Notice

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

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

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

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

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

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

To all our customers

Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

RENESAS
Renesas Technology Corp.

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.

User's Manual

8

H8/3502

Hardware Manual

HD6433502

Renesas Electronics
www.renesas.com

Rev.1.0 1997.09

Notice

When using this document, keep the following in mind:

1. This document may, wholly or partially, be subject to change without notice.
2. All rights are reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without Hitachi's permission.
3. Hitachi will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's unit according to this document.
4. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.
5. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.
6. MEDICAL APPLICATIONS: Hitachi's products are not authorized for use in MEDICAL APPLICATIONS without the written consent of the appropriate officer of Hitachi's sales company. Such use includes, but is not limited to, use in life support systems. Buyers of Hitachi's products are requested to notify the relevant Hitachi sales offices when planning to use the products in MEDICAL APPLICATIONS.

Preface

The H8/3502 is a high-performance single-chip microcomputer ideally suited for embedded control applications. The chip is built around a high-speed H8/300 CPU core. On-chip supporting modules include 16-kbyte ROM, 512-byte RAM, three types of timers, a serial communication interface, host interface, and I/O ports, for easy implementation of compact, high-performance control systems.

Development tools that support the functionally higher-end H8/3217 Series should be used for H8/3502 program development.

The H8/3502 is also available as a ZTAT™ (Zero Turn-Around Time) version of the functionally higher-end H8/3214. This version enables the user to respond quickly and flexibly to changing application system specifications and the demands of the transition from initial to full-fledged volume production.

There are a number of differences between the H8/3502 and the functionally higher-end H8/3217 Series. In terms of functions, the H8/3502 is available in only one ROM/RAM configuration, has a maximum operating frequency of 10 MHz, and does not offer a guaranteed current dissipation figure in standby mode, one of its power-down states.

The H8/3502 single-chip microcomputer is intended for consumer applications. If the user requires a ZTAT™ version, larger ROM/RAM capacity, processing at a maximum 16 MHz, significant power reduction in standby mode for portable systems, etc., or the high reliability essential for automotive or industrial applications, the H8/3217 Series should be used.

This manual describes the H8/3502 hardware. Refer to the *H8/300 Series Programming Manual* for a detailed description of the instruction set, and to the *H8/3217 Series Hardware Manual* for details of higher-end products, including ZTAT™ versions.

Note: ZTAT is a trademark of Hitachi, Ltd.

Contents

Section 1 Overview

1.1 Overview

The H8/3502 is a series of single-chip microcomputers integrating a CPU core together with a variety of peripheral functions needed in control systems.

The H8/300 CPU is a high-speed processor featuring powerful bit-manipulation instructions, ideally suited for realtime control applications. On-chip supporting modules necessary for system configuration include 16-kbyte ROM, 512-byte RAM, three types of timers (16-bit free-running timer, 8-bit timer, and watchdog timer), a serial communication interface (SCI), host interface (HIF), and I/O ports.

The H8/3502 can operate in single-chip mode or in two expanded modes, depending on the memory requirements of the application.

Development tools that support the functionally higher-end H8/3217 Series should be used for H8/3502 program development. As a ZTAT™ version, use the ZTAT™ version of the H8/3214. Registers related to higher-level functions should not be accessed in this case. In particular, 1 must not be written in the IICS, IICX1, IICX0, SYNCE, PWCKE, and PWCKS bits in the serial/timer control register (STCR).

Note: ZTAT is a trademark of Hitachi, Ltd.

Table 1-1 lists the features of the H8/3502.

Table 1-1 Features

Feature	Description
CPU	General register architecture • Eight 16-bit general registers, or • Sixteen 8-bit general registers High speed • Maximum clock rate: 10 MHz/5 V (ø clock) • Add/subtract: 200 ns (10 MHz operation) • Multiply/divide: 1400 ns (10 MHz operation) Concise, streamlined instruction set • All instructions are 2 or 4 bytes long • Register-register arithmetic and logic operations • Register-memory data transfer by MOV instruction Instruction set features • Multiply instruction (8 bits × 8 bits) • Divide instruction (16 bits ÷ 8 bits) • Bit-accumulator instructions • Register-indirect specification of bit positions
Memory	• ROM: 16 kbytes • RAM: 512 bytes
16-Bit free-running timer (FRT: 1 channel)	• One 16-bit free-running counter (also usable for external event counting) • Two compare outputs • One capture input
8-bit timer (TMR: 2 channels)	Each channel has: • One 8-bit up-counter (also usable for external event counting) • Two time constant registers
Watchdog timer (WDT: 1 channel)	• Reset or NMI generation by overflow • Can be switched to interval timer mode

Table 1-1 Features (cont)

<table>
<tr><th>Feature</th><th colspan="4">Description</th></tr>
<tr><td>Serial communication interface (SCI: 2 channels)</td><td colspan="4">• Selection of asynchronous and synchronous modes
• Simultaneous transmit and receive (full duplex operation)
• On-chip baud rate generator</td></tr>
<tr><td>Host interface (HIF)</td><td colspan="4">• 8-bit host interface port
• Three host interrupt requests ($HIRQ_1$, $HIRQ_{11}$, $HIRQ_{12}$)
• Normal and fast A_{20} gate output
• Two register sets (each comprising two data registers and a status register)</td></tr>
<tr><td>Keyboard controller</td><td colspan="4">• Controls a matrix keyboard using a keyboard scan with wake-up interrupt and sense port configuration</td></tr>
<tr><td>I/O ports</td><td colspan="4">• 53 input/output pins (of which 24 can drive large current loads)</td></tr>
<tr><td>Interrupts</td><td colspan="4">• Four external interrupt pins: $\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_2}$
• Eight key-sense interrupt pins: $\overline{KEYIN_0}$ to $\overline{KEYIN_7}$
• Twenty-one on-chip interrupt sources</td></tr>
<tr><td>Operating modes</td><td colspan="4">• Mode 1: expanded mode with on-chip ROM disabled
• Mode 2: expanded mode with on-chip ROM enabled
• Mode 3: single-chip mode</td></tr>
<tr><td>Power-down state</td><td colspan="4">• Sleep mode
• Software standby mode
• Hardware standby mode</td></tr>
<tr><td>Other features</td><td colspan="4">• On-chip clock oscillator</td></tr>
<tr><td rowspan="3">Product lineup</td><td><b>Product Name</b></td><td><b>Type Code</b></td><td><b>Package</b></td><td><b>ROM</b></td></tr>
<tr><td rowspan="2">H8/3502</td><td>HD6433502P10</td><td>64-pin shrink DIP (DP-64S)</td><td rowspan="2">Mask ROM</td></tr>
<tr><td>HD6433502F10</td><td>64-pin QFP (FP-64A)</td></tr>
</table>

1.2 Block Diagram

Figure 1-1 shows a block diagram of the H8/3502.

See table 1-2, Pin Assignments in Each Operating Mode, for differences in the pin functions.

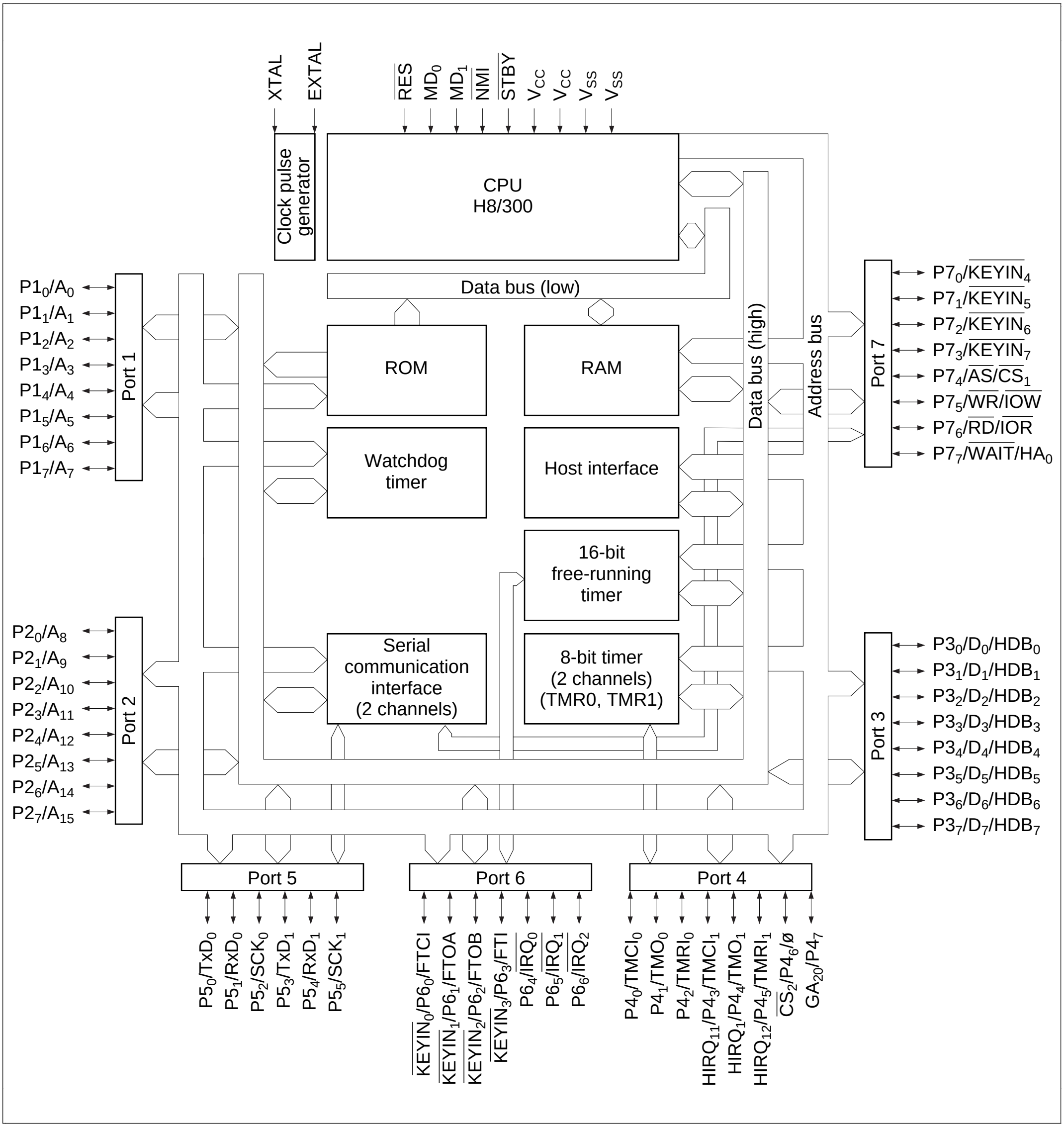


Figure 1-1 Block Diagram

1.3 Pin Assignments and Functions

1.3.1 Pin Arrangement

Figure 1-2 shows the pin arrangement of the H8/3502 in the DP-64S packages. Figure 1-3 shows the pin arrangement in the FP-64A package.

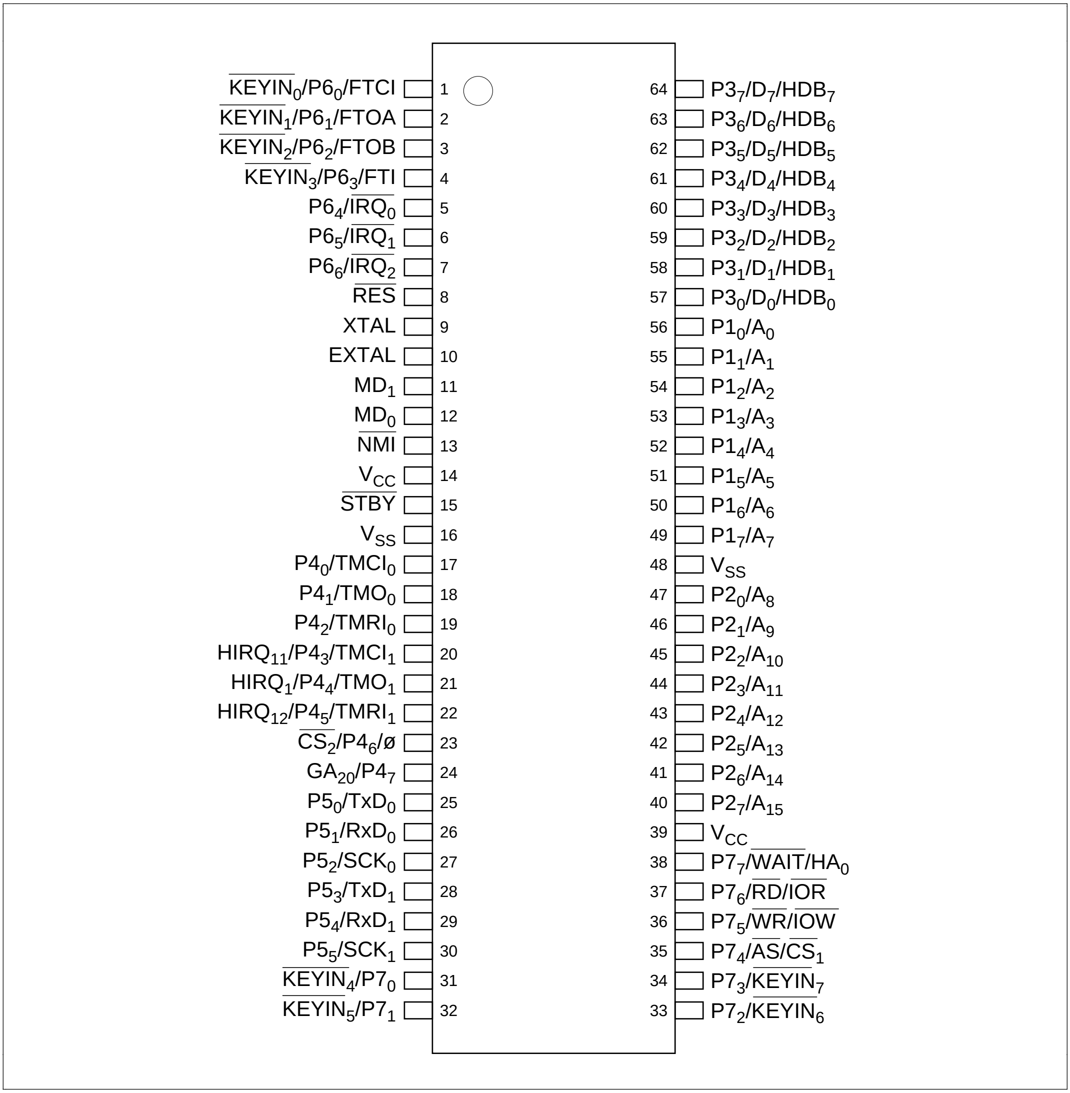

$\overline{\mathrm{KEYIN}}_0$/P6$_0$/FTCI 1
$\overline{\mathrm{KEYIN}}_1$/P6$_1$/FTOA 2
$\overline{\mathrm{KEYIN}}_2$/P6$_2$/FTOB 3
$\overline{\mathrm{KEYIN}}_3$/P6$_3$/FTI 4
P6$_4$/$\overline{\mathrm{IRQ}_0}$ 5
P6$_5$/$\overline{\mathrm{IRQ}_1}$ 6
P6$_6$/$\overline{\mathrm{IRQ}_2}$ 7
$\overline{\mathrm{RES}}$ 8
XTAL 9
EXTAL 10
MD$_1$ 11
MD$_0$ 12
$\overline{\mathrm{NMI}}$ 13
V$_{CC}$ 14
$\overline{\mathrm{STBY}}$ 15
V$_{SS}$ 16
P4$_0$/TMCI$_0$ 17
P4$_1$/TMO$_0$ 18
P4$_2$/TMRI$_0$ 19
HIRQ$_{11}$/P4$_3$/TMCI$_1$ 20
HIRQ$_1$/P4$_4$/TMO$_1$ 21
HIRQ$_{12}$/P4$_5$/TMRI$_1$ 22
$\overline{\mathrm{CS}}_2$/P4$_6$/ø 23
GA$_{20}$/P4$_7$ 24
P5$_0$/TxD$_0$ 25
P5$_1$/RxD$_0$ 26
P5$_2$/SCK$_0$ 27
P5$_3$/TxD$_1$ 28
P5$_4$/RxD$_1$ 29
P5$_5$/SCK$_1$ 30
$\overline{\mathrm{KEYIN}}_4$/P7$_0$ 31
$\overline{\mathrm{KEYIN}}_5$/P7$_1$ 32
64 P3$_7$/D$_7$/HDB$_7$
63 P3$_6$/D$_6$/HDB$_6$
62 P3$_5$/D$_5$/HDB$_5$
61 P3$_4$/D$_4$/HDB$_4$
60 P3$_3$/D$_3$/HDB$_3$
59 P3$_2$/D$_2$/HDB$_2$
58 P3$_1$/D$_1$/HDB$_1$
57 P3$_0$/D$_0$/HDB$_0$
56 P1$_0$/A$_0$
55 P1$_1$/A$_1$
54 P1$_2$/A$_2$
53 P1$_3$/A$_3$
52 P1$_4$/A$_4$
51 P1$_5$/A$_5$
50 P1$_6$/A$_6$
49 P1$_7$/A$_7$
48 V$_{SS}$
47 P2$_0$/A$_8$
46 P2$_1$/A$_9$
45 P2$_2$/A$_{10}$
44 P2$_3$/A$_{11}$
43 P2$_4$/A$_{12}$
42 P2$_5$/A$_{13}$
41 P2$_6$/A$_{14}$
40 P2$_7$/A$_{15}$
39 V$_{CC}$
38 P7$_7$/$\overline{\mathrm{WAIT}}$/HA$_0$
37 P7$_6$/$\overline{\mathrm{RD}}$/$\overline{\mathrm{IOR}}$
36 P7$_5$/$\overline{\mathrm{WR}}$/$\overline{\mathrm{IOW}}$
35 P7$_4$/$\overline{\mathrm{AS}}$/$\overline{\mathrm{CS}}_1$
34 P7$_3$/$\overline{\mathrm{KEYIN}}_7$
33 P7$_2$/$\overline{\mathrm{KEYIN}}_6$

Figure 1-2 Pin Arrangement (DP-64S, Top View)

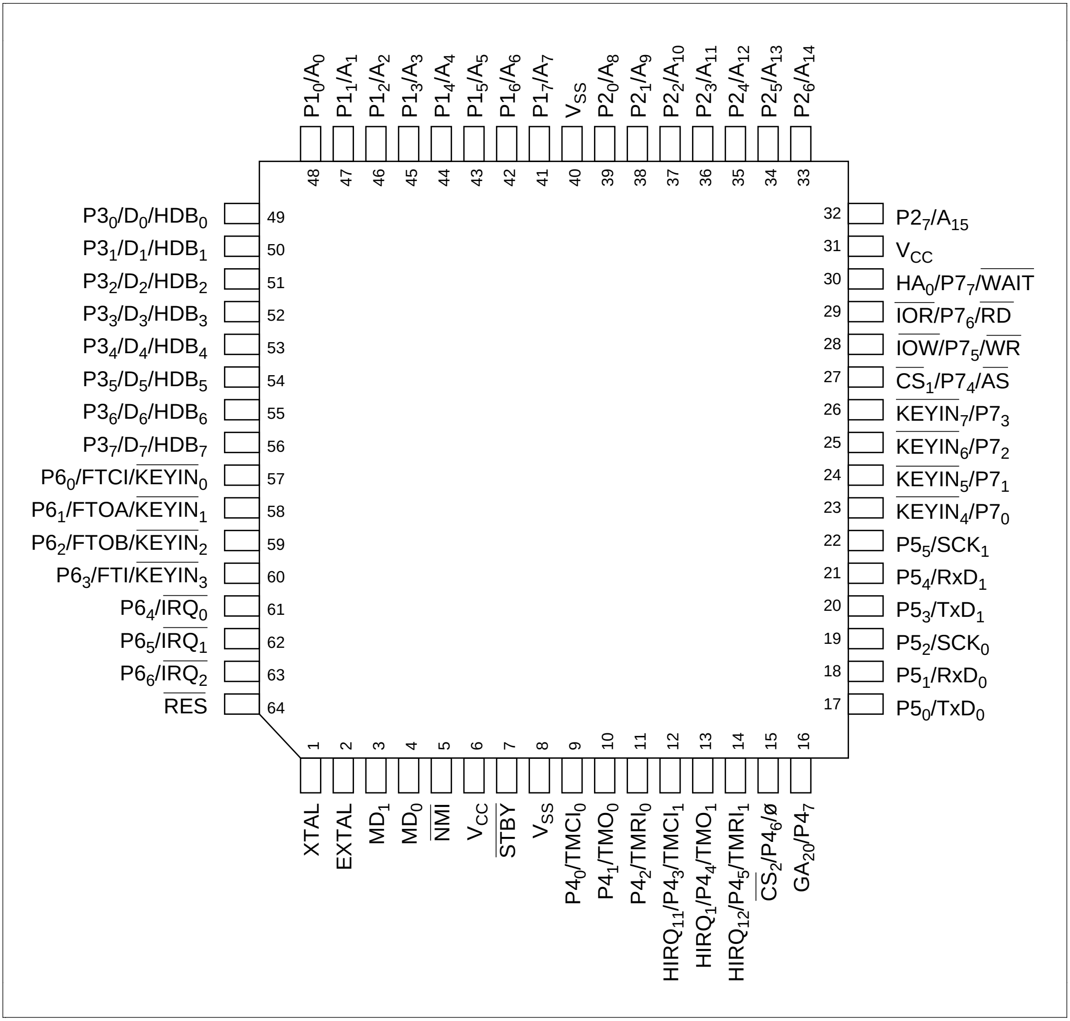


Figure 1-3 Pin Arrangement (FP-64A, Top View)

1.3.2 Pin Functions

(1) Pin Assignments in Each Operating Mode: Table 1-2 list the assignments of the pins of the DP-64S and FP-64A packages in each operating mode.

Table 1-2 Pin Assignments in Each Operating Mode

Pin No.		Expanded Modes		Single-Chip Mode
DP-64S	**FP-64A**	**Mode 1**	**Mode 2**	**Mode 3**
1	57	P6$_0$/FTCI/$\overline{\text{KEYIN}}_0$	P6$_0$/FTCI/$\overline{\text{KEYIN}}_0$	P6$_0$/FTCI/$\overline{\text{KEYIN}}_0$
2	58	P6$_1$/FTOA/$\overline{\text{KEYIN}}_1$	P6$_1$/FTOA/$\overline{\text{KEYIN}}_1$	P6$_1$/FTOA/$\overline{\text{KEYIN}}_1$
3	59	P6$_2$/FTOB/$\overline{\text{KEYIN}}_2$	P6$_2$/FTOB/$\overline{\text{KEYIN}}_2$	P6$_2$/FTOB/$\overline{\text{KEYIN}}_2$
4	60	P6$_3$/FTI/$\overline{\text{KEYIN}}_3$	P6$_3$/FTI/$\overline{\text{KEYIN}}_3$	P6$_3$/FTI/$\overline{\text{KEYIN}}_3$
5	61	P6$_4$/$\overline{\text{IRQ}}_0$	P6$_4$/$\overline{\text{IRQ}}_0$	P6$_4$/$\overline{\text{IRQ}}_0$
6	62	P6$_5$/$\overline{\text{IRQ}}_1$	P6$_5$/$\overline{\text{IRQ}}_1$	P6$_5$/$\overline{\text{IRQ}}_1$
7	63	P6$_6$/$\overline{\text{IRQ}}_2$	P6$_6$/$\overline{\text{IRQ}}_2$	P6$_6$/$\overline{\text{IRQ}}_2$
8	64	$\overline{\text{RES}}$	$\overline{\text{RES}}$	$\overline{\text{RES}}$
9	1	XTAL	XTAL	XTAL
10	2	EXTAL	EXTAL	EXTAL
11	3	MD$_1$	MD$_1$	MD$_1$
12	4	MD$_0$	MD$_0$	MD$_0$
13	5	$\overline{\text{NMI}}$	$\overline{\text{NMI}}$	$\overline{\text{NMI}}$
14	6	V_{CC}	V_{CC}	V_{CC}
15	7	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$
16	8	V_{SS}	V_{SS}	V_{SS}
17	9	P4$_0$/TMCI$_0$	P4$_0$/TMCI$_0$	P4$_0$/TMCI$_0$
18	10	P4$_1$/TMO$_0$	P4$_1$/TMO$_0$	P4$_1$/TMO$_0$
19	11	P4$_2$/TMRI$_0$	P4$_2$/TMRI$_0$	P4$_2$/TMRI$_0$
20	12	P4$_3$/TMCI$_1$	P4$_3$/TMCI$_1$	P4$_3$/TMCI$_1$/HIRQ$_{11}$
21	13	P4$_4$/TMO$_1$	P4$_4$/TMO$_1$	P4$_4$/TMO$_1$/HIRQ$_1$
22	14	P4$_5$/TMRI$_1$	P4$_5$/TMRI$_1$	P4$_5$/TMRI$_1$/HIRQ$_{12}$
23	15	ø	ø	P4$_6$/ø/$\overline{\text{CS}}_2$
24	16	P4$_7$	P4$_7$	P4$_7$/GA$_{20}$
25	17	P5$_0$/TxD$_0$	P5$_0$/TxD$_0$	P5$_0$/TxD$_0$
26	18	P5$_1$/RxD$_0$	P5$_1$/RxD$_0$	P5$_1$/RxD$_0$
27	19	P5$_2$/SCK$_0$	P5$_2$/SCK$_0$	P5$_2$/SCK$_0$

Table 1-2 Pin Assignments in Each Operating Mode (cont)

Pin No.		Expanded Modes		Single-Chip Mode
DP-64S	FP-64A	Mode 1	Mode 2	Mode 3
28	20	$P5_3$/TxD_1	$P5_3$/TxD_1	$P5_3$/TxD_1
29	21	$P5_4$/RxD_1	$P5_4$/RxD_1	$P5_4$/RxD_1
30	22	$P5_5$/SCK_1	$P5_5$/SCK_1	$P5_5$/SCK_1
31	23	$P7_0$/$\overline{KEYIN}_4$	$P7_0$/$\overline{KEYIN}_4$	$P7_0$/$\overline{KEYIN}_4$
32	24	$P7_1$/$\overline{KEYIN}_5$	$P7_1$/$\overline{KEYIN}_5$	$P7_1$/$\overline{KEYIN}_5$
33	25	$P7_2$/$\overline{KEYIN}_6$	$P7_2$/$\overline{KEYIN}_6$	$P7_2$/$\overline{KEYIN}_6$
34	26	$P7_3$/$\overline{KEYIN}_7$	$P7_3$/$\overline{KEYIN}_7$	$P7_3$/$\overline{KEYIN}_7$
35	27	$\overline{AS}$	$\overline{AS}$	$P7_4$/$\overline{CS}_1$
36	28	$\overline{WR}$	$\overline{WR}$	$P7_5$/$\overline{IOW}$
37	29	$\overline{RD}$	$\overline{RD}$	$P7_6$/$\overline{IOR}$
38	30	$P7_7$/$\overline{WAIT}$	$P7_7$/$\overline{WAIT}$	$P7_7$/HA_0
39	31	V_{CC}	V_{CC}	V_{CC}
40	32	A_{15}	$P2_7$/A_{15}	$P2_7$
41	33	A_{14}	$P2_6$/A_{14}	$P2_6$
42	34	A_{13}	$P2_5$/A_{13}	$P2_5$
43	35	A_{12}	$P2_4$/A_{12}	$P2_4$
44	36	A_{11}	$P2_3$/A_{11}	$P2_3$
45	37	A_{10}	$P2_2$/A_{10}	$P2_2$
46	38	A_9	$P2_1$/A_9	$P2_1$
47	39	A_8	$P2_0$/A_8	$P2_0$
48	40	V_{SS}	V_{SS}	V_{SS}
49	41	A_7	$P1_7$/A_7	$P1_7$
50	42	A_6	$P1_6$/A_6	$P1_6$
51	43	A_5	$P1_5$/A_5	$P1_5$
52	44	A_4	$P1_4$/A_4	$P1_4$
53	45	A_3	$P1_3$/A_3	$P1_3$
54	46	A_2	$P1_2$/A_2	$P1_2$
55	47	A_1	$P1_1$/A_1	$P1_1$
56	48	A_0	$P1_0$/A_0	$P1_0$

Table 1-2 Pin Assignments in Each Operating Mode (cont)

Pin No.		Expanded Modes		Single-Chip Mode
DP-64S	**FP-64A**	**Mode 1**	**Mode 2**	**Mode 3**
57	49	D_0	D_0	$P3_0$/HDB_0
58	50	D_1	D_1	$P3_1$/HDB_1
59	51	D_2	D_2	$P3_2$/HDB_2
60	52	D_3	D_3	$P3_3$/HDB_3
61	53	D_4	D_4	$P3_4$/HDB_4
62	54	D_5	D_5	$P3_5$/HDB_5
63	55	D_6	D_6	$P3_6$/HDB_6
64	56	D_7	D_7	$P3_7$/HDB_7

(2) Pin Functions: Table 1-3 gives a concise description of the function of each pin.

Table 1-3 Pin Functions

		Pin No.			
Type	**Symbol**	**DP-64S**	**FP-64A**	**I/O**	**Name and Function**
Power	V_{CC}	14, 39	6, 31	I	**Power:** Connected to the power supply. Connect both V_{CC} pins to the system power supply.
	V_{SS}	16, 48	8, 40	I	**Ground:** Connected to ground (0 V). Connect all V_{SS} pins to the system power supply (0 V).
Clock	XTAL	9	1	I	**Crystal:** Connected to a crystal oscillator. The crystal frequency must be the same as the desired system clock frequency. If an external clock is input at the EXTAL pin, a reverse-phase clock should be input at the XTAL pin.
	EXTAL	10	2	I	**External crystal:** Connected to a crystal oscillator or external clock. The frequency of the external clock must be the same as the desired system clock frequency. See section 6, Clock Pulse Generator, for examples of connections to a crystal and external clock.
	ø	23	15	O	**System clock:** Supplies the system clock to peripheral devices.
System control	$\overline{\text{RES}}$	8	64	I	**Reset:** A low input causes the chip to reset.
	$\overline{\text{STBY}}$	15	7	I	**Standby:** A transition to the hardware standby mode (a power-down state) occurs when a low input is received at the $\overline{\text{STBY}}$ pin.
Address bus	A_{15} to A_0	40 to 47, 49 to 56,	32 to 39, 41 to 48	O	**Address bus:** Address output pins.

Table 1-3 Pin Functions (cont)

<table>
<tr><th></th><th></th><th colspan="2">Pin No.</th><th></th><th></th></tr>
<tr><th>Type</th><th>Symbol</th><th>DP-64S</th><th>FP-64A</th><th>I/O</th><th>Name and Function</th></tr>
<tr><td>Data bus</td><td>D_7 to D_0</td><td>64 to 57</td><td>56 to 49</td><td>I/O</td><td><b>Data bus:</b> 8-bit bidirectional data bus.</td></tr>
<tr><td>Bus control</td><td>$\overline{WAIT}$</td><td>38</td><td>30</td><td>I</td><td><b>Wait:</b> Requests the CPU to insert T_W states into the bus cycle when an off-chip address is accessed.</td></tr>
<tr><td></td><td>$\overline{RD}$</td><td>37</td><td>29</td><td>O</td><td><b>Read:</b> Goes low to indicate that the CPU is reading an external address.</td></tr>
<tr><td></td><td>$\overline{WR}$</td><td>36</td><td>28</td><td>O</td><td><b>Write</b>: Goes low to indicate that the CPU is writing to an external address.</td></tr>
<tr><td></td><td>$\overline{AS}$</td><td>35</td><td>27</td><td>O</td><td><b>Address strobe:</b> Goes low to indicate that there is a valid address on the address bus.</td></tr>
<tr><td>Interrupt signals</td><td>$\overline{NMI}$</td><td>13</td><td>5</td><td>I</td><td><b>Non maskable interrupt:</b> Highest-priority interrupt request. The NMIEG bit in the system control register determines whether the interrupt is requested on the rising or falling edge of the $\overline{NMI}$ input.</td></tr>
<tr><td></td><td>$\overline{IRQ}_0$ to $\overline{IRQ}_2$</td><td>5 to 7</td><td>61 to 63</td><td>I</td><td><b>Interrupt request 0 to 2:</b> Maskable interrupt request pins.</td></tr>
<tr><td>Operating mode control</td><td>MD_1,
MD_0</td><td>11
12</td><td>3
4</td><td>I</td><td><b>Mode:</b> Input pins for setting the MCU operating mode according to the table below.
<table>
<tr><th>MD_1</th><th>MD_0</th><th>Mode</th><th>Description</th></tr>
<tr><td>0</td><td>1</td><td>Mode 1</td><td>Expanded mode with on-chip ROM disabled</td></tr>
<tr><td>1</td><td>0</td><td>Mode 2</td><td>Expanded mode with on-chip ROM enabled</td></tr>
<tr><td>1</td><td>1</td><td>Mode 3</td><td>Single-chip mode</td></tr>
</table></td></tr>
</table>

Table 1-3 Pin Functions (cont)

		Pin No.			
Type	**Symbol**	**DP-64S**	**FP-64A**	**I/O**	**Name and Function**
16-bit free-running timer	FTCI	1	57	I	**FRT counter clock input:** Input pin for an external clock signal for the free-running counter.
	FTOA	2	58	O	**FRT output compare A:** Output pins controlled by comparator A of the free-running timer.
	FTOB	3	59	O	**FRT output compare B:** Output pins controlled by comparator B of the free-running timer.
	FTI	4	60	I	**FRT input capture:** Input capture pin for the free-running timer.
8-bit timer	TMO_0, TMO_1	18 21	10 13	O	**8-bit timer output (channels 0 and 1):** Compare- match output pins for the 8-bit timers.
	$TMCI_0$, $TMCI_1$	17 20	9 12	I	**8-bit timer clock input (channels 0 and 1):** External clock input pins for the 8-bit timer counters.
	$TMRI_0$, $TMRI_1$	19 22	11 14	I	**8-bit timer reset input (channels 0 and 1):** High input at these pins resets the 8-bit timers.
Serial commu-nication interface (SCI)	TxD_0 TxD_1	25 28	17 20	O	**Serial transmit data (channels 0 and 1):** Data output pins for the serial communication interface.
	RxD_0 RxD_1	26 29	18 21	I	**Serial receive data (channels 0 and 1):** Data input pins for the serial communication interface.
	SCK_0 SCK_1	27 30	19 22	I/O	**Serial clock (channels 0 and 1):** Input/output pins for the serial clock signals.

Table 1-3 Pin Functions (cont)

		Pin No.			
Type	**Symbol**	**DP-64S**	**FP-64A**	**I/O**	**Name and Function**
General-purpose I/O	$P1_7$ to $P1_0$	49 to 56	41 to 48	I/O	**Port 1:** An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 1 data direction register (P1DDR).
	$P2_7$ to $P2_0$	40 to 47	32 to 39	I/O	**Port 2:** An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 2 data direction register (P2DDR).
	$P3_7$ to $P3_0$	64 to 57	56 to 49	I/O	**Port 3:** An 8-bit input/output port with programmable MOS input pull-ups and LED drive capability. The direction of each bit can be selected in the port 3 data direction register (P3DDR).
	$P4_7$ to $P4_0$	24 to 17	16 to 9	I/O	**Port 4:** An 8-bit input/output port. The direction of each bit (except $P4_6$) can be selected in the port 4 data direction register (P4DDR).
	$P5_5$ to $P5_0$	30 to 25	22 to 17	I/O	**Port 5:** A 6-bit input/output port. The direction of each bit can be selected in the port 5 data direction register (P5DDR).
	$P6_6$ to $P6_0$	7 to 1	63 to 57	I/O	**Port 6:** A 7-bit input/output port. The direction of each bit can be selected in the port 6 data direction register (P6DDR).
	$P7_7$ to $P7_0$	38 to 31	30 to 23	I/O	**Port 7:** An 8-bit input/output port. The direction of each bit can be selected in the port 7 data direction register (P7DDR).

Table 1-3 Pin Functions (cont)

		Pin No.			
Type	**Symbol**	**DP-64S**	**FP-64A**	**I/O**	**Name and Function**
Host interface (HIF)	HDB_0 to HDB_7	57 to 64	49 to 56	I/O	**Host interface data bus:** Bidirectional 8-bit bus for host interface access by the host.
	$\overline{CS}_1$ $\overline{CS}_2$	35 23	27 15	I	**Chip select 1 and 2:** Input pins for selecting host interface channel 1 or channel 2.
	$\overline{IOR}$	37	29	I	**I/O read:** Input pin that enables reads on the host interface.
	$\overline{IOW}$	36	28	I	**I/O write:** Input pin that enables writes to the host interface.
	HA_0	38	30	I	**Command/data:** Input pin that indicates a data access or command access.
	GA_{20}	24	16	O	**GATE A_{20}:** GATE A_{20} control signal output pin.
	$HIRQ_1$ $HIRQ_{11}$ $HIRQ_{12}$	21 20 22	13 12 14	O	**Host interrupt 1, 11, 12:** Output pins for interrupt requests to the host.
Keyboard control	$\overline{KEYIN}_0$ to $\overline{KEYIN}_7$	1 to 4 31 to 34	57 to 60 23 to 26	I	**Keyboard input:** Input pins for a matrix keyboard. ($P1_0$ to $P1_7$ and $P2_0$ to $P2_7$ are normally used as keyboard scan outputs, enabling a maximum 16-output × 8-input, 128-key matrix to be configured. The number of keys can be increased by using other port outputs.)

Section 2 CPU

2.1 Overview

The H8/3502 has the generic H8/300 CPU: an 8-bit central processing unit with a speed-oriented architecture featuring sixteen general registers. This section describes the CPU features and functions, including a concise description of the addressing modes and instruction set. For further details on the instructions, see the *H8/300 Series Programming Manual*.

2.1.1 Features

The main features of the H8/300 CPU are listed below.

- Two-way register configuration
  - Sixteen 8-bit general registers, or
  - Eight 16-bit general registers
- Instruction set with 57 basic instructions, including:
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
- Eight addressing modes
  - Register direct (Rn)
  - Register indirect (@Rn)
  - Register indirect with displacement (@(d:16, Rn))
  - Register indirect with post-increment or pre-decrement (@Rn+ or @–Rn)
  - Absolute address (@aa:8 or @aa:16)
  - Immediate (#xx:8 or #xx:16)
  - PC-relative (@(d:8, PC))
  - Memory indirect (@@aa:8)
- Maximum 64-kbyte address space
- High-speed operation
  - All frequently-used instructions are executed two to four states
  - The maximum clock rate is 10 MHz/5 V (ø clock)
  - 8- or 16-bit register-register add or subtract: 200 ns (10 MHz)
  - 8 × 8-bit multiply: 1400 ns (10 MHz)
  - 16 ÷ 8-bit divide: 1400 ns (10 MHz)
- Power-down mode
  - SLEEP instruction

2.1.2 Address Space

The H8/300 CPU supports an address space of up to 64 kbytes for storing program code and data. The memory map is different for each mode (modes 1, 2, and 3). See section 3.5, Address Space Maps for Each Operating Mode, for details.

2.1.3 Register Configuration

Figure 2-1 shows the register structure of the CPU. There are two groups of registers: the general registers and control registers.

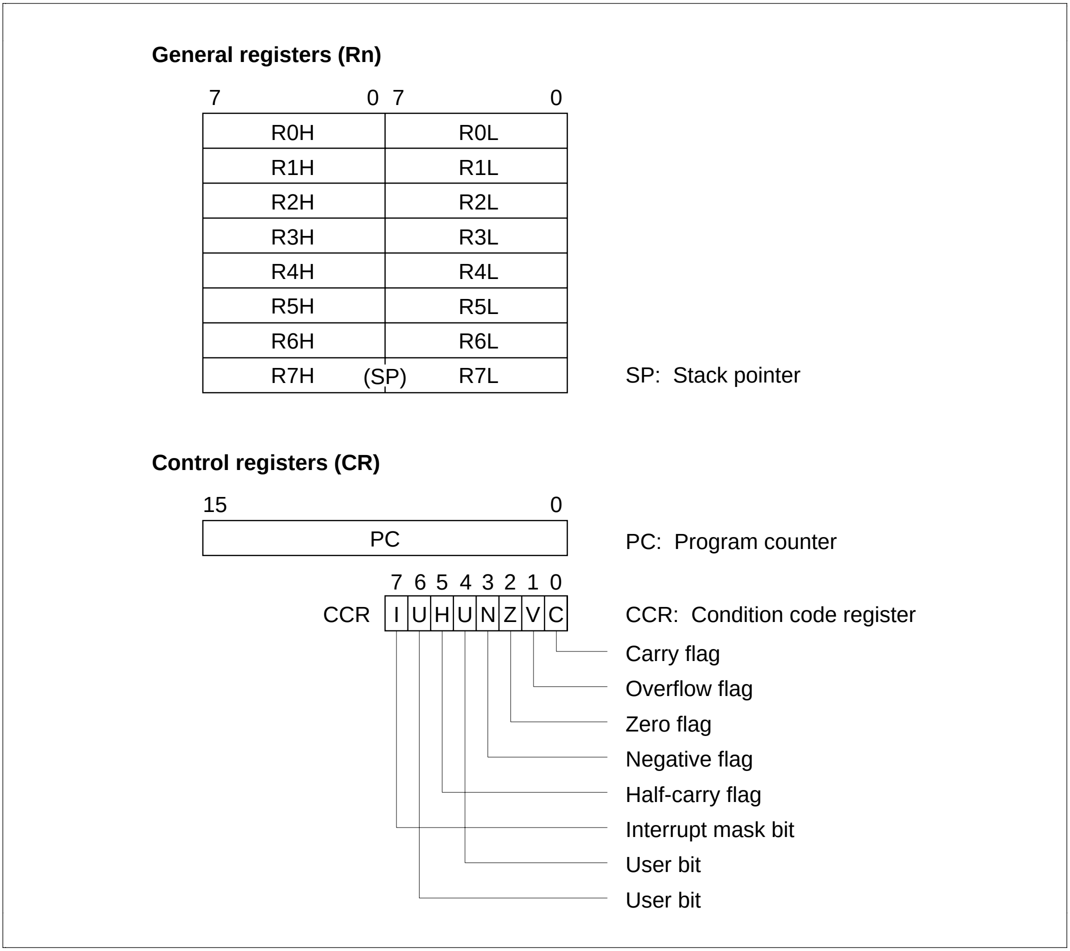


Figure 2-1 CPU Registers

2.2 Register Descriptions

2.2.1 General Registers

All the general registers can be used as both data registers and address registers. When used as address registers, the general registers are accessed as 16-bit registers (R0 to R7). When used as data registers, they can be accessed as 16-bit registers, or the high and low bytes can be accessed separately as 8-bit registers.

R7 also functions as the stack pointer, used implicitly by hardware in processing interrupts and subroutine calls. In assembly-language coding, R7 can also be denoted by the letters SP. As indicated in figure 2-2, R7 (SP) points to the top of the stack.

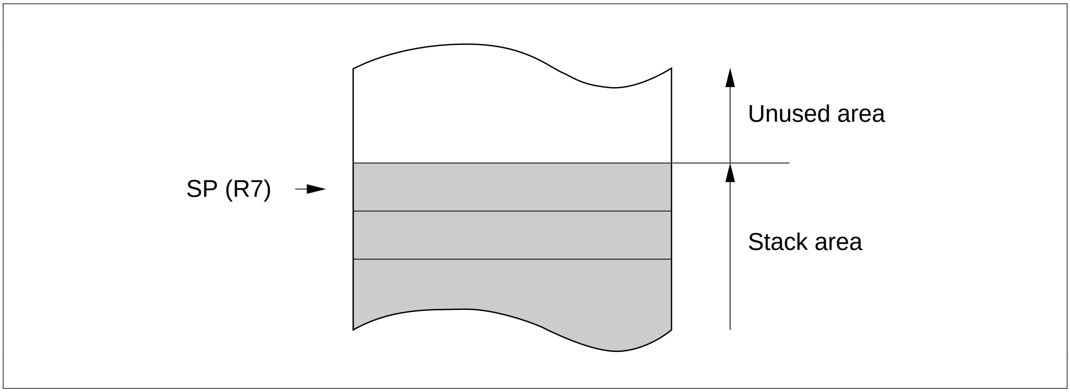


Figure 2-2 Stack Pointer

2.2.2 Control Registers

The CPU control registers include a 16-bit program counter (PC) and an 8-bit condition code register (CCR).

(1) Program Counter (PC): This 16-bit register indicates the address of the next instruction the CPU will execute. Each instruction is accessed in 16 bits (1 word), so the least significant bit of the PC is ignored (always regarded as 0).

(2) Condition Code Register (CCR): This 8-bit register contains internal status information, including carry (C), overflow (V), zero (Z), negative (N), and half-carry (H) flags and the interrupt mask bit (I).

Bit 7—Interrupt Mask Bit (I): When this bit is set to 1, all interrupts except NMI are masked. This bit is set to 1 automatically by a reset and at the start of interrupt handling.

Bit 6—User Bit (U): This bit can be written and read by software for its own purposes (using the LDC, STC, ANDC, ORC, and XORC instructions).

Bit 5—Half-Carry (H): This bit is set to 1 when the ADD.B, ADDX.B, SUB.B, SUBX.B, NEG.B, or CMP.B instruction causes a carry or borrow out of bit 3, and is cleared to 0 otherwise. Similarly, it is set to 1 when the ADD.W, SUB.W, or CMP.W instruction causes a carry or borrow out of bit 11, and cleared to 0 otherwise. It is used implicitly in the DAA and DAS instructions.

Bit 4—User Bit (U): This bit can be written and read by software for its own purposes (using the LDC, STC, ANDC, ORC, and XORC instructions).

Bit 3—Negative (N): This bit indicates the most significant bit (sign bit) of the result of an instruction.

Bit 2—Zero (Z): This bit is set to 1 to indicate a zero result and cleared to 0 to indicate a nonzero result.

Bit 1—Overflow (V): This bit is set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.

Bit 0—Carry (C): This bit is used by:

- Add and subtract instructions, to indicate a carry or borrow at the most significant bit of the result
- Shift and rotate instructions, to store the value shifted out of the most significant or least significant bit
- Bit manipulation and bit load instructions, as a bit accumulator

The LDC, STC, ANDC, ORC, and XORC instructions enable the CPU to load and store the CCR, and to set or clear selected bits by logic operations. The N, Z, V, and C flags are used in conditional branching instructions (Bcc).

Some instructions leave some or all of the flag bits unchanged. The action of each instruction on the flag bits is shown in Appendix A.1, Instruction Set List. See the *H8/300 Series Programming Manual* for further details.

2.2.3 Initial Register Values

When the CPU is reset, the program counter (PC) is loaded from the vector table and the interrupt mask bit (I) in the CCR is set to 1. The other CCR bits and the general registers are not initialized.

In particular, the stack pointer (R7) is not initialized. To prevent program crashes the stack pointer should be initialized by software, by the first instruction executed after a reset.

2.3 Data Formats

The H8/300 CPU can process 1-bit data, 4-bit (BCD) data, 8-bit (byte) data, and 16-bit (word) data.

- Bit manipulation instructions operate on 1-bit data specified as bit n (n = 0, 1, 2, ..., 7) in a byte operand.

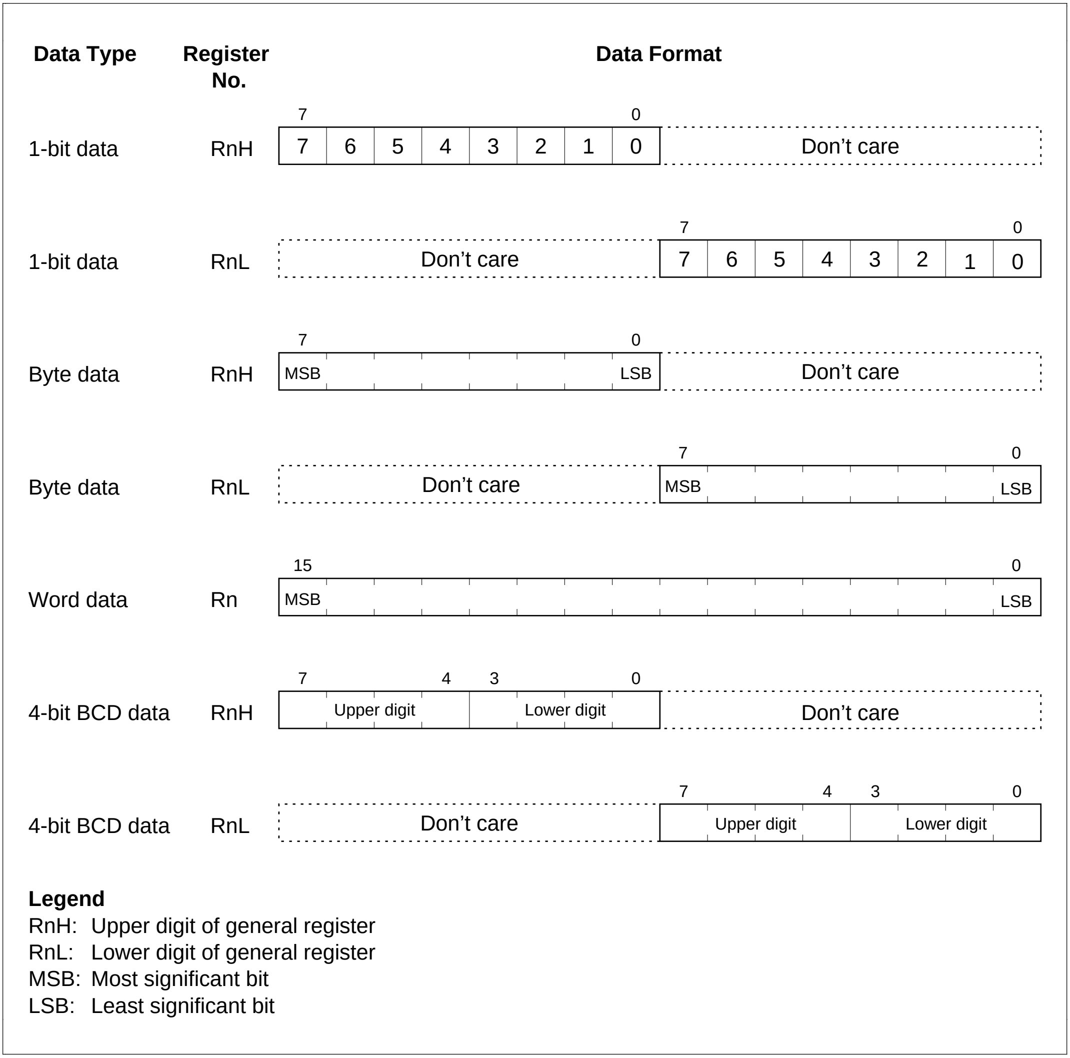
- All arithmetic and logic instructions except ADDS and SUBS can operate on byte data.

- The DAA and DAS instruction perform decimal arithmetic adjustments on byte data in packed BCD form. Each nibble of the byte is treated as a decimal digit.

- The MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions operate on word data.

2.3.1 Data Formats in General Registers

Data of all the sizes above can be stored in general registers as shown in figure 2-3.



Legend

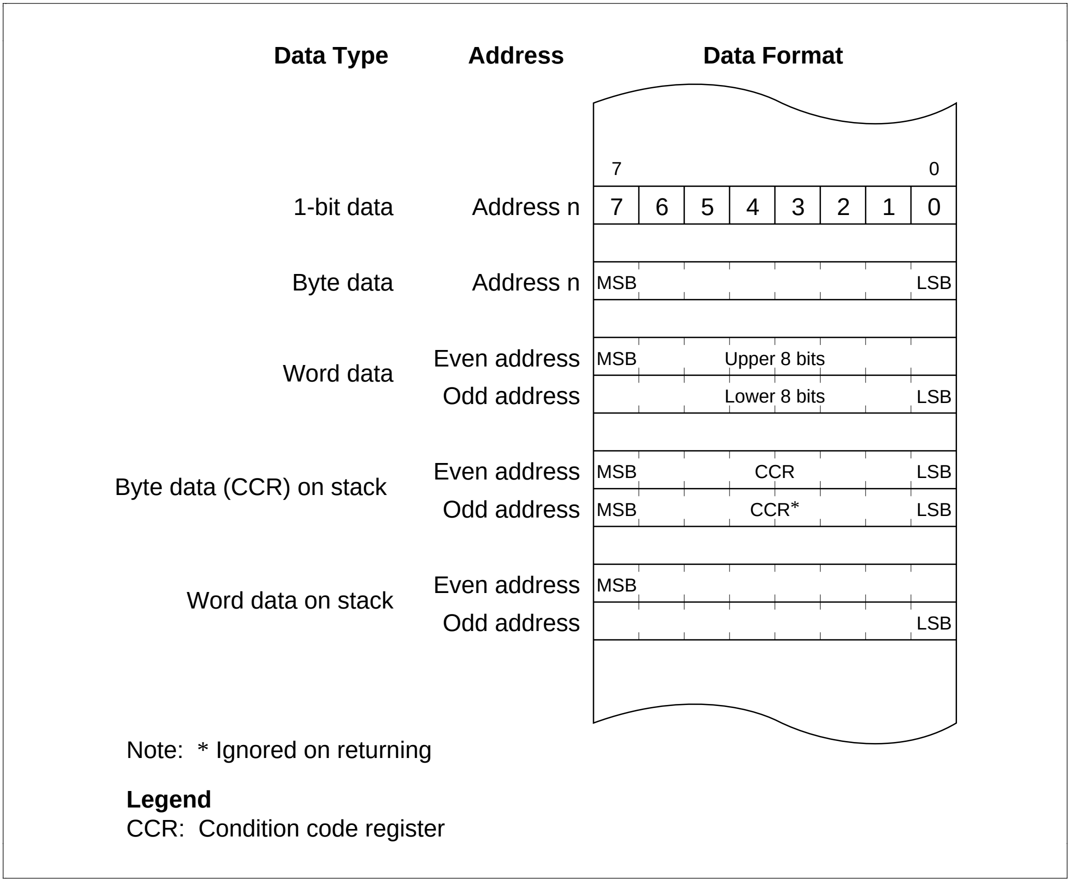
RnH: Upper digit of general register
RnL: Lower digit of general register
MSB: Most significant bit
LSB: Least significant bit

Figure 2-3 Register Data Formats

2.3.2 Memory Data Formats

Figure 2-4 indicates the data formats in memory.

Word data stored in memory must always begin at an even address. In word access the least significant bit of the address is regarded as 0. If an odd address is specified, no address error occurs but the access is performed at the preceding even address. This rule affects MOV.W instructions and branching instructions, and implies that only even addresses should be stored in the vector table.



Note: * Ignored on returning

Legend
CCR: Condition code register

Figure 2-4 Memory Data Formats

The stack must always be accessed a word at a time. When the CCR is pushed on the stack, two identical copies of the CCR are pushed to make a complete word. When they are returned, the lower byte is ignored.

2.4 Addressing Modes

2.4.1 Addressing Modes

The H8/300 CPU supports eight addressing modes. Each instruction uses a subset of these addressing modes.

(1) Register Direct—Rn: The register field of the instruction specifies an 8- or 16-bit general register containing the operand. In most cases the general register is accessed as an 8-bit register. Only the MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions have 16-bit operands.

(2) Register indirect—@Rn: The register field of the instruction specifies a 16-bit general register containing the address of the operand.

(3) Register Indirect with Displacement—@(d:16, Rn): This mode, which is used only in MOV instructions, is similar to register indirect but the instruction has a second word (bytes 3 and 4) which is added to the contents of the specified general register to obtain the operand address. For the MOV.W instruction, the resulting address must be even.

(4) Register Indirect with Post-Increment or Pre-Decrement—@Rn+ or @–Rn:

- Register indirect with Post-Increment—@Rn+

  The @Rn+ mode is used with MOV instructions that load registers from memory. It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is incremented after the operand is accessed. The size of the increment is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

- Register Indirect with Pre-Decrement—@–Rn

  The @–Rn mode is used with MOV instructions that store register contents to memory. It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is decremented before the operand is accessed. The size of the decrement is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

(5) Absolute Address—@aa:8 or @aa:16: The instruction specifies the absolute address of the operand in memory. The MOV.B instruction uses an 8-bit absolute address of the form H'FFxx. The upper 8 bits are assumed to be 1, so the possible address range is H'FF00 to H'FFFF (65280 to 65535). The MOV.B, MOV.W, JMP, and JSR instructions can use 16-bit absolute addresses.

(6) Immediate—#xx:8 or #xx:16: The instruction contains an 8-bit operand in its second byte, or a 16-bit operand in its third and fourth bytes. Only MOV.W instructions can contain 16-bit immediate values.

The ADDS and SUBS instructions implicitly contain the value 1 or 2 as immediate data. Some bit manipulation instructions contain 3-bit immediate data (#xx:3) in the second or fourth byte of the instruction, specifying a bit number.

(7) PC-Relative—@(d:8, PC): This mode is used to generate branch addresses in the Bcc and BSR instructions. An 8-bit value in byte 2 of the instruction code is added as a sign-extended value to the program counter contents. The result must be an even number. The possible branching range is –126 to +128 bytes (–63 to +64 words) from the current address.

(8) Memory Indirect—@@aa:8: This mode can be used by the JMP and JSR instructions. The second byte of the instruction code specifies an 8-bit absolute address from H'0000 to H'00FF (0 to 255). The word located at this address contains the branch address. Note that part of this area is located in the vector table. See section 3.5, Address Space Maps for Each Operating Mode, for details.

If an odd address is specified as a branch destination or as the operand address of a MOV.W instruction, the least significant bit is regarded as 0, causing word access to be performed at the address preceding the specified address. See section 2.3.2, Memory Data Formats, for further information.

2.4.2 Effective Address Calculation

Table 2-2 shows how an effective address (EA) is calculated in each addressing mode.

Arithmetic and logic instructions (ADD.B, ADDX.B, SUBX.B, CMP.B, AND.B, OR.B, XOR.B instructions) use (1) register direct and (6) immediate addressing modes.

Data transfer instructions can use all addressing modes except (7) program-counter relative and (8) memory indirect.

Bit manipulation instructions can use (1) register direct, (2) register indirect , or (5) absolute (@aa:8) addressing mode to specify an operand, and (1) register direct (BSET, BCLR, BNOT, and BTST instructions) or (6) immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2-2 Effective Address Calculation

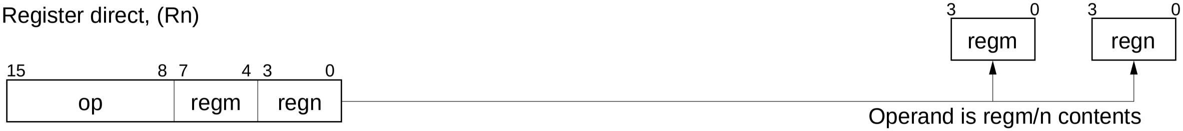
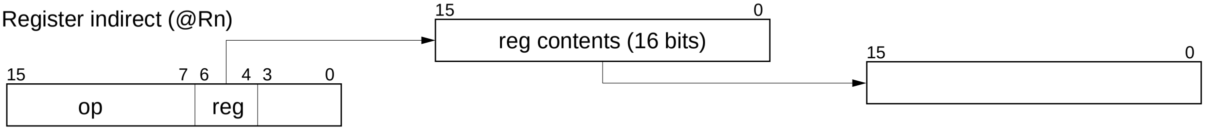
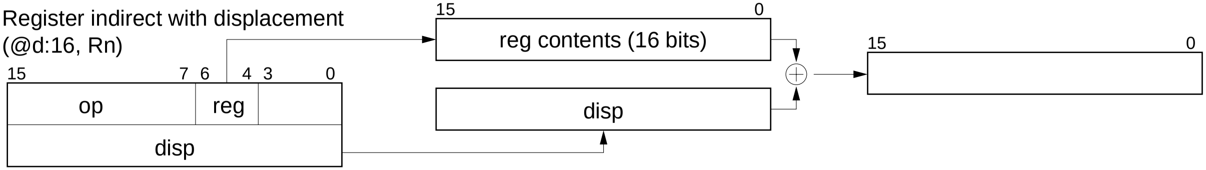
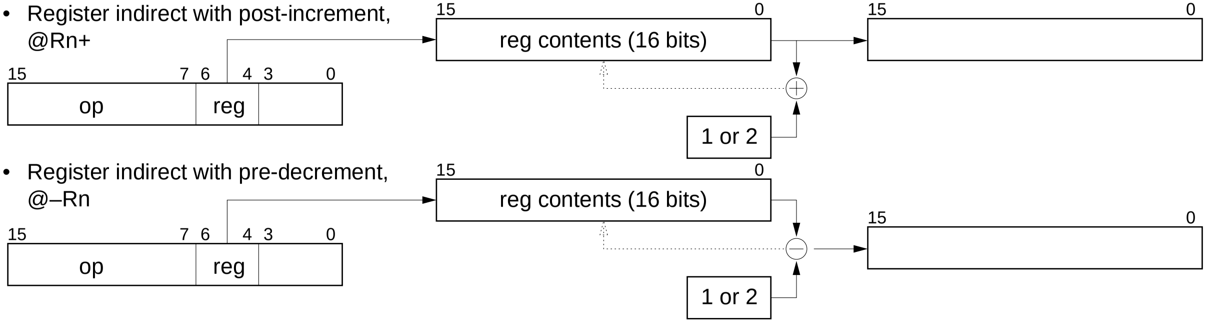
No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
(1)	Register direct, (Rn)	 15 8 7 4 3 0 op regm regn 3 0 regm 3 0 regn	Operand is regm/n contents
(2)	Register indirect (@Rn)	 15 7 6 4 3 0 op reg 15 0 reg contents (16 bits) 15 0	
(3)	Register indirect with displacement (@d:16, Rn)	 15 7 6 4 3 0 op reg disp 15 0 reg contents (16 bits) disp + 15 0	
(4)	Register indirect with post-increment or pre-decrement • Register indirect with post-increment, @Rn+ • Register indirect with pre-decrement, @–Rn	 15 7 6 4 3 0 op reg 15 0 reg contents (16 bits) + 1 or 2 15 0 15 7 6 4 3 0 op reg 15 0 reg contents (16 bits) – 1 or 2 15 0 Incremented or decremented by 1 if operand is byte size, and by 2 if word size	

Table 2-2 Effective Address Calculation (cont)

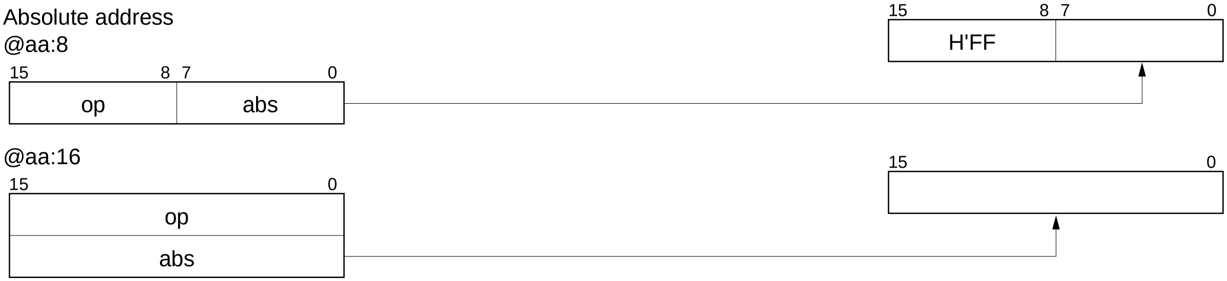
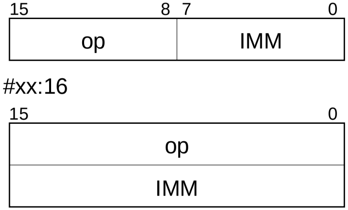
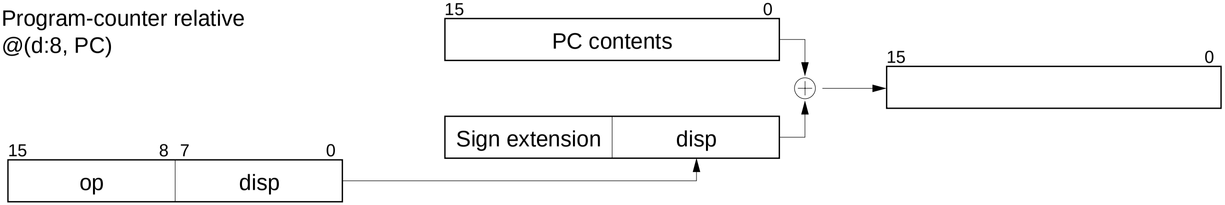
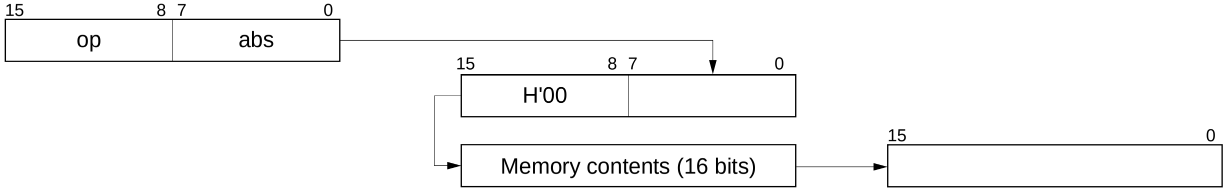
No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
(5)	Absolute address @aa:8 @aa:16	 15 8 7 0 op abs 15 0 op abs 15 8 7 0 H'FF 15 0	
(6)	Immediate #xx:8 #xx:16 		Operand is 1- or 2-byte immediate data
(7)	Program-counter relative @(d:8, PC)	 15 0 PC contents Sign extension disp 15 0 15 8 7 0 op disp	

Table 2-2 Effective Address Calculation (cont)

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
(8)	Memory indirect, @@aa:8	 15 8 7 0 op abs 15 8 7 0 H'00 Memory contents (16 bits) 15 0	

Legend

reg, regm, regn:	Register field
op:	Operation field
disp:	Displacement
IMM:	Immediate data
abs:	Absolute address

2.5 Instruction Set

Table 2-1 lists the H8/300 CPU instruction set.

Table 2-1 Instruction Classification

Function	Instructions	Types
Data transfer	MOV, MOVTPE*1, MOVFPE*1, PUSH*2, POP*2	3
Arithmetic operations	ADD, SUB, ADDX, SUBX, INC, DEC, ADDS, SUBS, DAA, DAS, MULXU, DIVXU, CMP, NEG	14
Logic operations	AND, OR, XOR, NOT	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	14
Branch	Bcc*3, JMP, BSR, JSR, RTS	5
System control	RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	8
Block data transfer	EEPMOV	1
	Total	57

Notes: 1. These instructions cannot be used with the H8/3502.

2. PUSH Rn is equivalent to MOV.W Rn, @–SP.
POP Rn is equivalent to MOV.W @SP+, Rn.

3. Bcc is a conditional branch instruction in which cc represents a condition code.

The following sections give a concise summary of the instructions in each category, and indicate the bit patterns of their object code. The notation used is defined next.

