shows the hardware status in the software STOP mode.

Table 6-4. Operating Status in Software STOP Mode

| Function | | Operating Status |
|-------------------------------------|---|---|
| Clock Generator | | Stops |
| Internal System Clock | | Stops |
| CPU | | Stops |
| I/O Line ^{Note} | | Retained |
| Peripheral Function ^{Note} | | Stops |
| Internal Data | | Status of all internal data immediately before software STOP mode is set, such as CPU registers, status, data, and internal RAM contents, are retained. |
| External Expansion Mode | AD0 to AD15 | High-impedance |
| | A16 to A23 | |
| | $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$ | |
| | $\overline{\text{R}/\overline{\text{W}}}$ | |
| | $\overline{\text{DSTB}}$ | |
| | ASTB | |
| | ST0, ST1 | |
| | $\overline{\text{HLDK}}$ | |
| CLKOUT | | Low-level output |

Note When the value of V_{DD} is within the operating range.

Even if V_{DD} drops below the minimum operating voltage, the contents of the internal RAM can be retained if the data retention voltage V_{DDR} is maintained.

(2) Releasing software STOP mode

The STOP mode is released by the NMI signal input or $\overline{\text{RESET}}$ signal input.

It is necessary to ensure the oscillation stabilization time when releasing from the STOP mode if the status is in the operating condition of the oscillator circuit (PLL mode (CKSEL pin = "0") and in the resonator connection mode (CESEL bit = "0")).

When using in the PROM programming mode, refer to **11.6 Cautions on STOP Mode Release when Using External Clock**.

(a) Releasing by NMI signal input

When the STOP mode is released by the NMI signal, the NMI request is also accepted.

If the STOP mode is set in an NMI processing routine, however, only the STOP mode is released, and the interrupt is not accepted. The interrupt request is retained and kept pending.

Caution If the external clock is input to the X1 pin, supply the external clock more than 150 μs before releasing by the NMI input.

NMI interrupt processing on releasing STOP mode

The interrupt processing that is started by the NMI signal input when the STOP mode is released is treated in the same manner as a normal NMI interrupt that is processed (because there is only one vector address of the NMI interrupt). Therefore, if it is necessary to distinguish between the two types of NMI interrupts, a software status should be defined in advance, and the status must be set before setting the STOP flag by the store/bit manipulation instruction. By checking this status during the NMI interrupt processing, the NMI used to release the STOP mode can be distinguished from the normal NMI.

(b) Releasing by $\overline{\text{RESET}}$ signal input

The same operation as the normal reset operation is performed.

Caution If the external clock is input to the X1 pin, supply the clock and maintain the low level width of the $\overline{\text{RESET}}$ pin for more than 150 μs .

6.6 Specifying Oscillation Stabilization Time

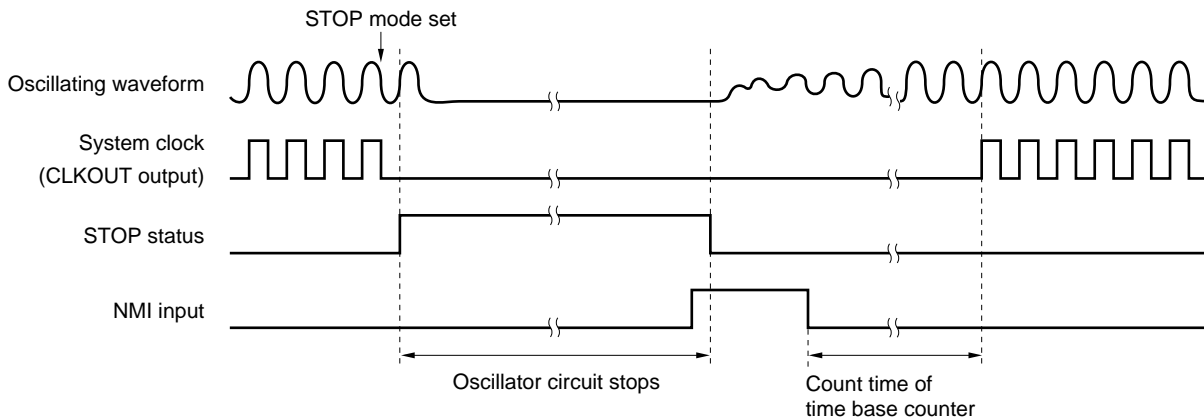
The time required for the oscillator circuit to become stabilized after the STOP mode has been released can be specified in the following two ways:

(1) By using internal time base counter (NMI signal input)

When the valid edge is input to the NMI pin, the STOP mode is released. When the inactive edge is input to the pin, the time base counter (TBC) starts counting, and the time required for the clock output from the oscillator circuit to become stabilized is specified by that count time.

Oscillation stabilization time = (Active level width after valid edge of NMI input has been detected) + (Count time of TBC)

After a specific time has elapsed, the system clock output is started, and execution branches to the vector address of the NMI interrupt.



Normally, the NMI pin should be kept at the inactive level (e.g. at a high level when the valid edge is specified to be the falling edge).

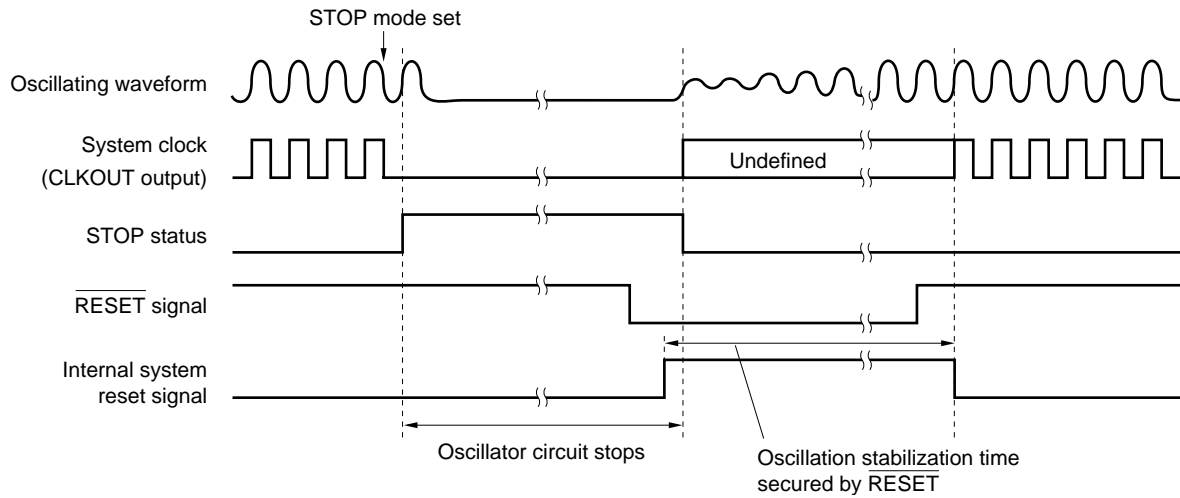
If an operation to enter the STOP mode is performed while a valid edge has been input to the NMI pin before the CPU accepts the interrupt, the STOP mode will immediately be released. Program execution is immediately started if the clock generator is in the direct mode (CKSEL = "1") or is driven by external clock (CESEL = 1). If the clock generator is in the PLL mode (CKSEL = "0") and is driven by a resonator (CESEL = 0), program execution is started after the oscillation stabilization time specified in the time base counter has elapsed, following the inactive edge input to the NMI pin.

(2) To specify time by signal level width ($\overline{\text{RESET}}$ signal input)

The STOP mode is released when the falling edge is input to the $\overline{\text{RESET}}$ pin.

The time required for the clock output from the oscillator circuit to become stabilized is specified by the low-level width of the signal input to the $\overline{\text{RESET}}$ pin.

After the rising edge has been input to the $\overline{\text{RESET}}$ pin, operation of the internal system clock begins, and execution branches to the vector address that is used when the system is reset.



Time base counter (TBC)

The time base counter is used to secure the oscillation stabilization time of the oscillator circuit when the software STOP mode is released.

- **When external clock is connected (CESEL bit of the PSC register = 1)**
Oscillation stabilization time count is not performed by TBC and the program execution starts immediately after the STOP mode cancelation.
- **When resonator is connected (CESEL bit of the PSC register = 0)**
Oscillation stabilization time is counted by TBC after the cancelation of the STOP mode and the program execution starts after counting finish.

The count clock of TBC is selected by the TBCS bit of the PSC register, and the following count time can be set:

Table 6-5. Example of Count Time
(a) At multiplication by 1 (PLLSEL = 0)

| TBCS | Count Clock | Count Time | | |
|------|--------------|-------------------------------|-------------------------------|-------------------------------|
| | | $f_{xx} = 13.500 \text{ MHz}$ | $f_{xx} = 20.000 \text{ MHz}$ | $f_{xx} = 25.000 \text{ MHz}$ |
| | | $\phi = 13.500 \text{ MHz}$ | $\phi = 20.000 \text{ MHz}$ | $\phi = 25.000 \text{ MHz}$ |
| 0 | $f_{xx}/2^8$ | 19.4 ms | 13.1 ms | 10.4 ms |
| 1 | $f_{xx}/2^9$ | 38.8 ms | 26.2 ms | 20.9 ms |

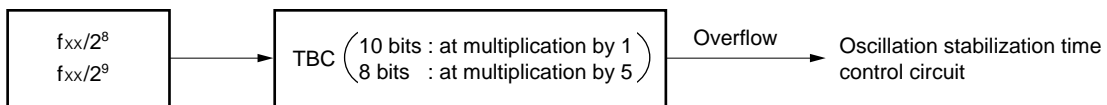
f_{xx} : external oscillator frequency
 ϕ : internal system clock frequency

(b) At multiplication by 5 (PLLSEL = 1)

| TBCS | Count Clock | Count Time | | |
|------|--------------|-------------------------------|-------------------------------|-------------------------------|
| | | $f_{xx} = 3.2768 \text{ MHz}$ | $f_{xx} = 4.0000 \text{ MHz}$ | $f_{xx} = 5.0000 \text{ MHz}$ |
| | | $\phi = 16.384 \text{ MHz}$ | $\phi = 20.000 \text{ MHz}$ | $\phi = 25.000 \text{ MHz}$ |
| 0 | $f_{xx}/2^8$ | 20.0 ms | 16.3 ms | 13.1 ms |
| 1 | $f_{xx}/2^9$ | 40.0 ms | 32.7 ms | 26.2 ms |

f_{xx} : external oscillator frequency
 ϕ : internal system clock frequency

Figure 6-1. Block Configuration



6.7 Clock Output Control

The operation mode of the CLKOUT pin can be selected by the DCLK0 and DCLK1 bits of the PSC register.

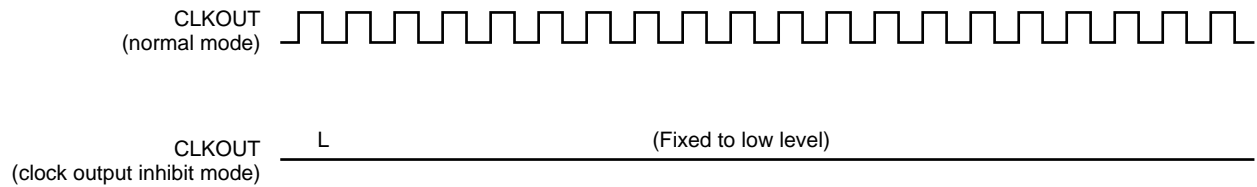
By using this operation mode in combination with the HALT, IDLE, or STOP mode, the power dissipation can be effectively reduced (for how to write these bits, refer to **6.5.2 Control registers**).

Clock output inhibit mode

The clock output from the CLKOUT pin is inhibited.

This mode is ideal for single-chip mode systems or systems that fetch instructions to external expansion devices or asynchronously access data.

Because the operation of CLKOUT is completely stopped in this mode, the power dissipation can be minimized and radiation noise from the CLKOUT pin can be suppressed.



[MEMO]

CHAPTER 7 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)**7.1 Features**

- Measures pulse intervals and frequency, and outputs programmable pulse
 - 16-bit measurement possible
 - Generates pulses of various shapes (interval pulse, one-shot pulse)
- Timer 1
 - 16-bit timer/event counter
 - Count clock source: 2 types (divided system clock and external pulse input)
 - Capture/compare register: 4
 - Count clear pin: TCLR1
 - Interrupt source: 5 types
 - External pulse output: 2
- Timer 4
 - 16-bit interval timer
 - Count clock selected from divided system clock
 - Compare register: 1
 - Interrupt source: 1

7.2 Basic Configuration

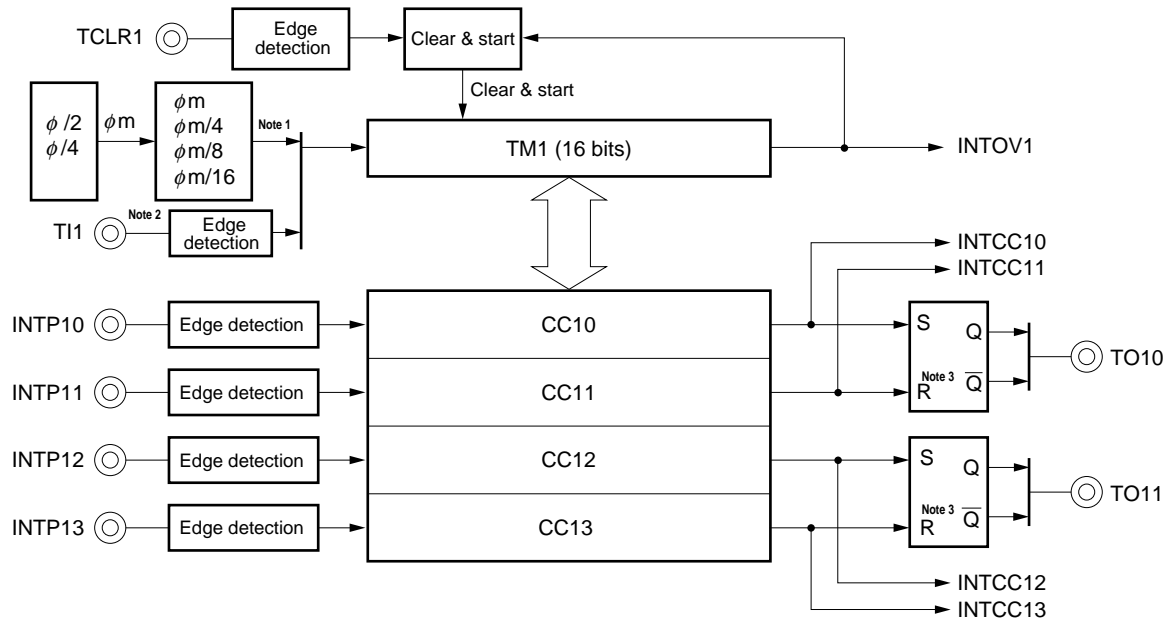
The basic configuration of the real-time pulse unit (RPU) is shown in the table below.

Table 7-1. Configuration of RPU

| Timer | Count Clock | Register | Read/Write | Generated Interrupt Signal | Capture Trigger | Timer Output SR | Other Function |
|---------|--|----------|------------|----------------------------|-----------------|-----------------|----------------|
| Timer 1 | $\phi / 2$ $\phi / 4$ $\phi / 8$ $\phi / 16$ $\phi / 32$ $\phi / 64$ T11 pin input | TM1 | Read | INTOV1 | – | – | External clear |
| | | CC10 | Read/write | INTCC10 | INTP10 | TO10 (S) | – |
| | | CC11 | Read/Write | INTCC11 | INTP11 | TO10 (R) | – |
| | | CC12 | Read/Write | INTCC12 | INTP12 | TO11 (S) | – |
| | | CC13 | Read/Write | INTCC13 | INTP13 | TO11 (R) | – |
| Timer 4 | $\phi / 32$ $\phi / 64$ $\phi / 128$ $\phi / 256$ | TM4 | Read | – | – | – | – |
| | | CM4 | Read/write | INTCM4 | – | – | – |

Remark ϕ : system clock
SR : set/reset

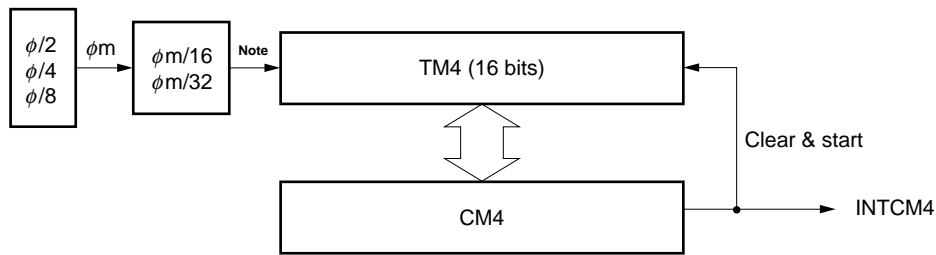
(1) Timer 1 (16-bit timer/event counter)



- Notes**
1. Internal count clock frequency
 2. External count clock frequency
 3. Reset priority

Remark ϕ indicates the system clock.

(2) Timer 4 (16-bit interval timer)



Note Internal count clock

Remark ϕ indicates the system clock.

7.2.1 Timer 1

(1) Timer 1 (TM1)

TM1 functions as a 16-bit free-running timer or event counter for external signals. Timer 1 is used to measure cycles and frequency, and also for pulse generation.

TM1 can be only read in 16-bit units.



TM1 counts up the internal count clock or external count clock. The TM1 is started or stopped by the CE1 bit of timer control register 1 (TMC1).

Whether the internal or external count clock is used is specified by the TMC1 register.

Caution Count clock cannot be changed during the timer operation.

(a) When external count clock is selected

TM1 operates as an event counter. The valid edge is specified by timer unit mode register 1 (TUM1), and TM1 counts up the signal input from the T11 pin

(b) When internal count clock is selected

TM1 operates as a free-running timer. The count clock selects the division by the prescaler by the TMC1 register from $\phi/2$, $\phi/4$, $\phi/8$, $\phi/16$, $\phi/32$, or $\phi/64$.

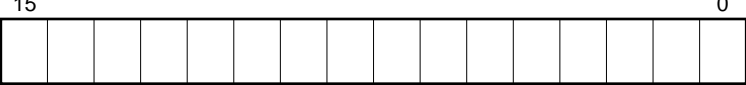
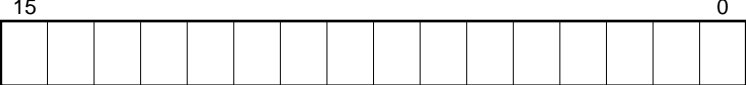
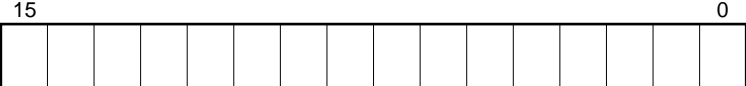
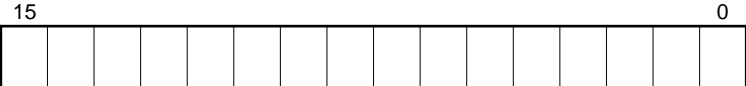
When the timer overflows, an overflow interrupt can be generated. The timer can be stopped after an overflow has occurred, if so specified by the TUM1 register.

The timer can be cleared and started by external TCLR1 input. At this time, the prescaler is cleared at the same time. As a result, the time from the TCLR1 input to the first count up by the timer is held constant, according to the division ratio of the prescaler. The operation is set by the TUM1 register.

When the $\overline{\text{RESET}}$ signal is input, all the bits of TM1 are cleared to 0.

(2) Capture/compare registers 10 to 13 (CC10 to CC13)

Capture/compare registers are 16-bit registers and are connected to the TM1. These registers can be used as capture or compare register depending on the specification of the timer unit mode register 1 (TUM1). They can be read/written in 16-bit units.

| | | | |
|------|--|-----------------------|-----------------------|
| CC10 |  | Address FFFFFF252H | At reset Undefined |
| CC11 |  | Address FFFFFF254H | At reset Undefined |
| CC12 |  | Address FFFFFF256H | At reset Undefined |
| CC13 |  | Address FFFFFF258H | At reset Undefined |

(a) When used as capture register

When a capture/compare register is used as a capture register, it detects the valid edge of the corresponding external interrupt (INTP_n (n = 10 to 13)) as a capture trigger. Timer 1 latches the count value in synchronization with the capture trigger (capture operation). The capture operation is performed asynchronously with the count clock. The latched value is held by the capture register, until the next capture operation is performed.

If the capture (latch) timing of the capture register contends with a register write operation by an instruction, the latter takes precedence, and the capture operation is ignored.

The valid edge of the external interrupt (rising, falling, or both edges) can be selected by external interrupt mode register (INTM2).

When a capture/compare register is used as a capture register, and when the valid edge of INTP_n is detected, an interrupt is generated. During this time, no interrupt can be generated by a coincidence signal INTCC_n (n = 10 to 13) of a compare register.

(b) When used as compare register

When a capture/compare register is used as a compare register, it compares its contents with the value of the timer at each clock tick. When the two values match, a coincidence signal INTCC_n is generated. The compare register is equipped with a set/reset output function. This function synchronizes with the generation of the coincidence signal and sets or resets the corresponding timer output.

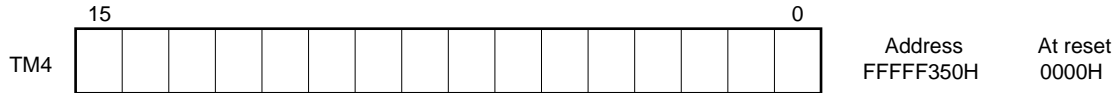
The interrupt source depends on the register mode, whether it is used as a capture or compare register. When used as a compare register, coincidence signal INTCC_n or the valid edge of INTP_n can be selected as an interrupt signal, depending on the specification of the TUM1 register.

When INTP_n is selected, an external interrupt from INTP_n is acknowledged and timer outputs by compare register's set/reset output function are enabled.

7.2.2 Timer 4

(1) Timer 4 (TM4)

TM4 is a 16-bit timer and is mainly used as an interval timer for software. This timer can be only read in 16-bit units.



TM4 is started or stopped by the CE4 bit of the timer control register 4 (TMC4).

The count clock selects the divider of the prescaler by the TMC4 register from $\phi/32$, $\phi/64$, $\phi/128$, or $\phi/256$.

All the bits of TM4 are cleared to 0 by the $\overline{\text{RESET}}$ input.

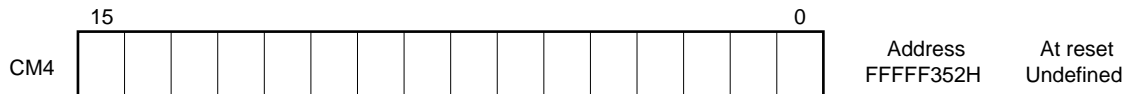
Cautions

1. When the value of the timer coincides with the value of the compare register, the timer is cleared by the next clock tick. When the division ratio is large, the timer value may not be cleared to 0 yet, even if the timer is read immediately after the occurrence of the coincidence signal interrupt.

2. Count clock cannot be changed during the timer operation.

(2) Compare register 4 (CM4)

CM4 is a 16-bit register and is connected to TM4. This register can be read/written in 16-bit units.



CM4 compares its value with the value of TM4 at each clock tick of TM4, and generates an interrupt (INTCM4) when the two values match or coincide with each other. TM4 is cleared in synchronization with this coincidence.

7.3 Control Registers

(1) Timer unit mode register 1 (TUM1)

TUM1 is a register that controls the operation of TM1, and specifies the operation mode of the capture/compare registers.

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

| | | | | | | | | | | | | | | | | | | |
|------|----|----|-----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|------------|----------|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TUM1 | 0 | 0 | OST | ECLR1 | TES11 | TES10 | CES11 | CES10 | CMS13 | CMS12 | CMS11 | CMS10 | IMS13 | IMS12 | IMS11 | IMS10 | Address | At reset |
| | | | | | | | | | | | | | | | | | FFFFFF240H | 0000H |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|--------------|--|-------|-------|------------|---|---|--------------|---|---|-------------|---|---|----------------|---|---|-------------------------------|
| 13 | OST | <p>Overflow Stop</p> <p>Specifies operation of timer after occurrence of overflow. This flag is valid only for TM1.</p> <p>0: Timer continues counting after overflow has occurred.</p> <p>1: Timer holds 0000H and stops after overflow has occurred.</p> <p>At this time, CE1 bit of TMC1 register remains "1".</p> <p>Timer resumes counting when following operation is performed:</p> <p>When ECLR1 = "0": Writing "1" to CE1 bit</p> <p>When ECLR1 = "1": Trigger input to timer clear pin (TCLR1)</p> | | | | | | | | | | | | | | | |
| 12 | ECLR1 | <p>External Input Timer Clear</p> <p>Enables clearing TM1 by external clear input (TCLR1)</p> <p>0: TM1 is not cleared by external input</p> <p>1: TM1 is cleared by external input</p> <p>After TM1 has been cleared, it starts counting.</p> | | | | | | | | | | | | | | | |
| 11, 10 | TES11, TES10 | <p>TI1 Edge Select</p> <p>Specifies valid edge of external clock input (TI1)</p> <table border="1"> <thead> <tr> <th>TES11</th> <th>TES10</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table> | TES11 | TES10 | Valid edge | 0 | 0 | Falling edge | 0 | 1 | Rising edge | 1 | 0 | RFU (reserved) | 1 | 1 | Both rising and falling edges |
| TES11 | TES10 | Valid edge | | | | | | | | | | | | | | | |
| 0 | 0 | Falling edge | | | | | | | | | | | | | | | |
| 0 | 1 | Rising edge | | | | | | | | | | | | | | | |
| 1 | 0 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 1 | Both rising and falling edges | | | | | | | | | | | | | | | |
| 9, 8 | CES11, CES10 | <p>TCLR1 Edge Select</p> <p>Specifies valid edge of external clear input (TCLR1)</p> <table border="1"> <thead> <tr> <th>CES11</th> <th>CES10</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edge</td> </tr> </tbody> </table> | CES11 | CES10 | Valid edge | 0 | 0 | Falling edge | 0 | 1 | Rising edge | 1 | 0 | RFU (reserved) | 1 | 1 | Both rising and falling edge |
| CES11 | CES10 | Valid edge | | | | | | | | | | | | | | | |
| 0 | 0 | Falling edge | | | | | | | | | | | | | | | |
| 0 | 1 | Rising edge | | | | | | | | | | | | | | | |
| 1 | 0 | RFU (reserved) | | | | | | | | | | | | | | | |
| 1 | 1 | Both rising and falling edge | | | | | | | | | | | | | | | |

| Bit Position | Bit Name | Function |
|--------------|----------------|---|
| 7 to 4 | CMS13 to CMS10 | Capture/Compare Mode Select Selects operation mode of capture/compare registers (CCn) 0: Capture register. However, capture operation is performed only when CE1 of TMC1 register = "1". 1: Compare register |
| 3 to 0 | IMS13 to IMS10 | Interrupt Mode Select Selects INTPn or INTCCn as interrupt source 0: Uses coincidence signal of INTCCn of compare register as interrupt signal 1: Uses external input signal INTPn as interrupt signal |

Remark n = 13 to 10

(2) Timer control register 1 (TMC1)

TMC1 controls the operation of TM1.

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

| | | | | | | | | | | |
|------|-----|---|---|-----|-------|-------|-------|---|------------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TMC1 | CE1 | 0 | 0 | ETI | PRS11 | PRS10 | PRM11 | 0 | Address | At reset |
| | | | | | | | | | FFFFFF242H | 00H |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|--------------|--|-------|-------|-------------|---|---|----------|---|---|------------|---|---|------------|---|---|-------------|
| 7 | CE1 | Count Enable Controls timer operation. 0: Timer stops at "0000H" and does not operate. 1: Timer performs count operation. However, it does not start counting when TUM1.ECLR1 = "1", until TCLR1 signal is input. When TUM1.ECLR1 = "0", starting counting of timer by CE1 = "1" is triggered by writing "1" to CE1 bit. Therefore, timer is not started even when TUM1.ECLR1 = "0" after CE1 has been set with TUM1.ECLR1 = "1". | | | | | | | | | | | | | | | |
| 4 | ETI | External TI1 Input Specifies external or internal count clock. 0: ϕ (internal) 1: TI1 (external) | | | | | | | | | | | | | | | |
| 3, 2 | PRS11, PRS10 | Prescaler Clock Select Selects internal count clock (ϕ_m is intermediate clock) <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>PRS11</th> <th>PRS10</th> <th>Count clock</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>ϕ_m</td> </tr> <tr> <td>0</td> <td>1</td> <td>$\phi_m/4$</td> </tr> <tr> <td>1</td> <td>0</td> <td>$\phi_m/8$</td> </tr> <tr> <td>1</td> <td>1</td> <td>$\phi_m/16$</td> </tr> </tbody> </table> | PRS11 | PRS10 | Count clock | 0 | 0 | ϕ_m | 0 | 1 | $\phi_m/4$ | 1 | 0 | $\phi_m/8$ | 1 | 1 | $\phi_m/16$ |
| PRS11 | PRS10 | Count clock | | | | | | | | | | | | | | | |
| 0 | 0 | ϕ_m | | | | | | | | | | | | | | | |
| 0 | 1 | $\phi_m/4$ | | | | | | | | | | | | | | | |
| 1 | 0 | $\phi_m/8$ | | | | | | | | | | | | | | | |
| 1 | 1 | $\phi_m/16$ | | | | | | | | | | | | | | | |
| 1 | PRM11 | Prescaler Clock Mode Selects intermediate clock ϕ_m of count clock (ϕ is system clock). 0: $\phi/2$ 1: $\phi/4$ | | | | | | | | | | | | | | | |

Caution Do not change the count clock frequency while the timer operates.

(3) Timer control register 4 (TMC4)

TMC4 controls the operation of TM4.

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

| | | | | | | | | | | |
|------|-----|---|---|---|---|-------|-------|-------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TMC4 | CE4 | 0 | 0 | 0 | 0 | PRS40 | PRM41 | PRM40 | Address FFFFFF342H | At reset 00H |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|--------------|---|-------|-------|----------|---|---|----------|---|---|----------|---|---|----------|---|---|----------------|
| 7 | CE4 | Count Enable Controls operation of timer. 0: Timer stops at "0000H" and does not operate. 1: Timer performs count operation. | | | | | | | | | | | | | | | |
| 2 | PRS40 | Prescaler Clock Select Selects internal count clock (ϕ_m is intermediate clock). 0: $\phi_m/16$ 1: $\phi_m/32$ | | | | | | | | | | | | | | | |
| 1, 0 | PRM41, PRM40 | Prescaler Clock Mode Selects intermediate clock ϕ_m of count clock (ϕ is system clock). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>PRM41</th> <th>PRM40</th> <th>ϕ_m</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>$\phi/2$</td> </tr> <tr> <td>0</td> <td>1</td> <td>$\phi/4$</td> </tr> <tr> <td>1</td> <td>0</td> <td>$\phi/8$</td> </tr> <tr> <td>1</td> <td>1</td> <td>RFU (reserved)</td> </tr> </tbody> </table> | PRM41 | PRM40 | ϕ_m | 0 | 0 | $\phi/2$ | 0 | 1 | $\phi/4$ | 1 | 0 | $\phi/8$ | 1 | 1 | RFU (reserved) |
| PRM41 | PRM40 | ϕ_m | | | | | | | | | | | | | | | |
| 0 | 0 | $\phi/2$ | | | | | | | | | | | | | | | |
| 0 | 1 | $\phi/4$ | | | | | | | | | | | | | | | |
| 1 | 0 | $\phi/8$ | | | | | | | | | | | | | | | |
| 1 | 1 | RFU (reserved) | | | | | | | | | | | | | | | |

Caution Do not change the count clock frequency while the timer operates.

(4) Timer output control register 1 (TOC1)

TOC1 controls the timer output from the TO10 and TO11 pins.

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

| | | | | | | | | | | |
|------|--------|-------|--------|-------|---|---|---|---|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TOC1 | ENTO11 | ALV11 | ENTO10 | ALV10 | 0 | 0 | 0 | 0 | Address FFFFFF244H | At reset 00H |

| Bit Position | Bit Name | Function |
|--------------|----------------|---|
| 7, 5 | ENTO11, ENTO10 | Enable TOxx pin xx = 11, 10 Enables corresponding timer output (TOxx). 0: Timer output is disabled. The reverse phase level of active level specified in the ALV bit (inactive level) is output from the corresponding TOxx pin. Even if coincidence signal is generated from corresponding compare register, level of TOxx pin does not change. 1: Timer output function is enabled. Timer output changes when coincidence signal is generated from corresponding compare register. After the timer output has been enabled before the first coincidence signal is generated, the reverse phase level of active level specified in the ALV bit is output. |
| 6, 4 | ALV11, ALV10 | Active Level TOxx pin xx = 11, 10 Specifies active level of timer output. 0: Active-low 1: Active-high |

Remark F/F of TOxx output gives priority to reset.

Caution The TOxx output is not changed by the external interrupt signal (INTPn) (n = 10 to 13). When using TOxx, specify a capture/compare register as a compare register (CMSn = 1) (n = 10 to 13).

(5) External interrupt mode register 2 (INTM2)

If n (n = 10 to 13) of TM1 is used as a capture register, a valid edge of external interrupt INTPn is used as a capture trigger. This valid edge can be specified with the INTM2 register. For details, refer to **Section 5.3.6 “External interrupt mode registers 1 and 2 (INTM1 and INTM2)”**.

(6) Timer overflow status register (TOVS)

Flags that indicate occurrence of an overflow from TM1 and TM4 are assigned to this register.

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

By testing and resetting the TOVS register via software, occurrence of an overflow can be polled.

| | | | | | | | | | | |
|------|---|---|---|------|---|---|------|---|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TOVS | 0 | 0 | 0 | OVF4 | 0 | 0 | OVF1 | 0 | Address FFFFFF230H | At reset 00H |

| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 4, 1 | OVFn | <p>Overflow Flag TMn (n = 4, 1) overflow flag. 0: No overflow from TMn 1: Overflow from TMn</p> <p>Caution The INTOV1 interrupt request signal is generated for the interrupt controller by synchronizing with the overflow from the TM1. However, the interrupt operations are independent from the TOVS. Overflow flags (OVF1) from the TM1 can be manipulated by the software like other overflow flags. At this time, the interrupt request flag (OVF1) in the interrupt controller for the INTOV1 will not be affected.</p> <p>The overflow flags will not be transmitted to the TOVS register while accessing from the CPU. Therefore, when overflows occur during reading of the TOVS register, this overflow condition will be reflected the next time the TOVS register is read, without changing the flag value.</p> |

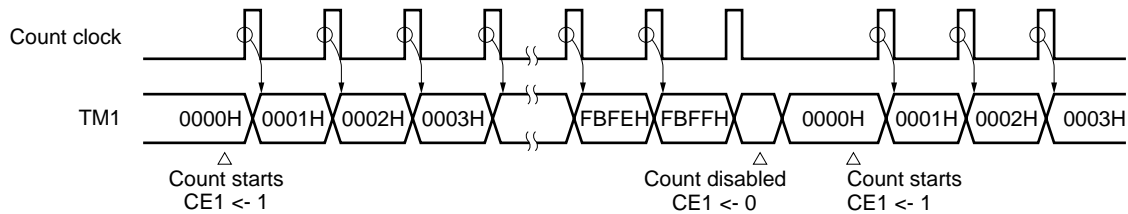
7.4 Timer 1 Operation

7.4.1 Count operation

Timer 1 functions as a 16-bit free-running timer or event counter, as specified by timer control register 1 (TMC1).

When it is used as a free-running timer, and when the count value of TM1 coincides with the value of the CCn (n = 10 to 13) register, an interrupt signal is generated, and timer output TOxx (xx = 10, 11) can be set/reset. In addition, a capture operation that holds the current count value of TM1 and loads it into the register CCn, is performed in synchronization with the valid edge detected from the corresponding external interrupt request pin as an external trigger. The captured value is retained until the next capture trigger is generated.

Figure 7-1. Basic Operation of Timer 1



7.4.2 Selecting count clock frequency

An internal or external count clock frequency can be input to timer 1. Which count clock frequency is used is specified by the ETI bit of the TMC1 register.

Caution Do not change the count clock frequency while the timer operates.

(1) Internal count clock (ETI bit = 0)

An internal count clock frequency is selected by the PRM11, PRS11, and PRS10 bits of the TMC1 register, from $\phi/2$, $\phi/4$, $\phi/8$, $\phi/16$, $\phi/32$, and $\phi/64$.

| PRS11 | PRS10 | PRM11 | Count clock frequency |
|-------|-------|-------|-----------------------|
| 0 | 0 | 0 | $\phi/2$ |
| 0 | 0 | 1 | $\phi/4$ |
| 0 | 1 | 0 | $\phi/8$ |
| 0 | 1 | 1 | $\phi/16$ |
| 1 | 0 | 0 | $\phi/16$ |
| 1 | 0 | 1 | $\phi/32$ |
| 1 | 1 | 0 | $\phi/32$ |
| 1 | 1 | 1 | $\phi/64$ |

(2) External count clock (ETI bit = 1)

The signal input to the TI1 pin is counted. At this time, timer 1 can operate as an event counter.