Operation Notation

Rd	General register (destination)
Rs	General register (source)
Rn, Rm	General register
r_n, r_m	General register field
<EAs>	Effective address: general register or memory location
(EAd)	Destination operand
(EAs)	Source operand
SP	Stack pointer
PC	Program counter
CCR	Condition code register
N	N (negative) bit of CCR
Z	Z (zero) bit of CCR
V	V (overflow) bit of CCR
C	C (carry) bit of CCR
#imm	Immediate data
#xx:3	3-bit immediate data
#xx:8	8-bit immediate data
#xx:16	16-bit immediate data

op	Operation field
disp	Displacement
abs	Absolute address
B	Byte
W	Word
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Exclusive logical OR
→	Move
↔	Exchange
¬	NOT (logical complement)
cc	Condition field

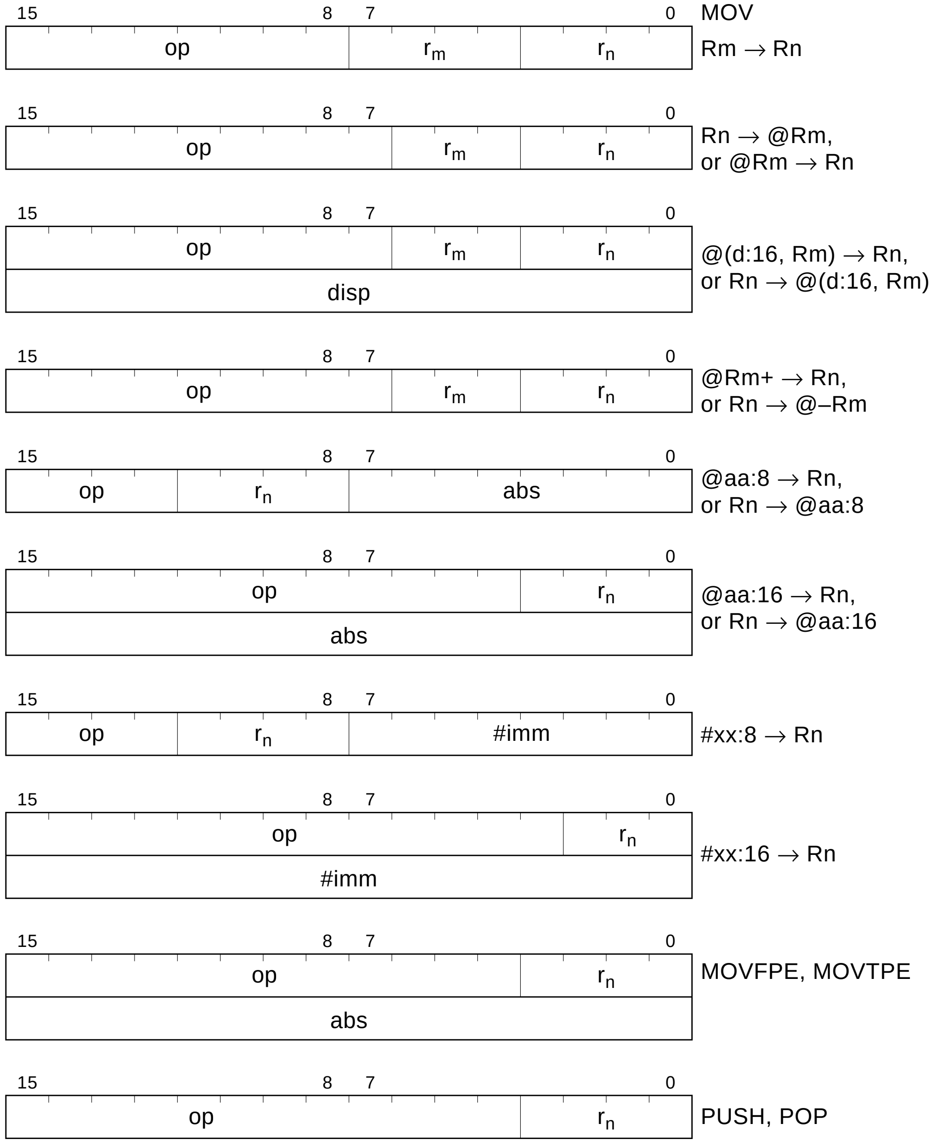
2.5.1 Data Transfer Instructions

Table 2-2 describes the data transfer instructions. Figure 2-5 shows their object code formats.

Table 2-2 Data Transfer Instructions

Instruction	Size*	Function
MOV	B/W	(EAs) → Rd, Rs → (EAd) Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. The Rn, @Rn, @(d:16, Rn), @aa:16, #xx:8 or #xx:16, @–Rn, and @Rn+ addressing modes are available for byte or word data. The @aa:8 addressing mode is available for byte data only. The @–R7 and @R7+ modes require word operands. Do not specify byte size for these two modes.
MOVTPE	B	Cannot be used with the H8/3502.
MOVFPE	B	Cannot be used with the H8/3502.
PUSH	W	Rn → @–SP Pushes a 16-bit general register onto the stack. Equivalent to MOV.W Rn, @–SP.
POP	W	@SP+ → Rn Pops a 16-bit general register from the stack. Equivalent to MOV.W @SP+, Rn.

Note: * Size: operand size
B: Byte
W: Word



Legend

op:	Operation field
r_m, r_n:	Register field
disp:	Displacement
abs:	Absolute address
#imm:	Immediate data

Figure 2-5 Data Transfer Instruction Codes

2.5.2 Arithmetic Operations

Table 2-3 describes the arithmetic instructions. See figure 2-6 in section 2.5.4, Shift Operations for their object codes.

Table 2-3 Arithmetic Instructions

Instruction	Size*	Function
ADD SUB	B/W	Rd ± Rs → Rd, Rd + #imm → Rd Performs addition or subtraction on data in two general registers, or addition on immediate data and data in a general register. Immediate data cannot be subtracted from data in a general register. Word data can be added or subtracted only when both words are in general registers.
ADDX SUBX	B	Rd ± Rs ± C → Rd, Rd ± #imm ± C → Rd Performs addition or subtraction with carry or borrow on byte data in two general registers, or addition or subtraction on immediate data and data in a general register.
INC DEC	B	Rd ± #1 → Rd Increments or decrements a general register.
ADDS SUBS	W	Rd ± #imm → Rd Adds or subtracts immediate data to or from data in a general register. The immediate data must be 1 or 2.
DAA DAS	B	Rd decimal adjust → Rd Decimal-adjusts (adjusts to packed BCD) an addition or subtraction result in a general register by referring to the CCR.
MULXU	B	Rd × Rs → Rd Performs 8-bit × 8-bit unsigned multiplication on data in two general registers, providing a 16-bit result.
DIVXU	B	Rd ÷ Rs → Rd Performs 16-bit ÷ 8-bit unsigned division on data in two general registers, providing an 8-bit quotient and 8-bit remainder.
CMP	B/W	Rd – Rs, Rd – #imm Compares data in a general register with data in another general register or with immediate data. Word data can be compared only between two general registers.
NEG	B	0 – Rd → Rd Obtains the two's complement (arithmetic complement) of data in a general register.

Note: * Size: operand size
B: Byte
W: Word

2.5.3 Logic Operations

Table 2-4 describes the four instructions that perform logic operations. See figure 2-6 in section 2.5.4, Shift Operations for their object codes.

Table 2-4 Logic Operation Instructions

Instruction	Size*	Function
AND	B	Rd ∧ Rs → Rd, Rd ∧ #imm → Rd Performs a logical AND operation on a general register and another general register or immediate data.
OR	B	Rd ∨ Rs → Rd, Rd ∨ #imm → Rd Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B	Rd ⊕ Rs → Rd, Rd ⊕ #imm → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B	¬ (Rd) → Rd Obtains the one’s complement (logical complement) of general register contents.

Note: * Size: operand size
B: Byte

2.5.4 Shift Operations

Table 2-5 describes the eight shift instructions. Figure 2-6 shows the object code formats of the arithmetic, logic, and shift instructions.

Table 2-5 Shift Instructions

Instruction	Size*	Function
SHAL SHAR	B	Rd shift → Rd Performs an arithmetic shift operation on general register contents.
SHLL SHLR	B	Rd shift → Rd Performs a logical shift operation on general register contents.
ROTL ROTR	B	Rd rotate → Rd Rotates general register contents.
ROTXL ROTXR	B	Rd rotate through carry → Rd Rotates general register contents through the C (carry) bit.

Note: * Size: operand size
B: Byte

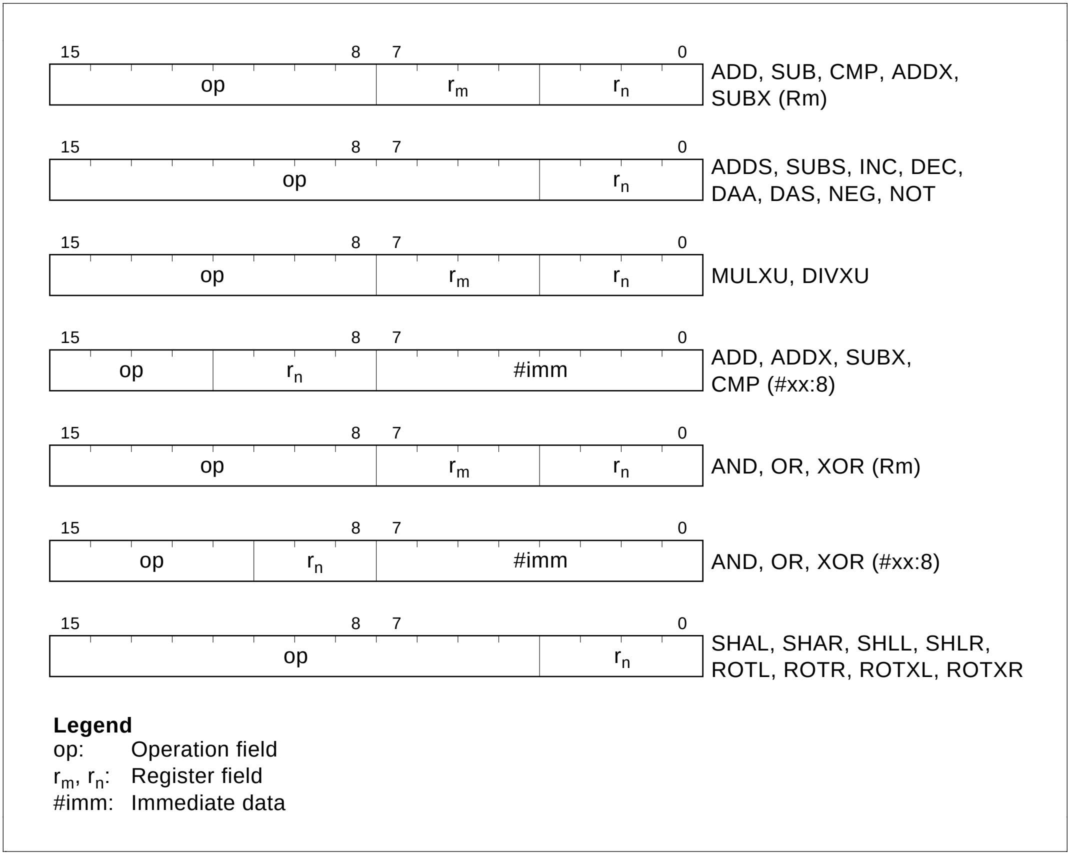


Figure 2-6 Arithmetic, Logic, and Shift Instruction Codes

2.5.5 Bit Manipulations

Table 2-6 describes the bit-manipulation instructions. Figure 2-7 shows their object code formats.

Table 2-6 Bit-Manipulation Instructions

Instruction	Size*	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in a general register or memory to 1. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BCLR	B	0 → (<bit-No.> of <EAd>) Clears a specified bit in a general register or memory to 0. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BNOT	B	¬(<bit-No.> of <EAd>) → (<bit-No.> of <EAd>) Inverts a specified bit in a general register or memory. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BTST	B	¬ (<bit-No.> of <EAd>) → Z Tests a specified bit in a general register or memory and sets or clears the Z flag accordingly. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BAND	B	C ∧ (<bit-No.> of <EAd>) → C ANDs the C flag with a specified bit in a general register or memory.
BIAND		C ∧ [¬ (<bit-No.> of <EAd>)] → C ANDs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BOR	B	C ∨ (<bit-No.> of <EAd>) → C ORs the C flag with a specified bit in a general register or memory.
BIOR		C ∨ [¬ (<bit-No.> of <EAd>)] → C ORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C XORs the C flag with a specified bit in a general register or memory.
BIXOR	B	C ⊕ [¬ (<bit-No.> of <EAd>)] → C XORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.

Table 2-6 Bit-Manipulation Instructions (cont)

Instruction	Size*	Function
BLD	B	(<bit-No.> of <EAd>) → C Copies a specified bit in a general register or memory to the C flag.
BILD		¬ (<bit-No.> of <EAd>) → C Copies the inverse of a specified bit in a general register or memory to the C flag.
		The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Copies the C flag to a specified bit in a general register or memory.
BIST		¬ C → (<bit-No.> of <EAd>) Copies the inverse of the C flag to a specified bit in a general register or memory.
		The bit number is specified by 3-bit immediate data.

Note: * Size: operand size
B: Byte

Notes on Bit Manipulation Instructions: BSET, BCLR, BNOT, BST, and BIST are read-modify-write instructions. They read a byte of data, modify one bit in the byte, then write the byte back. Care is required when these instructions are applied to registers with write-only bits and to the I/O port registers.

Order	Operation
Read	Read one data byte at the specified address
Modify	Modify one bit in the data byte
Write	Write the modified data byte back to the specified address

Example: BCLR is executed to clear bit 0 in the port 1 data direction register (P1DDR) under the following conditions.

$P1_7$:	Input pin, Low
$P1_6$:	Input pin, High
$P1_5$–$P1_0$:	Output pins, Low

The intended purpose of this BCLR instruction is to switch $P1_0$ from output to input.

Before Execution of BCLR Instruction

	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	Low
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	0

Execution of BCLR Instruction

```
    BCLR.B  #0  @P1DDR          ; Clear bit 0 in data direction register
```

After Execution of BCLR Instruction

	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$
Input/output	Output	Output	Output	Output	Output	Output	Output	Input
Pin state	Low	High	Low	Low	Low	Low	Low	High
DDR	1	1	1	1	1	1	1	0
DR	1	0	0	0	0	0	0	0

Explanation: To execute the BCLR instruction, the CPU begins by reading P1DDR. Since P1DDR is a write-only register, it is read as H'FF, even though its true value is H'3F.

Next the CPU clears bit 0 of the read data, changing the value to H'FE.

Finally, the CPU writes this value (H'FE) back to P1DDR to complete the BCLR instruction.

As a result, $P1_0$DDR is cleared to 0, making $P1_0$ an input pin. In addition, $P1_7$DDR and $P1_6$DDR are set to 1, making $P1_7$ and $P1_6$ output pins.

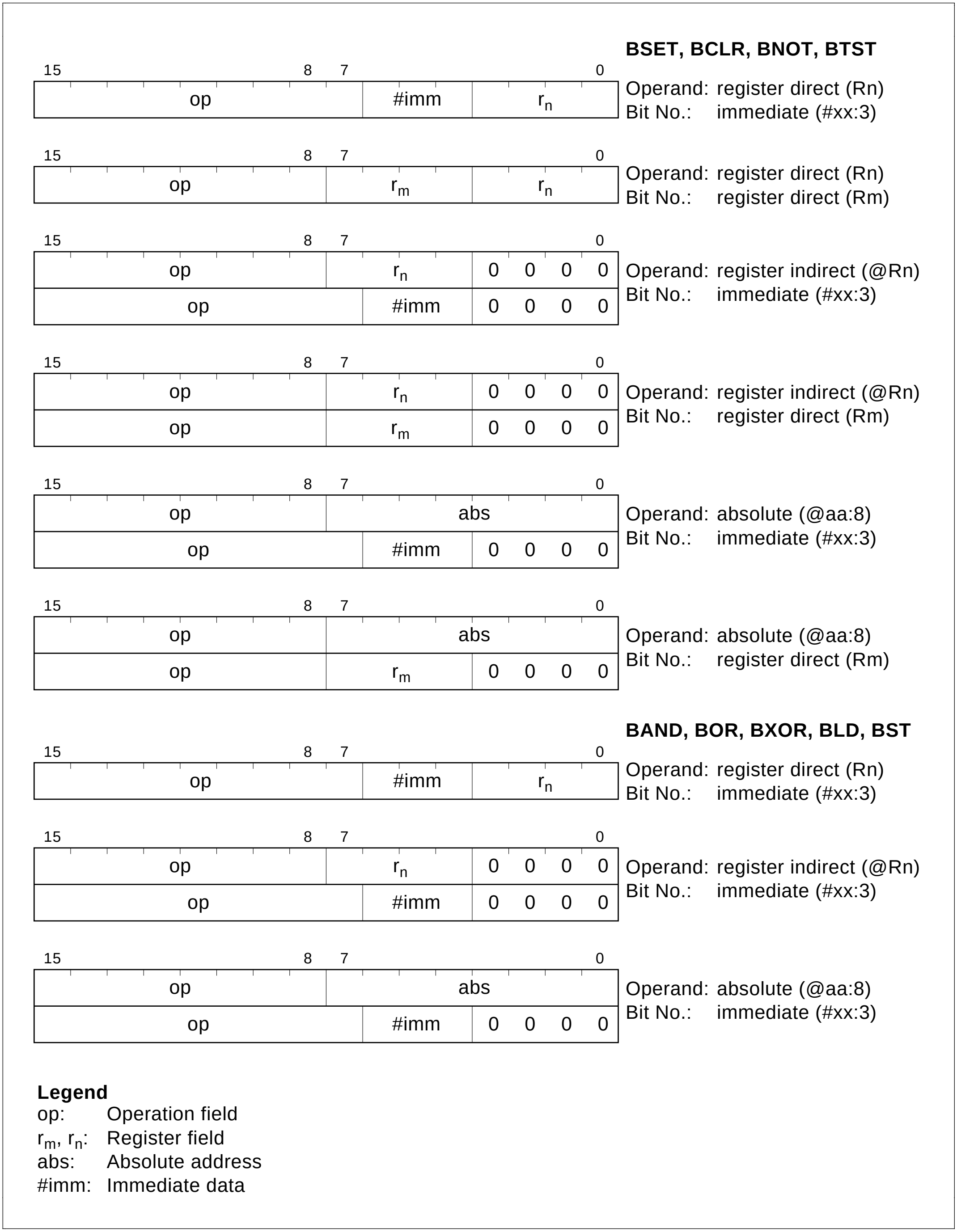
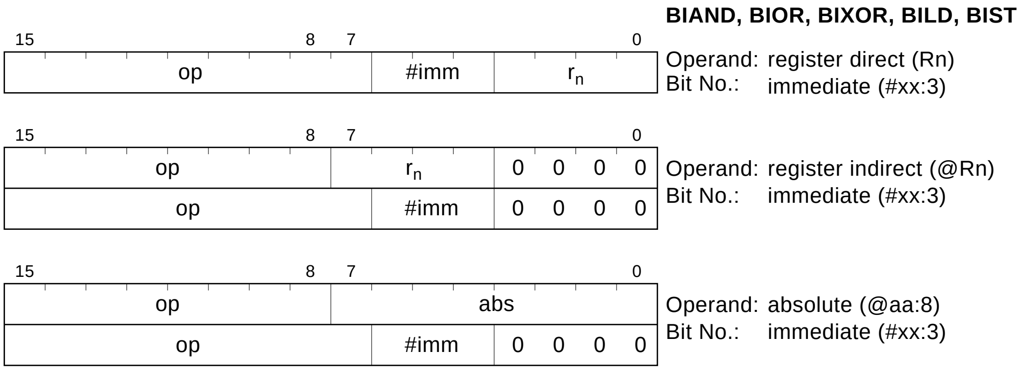

BSET, BCLR, BNOT, BTST
15 8 7 0
op | #imm | rn
Operand: register direct (Rn)
Bit No.: immediate (#xx:3)
15 8 7 0
op | rm | rn
Operand: register direct (Rn)
Bit No.: register direct (Rm)
15 8 7 0
op | rn | 0 0 0 0
op | #imm | 0 0 0 0
Operand: register indirect (@Rn)
Bit No.: immediate (#xx:3)
15 8 7 0
op | rn | 0 0 0 0
op | rm | 0 0 0 0
Operand: register indirect (@Rn)
Bit No.: register direct (Rm)
15 8 7 0
op | abs
op | #imm | 0 0 0 0
Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)
15 8 7 0
op | abs
op | rm | 0 0 0 0
Operand: absolute (@aa:8)
Bit No.: register direct (Rm)
BAND, BOR, BXOR, BLD, BST
15 8 7 0
op | #imm | rn
Operand: register direct (Rn)
Bit No.: immediate (#xx:3)
15 8 7 0
op | rn | 0 0 0 0
op | #imm | 0 0 0 0
Operand: register indirect (@Rn)
Bit No.: immediate (#xx:3)
15 8 7 0
op | abs
op | #imm | 0 0 0 0
Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)
Legend
op: Operation field
rm, rn: Register field
abs: Absolute address
#imm: Immediate data

Figure 2-7 Bit Manipulation Instruction Codes


BIAND, BIOR, BIXOR, BILD, BIST
15
8
7
0
op
#imm
r_n
Operand: register direct (Rn)
Bit No.: immediate (#xx:3)
15
8
7
0
op
r_n
0 0 0 0
op
#imm
0 0 0 0
Operand: register indirect (@Rn)
Bit No.: immediate (#xx:3)
15
8
7
0
op
abs
op
#imm
0 0 0 0
Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)

Legend

op:	Operation field
r_m, r_n:	Register field
abs:	Absolute address
#imm:	Immediate data

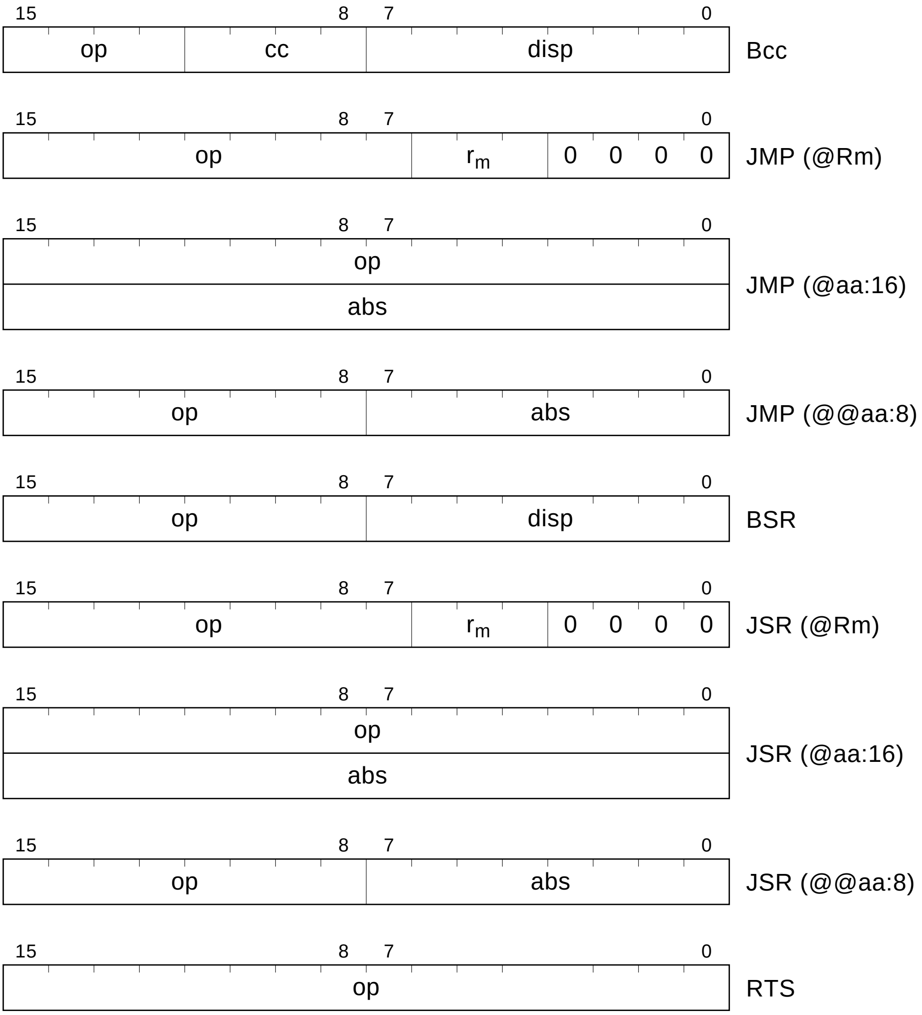
Figure 2-7 Bit Manipulation Instruction Codes (cont)

2.5.6 Branching Instructions

Table 2-7 describes the branching instructions. Figure 2-8 shows their object code formats.

Table 2-7 Branching Instructions

<table>
<tr><th>Instruction</th><th>Size</th><th>Function</th></tr>
<tr><td>Bcc</td><td>—</td><td>Branches if condition cc is true.
<table>
<tr><th>Mnemonic</th><th>cc Field</th><th>Description</th><th>Condition</th></tr>
<tr><td>BRA (BT)</td><td>0 0 0 0</td><td>Always (true)</td><td>Always</td></tr>
<tr><td>BRN (BF)</td><td>0 0 0 1</td><td>Never (false)</td><td>Never</td></tr>
<tr><td>BHI</td><td>0 0 1 0</td><td>High</td><td>C ∨ Z = 0</td></tr>
<tr><td>BLS</td><td>0 0 1 1</td><td>Low or same</td><td>C ∨ Z = 1</td></tr>
<tr><td>BCC (BHS)</td><td>0 1 0 0</td><td>Carry clear
(high or same)</td><td>C = 0</td></tr>
<tr><td>BCS (BLO)</td><td>0 1 0 1</td><td>Carry set (low)</td><td>C = 1</td></tr>
<tr><td>BNE</td><td>0 1 1 0</td><td>Not equal</td><td>Z = 0</td></tr>
<tr><td>BEQ</td><td>0 1 1 1</td><td>Equal</td><td>Z = 1</td></tr>
<tr><td>BVC</td><td>1 0 0 0</td><td>Overflow clear</td><td>V = 0</td></tr>
<tr><td>BVS</td><td>1 0 0 1</td><td>Overflow set</td><td>V = 1</td></tr>
<tr><td>BPL</td><td>1 0 1 0</td><td>Plus</td><td>N = 0</td></tr>
<tr><td>BMI</td><td>1 0 1 1</td><td>Minus</td><td>N = 1</td></tr>
<tr><td>BGE</td><td>1 1 0 0</td><td>Greater or equal</td><td>N ⊕ V = 0</td></tr>
<tr><td>BLT</td><td>1 1 0 1</td><td>Less than</td><td>N ⊕ V = 1</td></tr>
<tr><td>BGT</td><td>1 1 1 0</td><td>Greater than</td><td>Z ∨ (N ⊕ V) = 0</td></tr>
<tr><td>BLE</td><td>1 1 1 1</td><td>Less or equal</td><td>Z ∨ (N ⊕ V) = 1</td></tr>
</table></td></tr>
<tr><td>JMP</td><td>—</td><td>Branches unconditionally to a specified address.</td></tr>
<tr><td>JSR</td><td>—</td><td>Branches to a subroutine at a specified address.</td></tr>
<tr><td>BSR</td><td>—</td><td>Branches to a subroutine at a specified displacement from the current address.</td></tr>
<tr><td>RTS</td><td>—</td><td>Returns from a subroutine</td></tr>
</table>



Legend

op:	Operation field
cc:	Condition field
r_m:	Register field
disp:	Displacement
abs:	Absolute address

Figure 2-8 Branching Instruction Codes

2.5.7 System Control Instructions

Table 2-8 describes the system control instructions. Figure 2-9 shows their object code formats.

Table 2-8 System Control Instructions

Instruction	Size*	Function
RTE	—	Returns from an exception-handling routine.
SLEEP	—	Causes a transition to the power-down state.
LDC	B	Rs → CCR, #imm → CCR Moves immediate data or general register contents to the condition code register.
STC	B	CCR → Rd Copies the condition code register to a specified general register.
ANDC	B	CCR ∧ #imm → CCR Logically ANDs the condition code register with immediate data.
ORC	B	CCR ∨ #imm → CCR Logically ORs the condition code register with immediate data.
XORC	B	CCR ⊕ #imm → CCR Logically exclusive-ORs the condition code register with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

Note: * Size: operand size
B: Byte

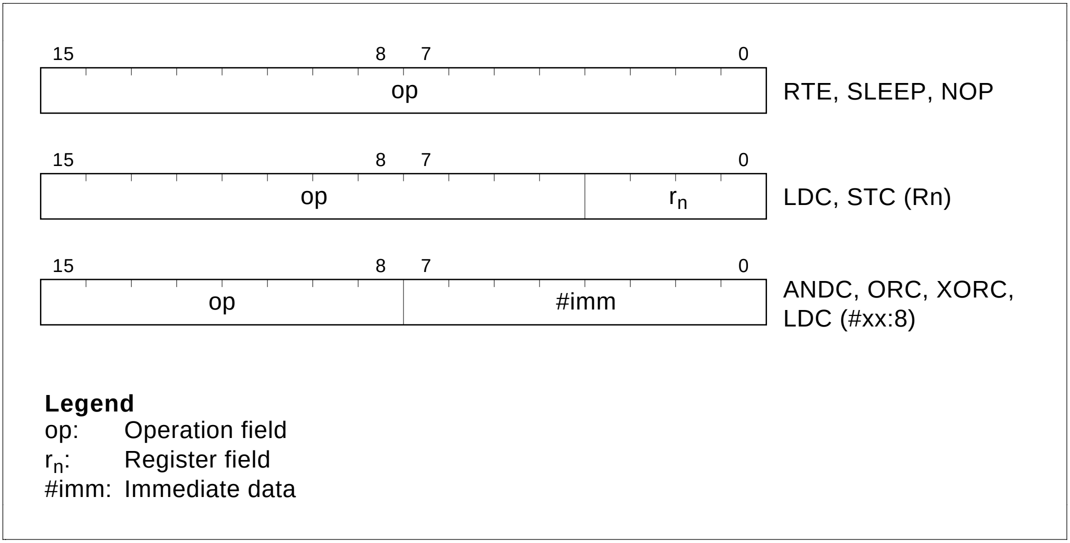


Figure 2-9 System Control Instruction Codes

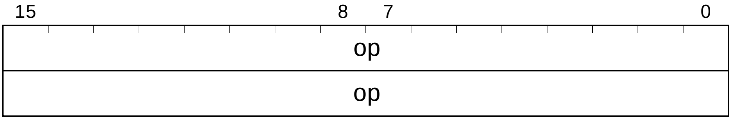
2.5.8 Block Data Transfer Instruction

Table 2-9 describes the EEPMOV instruction. Figure 2-10 shows its object code format.

Table 2-9 Block Data Transfer Instruction

Instruction	Size	Function

EEPMOV	—	if R4L ≠ 0 then
		  repeat @R5+ → @R6+
		    R4L – 1 → R4L
		  until R4L = 0
		else next;
		Moves a data block according to parameters set in general registers R4L, R5, and R6.
		R4L: size of block (bytes)
		R5: starting source address
		R6: starting destination address
		Execution of the next instruction starts as soon as the block transfer is completed.

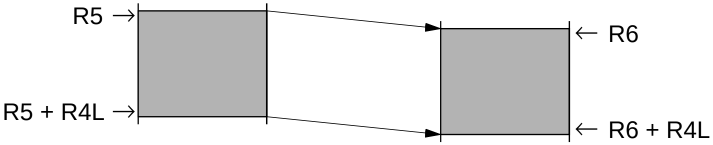


Legend
op: Operation field

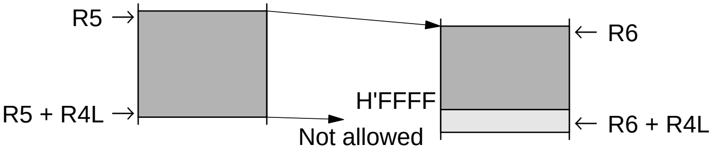
Figure 2-10 Block Data Transfer Instruction

Notes on EEPMOV Instruction

- The EEPMOV instruction is a block data transfer instruction. It moves the number of bytes specified by R4L from the address specified by R5 to the address specified by R6.



- When setting R4L and R6, make sure that the final destination address (R6 + R4L) does not exceed H'FFFF. The value in R6 must not change from H'FFFF to H'0000 during execution of the instruction.



2.6 CPU States

2.6.1 Overview

The CPU has three states: the program execution state, exception-handling state, and power-down state. The power-down state is further divided into three modes: the sleep mode, software standby mode, and hardware standby mode. Figure 2-11 summarizes these states, and figure 2-12 shows a map of the state transitions.

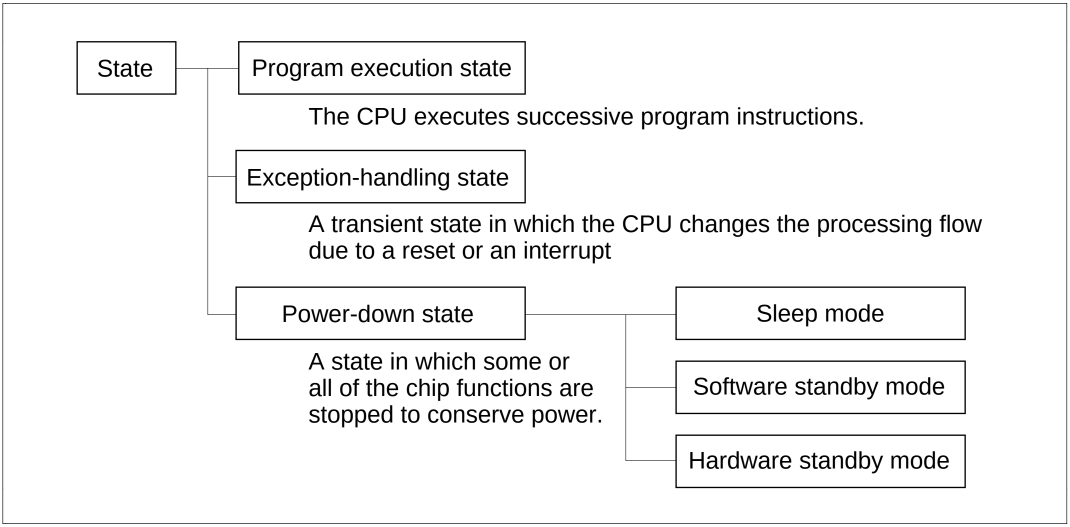
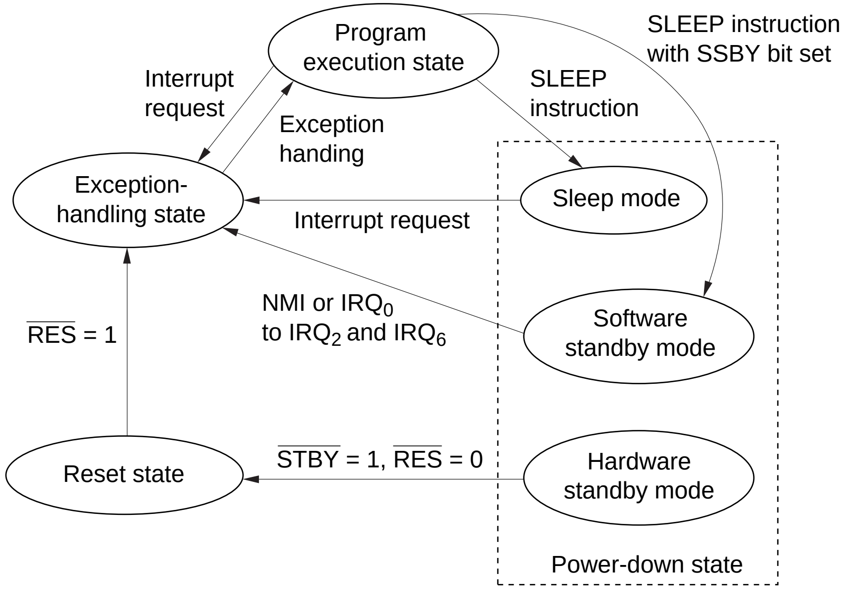

State
Program execution state
The CPU executes successive program instructions.
Exception-handling state
A transient state in which the CPU changes the processing flow due to a reset or an interrupt
Power-down state
A state in which some or all of the chip functions are stopped to conserve power.
Sleep mode
Software standby mode
Hardware standby mode

Figure 2-11 Operating States



Notes: 1. A transition to the reset state occurs when $\overline{\text{RES}}$ goes low, except when the chip is in the hardware standby mode.

2. A transition from any state to the hardware standby mode occurs when $\overline{\text{STBY}}$ goes low.

Figure 2-12 State Transitions

2.6.2 Program Execution State

In this state the CPU executes program instructions in sequence. The main program, subroutines, and interrupt-handling routines are all executed in this state.

2.6.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU is reset or interrupted and changes its normal processing flow. In interrupt exception handling, the CPU references the stack pointer (R7) and saves the program counter and condition code register on the stack. For further details see section 4, Exception Handling.

2.6.4 Power-Down State

The power-down state includes three modes: the sleep mode, the software standby mode, and the hardware standby mode.

(1) Sleep Mode: The sleep mode is entered when a SLEEP instruction is executed. The CPU halts, but CPU register contents remain unchanged and the on-chip supporting modules continue to function.

When an interrupt or reset signal is received, the CPU returns through the exception-handling state to the program execution state.

(2) Software Standby Mode: The software standby mode is entered if the SLEEP instruction is executed while the SSBY (Software Standby) bit in the system control register (SYSCR) is set. The CPU and all on-chip supporting modules halt. The on-chip supporting modules are initialized, but the contents of the on-chip RAM and CPU registers remain unchanged. I/O port outputs also remain unchanged.

(3) Hardware Standby Mode: The hardware standby mode is entered when the input at the $\overline{\text{STBY}}$ pin goes low. All chip functions halt, including I/O port output. The on-chip supporting modules are initialized, but on-chip RAM contents are held.

See section 18, Power-Down State, for further information.

2.7 Access Timing and Bus Cycle

The CPU is driven by the system clock (ø). The period from one rising edge of the system clock to the next is referred to as a “state.”

Memory access is performed in a two- or three-state bus cycle as described below. Different accesses are performed to on-chip memory, the on-chip register field, and external devices. For more detailed timing diagrams of the bus cycles, see section 19, Electrical Specifications.

2.7.1 Access to On-Chip Memory (RAM and ROM)

On-chip ROM and RAM are accessed in a cycle of two states designated T_1 and T_2. Either byte or word data can be accessed, via a 16-bit data bus. Figure 2-13 shows the on-chip memory access cycle. Figure 2-14 shows the associated pin states.

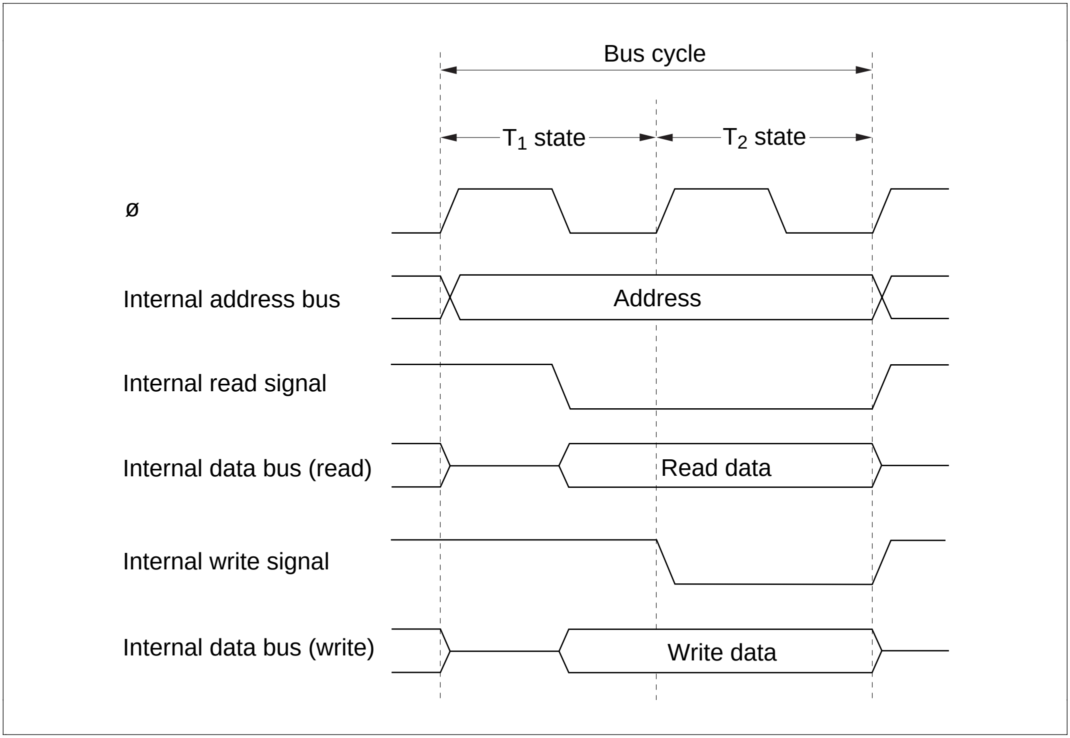


Figure 2-13 On-Chip Memory Access Cycle

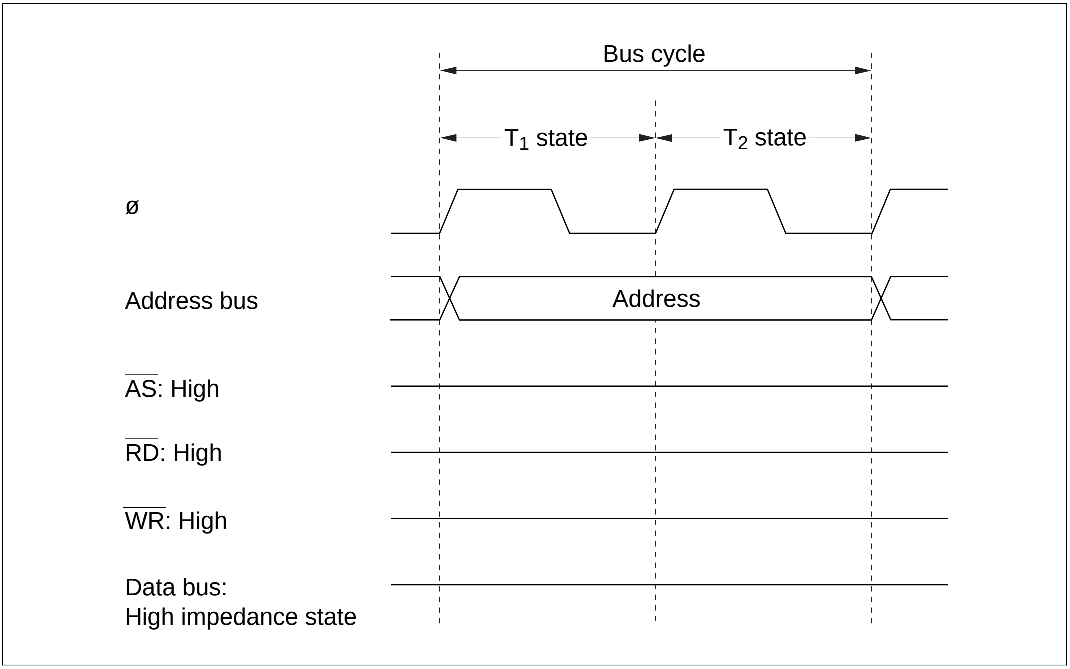


Figure 2-14 Pin States during On-Chip Memory Access Cycle

2.7.2 Access to On-Chip Register Field and External Devices

The on-chip register field (I/O ports, dual-port RAM, on-chip supporting module registers, etc.) and external devices are accessed in a cycle consisting of three states: T_1, T_2, and T_3. Only one byte of data can be accessed per cycle, via an 8-bit data bus. Access to word data or instruction codes requires two consecutive cycles (six states).

Figure 2-15 shows the access cycle for the on-chip register field. Figure 2-16 shows the associated pin states. Figures 2-17 (a) and (b) show the read and write access timing for external devices.

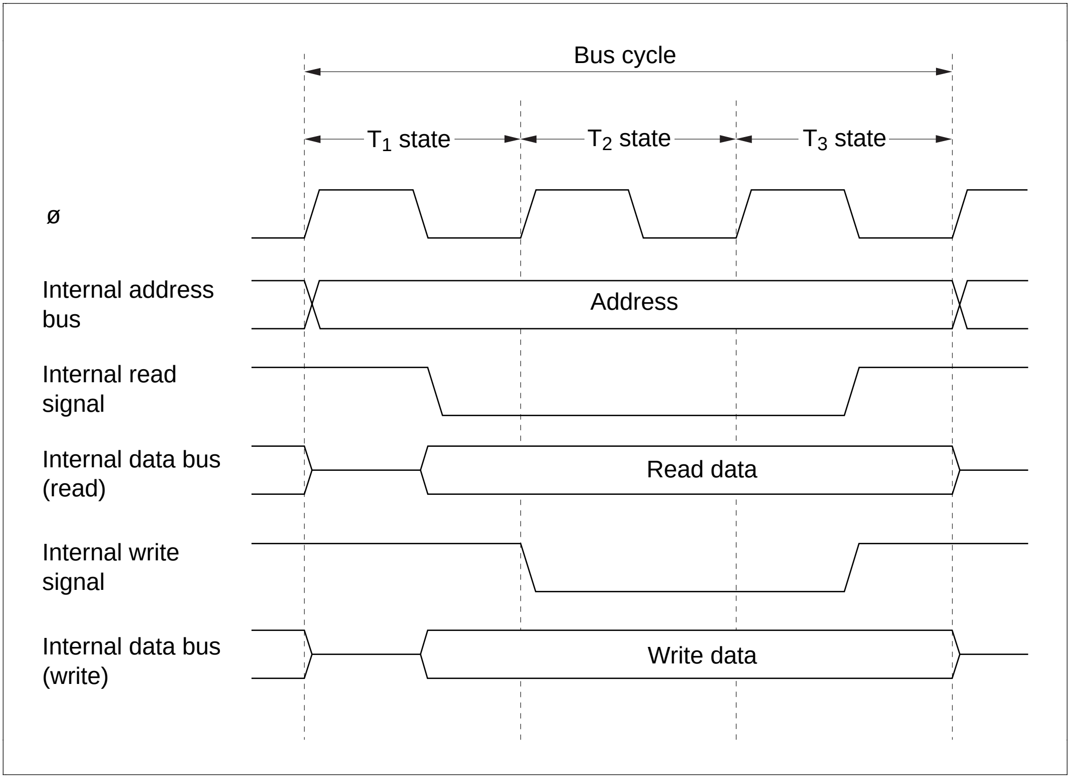


Figure 2-15 On-Chip Register Field Access Cycle

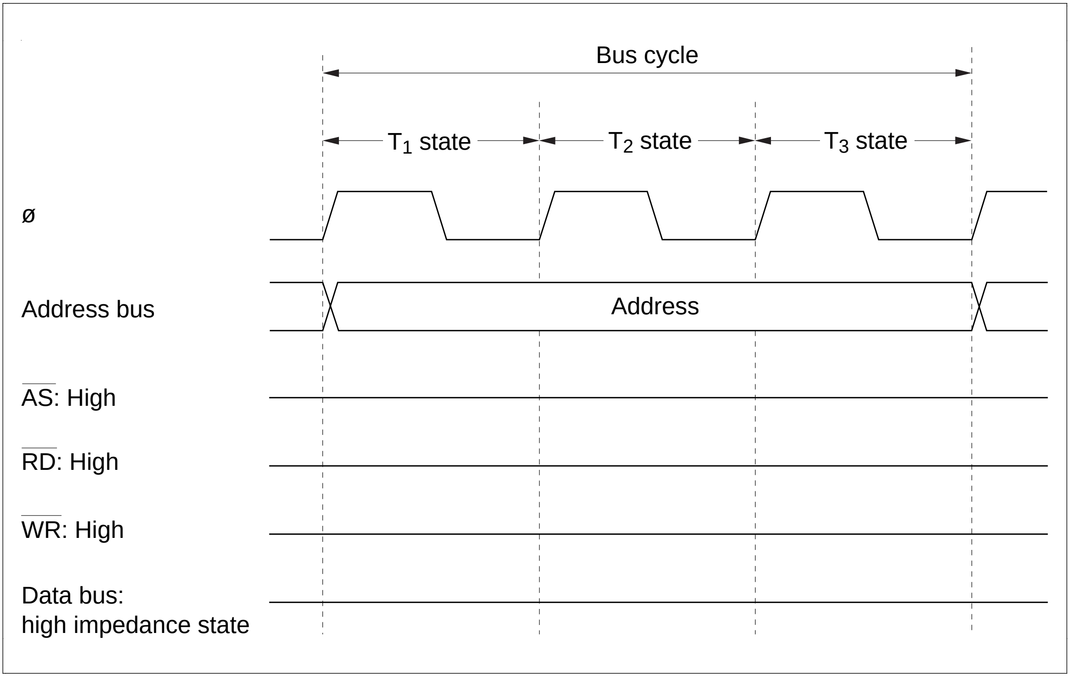


Figure 2-16 Pin States during On-Chip Supporting Module Access

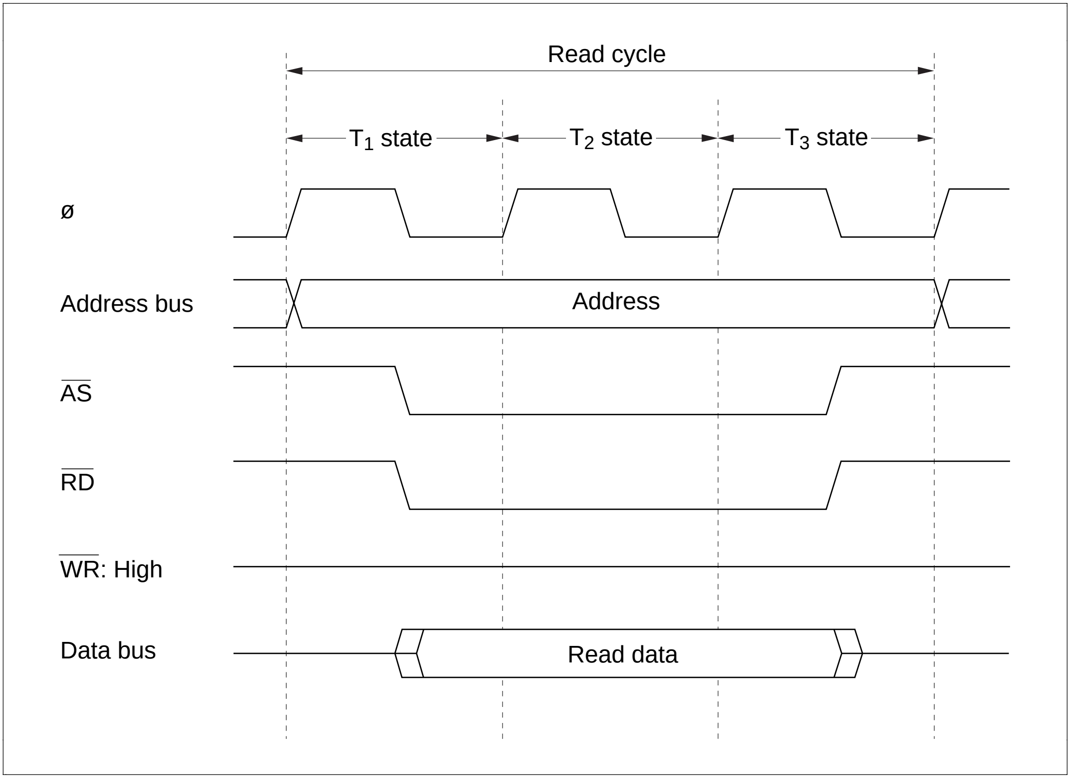


Figure 2-17 (a) External Device Access Timing (Read)

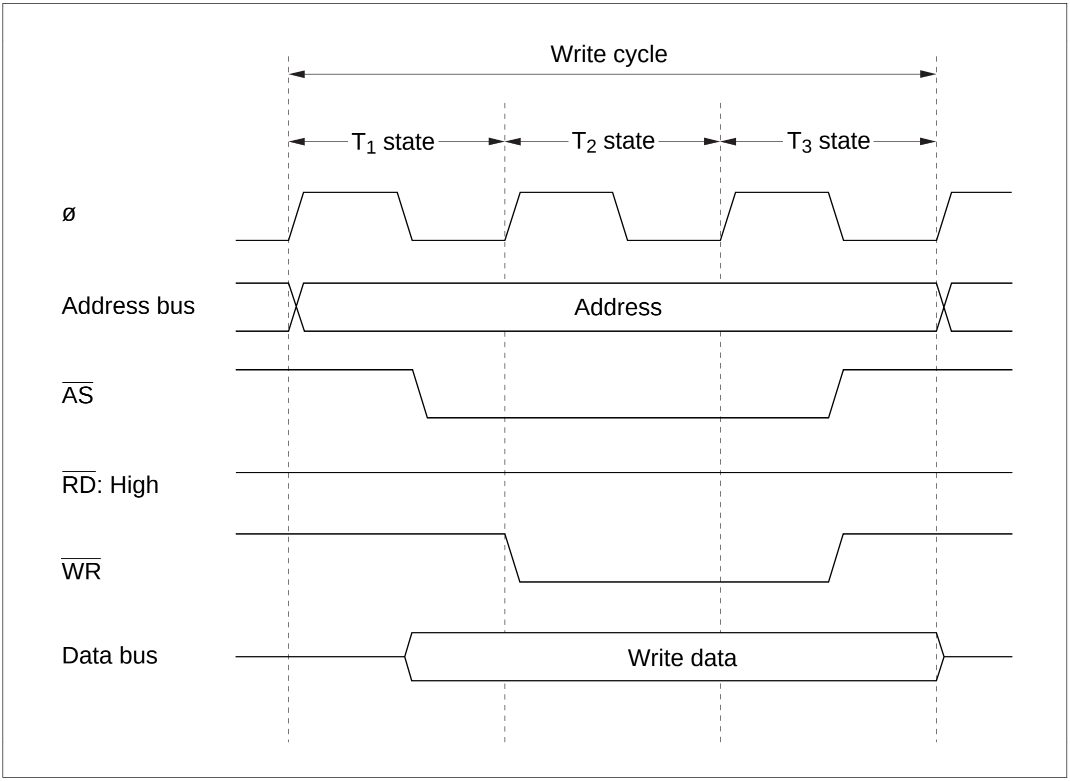


Figure 2-17 (b) External Device Access Timing (Write)

Section 3 MCU Operating Modes and Address Space

3.1 Overview

3.1.1 Operating Modes

The H8/3502 operates in three modes numbered 1, 2, and 3. The mode is selected by the inputs at the mode pins (MD$_1$ and MD$_0$). See table 3-1.

Table 3-1 Operating Modes

Mode	MD$_1$	MD$_0$	Address Space	On-Chip ROM	On-Chip RAM
Mode 0	Low	Low	—	—	—
Mode 1	Low	High	Expanded	Disabled	Enabled*
Mode 2	High	Low	Expanded	Enabled	Enabled*
Mode 3	High	High	Single-chip	Enabled	Enabled

Note: * If the RAME bit in the system control register (SYSCR) is cleared to 0, off-chip memory can be accessed instead.

Modes 1 and 2 are expanded modes that permit access to off-chip memory and peripheral devices. The maximum address space supported by these externally expanded modes is 64 kbytes.

In mode 3 (single-chip mode), only on-chip ROM and RAM and the on-chip register field are used. All ports are available for general-purpose input and output.

Mode 0 is inoperative in the H8/3502. Avoid setting the mode pins to mode 0.

3.1.2 Mode and System Control Registers

Table 3-2 lists the registers related to the chip’s operating mode: the system control register (SYSCR) and mode control register (MDCR). The mode control register indicates the inputs to the mode pins MD1 and MD0.

Table 3-2 Mode and System Control Registers

Name	Abbreviation	Read/Write	Address
System control register	SYSCR	R/W	H'FFC4
Mode control register	MDCR	R	H'FFC5

3.2 System Control Register (SYSCR)

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

The system control register (SYSCR) is an 8-bit register that controls the operation of the chip.

Bit 7—Software Standby (SSBY): Enables transition to the software standby mode. For details, see section 18, Power-Down State.

On recovery from software standby mode by an external interrupt, the SSBY bit remains set to 1. It can be cleared by writing 0.

Bit 7 SSBY	Description	
0	The SLEEP instruction causes a transition to sleep mode.	(Initial value)
1	The SLEEP instruction causes a transition to software standby mode.	

Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0): These bits select the clock settling time when the chip recovers from the software standby mode by an external interrupt. During the selected time the CPU and on-chip supporting modules continue to stand by. These bits should be set according to the clock frequency so that the settling time is at least 8 ms. For specific settings, see section 18.3.3, Clock Settling Time for Exit from Software Standby Mode.

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8,192 states	(Initial value)
0	0	1	Settling time = 16,384 states	
0	1	0	Settling time = 32,768 states	
0	1	1	Settling time = 65,536 states	
1	0	—	Settling time = 131,072 states	
1	1	—	Unused	

Bit 3—External Reset (XRST): Indicates the source of a reset. A reset can be generated by input of an external reset signal, or by a watchdog timer overflow when the watchdog timer is used. XRST is a read-only bit. It is set to 1 by an external reset, and cleared to 0 by watchdog timer overflow.

Bit 3 XRST	Description	
0	Reset was caused by watchdog timer overflow.	
1	Reset was caused by external input.	(Initial value)

Bit 2—NMI Edge (NMIEG): Selects the valid edge of the $\overline{\text{NMI}}$ input.

Bit 2 NMIEG	Description	
0	An interrupt is requested on the falling edge of the $\overline{\text{NMI}}$ input.	(Initial value)
1	An interrupt is requested on the rising edge of the $\overline{\text{NMI}}$ input.	

Bit 1—Host Interface Enable (HIE): Enables or disables the host interface function. When enabled, the host interface processes host-slave data transfers, operating in slave mode.

Bit 1 HIE	Description	
0	The host interface is disabled.	(Initial value)
1	The host interface is enabled (slave mode).	

Bit 0—RAM Enable (RAME): Enables or disables the on-chip RAM. The RAME bit is initialized by a reset, but is not initialized in the software standby mode.

Bit 0 RAME	Description	
0	The on-chip RAM is disabled.	
1	The on-chip RAM is enabled.	(Initial value)

3.3 Mode Control Register (MDCR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	MDS1	MDS0
Initial value	1	1	1	0	0	1	*	*
Read/Write	—	—	—	—	—	—	R	R

Note: * Initialized according to MD$_1$ and MD$_0$ inputs.

The mode control register (MDCR) is an 8-bit register that indicates the operating mode of the chip.

Bits 7 to 5—Reserved: These bits cannot be modified and are always read as 1.

Bits 4 and 3—Reserved: These bits cannot be modified and are always read as 0.

Bit 2—Reserved: This bit cannot be modified and is always read as 1.

Bits 1 and 0—Mode Select 1 and 0 (MDS1 and MDS0): These bits indicate the values of the mode pins (MD$_1$ and MD$_0$), thereby indicating the current operating mode of the chip. MDS1 corresponds to MD$_1$ and MDS$_0$ to MD$_0$. These bits can be read but not written. When the mode control register is read, the levels at the mode pins (MD$_1$ and MD$_0$) are latched in these bits.

3.4 Mode Descriptions

Mode 1 (Expanded Mode without On-Chip ROM): Mode 1 supports a 64-kbyte address space most of which is off-chip. In particular, the interrupt vector table is located in off-chip memory. The on-chip ROM is not used. Software can select whether to use the on-chip RAM. Ports 1, 2, 3 and 7 are used for the address and data bus lines and control signals as follows:

Ports 1 and 2:	Address bus
Port 3:	Data bus
Port 7 (partly):	Bus control signals

Mode 2 (Expanded Mode with On-Chip ROM): Mode 2 supports a 64-kbyte address space which includes the on-chip ROM. Software can select whether or not to use the on-chip RAM, and can select the usage of pins in ports 1 and 2.

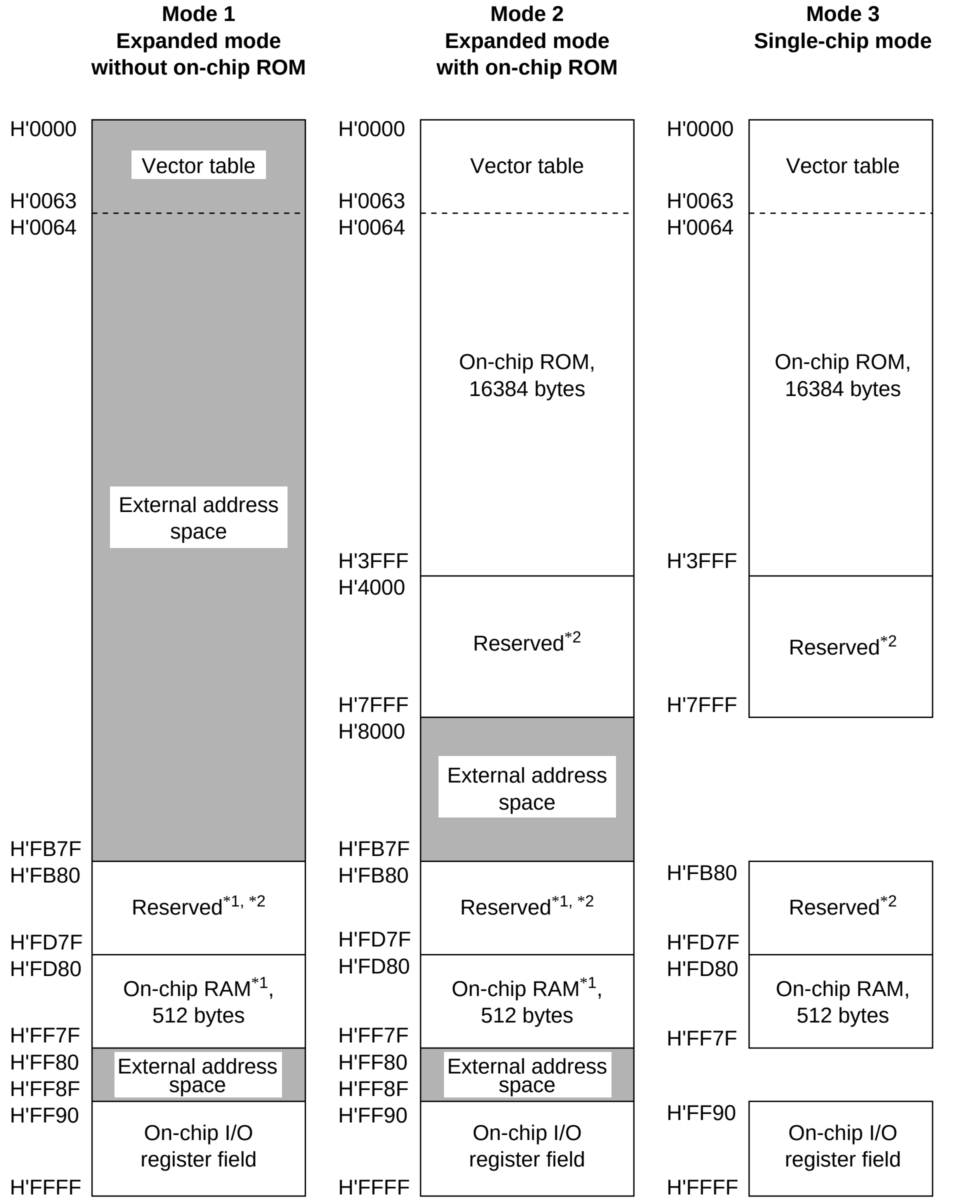
Ports 1 and 2:	Address bus (see note)
Port 3:	Data bus
Port 7 (partly):	Bus control signals

Note: In mode 2, ports 1 and 2 are initially general-purpose input ports. Software must change the desired pins to output before using them for the address bus. See section 7, I/O Ports for details.

Mode 3 (Single-Chip Mode): In this mode all memory is on-chip. Since no off-chip memory is accessed, there is no external address bus. All ports are available for general-purpose input and output.

3.5 Address Space Maps for Each Operating Mode

Figure 3-1 shows memory maps of the H8/3502 in each of the three operating modes.



Notes: *1. External memory can be accessed at these addresses when the RAME bit in the system control register (SYSCR) is cleared to 0.

*2. Do not access reserved areas.

Figure 3-1 Address Space Map

Section 4 Exception Handling

4.1 Overview

The H8/3502 recognizes only two kinds of exceptions: interrupts and the reset. Table 4-1 indicates their priority and the timing of their hardware exception-handling sequence.

Table 4-1 Reset and Interrupt Exceptions

Priority	Type of Exception	Detection Timing	Timing of Exception-Handling Sequence
High ↓	Reset	Clock synchronous	When $\overline{\text{RES}}$ goes low, the chip enters the reset state immediately. The hardware exception-handling sequence (reset sequence) begins as soon as $\overline{\text{RES}}$ goes high again.
Low	Interrupt	On completion of instruction execution*	When an interrupt is requested, the hardware exception-handling sequence (interrupt sequence) begins at the end of the current instruction, or at the end of the current hardware exception-handling sequence.

Note: * Not detected in case of ANDC, ORC, XORC, and LDC instructions.

4.2 Reset

4.2.1 Overview

A reset has the highest exception-handling priority. When the $\overline{\text{RES}}$ pin goes low or a watchdog reset is started (watchdog timer overflow for which the reset option is selected), all current processing stops and the chip enters the reset state. The internal state of the CPU and the registers of the on-chip supporting modules are initialized. When $\overline{\text{RES}}$ returns from low to high or the watchdog reset pulse ends, the chip comes out of the reset state via the reset exception-handling sequence.

4.2.2 Reset Sequence

The reset state begins when $\overline{\text{RES}}$ goes low or a watchdog reset occurs. To ensure correct resetting, at power-on the $\overline{\text{RES}}$ pin should be held low for at least 20 ms. In a reset during operation, the $\overline{\text{RES}}$ pin should be held low for at least 10 system clock (ø) cycles. The watchdog reset pulse width is always 518 system clock cycles. For details of pin states in a reset, see appendix D, Pin States.

When reset exception handling is started, hardware carries out the following reset sequence.

1. In the condition code register (CCR), the I bit is set to 1 to mask interrupts.
2. The registers of the I/O ports and on-chip supporting modules are initialized.
3. The CPU loads the program counter with the first word in the vector table (stored at addresses H'0000 and H'0001) and starts program execution.

The $\overline{\text{RES}}$ pin should be held low when power is switched off, as well as when power is switched on.

Figure 4-1 indicates the timing of the reset sequence when the vector table and reset routine are located in on-chip ROM (mode 2 or 3). Figure 4-2 indicates the timing when they are in off-chip memory (mode 1).