The valid edge of TI1 is specified by the TES11 and TES10 bits of the TUM1 register.

| TES11 | TES10 | Valid Edge |
|-------|-------|-------------------------------|
| 0 | 0 | Falling edge |
| 0 | 1 | Rising edge |
| 1 | 0 | RFU (reserved) |
| 1 | 1 | Both rising and falling edges |

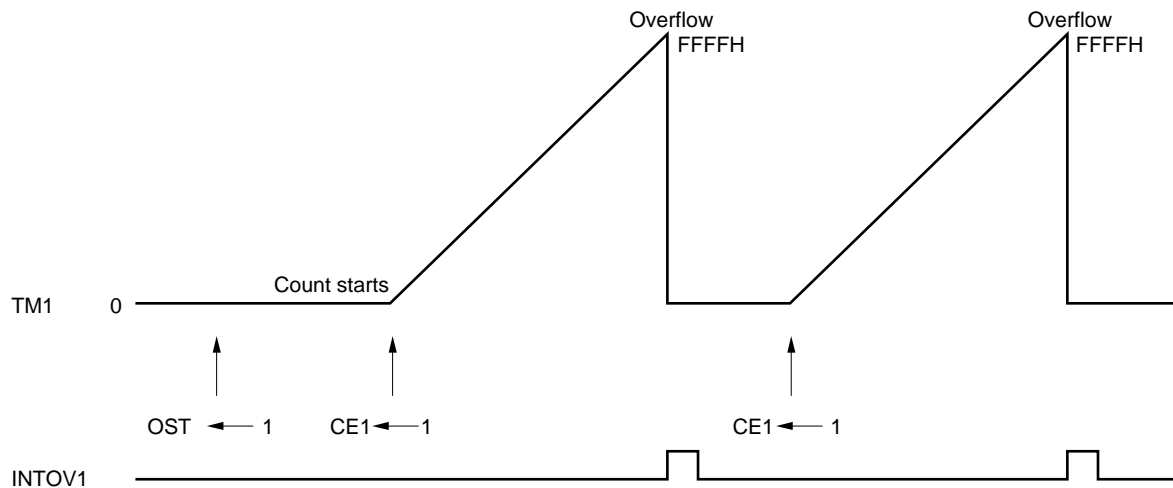
7.4.3 Overflow

- ★ If the TM1 register overflows as a result of counting the count clock frequency to FFFFH, the OVF1 bit of the TOVS register is set to 1, and an overflow interrupt (INTOV) is generated.

After the overflow has occurred, the timer can be stopped by setting the OST bit of the TUM1 register to "1". If the timer is stopped due to overflow, the counting operation is not resumed until CE is set to "1" by software.

The operation is not affected even if CE1 is set to 1 during count operation.

Figure 7-2. Operation after Occurrence of Overflow (when ECLR1 = 0, OST = 1)



7.4.4 Clearing/starting timer by TCLR1 input

Timer 1 usually starts the count operation when the CE1 bit of the TMC1 register is set to 1. It is also possible to clear TM1 and start the count operation by using external input TCLR1.

When the valid edge is input to TCLR1 after ECLR1 = 1, OST = 0, and CE1 is set to 1, the count operation is started. If the valid edge is input to TCLR1 during operation, TM1 clears its value and then resumes the count operation (refer to **Figure 7-3**).

When the valid edge is input to TCLR1 after ECLR1 = 1, OST = 1, and CE1 is set to 1 from 0, the count operation is started. When TM1 overflows, the count operation is stopped once and is not resumed until the valid edge is input to TCLR1. If the valid edge of TCLR1 is detected during count operation, TM1 is cleared and continues counting (refer to **Figure 7-4**). The count operation is not started if setting the CE1 to 1 after the overflow.

Figure 7-3. Clearing/Starting Timer by TCLR1 Input (when ECLR1 = 1, OST = 0)

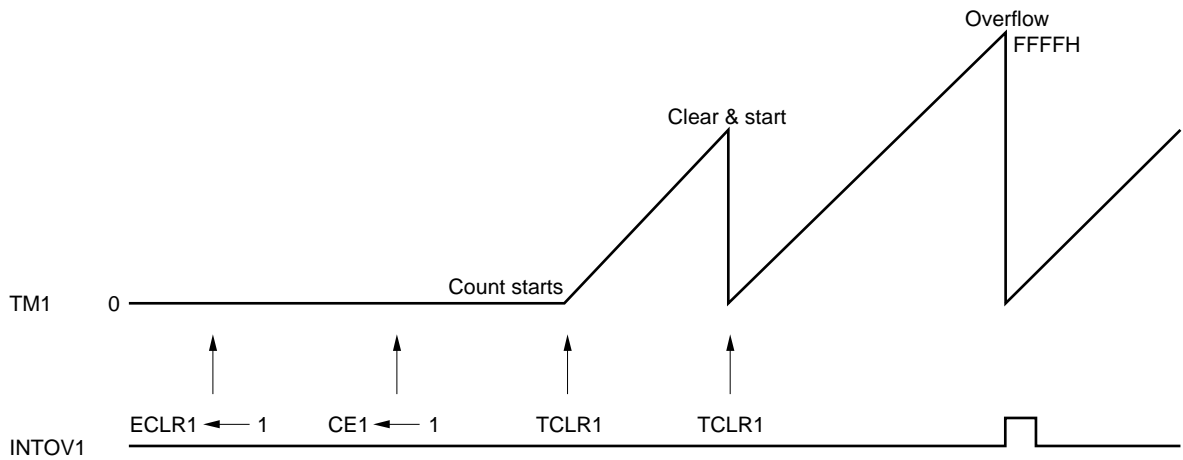
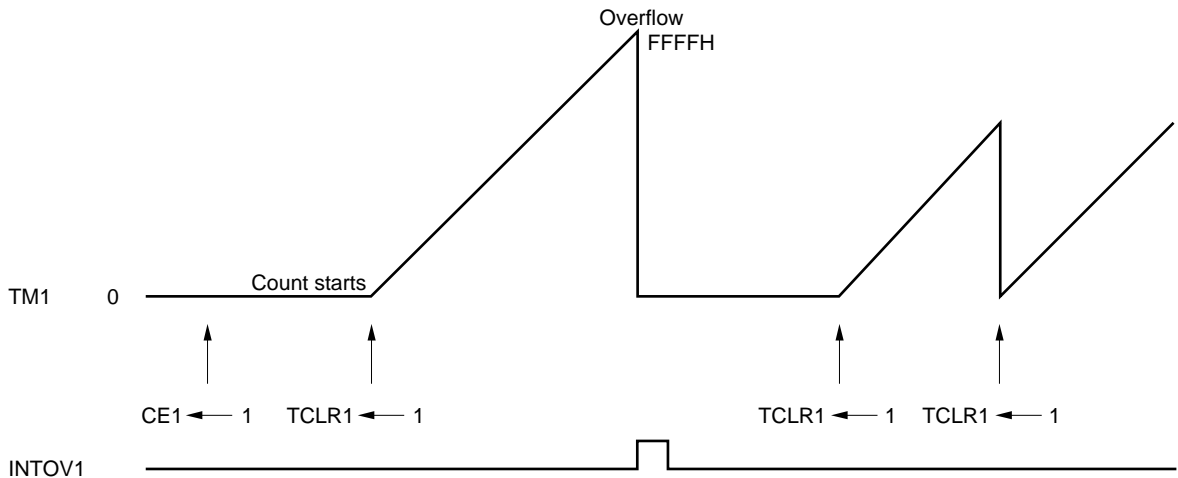


Figure 7-4. Relationships between Clear/Start by TCLR1 Input and Overflow (when ECLR1 = 1, OST = 1)



7.4.5 Capture operation

A capture operation that captures and holds the count value of TM1 and loads it to a capture register in asynchronization with an external trigger can be performed. The valid edge from the external interrupt request input pin INTP_n (n = 10 to 13) is used as the capture trigger. In synchronization with this capture trigger signal, the count value of TM1 during counting, is captured and loaded to the capture register. The value of the capture register is retained until the next capture trigger is generated.

Interrupt signal INTCC_n is generated from INTP_n input signal.

Table 7-2. Capture Trigger Signal to 16-Bit Capture Register (TM1)

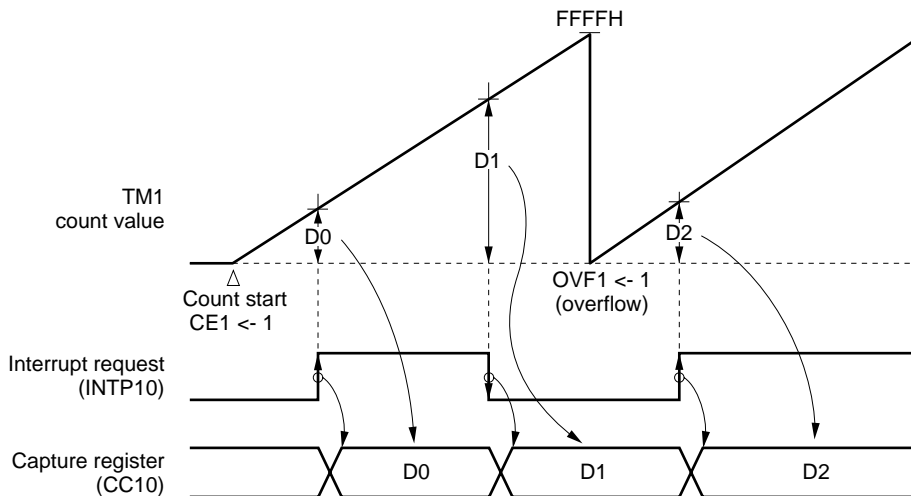
| Capture Register | Capture Trigger Signal |
|------------------|------------------------|
| CC10 | INTP10 |
| CC11 | INTP11 |
| CC12 | INTP12 |
| CC13 | INTP13 |

Remark CC10 to CC13 are capture/compare registers. Whether these registers are used as capture or compare registers is specified by timer unit mode register 1 (TUM1).

The valid edge of the capture trigger is set by external interrupt mode register (INTM2).

When both the rising and falling edges are specified as the capture trigger, the width of an externally input pulse can be measured. If either the rising or falling edge is specified as the capture trigger, the frequency of the input pulse can be measured.

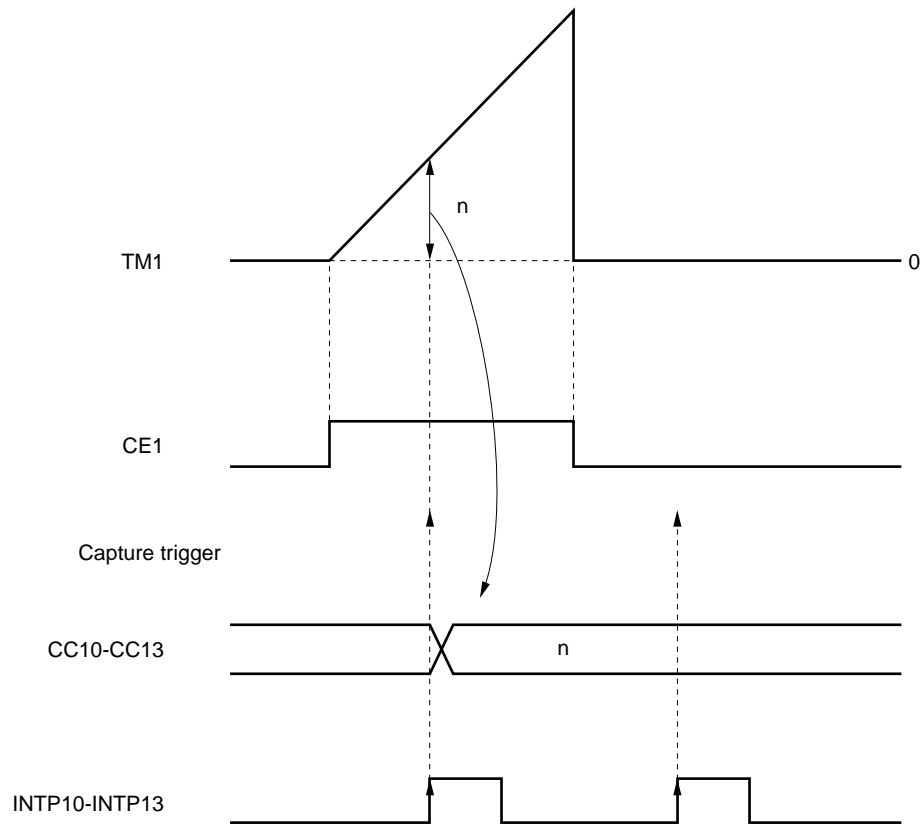
Figure 7-5. Example of TM1 Capture Operation (when both edges are specified)



Remark D_n (n = 0, 1, 2, ...): count value of TM1

The capture operation is not performed even if the interrupt signal is input when CE1 is cleared to 0.

Figure 7-6. Example of TM1 Capture Operation



7.4.6 Compare operation

A comparison between the value in a compare register with the count value of TM1 can be performed.

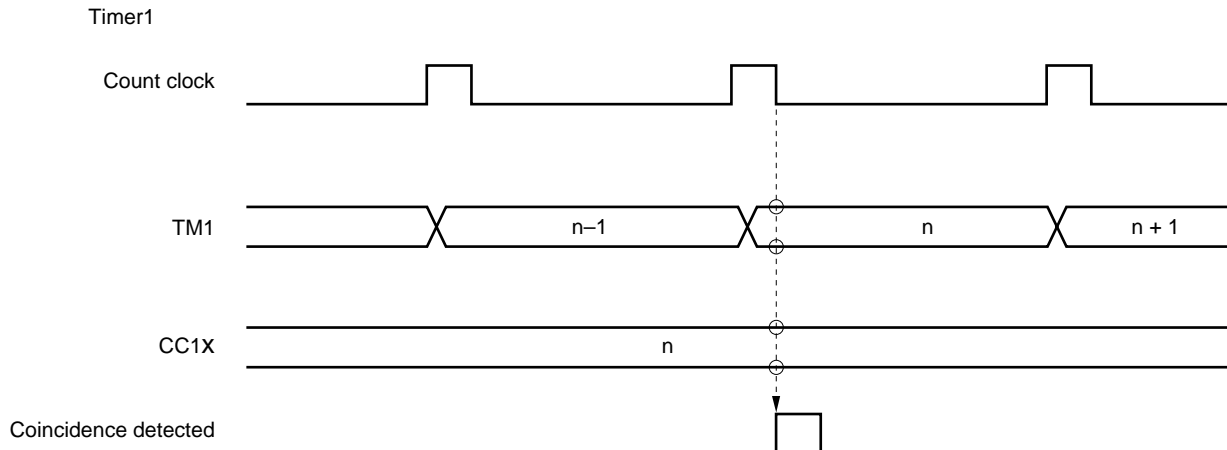
When the count value of TM1 coincides with the value of the compare register programmed in advance, a coincidence signal is sent to the output control circuit (refer to **Figure 7-7**). The levels of the timer output pins (TO10 and TO11) can be changed by the coincidence signal, and an interrupt request signal can be generated at the same time.

Table 7-3. Interrupt Request Signal from 16-Bit Compare Register (TM1)

| Compare Register | Interrupt Request Signal |
|------------------|--------------------------|
| CC10 | INTCC10 |
| CC11 | INTCC11 |
| CC12 | INTCC12 |
| CC13 | INTCC13 |

Remark CC10 to CC13 are capture/compare registers. Whether these registers are used as capture or compare registers is specified by timer unit mode register 1 (TUM1).

Figure 7-7. Example of Compare Operation



Remark Note that the coincidence signal is generated immediately after TM1 is incremented as shown above.

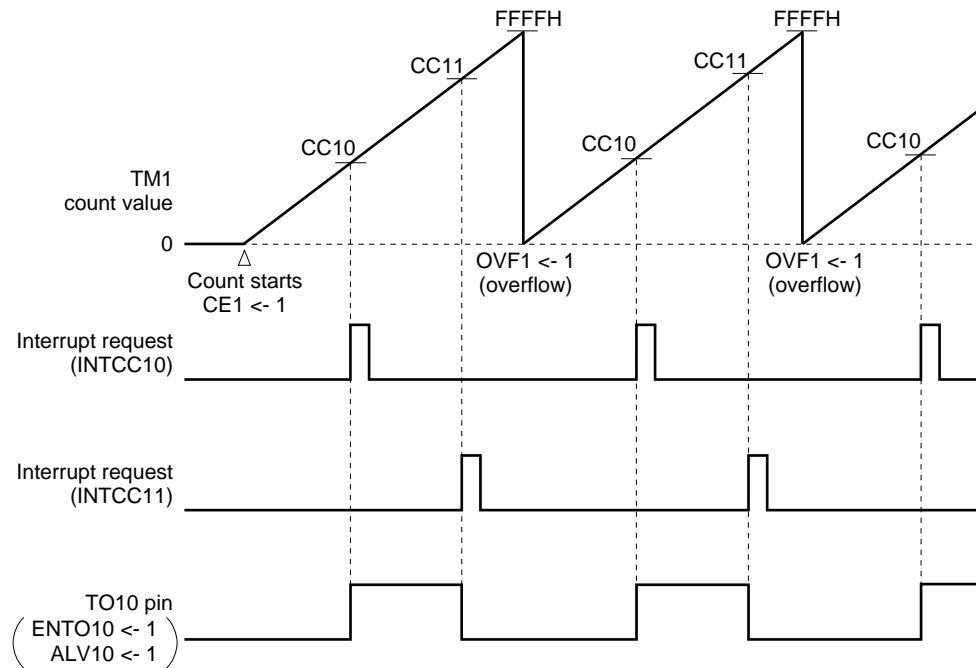
TM1 has two timer output pins: TO10 and TO11.

The count value of TM1 is compared with the value of CC10. When the two values coincide, the output level of the TO10 pin is set. The count value of TM1 is also compared with the value of CC11. When the two values coincide, the output level of the TO10 pin is reset.

Similarly, the count value of TM1 is compared with the value of CC12. When the two values coincide, the output level of the TO11 pin is set. The count value of TM1 is also compared with the value of CC13. When the two values coincide, the output level of TO11 pin is reset.

The output levels of the TO10 and TO11 pins can be specified by the TOC1 register.

Figure 7-8. Example of TM1 Compare Operation (set/reset output mode)



7.5 Timer 4 Operation

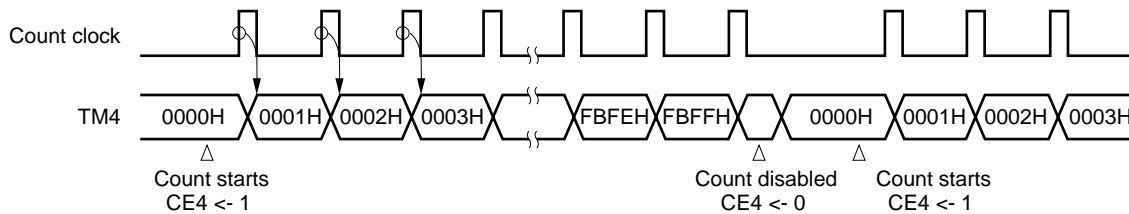
7.5.1 Count operation

Timer 4 functions as a 16-bit interval timer. The operation is specified by the timer control register 4 (TMC4).

The operation of timer 4 counts the internal count clocks ($\phi/32$ to $\phi/256$) specified by the PRS40, PRM41, and PRM40 bits of the TMC4 register.

If the count value of TM4 coincides with the value of CM4, the value TM4 is cleared while simultaneously a coincidence interrupt (INTCM4) is generated.

Figure 7-9. Basic Operation of Timer 4



7.5.2 Selecting the count clock frequency

An internal count clock frequency is selected by the PRS40, PRM40, and PRM41 bits of the TMC4 register, from $\phi/32$, $\phi/64$, $\phi/128$, and $\phi/256$.

Caution Do not change the count clock frequency while the timer operates.

| PRS40 | PRM40 | PRM41 | Count clock frequency |
|-------|-------|-------|-----------------------|
| 0 | 0 | 0 | $\phi/32$ |
| 0 | 0 | 1 | $\phi/64$ |
| 0 | 1 | 0 | $\phi/128$ |
| 0 | 1 | 1 | RFU (reserved) |
| 1 | 0 | 0 | $\phi/64$ |
| 1 | 0 | 1 | $\phi/128$ |
| 1 | 1 | 0 | $\phi/256$ |
| 1 | 1 | 1 | RFU (reserved) |

7.5.3 Overflow

If TM4 overflows, the OVF4 bit of the TOVS register is set to 1.

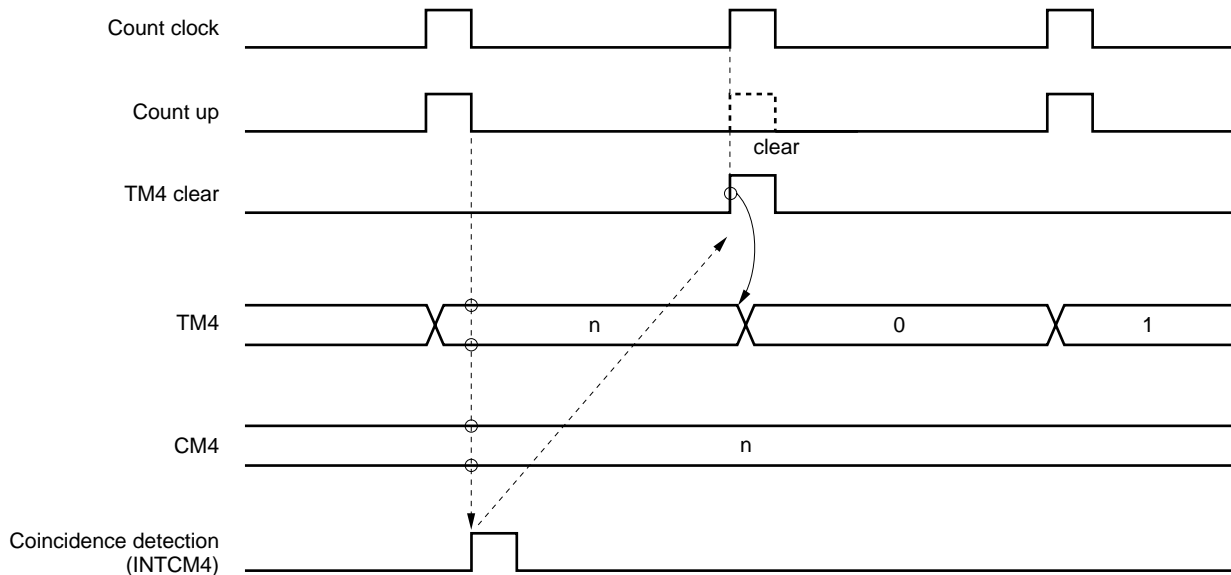
7.5.4 Compare operation

A comparison can be performed with the counter value of TM4 and the compare register (CM4).

When the count value of TM4 coincides with the value of the compare register, a coincidence interrupt (INTCM4) is generated. As a result, TM4 is cleared to 0 at the next count timing (refer to **Figure 7-10**). This function allows timer 4 to be used as an interval timer.

CM4 can be also set to 0. In this case, a coincidence is detected when TM4 overflows and is cleared to 0, and INTCM4 is generated. The value of TM4 is cleared to 0 at the next count timing, but INTCM4 is not generated when a coincidence occurs at this time (refer to **Figure 7-11**).

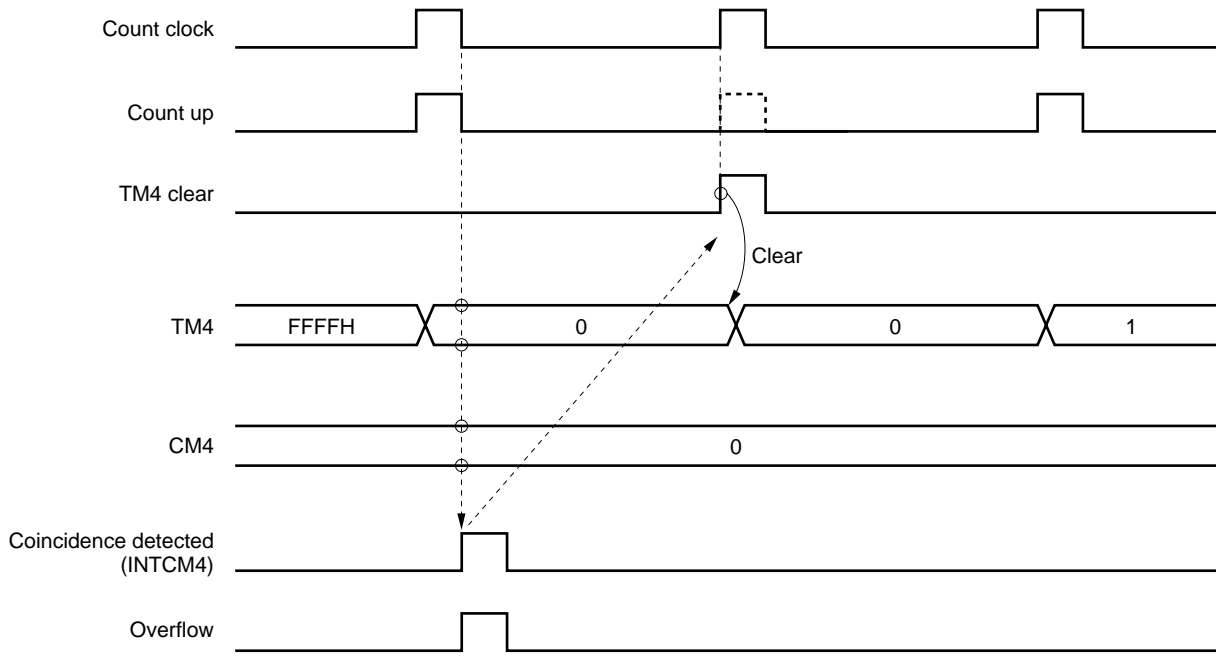
Figure 7-10. Operation with CM4 at 1 to FFFFH



Remark Interval time = $(n + 1) \times$ count clock cycle

$n = 1$ to 65535 (FFFFH)

Figure 7-11. When CM4 Is Set to 0



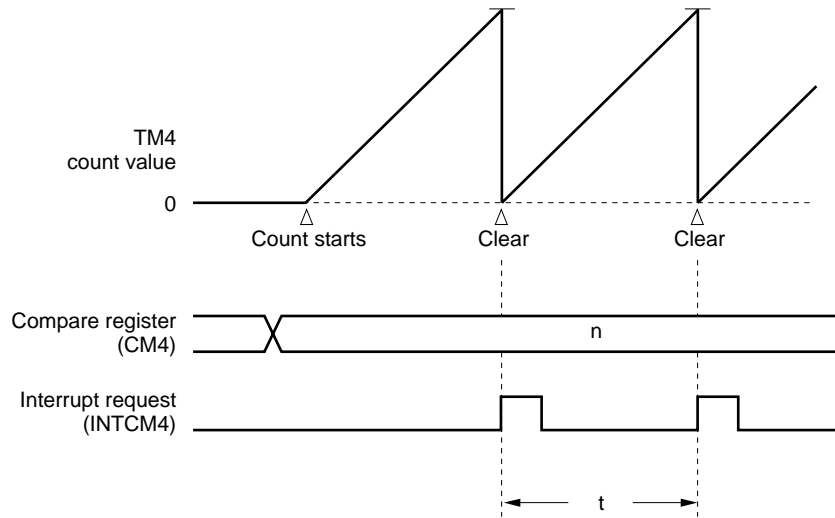
Remark Interval time = (FFFFH + 2) x count clock cycle

7.6 Application Examples

(1) Operation as interval timer (timer 4)

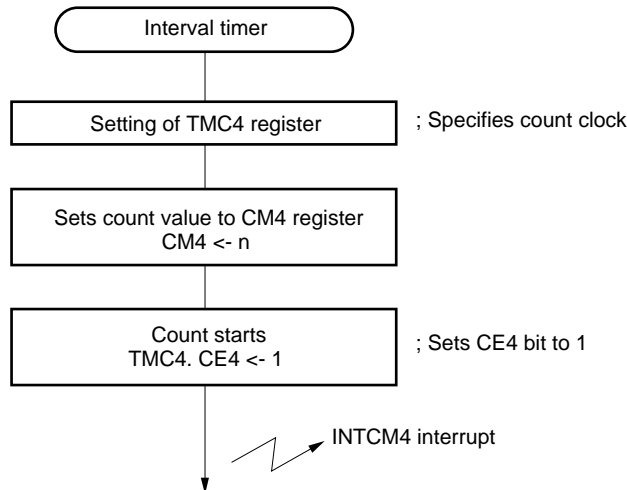
Timer 4 is used as an interval timer that repeatedly generates an interrupt request at time intervals specified by the count value set in advance to compare register CM4. Figure 7-12 shows the timing. Figure 7-13 illustrates the setting procedure.

Figure 7-12. Example of Timing of Interval Timer Operation (timer 4)



Remark n: value of CM4 register
 t: interval time = (n + 1) x count clock cycle

Figure 7-13. Setting Procedure of Interval Timer Operation (timer 4)



(2) Pulse width measurement (timer 1)

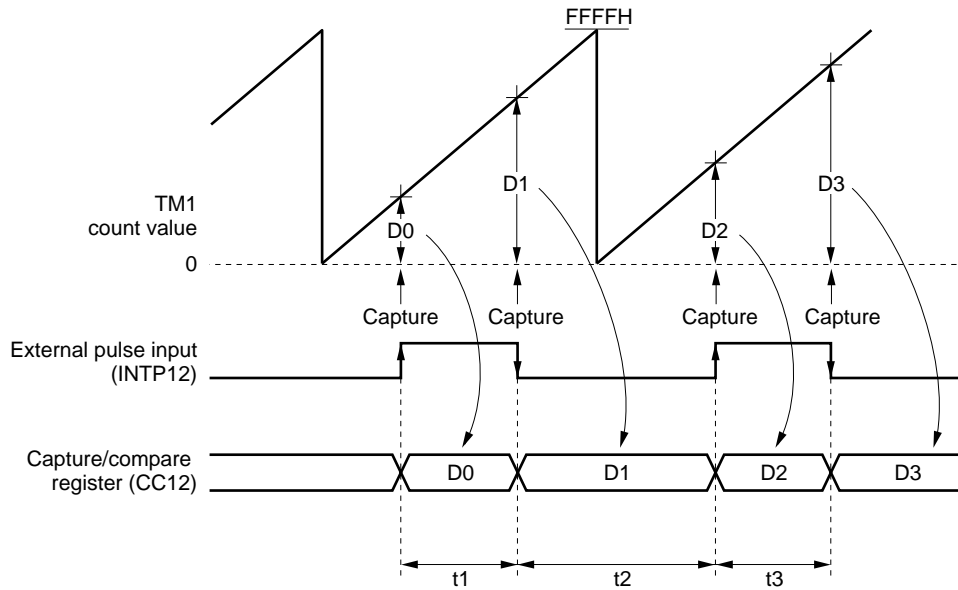
Timer 1 is used to measure pulse width.

In this example, the width of the high or low level of an external pulse input to the INTP12 pin is measured. The value of timer 1 (TM1) is captured to a capture/compare register (CC12) in synchronization with the valid edge of the INTP12 pin (both the rising and falling edges), as shown in Figure 7-14.

To calculate the pulse width, the difference between the count value of TM1 captured to the CC12 register on detection of valid edge n (D_n), and the count value on detection of valid edge $(n-1)$ (D_{n-1}) is calculated. This difference is multiplied by the count clock.

Figure 7-15 shows the setting procedure.

Figure 7-14. Pulse Width Measurement Timing (timer 1)



★

$$t_1 = (D_1 - D_0) \times \text{count clock cycle}$$

$$t_2 = \{(10000H - D_1) + D_2\} \times \text{count clock cycle}$$

$$t_3 = (D_3 - D_2) \times \text{count clock cycle}$$

Remark D_n : count value of TM1 ($n = 0, 1, 2, \dots$)

Figure 7-15. Setting Procedure for Pulse Width Measurement (timer 1)

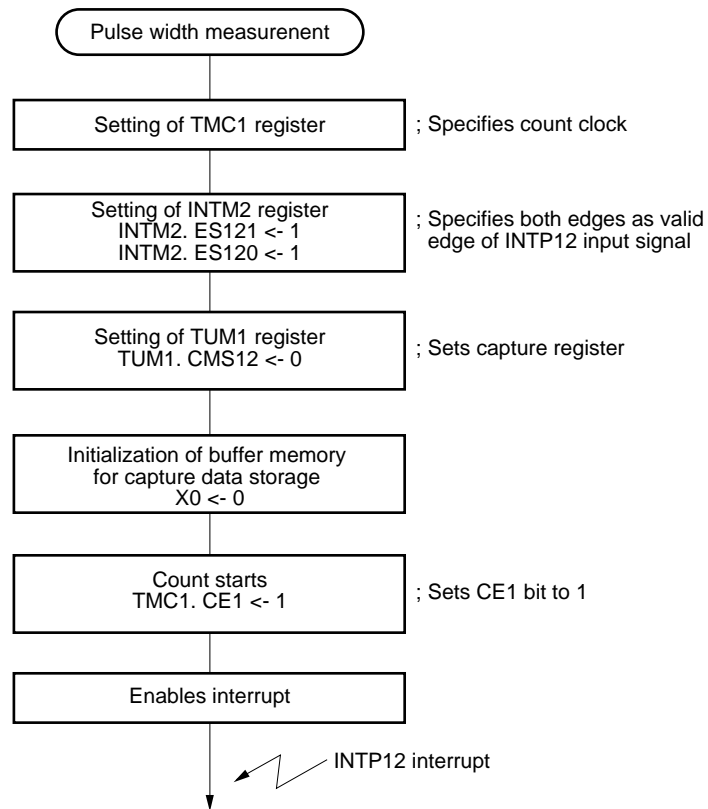
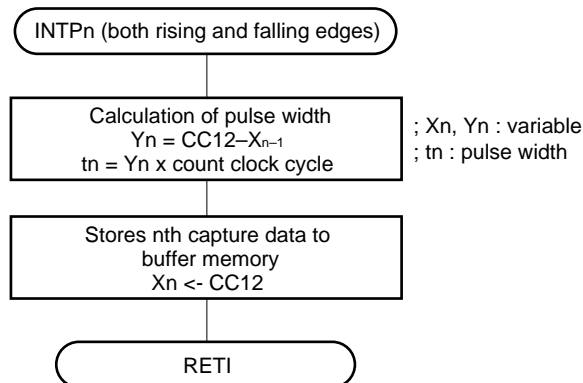


Figure 7-16. Interrupt Request Processing Routine Calculating Pulse Width (timer 1)



Caution If an overflow occurs two times or more between (n-1)th capture and nth capture, the pulse width cannot be measured.

(3) PWM output (timer 1)

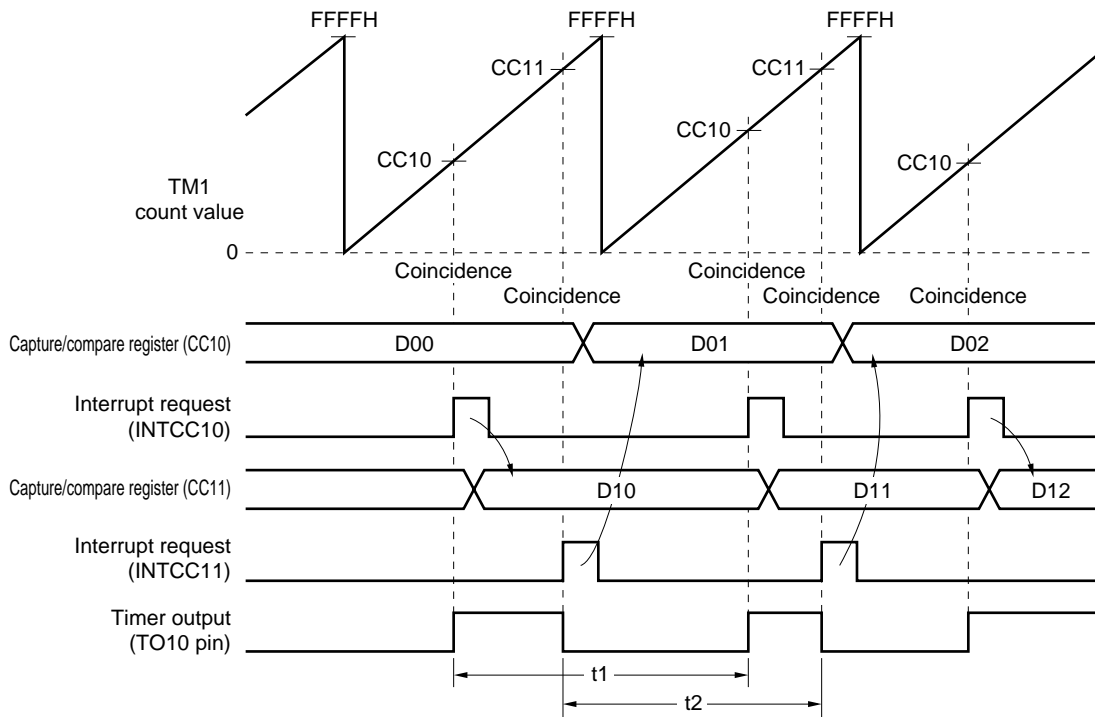
Any square wave can be output to timer output pins (TO10 and TO11) by combining the use of timer 1 and the timer output function.