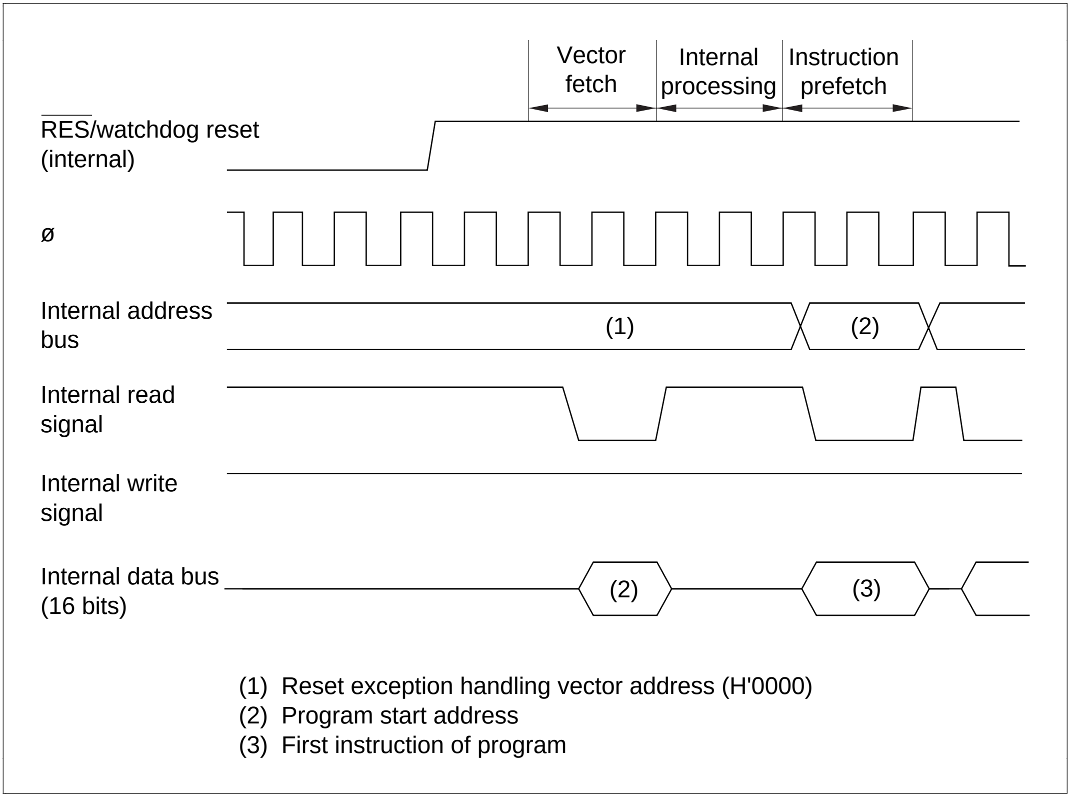
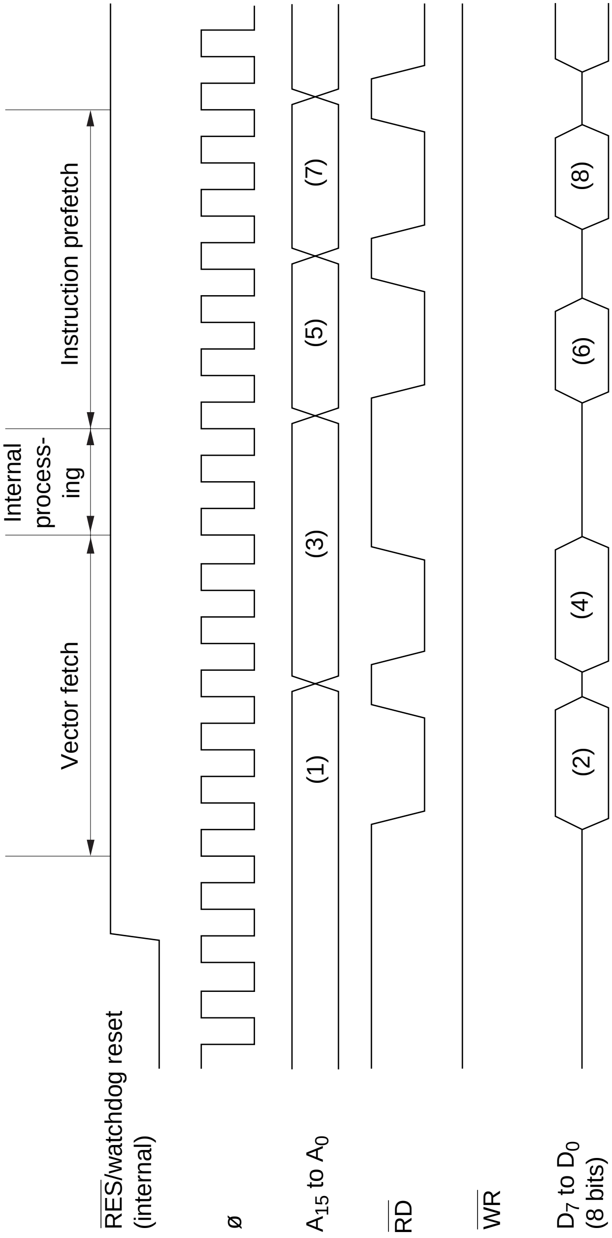

Vector fetch
Internal processing
Instruction prefetch
RES/watchdog reset (internal)
ø
Internal address bus
(1)
(2)
Internal read signal
Internal write signal
Internal data bus (16 bits)
(2)
(3)
(1) Reset exception handling vector address (H'0000)
(2) Program start address
(3) First instruction of program

Figure 4-1 Reset Sequence (Mode 2 or 3, Program Area in On-Chip ROM)



(1), (3) Reset exception handling vector address: (1) = H'0000, (3) = H'0001
(2), (4) Start address (contents of reset exception handling vector address): (2) = upper byte, (4) = lower byte
(5), (7) Start address: (5) = (2) (4), (7) = (2) (4) + 1
(6), (8) First instruction of program: (6) = first byte, (8) = second byte

Figure 4-2 Reset Sequence (Mode 1)

4.2.3 Disabling of Interrupts after Reset

All interrupts, including NMI, are disabled immediately after a reset. The first program instruction, located at the address specified at the top of the vector table, is therefore always executed. To prevent program crashes, this instruction should initialize the stack pointer (example: MOV.W #xx:16, SP). After execution of this instruction, the NMI interrupt is enabled. Other interrupts remain disabled until their enable bits are set to 1.

After reset exception handling, a CCR manipulation instruction can be executed to fix the CCR contents before the instruction that initializes the stack pointer. After the CCR manipulation instruction is executed, all interrupts, including NMI, are disabled. The next instruction should be the instruction that initializes the stack pointer.

4.3 Interrupts

4.3.1 Overview

There are twelve input pins for five external interrupt sources (NMI, IRQ_0 to IRQ_2, and IRQ_6). There are also 21 internal interrupts originating on-chip. The features of these interrupts are:

- All internal and external interrupts except NMI can be masked by the I bit in the CCR.

- IRQ_0 to IRQ_2 and IRQ_6 can be falling-edge-sensed or level-sensed. The type of sensing can be selected for each interrupt individually. NMI is edge-sensed, and either the rising or falling edge can be selected.

- Interrupts are individually vectored. The software interrupt-handling routine does not have to determine what type of interrupt has occurred.

- IRQ_6 is requested by eight external sources ($KEYIN_0$ to $KEYIN_7$). $KEYIN_0$ to $KEYIN_7$ can be masked individually by the user program.

- The watchdog timer can be made to generate an NMI interrupt or OVF interrupt according to its use. For details, see section 12, Watchdog Timer.

Table 4-2 lists all the interrupts in their order of priority and gives their vector numbers and the addresses of their entries in the vector table.

Table 4-2 Interrupts

Interrupt Source			No.	Address of Entry in Vector Table	Priority
NMI			3	H'0006 to H'0007	High
IRQ_0			4	H'0008 to H'0009	
IRQ_1			5	H'000A to H'000B	
IRQ_2			6	H'000C to H'000D	
Reserved			7 to 9	H'000E to H'0013	
IRQ_6	($KEYIN_0$ to $KEYIN_7$)		10	H'0014 to H'0015	
Reserved			11 to 16	H'0016 to H'0021	
Host interface	IBF1 (IDR1 reception complete)		17	H'0022 to H'0023	
	IBF2 (IDR2 reception complete)		18	H'0024 to H'0025	
16-bit free-running timer	ICI	(Input capture)	19	H'0026 to H'0027	
	OCIA	(Output compare A)	20	H'0028 to H'0029	
	OCIB	(Output compare B)	21	H'002A to H'002B	
	FOVI	(Overflow)	22	H'002C to H'002D	
8-bit timer 0	CMI0A	(Compare-match A)	23	H'002E to H'002F	
	CMI0B	(Compare-match B)	24	H'0030 to H'0031	
	OVI0	(Overflow)	25	H'0032 to H'0033	
8-bit timer 1	CMI1A	(Compare-match A)	26	H'0034 to H'0035	
	CMI1B	(Compare-match B)	27	H'0036 to H'0037	
	OVI1	(Overflow)	28	H'0038 to H'0039	
Serial communication interface 0	ERI0	(Receive error)	29	H'003A to H'003B	
	RXI0	(Receive end)	30	H'003C to H'003D	
	TXI0	(TDR empty)	31	H'003E to H'003F	
	TEI0	(TSR empty)	32	H'0040 to H'0041	
Serial communication interface 1	ERI1	(Receive error)	33	H'0042 to H'0043	
	RXI1	(Receive end)	34	H'0044 to H'0045	
	TXI1	(TDR empty)	35	H'0046 to H'0047	
	TEI1	(TSR empty)	36	H'0048 to H'0049	
Reserved			37 to 43	H'004A to H'0057	
Watchdog timer	WOVF	(WDT overflow)	44	H0058 to H'0059	Low
Reserved			45 to 49	H'005A to H'0063	

Notes: 1. H'0000 and H'0001 contain the reset vector.

2. H'0002 to H'0005 are reserved in the H8/3502 and are not available to the user.

4.3.2 Interrupt-Related Registers

The interrupt-related registers are the system control register (SYSCR), IRQ sense control register (ISCR), IRQ enable register (IER), and keyboard matrix interrupt mask register (KMIMR).

Table 4-3 Registers Read by Interrupt Controller

Name	Abbreviation	Read/Write	Address
System control register	SYSCR	R/W	H'FFC4
IRQ sense control register	ISCR	R/W	H'FFC6
IRQ enable register	IER	R/W	H'FFC7
Keyboard matrix interrupt mask register	KMIMR	R/W	H'FFF1

(1) System Control Register (SYSCR)—H'FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit 2—Nonmaskable Interrupt Edge (NMIEG): Determines whether a nonmaskable interrupt is generated on the falling or rising edge of the $\overline{\text{NMI}}$ input signal.

Bit 2 NMIEG	Description	
0	An interrupt is generated on the falling edge of $\overline{\text{NMI}}$	(Initial value)
1	An interrupt is generated on the rising edge of $\overline{\text{NMI}}$	

See section 3.2, System Control Register (SYSCR), for information on the other SYSCR bits.

(2) IRQ Sense Control Register (ISCR)—H'FFC6

Bit	7	6	5	4	3	2	1	0
	—	IRQ6SC	—	—	—	IRQ2SC	IRQ1SC	IRQ0SC
Initial value	1	0	1	1	1	0	0	0
Read/Write	—	R/W	—	—	—	R/W	R/W	R/W

Bits 0 to 2 and 6—IRQ_0 to IRQ_2, IRQ_6 Sense Control (IRQ0SC to IRQ2SC, IRQ6SC): These bits select how the input at pins $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$ and $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$ is sensed.

Bit i (i = 0 to 2, 6) IRQiSC	Description	
0	The low level of $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$ or $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$ generates an interrupt request	(Initial value)
1	The falling edge of $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$ or $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$ generates an interrupt request	

(3) IRQ Enable Register (IER)—H'FFC7

Bit	7	6	5	4	3	2	1	0
	—	IRQ6E	—	—	—	IRQ2E	IRQ1E	IRQ0E
Initial value	1	0	1	1	1	0	0	0
Read/Write	—	R/W	—	—	—	R/W	R/W	R/W

Bits 0 to 2, 6—IRQ_0 to IRQ_2 and IRQ_6 Enable (IRQ0E to IRQ2E, IRQ6E): These bits enable or disable the IRQ_0, IRQ_1, IRQ_2, and IRQ_6 interrupts individually.

Bit i (i = 0 to 2, 6) IRQiE	Description	
0	IRQ_0 to IRQ_2 and IRQ_6 are disabled	(Initial value)
1	IRQ_0 to IRQ_2 and IRQ_6 are enabled	

When edge sensing is selected (by setting bits IRQ0SC to IRQ2SC and IRQ6SC to 1), it is possible for an interrupt-handling routine to be executed even though the corresponding enable bit (IRQ0E to IRQ2E and IRQ6E) is cleared to 0 and the interrupt is disabled. If an interrupt is requested while the enable bit (IRQ0E to IRQ2E and IRQ6E) is set to 1, the request will be held pending until served. If the enable bit is cleared to 0 while the request is still pending, the request will remain pending, although new requests will not be recognized. If the interrupt mask bit (I) in the CCR is cleared to 0, the interrupt-handling routine can be executed even though the enable bit is now 0.

If execution of interrupt-handling routines under these conditions is not desired, it can be avoided by using the following procedure to disable and clear interrupt requests.

1. Set the I bit to 1 in the CCR, masking interrupts. Note that the I bit is set to 1 automatically when execution jumps to an interrupt vector.
2. Clear the desired bits from IRQ0E, IRQ1E, IRQ2E, and IRQ6E to 0 to disable new interrupt requests.
3. Clear the corresponding bits from IRQ0SC, IRQ1SC, IRQ2SC, and IRQ6SC to 0, then set them to 1 again. Pending IRQ_n interrupt requests are cleared when I = 1 in the CCR, IRQnSC = 0, and IRQnE = 0.

(4) Keyboard Matrix Interrupt Mask Register (KMIMR)

KMIMR is an 8-bit readable/writable register used in keyboard matrix scanning and sensing. To enable key-sense input interrupts from two or more pins during keyboard scanning and sensing, clear the corresponding mask bits to 0.

Bit	7	6	5	4	3	2	1	0
	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 7 to 0—Keyboard Matrix Interrupt Mask (KMIMR7 to KMIMR0): These bits control key-sense input interrupt requests $KEYIN_7$ to $KEYIN_0$.

Bits 7 to 0 KMIMR7 to KMIMR0	Description	
0	Key-sense input interrupt request is enabled.	
1	Key-sense input interrupt request is disabled.	(Initial value)

Figure 4-3 shows the relationship between the IRQ_6 interrupt and KMIMR.

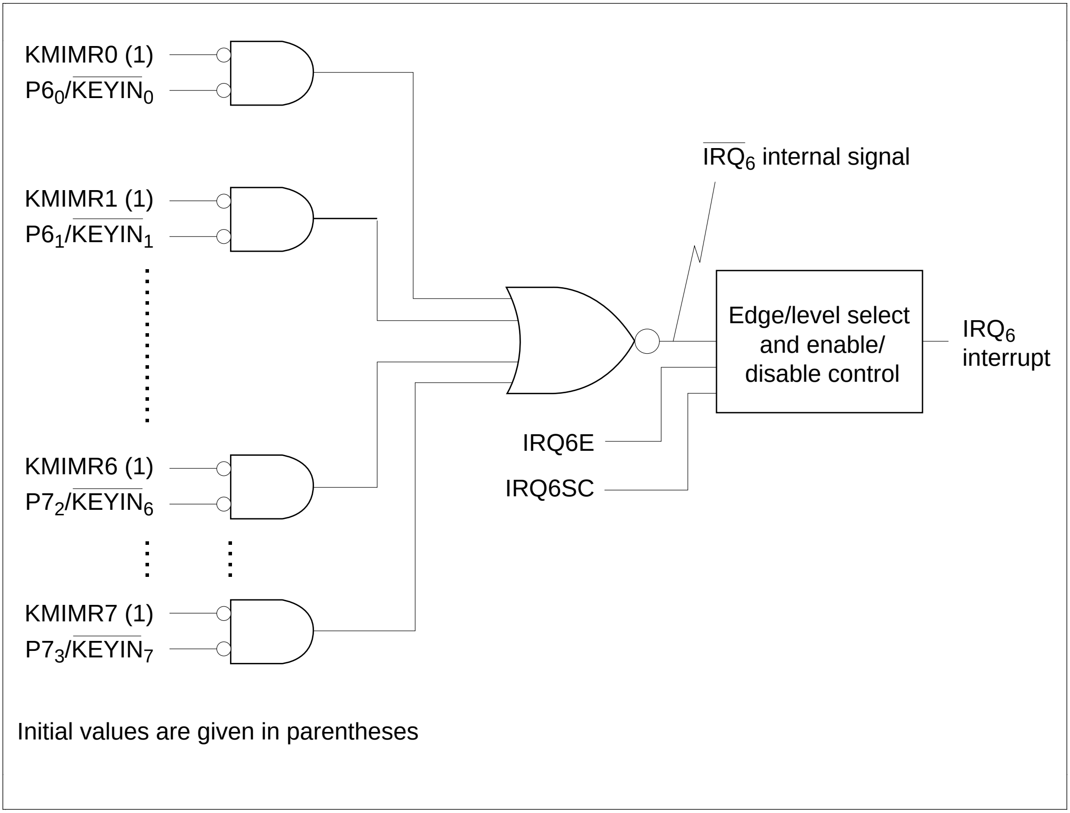

KMIMR0 (1)
P6$_0$/$\overline{\text{KEYIN}}_0$
KMIMR1 (1)
P6$_1$/$\overline{\text{KEYIN}}_1$
KMIMR6 (1)
P7$_2$/$\overline{\text{KEYIN}}_6$
KMIMR7 (1)
P7$_3$/$\overline{\text{KEYIN}}_7$
$\overline{\text{IRQ}}_6$ internal signal
Edge/level select and enable/ disable control
IRQ$_6$ interrupt
IRQ6E
IRQ6SC
Initial values are given in parentheses

Figure 4-3 KMIMR and IRQ$_6$ Interrupt

4.3.3 External Interrupts

There are five external interrupts: NMI, IRQ$_0$ to IRQ$_2$, and IRQ$_6$. These can be used to return from software standby mode.

(1) NMI: NMI is the highest-priority interrupt, and is always accepted regardless of the value of the I bit in CCR. Interrupts from the $\overline{\text{NMI}}$ pin are edge-sensed: rising edge or falling edge can be specified by the NMIEG bit in SYSCR. The NMI exception handling vector number is 3. NMI exception handling sets the I bit in CCR to 1.

(2) IRQ$_0$ to IRQ$_2$ and IRQ$_6$: Interrupts IRQ$_0$ to IRQ$_2$ are requested by input signals on pins $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$. The IRQ$_6$ interrupt is requested by input signals on pins $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$. Interrupts IRQ$_0$ to IRQ$_2$ and IRQ$_6$ can be specified as falling-edge-sensed or level-sensed by bits IRQ0SC to IRQ2SC and IRQ6SC in ISCR. Interrupt requests are enabled by set bits IRQ0E to IRQ2E and IRQ6E to 1 in IER. Interrupts are masked by setting the I bit to 1 in CCR.

The $\overline{\text{IRQ}}_6$ input signal is generated as the logical OR of the key-sense inputs. When pins $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$ (P6_0 to P6_3 and P7_0 to P7_3) are used as key-sense inputs, the corresponding KMIMR bits should be cleared to 0 to enable the corresponding key-sense interrupts. KMIMR bits corresponding to unused key-sense inputs should be set to 1 to disable those interrupts. All eight key-sense input interrupts are combined into a single IRQ_6 interrupt.

When one of these interrupts is accepted, the I bit is set to 1. IRQ_0 to IRQ_2 and IRQ_6 have interrupt vector numbers 4 to 6 and 10. They are prioritized in order from IRQ_6 (low) to IRQ_0 (high). For details, see table 4-2.

Interrupts IRQ_0 to IRQ_2 and IRQ_6 do not depend on whether pins $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$ and $\overline{\text{KEYIN}}_0$ to $\overline{\text{KEYIN}}_7$ are used as input pins or output pins. When interrupts IRQ_0 to IRQ_2 and IRQ_6 are requested by an external signal, clear the corresponding DDR bits to 0 and use the pins as input/output pins.

4.3.4 Internal Interrupts

Twenty-one internal interrupts can be requested by the on-chip supporting modules. All of them are masked when the I bit in the CCR is set. In addition, they can all be enabled or disabled by bits in the control registers of the on-chip supporting modules. When one of these interrupts is accepted, the I bit is set to 1 to mask further interrupts (except NMI).

The vector numbers of these interrupts are 17 to 36 and 44.

For the priority order of these interrupts, see table 4-2.

4.3.5 Interrupt Handling

Interrupts are controlled by an interrupt controller that arbitrates between simultaneous interrupt requests, commands the CPU to start the hardware interrupt exception-handling sequence, and furnishes the necessary vector number. Figure 4-4 shows a block diagram of the interrupt controller.

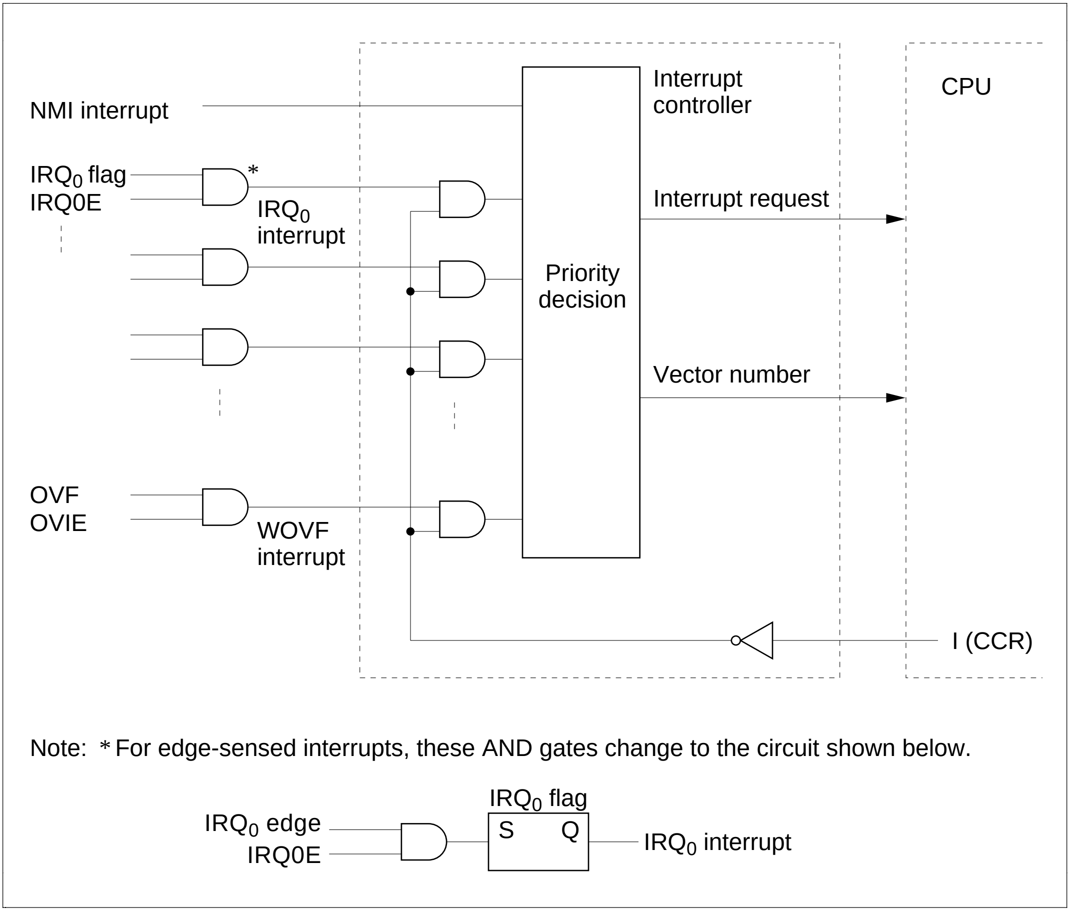


Figure 4-4 Block Diagram of Interrupt Controller

The IRQ interrupts and interrupts from the on-chip supporting modules (except for reset selected for a watchdog timer overflow) all have corresponding enable bits. When the enable bit is cleared to 0, the interrupt signal is not sent to the interrupt controller, so the interrupt is ignored. These interrupts can also all be masked by setting the CPU’s interrupt mask bit (I) to 1. Accordingly, these interrupts are accepted only when their enable bit is set to 1 and the I bit is cleared to 0.

The nonmaskable interrupt (NMI) is always accepted, except in the reset state and hardware standby mode.

When an NMI or another enabled interrupt is requested, the interrupt controller transfers the interrupt request to the CPU and indicates the corresponding vector number. (When two or more interrupts are requested, the interrupt controller selects the vector number of the interrupt with the highest priority.) When notified of an interrupt request, at the end of the current instruction or current hardware exception-handling sequence, the CPU starts the hardware exception-handling sequence for the interrupt and latches the vector number.

Figure 4-5 is a flowchart of the interrupt (and reset) operations. Figure 4-7 shows the interrupt timing sequence for the case in which the software interrupt-handling routine is in on-chip ROM and the stack is in on-chip RAM.

(1) An interrupt request is sent to the interrupt controller when an NMI interrupt occurs, and when an interrupt occurs on an IRQ input line or in an on-chip supporting module provided the enable bit of that interrupt is set to 1.

(2) The interrupt controller checks the I bit in CCR and accepts the interrupt request if the I bit is cleared to 0. If the I bit is set to 1 only NMI requests are accepted; other interrupt requests remain pending.

(3) Among all accepted interrupt requests, the interrupt controller selects the request with the highest priority and passes it to the CPU. Other interrupt requests remain pending.

(4) When it receives the interrupt request, the CPU waits until completion of the current instruction or hardware exception-handling sequence, then starts the hardware exception-handling sequence for the interrupt and latches the interrupt vector number.

(5) In the hardware exception-handling sequence, the CPU first pushes the PC and CCR onto the stack. See figure 4-6. The stacked PC indicates the address of the first instruction that will be executed on return from the software interrupt-handling routine.

(6) Next the I bit in CCR is set to 1, masking all further interrupts except NMI.

(7) The vector address corresponding to the vector number is generated, the vector table entry at this vector address is loaded into the program counter, and execution branches to the software interrupt-handling routine at the address indicated by that entry.

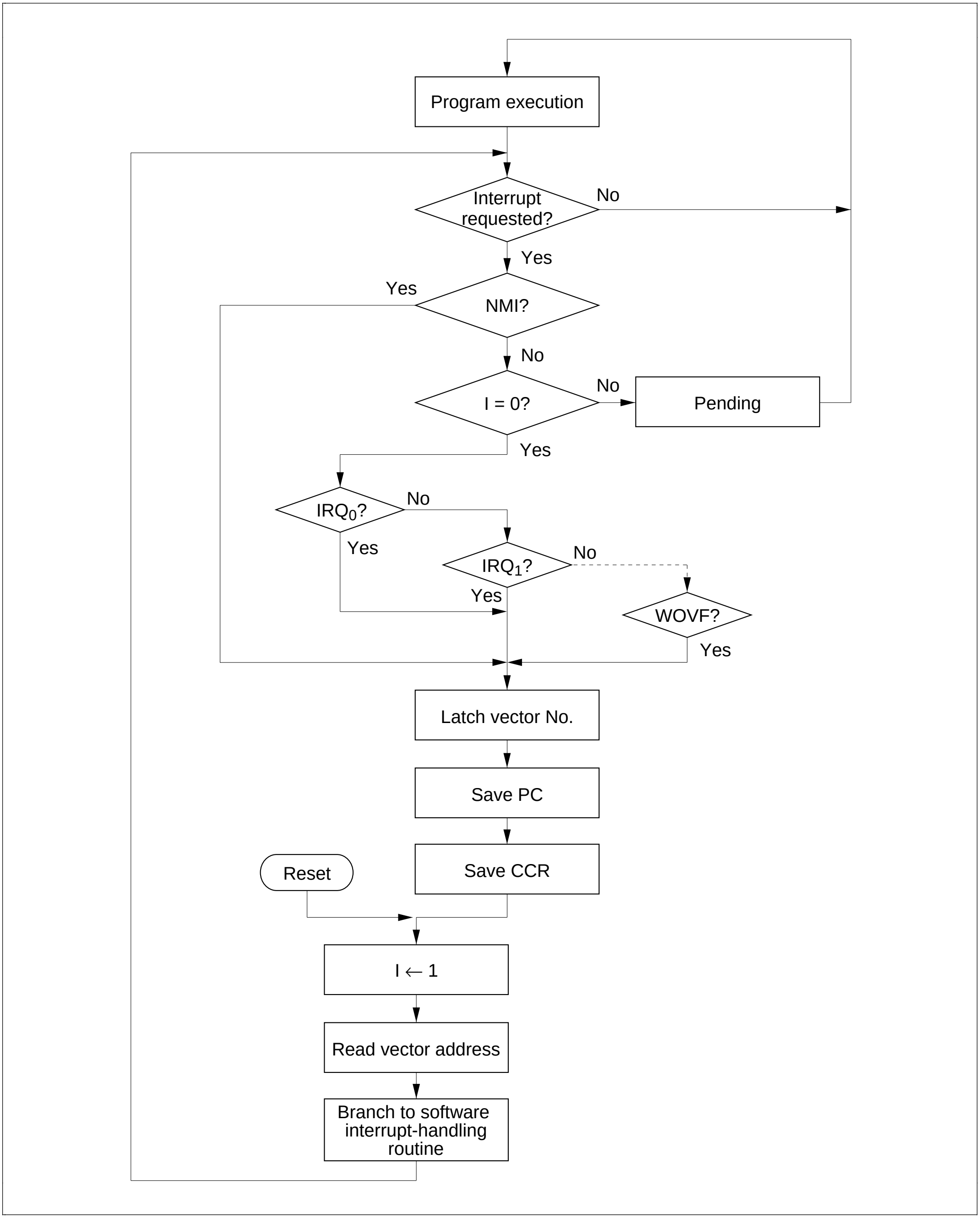
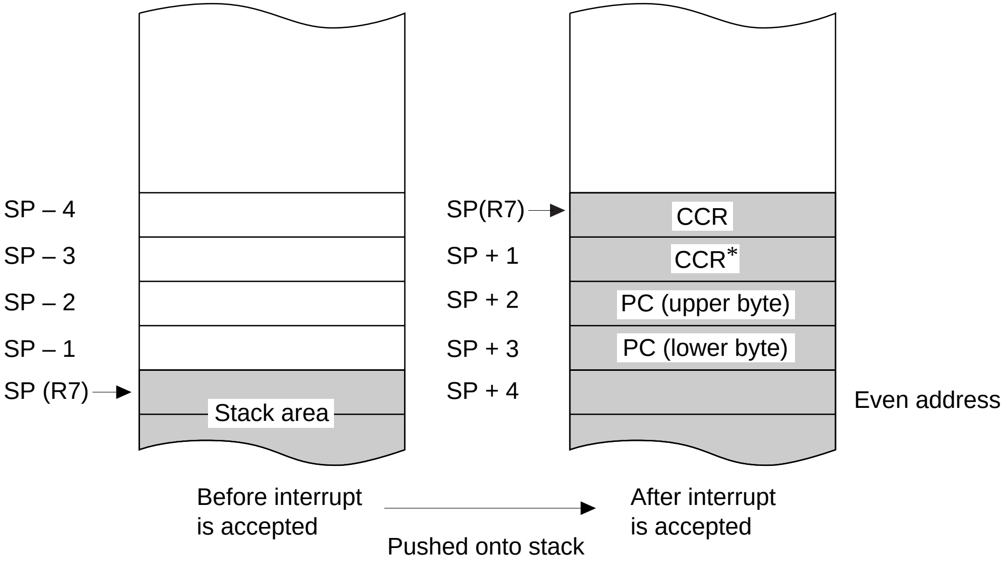


Figure 4-5 Hardware Interrupt-Handling Sequence



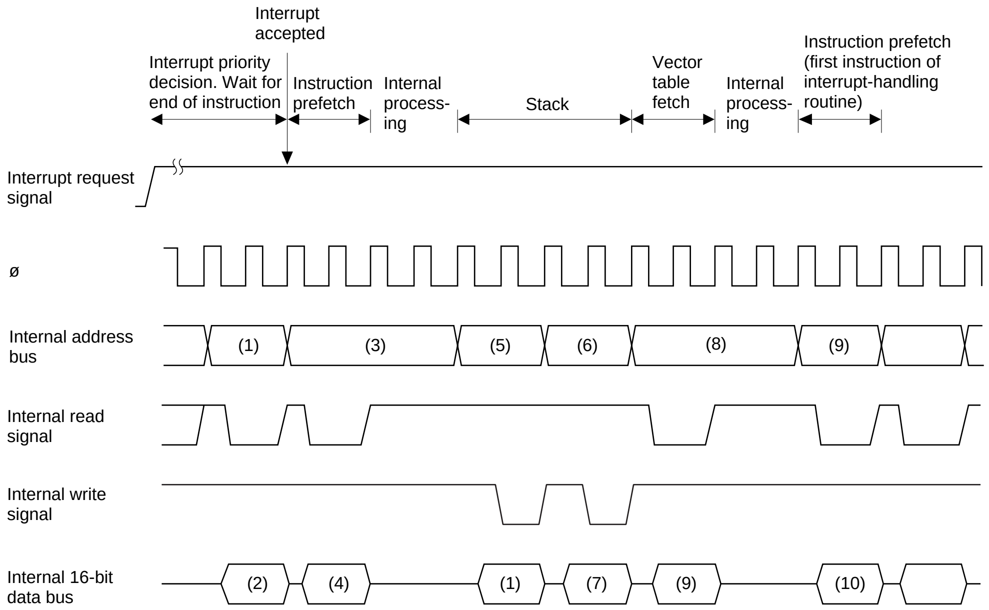
PC: Program counter
CCR: Condition code register
SP: Stack pointer

Notes: 1. The PC contains the address of the first instruction executed after return.
2. Registers must be saved and restored by word access at an even address.

* Ignored on return.

Figure 4-6 Usage of Stack in Interrupt Handling

Although the CCR consists of only one byte, it is treated as word data when pushed on the stack. In the hardware interrupt exception-handling sequence, two identical CCR bytes are pushed onto the stack to make a complete word. When popped from the stack by an RTE instruction, the CCR is loaded from the byte stored at the even address. The byte stored at the odd address is ignored.



(1)	Instruction prefetch address (Instruction is not executed. Address is saved as PC contents, becoming return address.)
(2) (4)	Instruction code (Not executed)
(3)	Instruction prefetch address (Not executed)
(5)	SP–2
(6)	SP–4
(7)	CCR
(8)	Vector address
(9)	Start address of interrupt-handling routine (contents of vector)
(10)	First instruction of interrupt-handling routine

Figure 4-7 Timing of Interrupt Sequence

4.3.6 Interrupt Response Time

Table 4-4 indicates the time that elapses from an interrupt request signal until the first instruction of the software interrupt-handling routine is executed. Since the H8/3502 accesses its on-chip memory 16 bits at a time, very fast interrupt service can be obtained by placing interrupt-handling routines in on-chip ROM and the stack in on-chip RAM.

Table 4-4 Number of States before Interrupt Service

		Number of States	
No.	**Reason for wait**	**On-Chip Memory**	**External Memory**
1	Interrupt priority decision	2*3	2*3
2	Wait for completion of current instruction*1	1 to 13	5 to 17*2
3	Save PC and CCR	4	12*2
4	Fetch vector	2	6*2
5	Fetch instruction	4	12*2
6	Internal processing	4	4
	Total	17 to 29	41 to 53*2

Notes: 1. These values do not apply if the current instruction is an EEPMOV instruction.
2. If wait states are inserted in external memory access, these values may be longer.
3. 1 for internal interrupts.

4.3.7 Precaution

Note that the following type of contention can occur in interrupt handling.

When software clears the enable bit of an interrupt to 0 to disable the interrupt, the interrupt becomes disabled after execution of the clearing instruction. If an enable bit is cleared by a BCLR or MOV instruction, for example, and the interrupt is requested during execution of that instruction, at the instant when the instruction ends the interrupt is still enabled, so after execution of the instruction, the hardware exception-handling sequence is executed for the interrupt. If a higher-priority interrupt is requested at the same time, however, the hardware exception-handling sequence is executed for the higher-priority interrupt and the interrupt that was disabled is ignored.

Similar considerations apply when an interrupt request flag is cleared to 0.

Figure 4-8 shows an example in which the OCIAE bit is cleared to 0.

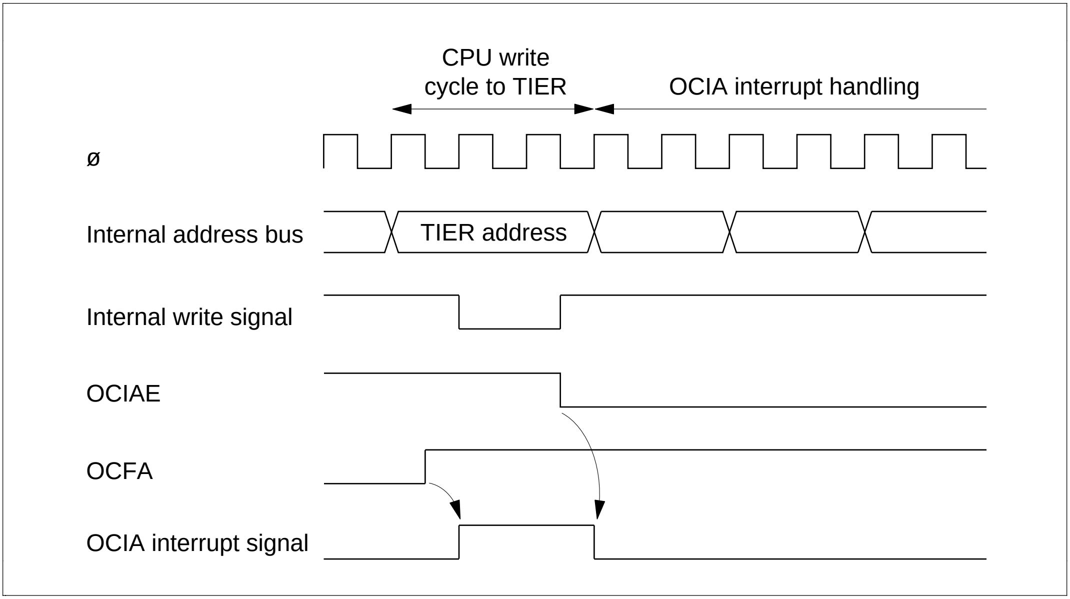


Figure 4-8 Contention between Interrupt and Disabling Instruction

The above contention does not occur if the enable bit or flag is cleared to 0 while the interrupt mask bit (I) is set to 1.

4.4 Note on Stack Handling

In word access, the least significant bit of the address is always assumed to be 0. The stack is always accessed by word access. Care should be taken to keep an even value in the stack pointer (general register R7). Use the PUSH and POP (or MOV.W Rn, @–SP and MOV.W @SP+, Rn) instructions to push and pop registers on the stack.

Setting the stack pointer to an odd value can cause programs to crash. Figure 4-9 shows an example of damage caused when the stack pointer contains an odd address.

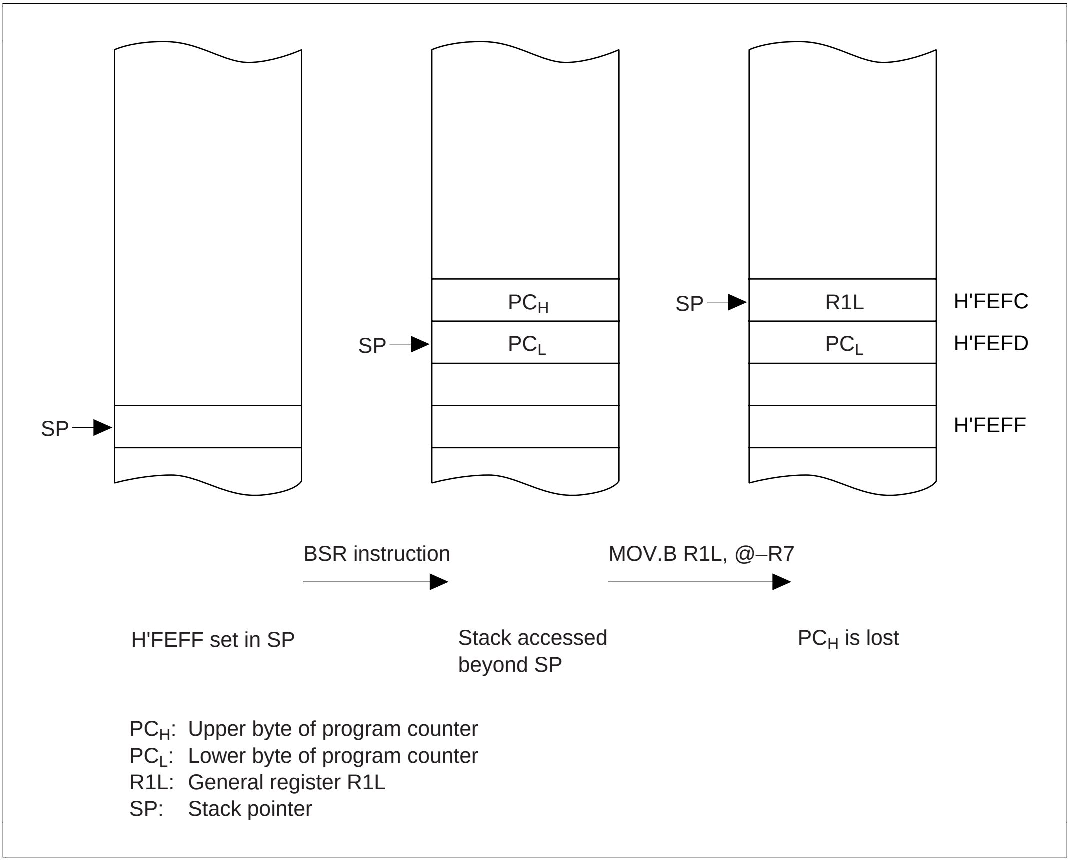


Figure 4-9 Example of Damage Caused by Setting an Odd Address in R7

4.5 Notes on the Use of Key-Sense Interrupts

The H8/3502 incorporates a key-sense interrupt function which can be used in any operating mode. When used in a mode other than slave mode (when the host interface is disabled), the following points must be noted.

In order to use the key-sense interrupt function, it is necessary to write to KMIMR to unmask the relevant $\overline{\text{KEYIN}}$ pins. If MOS pull-up transistors are provided on pins $P7_3$ to $P7_0$ and $P6_3$ to $P6_0$, KMPCR must also be written to.

KMIMR and KMPCR can only be accessed when the HIE bit in SYSCR is set to 1. Consequently, the chip is in slave mode during this period. In slave mode, pin states may vary.

(1) When KMIMR and KMPCR are set in the initialization routine directly after a reset

External circuitry must be used such that no problem will be caused irrespective of whether the host interface output and I/O pins retain the high-impedance state or are set to the output state. There are four host interface output pins—GA$_{20}$, HIRQ$_{12}$, HIRQ$_{1}$, and HIRQ$_{11}$—all of which are set to the port function (input state) initially. There are eight host interface I/O pins, HDB$_{7}$ to HDB$_{0}$; in single-chip mode, these are outputs when the P7$_{6}$/$\overline{\text{IOR}}$ pin is low and either one, or both, of the P7$_{5}$/$\overline{\text{CS}}_{1}$ and P4$_{5}$/$\overline{\text{CS}}_{2}$ pins is low. In expanded mode, these pins function as data bus pins (D$_{7}$ to D$_{0}$), and therefore the pin states do not vary.

(2) When KMIMR and KMPCR are set other than in the initialization routine

The states of the host interface input and I/O pins, and the pins with which they are multiplexed, may vary as a result of setting the HIE bit. P7$_{7}$/HA$_{0}$, P7$_{6}$/$\overline{\text{IOR}}$, P7$_{5}$/$\overline{\text{IOW}}$, P7$_{5}$/$\overline{\text{CS}}_{1}$, P4$_{6}$/$\overline{\text{CS}}_{2}$, and P3$_{7}$/HDB$_{7}$ to P3$_{0}$/HDB$_{0}$ automatically become input pins and I/O pins. When a particular pin is used, it is designated as a port input pin or expanded bus control pin, and in single-chip mode, it is necessary to prevent the occurrence of a low level of the P7$_{6}$/$\overline{\text{IOR}}$ pin together with a low level of the P7$_{5}$/$\overline{\text{CS}}_{1}$ pin or the P4$_{6}$/$\overline{\text{CS}}_{2}$ pin, or both.

In expanded mode, if external space is accessed when the HIE bit is set to 1, both the P7$_{6}$/$\overline{\text{IOR}}$/$\overline{\text{RD}}$ pin and the P7$_{5}$/$\overline{\text{CS}}_{1}$/$\overline{\text{AS}}$ pin are driven low automatically. Note that the output values of P4$_{4}$/HIRQ$_{12}$, P4$_{3}$/HIRQ$_{1}$, and P4$_{2}$/HIRQ$_{11}$ may vary as a result.

Section 5 Wait-State Controller

5.1 Overview

The H8/3502 has an on-chip wait-state controller that enables insertion of wait states into bus cycles for interfacing to low-speed external devices.

5.1.1 Features

Features of the wait-state controller are listed below.

- Three selectable wait modes: programmable wait mode, pin auto-wait mode, and pin wait mode
- Automatic insertion of zero to three wait states

5.1.2 Block Diagram

Figure 5-1 shows a block diagram of the wait-state controller.

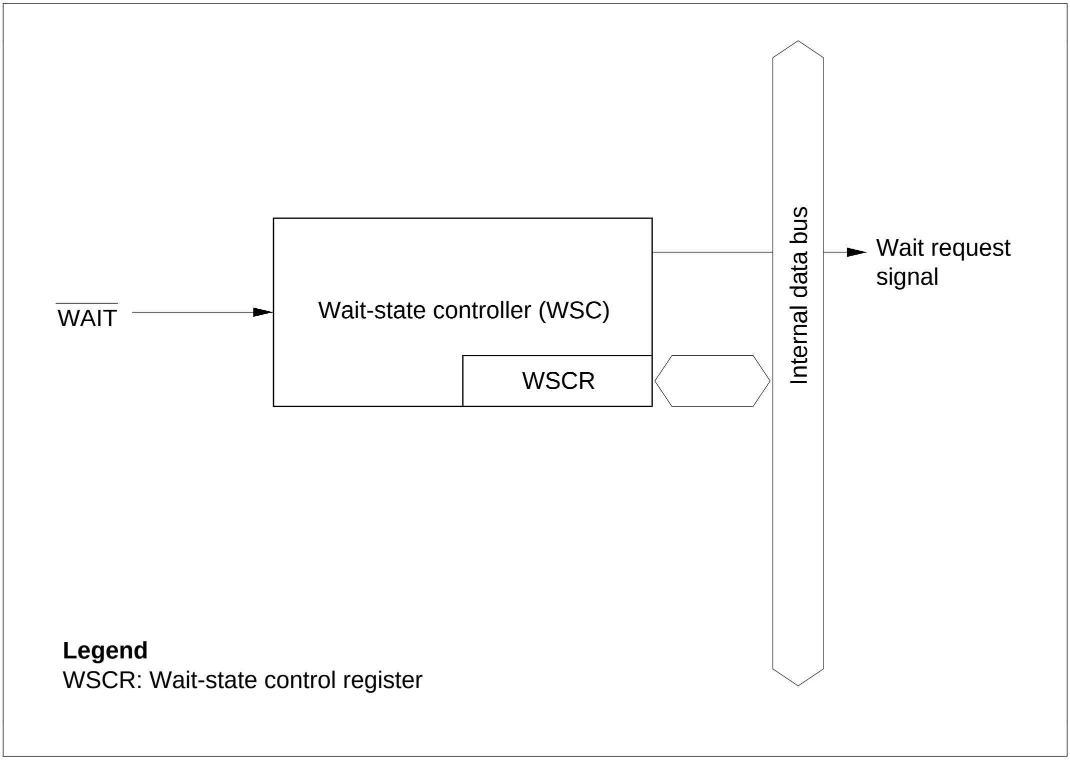


Figure 5-1 Block Diagram of Wait-State Controller

5.1.3 Input/Output Pins

Table 5-1 summarizes the wait-state controller’s input pin.

Table 5-1 Wait-State Controller Pins

Name	Abbreviation	I/O	Function
Wait	$\overline{\text{WAIT}}$	Input	Wait request signal for access to external addresses

5.1.4 Register Configuration

Table 5-2 summarizes the wait-state controller’s register.

Table 5-2 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address
Wait-state control register	WSCR	R/W	H'C8	H'FFC2

5.2 Register Description

5.2.1 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that selects the wait mode for the wait-state controller (WSC) and specifies the number of wait states. It also controls frequency division of the clock signals supplied to the supporting modules.

Bit	7	6	5	4	3	2	1	0
	—	—	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	1	1	0	0	1	0	0	0
Read/Write	—	—	R/W	R/W	R/W	R/W	R/W	R/W

WSCR is initialized to H'C8 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Bits 7 and 6—Reserved: These bits cannot be modified and are always read as 1.

Bit 5—Clock Double (CKDBL): Controls frequency division of clock signals supplied to supporting modules. For details, see section 6, Clock Pulse Generator.

Bit 4—Reserved: This bit is reserved, but it can be written and read. Its initial value is 0.

Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1 and WMS0): These bits select the wait mode.

Bit 3	Bit 2		
WMS1	**WMS0**	**Description**	
0	0	Programmable wait mode	
	1	No wait states inserted by wait-state controller	
1	0	Pin wait mode	(Initial value)
	1	Pin auto-wait mode	

Bits 1 and 0—Wait Count 1 and 0 (WC1 and WC0): These bits select the number of wait states inserted in access to external address areas.

Bit 1	Bit 0		
WC1	**WC0**	**Description**	
0	0	No wait states inserted by wait-state controller	(Initial value)
	1	1 state inserted	
1	0	2 states inserted	
	1	3 states inserted	

5.3 Wait Modes

Programmable Wait Mode: The number of wait states (T_W) selected by bits WC1 and WC0 are inserted in all accesses to external addresses. Figure 5-2 shows the timing when the wait count is 1 (WC1 = 0, WC0 = 1).

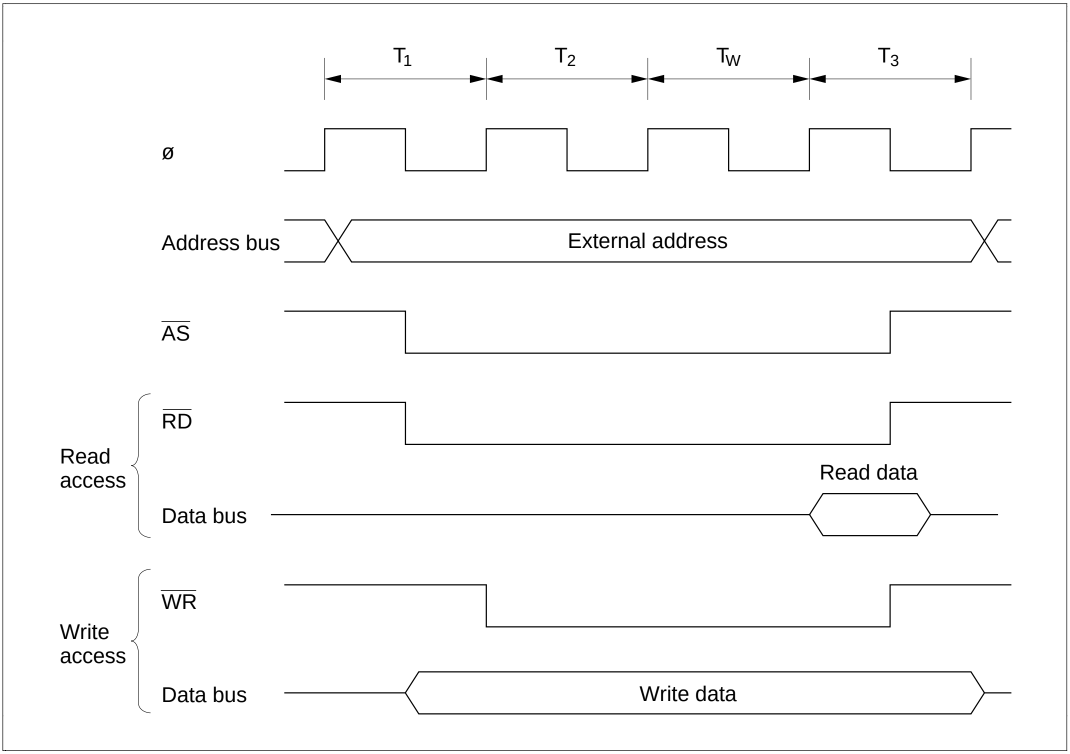
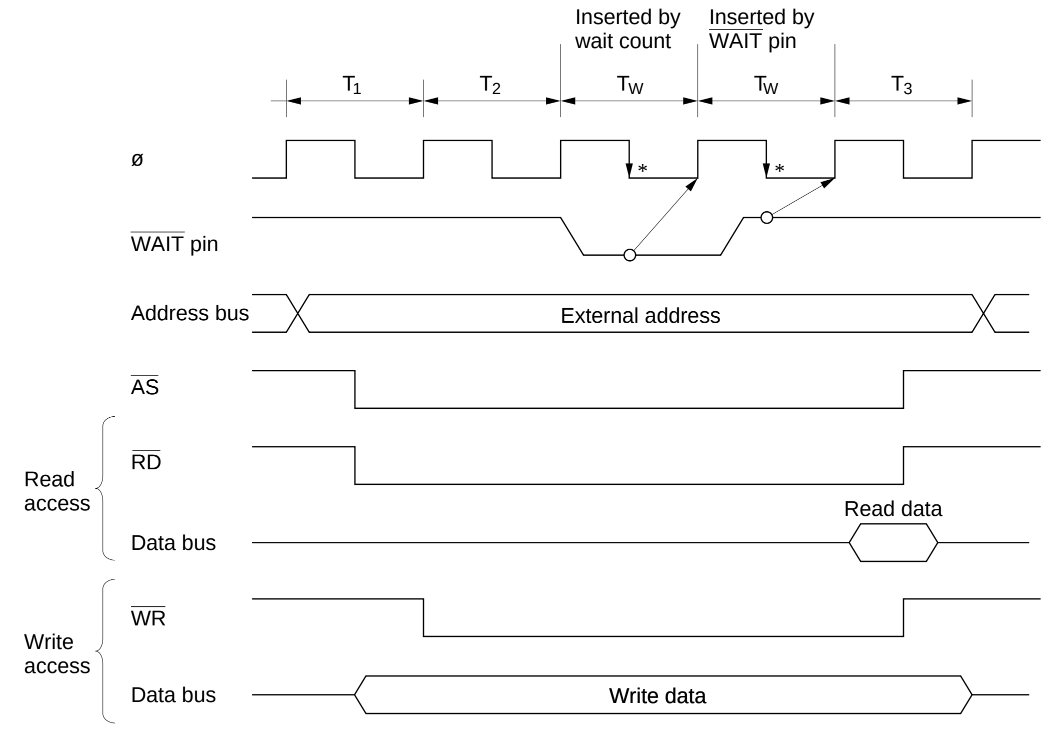


Figure 5-2 Programmable Wait Mode

Pin Wait Mode: In all accesses to external addresses, the number of wait states (T_W) selected by bits WC1 and WC0 are inserted. If the $\overline{\text{WAIT}}$ pin is low at the fall of the system clock (ø) in the last of these wait states, an additional wait state is inserted. If the $\overline{\text{WAIT}}$ pin remains low, wait states continue to be inserted until the $\overline{\text{WAIT}}$ signal goes high.

Pin wait mode is useful for inserting four or more wait states, or for inserting different numbers of wait states for different external devices.

Figure 5-3 shows the timing when the wait count is 1 (WC1 = 0, WC0 = 1) and one additional wait state is inserted by $\overline{\text{WAIT}}$ input.



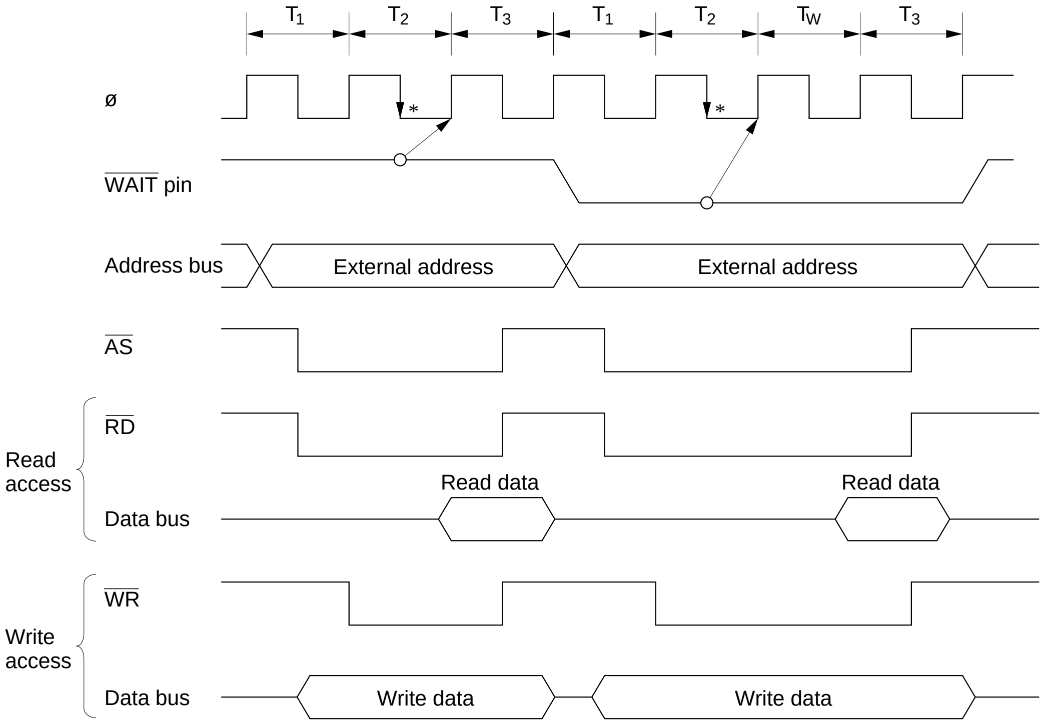
Note: * Arrows indicate time of sampling of the $\overline{\text{WAIT}}$ pin.

Figure 5-3 Pin Wait Mode

Pin Auto-Wait Mode: If the $\overline{\text{WAIT}}$ pin is low, the number of wait states (T_W) selected by bits WC1 and WC0 are inserted.

In pin auto-wait mode, if the $\overline{\text{WAIT}}$ pin is low at the fall of the system clock (ø) in the T_2 state, the number of wait states (T_W) selected by bits WC1 and WC0 are inserted. No additional wait states are inserted even if the $\overline{\text{WAIT}}$ pin remains low. Pin auto-wait mode can be used for an easy interface to low-speed memory, simply by routing the chip select signal to the $\overline{\text{WAIT}}$ pin.

Figure 5-4 shows the timing when the wait count is 1.



Note: * Arrows indicate time of sampling of the $\overline{\text{WAIT}}$ pin.

Figure 5-4 Pin Auto-Wait Mode

Section 6 Clock Pulse Generator

6.1 Overview

The H8/3502 has a built-in clock pulse generator (CPG) consisting of an oscillator circuit, a duty adjustment circuit, and a prescaler that generates clock signals for the on-chip supporting modules.

6.1.1 Block Diagram

Figure 6-1 shows a block diagram of the clock pulse generator.

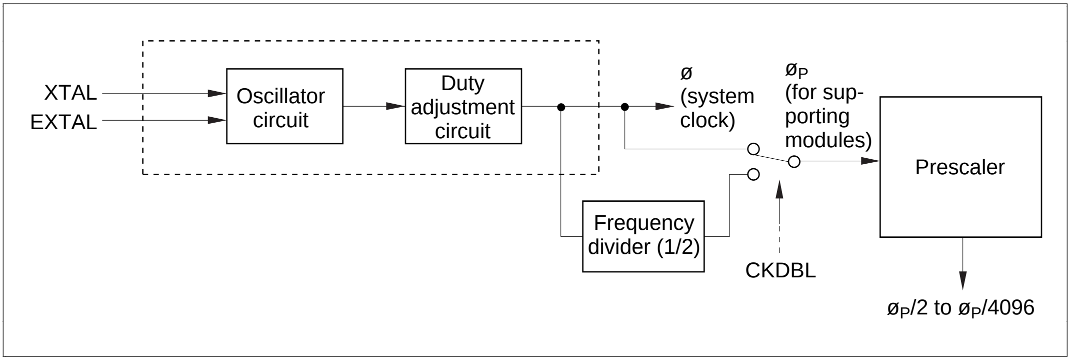


Figure 6-1 Block Diagram of Clock Pulse Generator

Input an external clock signal to the EXTAL pin, or connect a crystal resonator to the XTAL and EXTAL pins. The system clock frequency (ø) will be the same as the input frequency. This same system clock frequency ($ø_P$) can be supplied to timers and other supporting modules, or it can be divided by two. The selection is made by software, by controlling the CKDBL bit.

6.1.2 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that controls frequency division of the clock signals supplied to the supporting modules. It also controls wait-state insertion.

WSCR is initialized to H'C8 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Bit	7	6	5	4	3	2	1	0
	—	—	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	1	1	0	0	1	0	0	0
Read/Write	—	—	R/W	R/W	R/W	R/W	R/W	R/W

Bits 7 and 6—Reserved: These bits cannot be modified and are always read as 1.

Bit 5—Clock Double (CKDBL): Controls the frequency division of clock signals supplied to supporting modules.

CKDBL Bit 5	Description	
0	The undivided system clock (ø) is supplied as the clock ($ø_P$) for supporting modules	(Initial value)
1	The system clock (ø) is divided by two and supplied as the clock ($ø_P$) for supporting modules	

Bit 4—Reserved: This bit is reserved, but it can be written and read. Its initial value is 0.

Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1 and WMS0)

Bits 1 and 0—Wait Count 1 and 0 (WC1 and WC0)

These bits control wait-state insertion. For details, see section 5, Wait-State Controller.

6.2 Oscillator Circuit

If an external crystal is connected across the EXTAL and XTAL pins, the on-chip oscillator circuit generates a system clock signal. Alternatively, an external clock signal can be applied to the EXTAL pin.

(1) Connecting an External Crystal

Circuit Configuration: An external crystal can be connected as shown in the example in figure 6-2. Table 6-1 indicates the appropriate damping resistance Rd. An AT-cut parallel resonance crystal should be used.

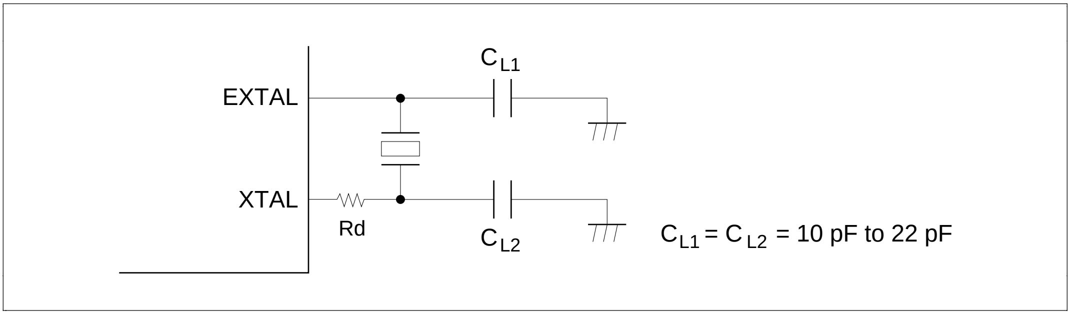


Figure 6-2 Connection of Crystal Oscillator (Example)

Table 6-1 Damping Resistance

Frequency (MHz)	4	8	10
Rd max (Ω)	500	200	0

Crystal Oscillator: Figure 6-3 shows an equivalent circuit of the crystal resonator. The crystal resonator should have the characteristics listed in table 6-2.

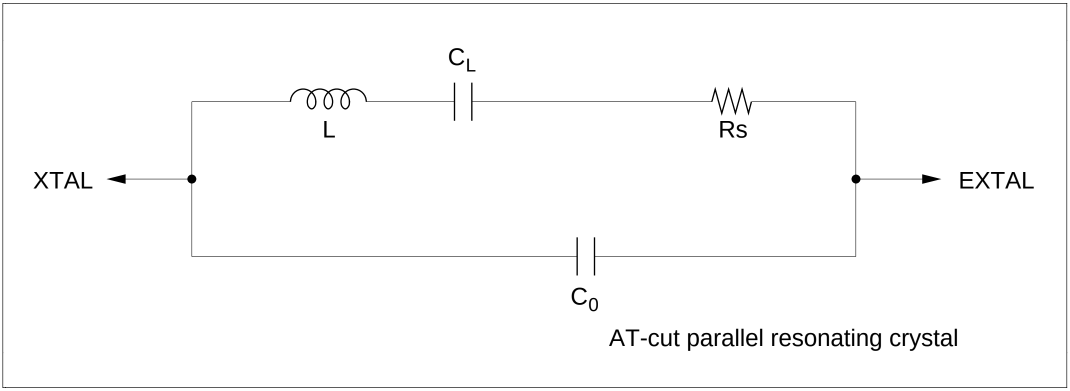


Figure 6-3 Equivalent Circuit of External Crystal

Table 6-2 External Crystal Parameters

Frequency (MHz)	4	8	10
Rs max (Ω)	120	80	70
C_0 (pF)	7 pF max		

Use a crystal with the same frequency as the desired system clock frequency (ø).

Note on Board Design: When an external crystal is connected, other signal lines should be kept away from the crystal circuit to prevent induction from interfering with correct oscillation. See figure 6-4. The crystal and its load capacitors should be placed as close as possible to the XTAL and EXTAL pins.

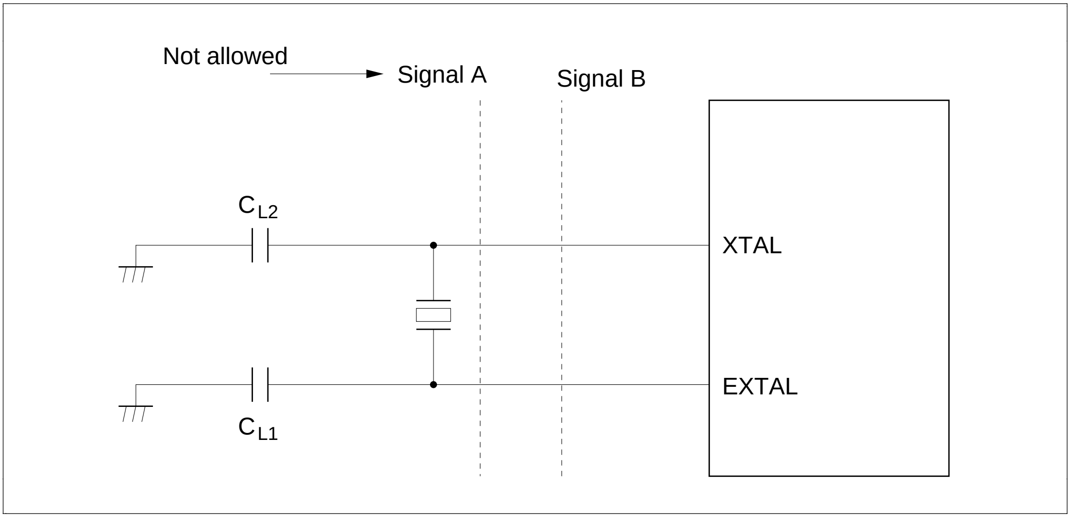
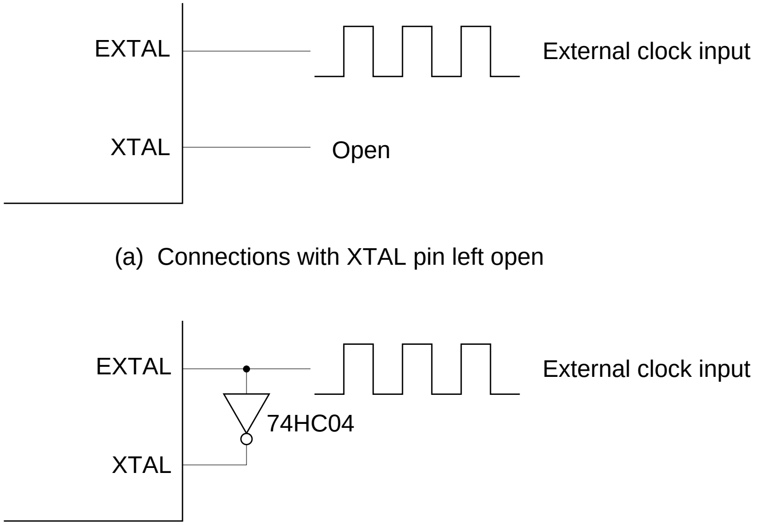


Figure 6-4 Notes on Board Design around External Crystal

(2) Input of External Clock Signal

Circuit Configuration: An external clock signal can be input as shown in the examples in figure 6-5. In example (b) in figure 6-5, the external clock signal should be kept high during standby.

If the XTAL pin is left open, make sure the stray capacitance does not exceed 10 pF.



(a) Connections with XTAL pin left open

(b) Connections with inverted clock input at XTAL pin

Figure 6-5 External Clock Input (Example)

External Clock Input: The external clock signal should have the same frequency as the desired system clock (ø). Clock timing parameters are given in table 6-3 and figure 6-6.