(a) Using timer 1

- ★ Two capture/compare registers, CC10 and CC11, are used in this example of PWM output. A PWM signal with an accuracy of 16 bits can be output from the TO10 pin. Figure 7-17 shows the timing. When timer 1 is used as a 16-bit timer, the rising timing of the PWM output is determined by the value set to capture/compare register CC10, and the falling timing is determined by the value set to capture/compare register CC11.

Figure 7-18 shows the programming procedure at this time.

Figure 7-17. PWM Output Timing (TM1)



Remark Dxx: set value of compare register

- ★ $t1 = \{(10000H - D00) + D01\} \times \text{count clock cycle}$
- ★ $t2 = \{(10000H - D10) + D11\} \times \text{count clock cycle}$

Figure 7-18. Programming Procedure of PWM Output (timer 1)

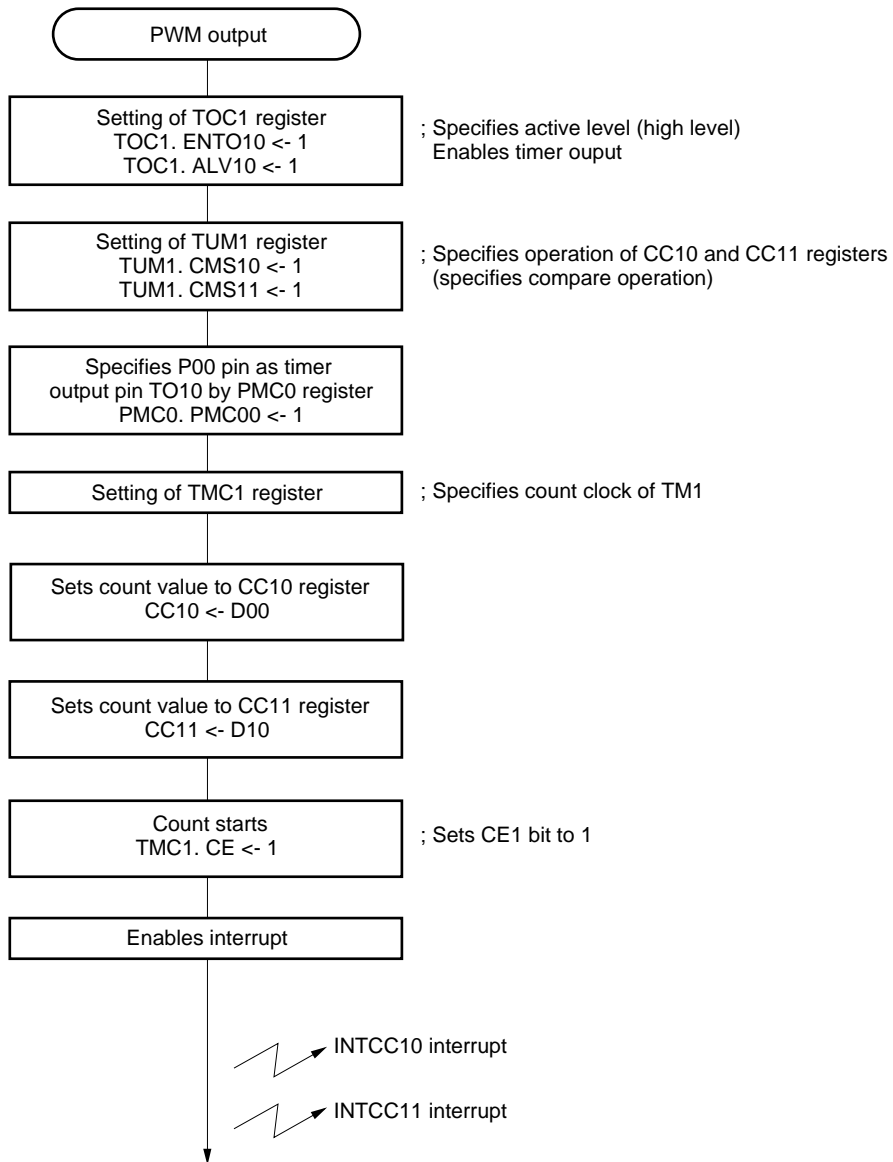
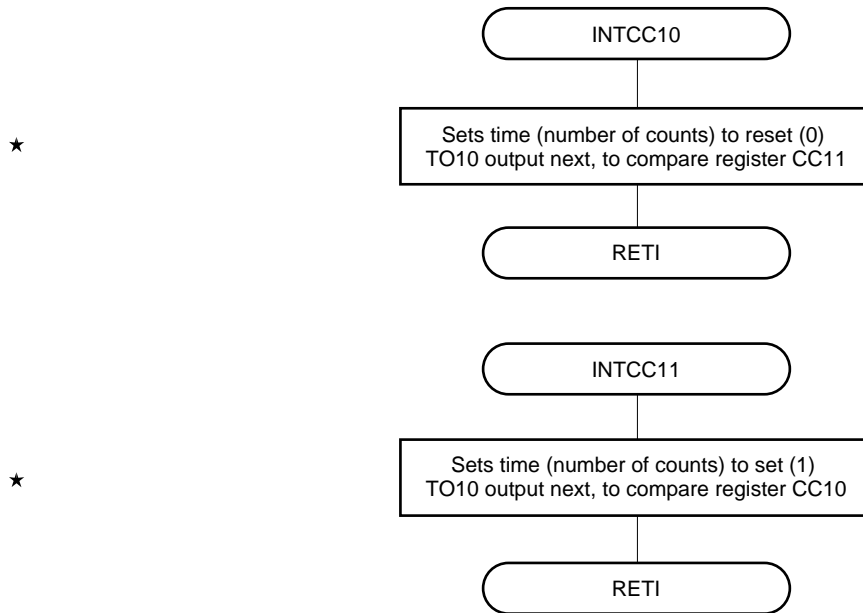


Figure 7-19. Interrupt Request Processing Routine, Modifying Compare Value (timer 1)

(4) Cycle measurement (timer 1)

Timer 1 can be used to measure the cycle or frequency of an external pulse input to the INTP_n pin (n = 10 to 13).

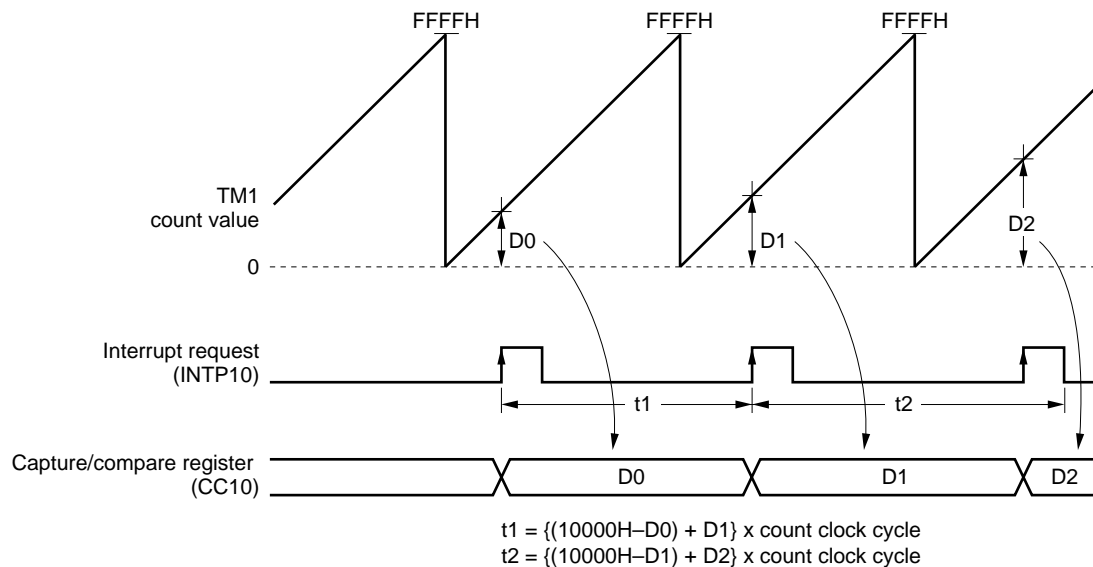
In this example, the cycle of the external pulse input to the INTP₁₀ pin is measured with a resolution of 16 bits, by combining the use of timer 1 and the capture/compare register CC₁₀.

The valid edge of the INTP₁₀ input signal is specified by the INTM₂ register to be the rising edge.

To calculate the cycle, the difference between the count value of TM1 captured to the CC₁₀ register at the *n*th rising edge (*D_n*), and the count value captured at the (*n*-1)th rising edge (*D_{n-1}*), is calculated, and the value multiplied by the count clock frequency.

Figure 7-21 shows the setting procedure at this time.

Figure 7-20. Cycle Measurement Timing (TM1)



Remark *D_n*: count value of TM1 (n = 0, 1, 2, ...)

Figure 7-21. Set-up Procedure for Cycle Measurement (timer 1)

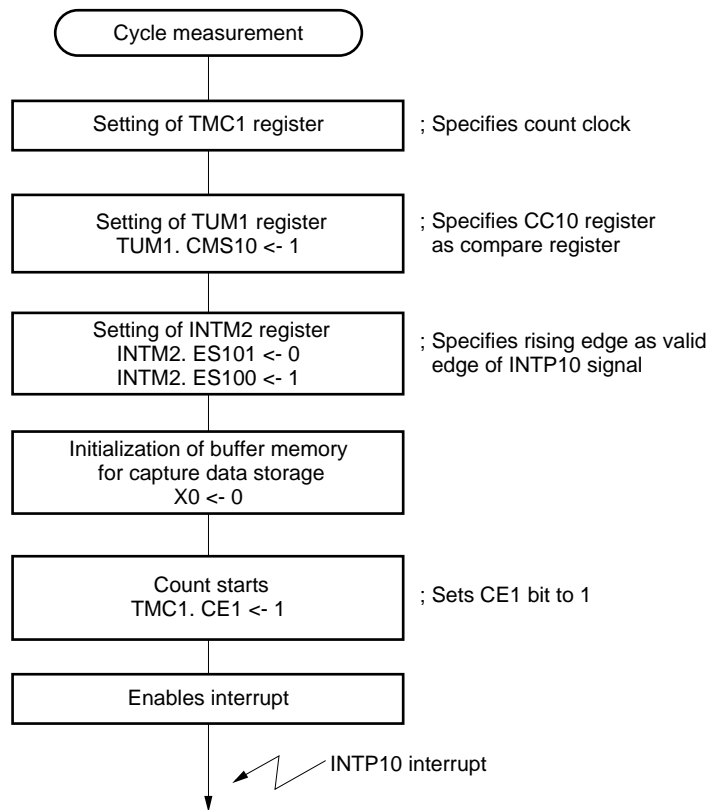
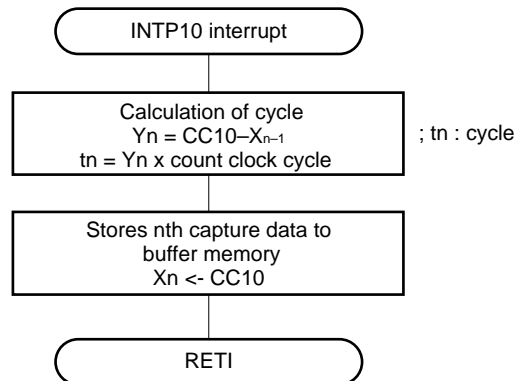


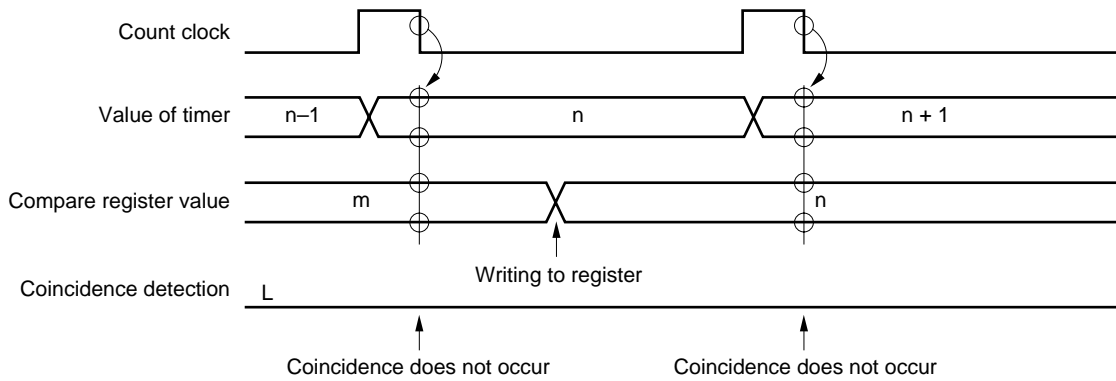
Figure 7-22. Interrupt Request Processing Routine Calculating Cycle (timer 1)



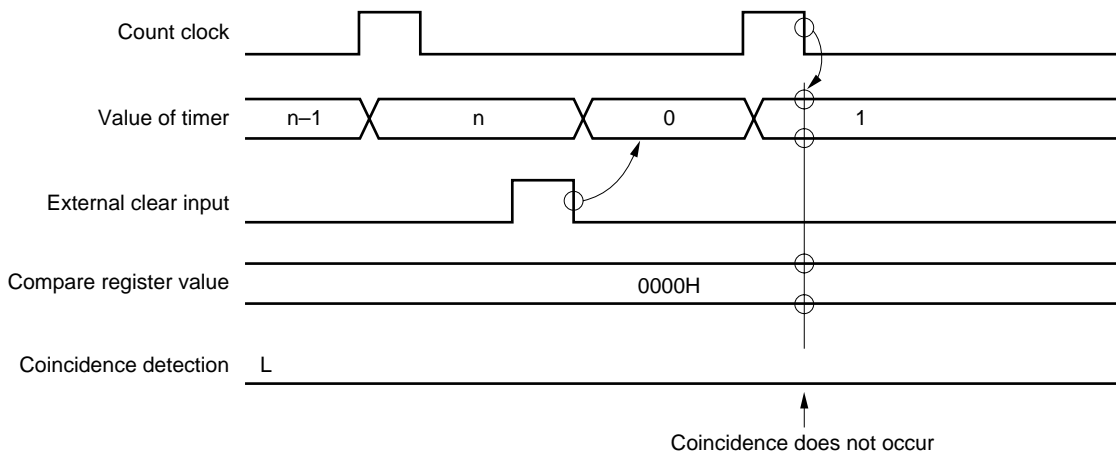
7.7 Note

Coincidence is detected by the compare register immediately after the timer value matches the compare register value, and does not take place in the following cases:

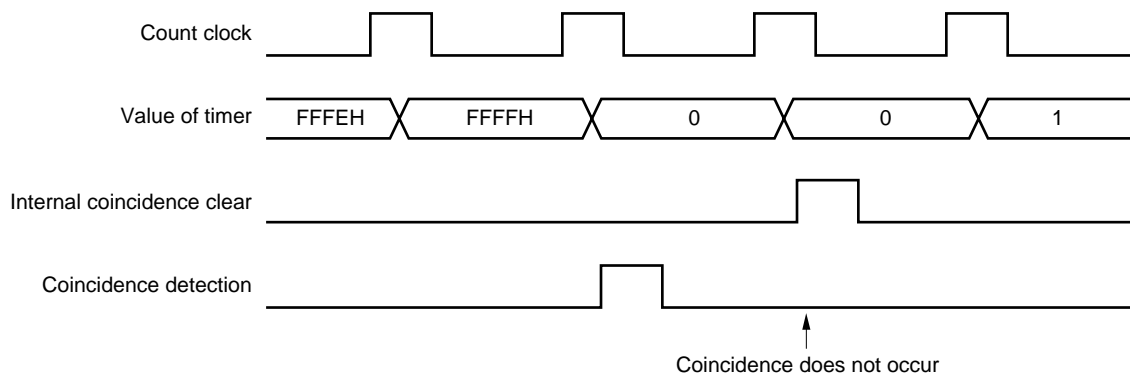
(1) When compare register is rewritten (TM1, TM4)



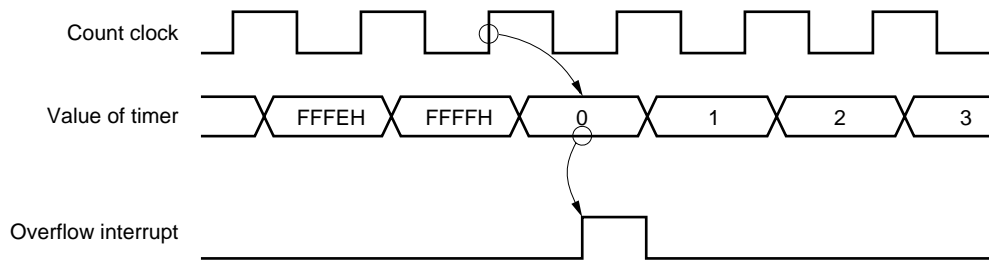
(2) When timer is cleared by external input (TM1)



(3) When timer is cleared (TM4)



When timer 1 is used as a free-running timer, the timer value is cleared to 0 when the timer overflows.



CHAPTER 8 SERIAL INTERFACE FUNCTION

8.1 Features

The V852 is provided with four transmission/reception channels for the serial interface function. There are the following two types of interfaces, and each of them functions independently.

- (1) Asynchronous serial interface (UART): 1 channel
- (2) Clocked serial interface (CSIn): 3 channels (n = 0 to 2)

The UART transmits/receives 1-byte serial data following a start bit and can perform full-duplex communication.

The CSI uses three signal lines to transfer data (3-wire serial I/O): the serial clock (\overline{SCKn}) (n = 0 to 2), serial input (SIn) (n = 0 to 2), and serial output (SOn) (n = 0 to 2) lines.

8.2 Asynchronous Serial Interface (UART)

8.2.1 Features

- Transfer rate: 110 bps to 38400 bps (with baud rate generator, at $\phi = 25$ MHz)
781 Kbps max. (with $\phi/2$, at $\phi = 25$ MHz)
- Full-duplex communication
- Two-pin configuration: TXD: transmit data output pin
RXD: receive data input pin
- Receive error detection function
 - Parity error
 - Framing error
 - Overrun error
- Three interrupt sources
 - Receive error interrupt (INTSER0)
 - Reception completion interrupt (INTSR0)
 - Transmission completion interrupt (INTST0)
- Character length of transmit/receive data is specified by ASIM00 and ASIM01 registers.
- Character length: 7, 8 bits
9 bits (when extended)
- Parity function: odd, even, 0, none
- Transmit stop bit: 1, 2 bits
- Internal baud rate generator

8.2.2 Configuration of asynchronous serial interface

The asynchronous serial interface is controlled by asynchronous serial interface mode register (ASIM) and asynchronous serial interface status register (ASIS). The receive data is stored in the receive buffer (RXB), and the transmit data is written to the transmit shift register (TXS).

Figure 8-1 shows the configuration of the asynchronous serial interface.

(1) Asynchronous serial interface mode registers (ASIM00, ASIM01)

ASIM00 and ASIM01 are 8-bit registers that specify the operation of the asynchronous serial interface.

(2) Asynchronous serial interface status register (ASIS0)

ASIS0 is a register containing flags that indicate receive errors, if any, and a transmit status flag. Each receive error flag is set to 1 when a receive error occurs, and is reset to 0 when data is read from the receive buffer (RXB0, RXB0L), or when new data is received (if the next data contains an error, the corresponding error flag is set).

The transmit status flag is set to 1 when transmission is started, and reset to 0 when transmission ends.

(3) Reception control parity check

The reception operation is controlled according to the contents programmed in the ASIM00 and ASIM01 registers. During the receive operation, errors such as parity error are also checked. If an error is found, the appropriate value is set to the ASIS0 register.

(4) Receive shift register

This shift register converts the serial data received on the RXD pin into parallel data. When it receives 1 byte of data, it transfers the receive data to the receive buffer.

The receive shift register cannot be accessed by the CPU.

(5) Receive buffer (RXB0, RXB0L)

RXB0 is a 9-bit buffer register that holds receive data. If data of 7 or 8 bits/character is received, 0 is stored to the most significant bit position of this register.

If this register is accessed in 16-bit units, RXB0 is specified. To access in lower 8-bit units, RXB0L is specified. While reception is enabled, the receive data is transferred from the receive shift register to the receive buffer in synchronization with shift-in processing of 1 frame.

When the data is transferred to the receive buffer, a reception completion interrupt request (INTSR0) occurs.

(6) Transmit shift register (TXS0, TXS0L)

TXS0 is a 9-bit shift register used for transmit operation. When data is written to this register, the transmission operation is started.

A transmission complete interrupt request (INTST0) is generated after each complete data frame is transmitted. When this register is accessed in 16-bit units, TXS0 is specified. To access in lower 8-bit units, TXS0L is specified.

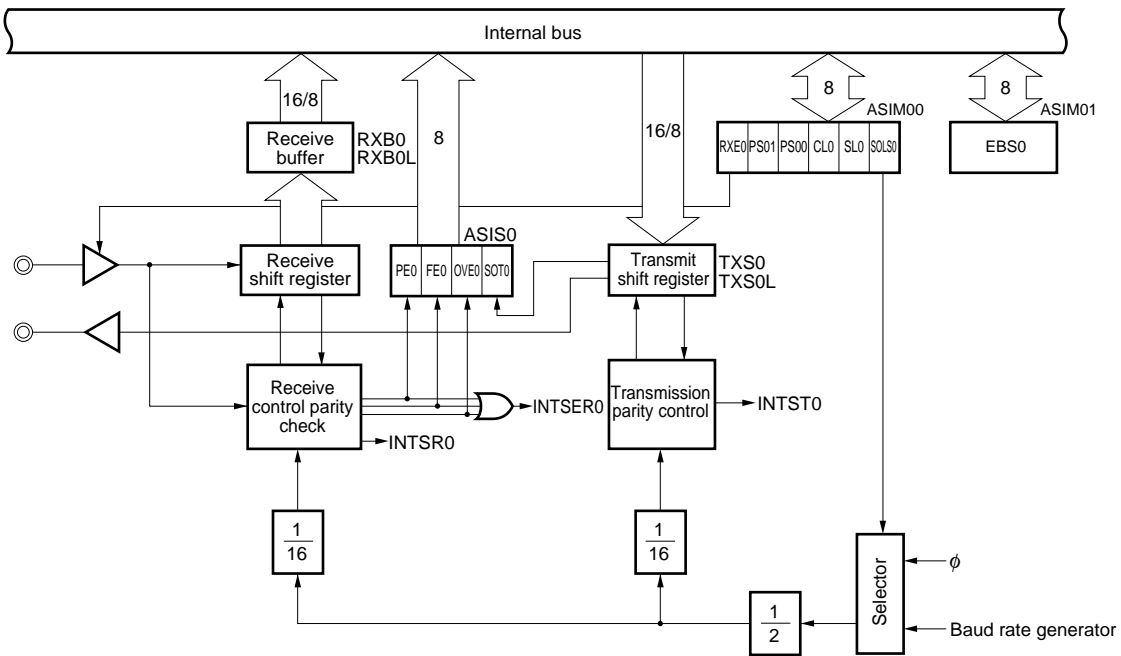
(7) Transmission parity control

A start bit, parity bit, and stop bit are appended to the data written to the TXS0 register, according to the contents programmed in the ASIM00 and ASIM01 registers, to control the transmission operation.

(8) Selector

Selects the source of the serial clock.

Figure 8-1. Block Diagram of Asynchronous Serial Interface



8.2.3 Mode registers and control registers

(1) Asynchronous serial interface mode registers (ASIM00 and ASIM01)

These registers specify the transfer mode of the UART.
They can be read/written in 8- or 1-bit units.

| | | | | | | | | | | |
|--------|---|------|------|------|-----|-----|---|-------|------------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| ASIM00 | 1 | RXE0 | PS01 | PS00 | CL0 | SL0 | 0 | SCLS0 | Address | At reset |
| | | | | | | | | | FFFFFF0C0H | 80H |

| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 6 | RXE0 | Receive Enable Enables/disables reception. 0: Disables reception 1: Enables reception When reception is disabled, the receive shift register does not detect the start bit. Data is not shifted into the receive shift register and neither is any transfer to the receive buffer performed. Therefore, the previous contents of receive buffer are retained. When reception is enabled, the data is shifted into the receive shift register and transferred to the receive buffer when one complete frame has been received. A reception completion interrupt (INTSR0) is generated in synchronization with the transfer to the receive buffer. |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | |
|--------------|------------|--|------|------|-----------|---|---|-----------------------------------|---|---|---|---|---|------------|---|---|-------------|
| 5, 4 | PS01, PS00 | <p>Parity Select Specifies parity bit.</p> <table border="1"> <thead> <tr> <th>PS01</th> <th>PS00</th> <th>Operation</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>No parity. Extended bit operation</td> </tr> <tr> <td>0</td> <td>1</td> <td>0 parity Transmission side -> Transmits with parity bit 0 Reception side -> Does not generate parity error on reception</td> </tr> <tr> <td>1</td> <td>0</td> <td>Odd parity</td> </tr> <tr> <td>1</td> <td>1</td> <td>Even parity</td> </tr> </tbody> </table> <ul style="list-style-type: none"> Even parity Parity bit is set to "1" when number of bits equal to one in received data is odd. If number of bits that are one is even, parity bit is cleared to 0. In this way, number of bits that are "1" in transmit data and parity bit is controlled to become even. During reception, number of bits that are "1" in receive data and parity bit are counted. If it is odd, parity error occurs. Odd parity In contrast to even parity, number of bits included in transmit data and parity bit that are "1" is controlled to become odd. During reception, parity error occurs if the number of "1"s in the receive data and parity bit are added up to become even. 0 parity Parity bit is cleared to "0" during transmission, regardless of transmit data. During reception, the parity bit is not checked. Therefore, parity error does not occur regardless of whether parity bit is "0" or "1". No parity No parity bit is appended to the transmit data. Reception is performed on assumption that there is no parity bit. Because no parity bit is used, parity error does not occur. Extended bit operation can be specified by EBS0 bit of ASIM01 register. | PS01 | PS00 | Operation | 0 | 0 | No parity. Extended bit operation | 0 | 1 | 0 parity Transmission side -> Transmits with parity bit 0 Reception side -> Does not generate parity error on reception | 1 | 0 | Odd parity | 1 | 1 | Even parity |
| PS01 | PS00 | Operation | | | | | | | | | | | | | | | |
| 0 | 0 | No parity. Extended bit operation | | | | | | | | | | | | | | | |
| 0 | 1 | 0 parity Transmission side -> Transmits with parity bit 0 Reception side -> Does not generate parity error on reception | | | | | | | | | | | | | | | |
| 1 | 0 | Odd parity | | | | | | | | | | | | | | | |
| 1 | 1 | Even parity | | | | | | | | | | | | | | | |
| 3 | CL0 | <p>Character Length Specifies character length of one frame.</p> <p>0: 7 bits 1: 8 bits</p> | | | | | | | | | | | | | | | |
| 2 | SL0 | <p>Stop Bit Length Specifies stop bit.</p> <p>0: 1 bit 1: 2 bits</p> | | | | | | | | | | | | | | | |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | |
|--------------|----------|--|--------|----------|--------|--------|----------|--------|-------|-------|-----------|-------|-------|-------|-------|-------|-------|-------|
| 0 | SCLS0 | <p>Serial Clock Source Specifies serial clock.</p> <p>0: Specified by baud rate generator and BPRM0 (baud rate generator prescaler mode register) 1: $\phi/2$</p> <ul style="list-style-type: none"> When SCLS0 = 1 $\phi/2$ (system clock) is selected as serial clock source. In asynchronous mode, baud rate is expressed as follows because sampling rate of x16 is used: $\text{Baud rate} = \frac{\phi/2}{16} \text{ bps}$ Value of baud rate when typical clock is used based on above expression is as follows: <table border="1"> <thead> <tr> <th>ϕ</th> <th>25 MHz</th> <th>20 MHz</th> <th>16 MHz</th> <th>12.5 MHz</th> <th>10 MHz</th> <th>8 MHz</th> <th>5 MHz</th> </tr> </thead> <tbody> <tr> <td>Baud rate</td> <td>781 K</td> <td>625 K</td> <td>500 K</td> <td>390 K</td> <td>312 K</td> <td>250 K</td> <td>156 K</td> </tr> </tbody> </table> <ul style="list-style-type: none"> When SCLS0 = 0 Baud rate generator output is selected as serial clock source. For details of baud rate generator, refer to 8.4 "Baud Rate Generator (BRG)". | ϕ | 25 MHz | 20 MHz | 16 MHz | 12.5 MHz | 10 MHz | 8 MHz | 5 MHz | Baud rate | 781 K | 625 K | 500 K | 390 K | 312 K | 250 K | 156 K |
| ϕ | 25 MHz | 20 MHz | 16 MHz | 12.5 MHz | 10 MHz | 8 MHz | 5 MHz | | | | | | | | | | | |
| Baud rate | 781 K | 625 K | 500 K | 390 K | 312 K | 250 K | 156 K | | | | | | | | | | | |

Caution The operation of UART is not guaranteed if the bits 0 to 6 of this register are changed while UART is transmitting/receiving data.

| | | | | | | | | | | |
|--------|---|---|---|---|---|---|---|------|-----------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| ASIM01 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EBS0 | Address | At reset |
| | | | | | | | | | FFFFF0C2H | 00H |

| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 0 | EBS0 | <p>Extended Bit Select Specifies extended bit operation of transmit/receive data when no parity is specified (PS01, PS00 = 00).</p> <p>0: Disables extended bit operation 1: Enables extended bit operation</p> <p>When extended bit operation is enabled, 1 data bit is appended as most significant bit to 8-bit transmit/receive data, and therefore 9-bit data is communicated.</p> <p>Extended bit operation is valid only when no parity is specified by ASIM00 register. If zero, even, or odd parity is specified, specification by EBS0 bit is invalid, and extended bit is not appended.</p> |

(2) Asynchronous serial interface status register 0 (ASIS0)

This register contains three error flags that indicate the receive error status for each character received and the status of the transmit shift register.

The error flags always indicate the status of an error that has occurred most recently. If two or more errors occur before the current received data, only the status of the error that has occurred last is retained.

If a receive error occurs, read the receive buffer RXB0 or RXB0L after reading the ASIS0 register, and then clear the error flag.

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

| | | | | | | | | | | |
|-------|------|---|---|---|---|-----|-----|------|-----------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| ASIS0 | SOT0 | 0 | 0 | 0 | 0 | PE0 | FE0 | OVE0 | Address | At reset |
| | | | | | | | | | FFFFF0C4H | 00H |

| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 7 | SOT0 | Status Of Transmission Status flag that indicates transmission operation status. Set (1) : Beginning of transmission of a data frame (writing to TXS register) Clear (0): End of transmission of a data frame (occurrence of INTST0) When serial data transfer begins, this flag will indicate if the transmit shift register is ready to be written or not. |
| 2 | PE0 | Parity Error Status flag that indicates parity error. Set (1) : Transmit parity and receive parity do not match Clear (0): The data is read from the receive buffer. |
| 1 | FE0 | Framing Error Status flag that indicates framing error. Set (1) : Stop bit is not detected Clear (0): The data is read from the receive buffer. |
| 0 | OVE0 | Overrun Error Status flag that indicates overrun error. Set (1) : The UART completes the next receive process before taking the receive data from the receive buffer. Clear (0): The data is read from the receive buffer. Because contents of receive shift register are transferred to receive buffer each time one frame of data has been received, if overrun error occurs, next receive data is written over contents of receive buffer, and previous receive data is discarded. |

(3) Receive buffers (RXB0 and RXB0L)

RXB0 is a 9-bit buffer register that holds the receive data. When 7- or 8-bit/character is received, the higher bit of this register is 0.

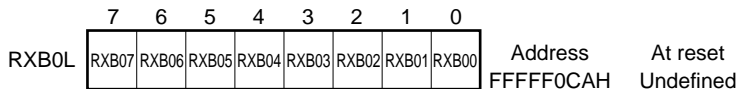
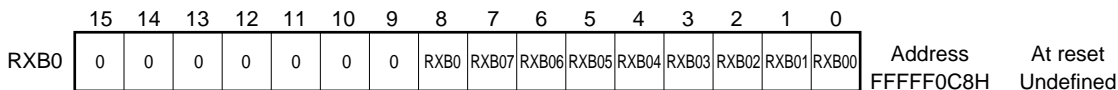
When this register is accessed in 16-bit units, RXB0 is specified. To access in lower 8-bit units, RXB0L is specified.

When reception is enabled, the receive data is transferred from the receive shift register to the receive buffer when one complete frame of data (or character) has been received.

When the receive data is transferred to the receive buffer, a reception completion interrupt request (INTSR0) occurs.

When reception is disabled, the data is not shifted into the receive shift register and the reception completion interrupt is not generated. The previous contents of the receive buffer are retained.

RXB0 enables 16-bit read access only, and RXB0L enables 8-/1-bit read access only.



| Bit Position | Bit Name | Function |
|--------------|-----------------------|--|
| 8 | RXEBO | Receive Extended Buffer Extended bit when 9-bit/character is received. This bit is cleared to zero when 7- or 8-bit/character is received. |
| 7 to 0 | RXB0n (n = 7 to 0) | Receive Buffer These bits store receive data. The RXB07 bit is cleared to zero when 7-bit/character is received. |

(4) Transmit shift registers (TXS0, TXS0L)

TXS0 is a 9-bit shift register for data transmission. The transmit operation is started when data is written to this register.

Transmission complete interrupt request (INTST0) is generated after each complete data frame is transmitted. When this register is accessed in 16-bit units, TXS0 is specified. To access in lower 8-bit units, TXS0L is specified.

TXS0 enables 16-bit write access only, and TXS0L enables 8-bit write access only.

| | | | | | | | | | | | | | | | | | | |
|------|----|----|----|----|----|----|---|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----------|-----------|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TXS0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TXED0 | TXS07 | TXS06 | TXS05 | TXS04 | TXS03 | TXS02 | TXS01 | TXS00 | Address | At reset |
| | | | | | | | | | | | | | | | | | FFFFF0CCH | Undefined |

| | | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----------|-----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| TXS0L | TXS07 | TXS06 | TXS05 | TXS04 | TXS03 | TXS02 | TXS01 | TXS00 | Address | At reset |
| | | | | | | | | | FFFFF0CEH | Undefined |

| Bit Position | Bit Name | Function |
|--------------|-----------------------|---|
| 8 | TXED0 | Transmit Extended Data Extended bit on transmission of 9-bit/character |
| 7 to 0 | TXS0n (n = 7 to 0) | Transmit Shifter Writes transmit data. |

Caution As the UART of the V852 does not have a transmit buffer, an interrupt request synchronizing with the completion of the transmission of one frame of data is generated instead of the interrupt request generated at the end of transmission (completion of transfer to buffer).

8.2.4 Interrupt request

UART generates the following three types of interrupt requests:

- Receive error interrupt
- Reception completion interrupt
- Transmission completion interrupt

Of these three, the receive error interrupt has the highest default priority, followed by the reception completion interrupt and transmission completion interrupt.

Table 8-1. Default Priority of Interrupts

| Interrupt | Priority |
|-------------------------|----------|
| Receive error | 1 |
| Reception completion | 2 |
| Transmission completion | 3 |

(1) Receive error interrupt (INTSER0)

A receive error interrupt occurs as a result of ORing the three types of receive errors described in description of the ASIS0 register when reception is enabled.

This interrupt does not occur when reception is disabled.

(2) Reception completion interrupt (INTSR0)

The reception completion interrupt occurs if data is received in the receive shift register and then transferred to the receive buffer when reception is enabled.

This interrupt also occurs when a receive error occurs, but the receive error interrupt has the higher priority.

The reception completion interrupt does not occur when reception is disabled.

(3) Transmission completion interrupt (INTST0)

Because the UART of the V852 does not have a transmit buffer, a transmission completion interrupt occurs when one frame of transmit data containing a 7-/8-/9-bit character is shifted out from the transmit shift register.

The transmission completion interrupt is output when the last bit of data has been transmitted.

8.2.5 Operation

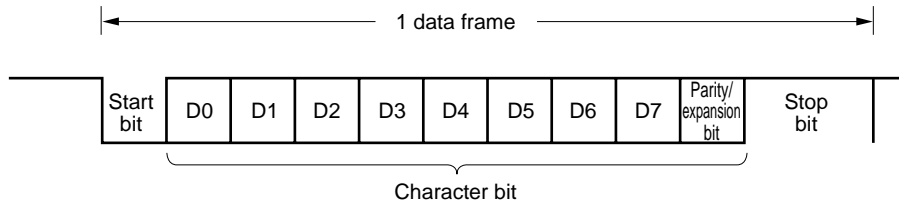
(1) Data format

Full-duplex serial data is transmitted/received.

One data frame of the transmit/receive data consists of a start bit, character bits, parity bit, and stop bit, as shown in Figure 8-2.

The length of the character bit, parity, and the length of the stop bit in one data frame are specified by the asynchronous serial interface mode registers (ASIM00 and ASIM01).

Figure 8-2. Format of Transmit/Receive Data of Asynchronous Serial Interface



- Start bit 1 bit
- Character bit 7/8/9 bits (with extended bit)
- Parity/expansion bit Even/odd/0/none/expansion bit
- Stop bit 1/2 bits

(2) Transmission

Transmission is started when data is written to the transmit shift register (TXS0 or TXS0L). The next data is written to the TXS0 or TXS0L register by the processing routine of the transmission completion interrupt (INTST0).

(a) Transmission enabled status

The UART of the V852 is always enabled to transmit data. Because the V852 does not have a pin that inputs a transmit enable signal, a general input port is used when it is necessary to check whether the other party is ready to receive data.

(b) Starting transmission

Transmission is started by writing data to the transmit shift register (TXS0, TXS0L). The transmit data is transferred starting from the start bit with the LSB first. The start bit, parity/expansion bit, and stop bit are automatically appended.

(c) Transmission interrupt request

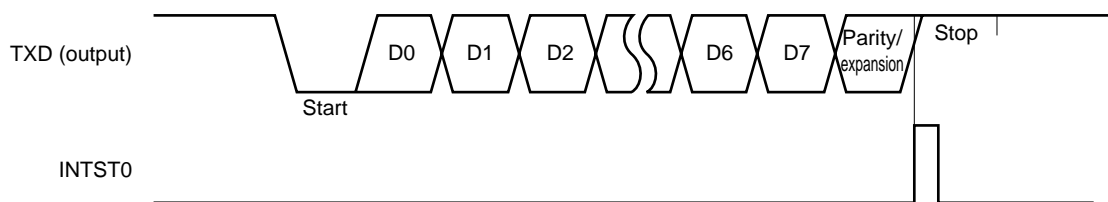
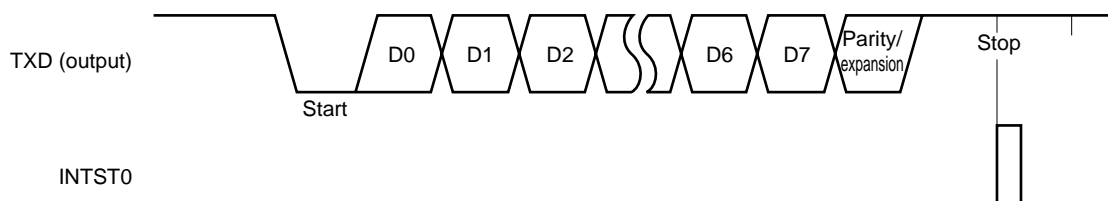
When one frame of data or character has been completely transferred, a transmission completion interrupt request (INTST0) occurs.

Unless the data to be transmitted next is written to the TXS0 or TXS0L register, the transmit operation is aborted.

The communication rate drops unless the next transmit data is written to the TXS0 or TXS0L register immediately after transmission has been completed.

- Cautions**
1. The transmission completion interrupt request (INTST0) is generated after each complete data frame is transmitted out of the transmit shift register. It is not generated by the empty state of TXS0 or TXS0L. Because of this, the INTST0 interrupt will not be generated immediately after reset.
 2. During the transmit operation, writing data into the TXS0 or TXS0L register is ignored (the data is discarded) until INTST0 is generated.

Figure 8-3. Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1**(b) Stop bit length: 2**

(3) Reception

When reception is enabled, sampling of the RXD pin is started, and reception of data begins when the start bit is detected. Each time one frame of data or character has been received, the reception completion interrupt (INTSR0) occurs. Usually, the receive data is transferred from the receive buffer (RXB0, RXB0L) to memory by this interrupt processing.

(a) Reception enabled status

Reception is enabled when the RXE0 bit of the ASIM00 register is set to 1.

RXE0 = 1: Reception is enabled

RXE0 = 0: Reception is disabled

When reception is disabled, the receive hardware stands by in the initial status.

At this time, the reception completion interrupt/receive error interrupt does not occur, and the contents of the receive buffer are retained.

(b) Starting reception

Reception is started when the start bit is detected.

The RXD pin is sampled with the serial clock specified by the ASIM00 register. The RXD pin is sampled again eight clocks after the falling edge of the RXD pin has been detected. If the RXD pin is low at this time, it is recognized as the start bit, and reception is started. After that, the RXD pin is sampled in 16 clock ticks.

If the RXD pin is high eight clocks after the falling edge of the RXD pin has been detected, this falling edge is not recognized as the start bit. The serial clock counter is reinitialized, and the UART waits for the input of the next falling edge or valid start bit.

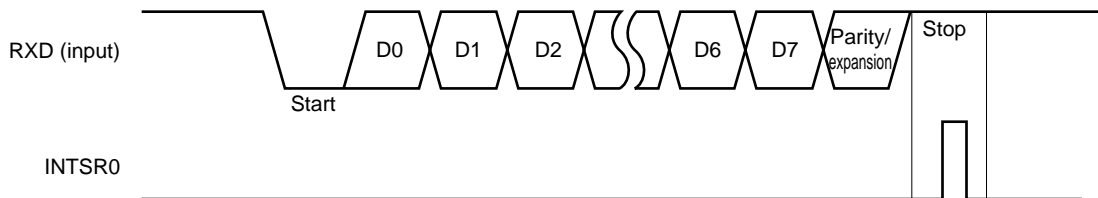
(c) Reception completion interrupt request

When one frame of data has been received with RXE0 = 1, the receive data in the shift register is transferred to RXB0, and a reception completion interrupt request (INTSR0) is generated.

If an error occurs, the receive data that contains an error is transferred to the receive buffer (RXB0, RXB0L), and the transmission completion interrupt (INTSR0) and receive error interrupt (INTSER0) occur simultaneously.

When the RXE0 bit is reset to 0 during reception, the receive operation is immediately disabled. The contents of the receive buffer (RXB0, RXB0L) and asynchronous serial interface status register (ASIS0) are not changed, and the reception completion interrupt (INTSR0) and receive error interrupt (INTSER0) will not be generated.

Figure 8-4. Asynchronous Serial Interface Reception Completion Interrupt Timing



(d) Reception error flag

Three error flags, parity error, framing error, and overrun error flags, are related with the reception operation.

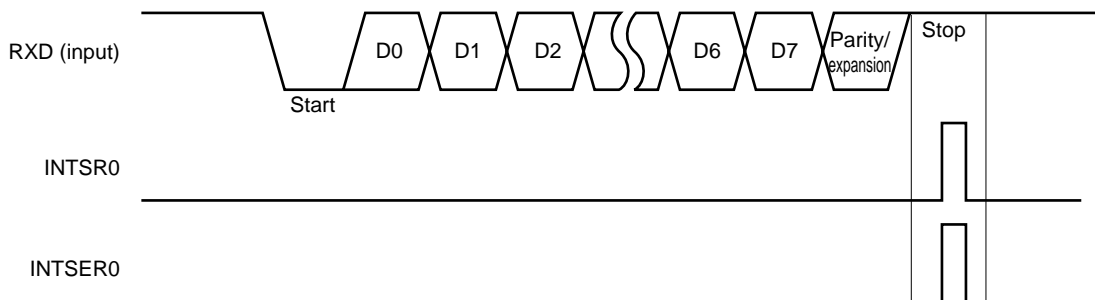
The receive error interrupt request occurs as a result of ORing these three error flags.

By reading the contents of the ASIS0 register, the error which caused the receive error interrupt (INTSER0) can be identified.

The contents of the ASIS0 register are reset to 0 when the receive buffer (RXB0, RXB0L) is read or the next data frame is received (if the next data contains an error, the corresponding error flag is set).

| Receive | Error Cause |
|---------------|--|
| Parity Error | Parity specified during transmission does not coincide with parity of receive data |
| Framing error | Stop bit is not detected |
| Overrun error | Next data is completely received before data is read from receive buffer |

Figure 8-5. Receive Error Timing



8.3 Clocked Serial Interface 0 to 2 (CSI0 to CSI2)

8.3.1 Features

- Number of channels: 3 channels (CSIn) (n = 0 to 2)
- High transfer speed: 6.25 Mbps max. (with $\phi/2$, at $\phi = 25$ MHz)
- Half duplex communication
- Character length: 8 bits
- MSB first/LSB first selectable
- External serial clock input/internal serial clock output selectable
- 3 lines: SOn : serial data output (n = 0 to 2)
 SIn : serial data input (n = 0 to 2)
 SCKn: serial clock I/O (n = 0 to 2)
- Interrupt source: 3
 - Interrupt request signal (INTCSIn) (n = 0 to 2)

The CSIn is controlled by the clocked serial interface mode register (CSIMn) (n = 0 to 2). The transmit/receive data is read/written from/to the serial I/O shift register (SIO n) (n = 0 to 2).

(1) Clocked serial interface mode register (CSIMn)

CSIMn is an 8-bit register that specifies the operation of the clocked serial interface.

(2) Serial I/O shift register (SIO n)

SIO n is an 8-bit register that converts serial data into parallel data, and vice versa. SIO n is used for both transmission and reception.

Data is shifted in (received) or shifted out (transmitted) from the MSB or LSB side.

The actual transmitting and receiving of data is actually performed by writing data to and reading data from the SIO n.

(3) Serial clock selector

Selects the serial clock to be used.

(4) Serial clock control circuit

Controls supply of the serial clock to the SIO n. When the internal clock is used, it also controls the clock output to the SCKn (n = 0 to 2) pin.

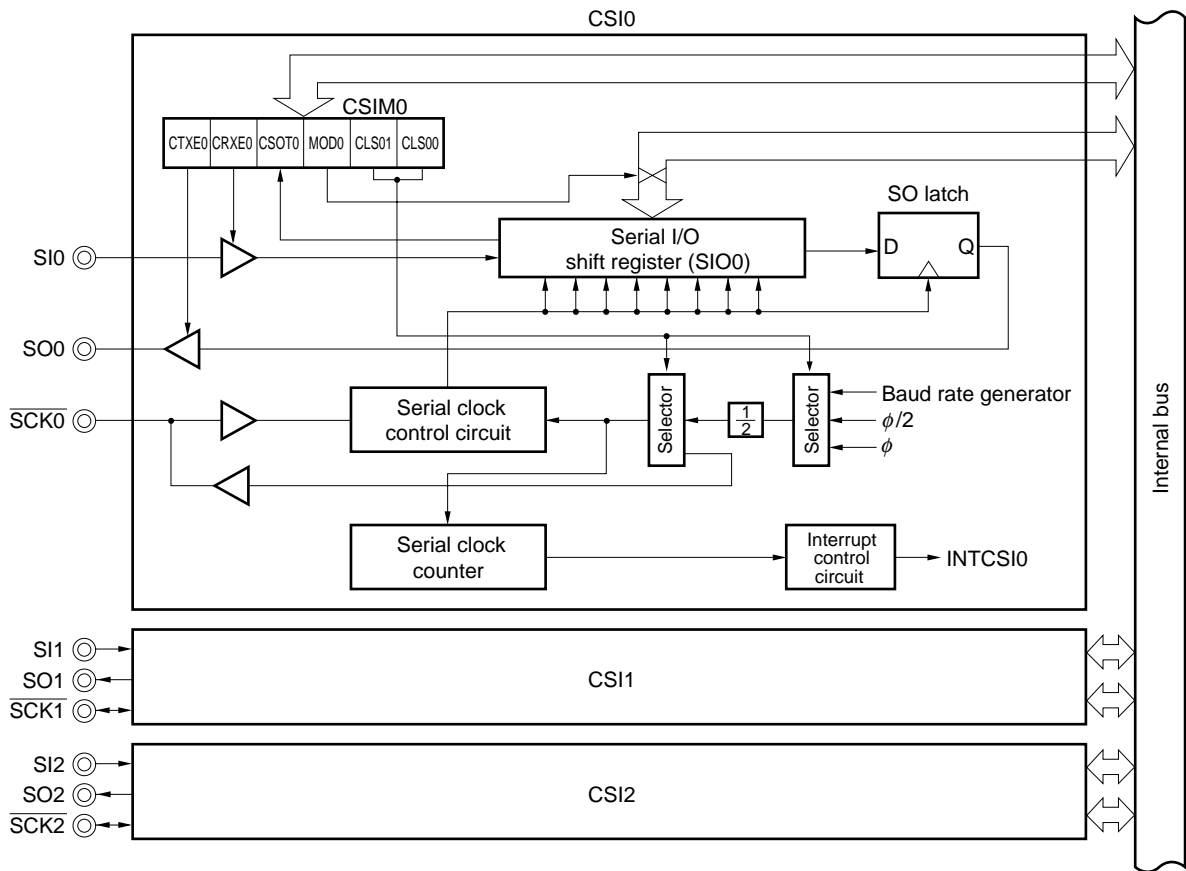
(5) Serial clock counter

Counts the serial clocks being output and the serial clocks received during transmission/reception to check whether 8-bit data has been transmitted or received.

(6) Interrupt signal generation control circuit

Controls whether an interrupt request is generated when the serial clock counter has counted eight serial clocks.

8.3.2 Configuration



8.3.3 Mode registers and control registers

(1) Clocked serial interface mode register n (CSIMn) (n = 0 to 2)

This register specifies the basic operation mode of CSIn.

It can be read/written in 8- or 1-bit units (note, however, that bit 5 can only be read).

| | | | | | | | | | | |
|-------|-------|-------|-------|---|---|------|-------|-------|----------------------|-----------------|
| CSIM0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFF088H | At reset 00H |
| | CTXE0 | CRXE0 | CSOT0 | 0 | 0 | MOD0 | CLS01 | CLS00 | | |
| CSIM1 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFF098H | At reset 00H |
| | CTXE1 | CRXE1 | CSOT1 | 0 | 0 | MOD1 | CLS11 | CLS10 | | |
| CSIM2 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFF0A8H | At reset 00H |
| | CTXE2 | CRXE2 | CSOT2 | 0 | 0 | MOD2 | CLS21 | CLS20 | | |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | | | |
|--------------|--------------|--|--|--------|------------------------|----------|---|---|----------------|-------|---|---|----------------|--|---|---|---------------------------|--------|---|---|---------------------------|--------|
| 7 | CTXEn | CSI Transmit Enable Enables or disables transmission. 0: Disables transmission 1: Enables transmission When CTXEn = "0", output buffers of both SO and SI pins go into high-impedance state. | | | | | | | | | | | | | | | | | | | | |
| 6 | CRXEn | CSI Receive Enable Disables or enables reception. 0: Disables reception 1: Enables reception If serial clock is received when transmission enabled (CTXEn = 1) and reception are disabled, "0" is input to SIO _n . When reception is disabled (CRXEn = 0) during reception, the contents of SIO _n become undefined. | | | | | | | | | | | | | | | | | | | | |
| 5 | CSOTn | CSI Status Of Transmission Indicates that transfer operation is in progress. Set (1): Transfer start timing (writing to SIO0 register) Clear (0): Transfer end timing (INTCSI occurs) This bit is used to check whether writing to SIO _n is permitted or not. Serial data transfer is started by enabling transmission (CTXEn = 1). | | | | | | | | | | | | | | | | | | | | |
| 2 | MODn | Mode Specifies first bit. 0: MSB first 1: LSB first | | | | | | | | | | | | | | | | | | | | |
| 1, 0 | CLSn1, CLSn0 | Clock Source Specifies serial clock. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>CLSn1</th> <th>CLSn0</th> <th>Specifies serial clock</th> <th>SCKn pin</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>External clock</td> <td>Input</td> </tr> <tr> <td>0</td> <td>1</td> <td rowspan="3">Internal clock</td> <td>Specified by BPRM_m register^{Note1}</td> </tr> <tr> <td>1</td> <td>0</td> <td>$\phi/4$^{Note2}</td> <td>Output</td> </tr> <tr> <td>1</td> <td>1</td> <td>$\phi/2$^{Note2}</td> <td>Output</td> </tr> </tbody> </table> <p>Notes 1. For setting of BPRM_m (m = 0, 1) register, refer to section 8.4 "Baud Rate Generator (BRG)". 2. $\phi/4$ and $\phi/2$ indicate divider signal (ϕ = system clock).</p> | CLSn1 | CLSn0 | Specifies serial clock | SCKn pin | 0 | 0 | External clock | Input | 0 | 1 | Internal clock | Specified by BPRM _m register ^{Note1} | 1 | 0 | $\phi/4$ ^{Note2} | Output | 1 | 1 | $\phi/2$ ^{Note2} | Output |
| CLSn1 | CLSn0 | Specifies serial clock | SCKn pin | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | External clock | Input | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | Internal clock | Specified by BPRM _m register ^{Note1} | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | | $\phi/4$ ^{Note2} | Output | | | | | | | | | | | | | | | | | | |
| 1 | 1 | | $\phi/2$ ^{Note2} | Output | | | | | | | | | | | | | | | | | | |

★ Remark n = 0 to 2

(2) Serial I/O shift register n (SIO_n) (n = 0 to 2)

This register converts 8-bit serial data into parallel data, and vice versa. The actual transmitting and receiving of data is performed by writing data to and reading data from the SIO_n.

A shift operation of SIO_n is performed when CTXEn (n = 0 to 2) = "1" or CRXEn (n = 0 to 2) = "1".

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

| | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-----------------------|-----------------------|
| SIO0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFFF08AH | At reset Undefined |
| | SIO07 | SIO06 | SIO05 | SIO04 | SIO03 | SIO02 | SIO01 | SIO00 | | |
| SIO1 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFFF09AH | At reset Undefined |
| | SIO17 | SIO16 | SIO15 | SIO14 | SIO13 | SIO12 | SIO11 | SIO10 | | |
| SIO2 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address FFFFFF0AAH | At reset Undefined |
| | SIO27 | SIO26 | SIO25 | SIO24 | SIO23 | SIO22 | SIO21 | SIO20 | | |

| Bit Position | Bit Name | Function |
|--------------|--------------------|--|
| 7 to 0 | SIO _n m | Serial I/O Data is shifted in (received) or out (transmitted) from MSB or LSB side. |

Remark n = 0 to 2, m = 0 to 7

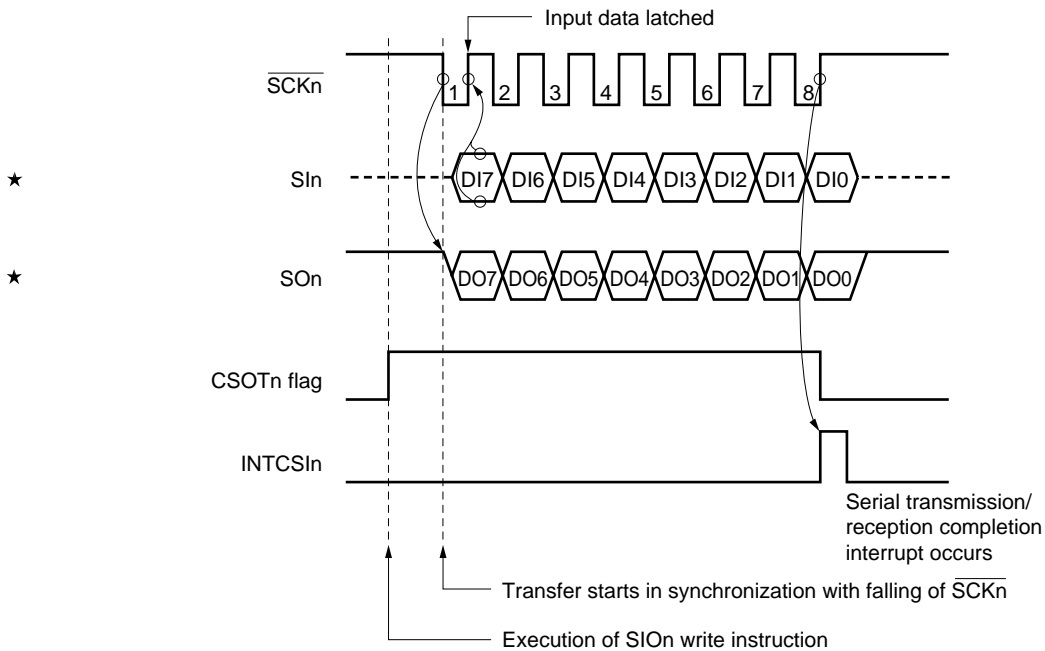
8.3.4 Basic operation

(1) Transfer format

The CSIn ($n = 0$ to 2) of the V852 performs interfacing by using three lines: one clock line and two data lines. Serial transfer is started by executing an instruction that writes transfer data to the SIO n ($n = 0$ to 2) register. During transmission, the data is output from the SO n ($n = 0$ to 2) pin in synchronization with the falling edge of $\overline{\text{SCK}}_n$ ($n = 0$ to 2).

During reception, the data input to the SIn ($n = 0$ to 2) pin is latched in synchronization with the rising edge of $\overline{\text{SCK}}_n$ ($n = 0$ to 2). $\overline{\text{SCK}}_n$ stops when the serial clock counter overflows (at the rising of the 8th count), and $\overline{\text{SCK}}_n$ remains high until the next data transmission or reception is started. At the same time, an interrupt request signal INTCSIn ($n = 0$ to 2) is generated.

Caution If CTXEn ($n = 0$ to 2) is changed from 0 to 1 after the transmit data is sent to the SIO n register, serial transfer will not begin.



Remark $n = 0$ to 2

(2) Enabling transmission/reception

The CSIn (n = 0 to 2) of the V852 has only one 8-bit shift register and does not have a buffer. Transmission and reception are therefore performed simultaneously.

(a) Transmission/reception enabling condition

The CSIn transmission/reception enabling conditions are specified using the CTXEn (n = 0 to 2) and CRXEn (n = 0 to 2) bits of the CSIMn (n = 0 to 2) register.

| CTXEn | CRXEn | Transmission/Reception |
|-------|-------|---------------------------------|
| 0 | 0 | Disables transmission/reception |
| 0 | 1 | Enables reception |
| 1 | 0 | Enables transmission |
| 1 | 1 | Enables transmission/reception |

Remark n = 0 to 2

(i) Disabling SIO_n output by CTXEn

When CTXEn = 0, the SIO_n pin output of CSIn goes into a high-impedance state.

When CTXEn = 1, the data of the SIO_n register of CSIn is output.

(ii) Disabling SIO_n input by CRXEn

When CRXEn = 0, the SIO_n register input of CSIn is "0".

When CRXEn = 1, the SIO_n pin input of CSIn is input to the shift register.

(iii) To check transmit data

To receive the transmit data and to check whether bus contention occurs, set CTXEn and CRXEn to 1.

(b) Starting transmission/reception

Transmission/reception is started by reading/writing the SIO_n register. Transmission/reception is controlled by setting the transmission enable bit (CTXEn) and reception enable bit (CRXEn) as follows:

| CTXEn | CRXEn | Start Condition |
|-------|--------|-----------------------------|
| 0 | 0 | Does not start |
| 0 | 1 | Reads from SIO _n |
| 1 | 0 | Writes to SIO _n |
| 1 | 1 | Writes to SIO _n |
| 0 | 0 -> 1 | Rewrites CRXEn bit |

Remark n = 0 to 2

If CTXEn bit is not changed from 0 to 1 before reading data from or writing data to the SIO_n register, transfer will not begin. The bottom of the table means that, if the CRXEn bit is changed from 0 to 1 when the CTXEn bit is "0", the serial clock will be generated to initiate receive operation of CSIn.

8.3.5 Transmission in 3-wire serial I/O mode

Transmission is started when data is written to the SIO_n (n = 0 to 2) register after transmission has been enabled by the CSIM_n (n = 0 to 2) register.

(1) Starting transmission

Transmission is started by writing the transmit data to the SIO_n register after the CTXEn (n = 0 to 2) bit of the CSIM_n register has been set (the CRXEn (n = 0 to 2) bit is cleared to "0").

If the CTXEn bit is reset to 0, the SO_n (n = 0 to 2) pin goes into a high-impedance state.

(2) Transmitting data in synchronization with serial clock

(a) When internal clock is selected as serial clock

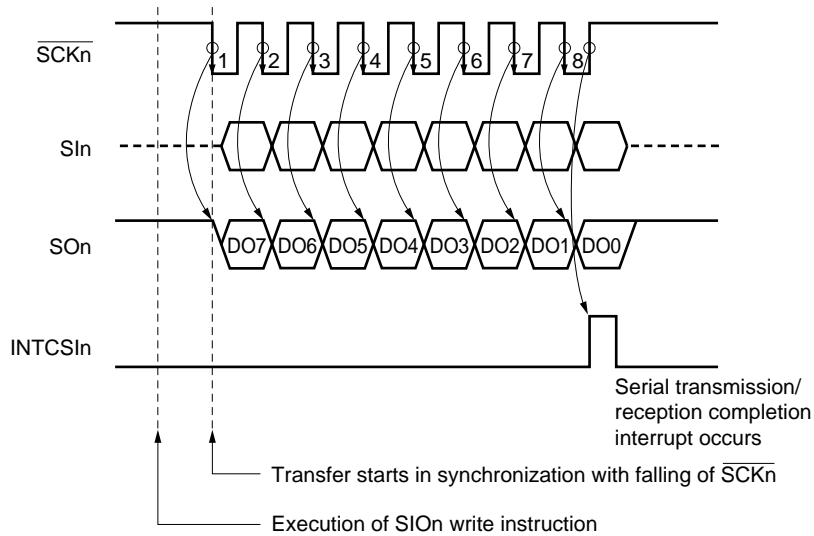
When transmission is started, the serial clock is output from the $\overline{\text{SCKn}}$ (n = 0 to 2) pin, and at the same time, data is sequentially output to the SO_n pin from SIO_n register in synchronization with the falling edge of the serial clock.

(b) When external clock is selected as serial clock

When transmission is started, the data is sequentially output from the SIO_n register to the SO_n pin in synchronization with the falling of the serial clock input to the $\overline{\text{SCKn}}$ pin immediately after transmission has been started. The shift operation is not performed even if the serial clock is input to the $\overline{\text{SCKn}}$ pin if transmission is not enabled, and the output level of the SO_n pin will not change.

★

Figure 8-6. Timing of 3-Wire Serial I/O Mode (transmission)



Remark n = 0 to 2

8.3.6 Reception in 3-wire serial I/O mode

Reception is started if the status is changed from reception disabled to reception enabled status by the CSIMn (n = 0 to 2) register or if the SIO_n (n = 0 to 2) register is read by the CPU with reception enabled.

(1) Starting reception

Reception can be started in the following two ways:

- <1> Changing the status of the CRXEn (n = 0 to 2) bit of the CSIMn register from "0" (reception disabled) to "1" (reception enabled)
- <2> Reading the receive data from the SIO_n when the CRXE0 bit of the CSIMn register is "1" (reception enabled)

If CRXEn has already been set to "1", writing "1" to this bit does not initiate receive operation. When CRXEn = 0, the input to SIO_n register is "0".

(2) Receiving data in synchronization with serial clock

(a) When internal clock is selected as serial clock

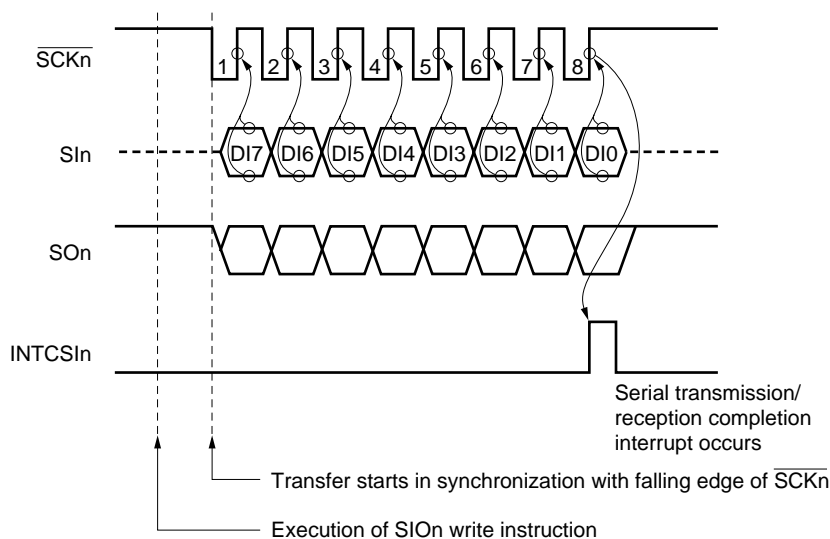
When reception is started, the serial clock is output from the $\overline{\text{SCKn}}$ (n = 0 to 2) pin, and at the same time, data is sequentially loaded from the SIn (n = 0 to 2) pin to the SIO_n register in synchronization with the rising edge of the serial clock.

(b) When external clock is selected as serial clock

When reception is started, the data is sequentially loaded from the SIn (n = 0 to 2) pin to SIO_n in synchronization with the rising of the serial clock input to the $\overline{\text{SCKn}}$ pin immediately after reception has been started. The shift operation is not performed even if the serial clock is input to the $\overline{\text{SCKn}}$ pin when reception is not enabled.

★

Figure 8-7. Timing of 3-Wire Serial I/O Mode (reception)



Remark n = 0 to 2

8.3.7 Transmission/reception in 3-wire serial I/O mode

Transmission and reception can be executed simultaneously if both transmission and reception are enabled by the CSIMn (n = 0 to 2) register.

(1) Starting transmission/reception

Transmission and reception can be performed simultaneously (transmission/reception operation) when both the CTXEn (n = 0 to 2) and CRXEn (n = 0 to 2) bits of the CSIMn register are set to 1.

- ★ Transmission/reception can be started by writing the transmit data to the SIO_n (n = 0 to 2) register when both the CTXEn and CRXEn bits of the CSIMn register are “1” (transmission/reception enabled).
If CRXEn has already been set to “1”, writing “1” to this bit does not initiate transmit/receive operation.

(2) Transmitting data in synchronization with serial clock

(a) When internal clock is selected as serial clock

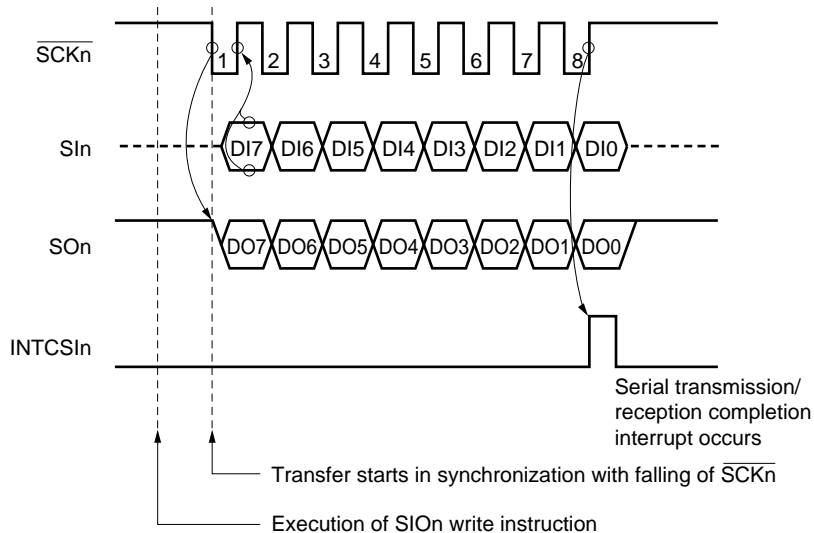
When transmission/reception is started, the serial clock is output from the $\overline{\text{SCKn}}$ (n = 0 to 2) pin, and at the same time, data is sequentially set to the SO_n (n = 0 to 2) pin from the SIO_n register in synchronization with the falling edge of the serial clock. Simultaneously, the data of the SI_n (n = 0 to 2) pin is sequentially loaded to SIO_n register in synchronization with the rising edge of the serial clock.

(b) When external clock is selected as serial clock

When transmission/reception is started, the data is sequentially output from the SIO_n register to the SO_n pin in synchronization with the falling edge of the serial clock input to the $\overline{\text{SCKn}}$ pin immediately after transmission/reception has been started. The data of the SI_n pin is sequentially loaded to the SIO_n register in synchronization with the rising edge of the serial clock. The shift operation is not performed even if the serial clock is input to the $\overline{\text{SCKn}}$ pin when transmission/reception is not enabled, and the output level of the SO_n pin does not change.

★

Figure 8-8. Timing of 3-Wire Serial I/O Mode (transmission/reception)



Remark n = 0 to 2

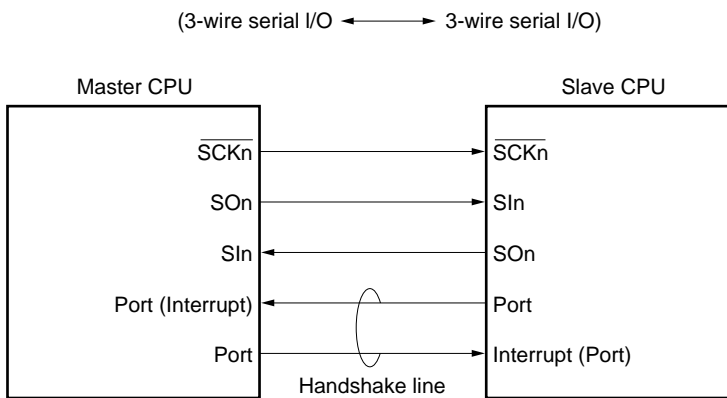
8.3.8 System Configuration Example

Data 8 bits long is transferred by using three signal lines: serial clock (\overline{SCKn}) (n = 0 to 2), serial input (SIn) (n = 0 to 2), and serial output (SOn) (n = 0 to 2). This feature is effective for connecting peripheral I/Os and display controllers that have a conventional clocked serial interface.

To connect two or more devices, a handshake line is necessary.

Various devices can be connected, because it can be specified whether the data is transmitted starting from the MSB or LSB.

Figure 8-9. Example of CSI System Configuration



Remark n = 0 to 2

8.4 Baud Rate Generator 0, 1 (BRG0, BRG1)

8.4.1 Configuration and function

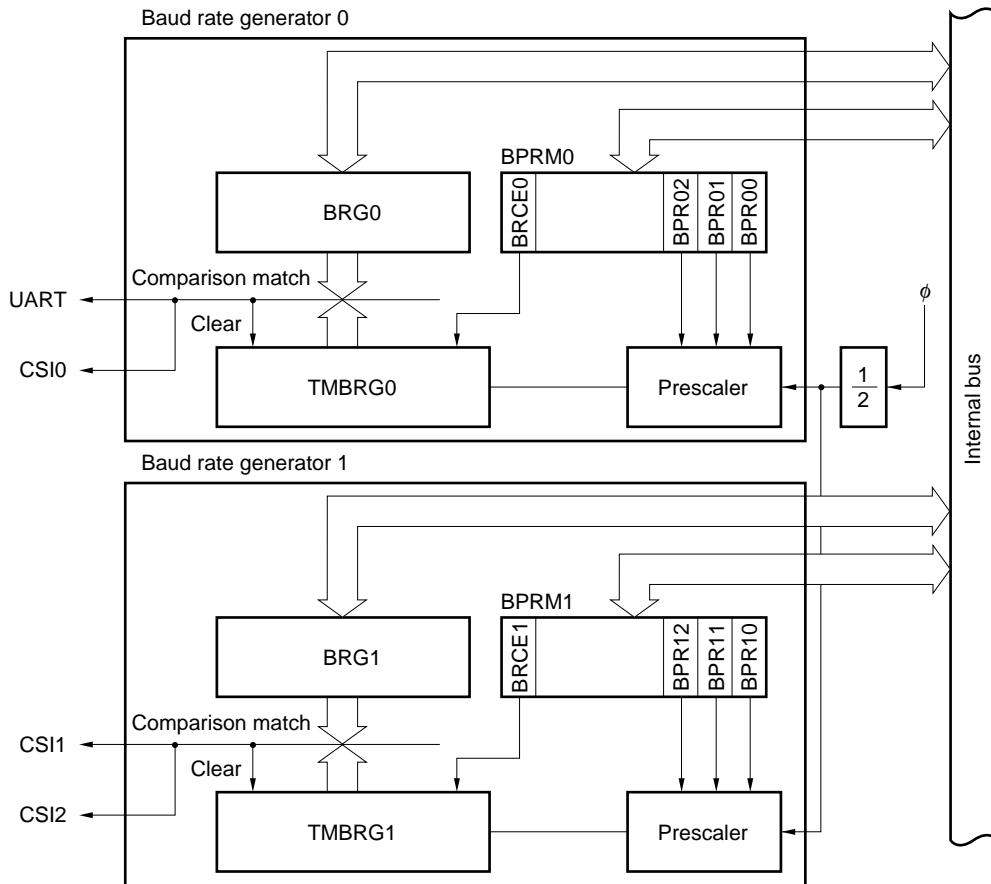
The serial interface can use the output of the internal baud rate generator or ϕ (system clock) as the serial clock.