Table 6-3 Clock Timing

		V_{CC} = 5.0 V ±10%				
Item	**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**	
Low pulse width of external clock input	t_{EXL}	40	—	ns	Figure 6-6	
High pulse width of external clock input	t_{EXH}	40	—	ns		
External clock rise time	t_{EXr}	—	10	ns		
External clock fall time	t_{EXf}	—	10	ns		
Clock pulse width low	t_{CL}	0.3	0.7	t_{cyc}	ø ≥ 5 MHz	Figure 16-4
		0.4	0.6	t_{cyc}	ø < 5 MHz	
Clock pulse width high	t_{CH}	0.3	0.7	t_{cyc}	ø ≥ 5 MHz	
		0.4	0.6	t_{cyc}	ø < 5 MHz	

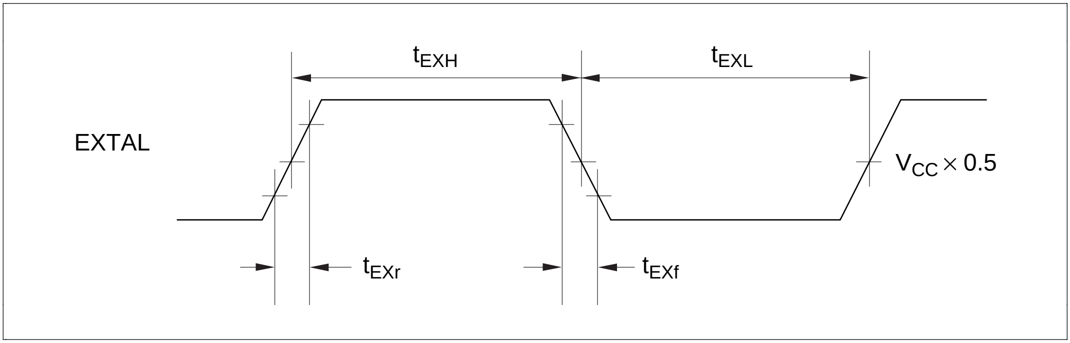


Figure 6-6 External Clock Input Timing

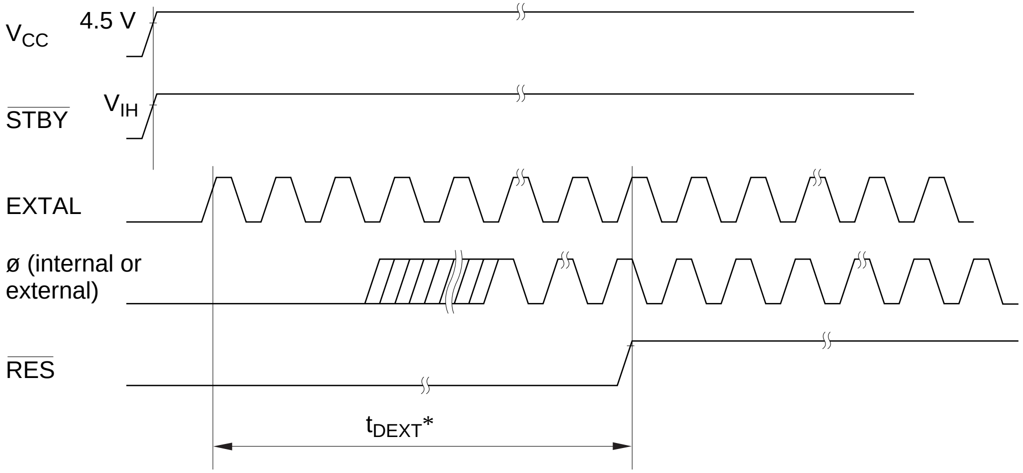
Table 6-4 shows the external clock output settling delay time, and figure 6-7 shows the external clock output settling delay timing. The oscillator circuit and duty adjustment circuit have a function for adjusting the waveform of the external clock input at the EXTAL pin. When the specified clock signal is input at the EXTAL pin, internal clock signal output is fixed after the elapse of the external clock output settling delay time (t_{DEXT}). As the clock signal output is not fixed during the t_{DEXT} period, the reset signal should be driven low to maintain the reset state during this time.

Table 6-4 External Clock Output Settling Delay Time

(Conditions: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V)

Item	Symbol	Min	Max	Unit	Notes
External clock output settling delay time	t_{DEXT}*	500	—	µs	Figure 6-7

Note: * t_{DEXT} includes an $\overline{RES}$ pulse width (t_{RESW}) of 10 t_{cyc}.



Note: * t_{DEXT} includes an $\overline{RES}$ pulse width (t_{RESW}) of 10 t_{cyc} .

Figure 6-7 External Clock Output Settling Delay Time Timing

6.3 Duty Adjustment Circuit

When the clock frequency is 5 MHz or above, the duty adjustment circuit adjusts the duty cycle of the signal from the oscillator circuit to generate the system clock (ø).

6.4 Prescaler

The 1/2 frequency divider generates an on-chip supporting module clock ($ø_P$) from the system clock (ø) according to the setting of the CKDBL bit. The prescaler divides the frequency of $ø_P$ to generate internal clock signals with frequencies from $ø_P/2$ to $ø_P/4096$.

Section 7 I/O Ports

7.1 Overview

The H8/3502 has five 8-bit input/output ports, one 7-bit input/output port, and one 6-bit input/output port.

Table 7-1 lists the functions of each port in each operating mode. As table 7-1 indicates, the port pins are multiplexed, and the pin functions differ depending on the operating mode.

Each port has a data direction register (DDR) that selects input or output, and a data register (DR) that stores output data. If bit manipulation instructions will be executed on the port data direction registers, see “Notes on Bit Manipulation Instructions” in section 2.5.5, Bit Manipulations.

Ports 1, 2, 3, 6, and 7 can drive one TTL load and a 90-pF capacitive load. Port 4 (excluding pin $P4_6$) and port 5 can drive one TTL load and a 30-pF capacitive load. Ports 1, 2, and 3 can drive LEDs (with 10-mA current sink). Ports 1 to 7 can drive a Darlington transistor. Ports 1 to 3 and pins $P6_0$ to $P6_3$ and $P7_0$ to $P7_3$ have built-in MOS pull-ups.

For block diagrams of the ports, see appendix C, I/O Port Block Diagrams.

Table 7-1 Port Functions

			Expanded Modes		Single-Chip Mode
Port	**Description**	**Pins**	**Mode 1**	**Mode 2**	**Mode 3**
Port 1	• 8-bit I/O port • Can drive LEDs • Built-in input pull-ups	P1$_7$ to P1$_0$/ A$_7$ to A$_0$	Lower address output (A$_7$ to A$_0$)	Lower address output (A$_7$ to A$_0$) or general input	General input/ output
Port 2	• 8-bit I/O port • Can drive LEDs • Built-in input pull-ups	P2$_7$ to P2$_0$/ A$_{15}$ to A$_8$	Upper address output (A$_{15}$ to A$_8$)	Upper address output (A$_{15}$ to A$_8$) or general input	General input/ output
Port 3	• 8-bit I/O port • Can drive LEDs • Built-in input pull-ups	P3$_7$ to P3$_0$/ D$_7$ to D$_0$/ HDB$_7$ to HDB$_0$	Data bus (D$_7$ to D$_0$)		Host interface data bus (HDB$_7$ to HDB$_0$) or general input/ output
Port 4	• 8-bit I/O port	P4$_7$/GA$_{20}$	Host interface control output (GA$_{20}$) or general input/ output		
		P4$_6$/ø/$\overline{CS}_2$	ø output		Host interface control input ($\overline{CS}_2$), general input, or ø output
		P4$_5$/TMRI$_1$/ HIRQ$_{12}$ P4$_4$/TMO$_1$/ HIRQ$_1$ P4$_3$/TMCI$_1$/ HIRQ$_{11}$ P4$_2$/TMRI$_0$ P4$_1$/TMO$_0$ P4$_0$/TMCI$_0$	Host interface host CPU interrupt request output (HIRQ$_{12}$, HIRQ$_1$, HIRQ$_{11}$), 8-bit timer 0 and 1 input/ output (TMCI$_0$, TMO$_0$, TMRI$_0$, TMCI$_1$, TMO$_1$, TMRI$_1$), and general input/output		

Table 7-1 Port Functions (cont)

			Expanded Modes		Single-Chip Mode
Port	**Description**	**Pins**	**Mode 1**	**Mode 2**	**Mode 3**
Port 5	• 6-bit I/O port	P5$_5$/SCK$_1$ P5$_4$/RxD$_1$ P5$_3$/TxD$_1$ P5$_2$/SCK$_0$ P5$_1$/RxD$_0$ P5$_0$/TxD$_0$	Serial communication interface 0 and 1 input/output (TxD$_0$, RxD$_0$, SCK$_0$, TxD$_1$, RxD$_1$, SCK$_1$) or 6-bit general input/output		
Port 6	• 7-bit I/O port • Built-in input pull-ups (P6$_3$ to P6$_0$)	P6$_6$/$\overline{\text{IRQ}}_2$ P6$_5$/$\overline{\text{IRQ}}_1$ P6$_4$/$\overline{\text{IRQ}}_0$	$\overline{\text{IRQ}}_2$ to $\overline{\text{IRQ}}_0$ or general input/output		
		P6$_3$/FTI/ $\overline{\text{KEYIN}}_3$ P6$_2$/FTOB/ $\overline{\text{KEYIN}}_2$ P6$_1$/FTOA/ $\overline{\text{KEYIN}}_1$ P6$_0$/FTCI/ $\overline{\text{KEYIN}}_0$	16-bit free-running timer input/output (FTCI, FTOA, FTOB, FTI) or general input/output (Can also be used as key-scanning key-sense input ($\overline{\text{KEYIN}}_3$ to $\overline{\text{KEYIN}}_0$))		
Port 7	• 8-bit I/O port • Bus buffer drive capability (P7$_3$ to P7$_0$) • Built-in input pull-ups (P7$_3$ to P7$_0$)	P7$_7$/$\overline{\text{WAIT}}$/ HA$_0$ P7$_6$/$\overline{\text{RD}}$/$\overline{\text{IOR}}$ P7$_5$/$\overline{\text{WR}}$/$\overline{\text{IOW}}$ P7$_4$/$\overline{\text{AS}}$/$\overline{\text{CS}}_1$	Expanded data bus control input/ output ($\overline{\text{WAIT}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{AS}}$)		Host interface control input (HA$_0$, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{CS}}_1$) or general input/output
		P7$_3$/$\overline{\text{KEYIN}}_7$ P7$_2$/$\overline{\text{KEYIN}}_6$ P7$_1$/$\overline{\text{KEYIN}}_5$ P7$_0$/$\overline{\text{KEYIN}}_4$	General input/output (Can also be used as key-scanning key-sense input ($\overline{\text{KEYIN}}_7$ to $\overline{\text{KEYIN}}_4$))		

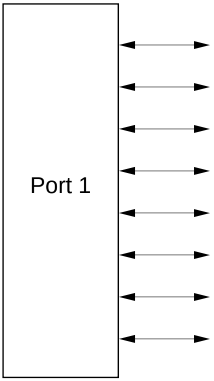
7.2 Port 1

7.2.1 Overview

Port 1 is an 8-bit input/output port with the pin configuration shown in figure 7-1. The pin functions differ depending on the operating mode.

Port 1 has built-in programmable MOS input pull-ups that can be used in modes 2 and 3.

Pins in port 1 can drive one TTL load and a 90-pF capacitive load. They can also drive LEDs and Darlington transistors.



Port 1 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled)	Pin configuration in mode 2 (expanded mode with on-chip ROM enabled)
$P1_7/A_7$	A_7 (output)	A_7 (output)/$P1_7$ (input)
$P1_6/A_6$	A_6 (output)	A_6 (output)/$P1_6$ (input)
$P1_5/A_5$	A_5 (output)	A_5 (output)/$P1_5$ (input)
$P1_4/A_4$	A_4 (output)	A_4 (output)/$P1_4$ (input)
$P1_3/A_3$	A_3 (output)	A_3 (output)/$P1_3$ (input)
$P1_2/A_2$	A_2 (output)	A_2 (output)/$P1_2$ (input)
$P1_1/A_1$	A_1 (output)	A_1 (output)/$P1_1$ (input)
$P1_0/A_0$	A_0 (output)	A_0 (output)/$P1_0$ (input)

Pin configuration in mode 3 (single-chip mode)

$P1_7$ (input/output)

$P1_6$ (input/output)

$P1_5$ (input/output)

$P1_4$ (input/output)

$P1_3$ (input/output)

$P1_2$ (input/output)

$P1_1$ (input/output)

$P1_0$ (input/output)

Figure 7-1 Port 1 Pin Configuration

7.2.2 Register Configuration and Descriptions

Table 7-2 summarizes the port 1 registers.

Table 7-2 Port 1 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 1 data direction register	P1DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB0
Port 1 data register	P1DR	R/W	H'00	H'FFB2
Port 1 input pull-up control register	P1PCR	R/W	H'00	H'FFAC

Port 1 Data Direction Register (P1DDR)

Bit	7	6	5	4	3	2	1	0
	P1$_7$DDR	P1$_6$DDR	P1$_5$DDR	P1$_4$DDR	P1$_3$DDR	P1$_2$DDR	P1$_1$DDR	P1$_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P1DDR controls the input/output direction of each pin in port 1.

Mode 1: The P1DDR values are fixed at 1. Port 1 consists of lower address output pins. P1DDR values cannot be modified and are always read as 1.

In hardware standby mode, the address bus is in the high-impedance state.

Mode 2: A pin in port 1 is used for address output if the corresponding P1DDR bit is set to 1, and for general input if this bit is cleared to 0.

Mode 3: A pin in port 1 is used for general output if the corresponding P1DDR bit is set to 1, and for general input if this bit is cleared to 0.

In modes 2 and 3, P1DDR is a write-only register. Read data is invalid. If read, all bits always read 1. P1DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P1DDR bit is set to 1, the corresponding pin remains in the output state.

Port 1 Data Register (P1DR)

Bit	7	6	5	4	3	2	1	0
	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P1DR is an 8-bit register that stores data for pins $P1_7$ to $P1_0$. When a P1DDR bit is set to 1, if port 1 is read, the value in P1DR is obtained directly, regardless of the actual pin state. When a P1DDR bit is cleared to 0, if port 1 is read the pin state is obtained.

P1DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

Port 1 Input Pull-Up Control Register (P1PCR)

Bit	7	6	5	4	3	2	1	0
	$P1_7$PCR	$P1_6$PCR	$P1_5$PCR	$P1_4$PCR	$P1_3$PCR	$P1_2$PCR	$P1_1$PCR	$P1_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P1PCR is an 8-bit readable/writable register that controls the MOS input pull-ups in port 1. If a P1DDR bit is cleared to 0 (designating input) and the corresponding P1PCR bit is set to 1, the MOS input pull-up is turned on.

P1PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

7.2.3 Pin Functions in Each Mode

Port 1 has different pin functions in different modes. A separate description for each mode is given below.

Pin Functions in Mode 1: In mode 1 (expanded mode with on-chip ROM disabled), port 1 is automatically used for lower address output (A_7 to A_0). Figure 7-2 shows the pin functions in mode 1.

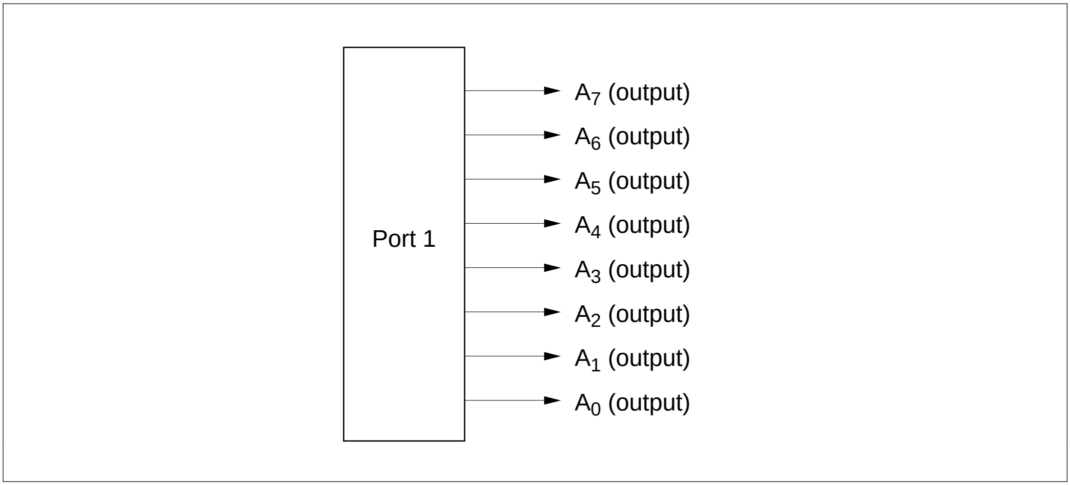


Figure 7-2 Pin Functions in Mode 1 (Port 1)

Mode 2: In mode 2 (expanded mode with on-chip ROM enabled), port 1 can provide lower address output pins, and general input pins. Each pin becomes a lower address output pin if its P1DDR bit is set to 1, and a general input pin if this bit is cleared to 0. Following a reset, all pins are input pins. To be used for address output or PWM output, their P1DDR bits must be set to 1. Figure 7-3 shows the pin functions in mode 2.

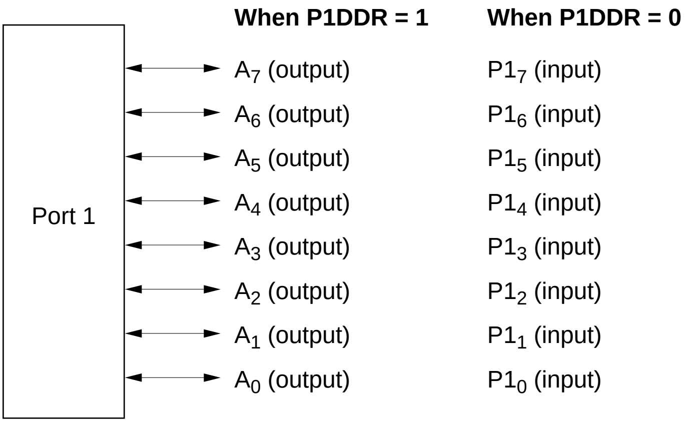


Figure 7-3 Pin Functions in Mode 2 (Port 1)

Mode 3: In mode 3 (single-chip mode), port 1 can provide general input/output pins. When used for general input/output, the input or output direction of each pin can be selected individually. A pin becomes a general input pin when its P1DDR bit is cleared to 0. When this bit is set to 1, the corresponding pin becomes a general output pin. Figure 7-4 shows the pin functions in mode 3.

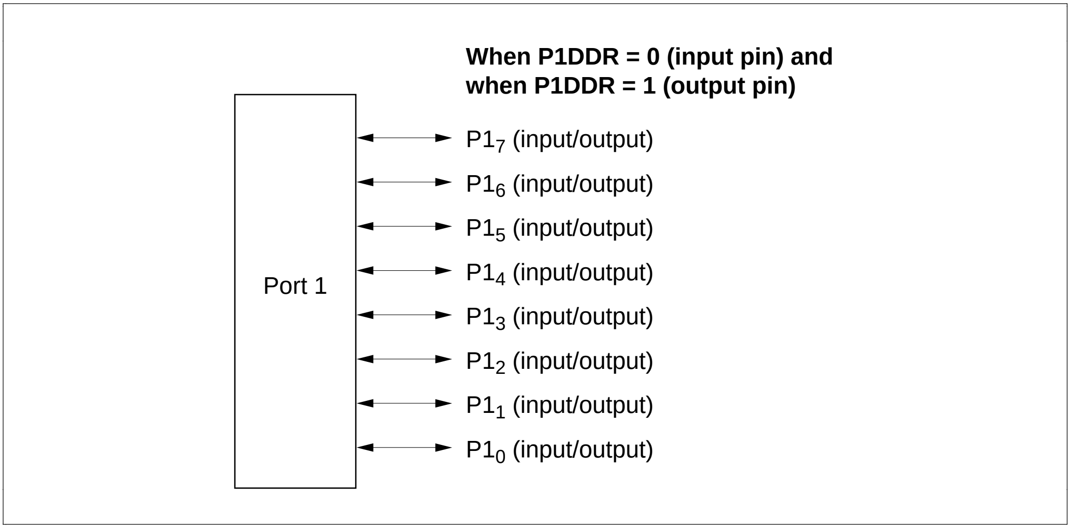

When P1DDR = 0 (input pin) and
when P1DDR = 1 (output pin)
Port 1
P1₇ (input/output)
P1₆ (input/output)
P1₅ (input/output)
P1₄ (input/output)
P1₃ (input/output)
P1₂ (input/output)
P1₁ (input/output)
P1₀ (input/output)

Figure 7-4 Pin Functions in Mode 3 (Port 1)

7.2.4 MOS Input Pull-Ups

Port 1 has built-in programmable MOS input pull-ups that are available in modes 2 and 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 2 or 3, set the corresponding P1PCR bit to 1 and clear the corresponding P1DDR bit to 0. P1PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-3 indicates the states of the MOS input pull-ups in each operating mode.

Table 7-3 States of MOS Input Pull-Ups (Port 1)

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The MOS input pull-up is always off.
On/off: The MOS input pull-up is on if P1PCR = 1 and P1DDR = 0, but off otherwise.

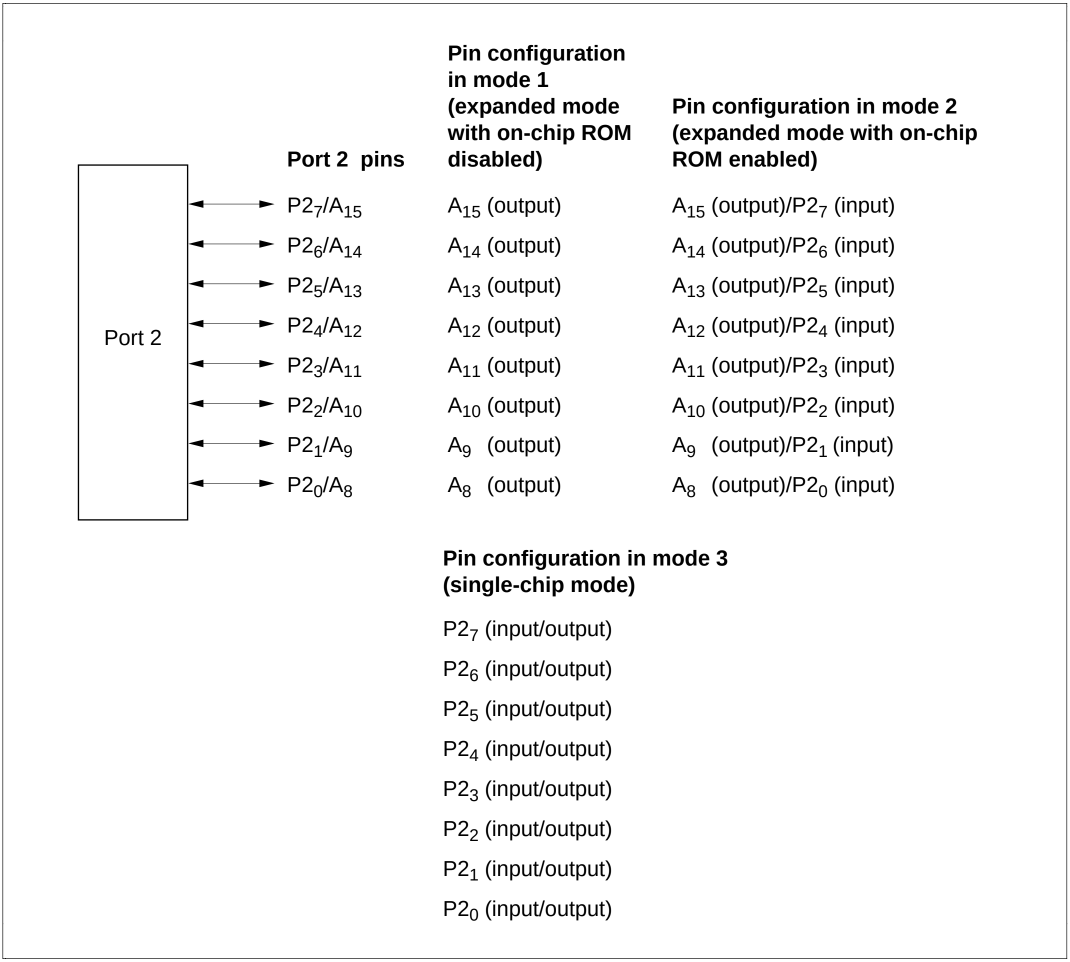
7.3 Port 2

7.3.1 Overview

Port 2 is an 8-bit input/output port with the pin configuration shown in figure 7-5. The pin functions differ depending on the operating mode.

Port 2 has built-in, software-controllable MOS input pull-ups that can be used in modes 2 and 3.

Pins in port 2 can drive one TTL load and a 90-pF capacitive load. They can also drive LEDs and Darlington transistors.



Port 2 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled)	Pin configuration in mode 2 (expanded mode with on-chip ROM enabled)
$P2_7/A_{15}$	A_{15} (output)	A_{15} (output)/$P2_7$ (input)
$P2_6/A_{14}$	A_{14} (output)	A_{14} (output)/$P2_6$ (input)
$P2_5/A_{13}$	A_{13} (output)	A_{13} (output)/$P2_5$ (input)
$P2_4/A_{12}$	A_{12} (output)	A_{12} (output)/$P2_4$ (input)
$P2_3/A_{11}$	A_{11} (output)	A_{11} (output)/$P2_3$ (input)
$P2_2/A_{10}$	A_{10} (output)	A_{10} (output)/$P2_2$ (input)
$P2_1/A_9$	A_9 (output)	A_9 (output)/$P2_1$ (input)
$P2_0/A_8$	A_8 (output)	A_8 (output)/$P2_0$ (input)

Pin configuration in mode 3 (single-chip mode)
$P2_7$ (input/output)
$P2_6$ (input/output)
$P2_5$ (input/output)
$P2_4$ (input/output)
$P2_3$ (input/output)
$P2_2$ (input/output)
$P2_1$ (input/output)
$P2_0$ (input/output)

Figure 7-5 Port 2 Pin Configuration

7.3.2 Register Configuration and Descriptions

Table 7-4 summarizes the port 2 registers.

Table 7-4 Port 2 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 2 data direction register	P2DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB1
Port 2 data register	P2DR	R/W	H'00	H'FFB3
Port 2 input pull-up control register	P2PCR	R/W	H'00	H'FFAD

Port 2 Data Direction Register (P2DDR)

Bit	7	6	5	4	3	2	1	0
	P2$_7$DDR	P2$_6$DDR	P2$_5$DDR	P2$_4$DDR	P2$_3$DDR	P2$_2$DDR	P2$_1$DDR	P2$_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P2DDR controls the input/output direction of each pin in port 2.

Mode 1: The P2DDR values are fixed at 1. Port 2 consists of upper address output pins. P2DDR values cannot be modified and are always read as 1.

In hardware standby mode, the address bus is in the high-impedance state.

Mode 2: A pin in port 2 is used for address output if the corresponding P2DDR bit is set to 1, and for general input if this bit is cleared to 0.

Mode 3: A pin in port 2 is used for general output if the corresponding P2DDR bit is set to 1, and for general input if this bit is cleared to 0.

In modes 2 and 3, P2DDR is a write-only register. Read data is invalid. If read, all bits always read 1. P2DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P2DDR bit is set to 1, the corresponding pin remains in the output state.

Port 2 Data Register (P2DR)

Bit	7	6	5	4	3	2	1	0
	$P2_7$	$P2_6$	$P2_5$	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2DR is an 8-bit register that stores data for pins $P2_7$ to $P2_0$. When a P2DDR bit is set to 1, if port 2 is read, the value in P2DR is obtained directly, regardless of the actual pin state. When a P2DDR bit is cleared to 0, if port 2 is read the pin state is obtained.

P2DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

Port 2 Input Pull-Up Control Register (P2PCR)

Bit	7	6	5	4	3	2	1	0
	$P2_7$PCR	$P2_6$PCR	$P2_5$PCR	$P2_4$PCR	$P2_3$PCR	$P2_2$PCR	$P2_1$PCR	$P2_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2PCR is an 8-bit readable/writable register that controls the MOS input pull-ups in port 2. If a P2DDR bit is cleared to 0 (designating input) and the corresponding P2PCR bit is set to 1, the MOS input pull-up is turned on.

P2PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

7.3.3 Pin Functions in Each Mode

Port 2 has different pin functions in different modes. A separate description for each mode is given below.

Pin Functions in Mode 1: In mode 1 (expanded mode with on-chip ROM disabled), port 2 is automatically used for upper address output (A_{15} to A_8). Figure 7-6 shows the pin functions in mode 1.

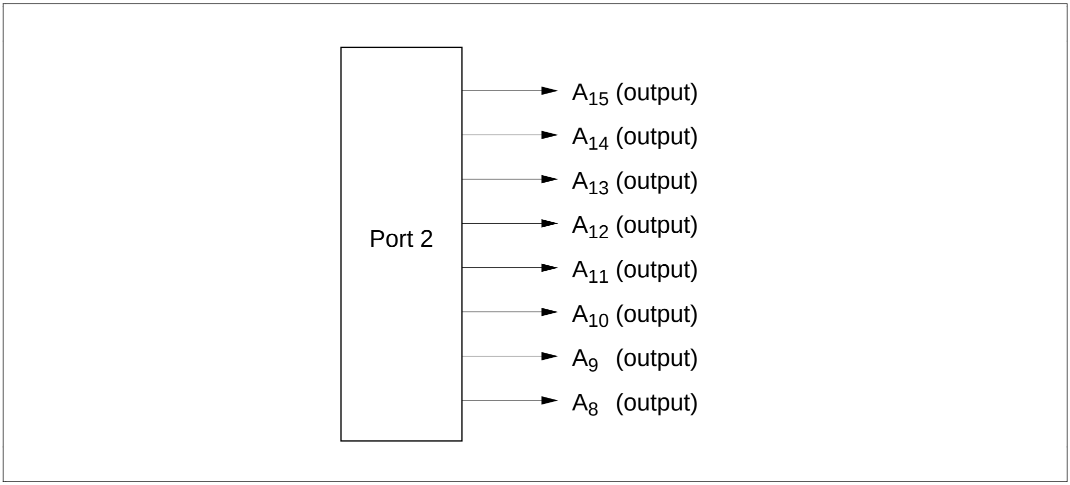


Figure 7-6 Pin Functions in Mode 1 (Port 2)

Mode 2: In mode 2 (expanded mode with on-chip ROM enabled), port 2 can provide upper address output pins, and general input pins. Each pin becomes an upper address output pin if its P2DDR bit is set to 1, and a general input pin if this bit is cleared to 0. Following a reset, all pins are input pins. To be used for address output, their P2DDR bits must be set to 1. Figure 7-7 shows the pin functions in mode 2.

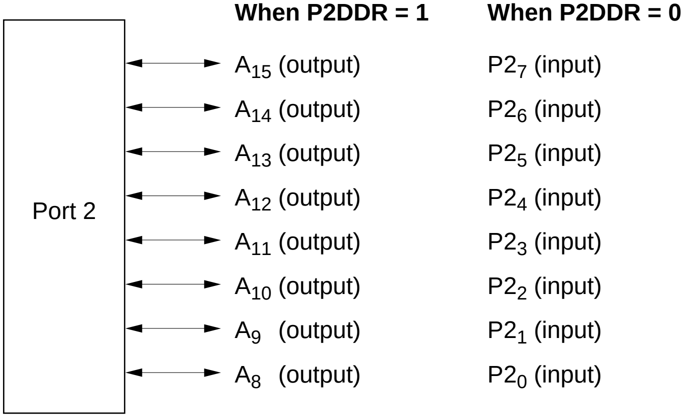


Figure 7-7 Pin Functions in Mode 2 (Port 2)

Mode 3: In mode 3 (single-chip mode) port 2 can provide general input/output pins. When used for general input/output, the input or output direction of each pin can be selected individually. A pin becomes a general input pin when its P2DDR bit is cleared to 0. When this bit is set to 1, the corresponding pin becomes a general output pin. Figure 7-8 shows the pin functions in mode 3.

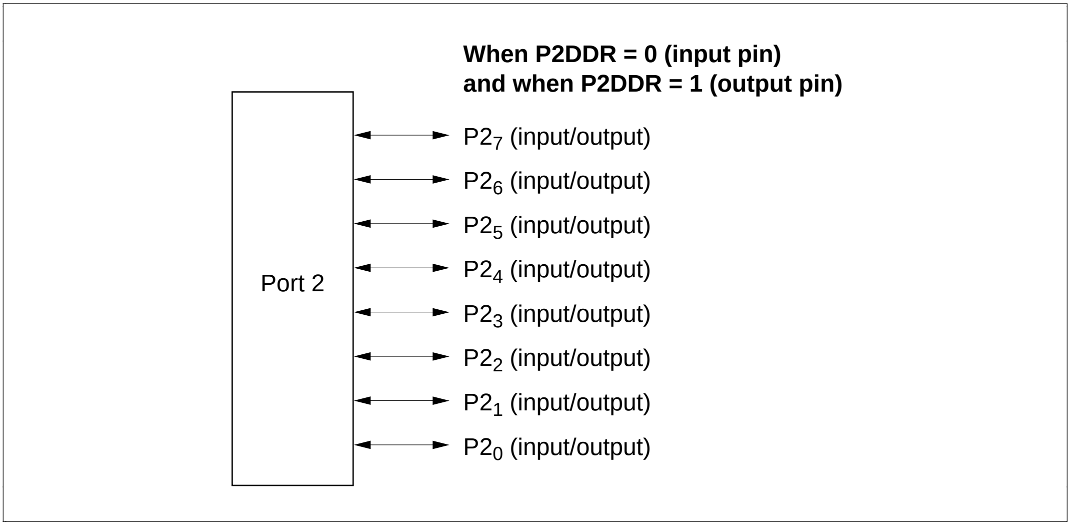


Figure 7-8 Pin Functions in Mode 3 (Port 2)

7.3.4 MOS Input Pull-Ups

Port 2 has built-in programmable MOS input pull-ups that are available in modes 2 and 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 2 or 3, set the corresponding P2PCR bit to 1 and clear the corresponding P2DDR bit to 0. P2PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-5 indicates the states of the input pull-up transistors in each operating mode.

Table 7-5 States of MOS Input Pull-Ups (Port 2)

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The MOS input pull-up is always off.

On/off: The MOS input pull-up is on if P2PCR = 1 and P2DDR = 0, but off otherwise.

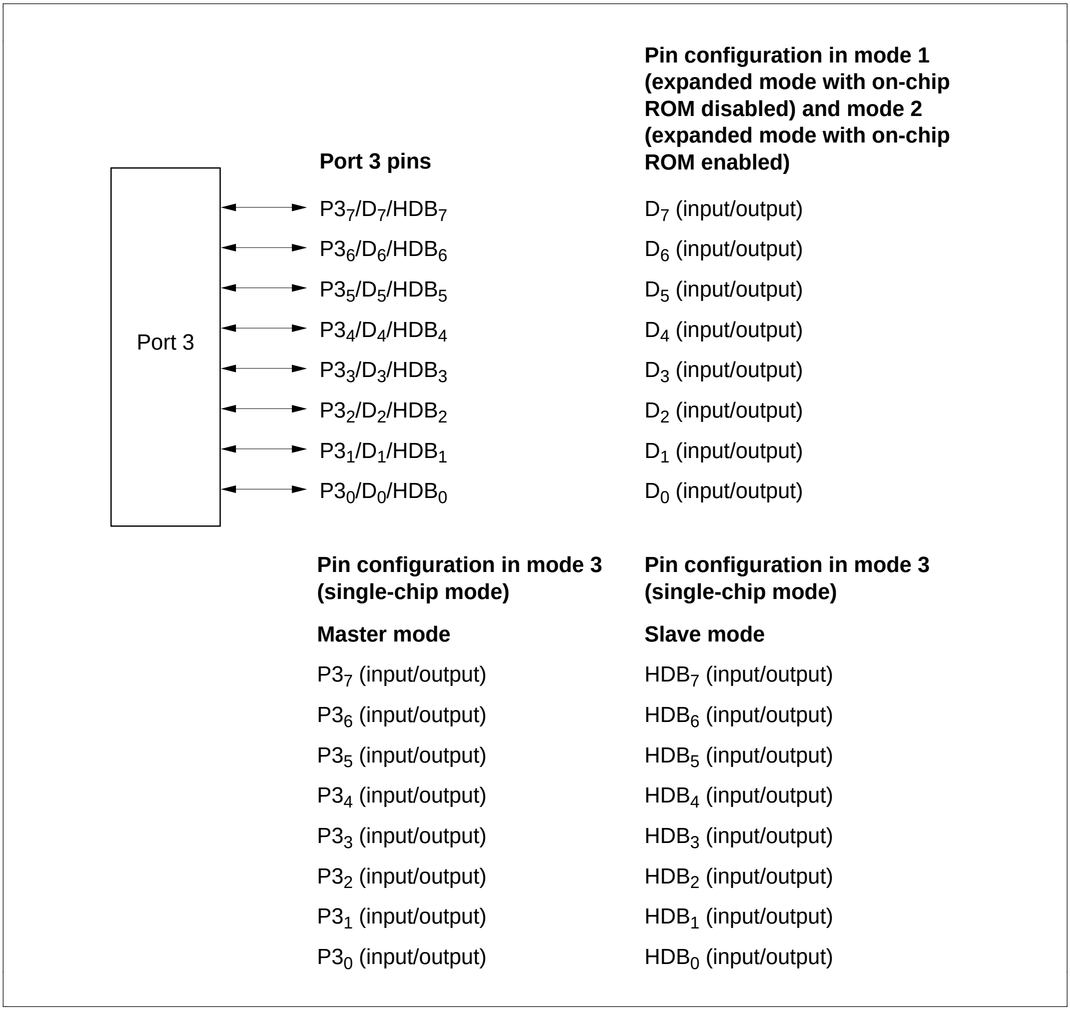
7.4 Port 3

7.4.1 Overview

Port 3 is an 8-bit input/output port that is multiplexed with the data bus and host interface data bus. Its pin configuration is shown in figure 7-9. The pin functions differ depending on the operating mode.

Port 3 has built-in programmable MOS input pull-ups that can be used in mode 3.

Pins in port 3 can drive one TTL load and a 90-pF capacitive load. They can also drive a LED and a Darlington transistor.



Port 3 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled)
$P3_7/D_7/HDB_7$	D_7 (input/output)
$P3_6/D_6/HDB_6$	D_6 (input/output)
$P3_5/D_5/HDB_5$	D_5 (input/output)
$P3_4/D_4/HDB_4$	D_4 (input/output)
$P3_3/D_3/HDB_3$	D_3 (input/output)
$P3_2/D_2/HDB_2$	D_2 (input/output)
$P3_1/D_1/HDB_1$	D_1 (input/output)
$P3_0/D_0/HDB_0$	D_0 (input/output)

Pin configuration in mode 3 (single-chip mode)	Pin configuration in mode 3 (single-chip mode)
Master mode	Slave mode
$P3_7$ (input/output)	HDB_7 (input/output)
$P3_6$ (input/output)	HDB_6 (input/output)
$P3_5$ (input/output)	HDB_5 (input/output)
$P3_4$ (input/output)	HDB_4 (input/output)
$P3_3$ (input/output)	HDB_3 (input/output)
$P3_2$ (input/output)	HDB_2 (input/output)
$P3_1$ (input/output)	HDB_1 (input/output)
$P3_0$ (input/output)	HDB_0 (input/output)

Figure 7-9 Port 3 Pin Configuration

7.4.2 Register Configuration and Descriptions

Table 7-6 summarizes the port 3 registers.

Table 7-6 Port 3 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 3 data direction register	P3DDR	W	H'00	H'FFB4
Port 3 data register	P3DR	R/W	H'00	H'FFB6
Port 3 input pull-up control register	P3PCR	R/W	H'00	H'FFAE

Port 3 Data Direction Register (P3DDR)

Bit	7	6	5	4	3	2	1	0
	$P3_7DDR$	$P3_6DDR$	$P3_5DDR$	$P3_4DDR$	$P3_3DDR$	$P3_2DDR$	$P3_1DDR$	$P3_0DDR$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P3DDR is an 8-bit readable/writable register that controls the input/output direction of each pin in port 3. P3DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

Modes 1 and 2: In mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled), the input/output directions designated by P3DDR are ignored. Port 3 automatically consists of the input/output pins of the 8-bit data bus (D_7 to D_0).

The data bus is in the high-impedance state during reset, and during hardware and software standby.

Mode 3: A pin in port 3 is used for general output if the corresponding P3DDR bit is set to 1, and for general input if this bit is cleared to 0. P3DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P3DDR bit is set to 1, the corresponding pin remains in the output state.

Port 3 Data Register (P3DR)

Bit	7	6	5	4	3	2	1	0
	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3DR is an 8-bit register that stores data for pins $P3_7$ to $P3_0$. When a P3DDR bit is set to 1, if port 3 is read, the value in P3DR is obtained directly, regardless of the actual pin state. When a P3DDR bit is cleared to 0, if port 3 is read the pin state is obtained.

P3DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

Port 3 Input Pull-Up Control Register (P3PCR)

Bit	7	6	5	4	3	2	1	0
	$P3_7$PCR	$P3_6$PCR	$P3_5$PCR	$P3_4$PCR	$P3_3$PCR	$P3_2$PCR	$P3_1$PCR	$P3_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3PCR is an 8-bit readable/writable register that controls the MOS input pull-ups in port 3. If a P3DDR bit is cleared to 0 (designating input) and the corresponding P3PCR bit is set to 1, the MOS input pull-up is turned on.

P3PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

The MOS input pull-ups cannot be used in slave mode (when the host interface is enabled).

7.4.3 Pin Functions in Each Mode

Port 3 has different pin functions in different modes. A separate description for each mode is given below.

Pin Functions in Modes 1 and 2: In mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled), port 3 is automatically used for the input/output pins of the data bus (D_7 to D_0). Figure 7-10 shows the pin functions in modes 1 and 2.

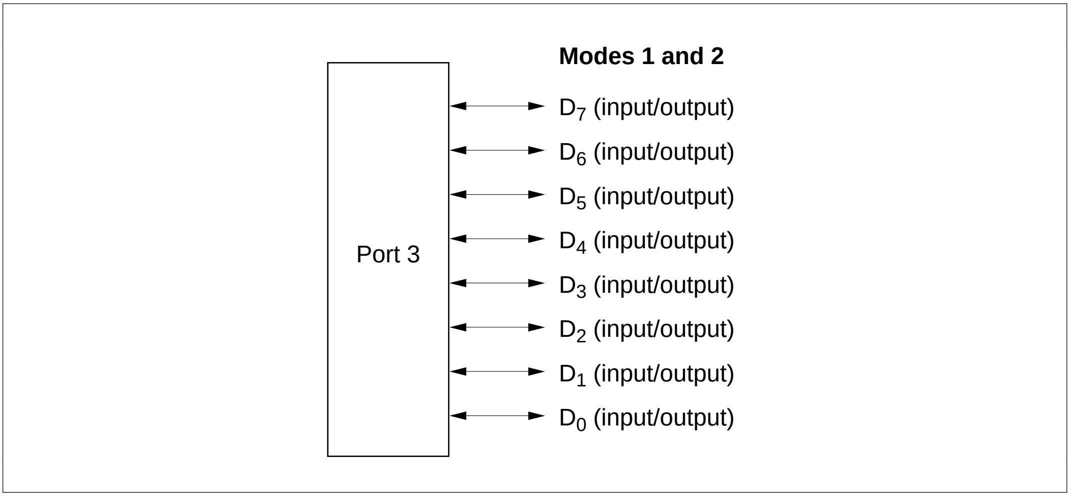


Figure 7-10 Pin Functions in Modes 1 and 2 (Port 3)

Mode 3: In mode 3 (single-chip mode), port 3 is an input/output port when the host interface enable bit (HIE) in the system control register (SYSCR) is cleared to 0.

If the HIE bit is set to 1 and a transition is made to slave mode, port 3 becomes the host interface data bus (HDB_7 to HDB_0). In slave mode, P3DR and P3DDR should be cleared to H'00.

Figure 7-11 shows the pin functions in mode 3.

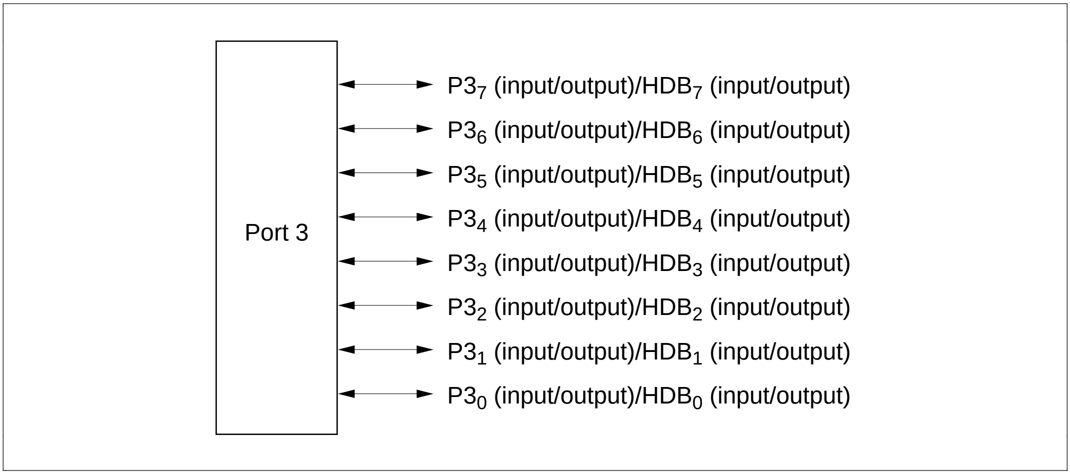

Port 3
$P3_7$ (input/output)/HDB_7 (input/output)
$P3_6$ (input/output)/HDB_6 (input/output)
$P3_5$ (input/output)/HDB_5 (input/output)
$P3_4$ (input/output)/HDB_4 (input/output)
$P3_3$ (input/output)/HDB_3 (input/output)
$P3_2$ (input/output)/HDB_2 (input/output)
$P3_1$ (input/output)/HDB_1 (input/output)
$P3_0$ (input/output)/HDB_0 (input/output)

Figure 7-11 Pin Functions in Mode 3 (Port 3)

7.4.4 Input Pull-Up Transistors

Port 3 has built-in programmable MOS input pull-ups that are available in mode 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 3, set the corresponding P3PCR bit to 1 and clear the corresponding P3DDR bit to 0. P3PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-7 indicates the states of the input MOS pull-ups in each operating mode.

Table 7-7 States of MOS Input Pull-Ups (Port 3)

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The MOS input pull-up is always off.
On/off: The MOS input pull-up is on if P3PCR = 1 and P3DDR = 0, but off otherwise.

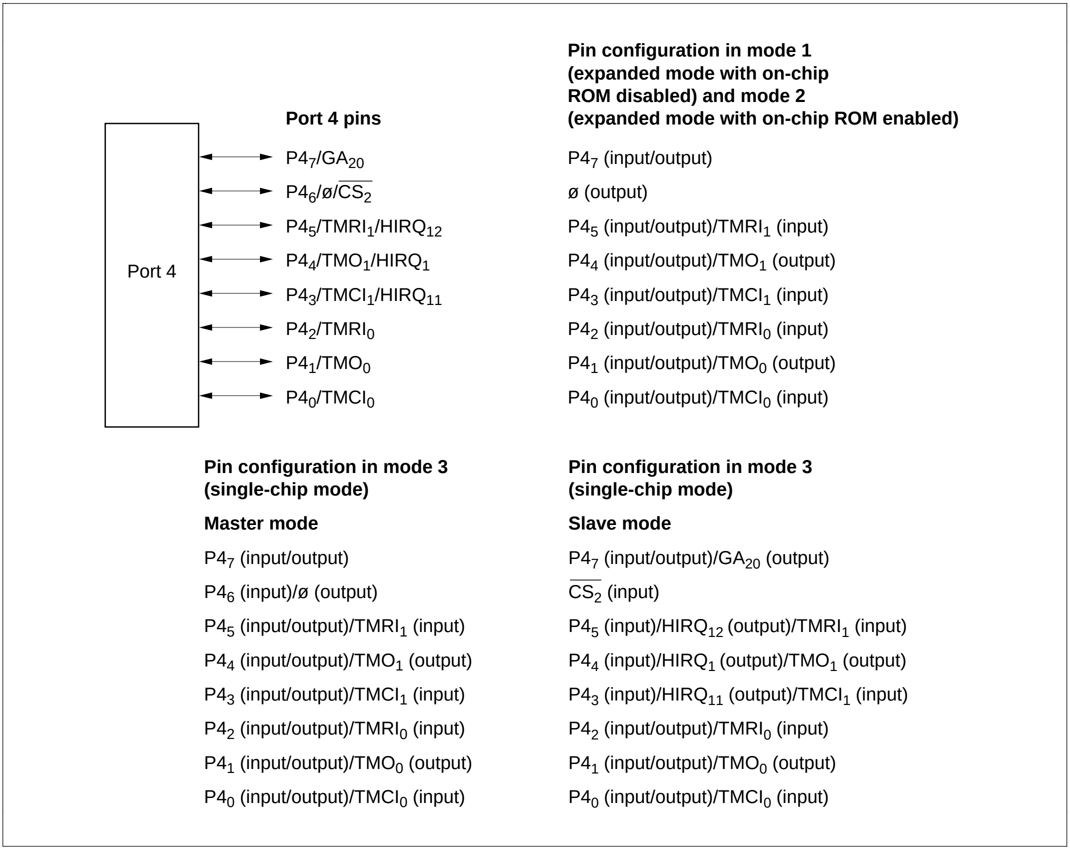
7.5 Port 4

7.5.1 Overview

Port 4 is an 8-bit input/output port that is multiplexed with the host interface (HIF) input/output pins (GA_{20}, $\overline{CS}_2$), host interrupt request output pins ($HIRQ_{12}$, $HIRQ_1$, $HIRQ_{11}$), 8-bit timer 0, and 1, and input/output pins ($TMRI_0$, $TMRI_1$, $TMCI_0$, $TMCI_1$, TMO_0, TMO_1) and the ø clock output pin. Pins $P4_7$ and $P4_5$ to $P4_0$ have the same functions in all operating modes, but the slave mode function which enables the host interface is only valid in single-chip mode. The function of pin $P4_6$ differs depending on the operating mode.

Figure 7-12 shows the pin configuration of port 4.

Pins in port 4 (except $P4_6$) can drive one TTL load and a 30-pF capacitive load. The ø clock output pin can drive one TTL load and a 90-pF capacitive load. Port 4 pins can also drive a Darlington transistor.



Port 4 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled)
$P4_7/GA_{20}$	$P4_7$ (input/output)
$P4_6$/ø/$\overline{CS}_2$	ø (output)
$P4_5/TMRI_1/HIRQ_{12}$	$P4_5$ (input/output)/$TMRI_1$ (input)
$P4_4/TMO_1/HIRQ_1$	$P4_4$ (input/output)/TMO_1 (output)
$P4_3/TMCI_1/HIRQ_{11}$	$P4_3$ (input/output)/$TMCI_1$ (input)
$P4_2/TMRI_0$	$P4_2$ (input/output)/$TMRI_0$ (input)
$P4_1/TMO_0$	$P4_1$ (input/output)/TMO_0 (output)
$P4_0/TMCI_0$	$P4_0$ (input/output)/$TMCI_0$ (input)

Pin configuration in mode 3 (single-chip mode)	Pin configuration in mode 3 (single-chip mode)
Master mode	Slave mode
$P4_7$ (input/output)	$P4_7$ (input/output)/GA_{20} (output)
$P4_6$ (input)/ø (output)	$\overline{CS}_2$ (input)
$P4_5$ (input/output)/$TMRI_1$ (input)	$P4_5$ (input)/$HIRQ_{12}$ (output)/$TMRI_1$ (input)
$P4_4$ (input/output)/TMO_1 (output)	$P4_4$ (input)/$HIRQ_1$ (output)/TMO_1 (output)
$P4_3$ (input/output)/$TMCI_1$ (input)	$P4_3$ (input)/$HIRQ_{11}$ (output)/$TMCI_1$ (input)
$P4_2$ (input/output)/$TMRI_0$ (input)	$P4_2$ (input/output)/$TMRI_0$ (input)
$P4_1$ (input/output)/TMO_0 (output)	$P4_1$ (input/output)/TMO_0 (output)
$P4_0$ (input/output)/$TMCI_0$ (input)	$P4_0$ (input/output)/$TMCI_0$ (input)

Figure 7-12 Port 4 Pin Configuration

7.5.2 Register Configuration and Descriptions

Table 7-8 summarizes the port 4 registers.

Table 7-8 Port 4 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 4 data direction register	P4DDR	W	H'40 (mode 1 and 2) H'00 (mode 3)	H'FFB5
Port 4 data register	P4DR	R/W*1	Undetermined*2	H'FFB7

Notes: 1. Bit 6 is read-only.
2. Bit 6 only is undetermined; the other bits are 0.

Port 4 Data Direction Register (P4DDR)

Bit	7	6	5	4	3	2	1	0
	$P4_7DDR$	$P4_6DDR$	$P4_5DDR$	$P4_4DDR$	$P4_3DDR$	$P4_2DDR$	$P4_1DDR$	$P4_0DDR$
Mode 1 and 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P4DDR is an 8-bit register that controls the input/output direction of each pin in port 4. A pin functions as an output pin if the corresponding P4DDR bit is set to 1, and as an input pin if this bit is cleared to 0. However, in modes 1 and 2, $P4_6DDR$ is fixed at 1 and cannot be modified.

P4DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P4DDR is initialized—to H'40 in modes 1 and 2, and to H'00 in mode 3—by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P4DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 4 is being used by an on-chip supporting module (for example, for 8-bit timer output), the on-chip supporting module will be initialized, so the pin will revert to general-purpose input/output, controlled by P4DDR and P4DR.

Port 4 Data Register (P4DR)

Bit	7	6	5	4	3	2	1	0
	$P4_7$	$P4_6$	$P4_5$	$P4_4$	$P4_3$	$P4_2$	$P4_1$	$P4_0$
Initial value	0	*	0	0	0	0	0	0
Read/Write	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Note: * Depends on the state of the $P4_6$ pin.

P4DR is an 8-bit register that stores data for port 4 pins $P4_7$ to $P4_0$. With the exception of $P4_6$, when a P4DDR bit is set to 1, if port 4 is read, the value in P4DR is obtained directly. When a P4DDR bit is cleared to 0, if port 4 is read the pin state is obtained. When $P4_6$ is read, the pin state is always obtained. This also applies to the clock output pin and pins used by the on-chip supporting modules.

P4DR bits other than $P4_6$ are initialized to 0 by a reset and in hardware standby mode. In software standby mode, P4DR retains its values prior to the mode transition.

7.5.3 Pin Functions

Port 4 pins are used for 8-bit timer and host interface input/output and øclock output. Table 7-9 indicates the pin functions of port 4.

Table 7-9 Port 4 Pin Functions

Pin — **Pin Functions and Selection Method**

P4$_7$/GA$_{20}$ — Bit FGA20E in HICR, bit P4$_7$DDR, and the operating mode select the pin function as follows

P4$_7$DDR	0	1		
FGA20E	—	0	1	
Operating mode	—	—	Other than slave mode	Slave mode
Pin function	P4$_7$ input	P4$_7$ output		GA$_{20}$ output

P4$_6$/ø/$\overline{CS}_2$ — Bit P4$_6$DDR and the operating mode select the pin function as follows

Operating mode	Modes 1 and 2	Mode 3		
	—	Other than slave mode		Slave mode
P4$_6$DDR	—	0	1	—
Pin function	ø clock output	P4$_6$ input	ø clock output	$\overline{CS}_2$ input

P4$_5$/TMRI$_1$/HIRQ$_{12}$

P4$_5$DDR	0	1	
Operating mode	—	Other than slave mode	Slave mode
Pin function	P4$_5$ input	P4$_5$ output	HIRQ$_{12}$ output
	TMRI$_1$ input		

TMRI$_1$ input is usable when bits CCLR1 and CCLR0 are both set to 1 in TCR of 8-bit timer 1

Table 7-9 Port 4 Pin Functions (cont)

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P4_4$/TMO_1/
$HIRQ_1$</td><td>Bits OS3 to OS0 in TCSR of 8-bit timer 1, bit $P4_4$DDR, and the operating mode select the pin function as follows
<table>
<tr><td>OS3 to OS0</td><td colspan="3">All 0</td><td>Not all 0</td></tr>
<tr><td>$P4_4$DDR</td><td>0</td><td colspan="2">1</td><td>—</td></tr>
<tr><td>Operating mode</td><td>—</td><td>Other than slave mode</td><td>Slave mode</td><td>—</td></tr>
<tr><td>Pin function</td><td>$P4_4$ input</td><td>$P4_4$ output</td><td>$HIRQ_1$ output</td><td>TMO_1 output</td></tr>
</table>
</td></tr>
<tr><td>$P4_3$/$TMCI_1$/
$HIRQ_{11}$</td><td>
<table>
<tr><td>$P4_3$DDR</td><td>0</td><td colspan="2">1</td></tr>
<tr><td>Operating mode</td><td>—</td><td>Other than slave mode</td><td>Slave mode</td></tr>
<tr><td rowspan="2">Pin function</td><td>$P4_3$ input</td><td>$P4_3$ output</td><td>$HIRQ_{11}$ output</td></tr>
<tr><td colspan="3">$TMCI_1$ input</td></tr>
</table>
$TMCI_1$ input is usable when bits CKS2 to CKS0 in TCR of 8-bit timer 1 select an external clock source
</td></tr>
<tr><td>$P4_2$/$TMRI_0$</td><td>
<table>
<tr><td>$P4_2$DDR</td><td>0</td><td>1</td></tr>
<tr><td rowspan="2">Pin function</td><td>$P4_2$ input</td><td>$P4_2$ output</td></tr>
<tr><td colspan="2">$TMRI_0$ input</td></tr>
</table>
$TMRI_0$ input is usable when bits CCLR1 and CCLR0 are both set to 1 in TCR of 8-bit timer 0
</td></tr>
</table>

Table 7-9 Port 4 Pin Functions (cont)

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P4_1/TMO_0$</td><td>Bits OS3 to OS0 in TCSR of 8-bit timer 0 and bit $P4_1$DDR select the pin function as follows
<table>
<tr><td>OS3 to OS0</td><td colspan="2">All 0</td><td>Not all 0</td></tr>
<tr><td>$P4_1$DDR</td><td>0</td><td>1</td><td>—</td></tr>
<tr><td>Pin function</td><td>$P4_1$ input</td><td>$P4_1$ output</td><td>TMO_0 output</td></tr>
</table></td></tr>
<tr><td>$P4_0/TMCI_0$</td><td>
<table>
<tr><td>$P4_0$DDR</td><td>0</td><td>1</td></tr>
<tr><td rowspan="2">Pin function</td><td>$P4_0$ input</td><td>$P4_0$ output</td></tr>
<tr><td colspan="2">$TMCI_0$ input</td></tr>
</table>
$TMCI_0$ input is usable when bits CKS2 to CKS0 in TCR of 8-bit timer 0 select an external clock source</td></tr>
</table>

7.6 Port 5

7.6.1 Overview

Port 5 is a 6-bit input/output port that is multiplexed with input/output pins (TxD_0, RxD_0, SCK_0, TxD_1, RxD_1, SCK_1) of serial communication interfaces 0 and 1. The port 5 pin functions are the same in all operating modes. Figure 7-13 shows the pin configuration of port 5.

Pins in port 5 can drive one TTL load and a 30-pF capacitive load. They can also drive a Darlington transistor.

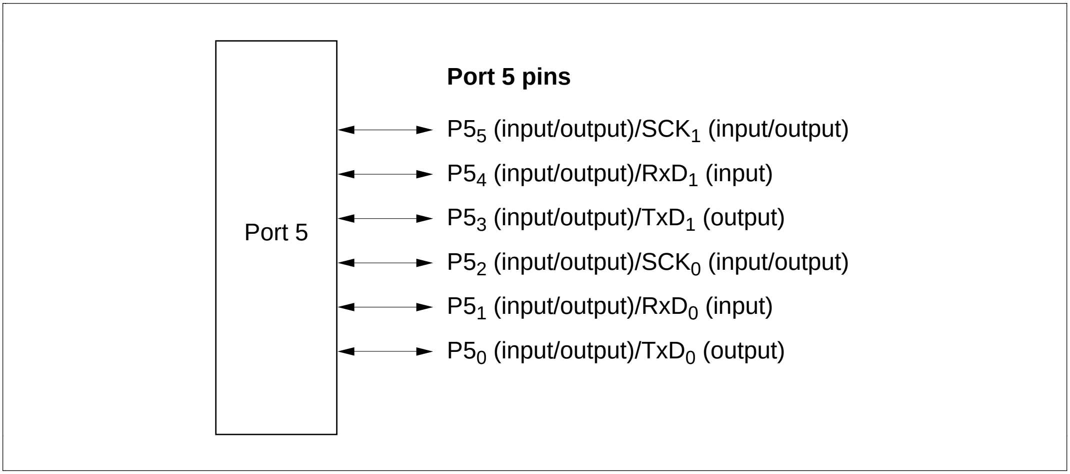

Port 5
Port 5 pins
$P5_5$ (input/output)/SCK_1 (input/output)
$P5_4$ (input/output)/RxD_1 (input)
$P5_3$ (input/output)/TxD_1 (output)
$P5_2$ (input/output)/SCK_0 (input/output)
$P5_1$ (input/output)/RxD_0 (input)
$P5_0$ (input/output)/TxD_0 (output)

Figure 7-13 Port 5 Pin Configuration

7.6.2 Register Configuration and Descriptions

Table 7-10 summarizes the port 5 registers.

Table 7-10 Port 5 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 5 data direction register	P5DDR	W	H'C0	H'FFB8
Port 5 data register	P5DR	R/W	H'C0	H'FFBA

Port 5 Data Direction Register (P5DDR)

	7	6	5	4	3	2	1	0
Bit	—	—	$P5_5$DDR	$P5_4$DDR	$P5_3$DDR	$P5_2$DDR	$P5_1$DDR	$P5_0$DDR
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	W	W	W	W	W	W

P5DDR is an 8-bit register that controls the input/output direction of each pin in port 5. A pin functions as an output pin if the corresponding P5DDR bit is set to 1, and as an input pin if this bit is cleared to 0.

P5DDR is a write-only register. Read data is invalid. Bits 7 and 6 are reserved. If read, all bits always read 1.

P5DDR is initialized to H'C0 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P5DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 5 is being used by the SCI, the SCI will be initialized, so the pin will revert to general-purpose input/output, controlled by P5DDR and P5DR.

Port 5 Data Register (P5DR)

Bit	7	6	5	4	3	2	1	0
	—	—	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	R/W	R/W	R/W	R/W	R/W	R/W

P5DR is an 8-bit register that stores data for pins $P5_5$ to $P5_0$. Bits 7 and 6 are reserved. They cannot be modified, and are always read as 1.

When a P5DDR bit is set to 1, if port 5 is read, the value in P5DR is obtained directly, regardless of the actual pin state. When a P5DDR bit is cleared to 0, if port 5 is read the pin state is obtained. This also applies to pins used as SCI pins.

P5DR is initialized to H'C0 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

7.6.3 Pin Functions

Port 5 has the same pin functions in each operating mode. Individual pins can also be used as SCI0 or SCI1 input/output pins. Table 7-11 indicates the pin functions of port 5.

Table 7-11 Port 5 Pin Functions

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P5_5/SCK_1$</td><td>Bit $C/\overline{A}$ in SMR of SCI1, bits CKE0 and CKE1 in SCR of SCI1, and bit $P5_5$DDR select the pin function as follows
<table>
<tr><td>CKE1</td><td colspan="4">0</td><td>1</td></tr>
<tr><td>$C/\overline{A}$</td><td colspan="3">0</td><td>1</td><td>—</td></tr>
<tr><td>CKE0</td><td colspan="2">0</td><td>1</td><td>—</td><td>—</td></tr>
<tr><td>$P5_5$DDR</td><td>0</td><td>1</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td>Pin function</td><td>$P5_5$ input</td><td>$P5_5$ output</td><td>SCK_1 output</td><td>SCK_1 output</td><td>SCK_1 input</td></tr>
</table></td></tr>
<tr><td>$P5_4/RxD_1$</td><td>Bit RE in SCR of SCI1 and bit $P5_4$DDR select the pin function as follows
<table>
<tr><td>RE</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$P5_4$DDR</td><td>0</td><td>1</td><td>—</td></tr>
<tr><td>Pin function</td><td>$P5_4$ input</td><td>$P5_4$ output</td><td>RxD_1 input</td></tr>
</table></td></tr>
<tr><td>$P5_3/TxD_1$</td><td>Bit TE in SCR of SCI1 and bit $P5_3$DDR select the pin function as follows
<table>
<tr><td>TE</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$P5_3$DDR</td><td>0</td><td>1</td><td>—</td></tr>
<tr><td>Pin function</td><td>$P5_3$ input</td><td>$P5_3$ output</td><td>TxD_1 output</td></tr>
</table></td></tr>
</table>

Table 7-11 Port 5 Pin Functions (cont)

Pin	Pin Functions and Selection Method
P5$_2$/SCK$_0$	Bit $C/\overline{A}$ in SMR of SCI0, bits CKE0 and CKE1 in SCR of SCI0 and bit P5$_2$DDR select the pin function as follows

CKE1	0				1
$C/\overline{A}$	0			1	—
CKE0	0		1	—	—
P5$_2$DDR	0	1	—	—	—
Pin function	P5$_2$ input	P5$_2$ output	SCK$_0$ output	SCK$_0$ output	SCK$_0$ input

Pin	Pin Functions and Selection Method
P5$_1$/RxD$_0$	Bit RE in SCR of SCI0 and bit P5$_1$DDR select the pin function as follows

RE	0		1
P5$_1$DDR	0	1	—
Pin function	P5$_1$ input	P5$_1$ output	RxD$_0$ input

Pin	Pin Functions and Selection Method
P5$_0$/TxD$_0$	Bit TE in SCR of SCI0 and bit P5$_0$DDR select the pin function as follows

TE	0		1
P5$_0$DDR	0	1	—
Pin function	P5$_0$ input	P5$_0$ output	TxD$_0$ output

7.7 Port 6

7.7.1 Overview

Port 6 is a 7-bit input/output port that is multiplexed with 16-bit free-running timer (FRT) input/output pins (FTCI, FTOA, FTOB, FTI), key-sense input pins and with $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$ input pins. The port 6 pin functions are the same in all operating modes. Pins $P6_0$ to $P6_3$ in port 6 have program-controllable built-in MOS pull-ups. Figure 7-14 shows the pin configuration of port 6.

Pins in port 6 can drive one TTL load and a 90-pF capacitive load. They can also drive a Darlington transistor.

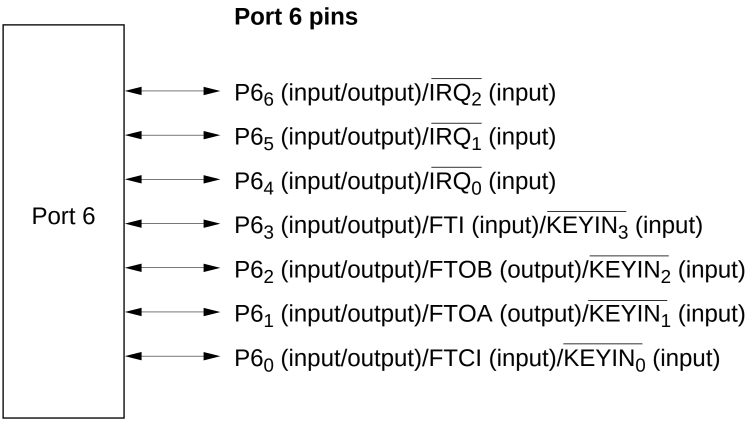

Port 6 pins
Port 6
P6$_6$ (input/output)/$\overline{\text{IRQ}_2}$ (input)
P6$_5$ (input/output)/$\overline{\text{IRQ}_1}$ (input)
P6$_4$ (input/output)/$\overline{\text{IRQ}_0}$ (input)
P6$_3$ (input/output)/FTI (input)/$\overline{\text{KEYIN}_3}$ (input)
P6$_2$ (input/output)/FTOB (output)/$\overline{\text{KEYIN}_2}$ (input)
P6$_1$ (input/output)/FTOA (output)/$\overline{\text{KEYIN}_1}$ (input)
P6$_0$ (input/output)/FTCI (input)/$\overline{\text{KEYIN}_0}$ (input)

Figure 7-14 Port 6 Pin Configuration

7.7.2 Register Configuration and Descriptions

Table 7-12 summarizes the port 6 registers.

Table 7-12 Port 6 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 6 data direction register	P6DDR	W	H'80	H'FFB9
Port 6 data register	P6DR	R/W	H'80	H'FFBB
Key-sense MOS pull-up control register	KMPCR	R/W	H'00	H'FFF2

Port 6 Data Direction Register (P6DDR)

Bit	7	6	5	4	3	2	1	0
	—	P6$_6$DDR	P6$_5$DDR	P6$_4$DDR	P6$_3$DDR	P6$_2$DDR	P6$_1$DDR	P6$_0$DDR
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

P6DDR is an 8-bit register that controls the input/output direction of each pin in port 6. A pin functions as an output pin if the corresponding P6DDR bit is set to 1, and as an input pin if this bit is cleared to 0.

P6DDR is a write-only register. Read data is invalid. Bit 7 is reserved. If read, all bits always read 1.

P6DDR is initialized to H'80 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P6DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 6 is being used by an on-chip supporting module (for example, the free-running timer), the on-chip supporting module will be initialized, so the pin will revert to general-purpose input/output, controlled by P6DDR and P6DR.