The serial clock source for the UART is specified by the SCLS0 bit of the ASIM00 register. The serial clock source for the CSIn (n = 0 to 2) is specified by the CLSn0 and CLSn1 bits of the CSIMn (n = 0 to 2) register.

When the output of the baud rate generator is specified, the baud rate generator will be used as the clock source.

Because the serial clock for transmission/reception is shared by both the transmission and reception portions, the same baud rate is used for both transmission and reception.

Figure 8-10. Block Diagram of Baud Rate Generator



(1) Dedicated baud rate generator (BRG0 and BRG1)

The dedicated baud rate generator BRG consists of an 8-bit timer (TMBRG0, TMBRG1) that generates a shift clock for transmission/reception, a compare register (BRG0, BRG1), and a prescaler.

(a) Input clock

System clock ϕ is input to the BRG0 and BRG1 register.

(b) Set-up value of BRG0 and BRG1 register**(i) UART**

If the dedicated baud rate generator is specified for UART, the actual baud rate can be calculated by the following expression, because a sampling rate of x16 is used:

$$\text{Baud rate} = \frac{\phi}{2 \times m \times 2^n \times 16 \times 2} \text{ [bps]}$$

where,

ϕ : system clock frequency [Hz]

m : Timer-counted value ($1 \leq m \leq 256$ ^{Note}) : set by BRG0, BRG1

n : Prescaler set-up value ($n = 0, 1, 2, 3, 4$) : set by BPRM0, BPRM1

Note $m = 256$ is set by writing 0 to the BRG register.

(ii) CSI0 to CSI2

If the dedicated baud rate generator is specified for CSI0 to CSI2, the actual baud rate can be calculated by the following expression:

$$\text{Baud rate} = \frac{\phi}{2 \times m \times 2^n \times 2} \text{ [bps]}$$

where,

ϕ : system clock frequency [Hz]

m : Timer-counted value ($1 \leq m \leq 256$ ^{Note}) : set by BRG0, BRG1

n : Prescaler set-up value ($n = 0, 1, 2, 3, 4$) : set by BPRM0, BPRM1

Note $m = 256$ is set by writing 0 to the BRG register.

Table 8-2 shows the set-up values of the baud rate generator when the typical clocks are used:

Table 8-2. BRG Set-up Values

| Baud Rate [bps] | | $\phi = 25$ MHz | | | $\phi = 16$ MHz | | | $\phi = 13.5$ MHz | | | $\phi = 12.288$ MHz | | |
|-----------------|---------|-----------------|------|--------|-----------------|------|------------------------|-------------------|------|-----------------------|---------------------|------|------------------------|
| UART | CSI | BPR | BRG0 | Error | BPR | BRG0 | Error | BPR | BRG0 | Error | BPR | BRG0 | Error |
| 110 | 1760 | 4 | 222 | 0.02 % | 4 | 142 | 0.03 % | 3 | 240 | 0.12 % | 3 | 218 | 0.08 % |
| 150 | 2400 | 4 | 163 | 0.15 % | 3 | 208 | 0.16 % | 3 | 176 | 0.12 % | 3 | 160 | 0.0 % |
| 300 | 4800 | 3 | 163 | 0.15 % | 2 | 208 | 0.16 % | 2 | 176 | 0.12 % | 2 | 160 | 0.0 % |
| 600 | 9600 | 2 | 163 | 0.15 % | 1 | 208 | 0.16 % | 1 | 176 | 0.12 % | 1 | 160 | 0.0 % |
| 1200 | 19200 | 1 | 163 | 0.15 % | 0 | 208 | 0.16 % | 0 | 176 | 0.12 % | 0 | 160 | 0.0 % |
| 2400 | 38400 | 0 | 163 | 0.15 % | 0 | 104 | 0.16 % | 0 | 88 | 0.12 % | 0 | 80 | 0.0 % |
| 4800 | 76800 | 0 | 81 | 0.47 % | 0 | 52 | 0.16 % | 0 | 44 | 0.12 % | 0 | 40 | 0.0 % |
| 9600 | 153600 | 0 | 41 | 0.76 % | 0 | 26 | 0.16 % | 0 | 22 | 0.12 % | 0 | 20 | 0.0 % |
| 10400 | 166400 | 0 | 38 | 1.16 % | 0 | 24 | 1.16 % | 0 | 20 | 0.12 % | 0 | 18 | 2.6 % |
| 19200 | 307200 | 0 | 20 | 1.73 % | 0 | 13 | 1.16 % | 0 | 11 | 0.12 % | 0 | 10 | 0.0 % |
| 38400 | 614400 | 0 | 10 | 1.73 % | 0 | 7 | 6.99 % ^{Note} | 0 | 6 | 8.4 % ^{Note} | 0 | 5 | 0.0 % |
| 76800 | 1228800 | 0 | 5 | 1.73 % | – | – | – | – | – | – | 0 | 3 | 16.7 % ^{Note} |
| 153600 | 2457600 | 0 | 2 | 27.2 % | – | – | – | – | – | – | – | – | – |

| Baud Rate [bps] | | $\phi = 20$ MHz | | | $\phi = 14.746$ MHz | | | $\phi = 12.5$ MHz | | | $\phi = 9.830$ MHz | | |
|-----------------|---------|-----------------|------|--------|---------------------|------|------------------------|-------------------|------|------------------------|--------------------|------|--------|
| UART | CSI | BPR | BRG0 | Error | BPR | BRG0 | Error | BPR | BRG0 | Error | BPR | BRG0 | Error |
| 110 | 1760 | 4 | 178 | 0.25 % | 4 | 131 | 0.07 % | 3 | 222 | 0.02 % | 3 | 175 | 0.26 % |
| 150 | 2400 | 4 | 130 | 0.16 % | 3 | 192 | 0.0 % | 3 | 163 | 0.15 % | 3 | 128 | 0.0 % |
| 300 | 4800 | 3 | 130 | 0.16 % | 2 | 192 | 0.0 % | 2 | 163 | 0.15 % | 2 | 128 | 0.0 % |
| 600 | 9600 | 2 | 130 | 0.16 % | 1 | 192 | 0.0 % | 1 | 163 | 0.15 % | 1 | 128 | 0.0 % |
| 1200 | 19200 | 1 | 130 | 0.16 % | 0 | 192 | 0.0 % | 0 | 163 | 0.15 % | 0 | 128 | 0.0 % |
| 2400 | 38400 | 0 | 130 | 0.16 % | 0 | 96 | 0.0 % | 0 | 81 | 0.47 % | 0 | 64 | 0.0 % |
| 4800 | 76800 | 0 | 65 | 0.16 % | 0 | 48 | 0.0 % | 0 | 41 | 0.76 % | 0 | 32 | 0.0 % |
| 9600 | 153600 | 0 | 33 | 1.36 % | 0 | 24 | 0.0 % | 0 | 20 | 1.73 % | 0 | 16 | 0.0 % |
| 10400 | 166400 | 0 | 30 | 0.16 % | 0 | 22 | 0.7 % | 0 | 19 | 1.16 % | 0 | 15 | 1.5 % |
| 19200 | 307200 | 0 | 16 | 1.73 % | 0 | 12 | 0.0 % | 0 | 10 | 1.73 % | 0 | 8 | 0.0 % |
| 38400 | 614400 | 0 | 8 | 1.73 % | 0 | 6 | 0.0 % | 0 | 5 | 1.73 % | 0 | 4 | 0.0 % |
| 76800 | 1228800 | 0 | 4 | 1.73 % | 0 | 3 | 0.0 % | 0 | 3 | 15.2 % ^{Note} | 0 | 2 | 0.0 % |
| 153600 | 2457600 | 0 | 2 | 1.73 % | 0 | 2 | 25.0 % ^{Note} | – | – | – | 0 | 1 | 0.0 % |

Note Cannot be used because the error is too great.

(c) Error of baud rate generator

The error of the baud rate generator is calculated as follows:

$$\text{Error [\%]} = \left[\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (normal baud rate)}} - 1 \right] \times 100$$

Example: $(9520/9600-1) \times 100 = -0.833 \text{ [\%]}$
 $(5000/4800-1) \times 100 = +4.167 \text{ [\%]}$

(2) Allowable error range of baud rate generator

The allowable error range depends on the number of bits of one frame.

The basic limit is $\pm 5 \%$ of baud rate error and $\pm 4.5 \%$ of sample timing with an accuracy of 16 bits. However, the practical limit should be $\pm 2.3 \%$ of baud rate error, assuming that both the transmission and reception sides contain an error.

8.4.2 Baud rate generator register 0, 1 (BRG0, BRG1)

This is an 8-bit compare register that sets a timer/count value for the dedicated baud rate generator.

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

| | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| BRG0 | BRG07 | BRG06 | BRG05 | BRG04 | BRG03 | BRG02 | BRG01 | BRG00 | Address FFFFFF084H | At reset Undefined |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| BRG1 | BRG17 | BRG16 | BRG15 | BRG14 | BRG13 | BRG12 | BRG11 | BRG10 | Address FFFFFF094H | At reset Undefined |

Caution The internal timer (TMBRGn) (n = 0 and 1) is cleared by writing the BRGn (n = 0 and 1) register. Therefore, do not rewrite or program the BRGn register during transmission/reception operation.

8.4.3 Baud rate generator prescaler mode register 0, 1 (BPRM0, BPRM1)

This register controls the timer/count operation of the dedicated baud rate generator and selects a count clock.

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

| | | | | | | | | | | |
|-------|-------|---|---|---|---|-------|-------|-------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| BPRM0 | BRCE0 | 0 | 0 | 0 | 0 | BPR02 | BPR01 | BPR00 | Address FFFFFF086H | At reset 00H |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| BPRM1 | BRCE1 | 0 | 0 | 0 | 0 | BPR12 | BPR11 | BPR10 | Address FFFFFF096H | At reset 00H |

| Bit Position | Bit Name | Function | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|----------------|--|-------------------|-------|-------|-------------|---|---|---|------------------|---|---|---|------------------|---|---|---|------------------|---|---|---|-------------------|---|---|---|-------------------|
| 7 | BRCEm | Baud Rate Generator Count Enable Controls count operation of BRG. 0: Stops count operation with cleared 1: Enables count operation | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 to 0 | BPRm2 to BPRm0 | Baud Rate Generator Prescaler Specifies count clock input to TMBRG. <table border="1" style="margin: 10px auto;"> <thead> <tr> <th>BPR02</th> <th>BPR01</th> <th>BPR00</th> <th>Count clock</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>$\phi/2$ (n = 0)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>$\phi/4$ (n = 1)</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>$\phi/8$ (n = 2)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>$\phi/16$ (n = 3)</td> </tr> <tr> <td>1</td> <td>x</td> <td>x</td> <td>$\phi/32$ (n = 4)</td> </tr> </tbody> </table> n: set value of prescaler, ϕ : system clock | BPR02 | BPR01 | BPR00 | Count clock | 0 | 0 | 0 | $\phi/2$ (n = 0) | 0 | 0 | 1 | $\phi/4$ (n = 1) | 0 | 1 | 0 | $\phi/8$ (n = 2) | 0 | 1 | 1 | $\phi/16$ (n = 3) | 1 | x | x | $\phi/32$ (n = 4) |
| BPR02 | BPR01 | BPR00 | Count clock | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | $\phi/2$ (n = 0) | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | $\phi/4$ (n = 1) | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | $\phi/8$ (n = 2) | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | $\phi/16$ (n = 3) | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | x | x | $\phi/32$ (n = 4) | | | | | | | | | | | | | | | | | | | | | | | |

Caution Do not change the count clock during transmission/reception operation.

Remark m = 0, 1

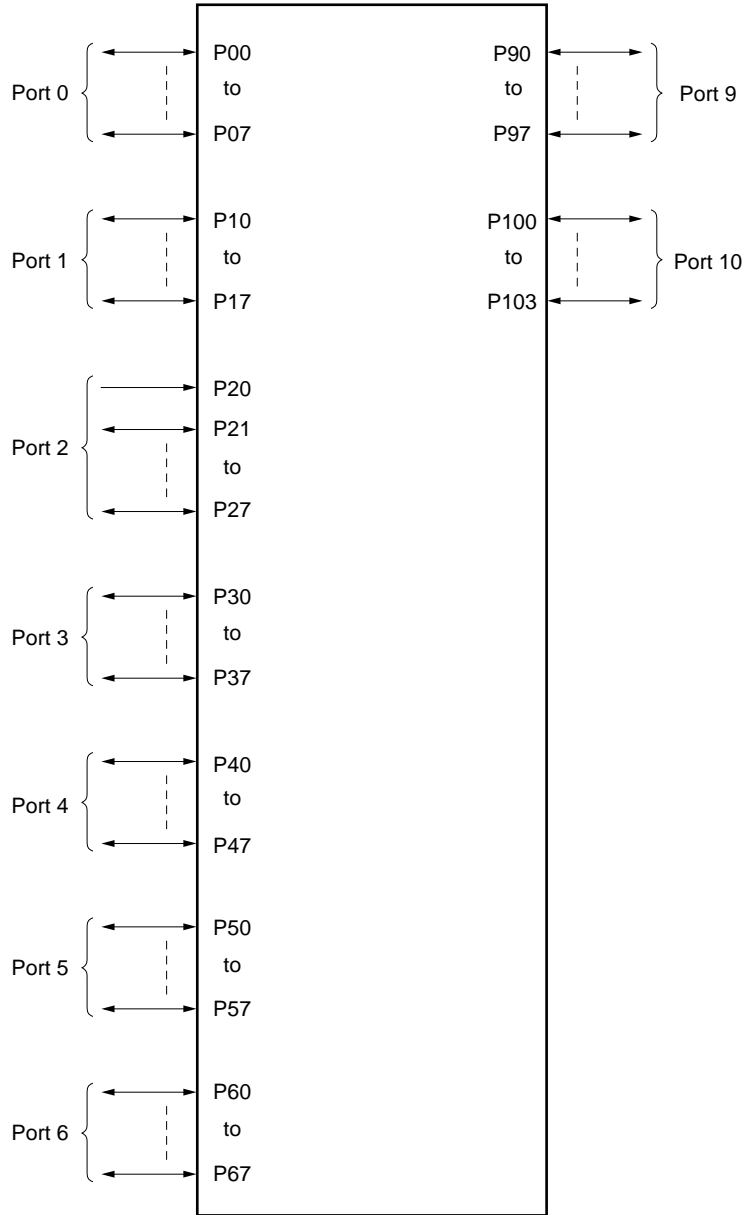
CHAPTER 9 PORT FUNCTION**9.1 Features**

The ports of the V852 have the following features:

- Number of pins: input : 1
 I/O : 67
- Multiplexed with I/O pins of other peripheral functions
- Can be set in input/output mode in 1-bit units
- Noise elimination
- Edge detection

9.2 Basic Configuration of Port

The V852 is provided with a total of 67 input/output port pins that make up ports 0 to 10. The configuration of the V852's ports is shown below.



(1) Function of each port

The ports of the V852 have the functions shown in the table below.

In addition to port functions, some ports have functions as internal hardware input/output pins, when placed in the control mode.

| Port Name | Port Function | Function in Control Mode | Remarks |
|-----------|---|--|--|
| Port 0 | 8-bit ^{Note} I/O port (Can specify I/O bitwise) | Real-time pulse unit (RPU) input/output | Can be set in port or control mode in 1-bit units |
| | | External interrupt request input | |
| Port 1 | | – | Fixed to port mode |
| Port 2 | | External interrupt request input | Can be set in port or control mode in 1-bit units |
| | | Serial interface (CSI2) input/output | |
| Port 3 | | Serial interface (UART, CSI0, CSI1) input/output | |
| Port 4 | | Address/data bus (AD0 to AD7) for external memory | Can be set in port or control mode in 8-bit units |
| Port 5 | | Address/data bus (AD8 to AD15) for external memory | |
| Port 6 | | Address bus (A16 to A23) for external memory | Can be set in port or control mode in 2-bit units |
| Port 9 | | Control signal output for external memory | Can be set in port or control mode in 5-, 2-, or 1-bit units |
| Port 10 | 4-bit I/O port (Can specify I/O bitwise) | Control signal input/output for system expansion | Can be set in port or control mode in 1-bit units |

Note The port 2 is a 7-bit I/O port.

★ **Caution** When switching a port that operates as an output pin or input/output pin during control mode, from port mode to control mode, the procedure below must be followed.

<1> Set the inactive level of the the signal which is output as control mode to the relevant bit of port n (Pn) (n = 0, 2, 3 to 6, 9, 10).

<2> Change to control mode by port n mode control register (PMCn).

If <1> is not done, a momentary output of the contents of port n (Pn) may occur on changing from port mode to control mode.

(2) Function of ports after reset and register to set port/control mode

(1/2)

| Port | Pin | Function at reset (Parentheses indicate I/O) | | Register to set mode |
|--------|------------------|--|---------------|----------------------|
| | | Single-chip mode | ROM-less mode | |
| Port 0 | P00/TO10 | P00 (Input) | | PMC0 |
| | P01/TO11 | P01 (Input) | | |
| | P02/TCLR1 | P02 (Input) | | |
| | P03/TI1 | P03 (Input) | | |
| | P04/INTP10 | P04 (Input) | | |
| | P05/INTP11 | P05 (Input) | | |
| | P06/INTP12 | P06 (Input) | | |
| | P07/INTP13 | P07 (Input) | | |
| Port 1 | P10-P17 | P10-P17 (All inputs) | | — |
| Port 2 | P20/NMI | NMI (Input) | | PMC2 |
| | P21/INTP00 | P21 (Input) | | |
| | P22/INTP01 | P22 (Input) | | |
| | P23/INTP02 | P23 (Input) | | |
| | P24/INTP03 | P24 (Input) | | |
| | P25/SO2 | P25 (Input) | | |
| | P26/SI2 | P26 (Input) | | |
| | P27/SCK2 | P27 (Input) | | |
| Port 3 | P30/SO0 | P30 (Input) | | PMC3 |
| | P31/SI0 | P31 (Input) | | |
| | P32/SCK0 | P32 (Input) | | |
| | P33/TXD | P33 (Input) | | |
| | P34/RXD | P34 (Input) | | |
| | P35/SO1 | P35 (Input) | | |
| | P36/SI1 | P36 (Input) | | |
| | P37/SCK1 | P37 (Input) | | |
| Port 4 | P40/AD0-P47/AD7 | P40-P47 (All inputs) | AD0-AD7 | MM |
| Port 5 | P50/AD8-P57/AD15 | P50-P57 (All inputs) | AD8-AD15 | MM |
| Port 6 | P60/A16-P67/A23 | P60-P67 (All inputs) | A16-A23 | MM |
| Port 9 | P90/LBEN | P90 (Input) | LBEN | MM |
| | P91/UBEN | P91 (Input) | UBEN | |
| | P92/R/W | P92 (Input) | R/W | |
| | P93/DSTB | P93 (Input) | DSTB | |
| | P94/ASTB | P94 (Input) | ASTB | |
| | P95/ST0 | P95 (Input) | | |
| | P96/ST1 | P96 (Input) | | |
| | P97 | P97 (Input) | | |

(2/2)

| Port | Pin | Function at reset (Parentheses indicate I/O) | | Register to set mode |
|---------|---------------------------------|--|---------------|----------------------|
| | | Single-chip mode | ROM-less mode | |
| Port 10 | P100/ $\overline{\text{HLDAK}}$ | P100 (Input) | | PMC10 |
| | P101/ $\overline{\text{HLDRQ}}$ | P101 (Input) | | |
| | P102 | P102 (Input) | | — |
| | P103 | P103 (Input) | | |

9.3 Port Pin Function

9.3.1 Port 0

Port 0 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P0 | P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 | Address FFFFFF00H | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P0n (n = 7 to 0) | Port 0 I/O port |

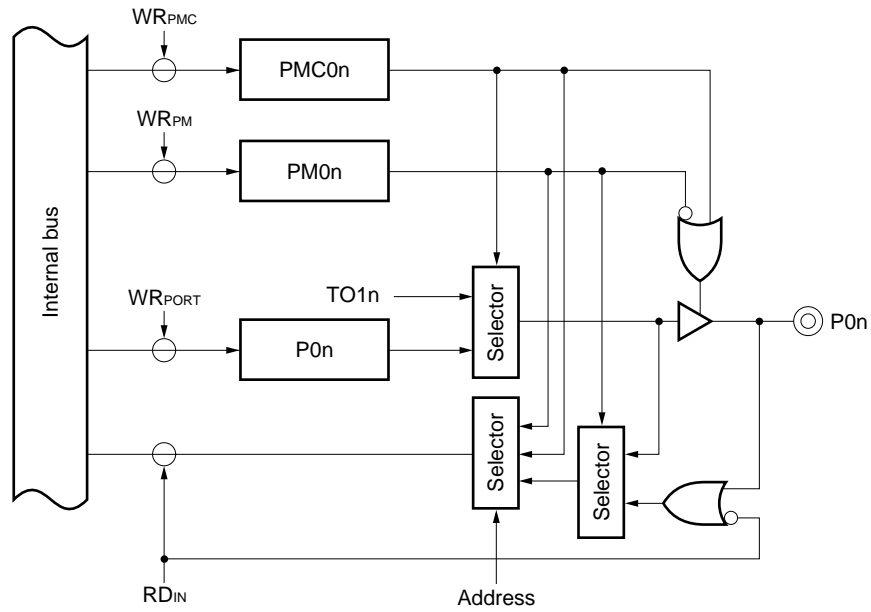
In addition to the function as a general I/O port, this port can also be used to input/output signals of the real-time pulse unit (RPU) and input external interrupt requests, when placed in the control mode.

Operation in control mode

| | Port | Control Mode | Remarks |
|--------|------------|------------------|-----------------------------------|
| Port 0 | P00 | TO10 | Real-time pulse unit (RPU) output |
| | P01 | TO11 | |
| | P02 | TCLR1 | Real-time pulse unit (RPU) input |
| | P03 | TI1 | |
| | P04 to P07 | INTP10 to INTP13 | External interrupt input |

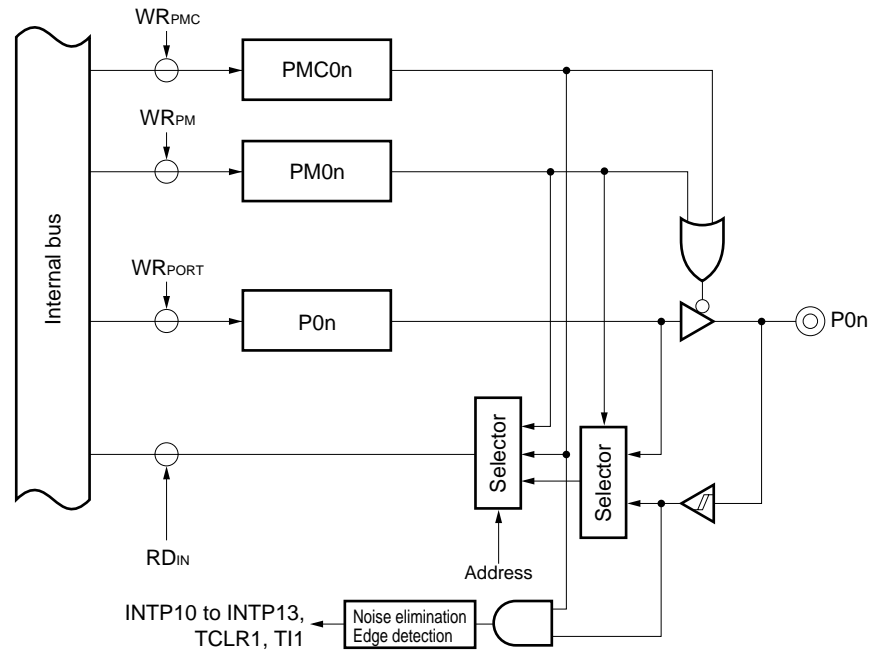
(1) Hardware configuration

Figure 9-1. Block Diagram of P00, P01 (Port 0)



Remark n = 0, 1

Figure 9-2. Block Diagram of P02 to P07 (Port 0)



Remark n = 2 to 7

(2) Setting input/output mode and control mode

The input/output mode of port 0 is set by port mode register 0 (PM0). The control mode is set by port mode control register 0 (PMC0).

Port 0 mode register (PM0)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM0 | PM07 | PM06 | PM05 | PM04 | PM03 | PM02 | PM01 | PM00 | Address FFFFFF020H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|--------------|---|
| 7 to 0 | PM00 to PM07 | Port Mode Sets P00 to P07 pins in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Port 0 mode control register (PMC0)

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

| | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PMC0 | PMC07 | PMC06 | PMC05 | PMC04 | PMC03 | PMC02 | PMC01 | PMC00 | Address FFFFFF040H | At reset 00H |

| Bit Position | Bit Name | Function |
|--------------|----------------|---|
| 7 to 4 | PMC07 to PMC04 | Port Mode Control Indicates operation mode of P0n pin. 0: I/O port mode 1: External interrupt request input (INTP13 to INTP10) |
| 3 | PMC03 | Port Mode Control Indicates operation mode of P03 pin. 0: I/O port mode 1: TI1 input mode |
| 2 | PMC02 | Port Mode Control Indicates operation mode of P02 pin. 0: I/O port mode 1: TCLR1 input mode |
| 1 | PMC01 | Port Mode Control Indicates operation mode of P01 pin. 0: I/O port mode 1: TO11 output mode |
| 0 | PMC00 | Port Mode Control Indicates operation mode of P00 pin. 0: I/O port mode 1: TO10 output mode |

9.3.2 Port 1

Port 1 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|---------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P1 | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | Address FFFFF02H | At reset Undefined |

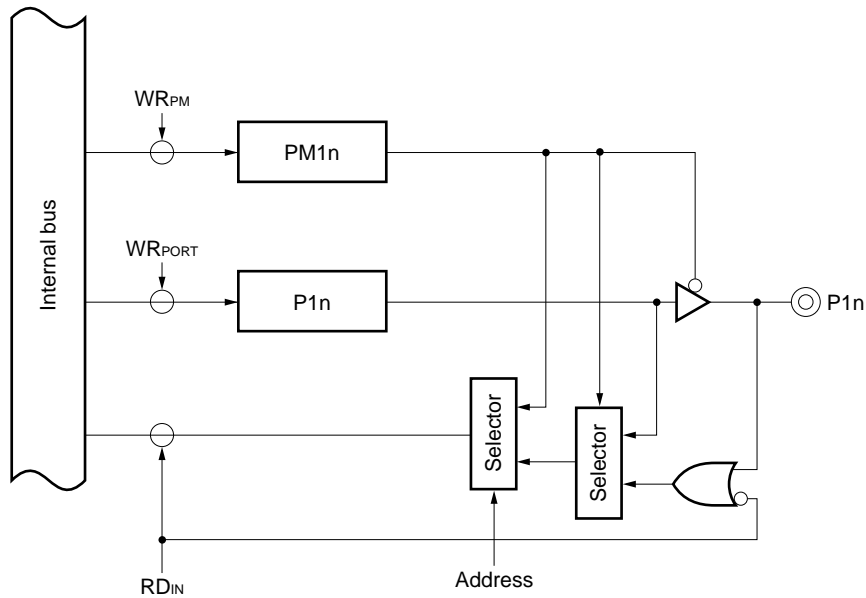
| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P1n (n = 7 to 0) | Port 1 I/O port |

Port 1 is not multiplexed with other functions and is fixed in the port mode.

| Port | Control Mode | Remarks |
|--------|--------------|--------------------|
| Port 1 | P10 to P17 | Fixed in port mode |

(1) Hardware configuration

Figure 9-3. Block Diagram of P10 to P17 (Port 1)



Remark n = 0 to 7

(2) Setting input/output mode

The input/output mode of port 1 is set by port mode register 1 (PM1).

Port 1 mode register (PM1)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 | Address FFFFF022H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM1n (n = 7 to 0) | Port Mode Sets P1n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

9.3.3 Port 2

Port 2 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units. However, P20 always operates as a NMI when an edge is input.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P2 | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 | Address FFFFF004H | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|-------------------------|
| 7 to 1 | P2n (n = 7 to 1) | Port 2 I/O port |
| 0 | P20 | Fixed to NMI input mode |

★

In addition to the function as a port, this port can also be used to input external interrupt requests and clocked serial interface (CSI) I/O in the control mode.

Operation in control mode

| Port | Control Mode | Remarks | |
|--------|--------------|--------------------------|---|
| Port 2 | P20 | NMI | Non-maskable interrupt request input |
| | P21 to 24 | INTP00 to INTP03 | External interrupt request input |
| | P25 | SO2 | I/O for clocked serial interface (CSI2) |
| | P26 | SI2 | |
| | P27 | $\overline{\text{SCK2}}$ | |

(1) Hardware configuration

Figure 9-4. Block Diagram of P20 (Port 2)

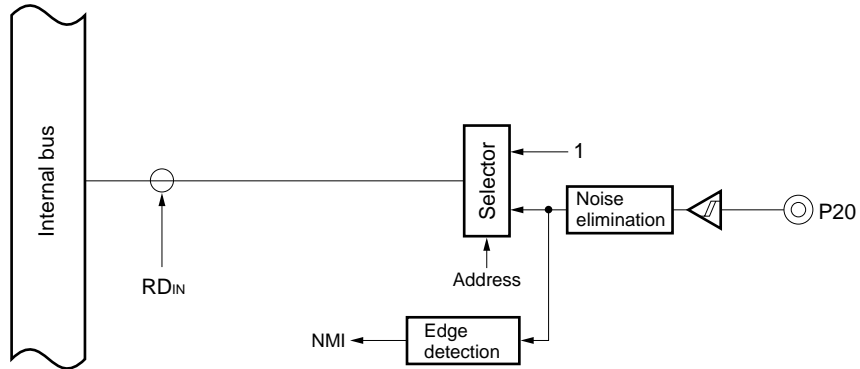
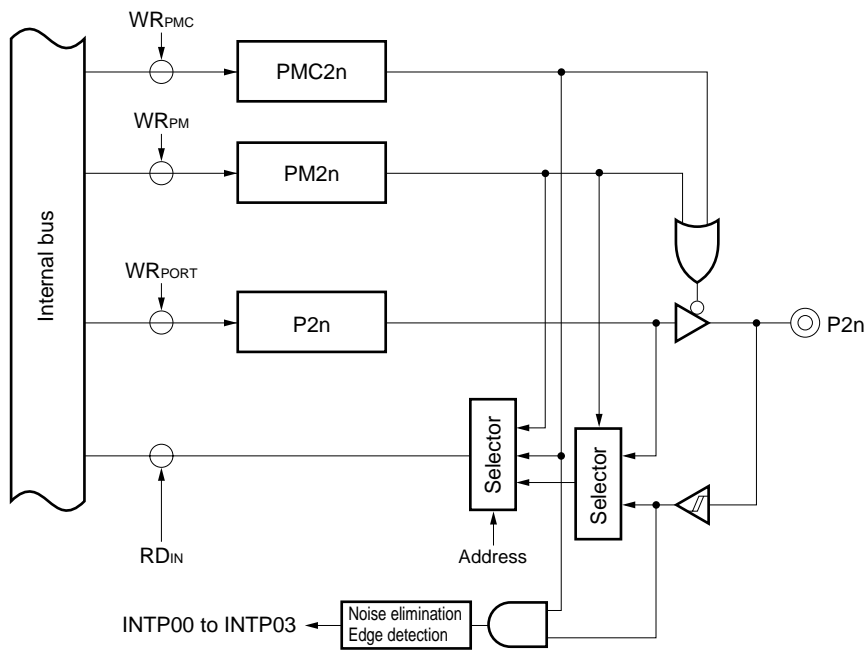


Figure 9-5. Block Diagram of P21 to P24 (Port 2)



Remark n = 1 to 4

Figure 9-6. Block Diagram of P25 (Port 2)

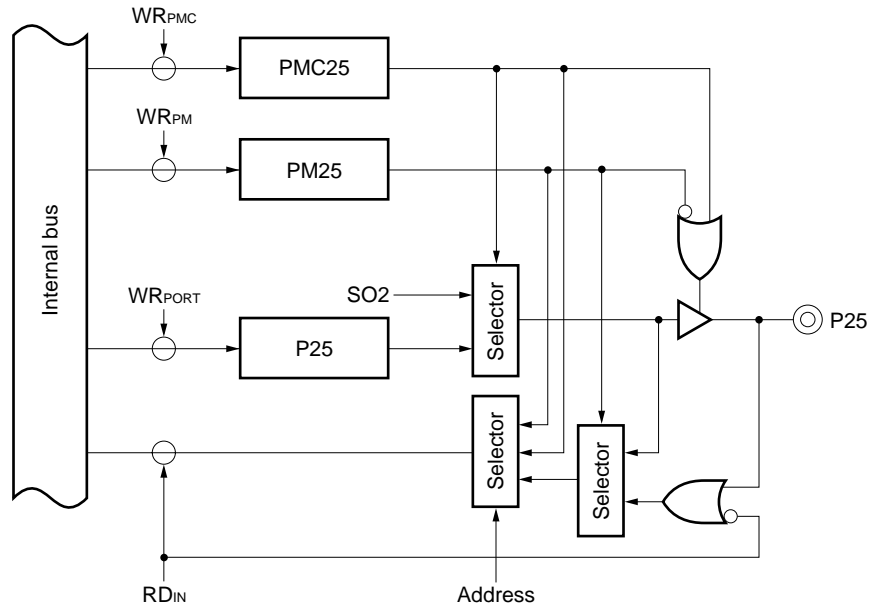


Figure 9-7. Block Diagram of P26 (Port 2)

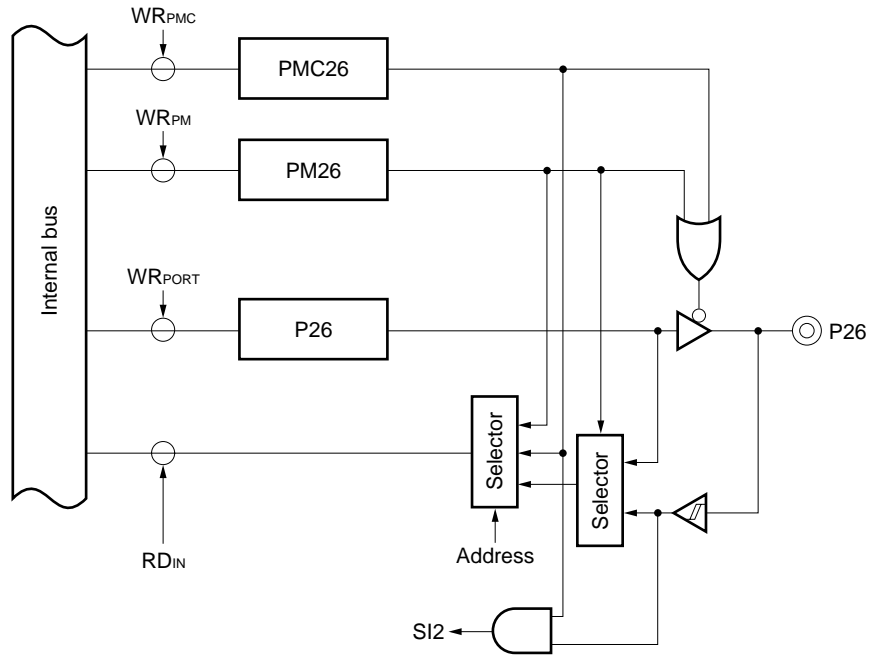
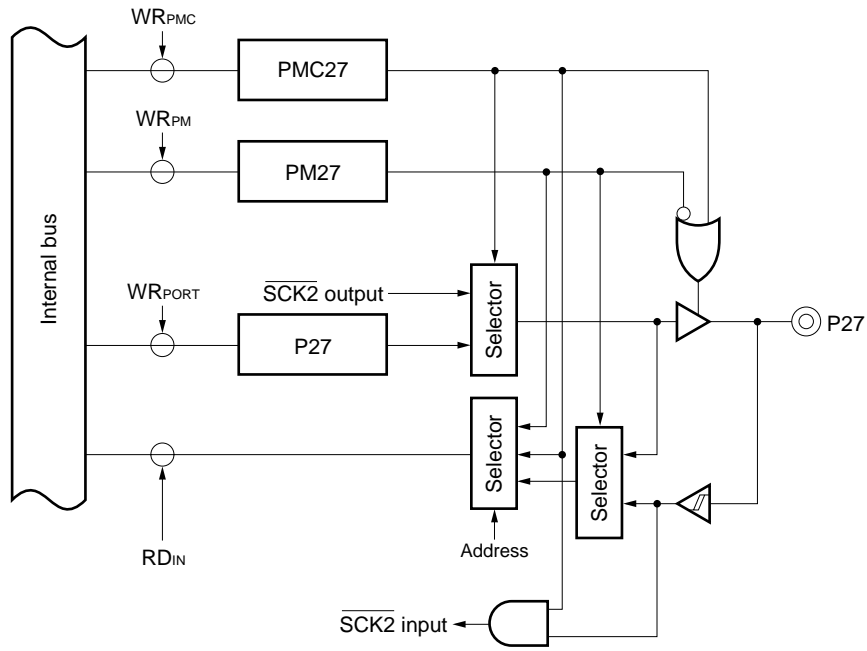


Figure 9-8. Block Diagram of P27 (Port 2)



(2) Setting input/output mode and control mode

The input/output mode of port 2 is set by port mode register 2 (PM2). The control mode is set by port mode control register 2 (PMC2).

P20 is fixed in the NMI input mode.

Port 2 mode register (PM2)

This register can be read/written in 8- or 1-bit units. However, bit 0 is fixed to "1" by hardware. Even if "0" is written to this bit, it is ignored.

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|---|---------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM2 | PM27 | PM26 | PM25 | PM24 | PM23 | PM22 | PM21 | 1 | Address FFFF024H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 1 | PM2n (n = 7 to 1) | Port Mode Sets P2n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Port 2 mode control register (PMC2)

This register can be read/written in 8- or 1-bit units. However, bit 0 is fixed to “1” by hardware. If “0” is written to this bit, it is ignored.

| | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|---|----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PMC2 | PMC27 | PMC26 | PMC25 | PMC24 | PMC23 | PMC22 | PMC21 | 1 | Address FFFFF044H | At reset 01H |

| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 7 | PMC27 | Port Mode Control Sets operation mode of P27 pin. 0: I/O port mode 1: $\overline{\text{SCK2}}$ I/O mode |
| 6 | PMC26 | Port Mode Control Sets operation mode of P26 pin. 0: I/O port mode 1: SI2 input mode |
| 5 | PMC25 | Port Mode Control Sets operation mode of P25 pin. 0: I/O port mode 1: SO2 output mode |
| 4 | PMC24 | Port Mode Control Sets operation mode of P24 pin. 0: I/O port mode 1: INTP03 input mode |
| 3 | PMC23 | Port Mode Control Sets operation mode of P23 pin. 0: I/O port mode 1: INTP02 input mode |
| 2 | PMC22 | Port Mode Control Sets operation mode of P22 pin. 0: I/O port mode 1: INTP01 input mode |
| 1 | PMC21 | Port Mode Control Sets operation mode of P21 pin. 0: I/O port mode 1: INTP00 input mode |

9.3.4 Port 3

Port 3 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P3 | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 | Address FFFFF006H | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P3n (n = 7 to 0) | Port 3 I/O port |

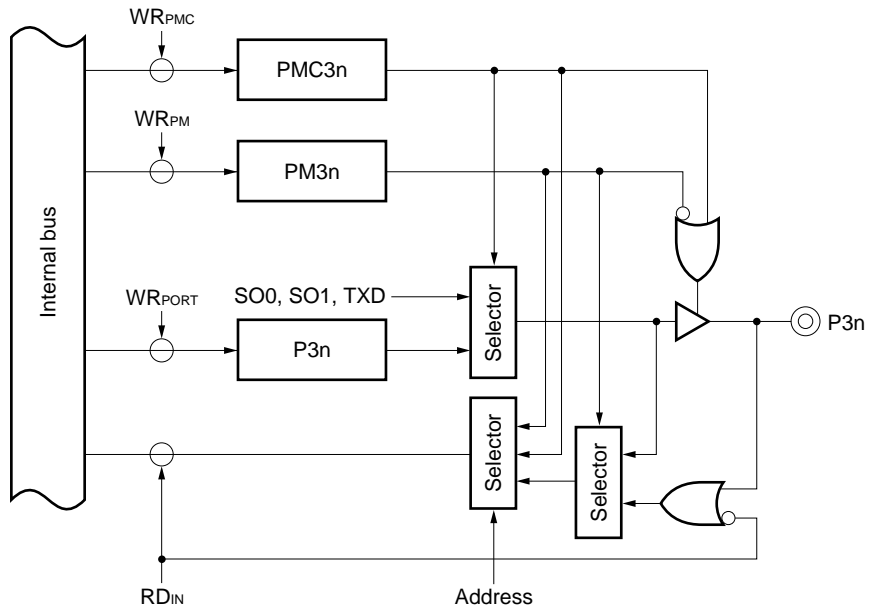
In addition to the function as a port, this port can also be used as the input/output lines of the serial interface (UART, CSI), when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks | |
|--------|--------------|--------------------------|---|
| Port 3 | P30 | SO0 | I/O for serial interface (UART, CSI0, CSI1) |
| | P31 | SI0 | |
| | P32 | $\overline{\text{SCK0}}$ | |
| | P33 | TXD | |
| | P34 | RXD | |
| | P35 | SO1 | |
| | P36 | SI1 | |
| | P37 | $\overline{\text{SCK1}}$ | |

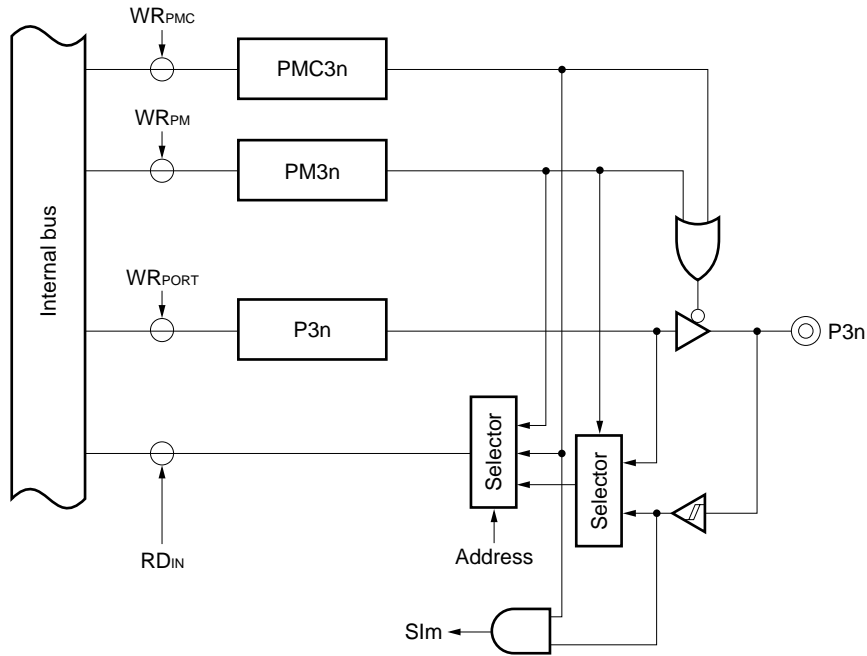
(1) Hardware configuration

Figure 9-9. Block Diagram of P30, P33, P35 (Port 3)



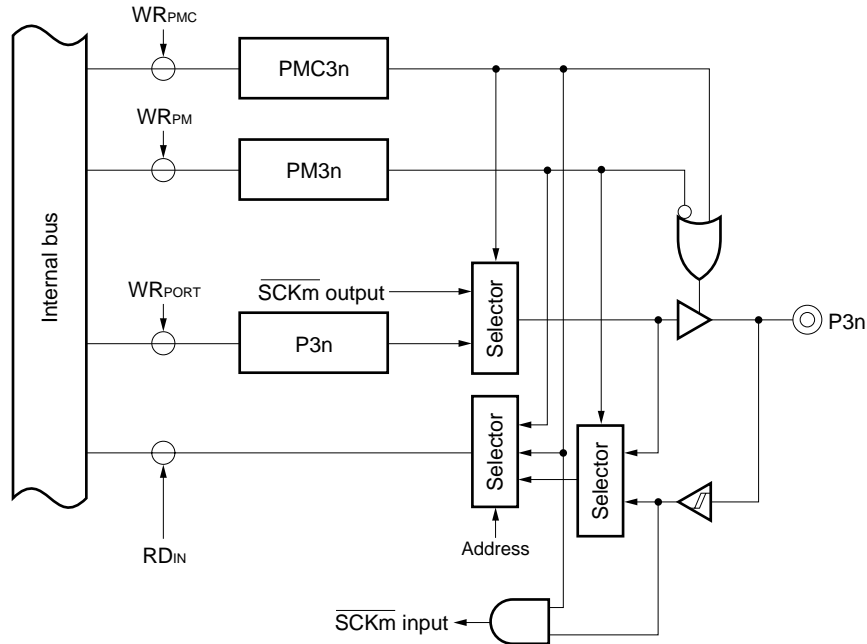
Remark $n = 0, 3, 5$

Figure 9-10. Block Diagram of P31, P36 (Port 3)



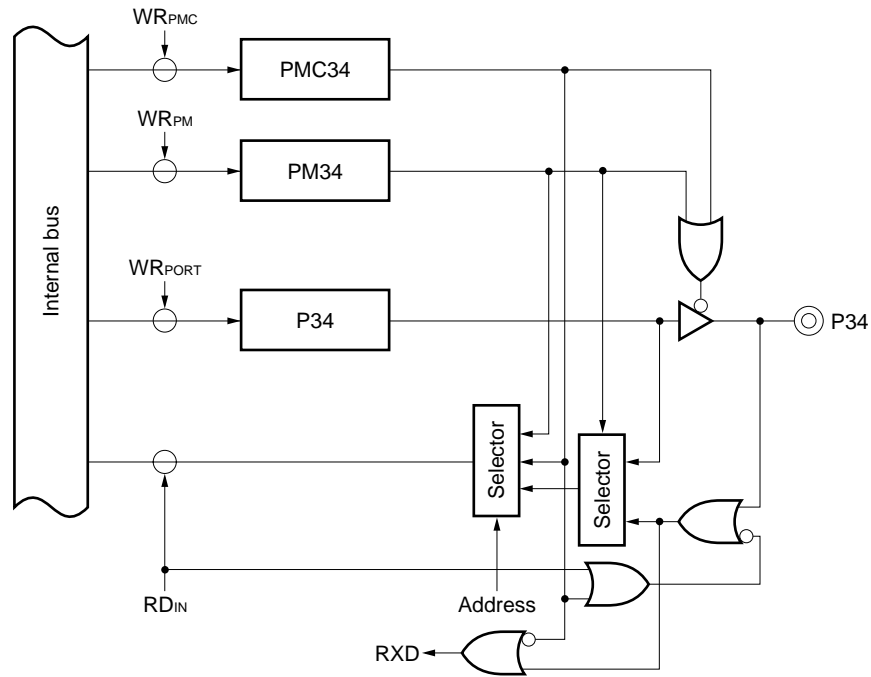
Remark $n = 1, 6$
 $m = 0, 1$

Figure 9-11. Block Diagram of P32, P37 (Port 3)



Remark $n = 2, 7$
 $m = 0, 1$

Figure 9-12. Block Diagram of P34 (Port 3)



(2) Setting input/output mode and control mode

The input/output mode of port 3 is set by port mode register 3 (PM3). The control mode is set by port mode control register 3 (PMC3).

Port 3 mode register (PM3)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM3 | PM37 | PM36 | PM35 | PM34 | PM33 | PM32 | PM31 | PM30 | Address | At reset |
| | | | | | | | | | FFFFF026H | FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM3n (n = 7 to 0) | Port Mode Sets P3n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Port 3 mode control register (PMC3)

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

| | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PMC3 | PMC37 | PMC36 | PMC35 | PMC34 | PMC33 | PMC32 | PMC31 | PMC30 | Address FFFFF046H | At reset 00H |

| Bit Position | Bit Name | Function |
|--------------|----------|---|
| 7 | PMC37 | Port Mode Control Sets operation mode of P37 pin. 0: I/O port mode 1: $\overline{\text{SCK1}}$ I/O mode |
| 6 | PMC36 | Port Mode Control Sets operation mode of P36 pin. 0: I/O port mode 1: S11 input mode |
| 5 | PMC35 | Port Mode Control Sets operation mode of P35 pin. 0: I/O port mode 1: SO1 output mode |
| 4 | PMC34 | Port Mode Control Sets operation mode of P34 pin. 0: I/O port mode 1: RXD input mode |
| 3 | PMC33 | Port Mode Control Sets operation mode of P33 pin. 0: I/O port mode 1: TXD output mode |
| 2 | PMC32 | Port Mode Control Sets operation mode of P32 pin. 0: I/O port mode 1: $\overline{\text{SCK0}}$ input/output mode |
| 1 | PMC31 | Port Mode Control Sets operation mode of P31 pin. 0: I/O port mode 1: S10 input mode |
| 0 | PMC30 | Port Mode Control Sets operation mode of P30 pin. 0: I/O port mode 1: SO0 output mode |

9.3.5 Port 4

Port 4 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P4 | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 | Address FFFFF008H | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P4n (n = 7 to 0) | Port 4 I/O port |

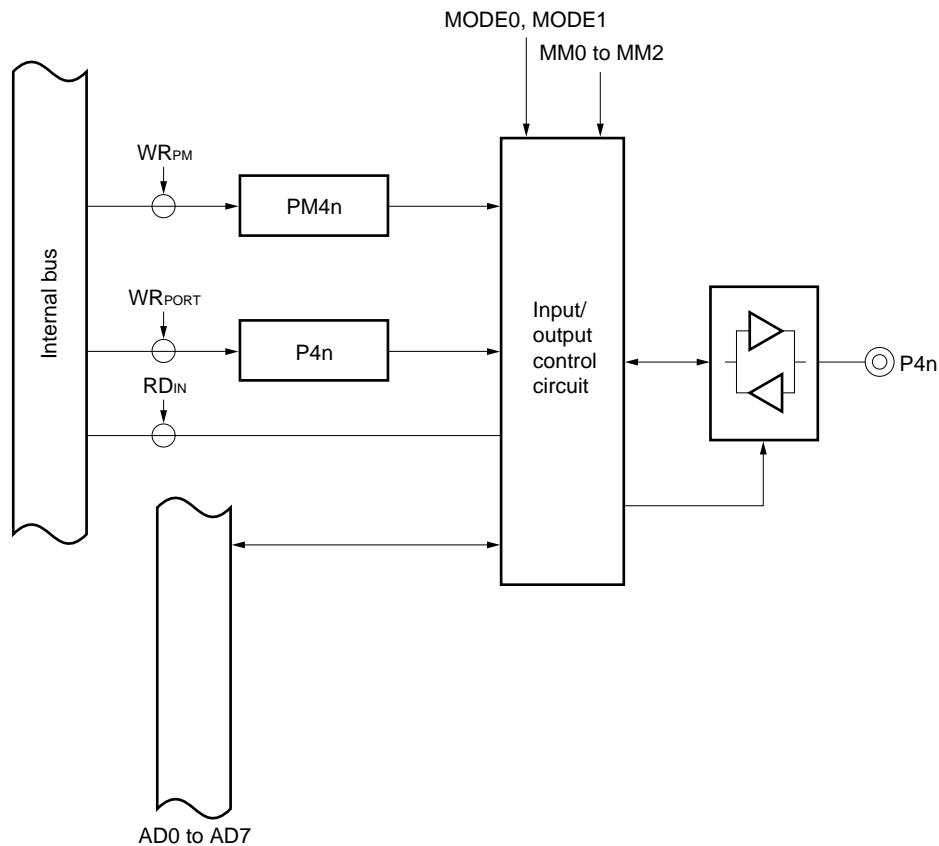
In addition to the function as a general I/O port, this port also serves as an external address/data bus, when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks | |
|--------|--------------|------------|--------------------------------------|
| Port 4 | P40 to 47 | AD0 to AD7 | Address/data bus for external memory |

(1) Hardware configuration

Figure 9-13. Block Diagram of P40 to P47 (Port 4)



Remark n = 0 to 7

(2) Setting input/output mode and control mode

The input/output mode of port 4 is set by port mode register 4 (PM4). The control mode (external expansion mode) is set by mode specification pins MODE0 and MODE1, and memory expansion mode register (MM: refer to 3.4.6 (1)).

Port 4 mode register (PM4)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM4 | PM47 | PM46 | PM45 | PM44 | PM43 | PM42 | PM41 | PM40 | Address FFFFFF028H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM4n (n = 7 to 0) | Port Mode Sets P4n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Operation mode of port 4

| Bit of MM Register | | | Operation Mode | | | | | | | |
|--------------------|-----|-----|----------------------------------|-----|-----|-----|-----|-----|-----|-----|
| MM2 | MM1 | MM0 | P40 | P41 | P42 | P43 | P44 | P45 | P46 | P47 |
| 0 | 0 | 0 | Port | | | | | | | |
| 0 | 1 | 1 | Address/data bus (AD0 to AD7) | | | | | | | |
| 1 | 0 | 0 | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | |
| Others | | | RFU (reserved) | | | | | | | |

For the details of mode selection by the MODE0 and MODE1 pins, refer to 3.3.2 Specifying operation mode.

When MODE0 and MODE1 = 00 (ROM-less mode), MM0 to MM2 bits are initialized to 111 at system reset, enabling the external expansion mode. External expansion can be disabled by programming the MM0 to MM2 bits and setting the port mode. If MM0 to MM2 are cleared to 000, the subsequent external instruction cannot be fetched.

9.3.6 Port 5

Port 5 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|------|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P5 | PM57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 | Address FFFFFF0AH | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P5n (n = 7 to 0) | Port 5 I/O port |

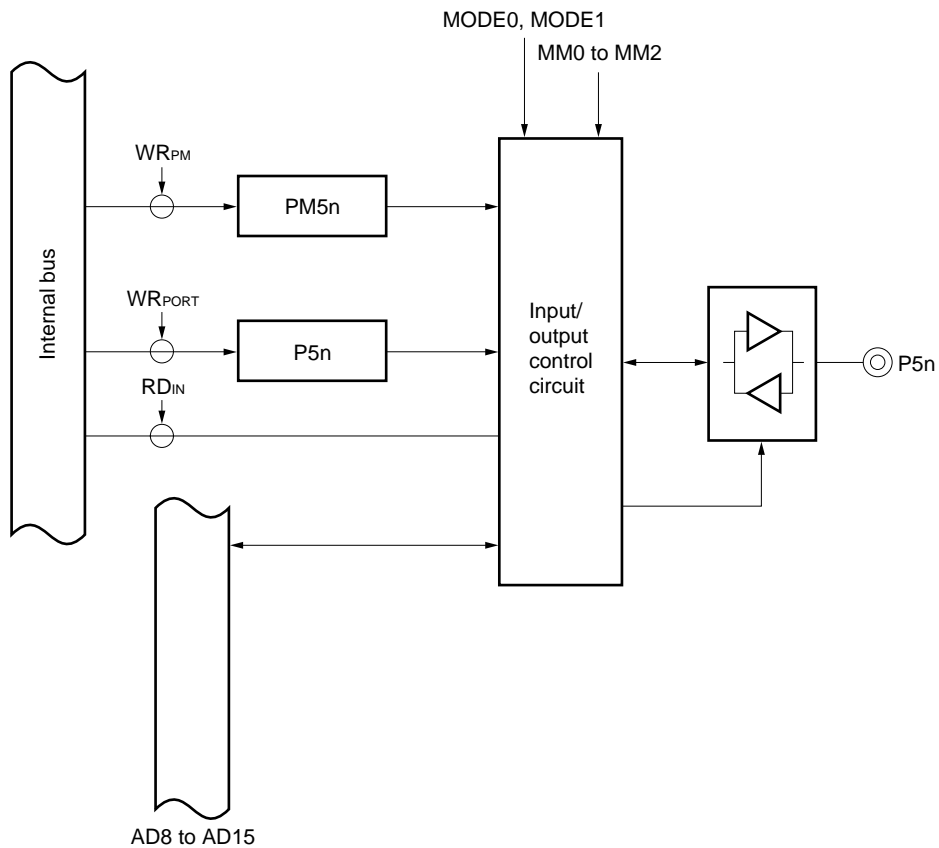
In addition to the function as a general I/O port, this port also serves as an external address/data bus, when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks |
|--------|--------------|--------------------------------------|
| Port 5 | P50 to 57 | AD8 to AD15 |
| | | Address/data bus for external memory |

(1) Hardware configuration

Figure 9-14. Block Diagram of P50 to P57 (Port 5)



Remark n = 0 to 7

(2) Setting input/output mode and control mode

The input/output mode of port 5 is set by port mode register 5 (PM5). The control mode (external expansion mode) is set by mode specification pins MODE0 and MODE1, and memory expansion mode register (MM: refer to 3.4.6 (1)).

Port 5 mode register (PM5)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM5 | PM57 | PM56 | PM55 | PM54 | PM53 | PM52 | PM51 | PM50 | Address FFFFFF02AH | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM5n (n = 7 to 0) | Port Mode Sets P5n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Operation mode of port 5

| Bit of MM Register | | | Operation Mode | | | | | | | |
|--------------------|-----|-----|-----------------------------------|-----|-----|-----|-----|-----|-----|-----|
| MM2 | MM1 | MM0 | P50 | P51 | P52 | P53 | P54 | P55 | P56 | P57 |
| 0 | 0 | 0 | Port | | | | | | | |
| 0 | 1 | 1 | Address/data bus (AD8 to AD15) | | | | | | | |
| 1 | 0 | 0 | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | |
| Others | | | RFU (reserved) | | | | | | | |

9.3.7 Port 6

Port 6 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P6 | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 | Address FFFFFF0CH | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P6n (n = 7 to 0) | Port 6 I/O port |

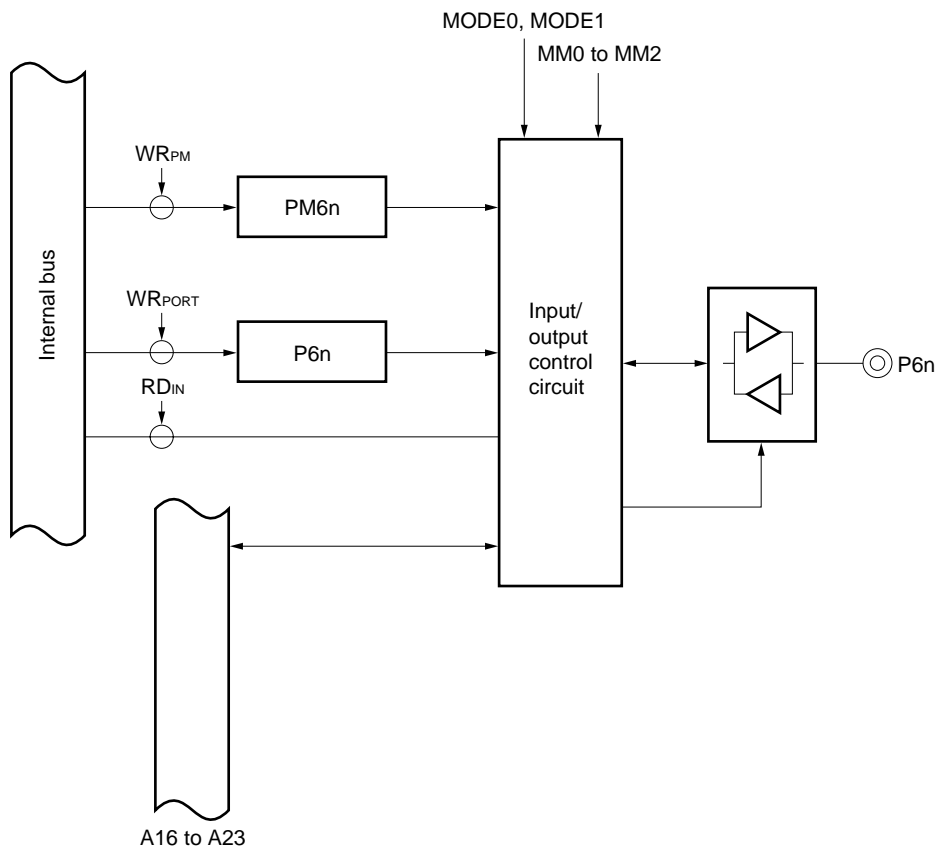
In addition to the function as a general I/O port, this port also serves as an external address bus, when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks |
|--------|--------------|---------------------------------|
| Port 6 | P60 to 67 | A16 to A23 |
| | | Address bus for external memory |

(1) Hardware configuration

Figure 9-15. Block Diagram of P60 to 67 (Port 6)



Remark n = 0 to 7

(2) Setting input/output mode and control mode

The input/output mode of port 6 is set by port mode register 6 (PM6). The control mode (external expansion mode) is set by mode specification pins MODE0 and MODE1, and memory expansion mode register (MM: refer to 3.4.6 (1)).

Port 6 mode register (PM6)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM6 | PM67 | PM66 | PM65 | PM64 | PM63 | PM62 | PM61 | PM60 | Address FFFFFF02CH | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM6n (n = 7 to 0) | Port Mode Sets P6n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Operation mode of port 6

| Bit of MM Register | | | Operation Mode | | | | | | | |
|--------------------|-----|-----|----------------|-----|-----|-----|-----|-----|-----|-----|
| MM2 | MM1 | MM0 | P60 | P61 | P62 | P63 | P64 | P65 | P66 | P67 |
| 0 | 0 | 0 | Port | | | | | | | |
| 0 | 1 | 1 | | | | | | | | |
| 1 | 0 | 0 | A16 | A17 | | | | | | |
| 1 | 0 | 1 | | | A18 | | A19 | | | |
| 1 | 1 | 0 | | | | | A20 | | A21 | |
| 1 | 1 | 1 | | | | | | | A22 | |
| Others | | | RFU (reserved) | | | | | | | |

9.3.8 Port 9

Port 9 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

| | | | | | | | | | | |
|----|-----|-----|-----|-----|-----|-----|-----|-----|-----------------------|-----------------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| P9 | P97 | P96 | P95 | P94 | P93 | P92 | P91 | P90 | Address FFFFFF012H | At reset Undefined |

| Bit Position | Bit Name | Function |
|--------------|---------------------|--------------------|
| 7 to 0 | P9n (n = 7 to 0) | Port 9 I/O port |

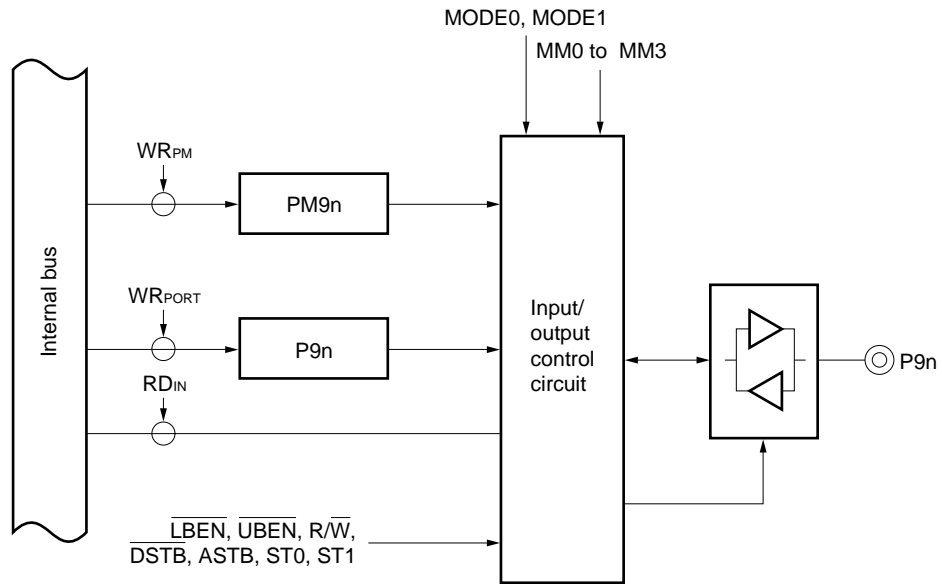
In addition to the function as a general I/O port, this port can also be used to output external bus control signals, when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks | |
|--------|--------------|--------------------------------|---|
| Port 9 | P90 | $\overline{\text{LBEN}}$ | Control signal output for external memory |
| | P91 | $\overline{\text{UBEN}}$ | |
| | P92 | $\text{R}/\overline{\text{W}}$ | |
| | P93 | $\overline{\text{DSTB}}$ | |
| | P94 | ASTB | |
| | P95 | ST0 | |
| | P96 | ST1 | |
| P97 | – | Fixed in port mode | |

(1) Hardware configuration

Figure 9-16. Block Diagram of P90 to P97 (Port 9)



Remark n = 0 to 7

(2) Setting input/output mode and control mode

The input/output mode of port 9 is set by port mode register 9 (PM9). The control mode (external expansion mode) is set by mode specification pins MODE0 and MODE1, and memory expansion mode register (MM: refer to 3.4.6 (1)).

Port 9 mode register (PM9)

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

| | | | | | | | | | | |
|-----|------|------|------|------|------|------|------|------|-----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM9 | PM97 | PM96 | PM95 | PM94 | PM93 | PM92 | PM91 | PM90 | Address FFFFFF032H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|----------------------|---|
| 7 to 0 | PM9n (n = 7 to 0) | Port Mode Sets P9n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Operation mode of port 9

P90 to P94

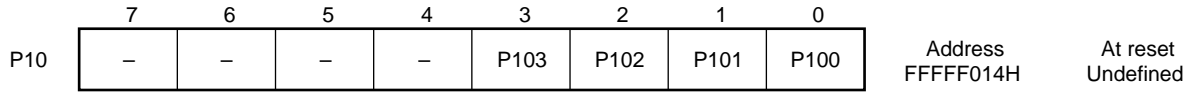
| Bit of MM Register | | | Operation Mode | | | | |
|--------------------|-----|-----|----------------|------|-----|------|------|
| MM2 | MM1 | MM0 | P90 | P91 | P92 | P93 | P94 |
| 0 | 0 | 0 | Port | | | | |
| 0 | 1 | 1 | LBEN | UBEN | R/W | DSTB | ASTB |
| 1 | 0 | 0 | | | | | |
| 1 | 0 | 1 | | | | | |
| 1 | 1 | 0 | | | | | |
| 1 | 1 | 1 | | | | | |
| Others | | | RFU (reserved) | | | | |

P95, P96

| MM3 | Operation Mode | P95 | P96 |
|-----|-------------------------|------|-----|
| 0 | Port mode | Port | |
| 1 | External expansion mode | ST0 | ST1 |

9.3.9 Port 10

Port 10 is a 4-bit input/output port that can be set in the input or output mode in 1-bit units.



| Bit Position | Bit Name | Function |
|--------------|----------------------|---------------------|
| 3 to 0 | P10n (n = 3 to 0) | Port 10 I/O port |

When port 10 is accessed in 8-bit units for write, the higher 4 bits are ignored. When it is accessed in 8-bit units for read, undefined data is read.

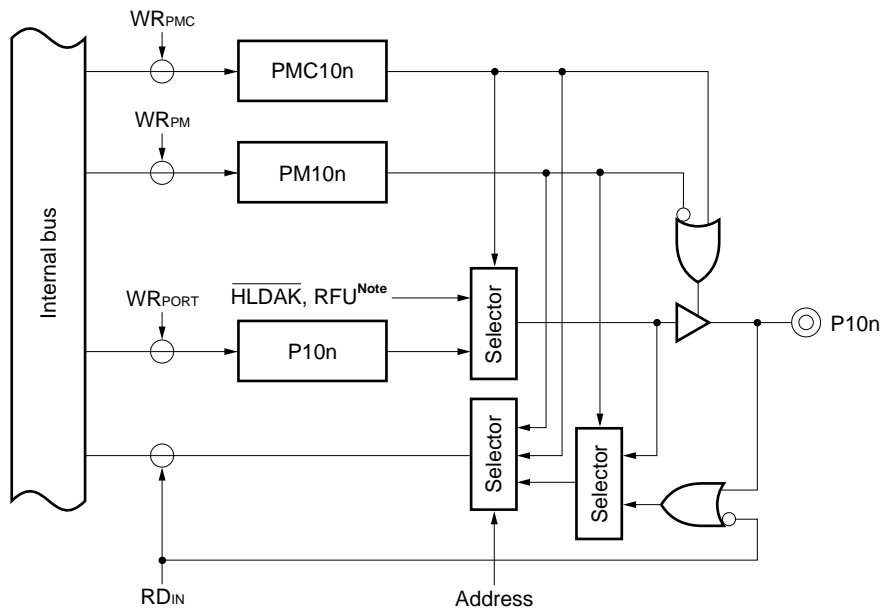
In addition to the function as a port, this port can also be used to input and output external control signals to a bus master or ASIC device, when placed in the control mode.

Operation in control mode

| Port | Control Mode | Remarks |
|---------|--------------|--------------------------------------|
| Port 10 | P100 | $\overline{\text{HLDAK}}$ |
| | P101 | $\overline{\text{HLDRQ}}$ |
| | P102, P103 | – |
| | | Bus hold control signal input/output |
| | | Fixed in port mode |

(1) Hardware configuration

Figure 9-17. Block Diagram of P100, P103 (Port 10)



Note RFU is an undefined value.

Remark n = 0, 3

Figure 9-18. Block Diagram of P101 (Port 10)

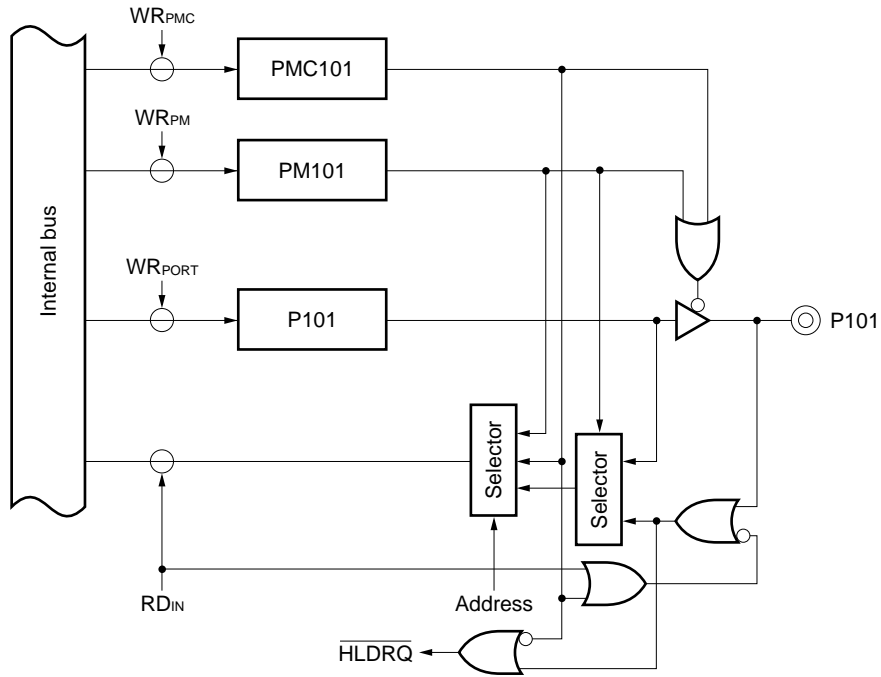
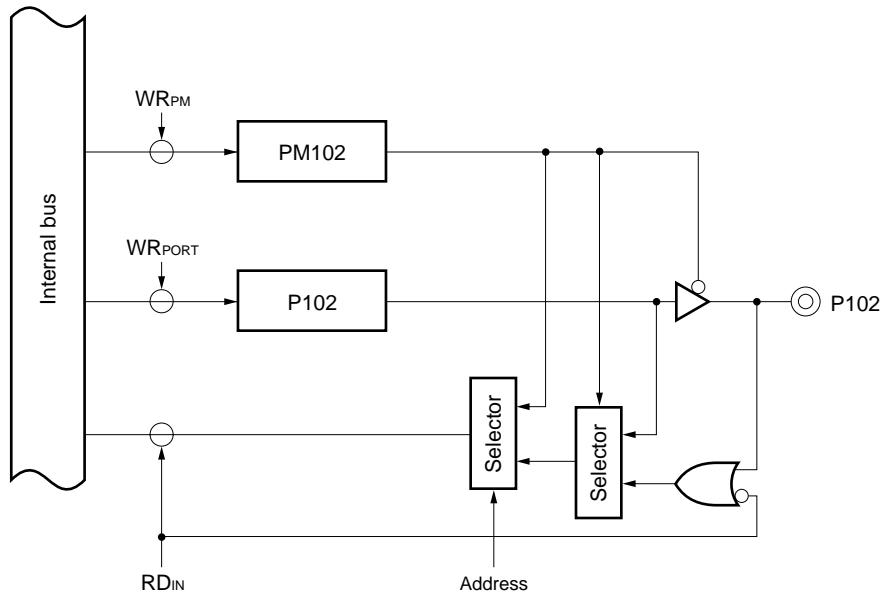


Figure 9-19. Block Diagram of P102 (Port 10)



(2) Setting input/output mode and control mode

The input/output mode of port 10 is set by port mode register 10 (PM10). The control mode is set by port mode control register 10 (PMC10).