Port 6 Data Register (P6DR)

Bit	7	6	5	4	3	2	1	0
	—	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P6DR is an 8-bit register that stores data for pins $P6_6$ to $P6_0$. Bit 7 is reserved; it cannot be modified and is always read as 1. When a P6DDR bit is set to 1, if port 6 is read, the value in P6DR is obtained directly, regardless of the actual pin state. When a P6DDR bit is cleared to 0, if port 6 is read the pin state is obtained. This also applies to pins used by the on-chip supporting modules.

P6DR is initialized to H'80 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

When a port P6DDR bit is cleared to 0, if port 6 is read, the pin state is obtained; this pin can be selected according to the contents of KMIMR7 to KMIMR4. When KMIMR is set to 1 (initial value), empty bit 7, pins $P6_6$, $P6_5$, and $P6_4$ are selected. When KMIMR is cleared to 0, pins $P7_3$, $P7_2$, $P7_1$, and $P7_0$ are selected, respectively, corresponding to KMIMR7, KMIMR6, KMIMR5, and KMIMR4.

Key-Sense MOS Pull-Up Control Register (KMPCR)

Bit	7	6	5	4	3	2	1	0
	KM_7PCR	KM_6PCR	KM_5PCR	KM_4PCR	KM_3PCR	KM_2PCR	KM_1PCR	KM_0PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

KMPCR is an 8-bit readable/writable register that controls the port 6 and port 7 built-in MOS pull-ups on a bit-by-bit basis.

When a P6DDR or P7DDR bit is cleared to 0 (input port state), if the corresponding KMPCR bit is set to 1 the MOS pull-up is turned on.

KM_7PCR to KM_4PCR correspond to $P7_3DDR$ to $P7_0DDR$ and pins $P7_3$ to $P7_0$, while KM_3PCR to KM_0PCR correspond to $P6_3DDR$ to $P6_0DDR$ and pins $P6_3$ to $P6_0$.

KMPCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

7.7.3 Pin Functions

Port 6 has the same pin functions in all operating modes. The pins are multiplexed with FRT input/output, key-sense input, and $\overline{\mathrm{IRQ}_0}$ to $\overline{\mathrm{IRQ}_2}$ input. Table 7-13 indicates the pin functions of port 6.

Table 7-13 Port 6 Pin Functions

Pin — **Pin Functions and Selection Method**

($P6_7$)

KMIMR7	0	1
Pin function	$P7_3$ pin input function in a $P6_7$DR read	1 input in a $P6_7$DR read

$P6_6$/$\overline{\mathrm{IRQ}_2}$

$P6_6$DDR	0		1
KMIMR6	0	1	—
Pin function	$P7_2$ pin input function in a $P6_6$DR read	$P6_6$ input	$P6_6$ output
	$\overline{\mathrm{IRQ}_2}$ input		

$\overline{\mathrm{IRQ}_2}$ input is usable when bit IRQ2E is set to 1 in IER

$P6_5$/$\overline{\mathrm{IRQ}_1}$

$P6_5$DDR	0		1
KMIMR5	0	1	—
Pin function	$P7_1$ pin input function in a $P6_5$DR read	$P6_5$ input	$P6_5$ output
	$\overline{\mathrm{IRQ}_1}$ input		

$\overline{\mathrm{IRQ}_1}$ input is usable when bit IRQ1E is set to 1 in IER

$P6_4$/$\overline{\mathrm{IRQ}_0}$

$P6_4$DDR	0		1
KMIMR4	0	1	—
Pin function	$P7_0$ pin input function in a $P6_4$DR read	$P6_4$ input	$P6_4$ output
	$\overline{\mathrm{IRQ}_0}$ input		

$\overline{\mathrm{IRQ}_0}$ input is usable when bit IRQ0E is set to 1 in IER

Table 7-13 Port 6 Pin Functions (cont)

Pin — **Pin Functions and Selection Method**

P6$_3$/FTI/$\overline{\text{KEYIN}}_3$

P6$_3$DDR	0	1
Pin function	P6$_3$ input	P6$_3$ output
	FTI input, or $\overline{\text{KEYIN}}_3$ input	

P6$_2$/FTOB/$\overline{\text{KEYIN}}_2$

Bit OEB in TCR of the FRT, and the P6$_2$DDR bit select the pin function as follows

OEB	0		1
P6$_2$DDR	0	1	—
Pin function	P6$_2$ input	P6$_2$ output	FTOB output
	$\overline{\text{KEYIN}}_2$ input		

P6$_1$/FTOA/$\overline{\text{KEYIN}}_1$

Bit OEA in TCR of the FRT and bit P6$_1$DDR select the pin function as follows

OEA	0		1
P6$_1$DDR	0	1	—
Pin function	P6$_1$ input	P6$_1$ output	FTOA output
	$\overline{\text{KEYIN}}_1$ input		

P6$_0$/FTCI/$\overline{\text{KEYIN}}_0$

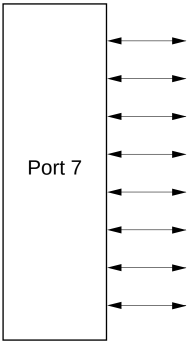
P6$_0$DDR	0	1
Pin function	P6$_0$ input	P6$_0$ output
	FTCI input or $\overline{\text{KEYIN}}_0$ input	

FTCI input is usable when bits CKS1 and CKS0 in TCR of the FRT select an external clock source

7.8 Port 7

7.8.1 Overview

Port 7 is an 8-bit input/output port that also provides the bus control signal input/output pins ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{AS}}$, $\overline{\text{WAIT}}$), host interface (HIF) input pins (HA_0, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{CS}_1}$), and key-sense input pins. The functions of pins P7_7 to P7_4 differ depending on the operating mode. Pins P7_0 to P7_3 have program-controllable built-in MOS pull-ups. Figure 7-15 shows the pin configuration of port 7. Pins in port 7 can drive one TTL load and a 90-pF capacitive load. They can also drive a Darlington transistor.



Port 7 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled)
$\text{P7}_7/\overline{\text{WAIT}}/\text{HA}_0$	$\overline{\text{WAIT}}$ (input)/P7_7 (input)
$\text{P7}_6/\overline{\text{RD}}/\overline{\text{IOR}}$	$\overline{\text{RD}}$ (output)
$\text{P7}_5/\overline{\text{WR}}/\overline{\text{IOW}}$	$\overline{\text{WR}}$ (output)
$\text{P7}_4/\overline{\text{AS}}/\overline{\text{CS}_1}$	$\overline{\text{AS}}$ (output)
$\text{P7}_3/\overline{\text{KEYIN}_7}$	P7_3 (input/output)/$\overline{\text{KEYIN}_7}$ (input)
$\text{P7}_2/\overline{\text{KEYIN}_6}$	P7_2 (input/output)/$\overline{\text{KEYIN}_6}$ (input)
$\text{P7}_1/\overline{\text{KEYIN}_5}$	P7_1 (input/output)/$\overline{\text{KEYIN}_5}$ (input)
$\text{P7}_0/\overline{\text{KEYIN}_4}$	P7_0 (input/output)/$\overline{\text{KEYIN}_4}$ (input)

Pin configuration in mode 3 (single-chip mode)	Pin configuration in mode 3 (single-chip mode)
Master mode	**Slave mode**
P7_7 (input/output)	HA_0 (input)
P7_6 (input/output)	$\overline{\text{IOR}}$ (input)
P75 (input/output)	$\overline{\text{IOW}}$ (input)
P7_4 (input/output)	$\overline{\text{CS}_1}$ (input)
P7_3 (input/output)/$\overline{\text{KEYIN}_7}$ (input)	P7_3 (input/output)/$\overline{\text{KEYIN}_7}$ (input)
P7_2 (input/output)/$\overline{\text{KEYIN}_6}$ (input)	P7_2 (input/output)/$\overline{\text{KEYIN}_6}$ (input)
P7_1 (input/output)/$\overline{\text{KEYIN}_5}$ (input)	P7_1 (input/output)/$\overline{\text{KEYIN}_5}$ (input)
P7_0 (input/output)/$\overline{\text{KEYIN}_4}$ (input)	P7_0 (input/output)/$\overline{\text{KEYIN}_4}$ (input)

Figure 7-15 Port 7 Pin Configuration

7.8.2 Register Configuration and Descriptions

Table 7-15 summarizes the port 7 registers.

Table 7-15 Port 7 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 7 data direction register	P7DDR	W	H'00	H'FFBC
Port 7 data register	P7DR	R/W	H'00	H'FFBE
Key-sense MOS pull-up control register	KMPCR	R/W	H'00	H'FFF2

Port 7 Data Direction Register (P7DDR)

	7	6	5	4	3	2	1	0
Bit	$P7_7$DDR	$P7_6$DDR	$P7_5$DDR	$P7_4$DDR	$P7_3$DDR	$P7_2$DDR	$P7_1$DDR	$P7_0$DDR
Initial value	0	0	0	0	0	0	0	1
Read/Write	W	W	W	W	W	W	W	W

P7DDR is an 8-bit register that controls the input/output direction of each pin in port 7. A pin functions as an output pin if the corresponding P7DDR bit is set to 1, and as an input pin if this bit is cleared to 0. P7DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P7DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode P7DDR retains its existing values, so if a transition to software standby mode occurs while a P7DDR bit is set to 1, the corresponding pin remains in the output state.

Port 7 Data Register (P7DR)

Bit	7	6	5	4	3	2	1	0
	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P7DR is an 8-bit register that stores data for pins $P7_7$ to $P7_0$.

When a P7DDR bit is set to 1, if port 7 is read, the value in P7DR is obtained directly, regardless of the actual pin state. When a P7DDR bit is cleared to 0, if port 7 is read the pin state is obtained.

P7DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

When a port P6DDR bit is cleared to 0, if port 6 is read, the pin state is obtained; this pin can be selected according to the contents of KMIMR7 to KMIMR4. When KMIMR is set to 1 (initial value), bit 7 is an empty bit, and pins $P6_6$, $P6_5$, and $P6_4$ are selected. When KMIMR is cleared to 0, pins $P7_3$, $P7_2$, $P7_1$, and $P7_0$ are selected, respectively, corresponding to KMIMR7, KMIMR6, KMIMR5, and KMIMR4.

Key-Sense MOS Pull-Up Control Register (KMPCR)

Bit	7	6	5	4	3	2	1	0
	KM_7PCR	KM_6PCR	KM_5PCR	KM_4PCR	KM_3PCR	KM_2PCR	KM_1PCR	KM_0PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

KMPCR is an 8-bit readable/writable register that controls the port 6 and port 7 built-in MOS pull-ups on a bit-by-bit basis.

When a P6DDR or P7DDR bit is cleared to 0 (input port state), if the corresponding KMPCR bit is set to 1 the MOS pull-up is turned on.

KM_7PCR to KM_4PCR correspond to $P7_3DDR$ to $P7_0DDR$ and pins $P7_3$ to $P7_0$, while KM_3PCR to KM_0PCR correspond to $P6_3DDR$ to $P6_0DDR$ and pins $P6_3$ to $P6_0$.

KMPCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

7.8.3 Pin Functions

The pins of port 7 have different functions in modes 1 and 2 and in mode 3. Individual pins are used as bus control signal input/output pins ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{AS}}$, $\overline{\text{WAIT}}$), host interface (HIF) input pins (HA_0, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{CS}}_1$), and key-sense input pins. Table 7-19 indicates the pin functions of port 7.

Table 7-16 Port 7 Pin Functions

Pin	Pin Functions and Selection Method
P7_7/$\overline{\text{WAIT}}$/HA_0	Bit P7_7DDR, the wait mode determined by WSCR, and the operating mode select the pin function as follows

Operating mode	Modes 1 and 2			Mode 3		
	—			Other than slave mode		Slave mode
Wait mode	$\overline{\text{WAIT}}$ used	$\overline{\text{WAIT}}$ not used		—		
P7_7DDR	—	0	1	0	1	—
Pin function	$\overline{\text{WAIT}}$ input	P7_7 input	—	P7_7 input	P7_7 output	HA_0 input

Pin	Pin Functions and Selection Method
P7_6/$\overline{\text{RD}}$/$\overline{\text{IOR}}$	Bit P7_6DDR and the operating mode select the pin function as follows

Operating mode	Modes 1 and 2	Mode 3		
	—	Other than slave mode		Slave mode
P7_6DDR	—	0	1	—
Pin function	$\overline{\text{RD}}$ output	P7_6 input	P7_6 output	$\overline{\text{IOR}}$ input

Pin	Pin Functions and Selection Method
P7_5/$\overline{\text{WR}}$/$\overline{\text{IOW}}$	Bit P7_5DDR and the operating mode select the pin function as follows

Operating mode	Modes 1 and 2	Mode 3		
	—	Other than slave mode		Slave mode
P7_5DDR	—	0	1	—
Pin function	$\overline{\text{WR}}$ output	P7_5 input	P7_5 output	$\overline{\text{IOW}}$ input

Table 7-16 Port 7 Pin Functions (cont)

Pin | **Pin Functions and Selection Method**

$P7_4/\overline{AS}/\overline{CS}_1$

Bit $P7_4$DDR and the operating mode select the pin function as follows

Operating mode	Modes 1 and 2	Mode 3		
	—	Other than slave mode		Slave mode
$P7_4$DDR	—	0	1	—
Pin function	$\overline{AS}$ output	$P7_4$ input	$P7_4$ output	$\overline{CS}_1$ input

$P7_3/\overline{KEYIN}_7$

Bit $P7_3$DDR select the pin function as follows

$P7_3$DDR	0	1
Pin function	$P7_3$ input	$P7_3$ output
	$\overline{KEYIN}_7$ input	

$P7_2/\overline{KEYIN}_6$

Bit $P7_2$DDR select the pin function as follows

$P7_2$DDR	0	1
Pin function	$P7_2$ input	$P7_2$ output
	$\overline{KEYIN}_6$ input	

$P7_1$

Bit $P7_1$DDR select the pin function as follows

$P7_1$DDR	0	1
Pin function	$P7_1$ input	$P7_1$ output
	$\overline{KEYIN}_5$ input	

$P7_0$

Bit $P7_0$DDR select the pin function as follows

$P7_0$DDR	0	1
Pin function	$P7_0$ input	$P7_0$ output
	$\overline{KEYIN}_4$ input	

Section 8 16-Bit Free-Running Timer

8.1 Overview

The H8/3502 has an on-chip 16-bit free-running timer (FRT) module that uses a 16-bit free-running counter as a time base. Applications of the FRT module include rectangular-wave output (up to two independent waveforms), input pulse width measurement, and measurement of external clock periods.

8.1.1 Features

The features of the free-running timer module are listed below.

- Selection of four clock sources

  The free-running counter can be driven by an internal clock source (ø$_P$/2, ø$_P$/8, or ø$_P$/32), or an external clock input (enabling use as an external event counter).

- Two independent comparators

  Each comparator can generate an independent waveform.

- Input capture

  The current count can be captured on the rising or falling edge (selectable) of an input signal.

- Counter can be cleared under program control

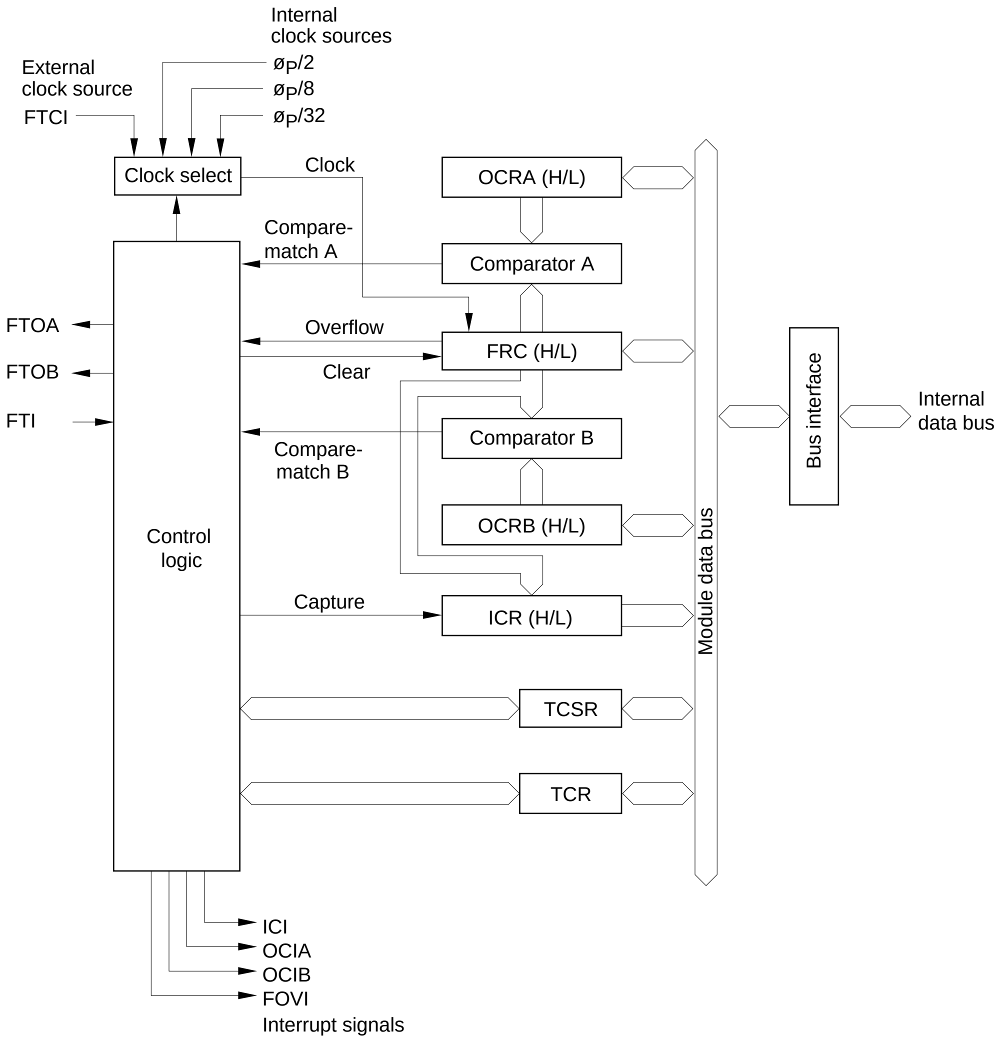
  The free-running counter can be cleared on compare-match A.

- Four interrupt sources

  Compare-match A and B, input capture, and overflow interrupts are requested independently.

8.1.2 Block Diagram

Figure 8-1 shows a block diagram of the free-running timer.



Legend

OCRA:	Output compare register A
OCRB:	Output compare register B
FRC:	Free-running counter
ICR:	Input capture register
TCSR:	Timer control/status register
TCR:	Timer control register

Figure 8-1 Block Diagram of 16-Bit Free-Running Timer

8.1.3 Input and Output Pins

Table 8-1 lists the input and output pins of the free-running timer module.

Table 8-1 Input and Output Pins of Free-Running Timer Module

Name	Abbreviation	I/O	Function
Counter clock input	FTCI	Input	Input of external free-running counter clock signal
Output compare A	FTOA	Output	Output controlled by comparator A
Output compare B	FTOB	Output	Output controlled by comparator B
Input capture	FTI	Input	Input capture trigger

8.1.4 Register Configuration

Table 8-2 lists the registers of the free-running timer module.

Table 8-2 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address
Timer control register	TCR	R/W	H'00	H'FF90
Timer control/status register	TCSR	R/(W)*	H'00	H'FF91
Free-running counter (high)	FRC (H)	R/W	H'00	H'FF92
Free-running counter (low)	FRC (L)	R/W	H'00	H'FF93
Output compare register A (high)	OCRA (H)	R/W	H'FF	H'FF94
Output compare register A (low)	OCRA (L)	R/W	H'FF	H'FF95
Output compare register B (high)	OCRB (H)	R/W	H'FF	H'FF96
Output compare register B (low)	OCRB (L)	R/W	H'FF	H'FF97
Input capture register (high)	ICR (H)	R	H'00	H'FF98
Input capture register (low)	ICR (L)	R	H'00	H'FF99

Note: * Software can write a 0 to clear bits 7 to 4, but cannot write a 1 in these bits.

8.2 Register Descriptions

8.2.1 Free-Running Counter (FRC)—H'FF92

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The FRC is a 16-bit readable/writable up-counter that increments on an internal pulse generated from a clock source. The clock source is selected by the clock select 1 and 0 bits (CKS1 and CKS0) of the timer control register (TCR).

When the FRC overflows from H'FFFF to H'0000, the overflow flag (OVF) in the timer control/status register (TCSR) is set to 1.

Because the FRC is a 16-bit register, a temporary register (TEMP) is used when the FRC is written or read. See section 8.3, CPU Interface, for details.

The FRC is initialized to H'0000 by a reset and in the standby modes.

8.2.2 Output Compare Registers A and B (OCRA and OCRB)—H'FF94 and H'FF96

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OCRA and OCRB are 16-bit readable/writable registers, the contents of which are continually compared with the value in the FRC. When a match is detected, the corresponding output compare flag (OCFA or OCFB) is set to 1 in the timer control/status register (TCSR).

In addition, if the output enable bit (OEA or OEB) in the timer output compare control register (TCR) is set to 1, when the output compare register and FRC values match, the logic level selected by the output level bit (OLVLA or OLVLB) in the TCSR is output at the output compare pin (FTOA or FTOB). After a reset, the output of FTOA and FTOB is 0 until the first compare-match event.

Because OCRA and OCRB are 16-bit registers, a temporary register (TEMP) is used for write access, as explained in section 8.3, CPU Interface.

OCRA and OCRB are initialized to H'FFFF by a reset and in the standby modes.

8.2.3 Input Capture Register (ICR)—H'FF98

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The input capture register is a 16-bit read-only register.

When the rising or falling edge of the signal at the input capture pin (FTI) is detected, the current value of the FRC is copied to the input capture register (ICR). At the same time, the input capture flag (ICF) in the timer control/status register (TCSR) is set to 1. The input capture edge is selected by the input edge select bit (IEDG) in the TCSR.

Because the input capture register is a 16-bit register, a temporary register (TEMP) is used when it is read. See Section 8.3, CPU Interface, for details.

To ensure input capture, the width of the input capture pulse (FTI) should be at least 1.5 system clock cycles (1.5 ø).

The input capture register is initialized to H'0000 by a reset and in the standby modes.

Note: When input capture is detected, the FRC value is transferred to the input capture register even if the input capture flag is already set.

8.2.4 Timer Control Register (TCR)—H'FF90

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCR is an 8-bit readable/writable register that enables and disables output signals and interrupts, and selects the timer clock source.

TCR is initialized to H'00 by a reset and in the standby modes.

Bit 7—Input Capture Interrupt Enable (ICIE): Selects whether to request an input capture interrupt (ICI) when the input capture flag (ICF) in the timer status/control register (TCSR) is set to 1.

Bit 7 ICIE	Description	
0	Input capture interrupt request (ICI) is disabled	(Initial value)
1	Input capture interrupt request (ICI) is enabled	

Bit 6—Output Compare Interrupt Enable B (OCIEB): Selects whether to request output compare interrupt B (OCIB) when output compare flag B (OCFB) in the timer status/control register (TCSR) is set to 1.

Bit 6 OCIEB	Description	
0	Output compare interrupt request B (OCIB) is disabled	(Initial value)
1	Output compare interrupt request B (OCIB) is enabled	

Bit 5—Output Compare Interrupt Enable A (OCIEA): Selects whether to request output compare interrupt A (OCIA) when output compare flag A (OCFA) in the timer status/control register (TCSR) is set to 1.

Bit 5 OCIEA	Description	
0	Output compare interrupt request A (OCIA) is disabled	(Initial value)
1	Output compare interrupt request A (OCIA) is enabled	

Bit 4—Timer Overflow Interrupt Enable (OVIE): Selects whether to request a free-running timer overflow interrupt (FOVI) when the timer overflow flag (OVF) in the timer status/control register (TCSR) is set to 1.

Bit 4 OVIE	Description	
0	Timer overflow interrupt request (FOVI) is disabled	(Initial value)
1	Timer overflow interrupt request (FOVI) is enabled	

Bit 3—Output Enable B (OEB): Enables or disables output of the output compare B signal (FTOB). If output compare B is enabled, the FTOB pin is driven to the level selected by OLVLB in the timer status/control register (TCSR) whenever the FRC value matches the value in output compare register B (OCRB).

Bit 3 OEB	Description	
0	Output compare B output is disabled	(Initial value)
1	Output compare B output is enabled	

Bit 2—Output Enable A (OEA): Enables or disables output of the output compare A signal (FTOA). If output compare A is enabled, the FTOA pin is driven to the level selected by OLVLA in the timer status/control register (TCSR) whenever the FRC value matches the value in output compare register A (OCRA).

Bit 2 OEA	Description	
0	Output compare A output is disabled	(Initial value)
1	Output compare A output is enabled	

Bits 1 and 0—Clock Select (CKS1 and CKS0): These bits select external clock input or one of three internal clock sources for the FRC. External clock pulses are counted on the rising edge at the external clock pin (FTCI).

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	$ø_P$/2 internal clock source	(Initial value)
0	1	$ø_P$/8 internal clock source	
1	0	$ø_P$/32 internal clock source	
1	1	External clock source (rising edge)	

8.2.5 Timer Control/Status Register (TCSR)—H'FF91

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/W	R/W	R/W	R/W

Note: * Software can write a 0 in bits 7 to 4 to clear the flags, but cannot write a 1 in these bits.

TCSR is an 8-bit readable and partially writable register that contains four status flags and selects the output compare levels, input capture edge, and whether to clear the counter on compare-match A.

TCSR is initialized to H'00 by a reset and in the standby modes.

Bit 7—Input Capture Flag (ICF): This status flag is set to 1 to indicate an input capture event, showing that the FRC value has been copied to the ICR.

ICF must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 7 **ICF**	**Description**	
0	To clear ICF, the CPU must read ICF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when an FTI input signal causes the FRC value to be copied to the ICR	

Bit 6—Output Compare Flag B (OCFB): This status flag is set to 1 when the FRC value matches the OCRB value.

OCFB must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 6 **OCFB**	**Description**	
0	To clear OCFB, the CPU must read OCFB after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when FRC = OCRB	

Bit 5—Output Compare Flag A (OCFA): This status flag is set to 1 when the FRC value matches the OCRA value.

OCFA must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 5 OCFA	Description	
0	To clear OCFA, the CPU must read OCFA after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when FRC = OCRA	

Bit 4—Timer Overflow Flag (OVF): This status flag is set to 1 when the FRC overflows (changes from H'FFFF to H'0000).

OVF must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 4 OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when FRC changes from H'FFFF to H'0000	

Bit 3—Output Level B (OLVLB): Selects the logic level output at the FTOB pin when the FRC and OCRB values match.

Bit 3 OLVLB	Description	
0	A 0 logic level is output for compare-match B	(Initial value)
1	A 1 logic level is output for compare-match B	

Bit 2—Output Level A (OLVLA): Selects the logic level output at the FTOA pin when the FRC and OCRA values match.

Bit 2 OLVLA	Description	
0	A 0 logic level is output for compare-match A	(Initial value)
1	A 1 logic level is output for compare-match A	

Bit 1—Input Edge Select (IEDG): Selects the rising or falling edge of the input capture signal (FTI).

Bit 1 IEDG	Description	
0	FRC contents are transferred to ICR on the falling edge of FTI	(Initial value)
1	FRC contents are transferred to ICR on the rising edge of FTI	

Bit 0—Counter Clear A (CCLRA): Selects whether to clear the FRC at compare-match A (when the FRC and OCRA values match).

Bit 0 CCLRA	Description	
0	The FRC is not cleared	(Initial value)
1	The FRC is cleared at compare-match A	

8.3 CPU Interface

The free-running counter (FRC), output compare registers (OCRA and OCRB), and input capture register (ICR) are 16-bit registers, but they are connected to an 8-bit data bus. When the CPU accesses these registers, to ensure that both bytes are written or read simultaneously, the access is performed using an 8-bit temporary register (TEMP).

These registers are written and read as follows:

- Register write

  When the CPU writes to the upper byte, the byte of write data is placed in TEMP. Next, when the CPU writes to the lower byte, this byte of data is combined with the byte in TEMP and all 16 bits are written in the register simultaneously.

- Register read

  When the CPU reads the upper byte, the upper byte of data is sent to the CPU and the lower byte is placed in TEMP. When the CPU reads the lower byte, it receives the value in TEMP.

  (As an exception, when the CPU reads OCRA or OCRB, it reads both the upper and lower bytes directly, without using TEMP.)

Programs that access these registers should normally use word access. Equivalently, they may access first the upper byte, then the lower byte by two consecutive byte accesses. Data will not be transferred correctly if the bytes are accessed in reverse order, if only one byte is accessed, or if the upper and lower bytes are accessed separately and another register is accessed in between, altering the value in TEMP.

Coding Examples

To write the contents of general register R0 to OCRA: `MOV.W  R0, @OCRA`

To transfer the ICR contents to general register R0: `MOV.W  @ICR, R0`

Figure 8-2 shows the data flow when the FRC is accessed. The other registers are accessed in the same way.

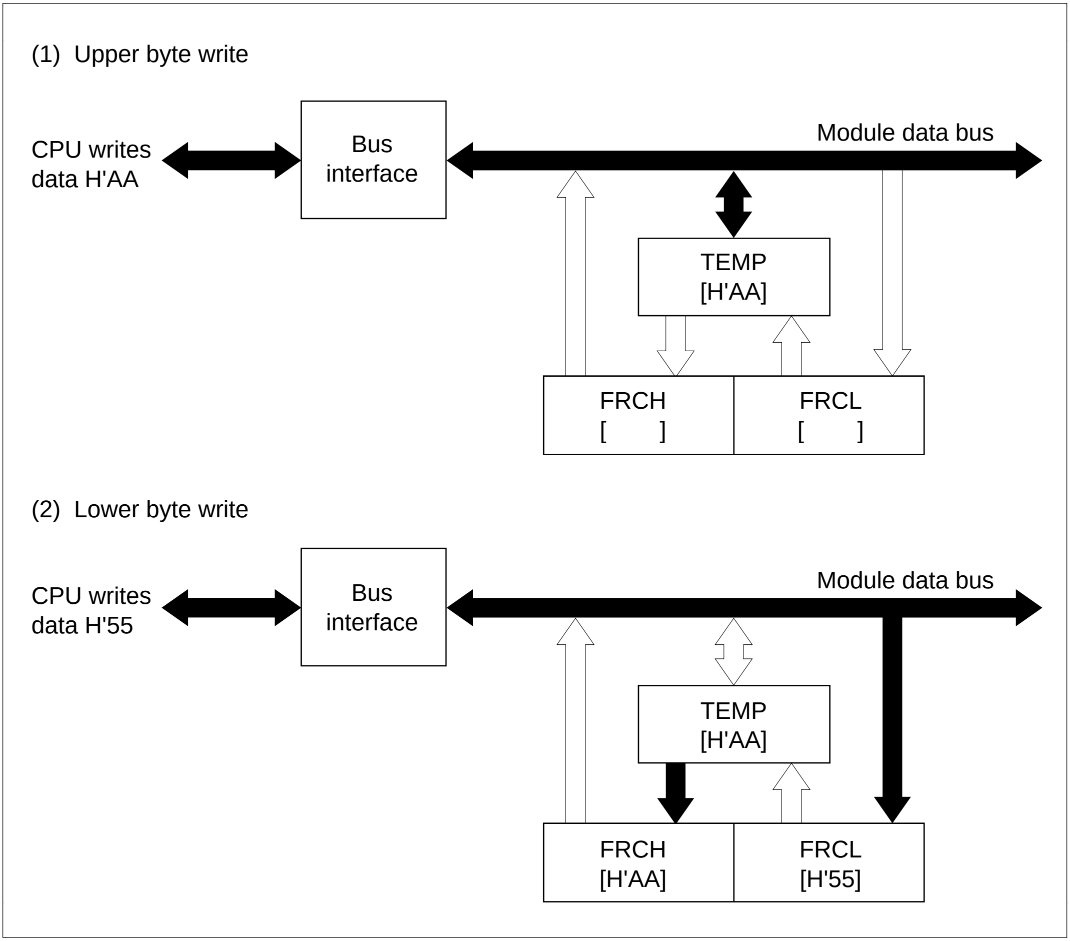


Figure 8-2 (a) Write Access to FRC (When CPU Writes H'AA55)

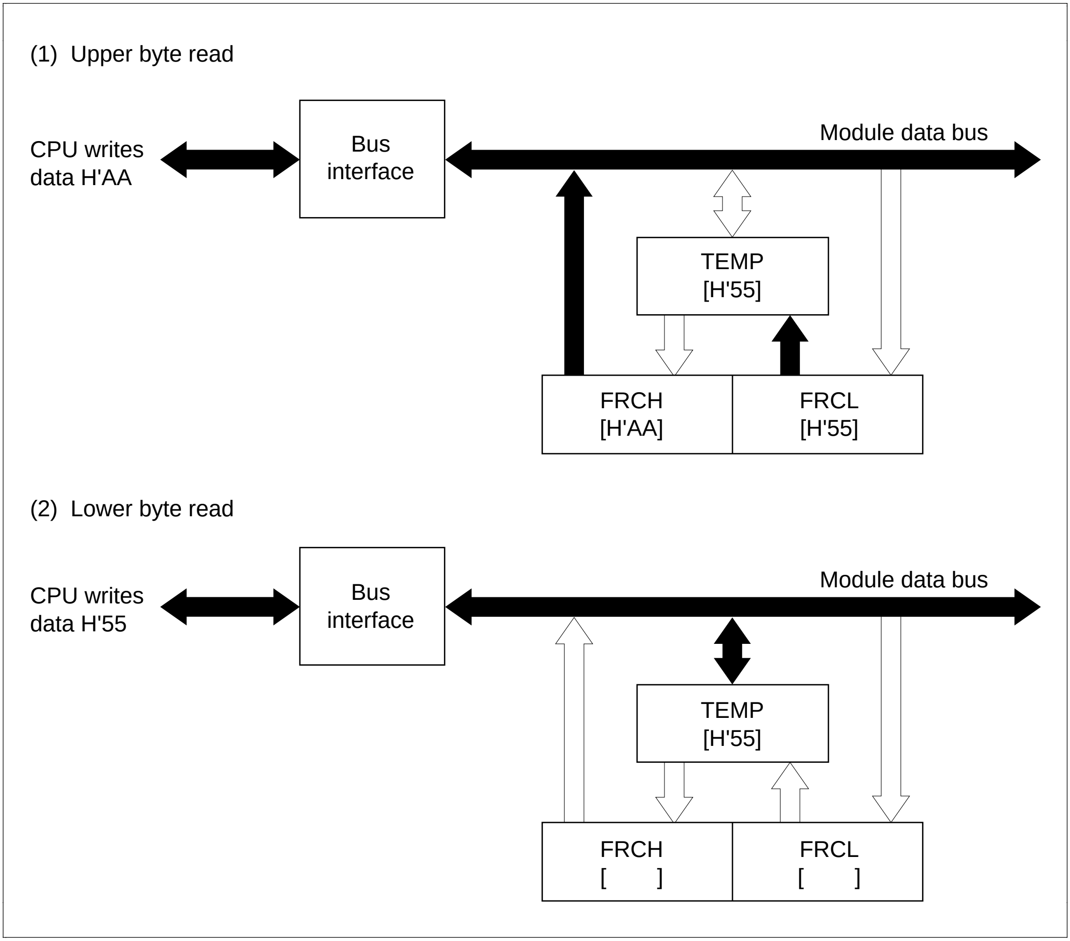


Figure 8-2 (b) Read Access to FRC (When FRC Contains H'AA55)

8.4 Operation

8.4.1 FRC Incrementation Timing

The FRC increments on a pulse generated once for each cycle of the selected (internal or external) clock source.

(1) Internal Clock Sources: Can be selected by the CKS1 and CKS0 bits in TCR. Internal clock sources are created by dividing the system clock (ø). Three internal clock sources are available: $ø_P/2$, $ø_P/8$, and $ø_P/32$. Figure 8-3 shows the increment timing.

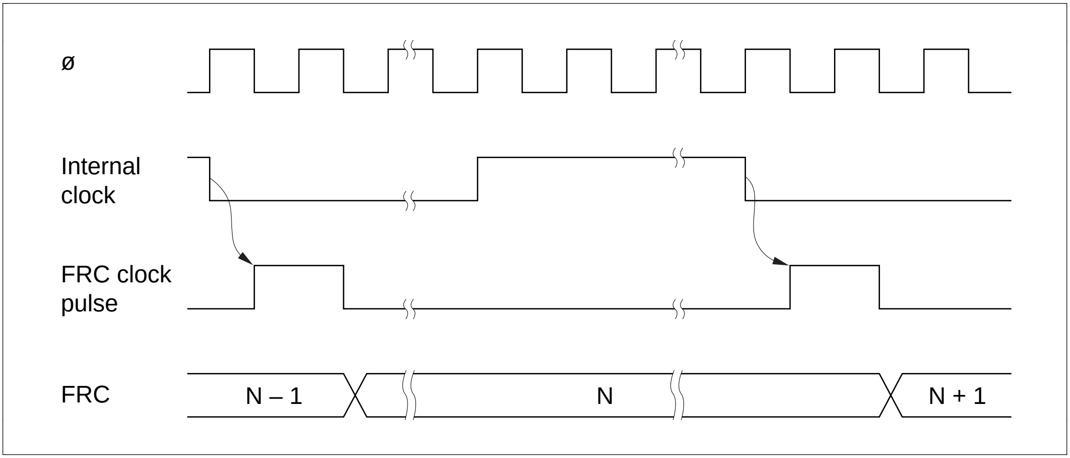


Figure 8-3 Increment Timing for Internal Clock Source

(2) External Clock Input: Can be selected by the CKS1 and CKS0 bits in the TCR. The FRC increments on the rising edge of the FTCI clock signal. The pulse width of the external clock signal must be at least 1.5 system clock (ø) cycles. The counter will not increment correctly if the pulse width is shorter than this.

Figure 8-4 shows the increment timing.

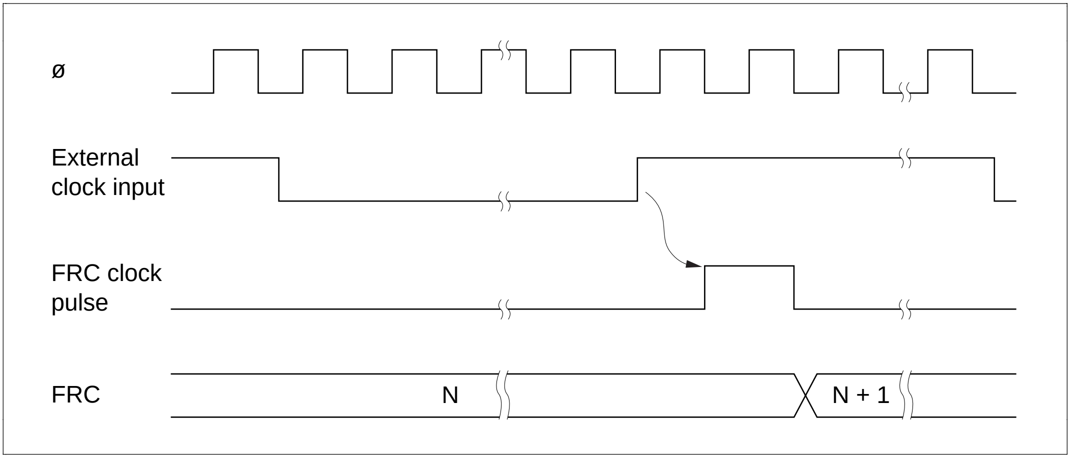


Figure 8-4 Increment Timing for External Clock Source

8.4.2 Output Compare Timing

When a compare-match occurs, the logic level selected by the output level bit (OLVLA or OLVLB) in TCSR is output at the output compare pin (FTOA or FTOB). Figure 8-5 shows the timing of this operation for compare-match A.

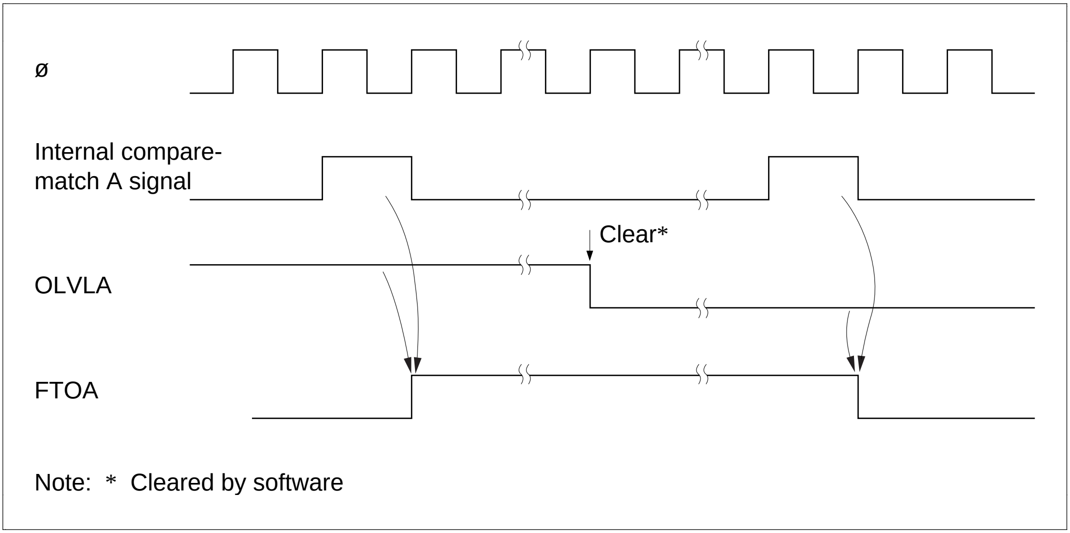


Figure 8-5 Timing of Output Compare A

8.4.3 FRC Clear Timing

If the CCLRA bit in TCSR is set to 1, the FRC is cleared when compare-match A occurs. Figure 8-6 shows the timing of this operation.

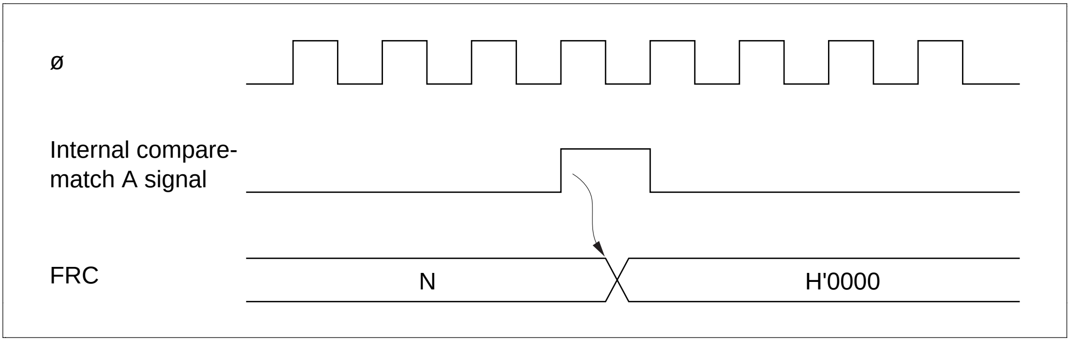


Figure 8-6 Clearing of FRC by Compare-Match A

8.4.4 Input Capture Timing

An internal input capture signal is generated from the rising or falling edge of the FTI input, as selected by the IEDG bit in TCSR. Figure 8-7 shows the usual input capture timing when the rising edge is selected (IEDG = 1).

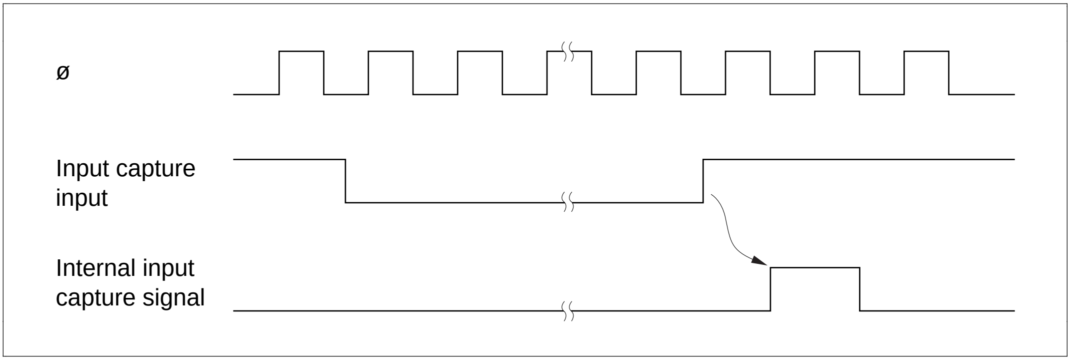


Figure 8-7 Input Capture Timing (Usual Case)

If the upper byte of ICR is being read when the internal input capture signal should be generated, the internal input capture signal is delayed by one state. Figure 8-8 shows the timing for this case.

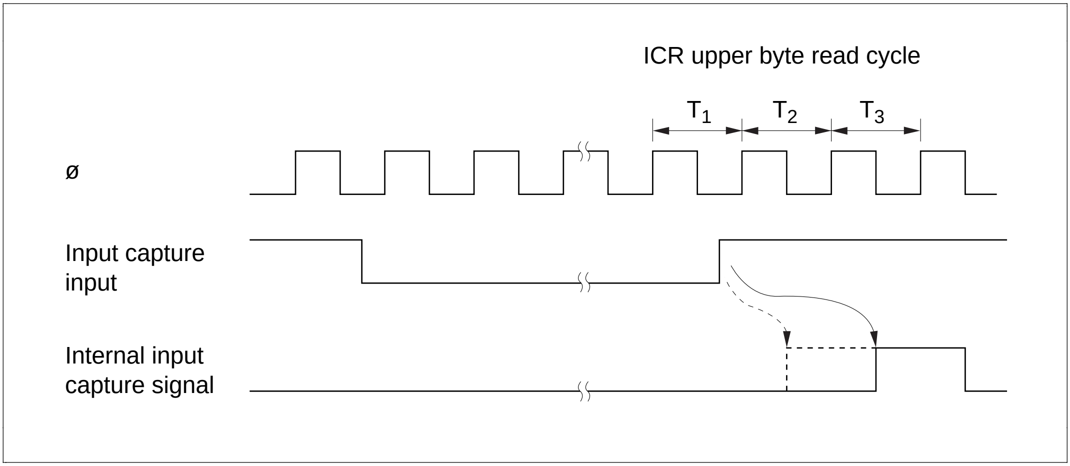


Figure 8-8 Input Capture Timing (1-State Delay Due to ICR Read)

8.4.5 Timing of Input Capture Flag (ICF) Setting

The input capture flag ICF is set to 1 by the internal input capture signal. The FRC contents are transferred to ICR at the same time. Figure 8-9 shows the timing of this operation.

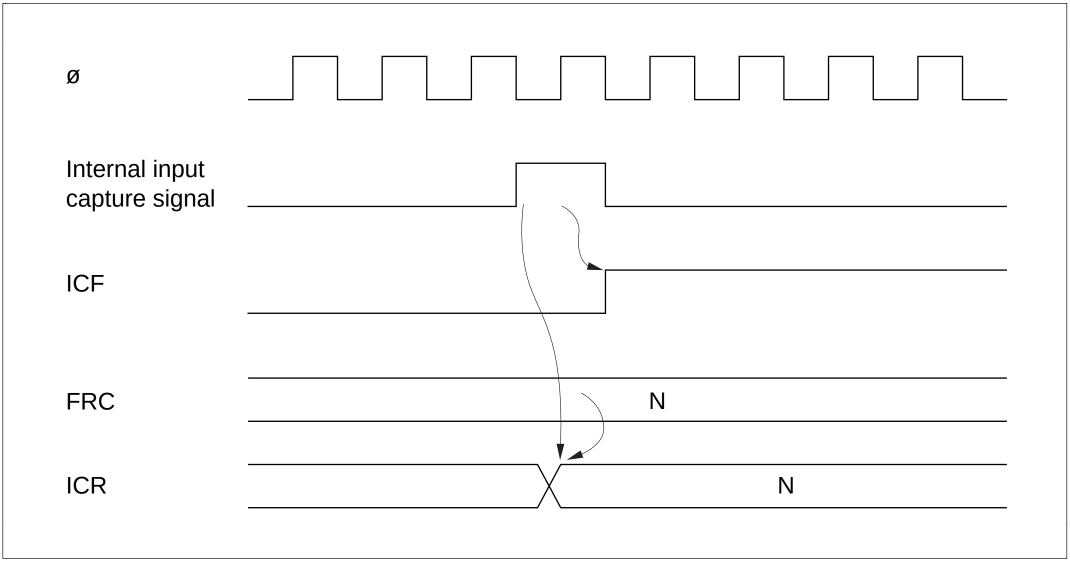


Figure 8-9 Setting of Input Capture Flag

8.4.6 Setting of FRC Overflow Flag (OVF)

The FRC overflow flag (OVF) is set to 1 when the FRC changes from H'FFFF to H'0000. Figure 8-10 shows the timing of this operation.

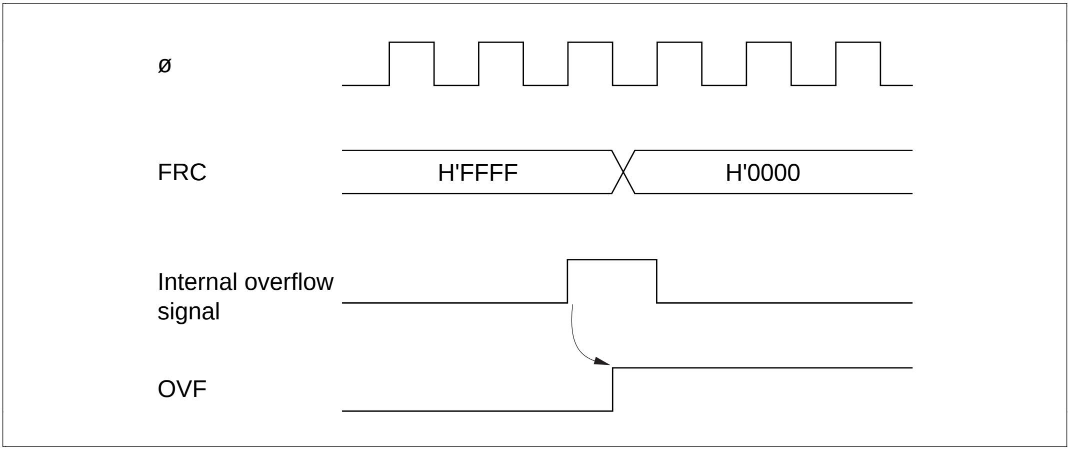


Figure 8-10 Setting of Overflow Flag

8.5 Interrupts

The free-running timer module can request four types of interrupts: input capture (ICI), output compare A and B (OCIA and OCIB), and overflow (FOVI). Each interrupt is requested when the corresponding flag bit is set, provided the corresponding enable bit is also set. Independent signals are sent to the interrupt controller for each type of interrupt. Table 8-3 lists information about these interrupts.

Table 8-3 Free-Running Timer Interrupts

Interrupt	Description	Priority
ICI	Requested when ICF is set	High
OCIA	Requested when OCFA is set	↕
OCIB	Requested when OCFB is set	
FOVI	Requested when OVF is set	Low

8.6 Sample Application

In the example below, the free-running timer module is used to generate two square-wave outputs with a 50% duty factor and arbitrary phase relationship. The programming is as follows:

1. The CCLRA bit in TCSR is set to 1.

2. Each time a compare-match interrupt occurs, software inverts the corresponding output level bit in TCSR (OLVLA or OLVLB).

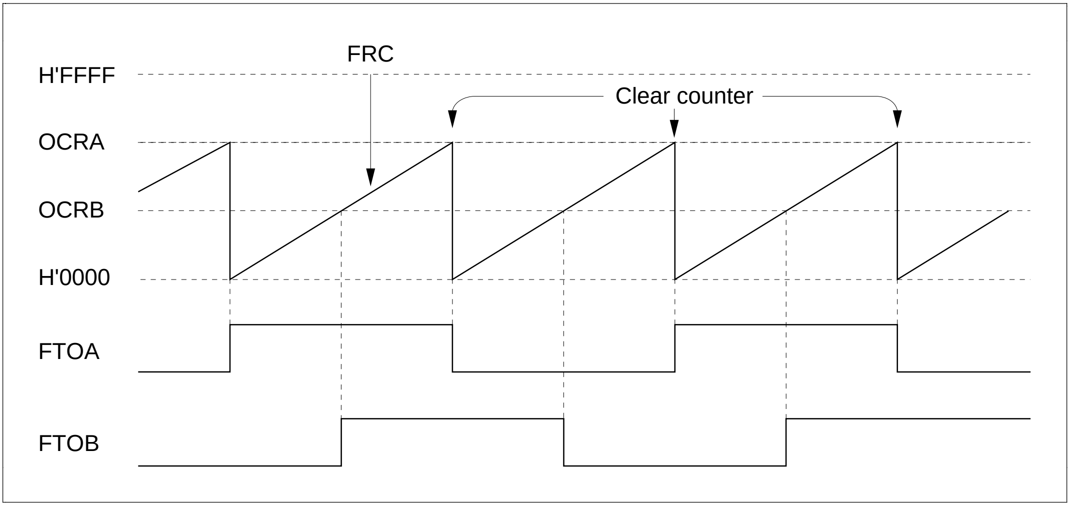


Figure 8-11 Square-Wave Output (Example)

8.7 Application Notes

Application programmers should note that the following types of contention can occur in the free-running timer.

(1) Contention between FRC Write and Clear: If an internal counter clear signal is generated during the T_3 state of a write cycle to the lower byte of the free-running counter, the clear signal takes priority and the write is not performed.

Figure 8-12 shows this type of contention.

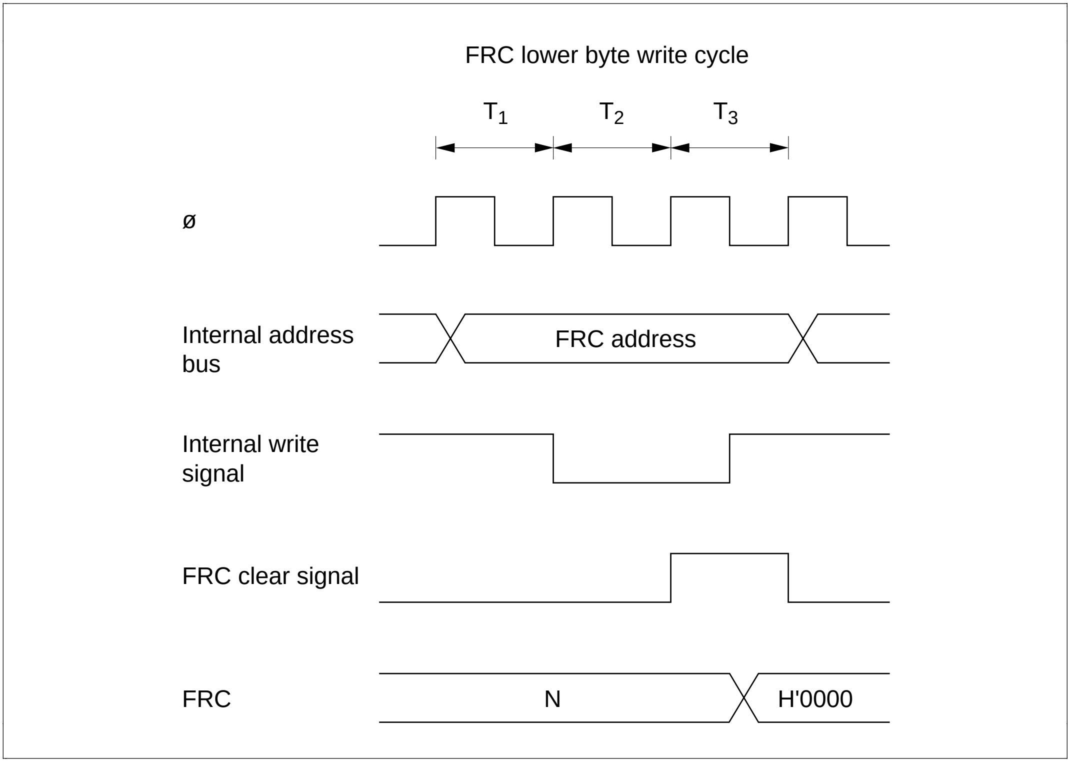


Figure 8-12 FRC Write-Clear Contention

(2) Contention between FRC Write and Increment: If an FRC increment pulse is generated during the T_3 state of a write cycle to the lower byte of the free-running counter, the write takes priority and the FRC is not incremented.

Figure 8-13 shows this type of contention.

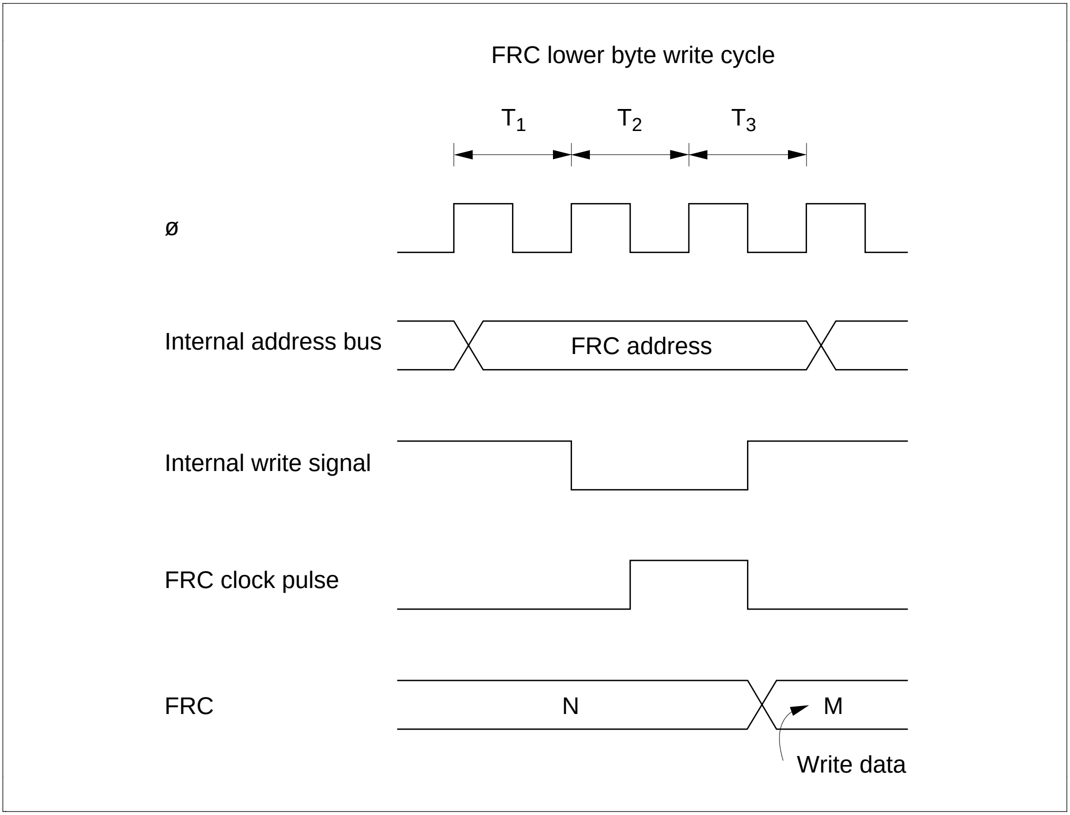


Figure 8-13 FRC Write-Increment Contention

(3) Contention between OCR Write and Compare-Match: If a compare-match occurs during the T_3 state of a write cycle to the lower byte of OCRA or OCRB, the write takes priority and the compare-match signal is inhibited.

Figure 8-14 shows this type of contention.

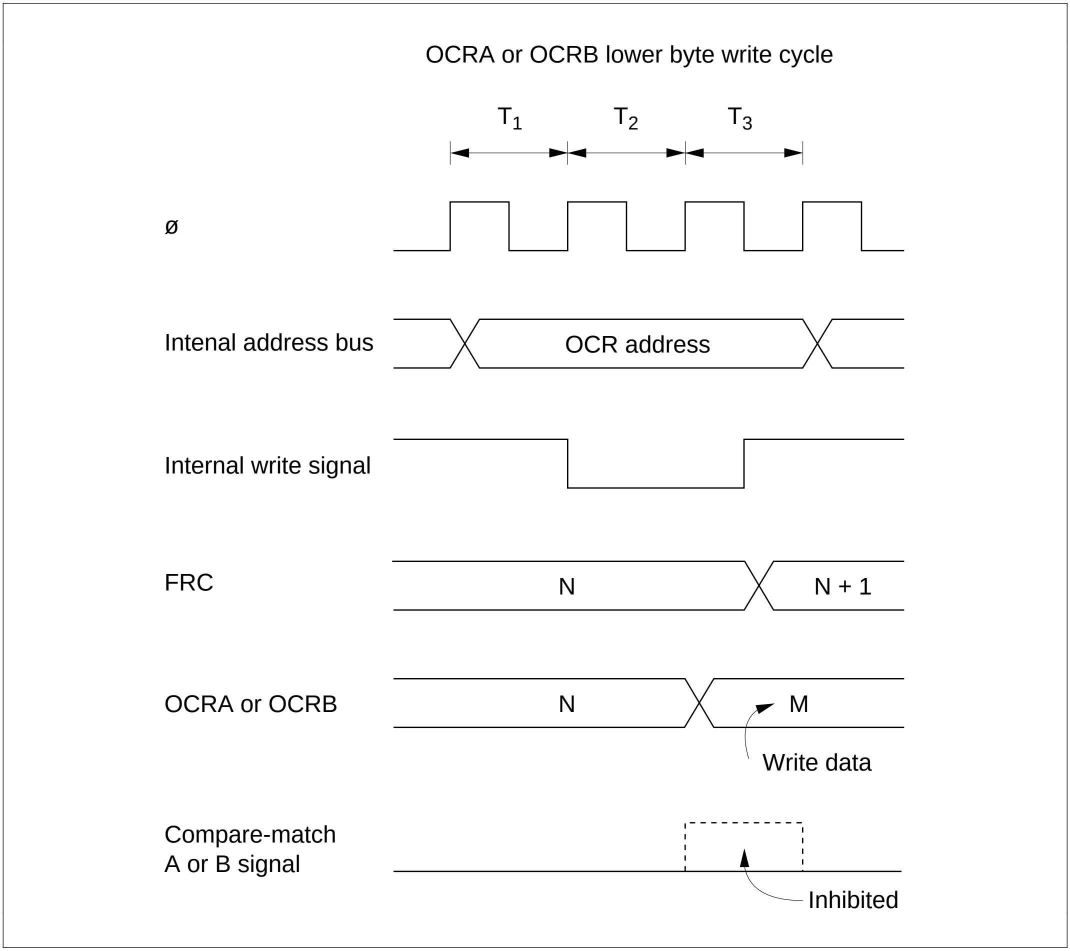


Figure 8-14 Contention between OCR Write and Compare-Match

(4) Increment Caused by Changing of Internal Clock Source: When an internal clock source is changed, the changeover may cause the FRC to increment. This depends on the time at which the clock select bits (CKS1 and CKS0) are rewritten, as shown in table 8-4.

The pulse that increments the FRC is generated at the falling edge of the internal clock source. If clock sources are changed when the old source is high and the new source is low, as in case No. 3 in table 8-4, the changeover generates a falling edge that triggers the FRC increment clock pulse.

Switching between an internal and external clock source can also cause the FRC to increment.

Table 8-4 Effect of Changing Internal Clock Sources

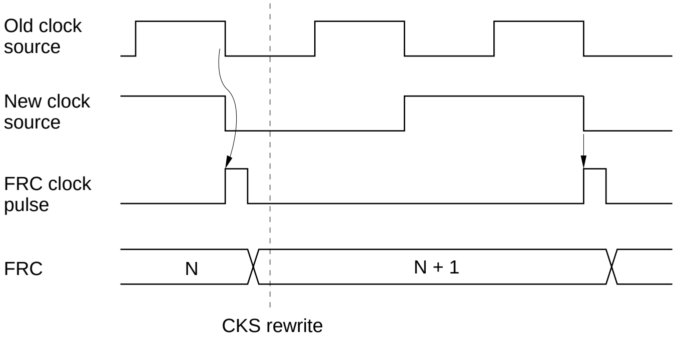
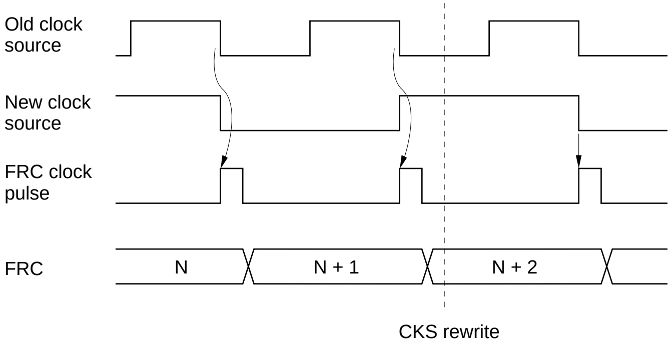
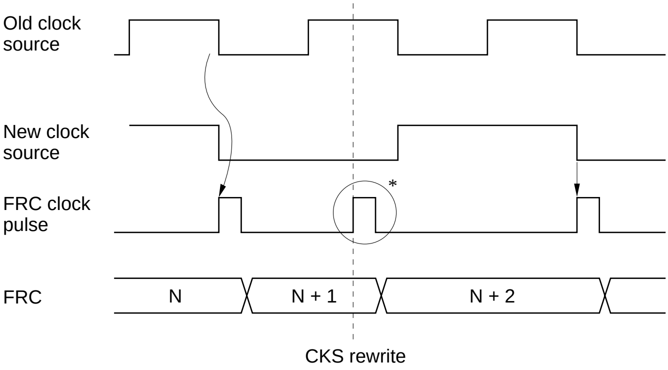
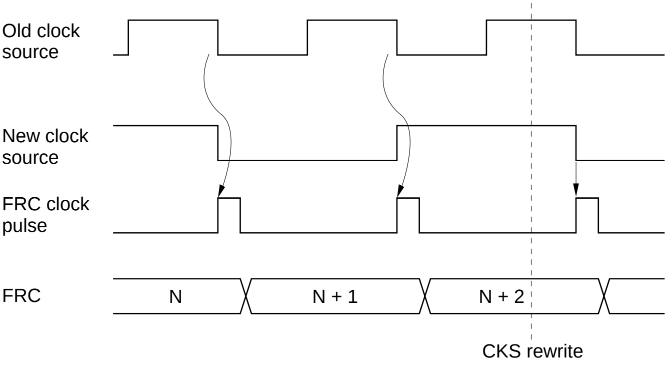
No.	Description	Timing Chart
1	Low → low: CKS1 and CKS0 are rewritten while both clock sources are low.	 Old clock source New clock source FRC clock pulse FRC N N + 1 CKS rewrite
2	Low → high: CKS1 and CKS0 are rewritten while old clock source is low and new clock source is high.	 Old clock source New clock source FRC clock pulse FRC N N + 1 N + 2 CKS rewrite

Table 8-4 Effect of Changing Internal Clock Sources (cont)

No.	Description	Timing Chart
3	High → low: CKS1 and CKS0 are rewritten while old clock source is high and new clock source is low.	 Old clock source New clock source FRC clock pulse * FRC N N + 1 N + 2 CKS rewrite
4	High → high: CKS1 and CKS0 are rewritten while both clock sources are high.	 Old clock source New clock source FRC clock pulse FRC N N + 1 N + 2 CKS rewrite

Note: * The switching of clock sources is regarded as a falling edge that increments the FRC.

Section 9 8-Bit Timers

9.1 Overview

The H8/3502 has an 8-bit timer module with two channels: timers 0 and 1. Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare-match events. One application of the 8-bit timer module is to generate a rectangular-wave output with an arbitrary duty factor.

9.1.1 Features

The features of the 8-bit timer module are listed below.

- Selection of seven clock sources for TMR0 and TMR1

  The counters can be driven by an internal clock signal (selection of six signals for TMR0 and TMR1) or an external clock input (enabling use as an external event counter).

- Selection of three ways to clear the counters

  The counters can be cleared on compare-match A or B, or by an external reset signal.