Port 10 mode register (PM10)

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

| | | | | | | | | | | |
|------|---|---|---|---|-------|-------|-------|-------|----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PM10 | 1 | 1 | 1 | 1 | PM103 | PM102 | PM101 | PM100 | Address FFFFF034H | At reset FFH |

| Bit Position | Bit Name | Function |
|--------------|-----------------------|--|
| 3 to 0 | PM10n (n = 3 to 0) | Port Mode Sets P10n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF) |

Port 10 mode control register (PMC10)

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

| | | | | | | | | | | |
|-------|---|---|---|---|---|---|--------|--------|----------------------|-----------------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| PMC10 | 0 | 0 | 0 | 0 | 0 | 0 | PMC101 | PMC100 | Address FFFFF054H | At reset 00H |

| Bit Position | Bit Name | Function |
|--------------|----------|--|
| 1 | PMC101 | Port Mode Control Sets operation mode of P101 pin. 0: I/O port mode 1: $\overline{\text{HLDRQ}}$ input mode |
| 0 | PMC100 | Port Mode Control Sets operation mode of P100 pin. 0: I/O port mode 1: $\overline{\text{HLD\text{A}K}}$ output mode |

9.4 Noise Elimination Circuit

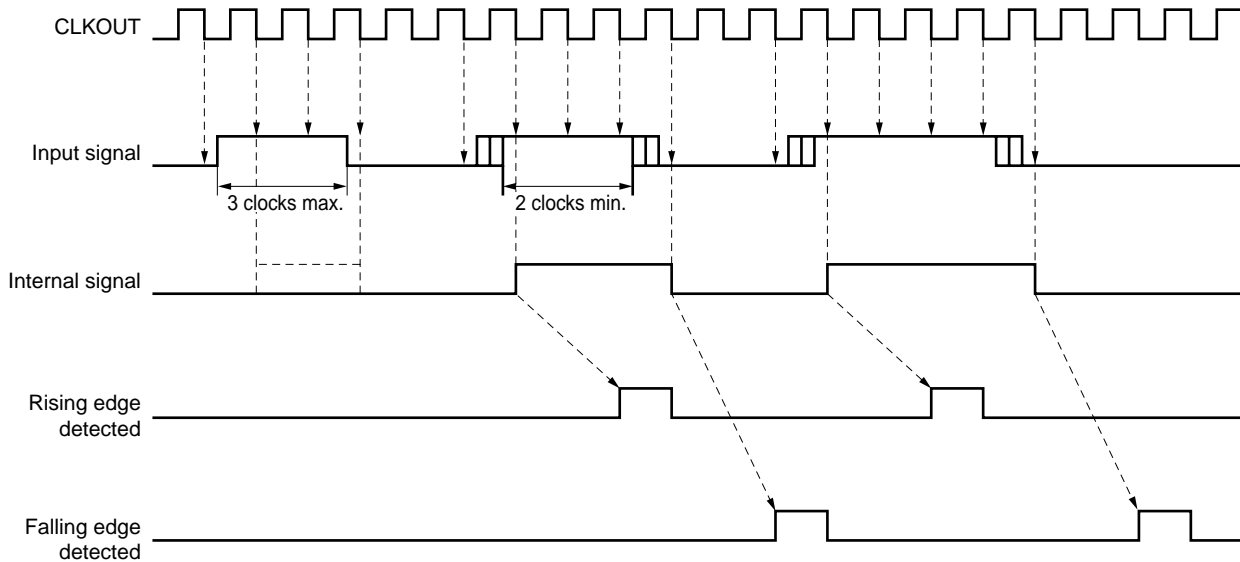
Pins operating with valid edge inputs in the control mode are provided with timing control circuits for maintaining the following noise elimination time.

A signal input changed less than the noise elimination time is not accepted internally.

| Pin | Noise Elimination Time |
|-------------------------|--------------------------------|
| P20/NMI ^{Note} | Analog delay (60 ns to 220 ns) |
| P02/TCLR1 | 2 to 3 system clocks |
| P03/T11 | |
| P04/INTP10 | |
| P05/INTP11 | |
| P06/INTP12 | |
| P07/INTP13 | |
| P21/INTP00 | |
| P22/INTP01 | |
| P23/INTP02 | |
| P24/INTP03 | |

Note The P20/NMI pin is used to release the STOP mode. In the STOP mode, the clock control timing circuit is not used because the clock is stopped.

Figure 9-20. Example of Noise Elimination Timing



CHAPTER 10 RESET FUNCTION

When the low-level is input to the $\overline{\text{RESET}}$ pin, the system is reset and each on-chip hardware is initialized to the initial state.

When the $\overline{\text{RESET}}$ pin changes from low-level to high-level, the reset state is released and the CPU starts executing the program. Initialize the contents of each register in the program as necessary.

10.1 Features

- Analog noise elimination circuit (delay of approx. 60 to 220 ns) provided on reset pin

10.2 Pin Function

During the reset state, all the pins (except CLKOUT, $\overline{\text{RESET}}$, X2, V_{DD}, V_{SS}, CV_{DD}, and CV_{SS} pins) are in the high-impedance state.

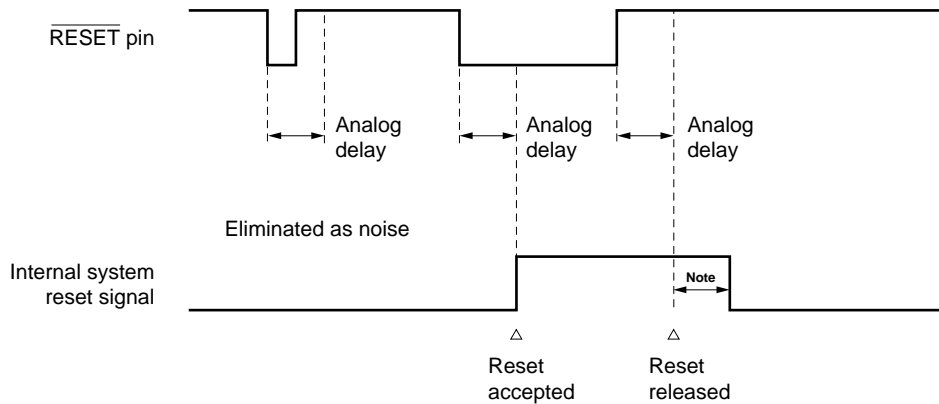
When an external memory is connected, a pull-up (or pull-down) resistor must be connected to each pin of ports 4, 5, 6, and 9. Otherwise, the memory contents may be lost if these pins go into a high-impedance state. Also treat signal outputs of the on-chip peripheral I/O function and the output port so that they will not be affected.

The internal system clock continues to generate the clock signal at CLKOUT pin while the device is in the reset state.

Table 10-1 shows the operating status of each pin during the reset period.

Table 10-1. Operating Status of Each Pin During Reset Period

| Pin | Operating Status |
|---|------------------|
| AD0 to AD15 | Hi-Z |
| A16 to A23 | |
| $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$ | |
| R/W | |
| $\overline{\text{DSTB}}$ | |
| ASTB | |
| ST0, ST1 | |
| HLD $\overline{\text{RQ}}$ | – |
| $\overline{\text{HLDAK}}$ | Hi-Z |
| $\overline{\text{WAIT}}$ | – |
| CLKOUT | Clock output |

(1) Accepting reset signal

Note The internal system reset signal remains active for the duration of at least 4 system clocks after the reset condition is removed from the $\overline{\text{RESET}}$ pin.

(2) Power-ON reset

In the reset operations at power-on (power is turned on), there is a need to maintain oscillation stabilization time of more than 10 ms from the start up of the power until reset is acknowledged to the low-level width of the $\overline{\text{RESET}}$ pin.

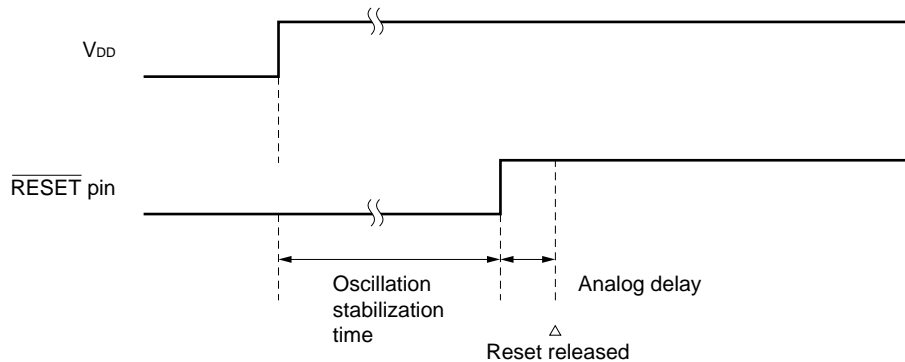
**10.3 Initialize**

Table 10-2 shows the initial value of each register after reset.

The contents of the registers must be initialized in the program as necessary. Especially, set the following registers as necessary because they are related to system setting:

- Power save control register (PSC) ... X1 and X2 pin function, CLKOUT pin operation, etc.
- Data wait control register (DWC) ... Number of data wait states

Caution In Table 10-2, “Undefined” means an undefined value due to power-on reset or data corruption when a falling edge of $\overline{\text{RESET}}$ coincides with a data write operation. The previous status of data is retained by a falling edge of $\overline{\text{RESET}}$ due to the cases other than the above.

Table 10-2. Initial Values of Each Register at Reset

| Register | | Initial Value at Reset |
|---|---|------------------------|
| r0 | | 00000000H |
| r1 to r31 | | Undefined |
| PC | | 00000000H |
| PSW | | 00000020H |
| EIPC | | Undefined |
| EIPSW | | Undefined |
| FEPC | | Undefined |
| FEPSW | | Undefined |
| ECR | | 00000000H |
| Internal RAM | | Undefined |
| ★ Port | Input/output latch (P0 to P6, P9, P10) | Undefined |
| | Mode register (PM0 to 6, PM9, PM10) | FFH |
| | Mode control register (PMC0, PMC3, PMC10) (PMC2) | 00H 01H |
| | Memory expansion mode register (MM) | B0H or B7H |
| Clock generator | System status register (SYS) | 0000000xB |
| | Power save control register (PSC) | 00H |
| Real-time pulse unit | Timer unit mode register (TUM1) | 0000H |
| | Timer control register (TMC1, TMC4) | 00H |
| | Timer output control register 1 (TOC1) | 00H |
| | Timer (TM1, TM4) | 0000H |
| | Capture/compare register (CC10 to CC13) | Undefined |
| | Compare register 4 (CM4) | Undefined |
| | Timer overflow status register (TOVS) | 00H |
| Serial interface | Asynchronous serial interface mode register 00 (ASIM00) | 80H |
| | Asynchronous serial interface mode register 01 (ASIM01) | 00H |
| | Asynchronous serial interface status register 0 (ASIS0) | 00H |
| | Receive buffer (RXB0, RXB0L) | Undefined |
| | Transmit shift register (TXS0, TXS0L) | Undefined |
| | Clocked serial interface mode register n (CSIMn) (n = 0 to 2) | 00H |
| | Serial I/O shift register n (SIO n) (n = 0 to 2) | Undefined |
| | Baud rate generator register 0, 1 (BRG0, BRG1) | Undefined |
| | Baud rate generator prescaler mode register 0, 1 (BPRM0, BPRM1) | 00H |
| Interrupt/exception processing function | Interrupt control register (xxCn) | 47H |
| | In-service priority register (ISPR) | 00H |
| | External interrupt mode register (INTM0, INTM1, INTM2) | 00H |
| Memory management function | Data wait control register (DWC) | FFFFH |
| | Bus cycle control register (BCC) | AAAAH |
| Power save control | Command register (PRCMD) | Undefined |
| | Power save control register (PSC) | 00H |

Remark x: undefined

Caution In the above table, “undefined” means the undefined value after power-on reset, or the undefined value caused by data destruction due to synchronization of $\overline{\text{RESET}} \downarrow$ input and data write timings. With other $\overline{\text{RESET}} \downarrow$ inputs, the data immediately before $\overline{\text{RESET}} \downarrow$ input is maintained.

CHAPTER 11 PROM MODE

The PROM model of the V852 is provided with 90 KB of one-time PROM. Instruction fetch to internal ROM is accessed in 1 clock in the same manner as the mask ROM model.

11.1 PROM Mode

The PROM mode is entered by the setting MODE0 and MODE1 pins.

Connect the pins not used in this mode as described in section 1.5.2 “PROM programming mode”.

| V _{PP} | MODE1 | MODE0 | Operation Mode |
|-----------------|-------|-------|------------------------------|
| 5.0 V | 1 | 1 | PROM mode (read mode) |
| 12.5 V | 1 | 1 | PROM mode (programming mode) |

V_{PP}: programming voltage

11.2 Operation Mode

Operation in the PROM programming mode is determined by the setting of the pins shown in the following table.

| Operation Mode | | Pin | P25/ $\overline{\text{CE}}$ | P26/ $\overline{\text{OE}}$ | P27/ $\overline{\text{PGM}}$ | V _{PP} | V _{DD} | P47/D7 to P40/D0 |
|------------------|-----------------|-----|-----------------------------|-----------------------------|------------------------------|-----------------|-----------------|----------------------|
| Read Mode | Read | | L | L | H | +5.0 V | +5.0 V | Data output |
| | Output disable | | L | H | x | | | Hi-Z ^{Note} |
| | Standby | | H | x | x | | | |
| Programming Mode | Page data latch | | H | L | H | +12.5 V | +6.5 V | Data input |
| | Page program | | H | H | L | | | Hi-Z |
| | Byte program | | L | H | L | | | Data input |
| | Program verify | | L | L | H | | | Data output |
| | Program inhibit | | x | L | L | | | Hi-Z ^{Note} |
| | | | H | H | | | | |

V_{PP}: programming voltage (12.5 V)

x : optional

Note In this case, the address input is invalid, and 1/0 can be input.

(1) Read mode

The read mode is set when $\overline{\text{CE}} = \text{L}$ and $\overline{\text{OE}} = \text{L}$.

(2) Output disable mode

- ★ The data output goes into a high-impedance state when $\overline{CE} = L$ and $\overline{OE} = H$, and the output disable mode is set. If two or more $\mu\text{PD70P3002s}$ are connected to the data bus, any one of the devices can be read by controlling the \overline{OE} pin.

(3) Standby mode

The standby mode is set when $\overline{CE} = H$.

In this mode, the data output goes into a high-impedance state regardless of the status of \overline{OE} .

(4) Page data latch mode

The page data latch mode is set when $\overline{CE} = H$, $\overline{OE} = L$, and $\overline{PGM} = H$ at the beginning of the page write mode.

In this mode, data of 1 page and 4 bytes is latched to the internal address/data latch circuit.

(5) Page write mode

Page write is executed in the page write mode by applying a 0.1-ms program pulse (active low) to the \overline{PGM} pin with $\overline{CE} = H$ and $\overline{OE} = H$, after an address and data of 1 page and 4 bytes have been latched. After that, the program can be verified when $\overline{CE} = L$ and $\overline{OE} = L$.

If the program cannot be written by one program pulse, write and verify are repeatedly executed X times ($X \leq 10$).

(6) Byte write mode

Byte write is executed by applying a 0.1-ms program pulse (active low) to the \overline{PGM} pin with $\overline{CE} = L$ and $\overline{OE} = H$. After that, the program can be verified when $\overline{OE} = L$.

If the program cannot be written by one program pulse, write and verify are repeatedly executed X times ($X \leq 10$).

(7) Program verify mode

The program verify mode is set by setting $\overline{CE} = L$, $\overline{PGM} = H$, and $\overline{OE} = L$. Use this mode to check if the program has been correctly written.

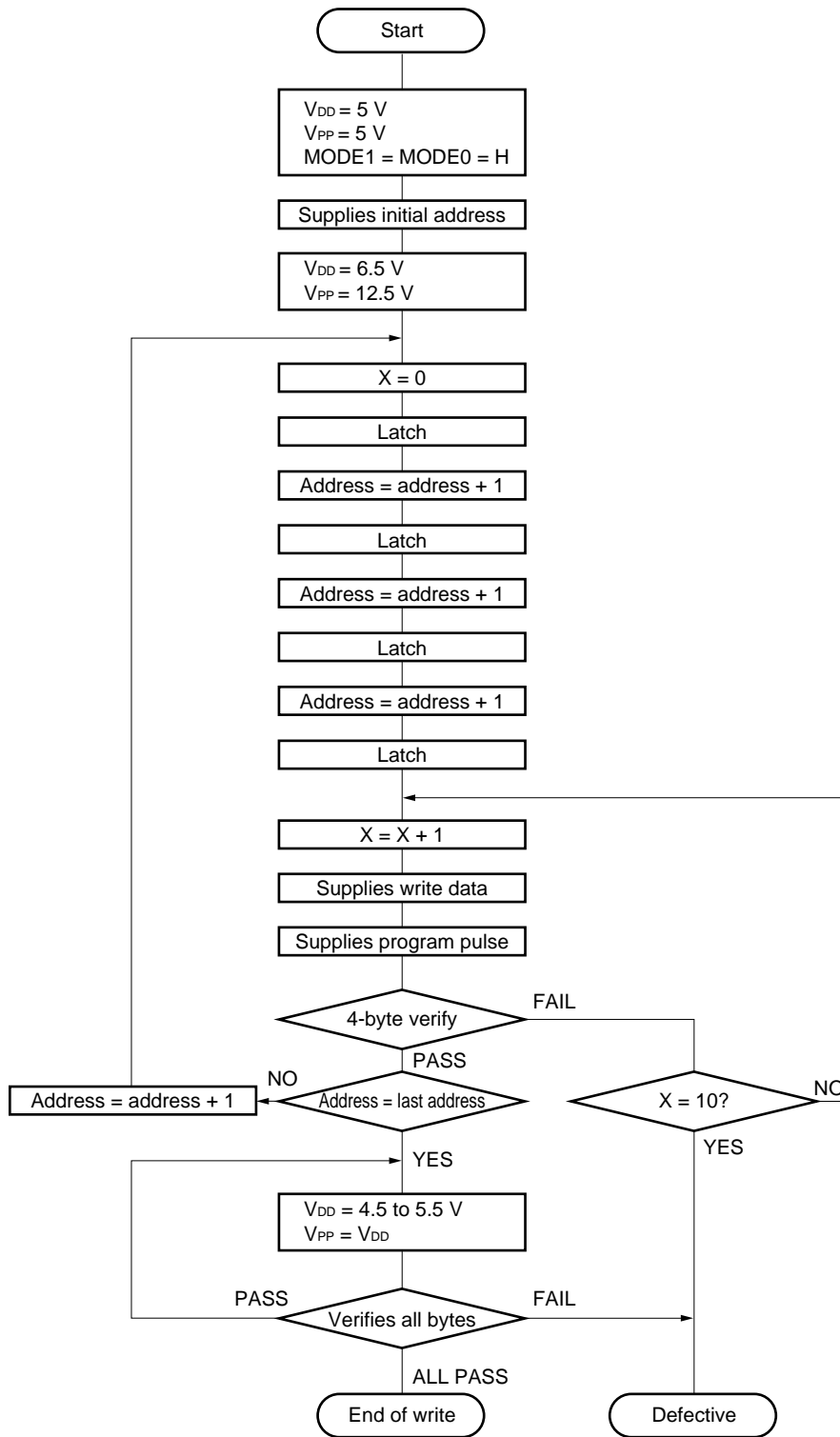
(8) Program inhibit mode

The program inhibit mode is used to write data to one of the $\mu\text{PD70P3002s}$ whose \overline{OE} , V_{PP} , and D0 to D7 pins are connected in parallel.

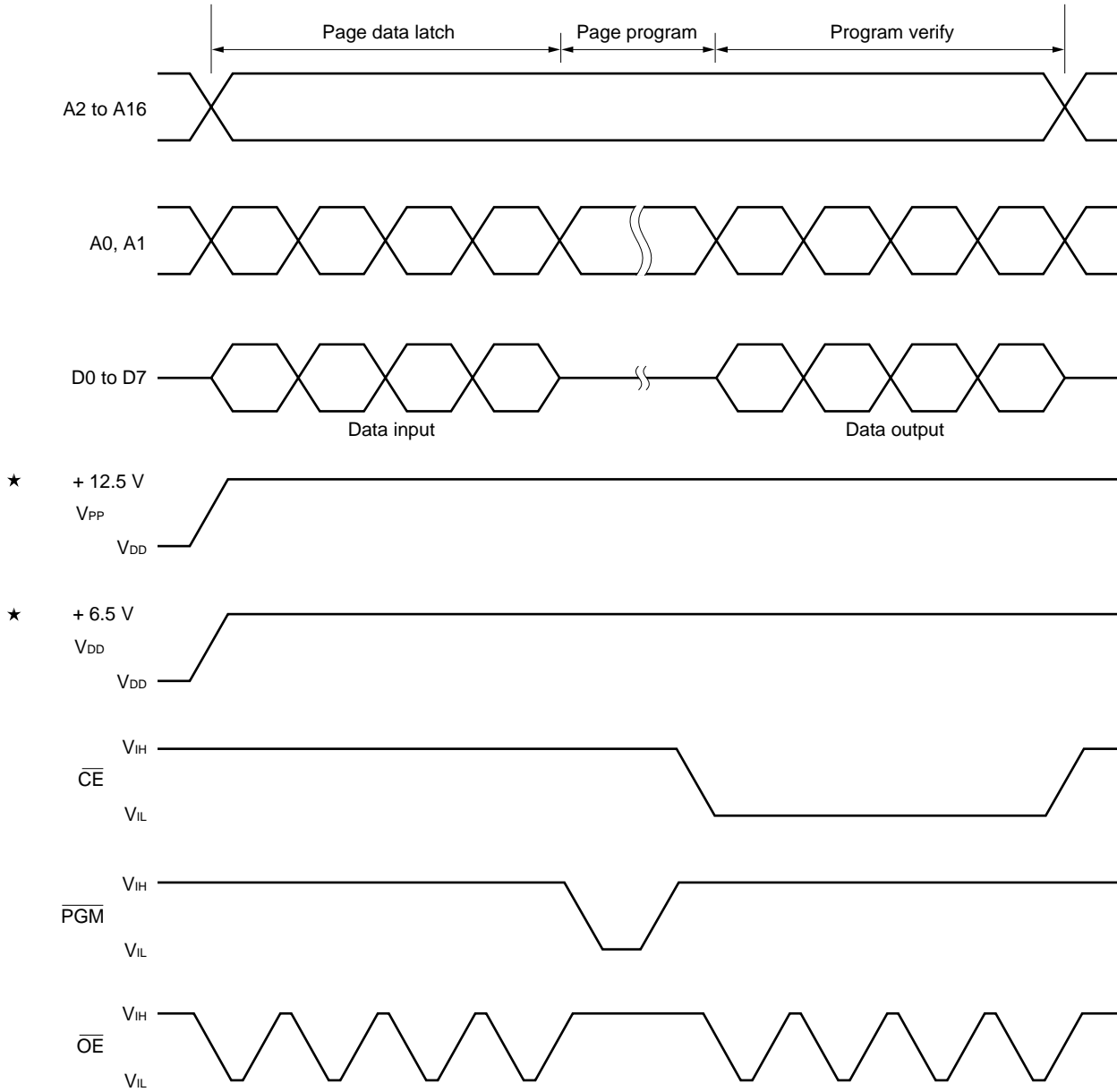
To write data, the page write mode or byte write mode is used. At this time, data cannot be written to a device whose \overline{PGM} pin is high.

11.3 PROM Write Procedure

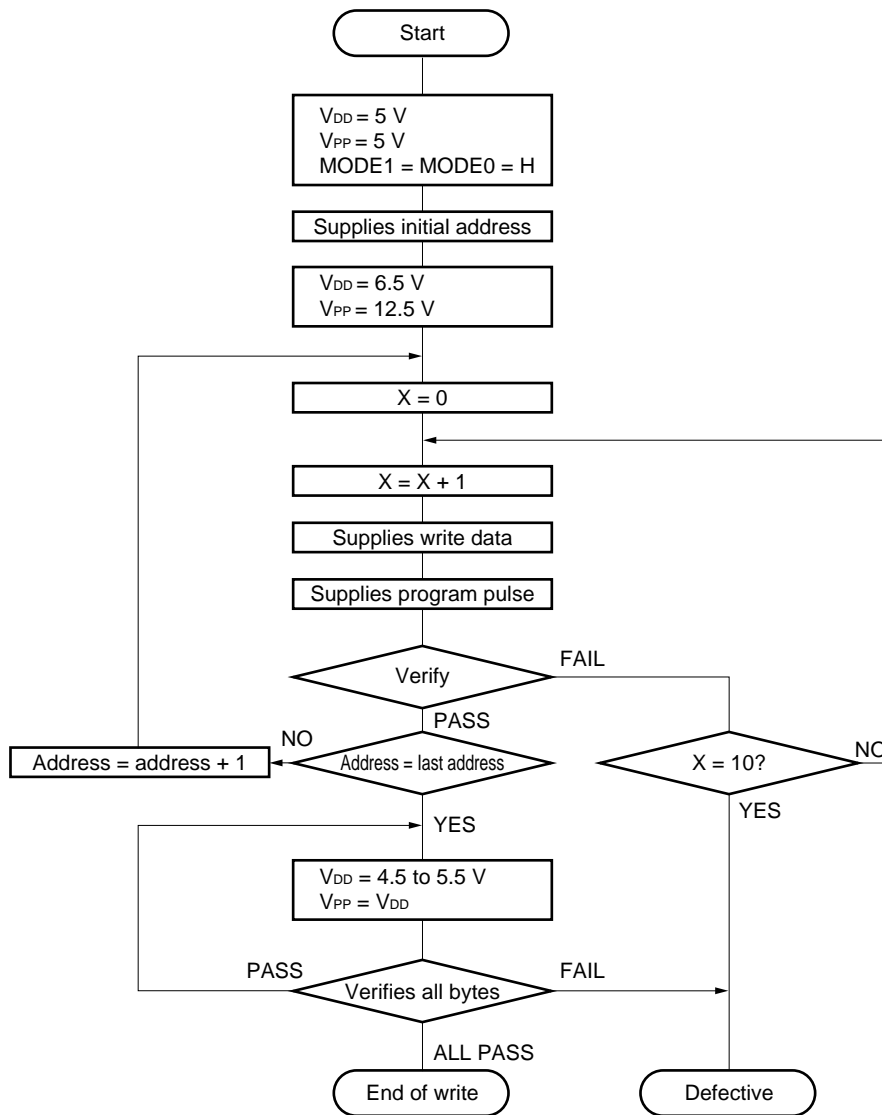
Page programming mode flowchart



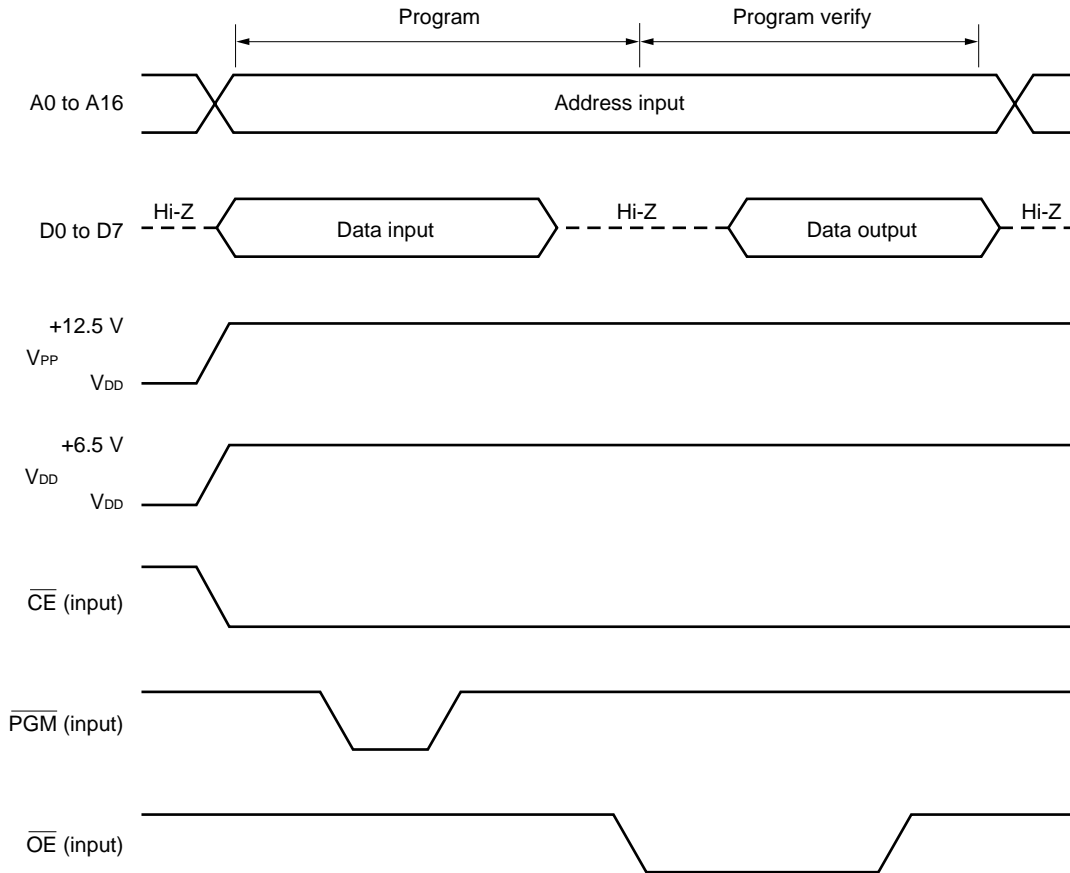
Page programming mode timing



Byte programming mode flowchart



Byte programming mode timing



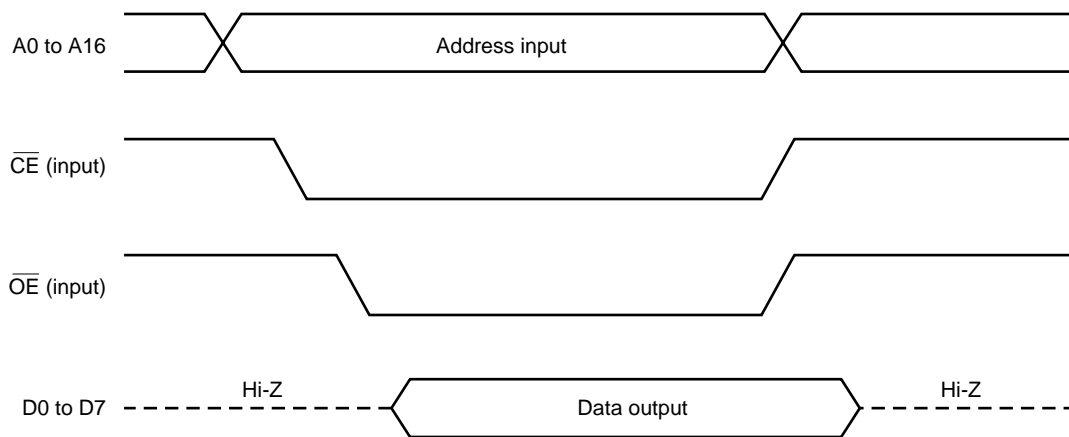
11.4 PROM Read Procedure

The contents of the PROM are read to the external data bus (D0 to D7) in the following procedure:

- ★ (1) Fix $\text{MODE0} = \text{L}$, and $\text{MODE1} = \text{L}$. Connect the unused pins as described in **1.5.2 PROM programming mode**.
- (2) Supply +5 V to the V_{DD} and V_{PP} pins.
- (3) Input the address of the data to be read to the A0 to A16 pins.
- (4) Read mode ($\overline{\text{CE}} = \text{L}$, $\overline{\text{OE}} = \text{L}$)
- (5) The data is output to the D0 to D7 pins.

Figure 11-1 shows the timing of (2) to (5) above.

Figure 11-1. PROM Read Timing



11.5 Screening of OTPROM Version

The one-time-programmable ROM (OTPROM) version, μ PD70P3002GC-7EA, cannot be completely tested by NEC before shipment. It is recommended to perform screening to verify the PROM after the PROM has been stored under the following conditions:

| Storage Temperature | Storage Time |
|---------------------|--------------|
| 125 °C | 24 hours |

11.6 Caution on STOP Mode Release when Using External Clock

When using the external clock, the clock supply is controlled by an external system.

Therefore, when releasing the STOP mode (released by $\overline{\text{RESET}}$ or NMI input), restart the clock supply more than 150 μ s before $\overline{\text{RESET}}$ or NMI input to secure the PROM stabilization time.