- Timer output controlled by two compare-match signals

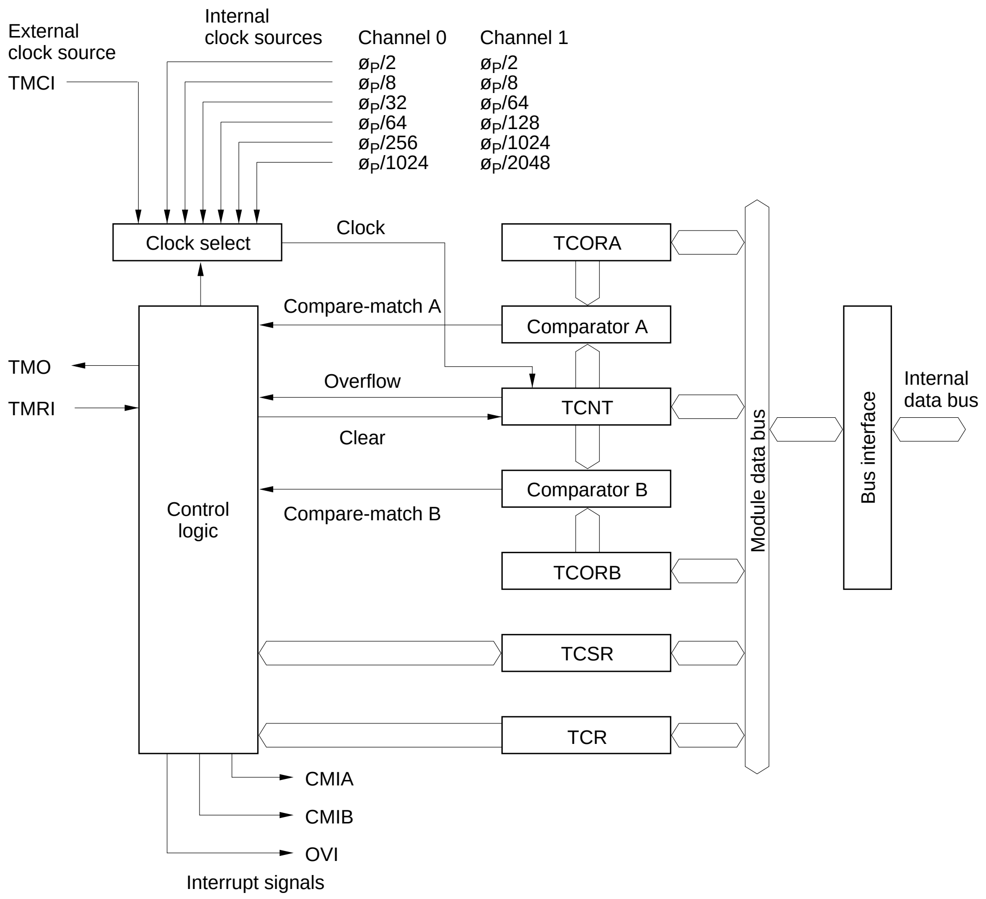
  The timer output signal in each channel is controlled by two independent compare-match signals, enabling the timer to generate output waveforms with an arbitrary duty factor. PWM mode can be selected, enabling PWM output of 0% to 100%.

- Three independent interrupts

  Compare-match A and B and overflow interrupts can be requested independently.

9.1.2 Block Diagram

Figure 9-1 shows a block diagram of one channel in the 8-bit timer module. The other channels are identical.



TCR: Timer control register (8 bits)
TCSR: Timer control status register (8 bits)
TCORA: Time constant register A (8 bits)
TCORB: Time constant register B (8 bits)
TCNT: Timer counter

Figure 9-1 Block Diagram of 8-Bit Timer (One Channel)

9.1.3 Input and Output Pins

Table 9-1 lists the input and output pins of the 8-bit timer.

Table 9-1 Input and Output Pins of 8-Bit Timer

Channel	Name	Abbreviation*	I/O	Function
0	Timer output	TMO_0	Output	Output controlled by compare-match
	Timer clock input	$TMCI_0$	Input	External clock source for the counter
	Timer reset input	$TMRI_0$	Input	External reset signal for the counter
1	Timer output	TMO_1	Output	Output controlled by compare-match
	Timer clock input	$TMCI_1$	Input	External clock source for the counter
	Timer reset input	$TMRI_1$	Input	External reset signal for the counter

Note: * The abbreviations TMO, TMCI, and TMRI are used in the text, omitting the channel number.

9.1.4 Register Configuration

Table 9-2 lists the registers of the 8-bit timer module. Each channel has an independent set of registers.

Table 9-2 8-Bit Timer Registers

			Initial	Address	
Name	Abbreviation	R/W	Value	TMR0	TMR1
Timer control register	TCR	R/W	H'00	H'FFC8	H'FFD0
Timer control/status register	TCSR	R/(W)*	H'00	H'FFC9	H'FFD1
Timer constant register A	TCORA	R/W	H'FF	H'FFCA	H'FFD2
Timer constant register B	TCORB	R/W	H'FF	H'FFCB	H'FFD3
Timer counter	TCNT	R/W	H'00	H'FFCC	H'FFD4
Serial timer control register	STCR	R/W	H'00	H'FFC3	H'FFC3

Note: * Software can write a 0 to clear bits 7 to 5, but cannot write a 1 in these bits.

9.2 Register Descriptions

9.2.1 Timer Counter (TCNT)—H'FFCC (TMR0), H'FFD4 (TMR1), H'FF9E (TMRX)

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Each timer counter (TCNT) is an 8-bit up-counter that increments on a pulse generated from the selected clock source. The clock source is selected by clock select bits 2 to 0 (CKS2 to CKS0) of the timer control register (TCR). The CPU can always read or write the timer counter.

The timer counter can be cleared by an external reset input or by an internal compare-match signal generated at a compare-match event. Counter clear bits 1 and 0 (CCLR1 and CCLR0) of the timer control register select the method of clearing.

When a timer counter overflows from H'FF to H'00, the overflow flag (OVF) in the timer control/status register (TCSR) is set to 1.

The timer counters are initialized to H'00 by a reset and in the standby modes.

9.2.2 Time Constant Registers A and B (TCORA and TCORB)—H'FFCA and H'FFCB (TMR0), H'FFD2 and H'FFD3 (TMR1), H'FF9C and H'FF9D (TMRX)

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCORA and TCORB are 8-bit readable/writable registers. The timer count is continually compared with the constants written in these registers. When a match is detected, the corresponding compare-match flag (CMFA or CMFB) is set in the timer control/status register (TCSR).

The timer output signal is controlled by these compare-match signals as specified by output select bits 3 to 0 (OS3 to OS0) in the timer control/status register (TCSR).

TCORA and TCORB are initialized to H'FF at a reset and in the standby modes.

9.2.3 Timer Control Register (TCR)—H'FFC8 (TMR0), H'FFD0 (TMR1), H'FF9A (TMRX)

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCR is an 8-bit readable/writable register that selects the clock source and the time at which the timer counter is cleared, and enables interrupts.

TCR is initialized to H'00 at a reset and in the standby modes.

For the timing, see section 9.3, Operation.

Bit 7—Compare-Match Interrupt Enable B (CMIEB): This bit selects whether to request compare-match interrupt B (CMIB) when compare-match flag B (CMFB) in the timer control/status register (TCSR) is set to 1.

Bit 7 CMIEB	Description	
0	Compare-match interrupt request B (CMIB) is disabled	(Initial value)
1	Compare-match interrupt request B (CMIB) is enabled	

Bit 6—Compare-Match Interrupt Enable A (CMIEA): This bit selects whether to request compare-match interrupt A (CMIA) when compare-match flag A (CMFA) in the timer control/status register (TCSR) is set to 1.

Bit 6 CMIEA	Description	
0	Compare-match interrupt request A (CMIA) is disabled	(Initial value)
1	Compare-match interrupt request A (CMIA) is enabled	

Bit 5—Timer Overflow Interrupt Enable (OVIE): This bit selects whether to request a timer overflow interrupt (OVI) when the overflow flag (OVF) in the timer control/status register (TCSR) is set to 1.

Bit 5 OVIE	Description	
0	The timer overflow interrupt request (OVI) is disabled	(Initial value)
1	The timer overflow interrupt request (OVI) is enabled	

Bits 4 and 3—Counter Clear 1 and 0 (CCLR1 and CCLR0): These bits select how the timer counter is cleared: by compare-match A or B or by an external reset input at the TMRI pin.

Bit 4 CCLR1	Bit 3 CCLR0	Description	
0	0	Not cleared	(Initial value)
0	1	Cleared on compare-match A	
1	0	Cleared on compare-match B	
1	1	Cleared on rising edge of external reset input signal	

Bits 2, 1, and 0—Clock Select (CKS2, CKS1, and CKS0): Together with the ICKS0 and ICKS1 bits in STCR, these bits select the internal or external clock source for the timer counter. For the external clock source they select whether to increment the count on the rising or falling edge of the external clock input (TMCI), or on both edges. For the internal clock sources the count is incremented on the falling edge of the clock input.

	TCR			**STCR**		
	Bit 2	**Bit 1**	**Bit 0**	**Bit 1**	**Bit 0**	
Channel	**CKS2**	**CKS1**	**CKS0**	**ICKS1**	**ICKS0**	**Description**
0	0	0	0	—	—	No clock source (timer stopped)
	0	0	1	—	0	ø$_P$/8 internal clock source, counted on the falling edge
	0	0	1	—	1	ø$_P$/2 internal clock source, counted on the falling edge
	0	1	0	—	0	ø$_P$/64 internal clock source, counted on the falling edge
	0	1	0	—	1	ø$_P$/32 internal clock source, counted on the falling edge
	0	1	1	—	0	ø$_P$/1024 internal clock source, counted on the falling edge
	0	1	1	—	1	ø$_P$/256 internal clock source, counted on the falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on the rising edge
	1	1	0	—	—	External clock source, counted on the falling edge
	1	1	1	—	—	External clock source, counted on both the rising and falling edges

	TCR			STCR		
Channel	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	Description
1	0	0	0	—	—	No clock source (timer stopped)
	0	0	1	0	—	$ø_P$/8 internal clock source, counted on the falling edge
	0	0	1	1	—	$ø_P$/2 internal clock source, counted on the falling edge
	0	1	0	0	—	$ø_P$/64 internal clock source, counted on the falling edge
	0	1	0	1	—	$ø_P$/128 internal clock source, counted on the falling edge
	0	1	1	0	—	$ø_P$/1024 internal clock source, counted on the falling edge
	0	1	1	1	—	$ø_P$/2048 internal clock source, counted on the falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on the rising edge
	1	1	0	—	—	External clock source, counted on the falling edge
	1	1	1	—	—	External clock source, counted on both the rising and falling edges

9.2.4 Timer Control/Status Register (TCSR)—H'FFC9 (TMR0), H'FFD1 (TMR1), H'FF9B (TMRX)

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	PWME	OS3	OS2	OS1	OS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/W	R/W	R/W	R/W	R/W

Note: * Software can write a 0 in bits 7 to 5 to clear the flags, but cannot write a 1 in these bits.

TCSR is an 8-bit readable and partially writable register that indicates compare-match and overflow status and selects the effect of compare-match events on the timer output signal.

TCSR is initialized to H'00 at a reset and in the standby modes.

Bit 7—Compare-Match Flag B (CMFB): This status flag is set to 1 when the timer count matches the time constant set in TCORB. CMFB must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 7 CMFB	Description	
0	To clear CMFB, the CPU must read CMFB after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when TCNT = TCORB	

Bit 6—Compare-Match Flag A (CMFA): This status flag is set to 1 when the timer count matches the time constant set in TCORA. CMFA must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 6 CMFA	Description	
0	To clear CMFA, the CPU must read CMFA after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when TCNT = TCORA	

Bit 5—Timer Overflow Flag (OVF): This status flag is set to 1 when the timer count overflows (changes from H'FF to H'00). OVF must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 5 OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when TCNT changes from H'FF to H'00	

Bit 4—PWM Mode Enable (PWME): This bit sets the timer output to PWM mode.

Bit 4 PWME	Description	
0	Normal timer mode	(Initial value)
1	PWM mode	

In PWM mode, bits CCLR1 and CCLR0 and bits OS3 to OS0 must be set so that the contents of TCORA determine the timer output period and the contents of TCORB determine the timer output duty cycle. The timer output pulse period, pulse width, and duty cycle are given by the following equations. If TCORA < TCORB, the output is saturated at a100% duty cycle.

(When TCORB ≤ TCORA)

Timer output pulse period = Selected internal clock period × (TCORA + 1)
Timer output pulse width = Selected internal clock period × TCORB
Timer output duty cycle = TCORB/(TCORA + 1)

	TCR		TCSR			
PWM Output Mode	**CCLR1**	**CCLR0**	**OS3**	**OS2**	**OS1**	**OS0**
Direct output (when the above timer pulse width is high)	0	1	0	1	1	0
Inverted output (when the above timer pulse width is low)	0	1	1	0	0	1

In PWM mode, a buffer register is inserted between TCORB and the module data bus, and the data written to TCORB is held in the buffer register until a TCORA compare-match occurs. This makes it easy to achieve PWM output with an undisturbed waveform. With the timer output specification made by bits OS3 to OS0, the priority of a change due to compare-match B is higher. Caution is required since the operation differs from that in normal timer mode.

Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0): These bits specify the effect of compare-match events on the timer output signal (TMO). Bits OS3 and OS2 control the effect of compare-match B on the output level. Bits OS1 and OS0 control the effect of compare-match A on the output level.

In normal timer mode, if compare-match A and B occur simultaneously, any conflict is resolved by giving highest priority to toggle, second-highest priority to 1 output, and third-highest priority to 0 output, as explained in item 9.6.4 in section 9.6, Application Notes.

After a reset, the timer output is 0 until the first compare-match event.

When all four output select bits (bits OS3 to OS0) are cleared to 0 the timer output signal is disabled.

Bit 3 OS3	Bit 2 OS2	Description	
0	0	No change when compare-match B occurs	(Initial value)
0	1	Output changes to 0 when compare-match B occurs	
1	0	Output changes to 1 when compare-match B occurs	
1	1	Output inverts (toggles) when compare-match B occurs	

Bit 1 OS1	Bit 0 OS0	Description	
0	0	No change when compare-match A occurs	(Initial value)
0	1	Output changes to 0 when compare-match A occurs	
1	0	Output changes to 1 when compare-match A occurs	
1	1	Output inverts (toggles) when compare-match A occurs	

9.2.5 Serial/Timer Control Register (STCR)

Bit	7	6	5	4	3	2	1	0
	(IICS)	(IICX1)	(IICX0)	(SYNCE)	(PWCKE)	(PWCKS)	ICKS1	ICKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

STCR is an 8-bit readable/writable register that selects the TCNT input clock source for the 8-bit timer.

STCR is initialized to H'00 by a reset.

Bits 7 to 5—I^{2}C Control (IICS, IICX1, IICX0): Reserved. Do not set these bits to 1.

Bit 4—Timer Connection Output Enable (SYNCE): Reserved. Do not set this bit to 1.

Bits 3 and 2—PWM Timer Control (PWCKE, PWCKS): Reserved. Do not set these bits to 1.

Bits 1 and 0—Internal Clock Select 1 and 0 (ICKS1 and ICKS0): These bits, together with bits CKS2 to CKS0 in TCR of the 8-bit timers, select the internal clock to be input to the timer counters (TCNT) in the 8-bit timers. For details, see section 9.2.3, Timer Control Register.

9.3 Operation

9.3.1 TCNT Incrementation Timing

The timer counter increments on a pulse generated once for each period of the clock source selected by bits CKS2 to CKS0 of the TCR.

Internal Clock: Internal clock sources are created from the system clock by a prescaler. The counter increments on an internal TCNT clock pulse generated from the falling edge of the prescaler output, as shown in figure 9-2. Bits CKS2 to CKS0 of the TCR can select one of six, or one of three, internal clocks.

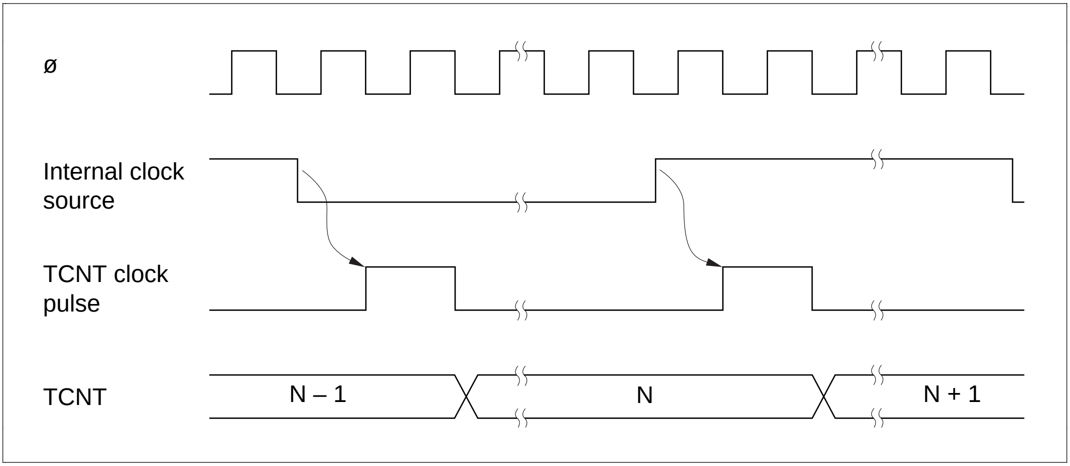


Figure 9-2 Count Timing for Internal Clock Input

External Clock: If external clock input (TMCI) is selected, the timer counter can increment on the rising edge, the falling edge, or both edges of the external clock signal. Figure 9-3 shows incrementation on both edges of the external clock signal.

The external clock pulse width must be at least 1.5 system clock periods for incrementation on a single edge, and at least 2.5 system clock periods for incrementation on both edges.

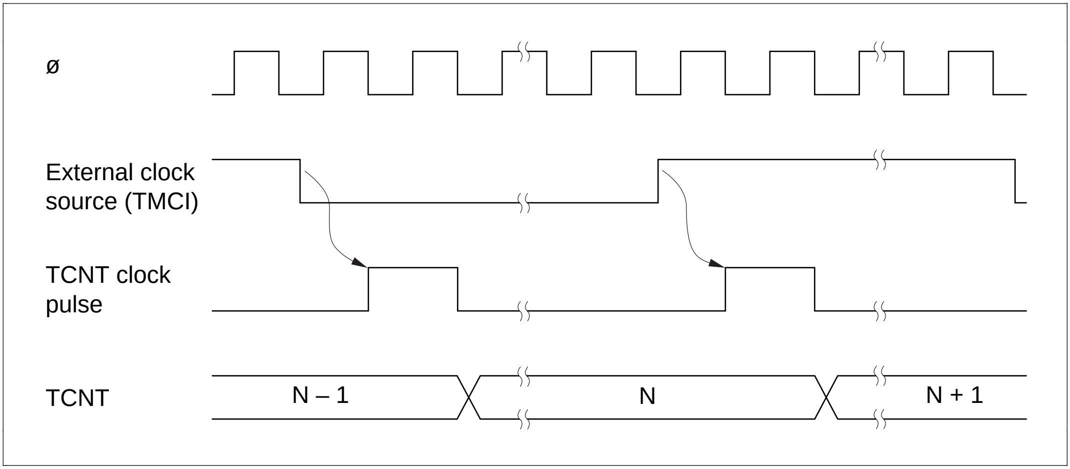


Figure 9-3 Count Timing for External Clock Input

9.3.2 Compare Match Timing

(1) Setting of Compare-Match Flags A and B (CMFA and CMFB): The compare-match flags are set to 1 by an internal compare-match signal generated when the timer count matches the time constant in TCNT or TCOR. The compare-match signal is generated at the last state in which the match is true, just before the timer counter increments to a new value.

Accordingly, when the timer count matches one of the time constants, the compare-match signal is not generated until the next period of the clock source. Figure 9-4 shows the timing of the setting of the compare-match flags.

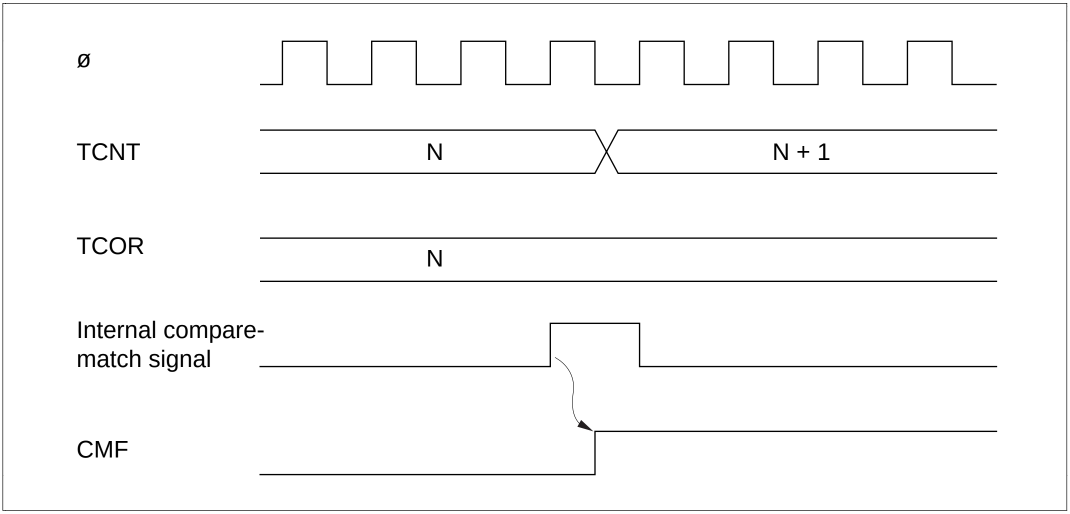


Figure 9-4 Setting of Compare-Match Flags

(2) Output Timing (Normal Timer Mode): When a compare-match event occurs, the timer output (TMO0 or TMO1) changes as specified by the output select bits (OS3 to OS0) in the TCSR. Depending on these bits, the output can remain the same, change to 0, change to 1, or toggle. If compare-match A and B occur simultaneously, the higher priority compare-match determines the output level. See item 9.6.4 in section 9.6, Application Notes, for details.

Figure 9-5 shows the timing when the output is set to toggle on compare-match A.

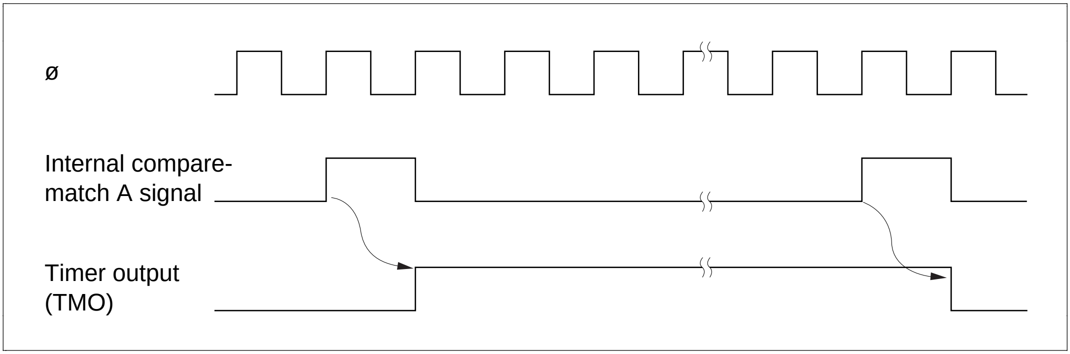


Figure 9-5 Timing of Timer Output

(3) Timing of Compare-Match Clear: Depending on the CCLR1 and CCLR0 bits in the TCR, the timer counter can be cleared when compare-match A or B occurs. Figure 9-6 shows the timing of this operation.

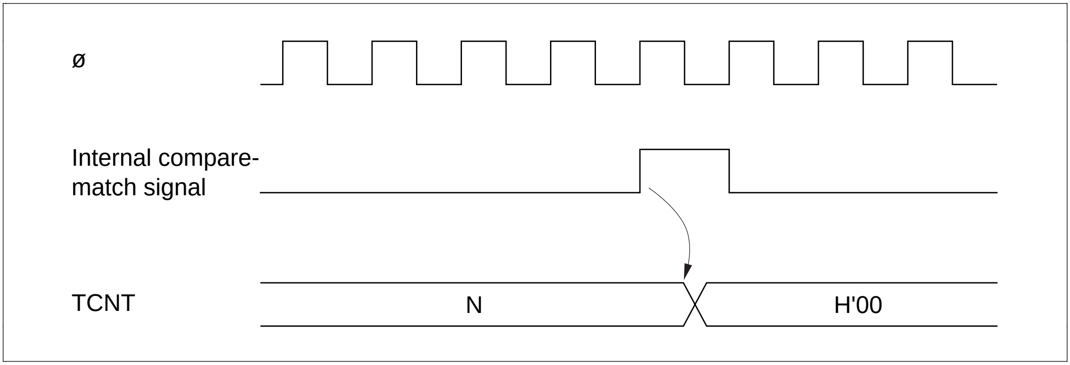


Figure 9-6 Timing of Compare-Match Clear

9.3.3 External Reset of TCNT

When the CCLR1 and CCLR0 bits in the TCR are both set to 1, the timer counter is cleared on the rising edge of an external reset input. Figure 9-7 shows the timing of this operation. The timer reset pulse width must be at least 1.5 system clock periods.

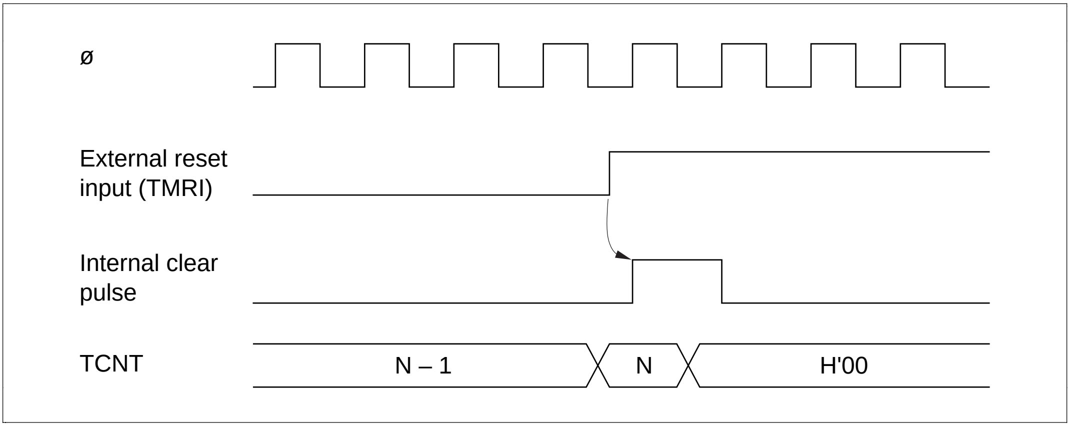


Figure 9-7 Timing of External Reset

9.3.4 Setting of TCSR Overflow Flag

(1) Setting of TCSR Overflow Flag (OVF): The overflow flag (OVF) is set to 1 when the timer count overflows (changes from H'FF to H'00). Figure 9-8 shows the timing of this operation.

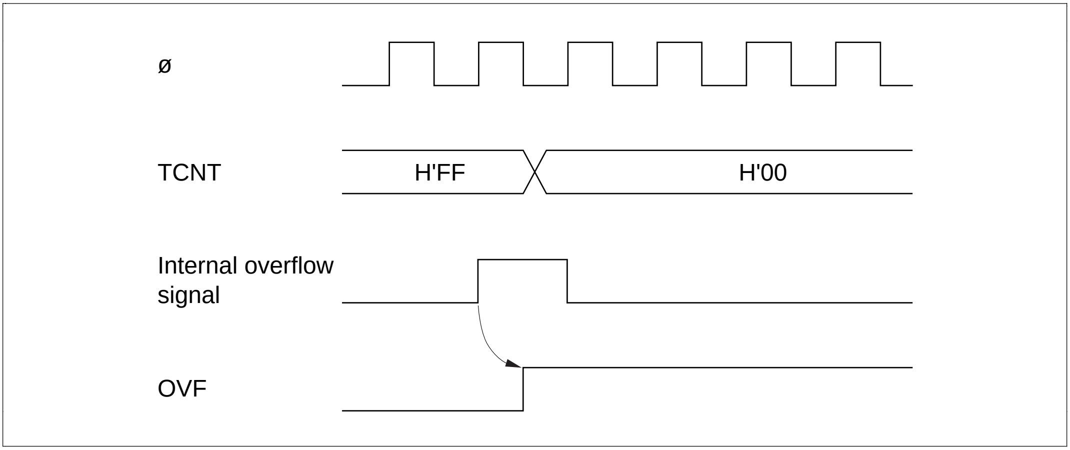


Figure 9-8 Setting of Overflow Flag

9.4 Interrupts

Each channel in the 8-bit timer can generate three types of interrupts: compare-match A and B (CMIA and CMIB), and overflow (OVI). Each interrupt is requested when the corresponding enable bits are set in the TCR and TCSR. Independent signals are sent to the interrupt controller for each interrupt. Table 9-3 lists information about these interrupts.

Table 9-3 8-Bit Timer Interrupts

Interrupt	Description	Priority
CMIA	Requested when CMFA and CMIEA are set	High
CMIB	Requested when CMFB and CMIEB are set	↕
OVI	Requested when OVF and OVIE are set	Low

9.5 Sample Application

In the example below, the 8-bit timer is used to generate a pulse output with a selected duty factor. The control bits are set as follows:

1. In the TCR, CCLR1 is cleared to 0 and CCLR0 is set to 1 so that the timer counter is cleared when its value matches the constant in TCORA.

2. In the TCSR, bits OS3 to OS0 are set to 0110, causing the output to change to 1 on compare-match A and to 0 on compare-match B.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.

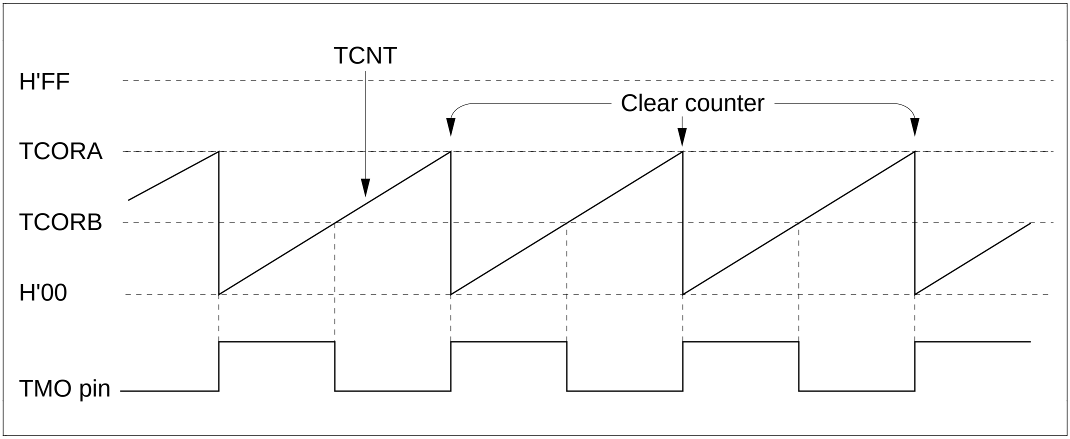


Figure 9-9 Example of Pulse Output

9.6 Application Notes

Application programmers should note that the following types of contention can occur in the 8-bit timer.

9.6.1 Contention between TCNT Write and Clear

If an internal counter clear signal is generated during the T_3 state of a write cycle to the timer counter, the clear signal takes priority and the write is not performed.

Figure 9-10 shows this type of contention.

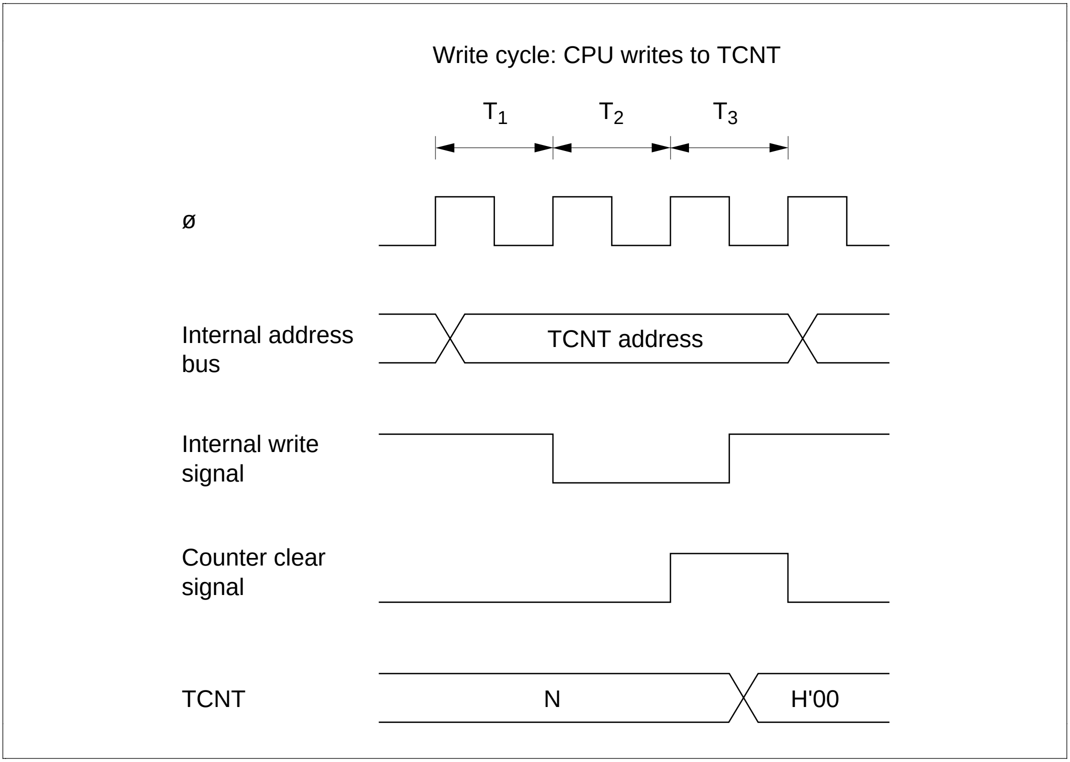


Figure 9-10 TCNT Write-Clear Contention

9.6.2 Contention between TCNT Write and Increment

If a timer counter increment pulse is generated during the T_3 state of a write cycle to the timer counter, the write takes priority and the timer counter is not incremented.

Figure 9-11 shows this type of contention.

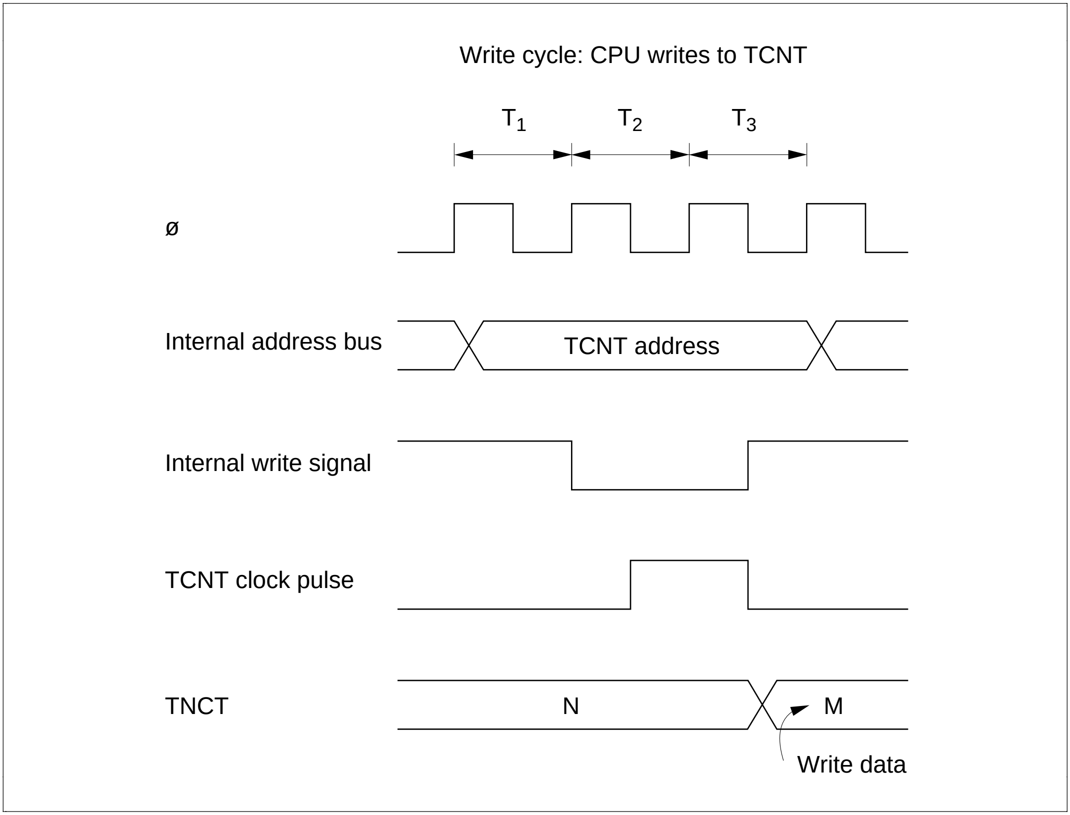


Figure 9-11 TCNT Write-Increment Contention

9.6.3 Contention between TCOR Write and Compare-Match

If a compare-match occurs during the T_3 state of a write cycle to TCOR, the write takes precedence and the compare-match signal is inhibited.

Figure 9-12 shows this type of contention (in normal timer mode).

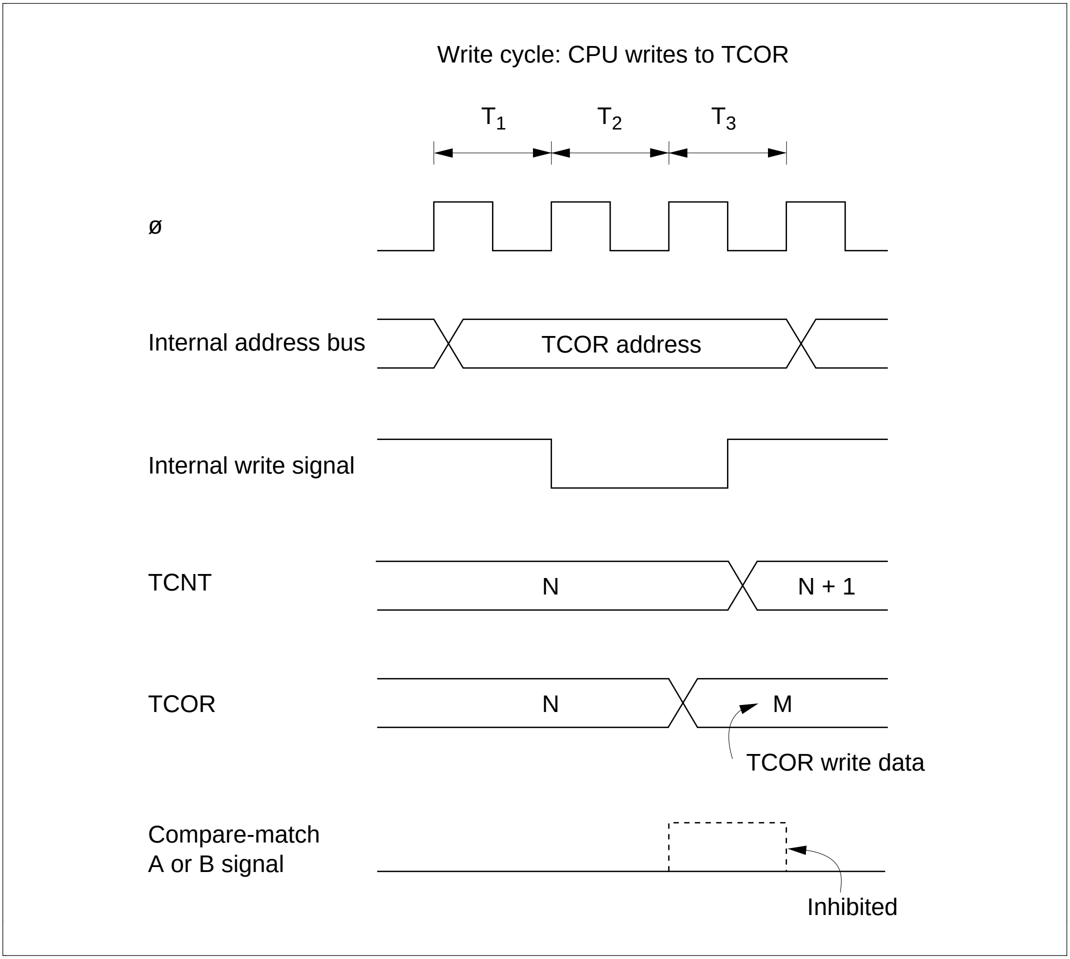


Figure 9-12 Contention between TCOR Write and Compare-Match

9.6.4 Contention between Compare-Match A and Compare-Match B

If identical time constants are written in TCORA and TCORB, causing compare-match A and B to occur simultaneously, any conflict between the output selections for compare-match A and B is resolved by following the priority order in table 9-4 (this applies to normal timer mode).

Table 9-4 Priority of Timer Output

Output Selection	Priority
Toggle	High
1 output	↑
0 output	
No change	Low

9.6.5 Incrementation Caused by Changing of Internal Clock Source

When an internal clock source is changed, the changeover may cause the timer counter to increment. This depends on the time at which the clock select bits (CKS1, CKS0) are rewritten, as shown in table 9-5.

The pulse that increments the timer counter is generated at the falling edge of the internal clock source signal. If clock sources are changed when the old source is high and the new source is low, as in case no. 3 in table 9-5, the changeover generates a falling edge that triggers the TCNT clock pulse and increments the timer counter.

Switching between an internal and external clock source can also cause the timer counter to increment.

Table 9-5 Effect of Changing Internal Clock Sources

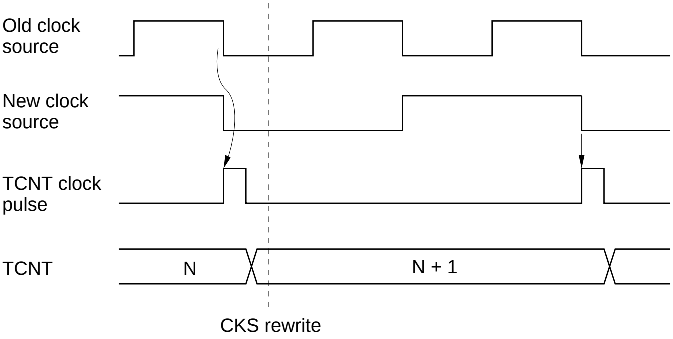
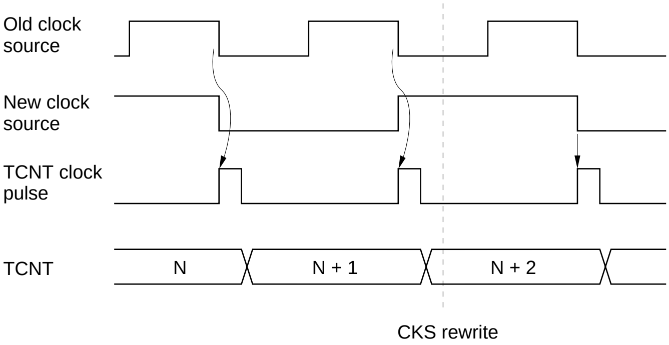
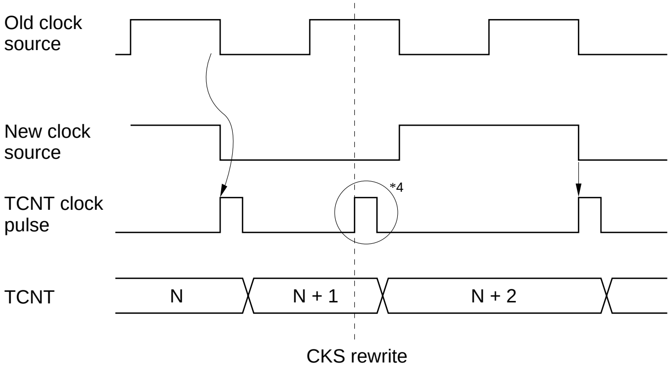
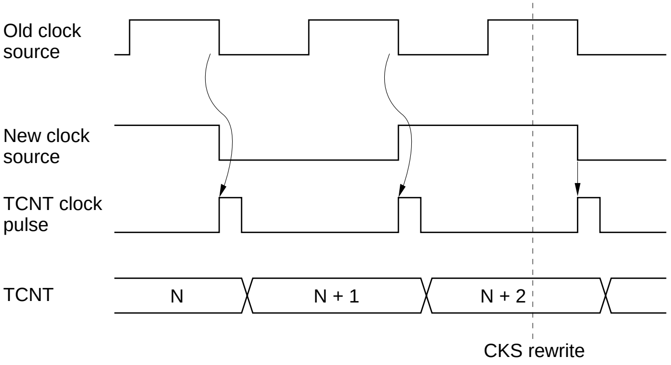
No.	Description	Timing
1	Low → low*1	 Old clock source New clock source TCNT clock pulse TCNT N N + 1 CKS rewrite
2	Low → high*2	 Old clock source New clock source TCNT clock pulse TCNT N N + 1 N + 2 CKS rewrite

Table 9-5 Effect of Changing Internal Clock Sources (cont)

No.	Description	Timing chart
3	High → low*3	 Old clock source New clock source TCNT clock pulse TCNT *4 N N + 1 N + 2 CKS rewrite
4	High → high	 Old clock source New clock source TCNT clock pulse TCNT N N + 1 N + 2 CKS rewrite

Notes: 1. Including a transition from low to the stopped state (CKS1 = 0, CKS0 = 0), or a transition from the stopped state to low.

2. Including a transition from the stopped state to high.

3. Including a transition from high to the stopped state.

4. The switching of clock sources is regarded as a falling edge that increments TCNT.

Section 10 Watchdog Timer

10.1 Overview

The H8/3502 has an on-chip watchdog timer (WDT) that can monitor system operation by resetting the CPU or generating a nonmaskable interrupt if a system crash allows the timer count to overflow.

When this watchdog function is not needed, the watchdog timer module can be used as an interval timer. In interval timer mode, it requests an OVF interrupt at each counter overflow.

10.1.1 Features

- Selection of eight clock sources
- Selection of two modes:
  - Watchdog timer mode
  - Interval timer mode
- Counter overflow generates an interrupt request or reset:
  - Reset or NMI request in watchdog timer mode
  - OVF interrupt request in interval timer mode

10.1.2 Block Diagram

Figure 10-1 is a block diagram of the watchdog timer.

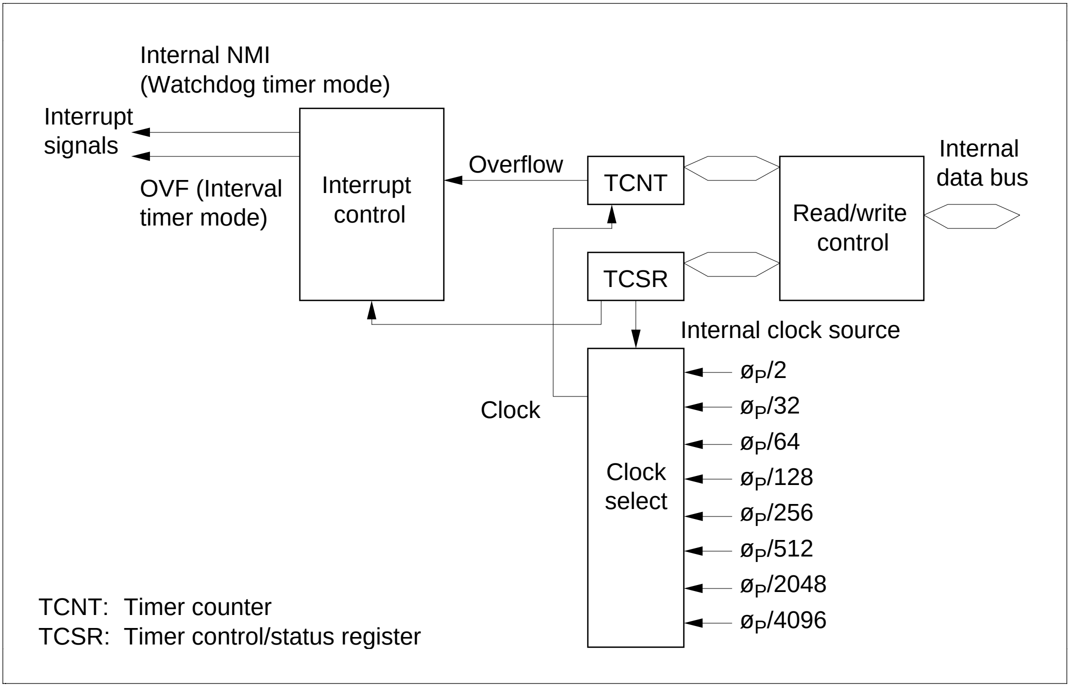


Figure 10-1 Block Diagram of Watchdog Timer

10.1.3 Register Configuration

Table 10-1 lists information on the watchdog timer registers.

Table 10-1 Register Configuration

				Addresses	
Name	**Abbreviation**	**R/W**	**Initial Value**	**Write**	**Read**
Timer control/status register	TCSR	R/(W)*	H'10	H'FFAA	H'FFAA
Timer counter	TCNT	R/W	H'00	H'FFAA	H'FFAB

Note: * Software can write a 0 in bit 7 to clear the flag, but cannot write 1.

10.2 Register Descriptions

10.2.1 Timer Counter (TCNT)

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCNT is an 8-bit readable/writable up-counter. When the timer enable bit (TME) in the timer control/status register (TCSR) is set to 1, the timer counter starts counting pulses of an internal clock source selected by clock select bits 2 to 0 (CKS2 to CKS0) in TCSR. When the count overflows (changes from H'FF to H'00), the overflow flag (OVF) in TCSR is set to 1.

TCNT is initialized to H'00 at a reset and when the TME bit is cleared to 0.

Note: TCNT is more difficult to write to than other registers. See section 10.2.3, Register Access, for details.

10.2.2 Timer Control/Status Register (TCSR)

Bit	7	6	5	4	3	2	1	0
	OVF	WT/$\overline{\text{IT}}$	TME	—	RST/$\overline{\text{NMI}}$	CKS2	CKS1	CKS0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/(W)*	R/W	R/W	—	R/W	R/W	R/W	R/W

Note: * Software can write a 0 in bit 7 to clear the flag, but cannot write a 1 in this bit. TCSR is more difficult to write to than other registers. See section 12.2.3, Register Access, for details.

TCSR is an 8-bit readable/writable register that selects the timer mode and clock source and performs other functions. (TCSR is write-protected by a password. See section 10.2.3, Register Access, for details.)

Bits 7 to 5 and bit 3 are initialized to 0 by a reset and in the standby modes. Bits 2 to 0 are initialized to 0 by a reset, but retain their values in the standby modes.

Bit 7—Overflow Flag (OVF): Indicates that the watchdog timer count has overflowed.

Bit 7 OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	Set to 1 when TCNT changes from H'FF to H'00	

Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$): Selects whether to operate in watchdog timer mode or interval timer mode. In interval timer mode, an OVF interrupt request is sent to the CPU when TCNT overflows. In watchdog timer mode, a reset or NMI interrupt is requested.

Bit 6 WT/$\overline{\text{IT}}$	Description	
0	Interval timer mode (OVF request)	(Initial value)
1	Watchdog timer mode (reset or NMI request)	

Bit 5—Timer Enable (TME): Enables or disables the timer.

Bit 5 TME	Description	
0	TCNT is initialized to H'00 and stopped	(Initial value)
1	TCNT runs and requests a reset or an interrupt when it overflows	

Bit 4—Reserved: This bit cannot be modified and is always read as 1.

Bit 3: Reset or NMI Select (RST/$\overline{\text{NMI}}$): Selects either an internal reset or the NMI function at watchdog timer overflow.

Bit 3 RST/$\overline{\text{NMI}}$	Description	
0	NMI function enabled	(Initial value)
1	Reset function enabled	

Bits 2—0: Clock Select (CKS2–CKS0): These bits select one of eight clock sources obtained by dividing the system clock (ø).

The overflow interval is the time from when the watchdog timer counter begins counting from H'00 until an overflow occurs. In interval timer mode, OVF interrupts are requested at this interval.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Clock Source	Overflow Interval (ø$_P$ = 10 MHz)	
0	0	0	ø$_P$/2	51.2 µs	(Initial value)
0	0	1	ø$_P$/32	819.2 µs	
0	1	0	ø$_P$/64	1.6 ms	
0	1	1	ø$_P$/128	3.3 ms	
1	0	0	ø$_P$/256	6.6 ms	
1	0	1	ø$_P$/512	13.1 ms	
1	1	0	ø$_P$/2048	52.4 ms	
1	1	1	ø$_P$/4096	104.9 ms	

10.2.3 Register Access

The watchdog timer's TCNT and TCSR registers are more difficult to write than other registers. The procedures for writing and reading these registers are given below.

Writing to TCNT and TCSR: Word access is required. Byte data transfer instructions cannot be used for write access.

The TCNT and TCSR registers have the same write address. The write data must be contained in the lower byte of a word written at this address. The upper byte must contain H'5A (password for TCNT) or H'A5 (password for TCSR). See figure 10-2. The result of the access depicted in figure 10-2 is to transfer the write data from the lower byte to TCNT or TCSR.

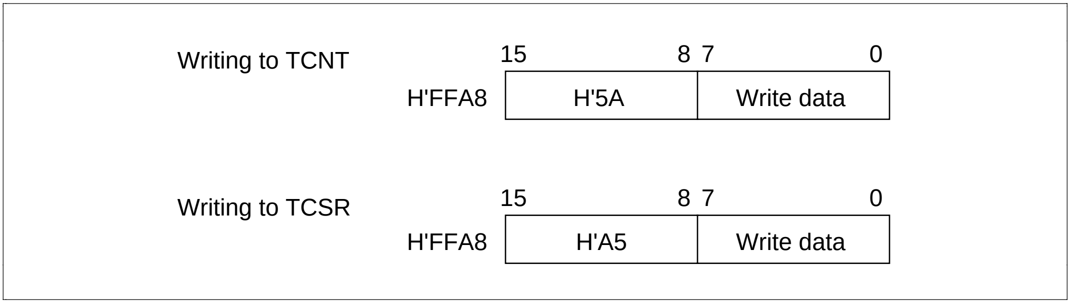


Figure 10-2 Writing to TCNT and TCSR

Reading TCNT and TCSR: The read addresses are H'FFA8 for TCSR and H'FFA9 for TCNT, as indicated in table 10-2.

These two registers are read like other registers. Byte access instructions can be used.

Table 10-2 Read Addresses of TCNT and TCSR

Read Address	Register
H'FFA8	TCSR
H'FFA9	TCNT

10.3 Operation

10.3.1 Watchdog Timer Mode

The watchdog timer function begins operating when software sets the WT/$\overline{\text{IT}}$ and TME bits to 1 in TCSR. Thereafter, software should periodically rewrite the contents of the timer counter (normally by writing H'00) to prevent the count from overflowing. If a program crash allows the timer count to overflow, the entire chip is reset for 518 system clocks (518 ø), or an NMI interrupt is requested. Figure 10-3 shows the operation.

NMI requests from the watchdog timer have the same vector as NMI requests from the $\overline{\text{NMI}}$ pin. Avoid simultaneous handling of watchdog timer NMI requests and NMI requests from pin $\overline{\text{NMI}}$.

A reset from the watchdog timer has the same vector as an external reset from the $\overline{\text{RES}}$ pin. The reset source can be determined by the XRST bit in SYSCR.

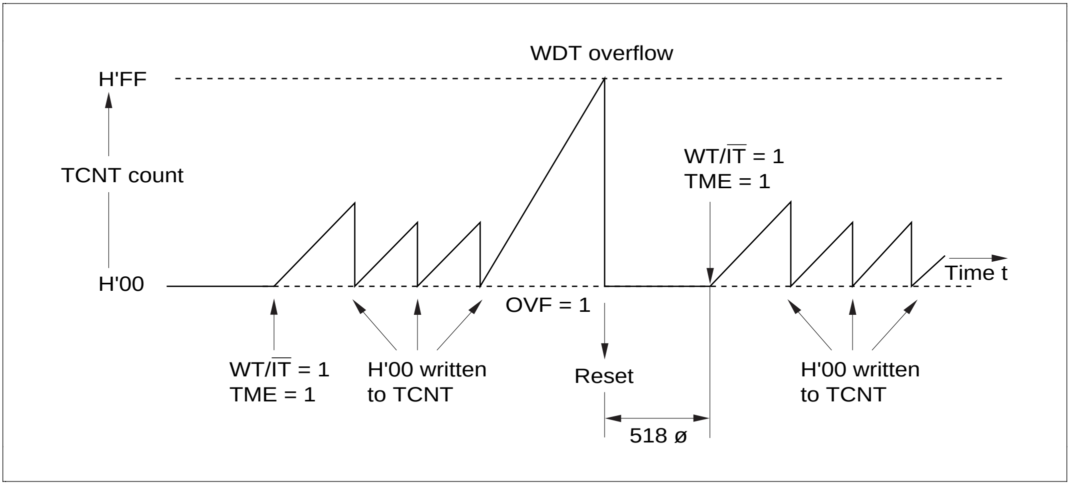


Figure 10-3 Operation in Watchdog Timer Mode

10.3.2 Interval Timer Mode

Interval timer operation begins when the WT/$\overline{\text{IT}}$ bit is cleared to 0 and the TME bit is set to 1.

In interval timer mode, an OVF request is generated each time the timer count overflows. This function can be used to generate OVF requests at regular intervals. See figure 10-4.

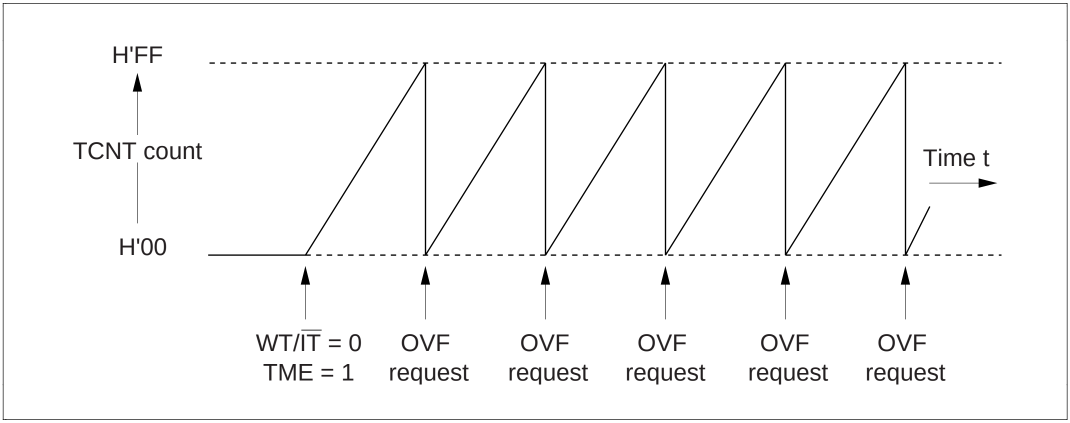


Figure 10-4 Operation in Interval Timer Mode

10.3.3 Setting the Overflow Flag

The OVF bit is set to 1 when the timer count overflows. Simultaneously, the WDT module requests an internal reset, NMI, or OVF interrupt. The timing is shown in figure 10-5.

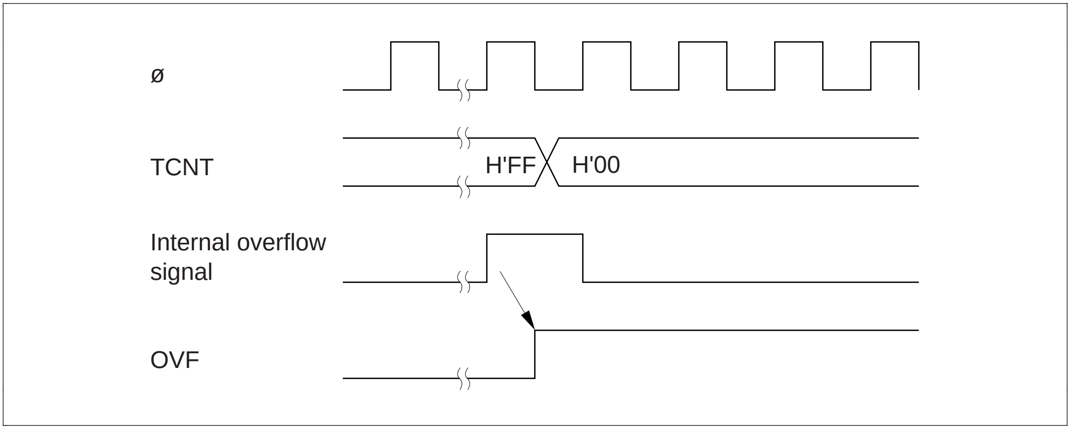


Figure 10-5 Setting the OVF Bit

10.4 Application Notes

10.4.1 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the T_3 state of a write cycle to the timer counter, the write takes priority and the timer counter is not incremented. See figure 10-6.

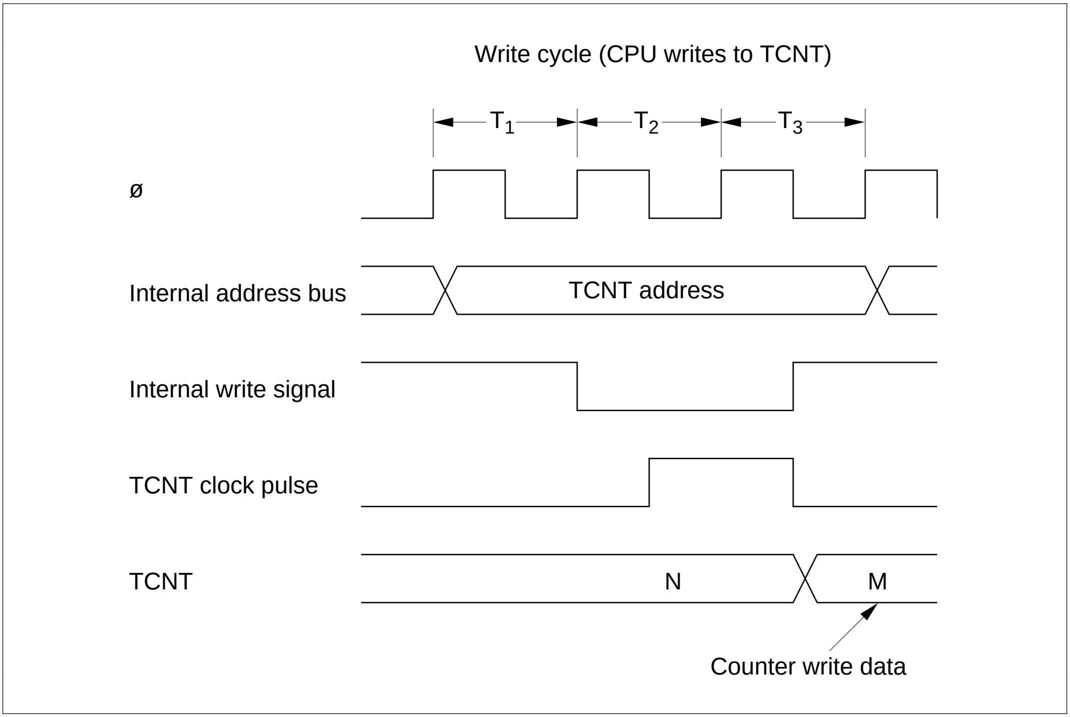


Figure 10-6 TCNT Write-Increment Contention

10.4.2 Changing the Clock Select Bits (CKS2 to CKS0)

Software should stop the watchdog timer (by clearing the TME bit to 0) before changing the value of the clock select bits. If the clock select bits are modified while the watchdog timer is running, the timer count may be incremented incorrectly.

10.4.3 Recovery from Software Standby Mode

TCSR bits, except bits 0–2, and the TCNT counter are reset when the chip recovers from software standby mode. Re-initialize the watchdog timer as necessary to resume normal operation.

Section 11 Serial Communication Interface

11.1 Overview

The H8/3502 includes two serial communication interface channels (SCI0 and SCI1) for transferring serial data to and from other chips. Either synchronous or asynchronous communication can be selected.

11.1.1 Features

The features of the on-chip serial communication interface are:

- Asynchronous mode

  The H8/3502 can communicate with a UART (Universal Asynchronous Receiver/Transmitter), ACIA (Asynchronous Communication Interface Adapter), or other chip that employs standard asynchronous serial communication. It also has a multiprocessor communication function for communication with other processors. Twelve data formats are available.

  - Data length: 7 or 8 bits
  - Stop bit length: 1 or 2 bits
  - Parity: Even, odd, or none
  - Multiprocessor bit: 1 or 0
  - Error detection: Parity, overrun, and framing errors
  - Break detection: When a framing error occurs, the break condition can be detected by reading the level of the RxD line directly.

- Synchronous mode

  The SCI can communicate with chips able to perform clocked synchronous data transfer.

  - Data length: 8 bits
  - Error detection: Overrun errors

- Full duplex communication

  The transmitting and receiving sections are independent, so each channel can transmit and receive simultaneously. Both the transmit and receive sections use double buffering, so continuous data transfer is possible in either direction.

- Built-in bit rate generator

  Any specified bit rate can be generated.

- Internal or external clock source

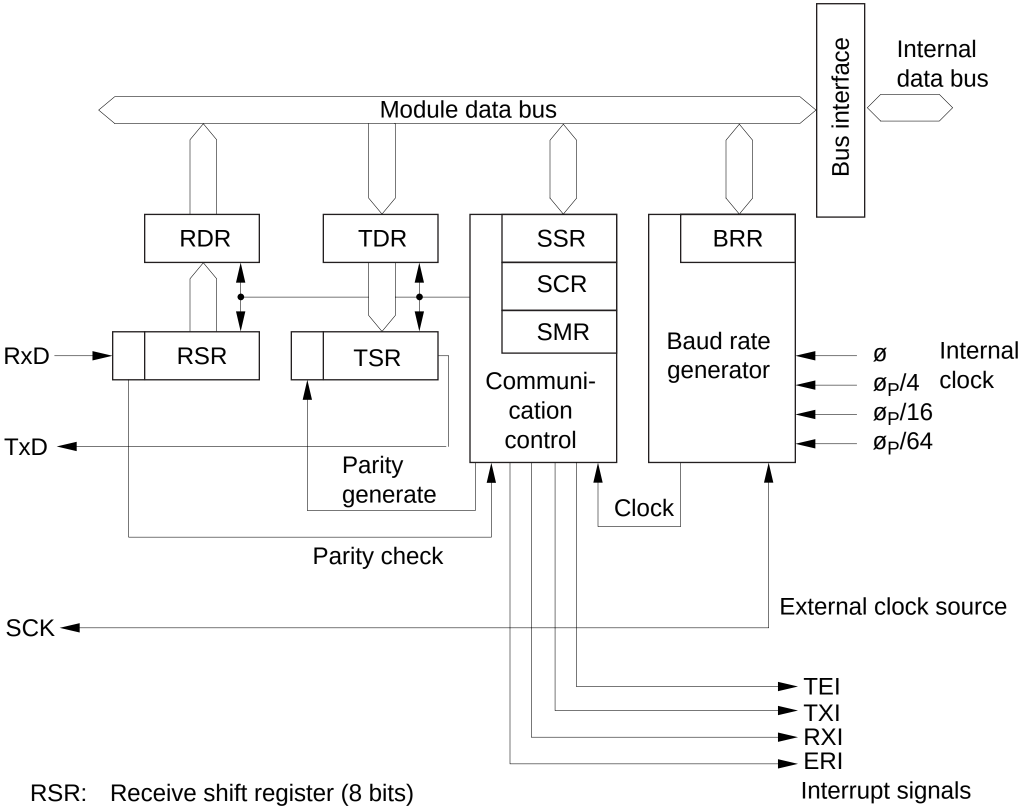
  The SCI can operate on an internal clock signal from the baud rate generator, or an external clock signal input at the SCK0 or SCK1 pin.

- Four interrupts

  TDR-empty, TSR-empty, receive-end, and receive-error interrupts are requested independently.

11.1.2 Block Diagram

Figure 11-1 shows a block diagram of one serial communication interface channel.



RSR:	Receive shift register (8 bits)
RDR:	Receive data register (8 bits)
TSR:	Transmit shift register (8 bits)
TDR:	Transmit data register (8 bits)
SMR:	Serial mode register (8 bits)
SCR:	Serial control register (8 bits)
SSR:	Serial status register (8 bits)
BRR:	Bit rate register (8 bits)

Figure 11-1 Block Diagram of Serial Communication Interface

11.1.3 Input and Output Pins

Table 11-1 lists the input and output pins used by the SCI module.

Table 11-1 SCI Input/Output Pins

Channel	Name	Abbr.	I/O	Function
0	Serial clock input/output	SCK_0	Input/output	Serial clock input and output
	Receive data input	RxD_0	Input	Receive data input
	Transmit data output	TxD_0	Output	Transmit data output
1	Serial clock input/output	SCK_1	Input/output	Serial clock input and output
	Receive data input	RxD_1	Input	Receive data input
	Transmit data output	TxD_1	Output	Transmit data output

Note: In this manual, the channel subscript has been deleted, and only SCK, RxD, and TxD are used.

11.1.4 Register Configuration

Table 11-2 lists the SCI registers. These registers specify the operating mode (synchronous or asynchronous), data format and bit rate, and control the transmit and receive sections.

Table 11-2 SCI Registers

Channel	Name	Abbr.	R/W	Value	Address
0	Receive shift register	RSR	—	—	—
	Receive data register	RDR	R	H'00	H'FFDD
	Transmit shift register	TSR	—	—	—
	Transmit data register	TDR	R/W	H'FF	H'FFDB
	Serial mode register	SMR	R/W	H'00	H'FFD8
	Serial control register	SCR	R/W	H'00	H'FFDA
	Serial status register	SSR	R/(W)*	H'84	H'FFDC
	Bit rate register	BRR	R/W	H'FF	H'FFD9
	Serial communication mode register	SCMR	R/W	H'F2	H'FFDE
1	Receive shift register	RSR	—	—	—
	Receive data register	RDR	R	H'00	H'FFE5
	Transmit shift register	TSR	—	—	—
	Transmit data register	TDR	R/W	H'FF	H'FFE3
	Serial mode register	SMR	R/W	H'00	H'FFE0
	Serial control register	SCR	R/W	H'00	H'FFE2
	Serial status register	SSR	R/(W)*	H'84	H'FFE4
	Bit rate register	BRR	R/W	H'FF	H'FFE1

Note: * Software can write a 0 to clear the flags in bits 7 to 3, but cannot write 1 in these bits.

11.2 Register Descriptions

11.2.1 Receive Shift Register (RSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

RSR is a shift register that converts incoming serial data to parallel data. When one data character has been received, it is transferred to the receive data register (RDR).

The CPU cannot read or write RSR directly.

11.2.2 Receive Data Register (RDR)

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

RDR stores received data. As each character is received, it is transferred from RSR to RDR, enabling RSR to receive the next character. This double-buffering allows the SCI to receive data continuously.

RDR is a read-only register. RDR is initialized to H'00 by a reset and in the standby modes.

11.2.3 Transmit Shift Register (TSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

TSR is a shift register that converts parallel data to serial transmit data. When transmission of one character is completed, the next character is moved from the transmit data register (TDR) to TSR and transmission of that character begins. If the TDRE bit is still set to 1, however, nothing is transferred to TSR.

The CPU cannot read or write TSR directly.

11.2.4 Transmit Data Register (TDR)

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TDR is an 8-bit readable/writable register that holds the next data to be transmitted. When TSR becomes empty, the data written in TDR is transferred to TSR. Continuous data transmission is possible by writing the next data in TDR while the current data is being transmitted from TSR.

TDR is initialized to H'FF by a reset and in the standby modes.

11.2.5 Serial Mode Register (SMR)

Bit	7	6	5	4	3	2	1	0
	$C/\overline{A}$	CHR	PE	$O/\overline{E}$	STOP	MP	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SMR is an 8-bit readable/writable register that controls the communication format and selects the clock source of the on-chip baud rate generator. It is initialized to H'00 by a reset and in the standby modes. For further information on the SMR settings and communication formats, see tables 11-5 and 11-7 in section 11.3, Operation.

Bit 7—Communication Mode ($C/\overline{A}$): This bit selects asynchronous or synchronous communication mode.

Bit 7 $C/\overline{A}$	Description	
0	Asynchronous communication	(Initial value)
1	Synchronous communication	

Bit 6—Character Length (CHR): This bit selects the character length in asynchronous mode. It is ignored in synchronous mode.

Bit 6 CHR	Description	
0	8 bits per character	(Initial value)
1	7 bits per character (Bits 6 to 0 of TDR and RDR are used for transmitting and receiving, respectively)	

Bit 5—Parity Enable (PE): This bit selects whether to add and check for a parity bit in asynchronous mode. It is ignored in synchronous mode, and when a multiprocessor format is used.

Bit 5 PE	Description	
0	Transmit: No parity bit is added	(Initial value)
	Receive: Parity is not checked	
1	Transmit: A parity bit is added	
	Receive: Parity is checked	

Bit 4—Parity Mode (O/$\overline{\text{E}}$): In asynchronous mode, when parity is enabled (PE = 1), this bit selects even or odd parity.

Even parity means that a parity bit is added to the data bits for each character to make the total number of 1’s even. Odd parity means that the total number of 1’s is made odd.

This bit is ignored when PE = 0, or when a multiprocessor format is used. It is also ignored in synchronous mode.

Bit 4 O/$\overline{\text{E}}$	Description	
0	Even parity	(Initial value)
1	Odd parity	

Bit 3—Stop Bit Length (STOP): This bit selects the number of stop bits. It is ignored in synchronous mode, and when a multiprocessor format is used.

Bit 3 STOP	Description
0	One stop bit (Initial value) Transmit: One stop bit is added Receive: One stop bit is checked to detect framing errors
1	Two stop bits Transmit: Two stop bits are added Receive: The first stop bit is checked to detect framing errors If the second stop bit is a space (0), it is regarded as the next start bit.

Bit 2—Multiprocessor Mode (MP): This bit selects the multiprocessor format. When multiprocessor format is selected, the parity settings of the parity enable bit (PE) and parity mode bit (O/$\overline{\text{E}}$) are ignored. The MP bit is valid only in asynchronous mode, and is ignored in synchronous mode.

Bit 2 MP	Description
0	Multiprocessor communication function is disabled (Initial value)
1	Multiprocessor communication function is enabled

Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0): These bits select the clock source of the on-chip baud rate generator.

Bit 1 CKS1	Bit 0 CKS0	Description
0	0	ø clock (Initial value)
0	1	ø$_{\text{P}}$/4 clock
1	0	ø$_{\text{P}}$/16 clock
1	1	ø$_{\text{P}}$/64 clock

11.2.6 Serial Control Register (SCR)

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCR is an 8-bit readable/writable register that enables or disables various SCI functions.
It is initialized to H'00 by a reset and in the standby modes.

Bit 7—Transmit Interrupt Enable (TIE): This bit enables or disables the TDR-empty interrupt (TXI) requested when the transmit data register empty (TDRE) bit in the serial status register (SSR) is set to 1.

Bit 7 TIE	Description	
0	The TDR-empty interrupt request (TXI) is disabled	(Initial value)
1	The TDR-empty interrupt request (TXI) is enabled	

Bit 6—Receive Interrupt Enable (RIE): This bit enables or disables the receive-end interrupt (RXI) requested when the receive data register full (RDRF) bit in the serial status register (SSR) is set to 1, and the receive error interrupt (ERI) requested when the overrun error (ORER), framing error (FER), or parity error (PER) bit in the serial status register (SSR) is set to 1.

Bit 6 RIE	Description	
0	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are disabled	(Initial value)
1	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are enabled	

Bit 5—Transmit Enable (TE): This bit enables or disables the transmit function. When the transmit function is enabled, the TxD pin is automatically used for output. When the transmit function is disabled, the TxD pin can be used as a general-purpose I/O port.

Bit 5 TE	Description	
0	The transmit function is disabled The TxD pin can be used for general-purpose I/O	(Initial value)
1	The transmit function is enabled The TxD pin is used for output	

Bit 4—Receive Enable (RE): This bit enables or disables the receive function. When the receive function is enabled, the RxD pin is automatically used for input. When the receive function is disabled, the RxD pin is available as a general-purpose I/O port.

Bit 4 RE	Description	
0	The receive function is disabled The RxD pin can be used for general-purpose I/O	(Initial value)
1	The receive function is enabled The RxD pin is used for input	

Bit 3—Multiprocessor Interrupt Enable (MPIE): When serial data is received in a multiprocessor format, this bit enables or disables the receive-end interrupt (RXI) and receive-error interrupt (ERI) until data with the multiprocessor bit set to 1 is received. It also enables or disables the transfer of receive data from RSR to RDR, and enables or disables setting of the RDRF, FER, PER, and ORER bits in the serial status register (SSR).

The MPIE bit is ignored when the MP bit is cleared to 0, and in synchronous mode.

Clearing the MPIE bit to 0 disables the multiprocessor receive interrupt function. In this condition data is received regardless of the value of the multiprocessor bit in the receive data.

Setting the MPIE bit to 1 enables the multiprocessor receive interrupt function. In this condition, if the multiprocessor bit in the receive data is 0, the receive-end interrupt (RXI) and receive-error interrupt (ERI) are disabled, the receive data is not transferred from RSR to RDR, and the RDRF, FER, PER, and ORER bits in the serial status register (SSR) are not set. If the multiprocessor bit is 1, however, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0, the receive data is transferred from RSR to RDR, the FER, PER, and ORER bits can be set, and the receive-end and receive-error interrupts are enabled.

Bit 3 MPIE	Description	
0	The multiprocessor receive interrupt function is disabled (Normal receive operation)	(Initial value)
1	The multiprocessor receive interrupt function is enabled. During the interval before data with the multiprocessor bit set to 1 is received, the receive interrupt request (RXI) and receive-error interrupt request (ERI) are disabled, the RDRF, FER, PER, and ORER bits are not set in the serial status register (SSR), and no data is transferred from the RSR to the RDR. The MPIE bit is cleared at the following times: (1) When 0 is written in MPIE (2) When data with the multiprocessor bit set to 1 is received	

Bit 2—Transmit-End Interrupt Enable (TEIE): This bit enables or disables the TSR-empty interrupt (TEI) requested when the transmit-end bit (TEND) in the serial status register (SSR) is set to 1.

Bit 2 TEIE	Description	
0	The TSR-empty interrupt request (TEI) is disabled	(Initial value)
1	The TSR-empty interrupt request (TEI) is enabled	

Bit 1—Clock Enable 1 (CKE1): This bit selects the internal or external clock source for the baud rate generator. When the external clock source is selected, the SCK pin is automatically used for input of the external clock signal.

Bit 1 CKE1	Description	
0	Internal clock source When $C/\overline{A}$ = 1, the serial clock signal is output at the SCK pin When $C/\overline{A}$ = 0, output depends on the CKE0 bit	(Initial value)
1	External clock source The SCK pin is used for input	

Bit 0—Clock Enable 0 (CKE0): When an internal clock source is used in asynchronous mode, this bit enables or disables serial clock output at the SCK pin.

This bit is ignored when the external clock is selected, or when synchronous mode is selected.

For further information on the communication format and clock source selection, see table 11-6 in section 11.3, Operation.

Bit 0 CKE0	Description	
0	The SCK pin is not used by the SCI (and is available as a general-purpose I/O port)	(Initial value)
1	The SCK pin is used for serial clock output	

11.2.7 Serial Status Register (SSR)

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: * Software can write a 0 in bits 7 to 3 to clear the flags, but cannot write a 1 in these bits.

SSR is an 8-bit register that indicates transmit and receive status. It is initialized to H'84 by a reset and in the standby modes.

Bit 7—Transmit Data Register Empty (TDRE): This bit indicates when transmit data can safely be written in TDR.

Bit 7 TDRE	Description	
0	To clear TDRE, the CPU must read TDRE after it has been set to 1, then write a 0 in this bit	
1	This bit is set to 1 at the following times: (1) When TDR contents are transferred to TSR (2) When the TE bit in SCR is cleared to 0	(Initial value)

Bit 6—Receive Data Register Full (RDRF): This bit indicates when one character has been received and transferred to the RDR.

Bit 6 RDRF	Description	
0	To clear RDRF, the CPU must read RDRF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when one character is received without error and transferred from RSR to RDR	

Bit 5—Overrun Error (ORER): This bit indicates an overrun error during reception.

Bit 5		
ORER	**Description**	
0	To clear ORER, the CPU must read ORER after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 if reception of the next character ends while the receive data register is still full (RDRF = 1)	

Bit 4—Framing Error (FER): This bit indicates a framing error during data reception in asynchronous mode. It has no meaning in synchronous mode.

Bit 4		
FER	**Description**	
0	To clear FER, the CPU must read FER after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 if a framing error occurs (stop bit = 0)	

Bit 3—Parity Error (PER): This bit indicates a parity error during data reception in asynchronous mode, when a communication format with parity bits is used.

This bit has no meaning in synchronous mode, or when a communication format without parity bits is used.

Bit 3		
PER	**Description**	
0	To clear PER, the CPU must read PER after it has been set to 1, then write a 0 in this bit	(Initial value)
1	This bit is set to 1 when a parity error occurs (the parity of the received data does not match the parity selected by the O/$\overline{\text{E}}$ bit in SMR)	

Bit 2—Transmit End (TEND): This bit indicates that the serial communication interface has stopped transmitting because there was no valid data in TDR when the last bit of the current character was transmitted. The TEND bit is also set to 1 when the TE bit in the serial control register (SCR) is cleared to 0.

The TEND bit is a read-only bit and cannot be modified directly. To use the TEI interrupt, first start transmitting data, which clears TEND to 0, then set TEIE to 1.

Bit 2 **TEND**	**Description**	
0	To clear TEND, the CPU must read TDRE after TDRE has been set to 1, then write a 0 in TDRE	(Initial value)
1	This bit is set to 1 when: (1) TE = 0 (2) TDRE = 1 at the end of transmission of a character	

Bit 1—Multiprocessor Bit (MPB): Stores the value of the multiprocessor bit in data received in a multiprocessor format in asynchronous communication mode. This bit retains its previous value in synchronous mode, when a multiprocessor format is not used, or when the RE bit is cleared to 0 even if a multiprocessor format is used.

MPB can be read but not written.

Bit 1 **MPB**	**Description**	
0	Multiprocessor bit = 0 in receive data	(Initial value)
1	Multiprocessor bit = 1 in receive data	

Bit 0—Multiprocessor Bit Transfer (MPBT): Stores the value of the multiprocessor bit inserted in transmit data when a multiprocessor format is used in asynchronous communication mode. The MPBT bit is double-buffered in the same way as TSR and TDR. The MPBT bit has no effect in synchronous mode, or when a multiprocessor format is not used.

Bit 0 **MPBT**	**Description**	
0	Multiprocessor bit = 0 in transmit data	(Initial value)
1	Multiprocessor bit = 1 in transmit data	

11.2.8 Bit Rate Register (BRR)

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BRR is an 8-bit register that, together with the CKS1 and CKS0 bits in SMR, determines the bit rate output by the baud rate generator.

BRR is initialized to H'FF by a reset and in the standby modes.

Tables 11-3 and 11-4 show examples of BRR settings.

Table 11-3 Examples of BRR Settings in Asynchronous Mode (When $ø_P = ø$)

	ø Frequency (MHz)											
	2			2.097152			2.4576			3		
Bit Rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
												+0.50
												+0.16
300	0	207	+0.16	0	217	+0.21	0	255				+0.16
600	0	103	+0.16	0	108	+0.21	0	127	0	0	155	+0.16
1200	0	51	+0.16	0	54	–0.70	0	63	0	0	77	+0.16
2400	0	25	+0.16	0	26	+1.14	0	31	0	0	38	+0.16
4800	0	12	+0.16	0	13	–2.48	0	15	0	0	19	–2.34
9600	—	—	—	0	6	–2.48	0	7	0	0	9	–2.34
19200	—	—	—	—	—	—	0	3	0	0	4	–2.34
31250	0	1	0	—	—	—	—	—	—	0	2	0
38400	—	—	—	—	—	—	0	1	0	—	—	—

Table 11-3 Examples of BRR Settings in Asynchronous Mode (When $ø_P = ø$) (cont)

	ø Frequency (MHz)											
	3.6864			4			4.9152			5		
Bit Rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
												–0.25
												+0.16
												+0.16
600	0	191	0	0	207	+0.16	0	255				+0.16
1200	0	95	0	0	103	+0.16	0	127	0	0	129	+0.16
2400	0	47	0	0	51	+0.16	0	63	0	0	64	+0.16
4800	0	23	0	0	25	+0.16	0	31	0	0	32	–1.36
9600	0	11	0	0	12	+0.16	0	15	0	0	15	+1.73
19200	0	5	0	—	—	—	0	7	0	0	7	+1.73
31250	—	—	—	0	3	0	0	4	–1.70	0	4	0
38400	0	2	0	—	—	—	0	3	0	0	3	+1.73

Table 11-3 Examples of BRR Settings in Asynchronous Mode (When $ø_P = ø$) (cont)

	ø Frequency (MHz)											
	6			6.144			7.3728			8		
Bit Rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
												+0.03
												+0.16
												+0.16
												+0.16
1200	0	155	+0.16	0	159	0	0	191	0	0	207	+0.16
2400	0	77	+0.16	0	79	0	0	95	0	0	103	+0.16
4800	0	38	+0.16	0	39	0	0	47	0	0	51	+0.16
9600	0	19	–2.34	0	19	0	0	23	0	0	25	+0.16
19200	0	9	–2.34	0	4	0	0	11	0	0	12	+0.16
31250	0	5	0	0	5	+2.40	—	—	—	0	7	0
38400	0	4	–2.34	0	4	0	0	5	0	—	—	—

Table 11-3 Examples of BRR Settings in Asynchronous Mode (When $ø_P = ø$) (cont)

	ø Frequency (MHz)					
	9.8304			10		
Bit Rate	n	N	Error (%)	n	N	Error (%)
						+0.88
						+0.16
						+0.16
						+0.16
1200	0	255				+0.16
2400	0	127	0	0	129	+0.16
4800	0	63	0	0	64	+0.16
9600	0	31	0	0	32	–1.36
19200	0	15	0	0	15	+1.73
31250	0	9	–1.70	0	9	0
38400	0	7	0	0	7	+1.73

Note: If possible, the error should be within 1%.

In the shaded section, if $ø_P = ø/2$, the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency (ø).

$$B = F \times 10^6/[64 \times 2^{2n-1} \times (N + 1)] \rightarrow N = F \times 10^6/[64 \times 2^{2n-1} \times B] - 1$$

B: Bit rate (bits/second)

N: BRR value ($0 \leq N \leq 255$)

F: $ø_P$ (MHz) when $n \neq 0$, or ø (MHz) when $n = 0$

n: Internal clock source (0, 1, 2, or 3)

The meaning of n is given by the table below:

n	CKS1	CKS0	Clock
0	0	0	ø
1	0	1	$ø_P/4$
2	1	0	$ø_P/16$
3	1	1	$ø_P/64$

Bit rate error can be calculated with the formula below.

$$\text{Error (\%)} = \left\{ \frac{F \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 11-4 Examples of BRR Settings in Synchronous Mode (When $ø_P = ø$)

	ø Frequency (MHz)									
	2		4		5		8		10	
Bit Rate	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**
										—
										—
										—
										—
2.5 k	0									249
5 k	0	99	0	199	0					124
10 k	0	49	0	99	0	124	0	199	0	249
25 k	0	19	0	39	0	49	0	79	0	99
50 k	0	9	0	19	0	24	0	39	0	49
100 k	0	4	0	9	—	—	0	19	0	24
250 k	0	1	0	3	0	4	0	7	0	9
500 k	0	0*	0	1	—	—	0	3	0	4
1 M			0	0*	—	—	0	1	—	—
2.5 M									0	0*
4 M										

Notes: In the shaded section, if $ø_P = ø/2$, the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency (ø).

Blank: No setting is available.

—: A setting is available, but the bit rate is inaccurate.

*: Continuous transfer is not possible.

$$B = F \times 10^6/[8 \times 2^{2n-1} \times (N + 1)] \rightarrow N = F \times 10^6/[8 \times 2^{2n-1} \times B] - 1$$

B: Bit rate (bits per second)

N: BRR value ($0 \leq N \leq 255$)

F: $ø_P$ (MHz) when $n \neq 0$, or ø (MHz) when n = 0

n: Internal clock source (0, 1, 2, or 3)

The meaning of n is given by the table below:

n	CKS1	CKS0	Clock
0	0	0	ø
1	0	1	$ø_P/4$
2	1	0	$ø_P/16$
3	1	1	$ø_P/64$

11.2.9 Serial Communication Mode Register (SCMR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	SDIR	SINV	—	SMIF
Initial value	1	1	1	1	0	0	1	0
Read/Write	—	—	—	—	R/W	R/W	—	R/W

SCMR is an 8-bit readable/writable register that selects the function of SCI0. SCMR is initialized to H'F2 by a reset and in the standby modes.

Bits 7 to 4—Reserved: These bits cannot be modified and are always read as 1.

Bit 3—Data Transfer Direction (SDIR): This bit selects the serial/parallel conversion format.

Bit 3 SDIR	Description	
0	TDR contents are transmitted LSB-first Receive data is stored in RDR LSB-first	(Initial value)
1	TDR contents are transmitted MSB-first Receive data is stored in RDR MSB-first	

Bit 2—Data Invert (SINV): This bit specifies inversion of the data logic level. Inversion specified by the SINV bit applies only to data bits D_7 to D_0. In order to invert the parity bit, the $O/\overline{E}$ bit in SMR must be inverted.

Bit 2 SINV	Description	
0	TDR contents are transmitted as they are TDR contents are stored in RDR as they are	(Initial value)
1	TDR contents are inverted before being transmitted Receive data is stored in RDR in inverted form	

Bit 1—Reserved: This bit cannot be modified and is always read as 1.

Bit 0—Serial Communication Mode Select (SMIF): This bit is reserved. A 1 must not be written to this bit.

Bit 0 SMIF	Description	
0	Normal SCI mode	(Initial value)
1	Reserved mode	

11.3 Operation

11.3.1 Overview

The SCI supports serial data transfer in two modes. In asynchronous mode each character is synchronized individually. In synchronous mode communication is synchronized with a clock signal.

The selection of asynchronous or synchronous mode and the communication format depend on SMR settings as indicated in table 11-5. The clock source depends on the settings of the C/$\overline{\text{A}}$ bit in SMR and the CKE1 and CKE0 bits in SCR as indicated in table 11-6.

Asynchronous Mode

- Data length: 7 or 8 bits can be selected.
- A parity bit or multiprocessor bit can be added, and stop bit lengths of 1 or 2 bits can be selected. (These selections determine the communication format and character length.)
- Framing errors (FER), parity errors (PER), and overrun errors (ORER) can be detected in receive data, and the line-break condition can be detected.
- SCI clock source: an internal or external clock source can be selected.
    - Internal clock: The SCI is clocked by the on-chip baud rate generator and can output a clock signal at the bit-rate frequency.
    - External clock: The external clock frequency must be 16 times the bit rate. (The on-chip baud rate generator is not used.)

Synchronous Mode

- Communication format: The data length is 8 bits.
- Overrun errors (ORER) can be detected in receive data.
- SCI clock source: an internal or external clock source can be selected.
    - Internal clock: The SCI is clocked by the on-chip baud rate generator and outputs a serial clock signal to external devices.
    - External clock: The on-chip baud rate generator is not used. The SCI operates on the input serial clock.

Table 11-5 Communication Formats Used by SCI

SMR Settings						Communication Format			
Bit 7 C/$\overline{A}$	Bit 6 CHR	Bit 2 MP	Bit 5 PE	Bit 3 STOP	Mode	Data Length	Multi-processor Bit	Parity Bit	Stop-Bit Length
0	0	0	0	0	Asynchronous mode	8 bits	None	None	1 bit
				1					2 bits
			1	0				Present	1 bit
				1					2 bits
	1		0	0		7 bits		None	1 bit
				1					2 bits
			1	0				Present	1 bit
				1					2 bits
	0	1	—	0	Asynchronous mode (multi-processor format)	8 bits	Present	None	1 bit
				1					2 bits
	1			0		7 bits			1 bit
				1					2 bits
1	—	—	—	—	Synchronous mode	8 bits	None		None

Table 11-6 SCI Clock Source Selection

SMR	SCR			Serial Transmit/Receive Clock	
Bit 7 C/$\overline{A}$	Bit 1 CKE1	Bit 0 CKE0	Mode	Clock Source	SCK Pin Function
0	0	0	Async	Internal	Input/output port (not used by SCI)
		1			Serial clock output at bit rate
	1	0		External	Serial clock input at 16 × bit rate
		1			
1	0	0	Sync	Internal	Serial clock output
		1			
	1	0		External	Serial clock input
		1			

11.3.2 Asynchronous Mode

In asynchronous mode, each transmitted or received character is individually synchronized by framing it with a start bit and stop bit.

Full duplex data transfer is possible because the SCI has independent transmit and receive sections. Double buffering in both sections enables the SCI to be programmed for continuous data transfer.

Figure 11-2 shows the general format of one character sent or received in asynchronous mode. The communication channel is normally held in the mark state (high). Character transmission or reception starts with a transition to the space state (low).

The first bit transmitted or received is the start bit (low). It is followed by the data bits, in which the least significant bit (LSB) comes first. The data bits are followed by the parity or multiprocessor bit, if present, then the stop bit or bits (high) confirming the end of the frame.

In receiving, the SCI synchronizes on the falling edge of the start bit, and samples each bit at the center of the bit (at the 8th cycle of the internal serial clock, which runs at 16 times the bit rate).

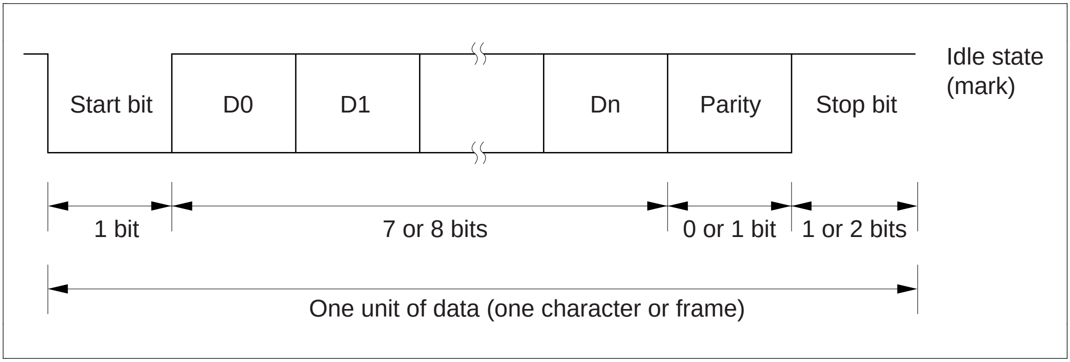


Figure 11-2 Data Format in Asynchronous Mode
(Example of 8-Bit Data with Parity Bit and Two Stop Bits)

(1) Data Format: Table 11-7 lists the data formats that can be sent and received in asynchronous mode. Twelve formats can be selected by bits in the serial mode register (SMR).

Table 11-7 Data Formats in Asynchronous Mode

SMR Bits															
CHR	**PE**	**MP**	**STOP**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	—	1	0	S	8-bit data								MPB	STOP	
0	—	1	1	S	8-bit data								MPB	STOP	STOP
1	—	1	0	S	7-bit data							MPB	STOP		
1	—	1	1	S	7-bit data							MPB	STOP	STOP	

Notes: SMR: Serial mode register
S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

(2) Clock: In asynchronous mode it is possible to select either an internal clock created by the on-chip baud rate generator, or an external clock input at the SCK pin. The selection is made by the C/$\overline{\text{A}}$ bit in the serial mode register (SMR) and the CKE1 and CKE0 bits in the serial control register (SCR). Refer to table 11-6.

If an external clock is input at the SCK pin, its frequency should be 16 times the desired bit rate.

If the internal clock provided by the on-chip baud rate generator is selected and the SCK pin is used for clock output, the output clock frequency is equal to the bit rate, and the clock pulse rises at the center of the transmit data bits. Figure 11-3 shows the phase relationship between the output clock and transmit data.

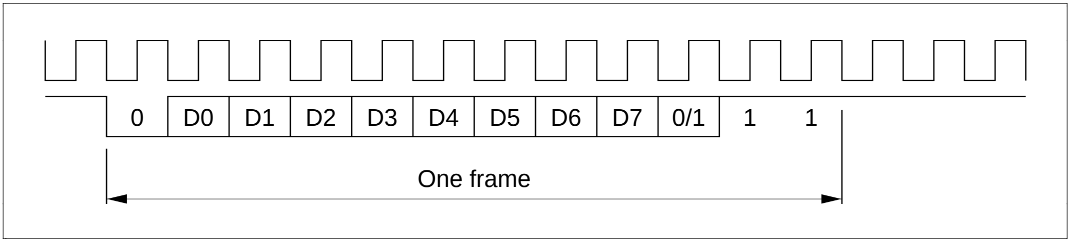


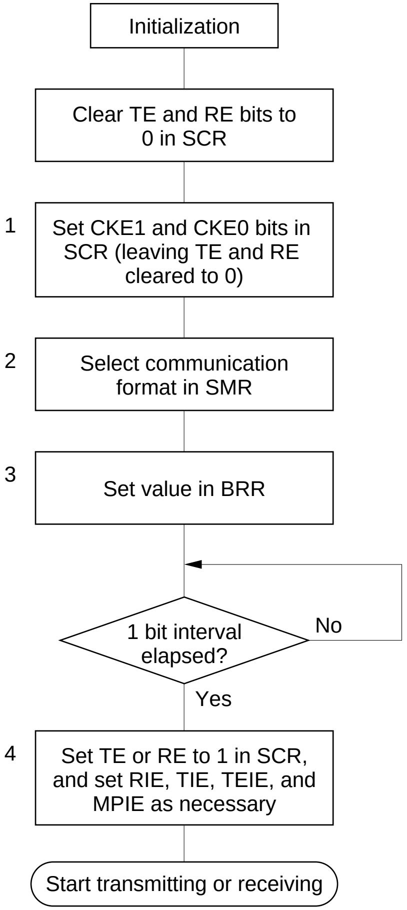
Figure 11-3 Phase Relationship between Clock Output and Transmit Data (Asynchronous Mode)

(3) Transmitting and Receiving Data

SCI Initialization: Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI following the procedure in figure 11-4.

Note: When changing the communication mode or format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCI operation becomes unreliable if the clock is stopped.


Initialization
Clear TE and RE bits to 0 in SCR
1
Set CKE1 and CKE0 bits in SCR (leaving TE and RE cleared to 0)
2
Select communication format in SMR
3
Set value in BRR
1 bit interval elapsed?
No
Yes
4
Set TE or RE to 1 in SCR, and set RIE, TIE, TEIE, and MPIE as necessary
Start transmitting or receiving

1. Select the clock source in the serial control register (SCR). Leave TE and RE cleared to 0. If clock output is selected, in asynchronous mode, clock output starts immediately after the setting is made in SCR.

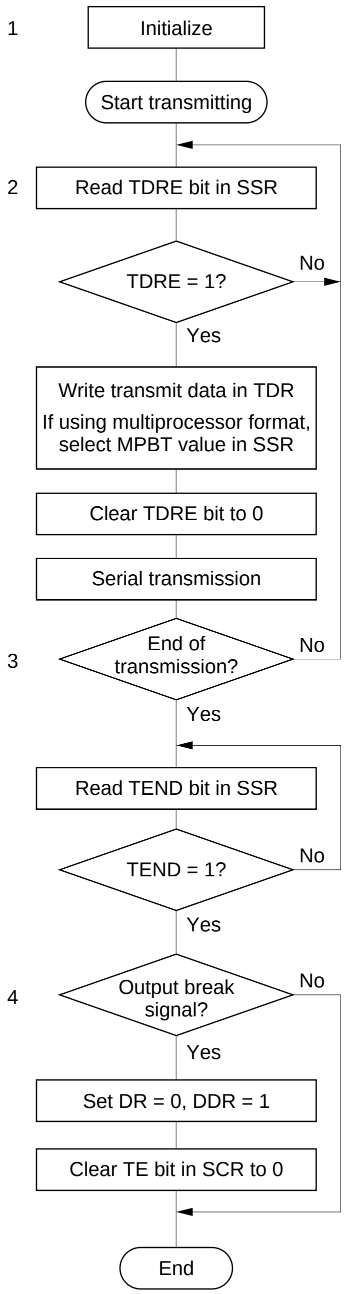
2. Select the communication format in the serial mode register (SMR).

3. Write the value corresponding to the bit rate in the bit rate register (BRR). This step is not necessary when an external clock is used.

4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR). Setting TE or RE enables the SCI to use the TxD or RxD pin.
   Also set the RIE, TIE, TEIE, and MPIE bits as necessary to enable interrupts. The initial states are the mark transmit state, and the idle receive state (waiting for a start bit).

Figure 11-4 Sample Flowchart for SCI Initialization

Transmitting Serial Data: Follow the procedure in figure 11-5 for transmitting serial data.



1. SCI initialization: the transmit data output function of the TxD pin is selected automatically.

2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. If a multiprocessor format is selected, after writing the transmit data write 0 or 1 in the multiprocessor bit transfer (MPBT) in SSR. Transition of the TDRE bit from 0 to 1 can be reported by an interrupt.

3. (a) To continue transmitting serial data: read the TDRE bit to check whether it is safe to write; if TDRE = 1, write data in TDR, then clear TDRE to 0.
   (b) To end serial transmission: end of transmission can be confirmed by checking transition of the TEND bit from 0 to 1. This can be reported by a TEI interrupt.

4. To output a break signal at the end of serial transmission: set the DDR bit to 1 and clear the DR bit to 0 (DDR and DR are I/O port registers), then clear TE to 0 in SCR.

Figure 11-5 Sample Flowchart for Transmitting Serial Data

In transmitting serial data, the SCI operates as follows.

1. The SCI monitors the TDRE bit in SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from TDR into the transmit shift register (TSR).

2. After loading the data from TDR into TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the TIE bit (TDR-empty interrupt enable) is set to 1 in SCR, the SCI requests a TXI interrupt (TDR-empty interrupt) at this time.

   Serial transmit data is transmitted in the following order from the TxD pin:

   a. Start bit: One 0 bit is output.
   b. Transmit data: Seven or eight bits are output, LSB-first.
   c. Parity bit or multiprocessor bit: One parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.
   d. Stop bit: One or two 1 bits (stop bits) are output.
   e. Mark state: Output of 1 bits continues until the start bit of the next transmit data.

3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, after loading new data from TDR into TSR and transmitting the stop bit, the SCI begins serial transmission of the next frame. If TDRE is 1, after setting the TEND bit to 1 in SSR and transmitting the stop bit, the SCI continues 1-level output in the mark state, and if the TEIE bit (TSR-empty interrupt enable) in SCR is set to 1, the SCI generates a TEI interrupt request (TSR-empty interrupt).

Figure 11-6 shows an example of SCI transmit operation in asynchronous mode.

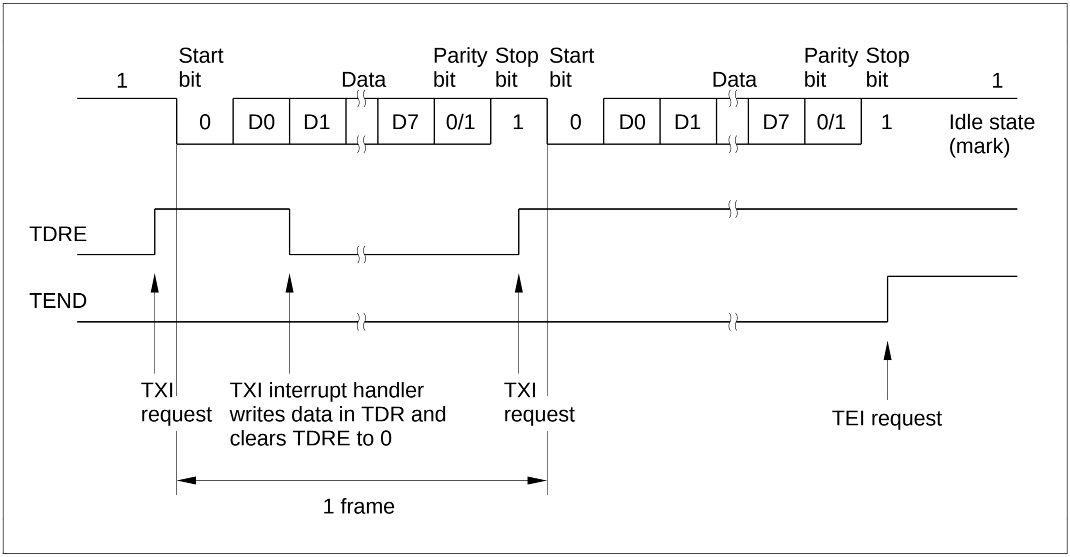
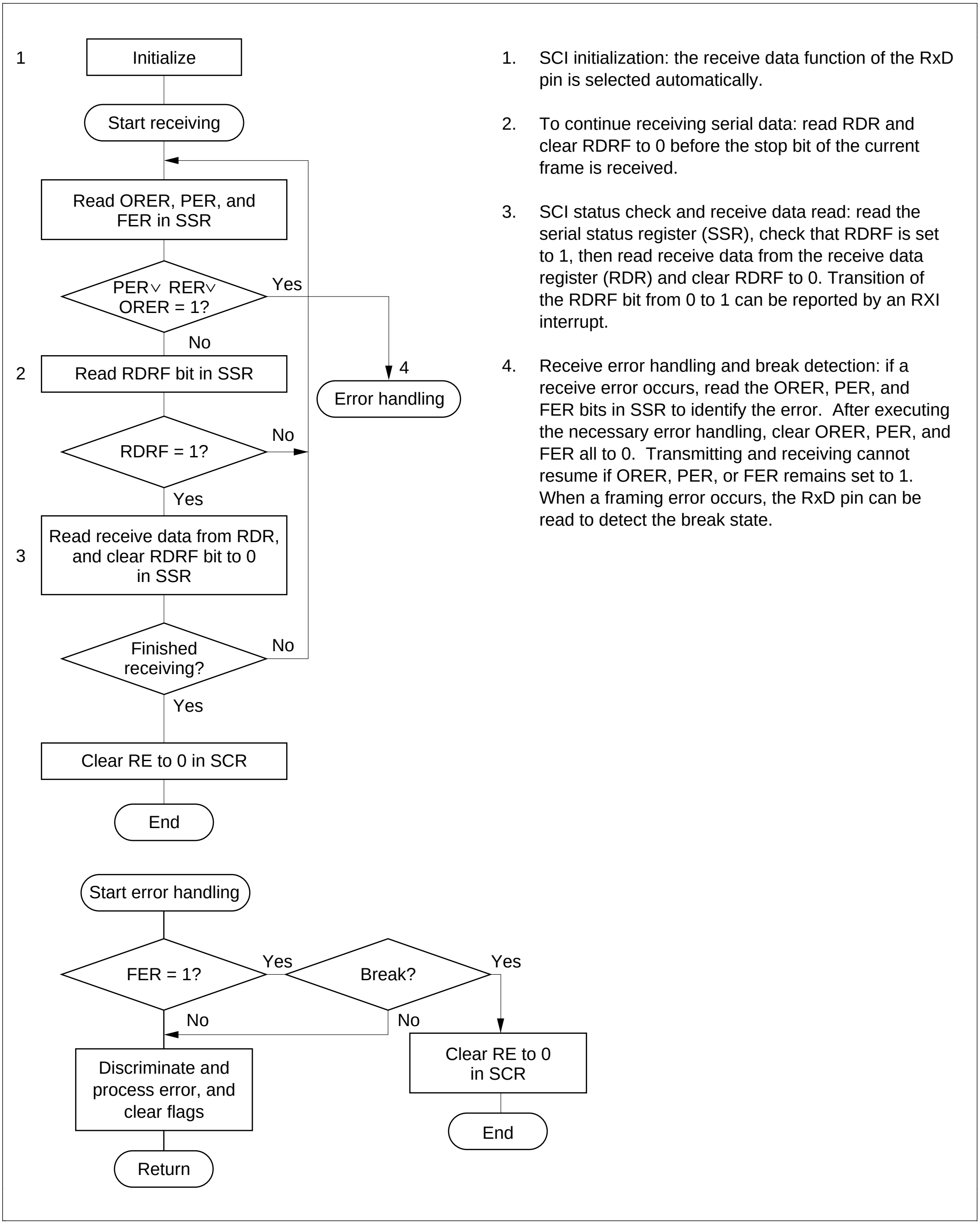


Figure 11-6 Example of SCI Transmit Operation (8-Bit Data with Parity and One Stop Bit)

Receiving Serial Data: Follow the procedure in figure 11-7 for receiving serial data.



1. SCI initialization: the receive data function of the RxD pin is selected automatically.

2. To continue receiving serial data: read RDR and clear RDRF to 0 before the stop bit of the current frame is received.

3. SCI status check and receive data read: read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. Transition of the RDRF bit from 0 to 1 can be reported by an RXI interrupt.

4. Receive error handling and break detection: if a receive error occurs, read the ORER, PER, and FER bits in SSR to identify the error. After executing the necessary error handling, clear ORER, PER, and FER all to 0. Transmitting and receiving cannot resume if ORER, PER, or FER remains set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.

Figure 11-7 Sample Flowchart for Receiving Serial Data

In receiving, the SCI operates as follows.

1. The SCI monitors the receive data line and synchronizes internally when it detects a start bit.

2. Receive data is shifted into RSR in order from LSB to MSB.

3. The parity bit and stop bit are received.

   After receiving these bits, the SCI makes the following checks:

   a. Parity check: The number of 1s in the receive data must match the even or odd parity setting of the O/$\overline{\text{E}}$ bit in SMR.
   b. Stop bit check: The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
   c. Status check: RDRF must be 0 so that receive data can be loaded from RSR into RDR.

   If these checks all pass, the SCI sets RDRF to 1 and stores the received data in RDR. If one of the checks fails (receive error), the SCI operates as indicated in table 11-8.

   Note: When a receive error flag is set, further receiving is disabled. The RDRF bit is not set to 1. Be sure to clear the error flags.

4. After setting RDRF to 1, if the RIE bit (receive-end interrupt enable) is set to 1 in SCR, the SCI requests an RXI (receive-end) interrupt. If one of the error flags (ORER, PER, or FER) is set to 1 and the RIE bit in SCR is also set to 1, the SCI requests an ERI (receive-error) interrupt.

Figure 11-8 shows an example of SCI receive operation in asynchronous mode.

Table 11-8 Receive Error Conditions and SCI Operation

Receive error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	Receiving of next data ends while RDRF is still set to 1 in SSR	Receive data not loaded from RSR into RDR
Framing error	FER	Stop bit is 0	Receive data loaded from RSR into RDR
Parity error	PER	Parity of receive data differs from even/odd parity setting in SMR	Receive data loaded from RSR into RDR

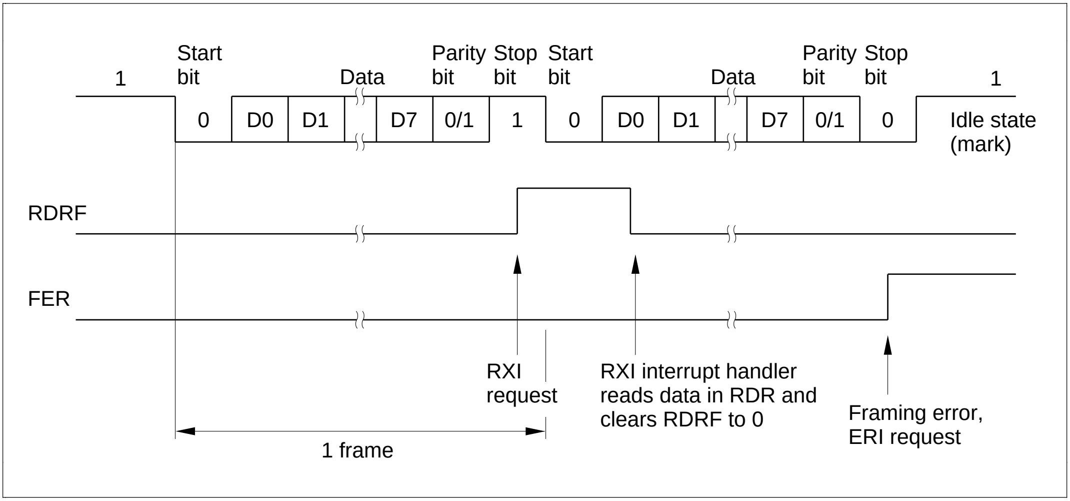


Figure 11-8 Example of SCI Receive Operation (8-Bit Data with Parity and One Stop Bit)

(4) Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line. The processors communicate in asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by an ID.

A serial communication cycle consists of two cycles: an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles.

The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1.

After receiving data with the multiprocessor bit set to 1, the receiving processor with an ID matching the received data continues to receive further incoming data. Multiple processors can send and receive data in this way.

Four formats are available. Parity-bit settings are ignored when a multiprocessor format is selected. For details see table 11-7.

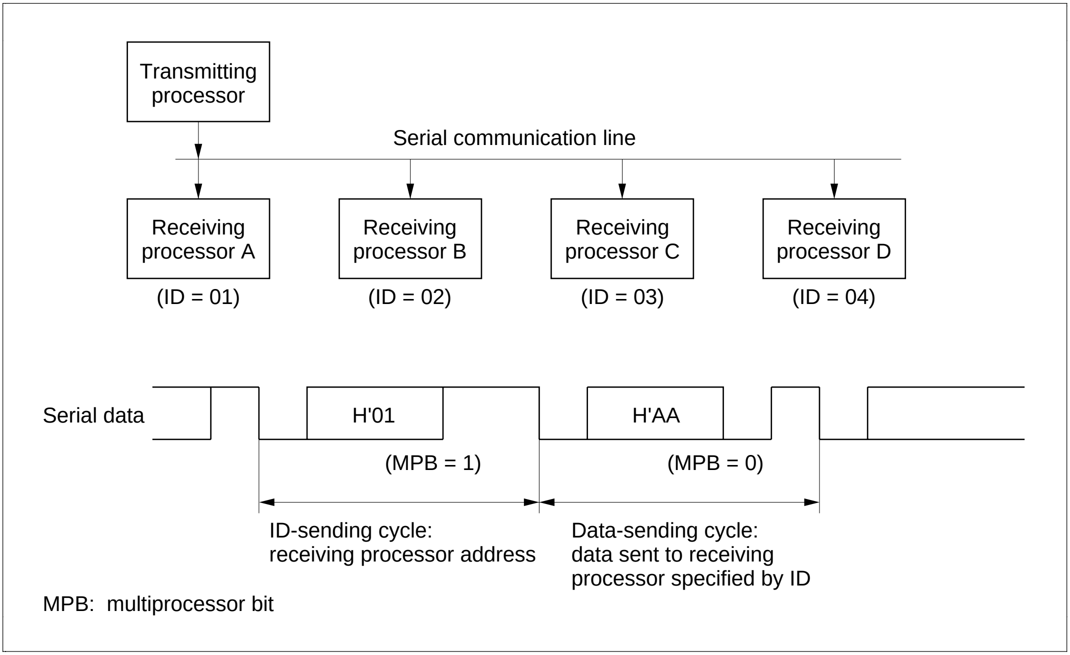
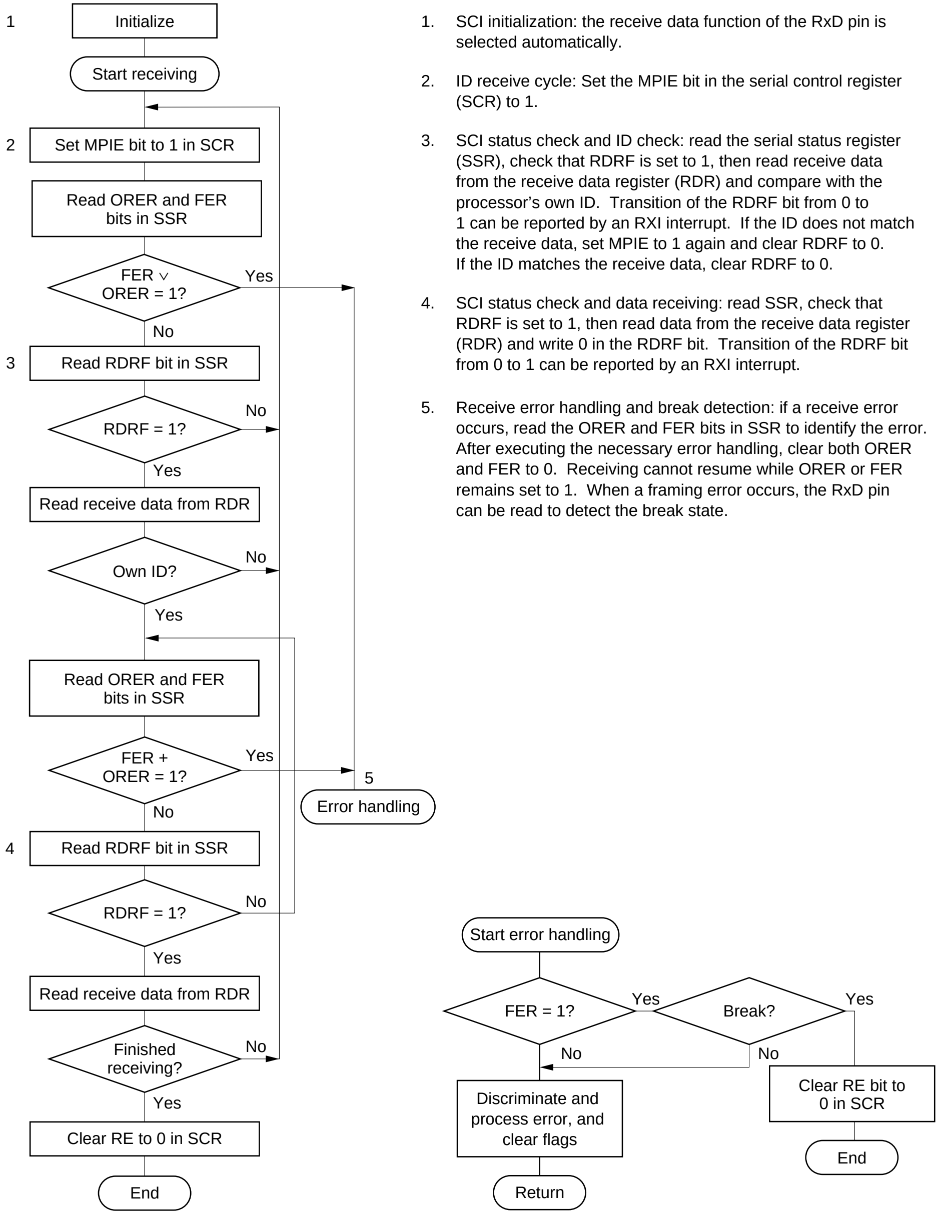


Figure 11-9 Example of Communication among Processors using Multiprocessor Format (Sending Data H'AA to Receiving Processor A)

Transmitting Multiprocessor Serial Data: See figures 11-5 and 11-6.

Receiving Multiprocessor Serial Data: Follow the procedure in figure 11-10 for receiving multiprocessor serial data.



1. SCI initialization: the receive data function of the RxD pin is selected automatically.

2. ID receive cycle: Set the MPIE bit in the serial control register (SCR) to 1.

3. SCI status check and ID check: read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and compare with the processor's own ID. Transition of the RDRF bit from 0 to 1 can be reported by an RXI interrupt. If the ID does not match the receive data, set MPIE to 1 again and clear RDRF to 0. If the ID matches the receive data, clear RDRF to 0.

4. SCI status check and data receiving: read SSR, check that RDRF is set to 1, then read data from the receive data register (RDR) and write 0 in the RDRF bit. Transition of the RDRF bit from 0 to 1 can be reported by an RXI interrupt.

5. Receive error handling and break detection: if a receive error occurs, read the ORER and FER bits in SSR to identify the error. After executing the necessary error handling, clear both ORER and FER to 0. Receiving cannot resume while ORER or FER remains set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.

Figure 11-10 Sample Flowchart for Receiving Multiprocessor Serial Data

Figure 11-11 shows an example of an SCI receive operation using a multiprocessor format (8-bit data with multiprocessor bit and one stop bit).

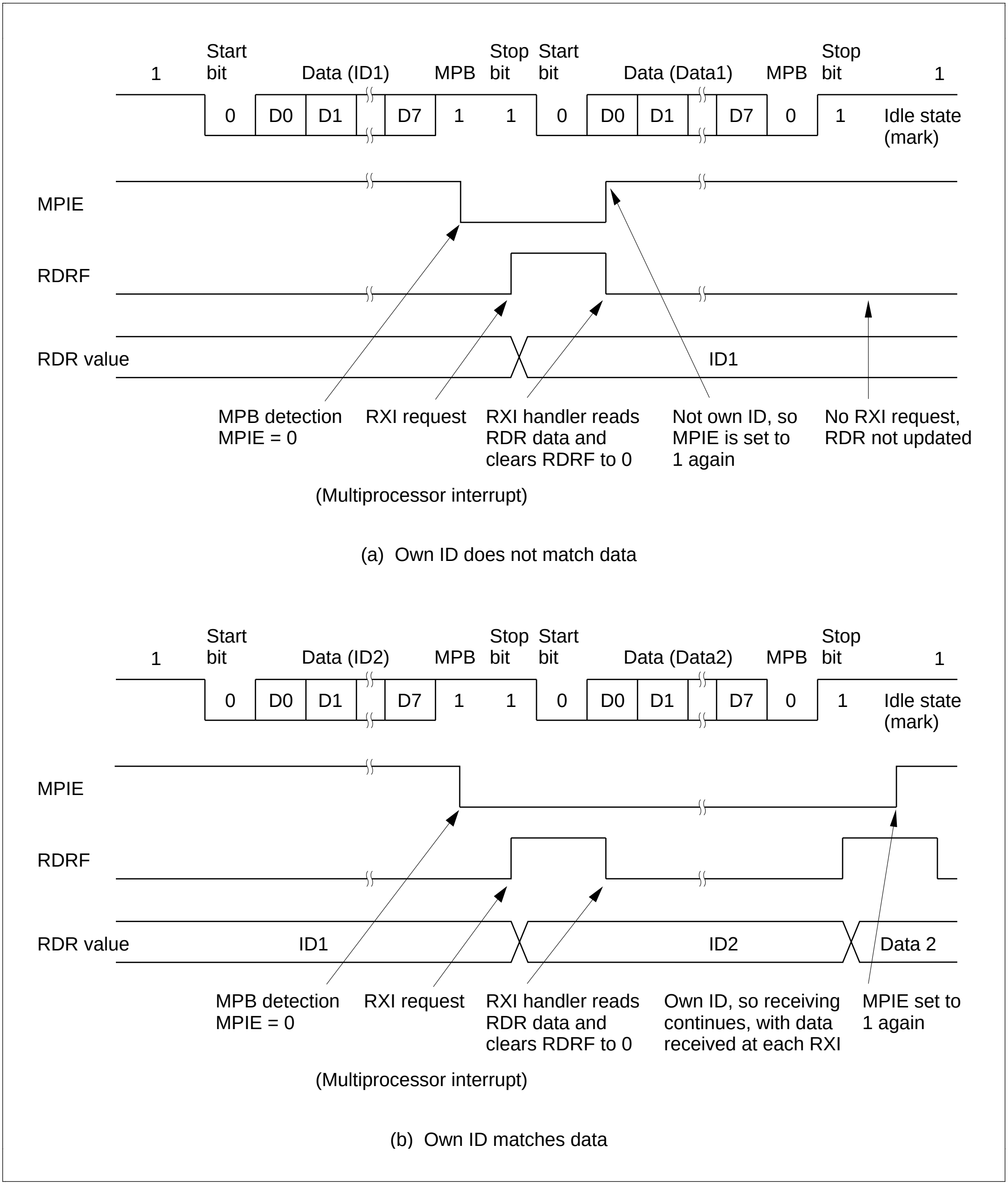


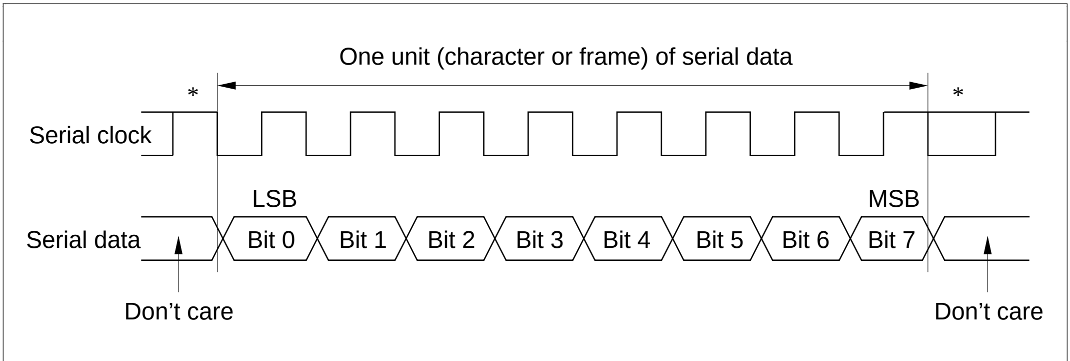
Figure 11-11 Example of SCI Receive Operation (8-Bit Data with Multiprocessor Bit and One Stop Bit)

11.3.3 Synchronous Mode

(1) Overview: In synchronous mode, the SCI transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI transmitter and receiver share the same clock but are otherwise independent, so full duplex communication is possible. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 11-12 shows the general format in synchronous serial communication.



Note: * High except in continuous transmitting or receiving

Figure 11-12 Data Format in Synchronous Communication

In synchronous serial communication, each data bit is sent on the communication line from one falling edge of the serial clock to the next. Data is received in synchronization with the rising edge of the serial clock.

In each character, the serial data bits are transmitted in order from LSB (first) to MSB (last). After output of the MSB, the communication line remains in the state of the MSB.

Communication Format: The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.

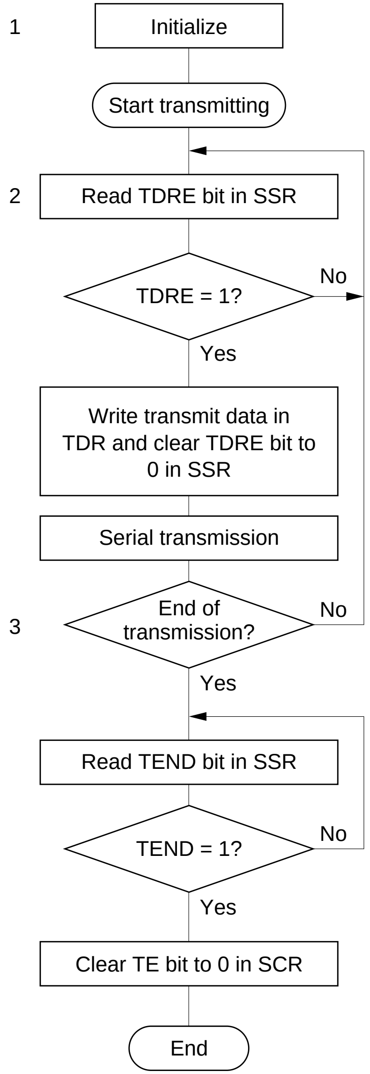
Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected by clearing or setting the CKE1 bit in the serial control register (SCR). See table 11-6.

When the SCI operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI is not transmitting or receiving, the clock signal remains at the high level.

(2) Transmitting and Receiving Data

SCI Initialization: The SCI must be initialized in the same way as in asynchronous mode. See figure 11-4. When switching from asynchronous mode to synchronous mode, check that the ORER, FER, and PER bits are cleared to 0. Transmitting and receiving cannot begin if ORER, FER, or PER is set to 1.

Transmitting Serial Data: Follow the procedure in figure 11-13 for transmitting serial data.



1. SCI initialization: the transmit data output function of the TxD pin is selected automatically.

2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. Transition of the TDRE bit from 0 to 1 can be reported by a TXI interrupt.

3. (a) To continue transmitting serial data: read the TDRE bit to check whether it is safe to write; if TDRE = 1, write data in TDR, then clear TDRE to 0.
   (b) To end serial transmission: end of transmission can be confirmed by checking transition of the TEND bit from 0 to 1. This can be reported by a TEI interrupt.

Figure 11-13 Sample Flowchart for Serial Transmitting

In transmitting serial data, the SCI operates as follows.

1. The SCI monitors the TDRE bit in SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from TDR into the transmit shift register (TSR).

2. After loading the data from TDR into TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the TIE bit (TDR-empty interrupt enable) in SCR is set to 1, the SCI requests a TXI interrupt (TDR-empty interrupt) at this time.

   If clock output is selected the SCI outputs eight serial clock pulses, triggered by the clearing of the TDRE bit to 0. If an external clock source is selected, the SCI outputs data in synchronization with the input clock.

   Data is output from the TxD pin in order from LSB (bit 0) to MSB (bit 7).

3. The SCI checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI loads data from TDR into TSR, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in SSR to 1, transmits the MSB, then holds the output in the MSB state. If the TEIE bit (transmit-end interrupt enable) in SCR is set to 1, a TEI interrupt (TSR-empty interrupt) is requested at this time.

4. After the end of serial transmission, the SCK pin is held at the high level.

Figure 11-14 shows an example of SCI transmit operation.

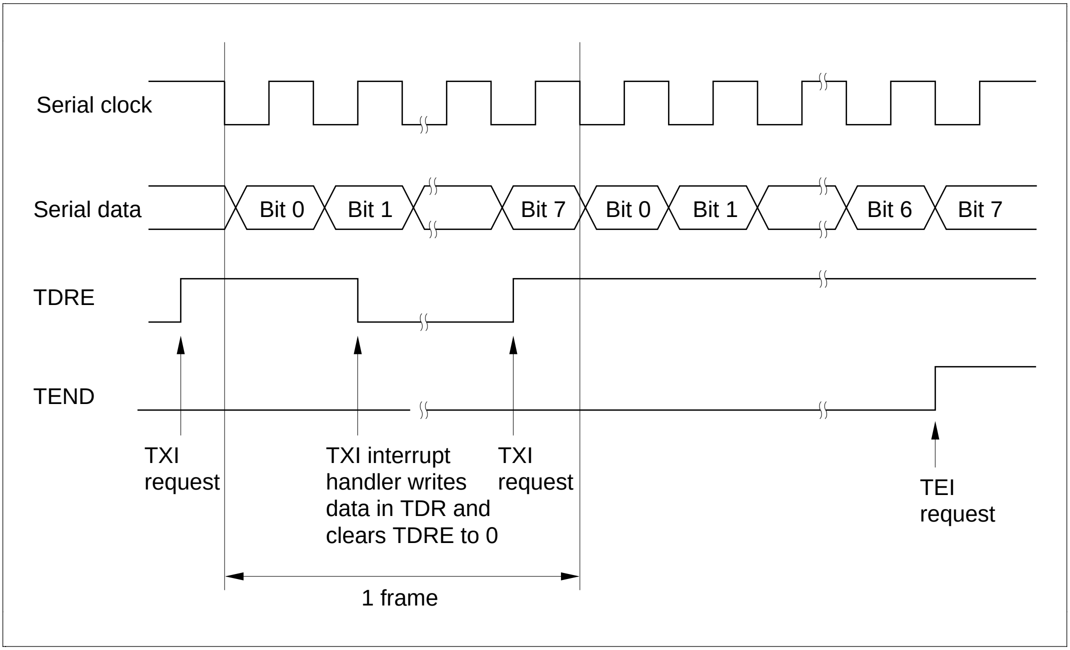
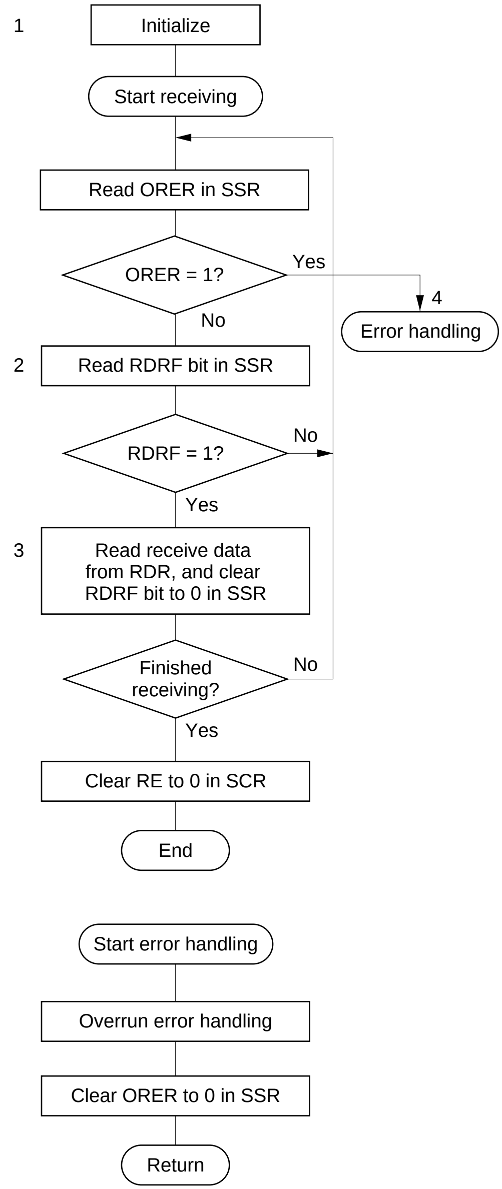


Figure 11-14 Example of SCI Transmit Operation

Receiving Serial Data: Follow the procedure in figure 11-15 for receiving serial data. When switching from asynchronous mode to synchronous mode, be sure to check that PER and FER are cleared to 0. If PER or FER is set to 1 the RDRF bit will not be set and both transmitting and receiving will be disabled.



1. SCI initialization: the receive data function of the RxD pin is selected automatically.

2. SCI status check and receive data read: read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. Transition of the RDRF bit from 0 to 1 can be reported by an RXI interrupt.

3. To continue receiving serial data: read RDR and clear RDRF to 0 before the MSB (bit 7) of the current frame is received.

4. Receive error handling: if a receive error occurs, read the ORER bit in SSR then, after executing the necessary error handling, clear ORER to 0. Neither transmitting nor receiving can resume while ORER remains set to 1. When clock output mode is selected, receiving can be halted temporarily by receiving one dummy byte and causing an overrun error. When preparations to receive the next data are completed, clear the ORER bit to 0. This causes receiving to resume, so return to the step marked 2 in the flowchart.

Figure 11-15 Sample Flowchart for Serial Receiving

In receiving, the SCI operates as follows.

1. If an external clock is selected, data is input in synchronization with the input clock. If clock output is selected, as soon as the RE bit is set to 1 the SCI begins outputting the serial clock and inputting data. If clock output is stopped because the ORER bit is set to 1, output of the serial clock and input of data resume as soon as the ORER bit is cleared to 0.

2. Receive data is shifted into RSR in order from LSB to MSB.

   After receiving the data, the SCI checks that RDRF is 0 so that receive data can be loaded from RSR into RDR. If this check passes, the SCI sets RDRF to 1 and stores the received data in RDR. If the check does not pass (receive error), the SCI operates as indicated in table 11-8.

   Note: Both transmitting and receiving are disabled while a receive error flag is set. The RDRF bit is not set to 1. Be sure to clear the error flag.

3. After setting RDRF to 1, if the RIE bit (receive-end interrupt enable) is set to 1 in SCR, the SCI requests an RXI (receive-end) interrupt. If the ORER bit is set to 1 and the RIE bit in SCR is set to 1, the SCI requests an ERI (receive-error) interrupt.

   When clock output mode is selected, clock output stops when the RE bit is cleared to 0 or the ORER bit is set to 1. To prevent clock count errors, it is safest to receive one dummy byte and generate an overrun error.

Figure 11-16 shows an example of SCI receive operation.

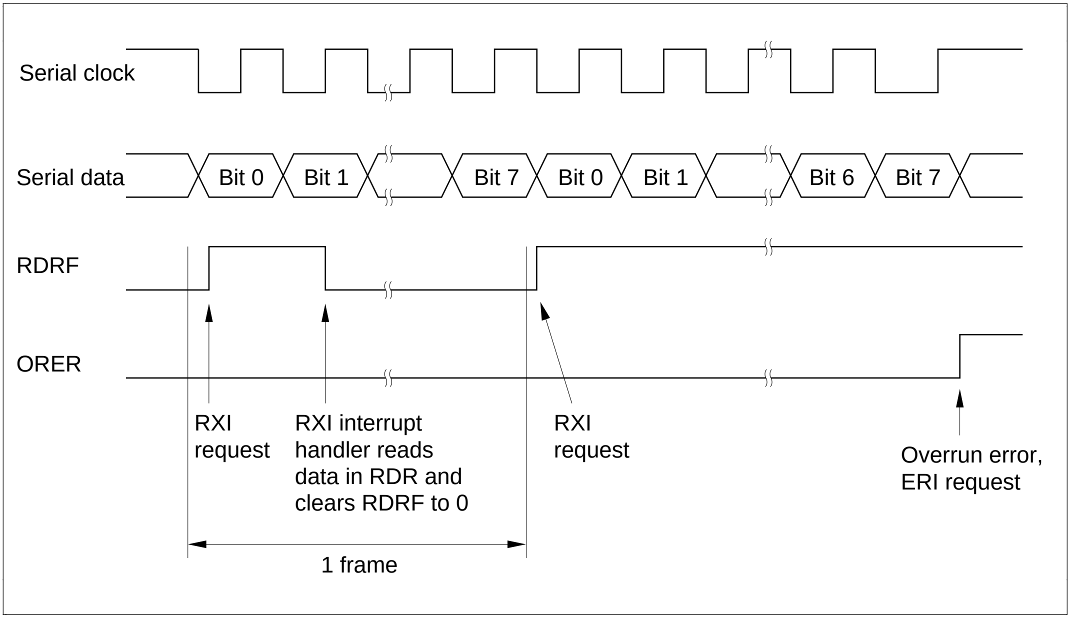
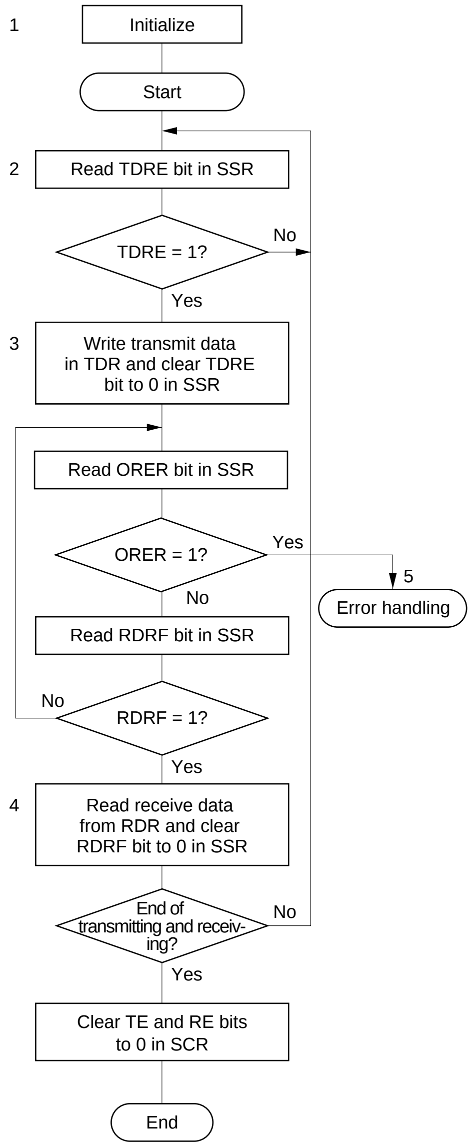


Figure 11-16 Example of SCI Receive Operation

Transmitting and Receiving Serial Data Simultaneously: Follow the procedure in figure 11-17 for transmitting and receiving serial data simultaneously. If clock output mode is selected, output of the serial clock begins simultaneously with serial transmission.


1
Initialize
Start
2
Read TDRE bit in SSR
TDRE = 1?
No
Yes
3
Write transmit data in TDR and clear TDRE bit to 0 in SSR
Read ORER bit in SSR
ORER = 1?
Yes
5
Error handling
No
Read RDRF bit in SSR
No
RDRF = 1?
Yes
4
Read receive data from RDR and clear RDRF bit to 0 in SSR
End of transmitting and receiv-ing?
No
Yes
Clear TE and RE bits to 0 in SCR
End

1. SCI initialization: the transmit data output function of the TxD pin and receive data input function of the RxD pin are selected, enabling simultaneous transmitting and receiving.

2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. Transition of the TDRE bit from 0 to 1 can be reported by a TXI interrupt.

3. SCI status check and receive data read: read the serial status register (SSR), check that the RDRF bit is 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. Transition of the RDRF bit from 0 to 1 can be reported by an RXI interrupt.

4. To continue transmitting and receiving serial data: read RDR and clear RDRF to 0 before the MSB (bit 7) of the current frame is received. Also read the TDRE bit and check that it is set to 1, indicating that it is safe to write; then write data in TDR and clear TDRE to 0 before the MSB (bit 7) of the current frame is transmitted.

5. Receive error handling: if a receive error occurs, read the ORER bit in SSR then, after executing the necessary error handling, clear ORER to 0. Neither transmitting nor receiving can resume while ORER remains set to 1.

Figure 11-17 Sample Flowchart for Serial Transmitting and Receiving

Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, clear both TE and RE to 0, then set both TE and RE to 1.

11.4 Interrupts

The SCI can request four types of interrupts: ERI, RXI, TXI, and TEI. Table 11-9 indicates the source and priority of these interrupts. The interrupt sources can be enabled or disabled by the TIE, RIE, and TEIE bits in the SCR. Independent signals are sent to the interrupt controller for each interrupt source, except that the receive-error interrupt (ERI) is the logical OR of three sources: overrun error, framing error, and parity error.

The TXI interrupt indicates that the next transmit data can be written. The TEI interrupt indicates that the SCI has stopped transmitting data.

Table 11-9 SCI Interrupt Sources

Interrupt	Description	Priority
ERI	Receive-error interrupt (ORER, FER, or PER)	High
RXI	Receive-end interrupt (RDRF)	↑
TXI	TDR-empty interrupt (TDRE)	
TEI	TSR-empty interrupt (TEND)	Low

11.5 Application Notes

Application programmers should note the following features of the SCI.

(1) TDR Write: The TDRE bit in SSR is simply a flag that indicates that the TDR contents have been transferred to TSR. The TDR contents can be rewritten regardless of the TDRE value. If a new byte is written in TDR while the TDRE bit is 0, before the old TDR contents have been moved into TSR, the old byte will be lost. Software should check that the TDRE bit is set to 1 before writing to TDR.

(2) Multiple Receive Errors: Table 11-10 lists the values of flag bits in SSR when multiple receive errors occur, and indicates whether the RSR contents are transferred to RDR.

Table 11-10 SSR Bit States and Data Transfer when Multiple Receive Errors Occur

	SSR Bits				
Receive error	**RDRF**	**ORER**	**FER**	**PER**	**RSR → RDR**[*2]
Overrun error	1[*1]	1	0	0	No
Framing error	0	0	1	0	Yes
Parity error	0	0	0	1	Yes
Overrun and framing errors	1[*1]	1	1	0	No
Overrun and parity errors	1[*1]	1	0	1	No
Framing and parity errors	0	0	1	1	Yes
Overrun, framing, and parity errors	1[*1]	1	1	1	No

Notes: 1. Set to 1 before the overrun error occurs.
2. Yes: The RSR contents are transferred to RDR.
No: The RSR contents are not transferred to RDR.

(3) Line Break Detection: When the RxD pin receives a continuous stream of 0's in asynchronous mode (line-break state), a framing error occurs because the SCI detects a 0 stop bit. The value H'00 is transferred from RSR to RDR. Software can detect the line-break state as a framing error accompanied by H'00 data in RDR.

The SCI continues to receive data, so if the FER bit is cleared to 0 another framing error will occur.

(4) Sampling Timing and Receive Margin in Asynchronous Mode: The serial clock used by the SCI in asynchronous mode runs at 16 times the bit rate. The falling edge of the start bit is detected by sampling the RxD input on the falling edge of this clock. After the start bit is detected, each bit of receive data in the frame (including the start bit, parity bit, and stop bit or bits) is sampled on the rising edge of the serial clock pulse at the center of the bit. See figure 11-18.

It follows that the receive margin can be calculated as in equation (1).

When the absolute frequency deviation of the clock signal is 0 and the clock duty cycle is 0.5, data can theoretically be received with distortion up to the margin given by equation (2). This is a theoretical limit, however. In practice, system designers should allow a margin of 20% to 30%.

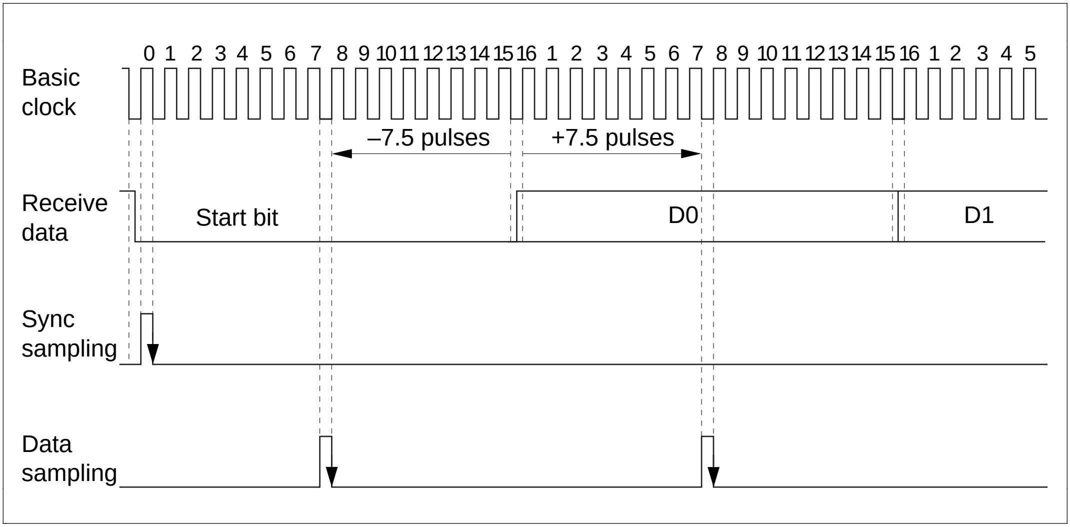


Figure 11-18 Sampling Timing (Asynchronous Mode)

M = {[0.5 – 1/(2N)] – (D – 0.5)/N – (L – 0.5)F} × 100 [%] (1)

M: Receive margin

N: Ratio of basic clock to bit rate (N=16)

D: Duty factor of clock—ratio of high pulse width to low width (0.5 to 1.0)

L: Frame length (9 to 12)

F: Absolute clock frequency deviation

When D = 0.5 and F = 0

M = (0.5 –1/2 × 16) × 100 [%] = 46.875% (2)

Section 12 Host Interface

12.1 Overview

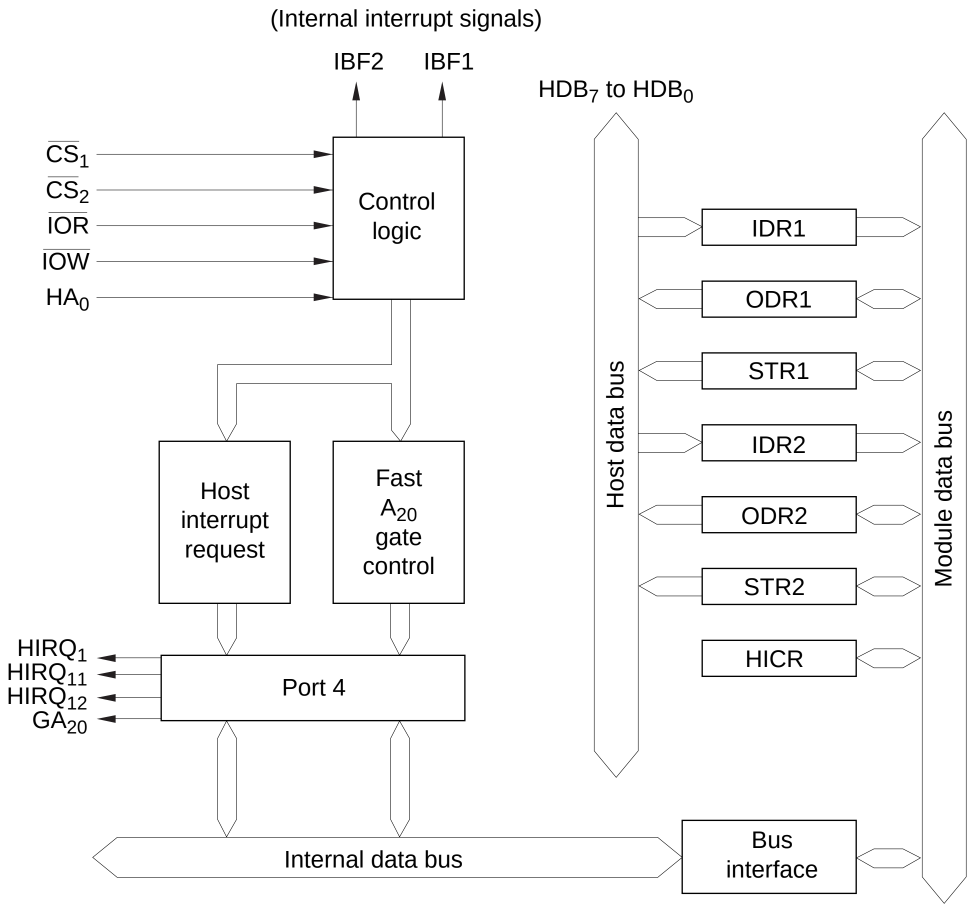
The H8/3502 has an on-chip host interface (HIF) that provides a dual-channel parallel interface between the on-chip CPU and a host processor. The host interface is available only when the HIE bit is set to 1 in SYSCR. This mode is called slave mode, because it is designed for a master-slave communication system in which the H8/3502 chip is slaved to a host processor.

The host interface consists of four 1-byte data registers, two 1-byte status registers, a 1-byte control register, fast A_{20} gate logic, and a host interrupt request circuit. Communication is carried out via five control signals from the host processor ($\overline{CS}_1$, $\overline{CS}_2$, HA_0, $\overline{IOR}$, and $\overline{IOW}$), four output signals to the host processor (GA_{20}, $HIRQ_1$, $HIRQ_{11}$, and $HIRQ_{12}$), and an 8-bit bidirectional command/data bus (HDB_7 to HDB_0). The $\overline{CS}_1$ and $\overline{CS}_2$ signals select one of the two interface channels.

Note: If one of the two interface channels will not be used, tie the unused $\overline{CS}$ pin to V_{CC}. For example, if interface channel 1 (IDR1, ODR1, STR1) is not used, tie $\overline{CS}_1$ to V_{CC}.

12.1.1 Block Diagram

Figure 12-1 is a block diagram of the host interface.



Legend
IDR1: Input data register 1
IDR2: Input data register 2
ODR1: Output data register 1
ODR2: Output data register 2
STR1: Status register 1
STR2: Status register 2
HICR: Host interface control register

Figure 12-1 Host Interface Block Diagram

12.1.2 Input and Output Pins

Table 12-1 lists the input and output pins of the host interface module.

Table 12-1 HIF Input/Output Pins

Name	Abbreviation	Port	I/O	Function
I/O read	$\overline{\text{IOR}}$	$P7_6$	Input	Host interface read signal
I/O write	$\overline{\text{IOW}}$	$P7_5$	Input	Host interface write signal
Chip select 1	$\overline{\text{CS}}_1$	$P7_4$	Input	Host interface chip select signal for IDR1, ODR1, STR1
Chip select 2	$\overline{\text{CS}}_2$	$P4_6$	Input	Host interface chip select signal for IDR2, ODR2, STR2
Command/data	HA_0	$P7_7$	Input	Host interface address select signal In host read access, this signal selects the status registers (STR1, STR2) or data registers (ODR1, ODR2). In host write access to the data registers (IDR1, IDR2), this signal indicates whether the host is writing a command or data.
Data bus	HDB_7 to HDB_0	$P3_7$ to $P3_0$	I/O	Host interface data bus (single-chip mode)
Host interrupt 1	$HIRQ_1$	$P4_4$	Output	Interrupt output 1 to host
Host interrupt 11	$HIRQ_{11}$	$P4_3$	Output	Interrupt output 11 to host
Host interrupt 12	$HIRQ_{12}$	$P4_5$	Output	Interrupt output 12 to host
Gate A_{20}	GA_{20}	$P4_7$	Output	A_{20} gate control signal output

12.1.3 Register Configuration

Table 12-2 lists the host interface registers.

Table 12-2 HIF Registers

		R/W		Initial	Slave	Master Address*4		
Name	**Abbreviation**	**Slave**	**Host**	**Value**	**Address*3**	$\overline{CS}_1$	$\overline{CS}_2$	HA_0
System control register	SYSCR	R/W*1	—	H'09	H'FFC4	—	—	—
Host interface control register	HICR	R/W	—	H'F8	H'FFF0	—	—	—
Input data register 1	IDR1	R	W	—	H'FFF4	0	1	0/1*5
Output data register 1	ODR1	R/W	R	—	H'FFF5	0	1	0
Status register 1	STR1	R/(W)*2	R	H'00	H'FFF6	0	1	1
Input data register 2	IDR2	R	W	—	H'FFFC	1	0	0
Output data register 2	ODR2	R/W	R	—	H'FFFD	1	0	0/1*5
Status register 2	STR2	R/(W)*2	R	H'00	H'FFFE	1	0	1
Serial/timer control register	STCR	R/W	—	H'00	H'FFC3	—	—	—

Notes: 1. Bit 3 is a read-only bit.

2. The user-defined bits (bits 7 to 4, 2) are read/write accessible from the slave processor.

3. Address when accessed from the slave processor.

4. Pin inputs used in access from the host processor.

5. The HA_0 input discriminates between writing of commands and data.

6. Registers in slave addresses H'FFF0 to H'FFFF can only be read or written to when the HIE bit in the system control register (SYSCR) is set to 1.

12.2 Register Descriptions

12.2.1 System Control Register (SYSCR)

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

SYSCR is an 8-bit readable/writable register which controls chip operations. Host interface functions are enabled or disabled by the HIE bit of SYSCR. See section 3.2, System Control Register, for information on other SYSCR bits. SYSCR is initialized to H'09 by an external reset and in the hardware standby modes.

Bit 1—Host Interface Enable (HIE): Enables or disables the host interface. When enabled, the host interface handles host-slave data transfers, operating in slave mode.

Bit 1 **HIE**	**Description**	
0	The host interface is disabled	(Initial value)
1	The host interface is enabled (slave mode)	

12.2.2 Host Interface Control Register (HICR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	IBFIE2	IBFIE1	FGA20E
Initial value	1	1	1	1	1	0	0	0
Slave Read/Write	—	—	—	—	—	R/W	R/W	R/W
Host Read/Write	—	—	—	—	—	—	—	—

HICR is an 8-bit readable/writable register which controls host interface interrupts and the fast A_{20} gate function. HICR is initialized to H'F8 by a reset and in the standby modes.

Bits 7 to 3—Reserved: These bits cannot be modified and are always read as 1.

Bit 2—Input Buffer Full Interrupt Enable 2 (IBFIE2): Enables or disables the IBF2 interrupt to the slave CPU.

Bit 2 IBFIE2	Description	
0	IDR2 input buffer full interrupt is disabled	(Initial value)
1	IDR2 input buffer full interrupt is enabled	

Bit 1— Input Buffer Full Interrupt Enable 1 (IBFIE1): Enables or disables the IBF1 interrupt to the slave CPU.

Bit 1 IBFIE1	Description	
0	IDR1 input buffer full interrupt is disabled	(Initial value)
1	IDR1 input buffer full interrupt is enabled	

Bit 0—Fast Gate A_{20} Enable (FGA20E): Enables or disables the fast A_{20} gate function. When the fast A_{20} gate is disabled, a regular-speed A_{20} gate signal can be implemented by using software to manipulate the $P8_1$ output.

Bit 0 FGA20E	Description	
0	Disables fast A_{20} gate function	(Initial value)
1	Enables fast A_{20} gate function	

12.2.3 Input Data Register 1 (IDR1)

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R	R	R	R	R	R	R	R
Host Read/Write	W	W	W	W	W	W	W	W

IDR1 is an 8-bit read-only register to the slave processor, and an 8-bit write-only register to the host processor. When $\overline{CS}_1$ is low, information on the host data bus is written into IDR1 at the rising edge of $\overline{IOW}$. The HA_0 state is also latched into the C/$\overline{D}$ bit in STR1 to indicate whether the written information is a command or data.

The initial values of IDR1 after a reset and in the standby modes are undetermined.

12.2.4 Output Data Register 1 (ODR1)

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Host Read/Write	R	R	R	R	R	R	R	R

ODR1 is an 8-bit readable/writable register to the slave processor, and an 8-bit read-only register to the host processor. The ODR1 contents are output on the host data bus when HA_0 is low, $\overline{CS}_1$ is low, and $\overline{IOR}$ is low.

The initial values of ODR1 after a reset and in standby mode are undetermined.

12.2.5 Status Register 1 (STR1)

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	C/$\overline{D}$	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

STR1 is an 8-bit register that indicates status information during host interface processing. Bits 3, 1, and 0 are read-only bits to both the host and slave processors.

STR1 is initialized to H'00 by a reset and in the standby modes.

Bits 7 to 4 and Bit 2—Defined by User (DBU): The user can use these bits as necessary.

Bit 3—Command/Data (C/$\overline{D}$): Receives the HA_0 input when the host processor writes to IDR1, and indicates whether IDR1 contains data or a command.

Bit 3 **C/$\overline{D}$**	**Description**	
0	Contents of IDR1 are data	(Initial value)
1	Contents of IDR1 are a command	

Bit 1—Input Buffer Full (IBF): Set to 1 when the host processor writes to IDR1. This bit is an internal interrupt source to the slave processor. IBF is cleared to 0 when the slave processor reads IDR1.

Bit 1 IBF	Description	
0	This bit is cleared when the slave processor reads IDR1	(Initial value)
1	This bit is set when the host processor writes to IDR1	

Bit 0—Output Buffer Full (OBF): Set to 1 when the slave processor writes to ODR1. Cleared to 0 when the host processor reads ODR1.

Bit 0 OBF	Description	
0	This bit is cleared when the host processor reads ODR1	(Initial value)
1	This bit is set when the slave processor writes to ODR1	

Table 12-3 shows the conditions for setting and clearing the STR1 flags.

Table 12-3 Set/Clear Timing for STR1 Flags

Flag	Setting Condition	Clearing Condition
$\text{C}/\overline{\text{D}}$	Rising edge of host’s write signal ($\overline{\text{IOW}}$) when HA_0 is high	Rising edge of host’s write signal ($\overline{\text{IOW}}$) when HA_0 is low
IBF	Rising edge of host’s write signal ($\overline{\text{IOW}}$) when writing to IDR1	Falling edge of slave’s internal read signal ($\overline{\text{RD}}$) when reading IDR1
OBF	Falling edge of slave’s internal write signal ($\overline{\text{WR}}$) when writing to ODR1	Rising edge of host’s read signal ($\overline{\text{IOR}}$) when reading ODR1

12.2.6 Input Data Register 2 (IDR2)

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R	R	R	R	R	R	R	R
Host Read/Write	W	W	W	W	W	W	W	W

IDR2 is an 8-bit read-only register to the slave processor, and an 8-bit write-only register to the host processor. When $\overline{\text{CS}}_2$ is low, information on the host data bus is written into IDR2 at the rising edge of $\overline{\text{IOW}}$. The HA$_0$ state is also latched into the C/$\overline{\text{D}}$ bit in STR2 to indicate whether the written information is a command or data.

The initial values of IDR2 after a reset and in the standby modes are undetermined.

12.2.7 Output Data Register 2 (ODR2)

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Host Read/Write	R	R	R	R	R	R	R	R

ODR2 is an 8-bit read/write register to the slave processor, and an 8-bit read-only register to the host processor. The ODR2 contents are output on the host data bus when HA$_0$ is low, $\overline{\text{CS}}_2$ is low, and $\overline{\text{IOR}}$ is low.

12.2.8 Status Register 2 (STR2)

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	C/$\overline{\text{D}}$	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

STR2 is an 8-bit register that indicates status information during host interface processing. Bits 3, 1, and 0 are read-only bits to both the host and slave processors.

STR2 is initialized to H'00 by a reset and in the standby modes.

Bits 7 to 4 and Bit 2—Defined by User (DBU): The user can use these bits as necessary.

Bit 3—Command/Data (C/$\overline{\text{D}}$): Receives the HA$_0$ input when the host processor writes to IDR2, and indicates whether IDR2 contains data or a command.

Bit 3 **C/$\overline{\text{D}}$**	**Description**	
0	Contents of IDR2 are data	(Initial value)
1	Contents of IDR2 are a command	

Bit 1—Input Buffer Full (IBF): Set to 1 when the host processor writes to IDR2. This bit is an internal interrupt source to the slave processor. IBF is cleared to 0 when the slave processor reads IDR2.

Bit 1 IBF	Description	
0	This bit is cleared when the slave processor reads IDR2	(Initial value)
1	This bit is set when the host processor writes to IDR2	

Bit 0—Output Buffer Full (OBF): Set to 1 when the slave processor writes to ODR2. Cleared to 0 when the host processor reads ODR2.

Bit 0 OBF	Description	
0	This bit is cleared when the host processor reads ODR2	(Initial value)
1	This bit is set when the slave processor writes to ODR2	

Table 12-4 shows the conditions for setting and clearing the STR2 flags.

Table 12-4 Set/Clear Timing for STR2 Flags

Flag	Setting Condition	Clearing Condition
C/$\overline{\text{D}}$	Rising edge of host's write signal ($\overline{\text{IOW}}$) when HA$_0$ is high	Rising edge of host's write signal ($\overline{\text{IOW}}$) when HA$_0$ is low
IBF	Rising edge of host's write signal ($\overline{\text{IOW}}$) when writing to IDR2	Falling edge of slave's internal read signal ($\overline{\text{RD}}$) when reading IDR2
OBF	Falling edge of slave's internal write signal ($\overline{\text{WR}}$) when writing to ODR2	Rising edge of host's read signal ($\overline{\text{IOR}}$) when reading ODR2

12.3 Operation

12.3.1 Host Interface Operation

The host interface is activated by setting the HIE bit (bit 1) to 1 in SYSCR, establishing slave mode. Activation of the host interface (entry to slave mode) appropriates the related I/O lines in port 3 (data), port 4 or 7 (control) and port 4 (host interrupt requests) for interface use.

For host interface read/write timing diagrams, see section 19.3.8, Host Interface Timing.

12.3.2 Control States

Table 12-5 indicates the slave operations carried out in response to host interface signals from the host processor.

Table 12-5 Host Interface Operation

$\overline{\text{CS}}_2$	$\overline{\text{CS}}_1$	$\overline{\text{IOR}}$	$\overline{\text{IOW}}$	HA_0	Operation
1	0	0	0	0	Prohibited
1	0	0	0	1	Prohibited
1	0	0	1	0	Data read from output data register 1 (ODR1)
1	0	0	1	1	Status read from status register 1 (STR1)
1	0	1	0	0	Data write to input data register 1 (IDR1)
1	0	1	0	1	Command write to input data register 1 (IDR1)
1	0	1	1	0	Idle state
1	0	1	1	1	Idle state
0	1	0	0	0	Prohibited
0	1	0	0	1	Prohibited
0	1	0	1	0	Data read from output data register 2 (ODR2)
0	1	0	1	1	Status read from status register 2 (STR2)
0	1	1	0	0	Data write to input data register 2 (IDR2)
0	1	1	0	1	Command write to input data register 2 (IDR2)
0	1	1	1	0	Idle state
0	1	1	1	1	Idle state

12.3.3 A_{20} Gate

The A_{20} gate signal can mask address A_{20} to emulate an addressing mode used by personal computers with an 8086*-family CPU. In slave mode, a regular-speed A_{20} gate signal can be output under software control, or a fast A_{20} gate signal can be output under hardware control. Fast A_{20} gate output is enabled by setting the FGA20E bit (bit 0) to 1 in HICR (H'FFF0).

Note: * Intel microprocessor.

Regular A_{20} Gate Operation: Output of the A_{20} gate signal can be controlled by an H'D1 command followed by data. When the slave processor receives data, it normally uses an interrupt routine activated by the IBF1 interrupt to read IDR1. If the data follows an H'D1 command, software copies bit 1 of the data and outputs it at the gate A_{20} pin ($P4_7/GA_{20}$).

Fast A_{20} Gate Operation: When the FGA20E bit is set to 1, $P4_7/GA_{20}$ is used for output of a fast A_{20} gate signal. Bit $P4_7$DDR must be set to 1 to assign this pin for output. The initial output from this pin will be a logic 1, which is the initial DR value. Afterward, the host processor can manipulate the output from this pin by sending commands and data. This function is available only when register IDR1 is accessed using $\overline{CS}_1$. Slave logic decodes the commands input from the host processor. When an H'D1 host command is detected, bit 1 of the data following the host command is output from the GA_{20} output pin. This operation does not depend on software or interrupts, and is faster than the regular processing using interrupts. Table 12-6 lists the conditions that set and clear GA_{20} ($P4_7$). Figure 12-2 describes the GA_{20} output in flowchart form. Table 12-7 indicates the GA_{20} output signal values.

Table 12-6 GA_{20} ($P4_7$) Set/Clear Timing

Pin Name	Setting Condition	Clearing Condition
GA_{20} ($P4_7$)	Rising edge of the host's write signal ($\overline{IOW}$) when bit 1 of the written data is 1 and the data follows an H'D1 host command	Rising edge of the host's write signal ($\overline{IOW}$) when bit 1 of the written data is 0 and the data follows an H'D1 host command

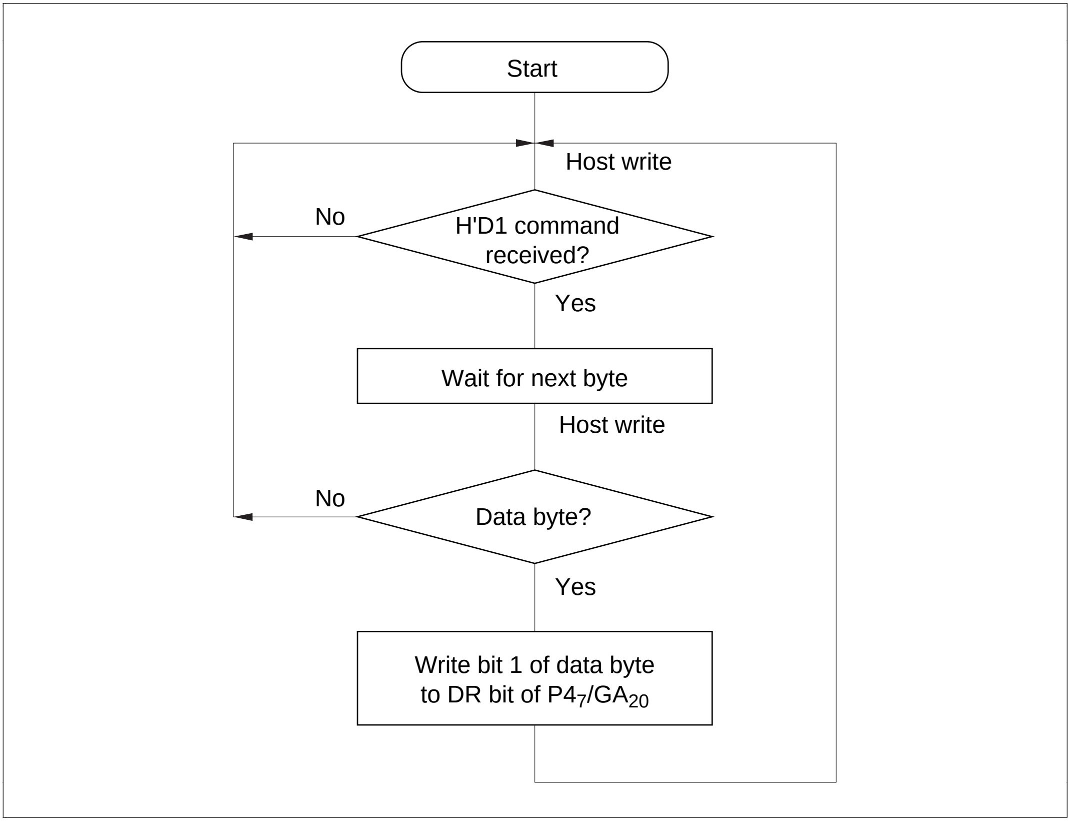


Figure 12-2 GA_{20} Output

Table 12-7 Fast A_{20} Gate Output Signal

HA_0	Data/Command	Internal CPU Interrupt Flag	GA_{20} ($P4_7$)	Remarks
1	H'D1 command	0	Q	Turn-on sequence
0	“1” data*1	0	1	
1	H'FF command	0	Q (1)	
1	H'D1 command	0	Q	Turn-off sequence
0	“0” data*2	0	0	
1	H'FF command	0	Q (0)	
1	H'D1 command	0	Q	Short turn-on sequence
0	“1” data*1	0	1	
1/0	Command other than H'FF and H'D1	1	Q (1)	
1	H'D1 command	0	Q	Short turn-off sequence
0	“0” data*2	0	0	
1/0	Command other than H'FF and H'D1	1	Q (0)	
1	H'D1 command	0	Q	Cancelled sequence
1	Command other than H'D1	1	Q	
1	H'D1 command	0	Q	Retriggered sequence
1	H'D1 command	0	Q	
1	H'D1 command	0	Q	Consecutively executed sequences
0	Any data	0	1/0	
1	H'D1 command	0	Q (1/0)	

Notes: 1. Arbitrary data with bit 1 set to 1.

2. Arbitrary data with bit 1 cleared to 0.

12.4 Interrupts

12.4.1 IBF1, IBF2

The host interface can request two interrupts to the slave CPU: IBF1 and IBF2. They are input buffer full interrupts for input data registers IDR1 and IDR2 respectively. Each interrupt is enabled when the corresponding enable bit is set (table 12-8).

Table 12-8 Input Buffer Full Interrupts

Interrupt	Description
IBF1	Requested when IBFIE1 is set to 1 and IDR1 is full
IBF2	Requested when IBFIE2 is set to 1 and IDR2 is full

12.4.2 HIRQ$_{11,}$ HIRQ$_1$, and HIRQ$_{12}$

In slave mode (when HIE = 1 in SYSCR), three bits in the port 4 data register (P4DR) can be used as host interrupt request latches.

These three P4DR bits are cleared to 0 by the host processor's read signal ($\overline{\text{IOR}}$). If $\overline{\text{CS}}_1$ and HA$_0$ are low, when $\overline{\text{IOR}}$ goes low and the host reads ODR1, HIRQ$_1$ and HIRQ$_{12}$ are cleared to 0. If $\overline{\text{CS}}_2$ and HA$_0$ are low, when $\overline{\text{IOR}}$ goes low and the host reads ODR2, HIRQ$_{11}$ is cleared to 0. To generate a host interrupt request, normally on-chip software writes 1 to the corresponding bit. In processing the interrupt, the host's interrupt-handling routine reads the output data register (ODR1 or ODR2), and this clears the host interrupt latch to 0.

Table 12-9 indicates how these bits are set and cleared. Figure 12-3 shows the processing in flowchart form.

Table 12-9 Host Interrupt Set/Clear Conditions

Host Interrupt Signal	Setting Condition	Clearing Condition
HIRQ$_{11}$ (P4$_3$)	Slave CPU reads 0 from P4DR bit 3, then writes 1	Slave CPU writes 0 in P4DR bit 3, or host reads output data register 2
HIRQ$_1$ (P4$_4$)	Slave CPU reads 0 from P4DR bit 4, then writes 1	Slave CPU writes 0 in P4DR bit 4, or host reads output data register 1
HIRQ$_{12}$ (P4$_5$)	Slave CPU reads 0 from P4DR bit 5, then writes 1	Slave CPU writes 0 in P4DR bit 5, orhost reads output data register 1

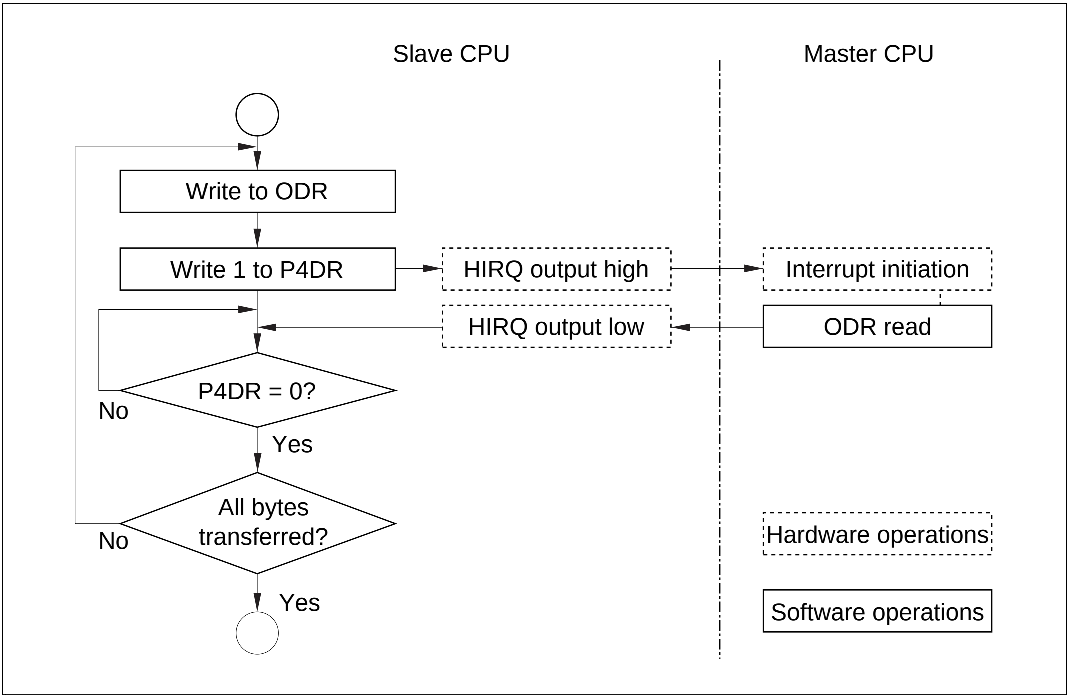


Figure 12-3 HIRQ Output Flowchart

12.5 Application Note

The host interface provides buffering of asynchronous data from the host and slave processors, but an interface protocol must be followed to implement necessary functions and avoid data contention. For example, if the host and slave processors try to access the same input or output data register simultaneously, the data will be corrupted. Interrupts can be used to design a simple and effective protocol.

Section 13 RAM

13.1 Overview

The H8/3502 have 512 bytes. The on-chip RAM is connected to the CPU by a 16-bit data bus. Both byte and word access to the on-chip RAM are performed in two states, enabling rapid data transfer and instruction execution.

The on-chip RAM occupies addresses H'FD80 to H'FF7F in the chip's address space. The RAM enable bit (RAME) in the system control register (SYSCR) can enable or disable the on-chip RAM.

13.2 Block Diagram

Figure 13-1 is a block diagram of the on-chip RAM.

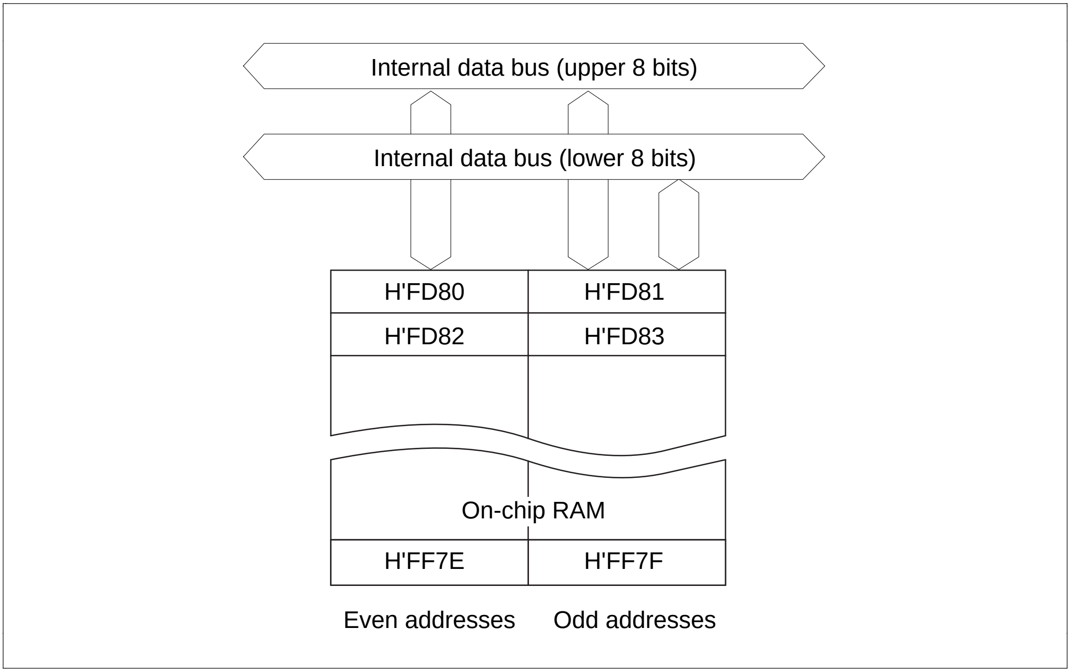


Figure 13-1 Block Diagram of On-Chip RAM

13.3 RAM Enable Bit (RAME)

The on-chip RAM is enabled or disabled by the RAME (RAM Enable) bit in the system control register (SYSCR). Table 13-1 lists information about the system control register.

Table 13-1 System Control Register

Name	Abbreviation	R/W	Initial value	Address
System control register	SYSCR	R/W	H'09	H'FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

The only bit in the system control register that concerns the on-chip RAM is the RAME bit. See section 3.2, System Control Register for the other bits.

Bit 0—RAM Enable (RAME): This bit enables or disables the on-chip RAM.

The RAME bit is initialized to 1 on the rising edge of the $\overline{\text{RES}}$ signal, so a reset enables the on-chip RAM. The RAME bit is not initialized in the software standby mode.

Bit 7 RAME	Description	
0	On-chip RAM is disabled	
1	On-chip RAM is enabled	(Initial value)

13.4 Operation

13.4.1 Expanded Modes (Modes 1 and 2)

If the RAME bit is set to 1, accesses to addresses H'FD80 to H'FF7F are directed to the on-chip RAM. If the RAME bit is cleared to 0, accesses to these addresses are directed to the external data bus.

13.4.2 Single-Chip Mode (Mode 3)

If the RAME bit is set to 1, accesses to addresses H'FD80 to H'FF7F are directed to the on-chip RAM. If the RAME bit is cleared to 0, the on-chip RAM cannot be accessed. Attempted write access has no effect. Attempted read access always results in H'FF data being read.

Note: Initial RAM data are unknown.
Be sure to initialize when use them as control bits.

Section 14 ROM

14.1 Overview

The H8/3502 has 16 kbytes of high-speed, on-chip ROM. The on-chip ROM is connected to the CPU via a 16-bit data bus. Both byte data and word data are accessed in two states, enabling rapid data transfer and instruction fetching.

Enabling or disabling of the on-chip ROM is determined by the inputs at the mode pins (MD1 and MD0) as shown in table 14-1.

Table 14-1 On-Chip ROM Usage in Each MCU Mode

	Mode Pins		
Mode	**MD_1**	**MD_0**	**On-Chip ROM**
Mode 1 (expanded mode)	0	1	Disabled (external addresses)
Mode 2 (expanded mode)	1	0	Enabled
Mode 3 (single-chip mode)	1	1	Enabled

14.1.1 Block Diagram

Figure 14-1 is a block diagram of the on-chip ROM.

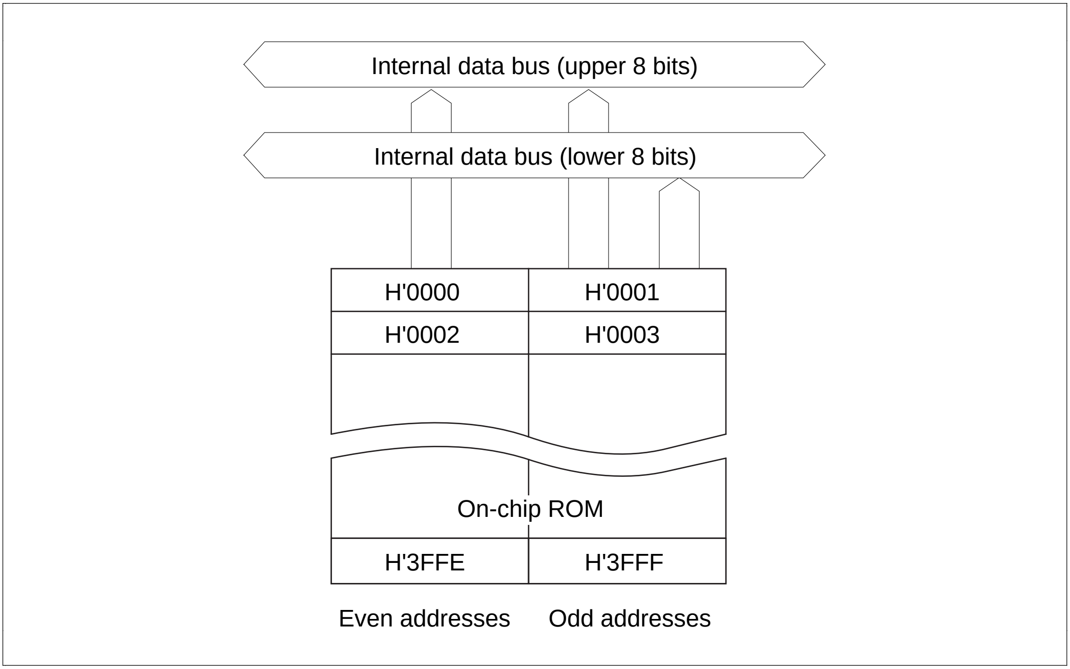


Figure 14-1 Block Diagram of On-Chip ROM

Section 15 Power-Down State

15.1 Overview

The H8/3502 has a sleep mode that reduces power consumption by stopping CPU functions. Although two standby modes can be set in addition to sleep mode, use of the standby modes is not recommended since a guaranteed value is not set for current dissipation in these modes.

1. Sleep mode
2. Software standby mode
3. Hardware standby mode

Table 15-1 lists the conditions for entering and leaving the power-down modes. It also indicates the status of the CPU, on-chip supporting modules, etc., in each power-down mode.

Table 15-1 Power-Down State

Mode	Entering Procedure	Clock	CPU	CPU Reg's.	Sup. Mod.*	RAM	I/O Ports	Exiting Methods
Sleep mode	Execute SLEEP instruction	Run	Halt	Held	Run	Held	Held	• Interrupt • $\overline{\text{RES}}$ • $\overline{\text{STBY}}$
Software standby mode	Set SSBY bit in SYSCR to 1, then execute SLEEP instruction	Halt	Halt	Held	Halt and initialized	Held	Held	• $\overline{\text{NMI}}$ • $\overline{\text{IRQ}_0}$–$\overline{\text{IRQ}_2}$ • $\overline{\text{KEYIN}_0}$–$\overline{\text{KEYIN}_7}$ • $\overline{\text{STBY}}$ • $\overline{\text{RES}}$
Hardware standby mode	Set $\overline{\text{STBY}}$ pin to low level	Halt	Halt	Not held	Halt and initialized	Held	High impedance state	• $\overline{\text{STBY}}$ high, then $\overline{\text{RES}}$ low → high

Notes: 1. SYSCR: System control register

2. SSBY: Software standby bit

* On-chip supporting modules.

15.1.1 System Control Register (SYSCR)

Bits 7 to 4 of the system control register (SYSCR) concern the power-down state. Specifically, they concern the software standby mode.

Table 15-2 lists the attributes of the system control register.

Table 15-2 System Control Register

Name	Abbreviation	R/W	Initial Value	Address
System control register	SYSCR	R/W	H'09	H'FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit 7—Software Standby (SSBY): This bit enables or disables the transition to the software standby mode.

On recovery from the software standby mode by an external interrupt SSBY remains set to 1. To clear this bit, software must write a 0.

Bit 7 SSBY	Description	
0	The SLEEP instruction causes a transition to the sleep mode	(Initial value)
1	The SLEEP instruction causes a transition to the software standby mode	

Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0): These bits select the clock settling time when the chip recovers from the software standby mode by means of an external interrupt. During the selected time, the clock oscillator runs but clock pulses are not supplied to the CPU or the on-chip supporting modules. Refer to table 15-3 to select an appropriate settling time for the operating frequency.

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8192 states	(Initial value)
0	0	1	Settling time = 16384 states	
0	1	0	Settling time = 32768 states	
0	1	1	Settling time = 65536 states	
1	0	—	Settling time = 131072 states	
1	1	—	Use prohibited	

15.2 Sleep Mode

15.2.1 Transition to Sleep Mode

When the SSBY bit in the system control register is cleared to 0, execution of the SLEEP instruction causes a transition from the program execution state to the sleep mode. After executing the SLEEP instruction, the CPU halts, but the contents of its internal registers remain unchanged. The on-chip supporting modules continue to operate normally.

15.2.2 Exit from Sleep Mode

The chip wakes up from the sleep mode when it receives an internal or external interrupt request, or a low input at the $\overline{\text{RES}}$ or $\overline{\text{STBY}}$ pin.

(1) Wake-Up by Interrupt: An interrupt releases the sleep mode and starts the CPU’s interrupt-handling sequence.

If an interrupt from an on-chip supporting module is disabled by the corresponding enable/disable bit in the module’s control register, the interrupt cannot be requested, so it cannot wake the chip up. Similarly, the CPU cannot be awoken by an interrupt other than NMI if the I (interrupt mask) bit in CCR (the condition code register) is set when the SLEEP instruction is executed.

(2) Wake-Up by $\overline{\textbf{RES}}$ pin: When the $\overline{\text{RES}}$ pin goes low, the chip exits from the sleep mode to the reset state.

(3) Wake-Up by $\overline{\textbf{STBY}}$ pin: When the $\overline{\text{STBY}}$ pin goes low, the chip exits from the sleep mode to the hardware standby mode.

15.3 Software Standby Mode

15.3.1 Transition to Software Standby Mode

To enter software standby mode, set the standby bit (SSBY) in the system control register (SYSCR) to 1, then execute the SLEEP instruction.

In software standby mode, the system clock stops and chip functions halt, including both CPU functions and the functions of the on-chip supporting modules. The on-chip supporting modules and their registers are reset to their initial states, but as long as a minimum necessary voltage supply is maintained, the contents of the CPU registers and on-chip RAM remain unchanged.

15.3.2 Exit from Software Standby Mode

The chip can be brought out of the software standby mode by an input at one of the following pins: $\overline{\text{NMI}}$, $\overline{\text{IRQ}_0}$, to $\overline{\text{IRQ}_2}$, $\overline{\text{KEYIN}_0}$ to $\overline{\text{KEYIN}_7}$, $\overline{\text{RES}}$, or $\overline{\text{STBY}}$.

(1) Recovery by External Interrupt: When an NMI, IRQ_0, IRQ_1, IRQ_2 or key-sense interrupt (IRQ_6) request signal is received, the clock oscillator begins operating. After the waiting time set in the system control register (bits STS2 to STS0), clock pulses are supplied to the CPU and on-chip supporting modules. The CPU executes the interrupt-handling sequence for the requested interrupt, then returns to the instruction after the SLEEP instruction.

See Section 15.1.1, System Control Register, for information about the STS bits.

(2) Recovery by $\overline{\textbf{RES}}$ Pin: When the $\overline{\text{RES}}$ pin goes low, the clock oscillator starts. Next, when the $\overline{\text{RES}}$ pin goes high, the CPU begins executing the reset sequence.

The $\overline{\text{RES}}$ pin must be held low long enough for the clock to stabilize.

(3) Recovery by $\overline{\textbf{STBY}}$ Pin: When the $\overline{\text{STBY}}$ pin goes low, the chip exits from the software standby mode to the hardware standby mode.

15.3.3 Clock Settling Time for Exit from Software Standby Mode

Set bits STS2 to STS0 in SYSCR as follows:

- Crystal oscillator

  Set STS2 to STS0 for a settling time of at least 8 ms. Table 15-3 lists the settling times selected by these bits at several clock frequencies.

- External clock

  The STS bits can be set to any value. Normally, the minimum time (STS2 = STS1 = STS0 = 0) is recommended.

Table 15-3 Times Set by Standby Timer Select Bits (Unit: ms)

			Settling Time	System Clock Frequency (MHz)			
STS2	**STS1**	**STS0**	**(States)**	**10**	**8**	**6**	**4**
0	0	0	8,192	0.8	1.0	1.4	2.0
0	0	1	16,384	1.6	2.0	2.7	4.1
0	1	0	32,768	3.3	4.1	5.5	**8.2**
0	1	1	65,536	6.6	**8.2**	**10.9**	16.4
1	0	0	131,072	**13.1**	16.4	21.8	32.8

Notes: 1. All times are in milliseconds.

2. Recommended values are printed in boldface.

15.3.4 Sample Application of Software Standby Mode

In this example the chip enters the software standby mode when $\overline{\text{NMI}}$ goes low and exits when $\overline{\text{NMI}}$ goes high, as shown in figure 15-1.

The NMI edge bit (NMIEG) in the system control register is originally cleared to 0, selecting the falling edge. When $\overline{\text{NMI}}$ goes low, the $\overline{\text{NMI}}$ interrupt handling routine sets NMIEG to 1 (selecting the rising edge), sets SSBY to 1, then executes the SLEEP instruction. The chip enters the software standby mode. It recovers from the software standby mode on the next rising edge of $\overline{\text{NMI}}$.

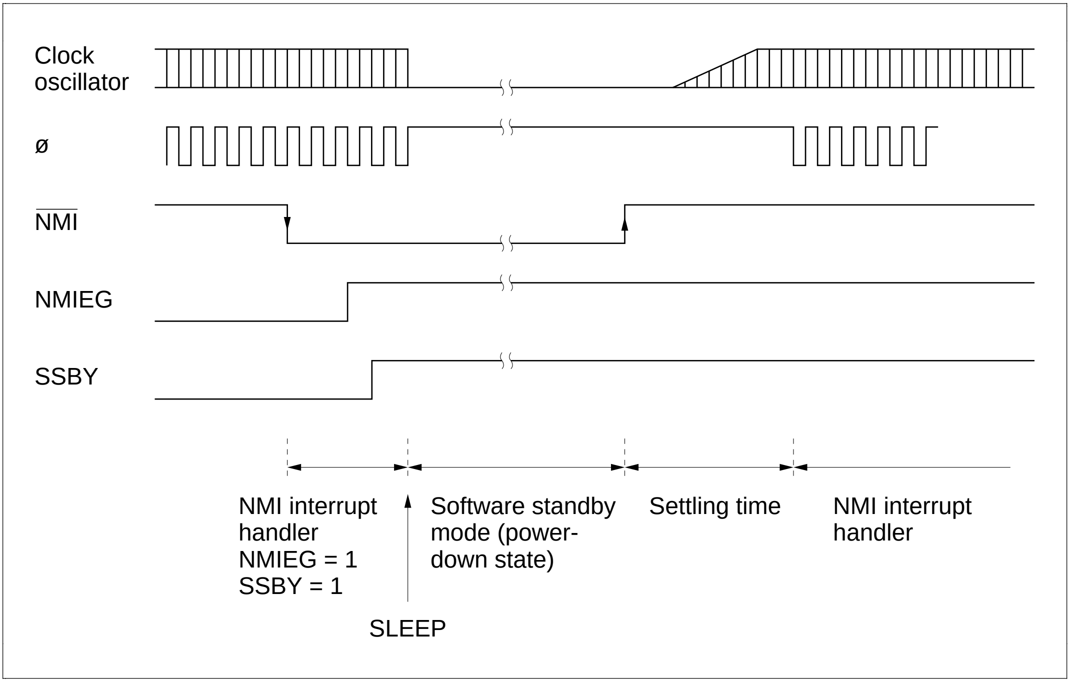


Figure 15-1 Software Standby Mode NMI Timing (Example)

15.3.5 Note on Current Dissipation

The I/O ports remain in their current states in software standby mode. If a port is in the high output state, it continues to dissipate power in proportion to the output current.

15.4 Hardware Standby Mode

15.4.1 Transition to Hardware Standby Mode

Regardless of its current state, the chip enters the hardware standby mode whenever the $\overline{\text{STBY}}$ pin goes low.

In hardware standby mode, the functions of the CPU and all on-chip supporting modules are halted. The on-chip supporting modules are placed in the reset state, but on-chip RAM data is retained provided the minimum necessary voltage is supplied. I/O ports go to the high-impedance state.

Notes: 1. The RAME bit in the system control register should be cleared to 0 before the $\overline{\text{STBY}}$ pin goes low, to disable the on-chip RAM during the hardware standby mode.

2. Do not change the inputs at the mode pins (MD_1, MD_0) during hardware standby mode. Be particularly careful not to let both mode pins go low in hardware standby mode, since that places the chip in PROM mode and increases current drain.

15.4.2 Recovery from Hardware Standby Mode

Recovery from the hardware standby mode requires inputs at both the $\overline{\text{STBY}}$ and $\overline{\text{RES}}$ pins.

When the $\overline{\text{STBY}}$ pin goes high the clock oscillator begins running. The $\overline{\text{RES}}$ pin should be low at this time and should be held low long enough for the clock to stabilize. When the $\overline{\text{RES}}$ pin changes from low to high, the reset sequence is executed and the chip returns to the program execution state.

15.4.3 Timing Relationships

Figure 15-2 shows the timing relationships in the hardware standby mode.

In the sequence shown, first $\overline{\text{RES}}$ goes low, then $\overline{\text{STBY}}$ goes low, at which point the chip enters the hardware standby mode. To recover, first $\overline{\text{STBY}}$ goes high, then after the clock settling time, $\overline{\text{RES}}$ goes high.

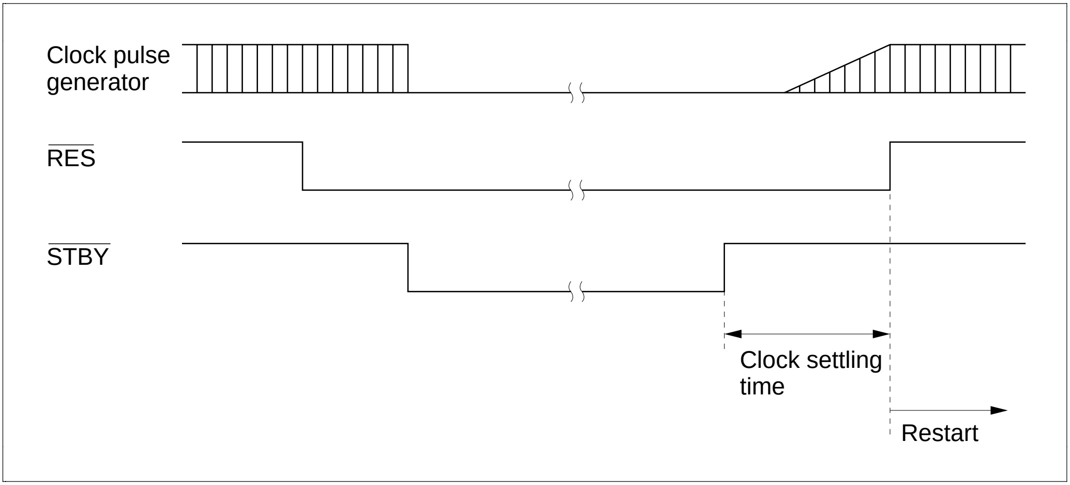


Figure 15-2 Hardware Standby Mode Timing

Section 16 Electrical Specifications

16.1 Absolute Maximum Ratings

Table 16-1 lists the absolute maximum ratings.

Table 16-1 Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Supply voltage	V_{CC}	–0.3 to +7.0	V
Input voltage	V_{in}	–0.3 to V_{CC} + 0.3	V
Operating temperature	T_{opr}	–20 to +75	°C
Storage temperature	T_{stg}	–55 to +125	°C

Note: Exceeding the absolute maximum ratings shown in table 16-1 can permanently damage the chip.

16.2 Electrical Characteristics

16.2.1 DC Characteristics

DC characteristics are shown in table 16-2, and allowable current output values in table 16-3.

Table 16-2 DC Characteristics **— Preliminary —**

Conditions: V_{CC} = 5.0 V ±10%, V_{SS} = 0 V, Ta = –20°C to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	$P7_7$, $P7_5$ to $P7_0$*2, FTCI, FTI, $TMRI_0$, $TMRI_1$, $TMCI_0$, $TMCI_1$, $\overline{KEYIN}_7$ to $\overline{KEYIN}_0$ $\overline{IRQ}_2$ to $\overline{IRQ}_0$ (1)	V_T^-	1.0	—	—	V	
		V_T^+	—	—	V_{CC} × 0.7		
		$V_T^+ – V_T^-$	0.4	—	—		
Input high voltage	$\overline{RES}$, $\overline{STBY}$, MD_1, MD_0, EXTAL, $\overline{NMI}$ (2)	V_{IH}	V_{CC} – 0.7	—	V_{CC} + 0.3	V	
	All input pins other than (1) and (2) above		2.0	—	V_{CC} + 0.3		
Input low voltage	$\overline{RES}$, $\overline{STBY}$, MD_1, MD_0 (3)	V_{IL}	–0.3	—	0.5	V	
	All input pins other than (1) and (3) above		–0.3	—	0.8		
Output high voltage	All output pins	V_{OH}	V_{CC} – 0.5	—	—	V	I_{OH} = –200 µA
			3.5	—	—		I_{OH} = –1.0 mA
Output low voltage	All output pins	V_{OL}	—	—	0.4	V	I_{OL} = 1.6 mA
	$P1_7$ to $P1_0$, $P2_7$ to $P2_0$, $P3_7$ to $P3_0$		—	—	1.0		I_{OL} = 10.0 mA

Table 16-2 DC Characteristics (cont) — Preliminary —

Conditions: V_{CC} = 5.0 V ±10%, V_{SS} = 0 V, Ta = –20°C to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input leakage current	$\overline{\text{RES}}$	$\lvert I_{in} \rvert$	—	—	10.0	µA	Vin = 0.5 V to V_{CC} – 0.5 V
	$\overline{\text{STBY}}$, $\overline{\text{NMI}}$, MD_1, MD_0		—	—	1.0		
Leakage current in three-state (off state)	Ports 1 to 7	$\lvert I_{TS1} \rvert$	—	—	1.0	µA	Vin = 0.5 V to V_{CC} – 0.5 V
Input pull-up MOS current	Ports 1 to 3	$-I_p$	30	—	250	µA	Vin = 0 V
	$P7_3$ to $P7_0$, $P6_3$ to $P6_0$		60	—	500		
Input capaci-tance	$\overline{\text{RES}}$ (4)	C_{in}	—	—	60	pF	Vin = 0 V, f = 1 MHz, Ta = 25°C
	$\overline{\text{NMI}}$		—	—	50		
	$P7_3$ to $P7_0$		—	—	20		
	All input pins other than (4)		—	—	15		
Current dissipa-tion*1	Normal operation	I_{CC}	—	23	40	mA	f = 10 MHz
	Sleep mode		—	15	25		f = 10 MHz

Notes: 1. Value when $V_{IH\ min} = V_{CC} - 0.5$ V, $V_{IL\ max}$ = 0.5 V, all output pins are unloaded, and input MOS pull-ups are off.

2. $P7_7$ and $P7_5$ to $P7_0$ do not include HA_0, $\overline{\text{IOW}}$, $\overline{\text{CS}}_1$, and $\overline{\text{WAIT.}}$

Table 16-3 Allowable Output Current Values

Conditions: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, Ta = –20°C to +75°C

Item		Symbol	Min	Typ	Max	Unit
Allowable output low current (per pin)	Ports 1, 2 and 3	I_{OL}	—	—	10	mA
	Other output pins		—	—	2	
Allowable output low current (total)	Ports 1, 2 and 3 total	ΣI_{OL}	—	—	80	mA
	Total of all output		—	—	120	
Allowable output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Allowable output high current (total)	Total of all output	$\Sigma -I_{OH}$	—	—	40	mA

Note: To avoid degrading the reliability of the chip, be careful not to exceed the output current values in table 16-3. In particular, when driving a Darlington transistor or LED direrctly, be sure to insert a current-limiting resistor in the output path. See figures 16-1 and 16-2.

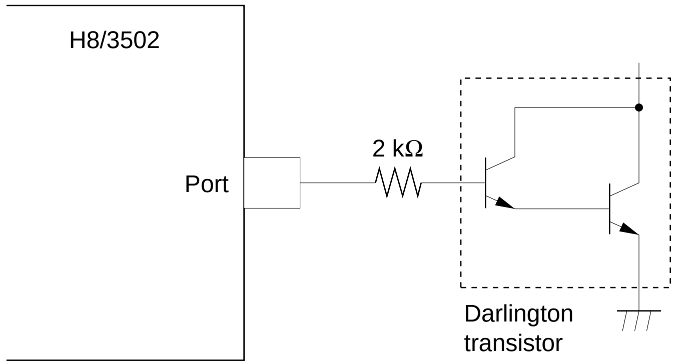


Figure 16-1 Example of Circuit for Driving a Darlington Transistor

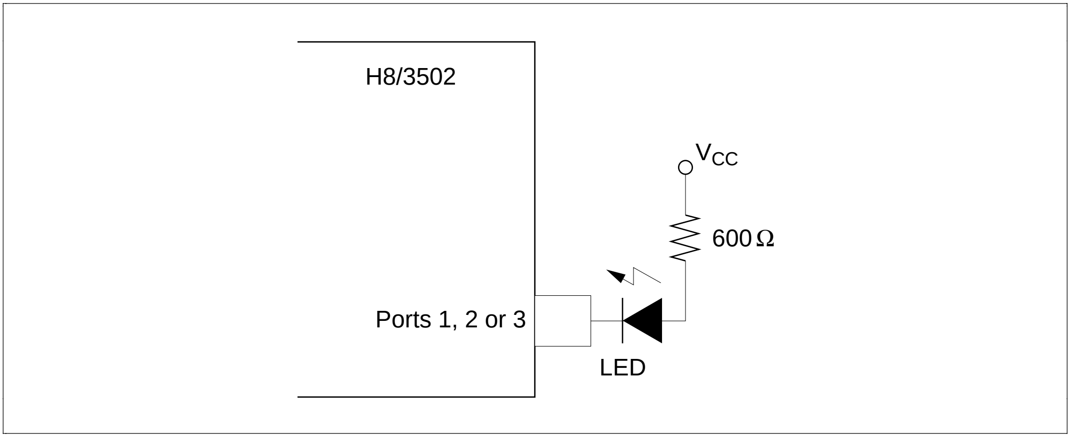


Figure 16-2   Example of Circuit for Driving an LED

16.2.2 AC Characteristics

The AC characteristics are listed in five tables. Bus timing parameters are given in table 16-4, control signal timing parameters in table 16-5, timing parameters of the on-chip supporting modules in table 16-6, External Clock Output Settling Delay Time in table 16-7.

Table 16-4 Bus Timing **— Preliminary —**

Condition: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, ø = 4.0 MHz to maximum operating frequency, Ta = –20°C to +75°C

		10 MHz			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
Clock cycle time	t_{cyc}	100	250	ns	Fig. 16-4
Clock pulse width low	t_{CL}	35	—		
Clock pulse width high	t_{CH}	35	—		
Clock rise time	t_{Cr}	—	15		
Clock fall time	t_{Cf}	—	15		
Address delay time	t_{AD}	—	50		
Address hold time	t_{AH}	20	—		
Address strobe delay time	t_{ASD}	—	40		
Write strobe delay time	t_{WSD}	—	50		
Strobe delay time	t_{SD}	—	50		
Write strobe pulse width*	t_{WSW}	120	—		
Address setup time 1*	t_{AS1}	15	—		
Address setup time 2*	t_{AS2}	65	—		
Read data setup time	t_{RDS}	35	—		
Read data hold time*	t_{RDH}	0	—		
Read data access time*	t_{ACC}	—	170		
Write data delay time	t_{WDD}	—	75		