APPENDIX A REGISTER INDEX

| Symbol | Name | Unit | Page |
|---------|---|------|------|
| ASIM00 | Asynchronous serial interface mode register 00 | UART | 151 |
| ASIM01 | Asynchronous serial interface mode register 01 | UART | 153 |
| ASIS0 | Asynchronous serial interface status register 0 | UART | 154 |
| BCC | Bus cycle control register | BCU | 56 |
| BPRM0 | Baud rate generator prescaler mode register 0 | BRG0 | 176 |
| BPRM1 | Baud rate generator prescaler mode register 1 | BRG1 | 176 |
| BRG0 | Baud rate generator register 0 | BRG0 | 176 |
| BRG1 | Baud rate generator register 1 | BRG1 | 176 |
| CC10 | Capture/compare register 10 | RPU | 119 |
| CC11 | Capture/compare register 11 | RPU | 119 |
| CC12 | Capture/compare register 12 | RPU | 119 |
| CC13 | Capture/compare register 13 | RPU | 119 |
| CM4 | Compare register 4 | RPU | 120 |
| CMIC4 | Interrupt control register | INTC | 83 |
| CSIC0 | Interrupt control register | INTC | 83 |
| ★ CSIC1 | Interrupt control register | INTC | 83 |
| ★ CSIC2 | Interrupt control register | INTC | 83 |
| CSIM0 | Clocked serial interface mode register 0 | CSI0 | 164 |
| CSIM1 | Clocked serial interface mode register 1 | CSI1 | 164 |
| CSIM2 | Clocked serial interface mode register 2 | CSI2 | 164 |
| DWC | Data wait control register | BCU | 54 |
| ECR | Interrupt source register | CPU | 30 |
| EIPC | Interrupt status save register | CPU | 30 |
| EIPSW | Interrupt status save register | CPU | 30 |
| FEPC | NMI status save register | CPU | 30 |
| FEPSW | NMI status save register | CPU | 30 |
| INTM0 | External interrupt mode register 0 | INTC | 73 |
| INTM1 | External interrupt mode register 1 | INTC | 84 |
| INTM2 | External interrupt mode register 2 | INTC | 84 |
| ISPR | In-service priority register | INTC | 85 |
| MM | Memory expansion mode register | Port | 46 |
| OVIC1 | Interrupt control register | INTC | 83 |
| P0 | Port 0 | Port | 182 |
| P1 | Port 1 | Port | 186 |
| P2 | Port 2 | Port | 187 |

| Symbol | Name | Unit | Page |
|--------|-------------------------------|------|-------------|
| P3 | Port 3 | Port | 192 |
| P4 | Port 4 | Port | 197 |
| P5 | Port 5 | Port | 199 |
| P6 | Port 6 | Port | 201 |
| P9 | Port 9 | Port | 202 |
| P10 | Port 10 | Port | 205 |
| P0IC0 | Interrupt control register | INTC | 83 |
| P0IC1 | Interrupt control register | INTC | 83 |
| P0IC2 | Interrupt control register | INTC | 83 |
| P0IC3 | Interrupt control register | INTC | 83 |
| P1IC0 | Interrupt control register | INTC | 83 |
| P1IC1 | Interrupt control register | INTC | 83 |
| P1IC2 | Interrupt control register | INTC | 83 |
| P1IC3 | Interrupt control register | INTC | 83 |
| PM0 | Port 0 mode register | Port | 184 |
| PM1 | Port 1 mode register | Port | 187 |
| PM2 | Port 2 mode register | Port | 191 |
| PM3 | Port 3 mode register | Port | 195 |
| PM4 | Port 4 mode register | Port | 198 |
| PM5 | Port 5 mode register | Port | 200 |
| PM6 | Port 6 mode register | Port | 202 |
| PM9 | Port 9 mode register | Port | 204 |
| PM10 | Port 10 mode register | Port | 207 |
| PMC0 | Port 0 mode control register | Port | 185 |
| PMC2 | Port 2 mode control register | Port | 192 |
| PMC3 | Port 3 mode control register | Port | 196 |
| PMC10 | Port 10 mode control register | Port | 207 |
| PRCMD | Command register | CG | 103 |
| PSC | Power save control register | CG | 101 |
| PSW | Program status word | CPU | 31,73,85,88 |
| RXB0 | Receive buffer 0 | UART | 155 |
| RXB0L | Receive buffer 0L | UART | 155 |
| SEIC0 | Interrupt control register | INTC | 83 |
| SIO0 | Serial I/O shift register 0 | CSI0 | 165 |
| SIO1 | Serial I/O shift register 1 | CSI1 | 165 |
| SIO2 | Serial I/O shift register 2 | CSI2 | 165 |
| SRIC0 | Interrupt control register | INTC | 83 |

| Symbol | Name | Unit | Page |
|--------|---------------------------------|------|---------|
| STIC0 | Interrupt control register | INTC | 83 |
| SYS | System status register | CG | 98, 103 |
| TM1 | Timer 1 | RPU | 118 |
| TM4 | Timer 4 | RPU | 120 |
| TMC1 | Timer control register 1 | RPU | 123 |
| TMC4 | Timer control register 4 | RPU | 124 |
| TOC1 | Timer output control register 1 | RPU | 125 |
| TOVS | Timer overflow status register | RPU | 126 |
| TUM1 | Timer unit mode register 1 | RPU | 121 |
| TXS0 | Transmit shift register 0 | UART | 156 |
| TXS0L | Transmit shift register 0L | UART | 156 |

Phase-out/Discontinued

[MEMO]

APPENDIX B INSTRUCTION SET LIST

Legend

(1) Symbols used for operand description

| Symbol | Description |
|--------|---|
| reg1 | General register (r0 to r31): Used as source register |
| reg2 | General register (r0 to r31): Mainly used as destination register |
| immx | x-bit immediate |
| dispx | x-bit displacement |
| regID | System register number |
| bit#3 | 3-bit data for bit number specification |
| ep | Element pointer (r30) |
| cccc | Condition code |
| vector | 5-bit data specifying trap vector (00H-1FH) |

(2) Symbols used for code

| Symbol | Description |
|--------|---|
| R | 1-bit data of code specifying reg1 or regID |
| r | 1-bit data of code specifying reg2 |
| d | 1-bit data of displacement |
| i | 1-bit data of immediate |
| cccc | 4-bit data for condition code specification |
| bbb | 3-bit data for bit number specification |

(3) Symbols used for operation description

| Symbol | Description |
|--------------------------|---|
| <- | Assignment |
| GR[] | General register |
| SR[] | System register |
| zero-extend (n) | Zero-extends n to word length |
| sign-extend (n) | Sign-extends n to word length |
| load-memory (a,b) | Reads data of size b from address a |
| store-memory (a,b,c) | Writes data b of size c to address a |
| load-memory-bit (a,b) | Reads bit b of address a |
| store-memory-bit (a,b,c) | Writes c to bit b of address a |
| saturated (n) | Performs saturated processing of n (n is 2's complement). If n is $n \geq 7FFFFFFH$ as result of calculation, $7FFFFFFH$. If n is $n \leq 80000000H$ as result of calculation, $80000000H$. |
| result | Reflects result on flag |
| Byte | Byte (8 bits) |

| Symbol | Description |
|-------------------------------|------------------------|
| Halfword | Half-word (16 bits) |
| Word | Word (32 bits) |
| + | Add |
| – | Subtract |
| | Bit concatenation |
| x | Multiply |
| ÷ | Divide |
| AND | Logical product |
| OR | Logical sum |
| XOR | Exclusive logical sum |
| NOT | Logical negate |
| logically shift left by | Logical left shift |
| logically shift right by | Logical right shift |
| arithmetically shift right by | Arithmetic right shift |

(4) Symbols used for execution clock description

| Symbol | Description |
|-------------|--|
| i : issue | To execute another instruction immediately after instruction execution |
| r : repeat | To execute same instruction immediately after instruction execution |
| l : latency | To reference result of instruction execution by the next instruction |

(5) Symbols used for flag operation

| Identifier | Description |
|------------|------------------------------------|
| (Blank) | Not affected |
| 0 | Cleared to 0 |
| x | Set or cleared according to result |
| R | Previously saved value is restored |

Condition code

| Condition Name (cond) | Condition Code (cccc) | Conditional Expression | Description |
|--------------------------|--------------------------|---|--|
| V | 0 0 0 0 | $OV = 1$ | Overflow |
| NV | 1 0 0 0 | $OV = 0$ | No overflow |
| C/L | 0 0 0 1 | $CY = 1$ | Carry Lower (Less than) |
| NC/NL | 1 0 0 1 | $CY = 0$ | No carry No lower (Greater than or equal) |
| Z/E | 0 0 1 0 | $Z = 1$ | Zero Equal |
| NZ/NE | 1 0 1 0 | $Z = 0$ | Not zero Not equal |
| NH | 0 0 1 1 | $(CY \text{ OR } Z) = 1$ | Not higher (Less than or equal) |
| H | 1 0 1 1 | $(CY \text{ OR } Z) = 0$ | Higher (Greater than) |
| N | 0 1 0 0 | $S = 1$ | Negative |
| P | 1 1 0 0 | $S = 0$ | Positive |
| T | 0 1 0 1 | — | Always (unconditional) |
| SA | 1 1 0 1 | $SAT = 1$ | Saturated |
| LT | 0 1 1 0 | $(S \text{ XOR } OV) = 1$ | Less than signed |
| GE | 1 1 1 0 | $(S \text{ XOR } OV) = 0$ | Greater than or equal signed |
| LE | 0 1 1 1 | $((S \text{ XOR } OV) \text{ OR } Z) = 1$ | Less than or equal signed |
| GT | 1 1 1 1 | $((S \text{ XOR } OV) \text{ OR } Z) = 0$ | Greater than signed |

Instruction Set (alphabetical order) (1/4)

| Mnemonic | Operand | Code | Operation | Execution Clock | | | Flag | | | | | |
|----------|---------------------|---|---|------------------------------|----|----|------|----|---|---|-----|--|
| | | | | i | r | l | CY | OV | S | Z | SAT | |
| ADD | reg1, reg2 | r r r r r 001110RRRRR | GR[reg2]<-GR[reg2]+GR[reg1] | 1 | 1 | 1 | x | x | x | x | | |
| | imm5, reg2 | r r r r r 010010 i i i i i | GR[reg2]<-GR[reg2]+sign-extend(imm5) | 1 | 1 | 1 | x | x | x | x | | |
| ADDI | imm16, reg1, reg2 | r r r r r 110000RRRRR i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1]+sign-extend(imm16) | 1 | 1 | 1 | x | x | x | x | | |
| AND | reg1, reg2 | r r r r r 001010RRRRR | GR[reg2]<-GR[reg2]AND GR[reg1] | 1 | 1 | 1 | | 0 | x | x | | |
| ANDI | imm16, reg1, reg2 | r r r r r 110110RRRRR i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1]AND zero-extend(imm16) | 1 | 1 | 1 | | 0 | 0 | x | | |
| Bcond | disp9 | d d d d d 1011 d d c c c c Note 1 | if conditions are satisfied | When condition satisfied | 3 | 3 | 3 | | | | | |
| | | | then PC<-PC+sign-extned(disp9) | When condition not satisfied | 1 | 1 | 1 | | | | | |
| CLR1 | bit#3, disp16[reg1] | 10 b b b 111110RRRRR d d d d d d d d d d d d d d d d | adr<-GR[reg1]+sign-extend(disp16) Z flag<-Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3.0) | 4 | 4 | 4 | | | | | x | |
| CMP | reg1, reg2 | r r r r r 001111RRRRR | result<-GR[reg2]-GR[reg1] | 1 | 1 | 1 | x | x | x | x | | |
| | imm5, reg2 | r r r r r 010011 i i i i i | result<-GR[reg2]-sign-extend(imm5) | 1 | 1 | 1 | x | x | x | x | | |
| DI | | 000001111100000 0000000101100000 | PSW.ID<-1 (Maskable interrupt disabled) | 1 | 1 | 1 | | | | | | |
| DIVH | reg1, reg2 | r r r r r 000010RRRRR | GR [reg2]<-GR [reg2]+GR [reg1] ^{Note2} (signed division) | 36 | 36 | 36 | | x | x | x | | |
| EI | | 100001111100000 0000000101100000 | PSW.ID<-0 (Maskable interrupt enabled) | 1 | 1 | 1 | | | | | | |
| HALT | | 000001111100000 0000000100100000 | Stops | 1 | 1 | 1 | | | | | | |
| JARL | disp22, reg2 | r r r r r 11110 d 0 Note 3 | GR[reg2]<-PC+4 PC<-PC+sign-extend(disp22) | 3 | 3 | 3 | | | | | | |
| JMP | [reg1] | 0000000011RRRRR | PC<-GR[reg1] | 3 | 3 | 3 | | | | | | |
| JR | disp22 | 0000011110 d 0 Note 3 | PC<-PC+sign-extend(disp22) | 3 | 3 | 3 | | | | | | |
| LD.B | disp16[reg1], reg2 | r r r r r 111000RRRRR d d d d d d d d d d d d d d d d | adr<-GR[reg1]+sign-extend(disp16) GR[reg2]<-sign-extend(Load-memory(adr, Byte)) | 1 | 1 | 2 | | | | | | |
| LD.H | disp16[reg1], reg2 | r r r r r 111001RRRRR d d d d d d d d d d d d d d d 0 Note 4 | adr<-GR[reg1]+sign-extend(disp16) GR[reg2]sign-extend(Load-memory(adr, Halfword)) | 1 | 1 | 2 | | | | | | |
| LD.W | disp16p[reg1], reg2 | r r r r r 111001RRRRR d d d d d d d d d d d d d d d 1 Note 4 | adr<-GR[reg1]+sign-extend(disp16) GR[reg2]<-Load-memory(adr, Word) | 1 | 1 | 2 | | | | | | |

- Notes**
1. d d d d d d d d is the higher 8 bits of disp9.
 2. Only the lower half-word is valid.
 3. d is the higher 21 bits of disp22.
 4. d d d d d d d d d d d d d d d d is the higher 15 bits of disp16.

Instruction Set (alphabetical order) (2/4)

| Mnemonic | Operand | Code | Operation | Execution Clock | | | Flag | | | | |
|----------|---------------------|--|--|-----------------|---|---|------|----|---|---|-----|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| LDSR | reg2, regID | r r r r r 1 1 1 1 1 1 R R R R R | SR[regID]<-GR[reg2] | 1 | 1 | 3 | | | | | |
| | | 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 | regID = EIPC, FEPC | | | | | | | | |
| | | Note 1 | regID = EIPSW, FEPSW | | | | 1 | x | x | x | x |
| MOV | reg1, reg2 | r r r r r 0 0 0 0 0 0 R R R R R | GR[reg2]<-GR[reg1] | 1 | 1 | 1 | | | | | |
| | imm5, reg2 | r r r r r 0 1 0 0 0 0 i i i i i | GR[reg2]<-sign-extend(imm5) | 1 | 1 | 1 | | | | | |
| MOVEA | imm16, reg1, reg2 | r r r r r 1 1 0 0 0 1 R R R R R i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1]+sign-extend(imm16) | 1 | 1 | 1 | | | | | |
| MOVHI | imm16, reg1, reg2 | r r r r r 1 1 0 0 1 0 R R R R R i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1]+(imm16 0 ¹⁶) | 1 | 1 | 1 | | | | | |
| MULH | reg1, reg2 | r r r r r 0 0 0 1 1 1 R R R R R | GR[reg2]<-GR[reg2] ^{Note2} xGR[reg1] ^{Note2} (Signed multiplication) | 1 | 1 | 2 | | | | | |
| | imm5, reg2 | r r r r r 0 1 0 1 1 1 i i i i i | GR[reg2]<-GR[reg2] ^{Note2} xsign-extend(imm5) (Signed multiplication) | 1 | 1 | 2 | | | | | |
| MULHI | imm16, reg1, reg2 | r r r r r 1 1 0 1 1 1 R R R R R i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1] ^{Note2} ximm16 (Signed multiplication) | 1 | 1 | 2 | | | | | |
| NOP | | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | Uses 1 clock cycle without doing anything | 1 | 1 | 1 | | | | | |
| NOT | reg1, reg2 | r r r r r 0 0 0 0 0 1 R R R R R | GR[reg2]<-NOT(GR[reg1]) | 1 | 1 | 1 | | 0 | x | x | |
| NOT1 | bit#3, disp16[reg1] | 0 1 b b b 1 1 1 1 1 0 R R R R R d d d d d d d d d d d d d d d d | adr<-GR[reg1]+sign-extend(disp16) Z flag<-Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3, Z flag) | 4 | 4 | 4 | | | | | x |
| OR | reg1, reg2 | r r r r r 0 0 1 0 0 0 R R R R R | GR[reg2]<-GR[reg2]OR GR[reg1] | 1 | 1 | 1 | | 0 | x | x | |
| ORI | imm16, reg1, reg2 | r r r r r 1 1 0 1 0 0 R R R R R i i i i i i i i i i i i i i i i | GR[reg2]<-GR[reg1]OR zero-extend(imm16) | 1 | 1 | 1 | | 0 | x | x | |
| RETI | | 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 | if PSW.EP = 1 then PC <-EIPC PSW <-EIPSW else if PSW.NP = 1 then PC <-FEPC PSW <-FEPSW else PC <-EIPC PSW <-EIPSW | 4 | 4 | 4 | R | R | R | R | R |
| SAR | reg1, reg2 | r r r r r 1 1 1 1 1 1 R R R R R 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 | GR[reg2]<-GR[reg2]arithmetically shift right by GR[reg1] | 1 | 1 | 1 | x | 0 | x | x | |
| | imm5, reg2 | r r r r r 0 1 0 1 0 1 i i i i i | GR[reg2]<-GR[reg2]arithmetically shift right by zero-extend(imm5) | 1 | 1 | 1 | x | 0 | x | x | |

- Notes**
- The op code of this instruction uses the field of reg1 though the source register is shown as reg2 in the above table. Therefore, the meaning of register specification for mnemonic description and op code is different from that of the other instructions.
 rrrrr = regID specification
 RRRRR = reg2 specification
 - Only the lower half-word data is valid.

Instruction Set (alphabetical order) (3/4)

| Mnemonic | Operand | Code | Operation | Execution Clock | | | Flag | | | | |
|-----------|---------------------|--|--|-----------------|---|---|------|----|---|---|-----|
| | | | | i | r | l | CY | OV | S | Z | SAT |
| SATADD | reg1, reg2 | r r r r r 000110RRRRR | GR[reg2]<-saturated(GR[reg2]+GR[reg1]) | 1 | 1 | 1 | x | x | x | x | x |
| | imm5, reg2 | r r r r r 010001 i i i i i | GR[reg2]<-saturated(GR[reg2]+sign-extend(imm5)) | 1 | 1 | 1 | x | x | x | x | x |
| SATSUB | reg1, reg2 | r r r r r 000101RRRRR | GR[reg2]<-saturated(GR[reg2]-GR[reg1]) | 1 | 1 | 1 | x | x | x | x | x |
| ★ SATSUBI | imm16, reg1, reg2 | r r r r r 110011RRRRR i i i i i i i i i i i i i i i i | GR[reg2]<-saturated(GR[reg1]-sign-extend(imm16)) | 1 | 1 | 1 | x | x | x | x | x |
| SATSUBR | reg1, reg2 | r r r r r 000100RRRRR | GR[reg2]<-saturated(GR[reg1]-GR[reg2]) | 1 | 1 | 1 | x | x | x | x | x |
| SETF | cccc, reg2 | r r r r r 1111110cccc 0000000000000000 | if conditions are satisfied then GR[reg2]<-00000001H else GR[reg2]<-00000000H | 1 | 1 | 1 | | | | | |
| SET1 | bit#3, disp16[reg1] | 00b b b 111110RRRRR d d d d d d d d d d d d d d d d | adr<-GR[reg1]+sign-extend(disp16) Z flag<-Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3, 1) | 4 | 4 | 4 | | | | x | |
| SHL | reg1, reg2 | r r r r r 111111RRRRR 0000000011000000 | GR[reg2]<-GR[reg2] logically shift left by GR[reg1] | 1 | 1 | 1 | x | 0 | x | x | |
| | imm5, reg2 | r r r r r 010110 i i i i i | GR[reg2]<-GR[reg1] logically shift left by zero-extend(imm5) | 1 | 1 | 1 | x | 0 | x | x | |
| SHR | reg1, reg2 | r r r r r 111111RRRRR 0000000010000000 | GR[reg2]<-GR[reg2] logically shift right by GR[reg1] | 1 | 1 | 1 | x | 0 | x | x | |
| | imm5, reg2 | r r r r r 010100 i i i i i | GR[reg2]<-GR[reg2] logically shift right by zero-extend(imm5) | 1 | 1 | 1 | x | 0 | x | x | |
| SLD.B | disp7[ep], reg2 | r r r r r 0110d d d d d d d | adr<-ep+zero-extend(disp7) GR[reg2]<-sign-extend(Load-memory(adr, Byte)) | 1 | 1 | 2 | | | | | |
| SLD.H | disp8[ep], reg2 | r r r r r 100d d d d d d d Note 1 | adr<-ep+zero-extend(disp8) GR[reg2]<-sign-extend(Load-memory(adr, Halfword)) | 1 | 1 | 2 | | | | | |
| SLD.W | disp8[ep], reg2 | r r r r r 1010d d d d d d 0 Note 2 | adr<-ep+zero-extend(disp8) GR[reg2]<-Load-memory(adr, Word) | 1 | 1 | 2 | | | | | |
| SST.B | reg2, disp7[ep] | r r r r r 0111d d d d d d d | adr<-ep+zero-extend(disp7) Store-memory(adr, GR[reg2], Byte) | 1 | 1 | 1 | | | | | |
| SST.H | reg2, disp8[ep] | r r r r r 1001d d d d d d d Note 1 | adr<-ep+zero-extend(disp8) Store-memory(adr, GR[reg2], Halfword) | 1 | 1 | 1 | | | | | |
| SST.W | reg2, disp8[ep] | r r r r r 1010d d d d d d 1 Note 2 | adr<-ep+zero-extend(disp8) Store-memory(adr, GR[reg2], Word) | 1 | 1 | 1 | | | | | |
| ST.B | reg2, disp16[reg1] | r r r r r 111010RRRRR d d d d d d d d d d d d d d d d | adr<-GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Byte) | 1 | 1 | 1 | | | | | |

- Notes**
1. d d d d d d d is the higher 7 bits of disp8.
 2. d d d d d d is the higher 6 bits of disp8.

Instruction Set (alphabetical order) (4/4)

| Mnemonic | Operand | Code | Operation | Execution Clock | | | Flag | | | | | |
|----------|---------------------|---|---|-----------------|---|---|------|----|---|---|-----|--|
| | | | | i | r | l | CY | OV | S | Z | SAT | |
| ST.H | reg2, disp16[reg1] | rrrrr111011RRRRR ddddddddddddddd0 Note | adr<-GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Halfword) | 1 | 1 | 1 | | | | | | |
| ST.W | reg2, disp16[reg1] | rrrrr111011RRRRR ddddddddddddddd1 Note | adr<-GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Word) | 1 | 1 | 1 | | | | | | |
| STSR | regID, reg2 | rrrrr111111RRRRR 0000000001000000 | GR[reg2]<-SR[regID] | 1 | 1 | 1 | | | | | | |
| SUB | reg1, reg2 | rrrrr001101RRRRR | GR[reg2]<-GR[reg2]-GR[reg1] | 1 | 1 | 1 | x | x | x | x | | |
| SUBR | reg1, reg2 | rrrrr001100RRRRR | GR[reg2]<-GR[reg1]-GR[reg2] | 1 | 1 | 1 | x | x | x | x | | |
| TRAP | vector | 000001111111iiii 0000000100000000 | EIPC <-PC+4(Restored PC) EIPSW <-PSW ECR.EICC <-Interrupt code PSW.EP <-1 PSW.ID <-1 PC <-00000040H(vector = 00H-0FH) 00000050H(vector = 10H-1FH) | 4 | 4 | 4 | | | | | | |
| TST | reg1, reg2 | rrrrr001011RRRRR | result<-GR[reg2] AND GR[reg1] | 1 | 1 | 1 | | 0 | x | x | | |
| TST1 | bit#3, disp16[reg1] | 11bbb111110RRRRR ddddddddddddddd | adr<-GR[reg1]+sign-extend(disp16) Z flag<-Not(Load-memory-bit(adr, bit#3)) | 3 | 3 | 3 | | | | | x | |
| XOR | reg1, reg2 | rrrrr001001RRRRR | GR[reg2]<-GR[reg2] XOR GR[reg1] | 1 | 1 | 1 | | 0 | x | x | | |
| XORI | imm16, reg1, reg2 | rrrrr110101RRRRR iiiiiiiiiiiiiiii | GR[reg2]<-GR[reg1] XOR zero-extend(imm16) | 1 | 1 | 1 | | 0 | x | x | | |

Note ddddddddddddd is the higher 15 bits of disp16.

Phase-out/Discontinued

[MEMO]

APPENDIX C INDEX

| | | | |
|---|------------|---|---------|
| [A] | | CC10 to CC13 | 119 |
| A0 to A16 | 23 | \overline{CE} | 23 |
| A16 to A23 | 19 | CE1 | 123 |
| AD0 to AD7 | 18 | CE4 | 124 |
| AD8 to AD15 | 18 | CES10, CES11 | 121 |
| Address space | 34, 35, 47 | CESEL | 101 |
| ALV10, ALV11 | 125 | CG | 8 |
| Application fields | 3 | CKSEL | 21 |
| ASIM00, ASIM01 | 151 | CL0 | 152 |
| ASIS0 | 154 | CLKOUT | 21 |
| Assembler reservation register | 29 | Clock | |
| ASTB | 20 | generation function | 95 |
| Asynchronous serial interface | 148 | generator | 8 |
| status register 0 | 154 | output control | 113 |
| mode register 00, 01 | 151 | output inhibit | 99, 113 |
| | | Clocked serial interface 0 to 2 | 162 |
| | | Clocked serial interface mode register 0 to 2 | 164 |
| | | CLSn1, CLSn0 (n = 0 to 2) | 164 |
| [B] | | CM4 | 120 |
| Basic operation (serial interface function) ... | 166 | CMIC4 | 83 |
| Baud rate | | CMIF4 | 83 |
| generator 0, 1 | 173 | CMMK4 | 83 |
| generator register 0, 1 | 176 | CMPR40 to CMPR42 | 83 |
| generator prescaler mode | | CMS10 to CMS 13 | 122 |
| register 0, 1 | 176 | Command register | 103 |
| BCC | 56 | Compare operation (timer 1) | 132 |
| BCn1 (n = 0 to 7) | 56 | Compare operation (timer 4) | 135 |
| BCU | 8 | Compare register 4 | 120 |
| Boundary operation condition | 64 | Connection of unused pins | 24 |
| BPRM0, BPRM1 | 176 | Control register | 101 |
| BPRm0 to BPRm2 (m = 0, 1) | 176 | Count clock selection (timer 1) | 127 |
| BRCE0, BRCE1 | 176 | Count operation (timer 1) | 127 |
| BRG0, BRG1 | 176 | Count operation (timer 4) | 134 |
| BRG set data | 174 | CPU | 8 |
| Bus | | address space | 34 |
| control function | 51 | function | 27 |
| control pin | 51 | register set | 28 |
| control unit | 8 | CRXE0 to CRXE2 | 164 |
| cycle control register | 56 | CSI system configuration | 172 |
| hold | 57, 64 | CSIC0 to CSIC2 | 83 |
| priority | 65 | CSIF0 to CSIF2 | 83 |
| timing | 58 | CSIM0 to CSIM2 | 164 |
| Byte write mode | 214 | CSMK0 to CSMK2 | 83 |
| | | CSI0 to CSI2 | 162 |
| [C] | | CSOT0 to CSOT2 | 164 |
| Capture operation | 130 | | |
| Capture/compare register 10 to 13 | 119 | | |

| | | | |
|-------------------------------------|------------|---|---------|
| CSPRn0 to CSPRn2 (n = 0 to 2) | 83 | FEPSW | 30 |
| CTXE0 to CTXE2 | 164 | Function block configuration | 7 |
| CV _{DD} | 22 | | |
| CV _{SS} | 22 | [G] | |
| CY | 31 | General register | 29 |
| Cycle measurement | 143 | Global pointer | 29 |
| | | | |
| [D] | | [H] | |
| D0 to D7 | 23 | HALT mode | 99, 104 |
| Data space | 36, 47, 65 | HLD _{AK} | 21 |
| Data wait control register | 54 | HLD _{RQ} | 21 |
| DCLK0, DCLK1 | 101 | | |
| Direct mode | 96 | [I] | |
| D _{STB} | 20 | IC0 | 22 |
| DWC | 54 | ID | 31, 85 |
| Dwn0 to Dwn1 (n = 0 to 7) | 54 | IDLE | 101 |
| | | Idle state insertion function | 56 |
| | | ILGOP | 68 |
| [E] | | Illegal op code | 88 |
| EBS0 | 153 | IMS10 IMS13 | 122 |
| ECLR1 | 121 | Initial register values | 213 |
| ECR | 30 | Initialize | 210 |
| EICC | 30 | Input clock selection (clock generator) | 96 |
| EIPC | 30 | Input clock selection (timer 4) | 134 |
| EIPSW | 30 | INTC | 8 |
| Element pointer | 29 | INTCM4 | 68 |
| ENTO10, ENTO11 | 125 | INTCSI0 to INTCSI2 | 68 |
| EP | 31, 88 | Internal | |
| ES0n0, ES0n1 (n = 0 to 3) | 84 | block diagram | 7 |
| ES1n0, ES1n1 (n = 0 to 3) | 84 | count clock | 127 |
| ESN0 | 73 | peripheral I/O interface | 66 |
| ETI | 123 | RAM area | 40 |
| Exception | | ROM/PROM area | 38 |
| processing function | 67 | units | 8 |
| table | 38 | Interrupt | |
| trap | 88 | controller | 8 |
| External count clock | 128 | control register | 82 |
| External expansion mode | 45 | latency time | 93 |
| External interrupt | | list | 68 |
| mode register 0 | 73 | processing (service) | 67 |
| mode register 1 | 84 | request | 157 |
| mode register 2 | 84, 125 | stack pointer | 29 |
| External memory area | 42 | source register | 30 |
| External wait function | 55 | table | 38 |
| | | Interval timer | 137 |
| [F] | | INTM0 | 73 |
| FE0 | 154 | INTM1 | 84 |
| FECC | 30 | INTM2 | 84, 125 |
| FEPC | 30 | | |

| | | | |
|---|-----------|--|---------|
| INTOV1 | 68 | OVIC1 | 83 |
| INTP00 to INTP03 | 17, 68 | OVIF1 | 83 |
| INTP10 to INTP13 | 16 | OVMK1 | 83 |
| INTPn/INTCCn (n = 10 to 13) | 68 | OVPR10 to OVPR12 | 83 |
| INTSER0 | 68 | | |
| INTSR0 | 68 | [P] | |
| INTST0 | 68 | P0IC0 to P0IC3 | 83 |
| ISPR (in-service priority register) | 85 | P0IF0 to P0IF3 | 83 |
| ISPR0 to ISPR7 | 85 | P0MK0 to P0MK3 | 83 |
| | | P0PR00 to P0PR02 | 83 |
| [L] | | P0PR10 to P0PR12 | 83 |
| LBEN | 19 | P0PR20 to P0PR22 | 83 |
| Link pointer | 29 | P0PR30 to P0PR32 | 83 |
| | | P1IC0 to P1IC3 | 83 |
| [M] | | P1IF0 to P1IF3 | 83 |
| Maskable interrupt | 74, 78 | P1MK0 to P1MK3 | 83 |
| status flag | 85 | P1PRn0 to P1PRn2 (n = 0 to 3) | 83 |
| Memory | | Page data latch mode | 214 |
| block function | 53 | Page write mode | 214 |
| expansion mode register | 46 | PC | 29 |
| map | 37, 48 | PE0 | 154 |
| read | 58 | Period where interrupt is not acknowledged | 93 |
| write | 62 | Peripheral I/O area | 41 |
| MM | 46 | Peripheral I/O register | 49 |
| MM0 to MM3 | 46 | PGM | 23 |
| MODE0, MODE1 | 22 | Pin configuration | 4 |
| MOD0 to MOD2 | 164 | Pin function | 11, 16 |
| Multiple interrupt | 91 | Pin I/O circuit | 25 |
| | | Pin status | 15 |
| [N] | | PLL mode | 96 |
| NMI | 17, 68 | PLL stabilization | 98 |
| Noise elimination circuit | 208 | PLLSEL | 21 |
| Normal operation mode | 4, 11, 16 | Port | 9, 179 |
| Note (timer/counter function) | 145 | Port 0 (P0) | 182 |
| NP | 31, 73 | block diagram of P00 and P01 | 183 |
| Number of access clock | 52 | block diagram of P02 to P07 | 183 |
| | | P00 to P07 | 16, 182 |
| [O] | | Port 0 mode control register (PMC0) | 185 |
| \overline{OE} | 23 | PMC00 to PMC07 | 185 |
| Operation mode | 32 | Port 0 mode register (PM0) | 184 |
| Ordering information | 3 | PM00 to PM07 | 184 |
| OST | 121 | Port 1 (P1) | 186 |
| Output disable mode | 214 | block diagram of P10 to P17 | 186 |
| OV | 31 | P10 to P17 | 16, 186 |
| OVE0 | 154 | Port 1 mode register (PM1) | 187 |
| Overflow (timer 1) | 128 | PM10 to PM17 | 187 |
| Overflow (timer 4) | 134 | Port 10 (P10) | 205 |
| OVF1, OVF4 | 126 | | |

| | | | |
|---|---------|---|----------------|
| block diagram of P100 and P103 | 207 | Port 9 mode register (PM9) | 204 |
| block diagram of P101 | 208 | PM90 to PM97 | 204 |
| block diagram of P102 | 208 | Power save control | 99 |
| P100 to P103 | 21, 207 | Power save control register | 101 |
| Port 10 mode control register (PMC10) | 209 | Power save mode operation | 56 |
| PMC100 and PMC101 | 209 | PRCMD | 103 |
| Port 10 mode register (PM10) | 209 | PRERR | 103 |
| PM100 to PM103 | 209 | Priority control | 91 |
| Port 2 (P2) | 187 | Priority of interrupt and exception | 91 |
| block diagram of P20 | 188 | PRM11 | 123 |
| block diagram of P21 to P24 | 188 | PRM40, PRM41 | 124 |
| block diagram of P25 | 189 | Program counter | 29 |
| block diagram of P26 | 189 | Program inhibit mode | 214 |
| block diagram of P27 | 190 | Program register set | 29 |
| P20 to P27 | 16, 187 | Program space | 36, 47, 65 |
| Port 2 mode control register (PMC2) | 192 | Program status word (PSW) | 31, 73, 85, 88 |
| PMC21 to PMC27 | 192 | Program verify mode | 214 |
| Port 2 mode register (PM2) | 191 | Programmable wait function | 54 |
| PM21 to PM27 | 191 | PROM | 8 |
| Port 3 (P3) | 192 | PROM mode | 213 |
| block diagram of P30, P33, P35 ... | 193 | PROM programming mode | 6, 14, 23 |
| block diagram of P31 and P36 | 194 | PROM read mode | 32 |
| block diagram of P32 and P37 | 194 | PROM read procedure | 219 |
| block diagram of P34 | 195 | PROM write procedure | 215 |
| P30 to P37 | 17, 192 | PRS10, PRS11 | 123 |
| Port 3 mode control register (PMC3) | 196 | PRS40 | 124 |
| PMC30 to PMC37 | 196 | PS00, PS01 | 152 |
| Port 3 mode register (PM3) | 195 | PSC | 101 |
| PM30 to PM37 | 195 | Pulse width measurement | 138 |
| Port 4 (P4) | 197 | PWM output | 140 |
| block diagram of P40 to P47 ... | 18, 197 | | |
| P40 to P47 | 197 | [R] | |
| Port 4 mode register (PM4) | 198 | r0 to r31 | 29 |
| PM40 to PM47 | 198 | R/W | 20 |
| Port 5 (P5) | 199 | RAM | 8 |
| block diagram of P50 to P57 | 199 | Read mode | 213 |
| P50 to P57 | 18, 199 | Real-time pulse unit | 8, 115 |
| Port 5 mode register (PM5) | 200 | Receive buffer 0, 0L | 155 |
| PM50 to PM57 | 200 | Reception completion interrupt | 157 |
| Port 6 (P6) | 201 | Receive error interrupt | 157 |
| block diagram of P60 to P67 | 201 | REG0 to REG7 | 103 |
| P60 to P67 | 19, 201 | RESET | 22 |
| Port 6 mode register (PM6) | 202 | RESET | 68 |
| PM60 to PM67 | 202 | Reset function | 211 |
| Port 9 (P9) | 202 | RETI instruction operation | 72, 77, 87, 90 |
| block diagram of P90 to P97 | 203 | RFU | 31 |
| P90 to P97 | 19, 202 | ROM | 8 |
| | | ROM-less mode | 32 |

| | |
|---|---------------|
| RPU | 8, 116 |
| RXB00 to RXB07 | 155 |
| RXB0, RXB0L | 155 |
| RXD | 18 |
| RXE0 | 151 |
| RXEB0 | 155 |
| [S] | |
| S | 31 |
| SAT | 31 |
| $\overline{\text{SCK0}}$ | 17 |
| $\overline{\text{SCK1}}$ | 17 |
| $\overline{\text{SCK2}}$ | 17 |
| SCLS0 | 153 |
| Screening of OTPROM version | 222 |
| SEIC0 | 83 |
| SEIF0 | 83 |
| SEMK0 | 83 |
| SEPR00 to SEPR02 | 83 |
| Serial I/O shift register 0 to 2 | 165 |
| Serial interface | 10, 147 |
| SI0 to SI2 | 17 |
| Single-chip mode | 32 |
| SIO | 8 |
| SIO0 to SIO2 | 165 |
| SIO _n 0 to SIO _n 7 (n = 0 to 2) | 165 |
| SL0 | 152 |
| SO0 to SO2 | 17 |
| Software exception | 86 |
| Software STOP mode | 99, 108 |
| SOT0 | 154 |
| Specifying oscillation stabilization time | 110 |
| SRIC0 | 83 |
| SRIF0 | 83 |
| SRMK0 | 83 |
| SRPR00 to SRPR02 | 83 |
| ST0, ST1 | 20 |
| Stack pointer | 29 |
| Standby mode | 214 |
| Status transition | 100 |
| STIC0 | 83 |
| STIF0 | 83 |
| STMK0 | 83 |
| STOR00 to STPR02 | 83 |
| STP | 101 |
| SYS | 98, 102 |
| System register set | 30 |
| [T] | |
| TBC | 112 |
| TBCS | 101 |
| TCLR1 | 16 |
| TES10 and TES11 | 121 |
| Text pointer | 29 |
| TI1 | 16 |
| Time base counter | 112 |
| Timer | |
| /counter function | 115 |
| control register 1 | 123 |
| control register 4 | 124 |
| output control register 1 | 125 |
| overflow status register | 126 |
| unit mode register 1 | 121 |
| Timer 1 | 118 |
| Timer 1 operation | 127 |
| Timer 4 | 120 |
| Timer 4 operation | 134 |
| Timing of 3-wire serial I/O mode | 168, 169, 171 |
| TM1 | 118 |
| capture operation example | 130 |
| compare operation example | 132 |
| TM4 | 120 |
| TMC1 | 123 |
| TMC4 | 124 |
| TO10, TO11 | 16 |
| TOC1 | 125 |
| TOVS | 126 |
| Transmission completion interrupt | 157 |
| Transmit shift register 0, 0L | 156 |
| TRAP _{0n} , TRAP _{1n} (n = 0 to F) | 68 |
| TUM1 | 121 |
| TXD | 17 |
| TXED0 | 156 |
| TXS0, TXS0L | 156 |
| TXS00 to TXS07 | 156 |
| [U] | |
| UART | 148 |
| $\overline{\text{UBEN}}$ | 19 |
| UNLOCK | 98 |
| [V] | |
| V _{DD} | 23 |
| V _{PP} | 23 |
| V _{SS} | 23 |

[W]

| | |
|------------------------------------|--------|
| WAIT | 22 |
| Wrap-around | 36, 47 |
| Wait function | 54 |
| Wait state inserting example | 55 |

[X]

| | |
|--------------|----|
| X1, X2 | 22 |
|--------------|----|

[Z]

| | |
|---------------------|----|
| Zero register | 29 |
|---------------------|----|

[Others]

| | |
|---------------------------|---|
| 100-pin plastic QFP | 3 |
|---------------------------|---|

Facsimile Message

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

From:

Name

Company

Tel.

FAX

Address

Thank you for your kind support.

North America

NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Europe

NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

Korea

NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Japan

NEC Corporation
Semiconductor Solution Engineering Division
Technical Information Support Dept.
Fax: 044-548-7900

South America

NEC do Brasil S.A.
Fax: +55-11-6465-6829

Taiwan

NEC Electronics Taiwan Ltd.
Fax: 02-719-5951

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

| Document Rating | Excellent | Good | Acceptable | Poor |
|--------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Clarity | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Technical Accuracy | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Organization | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |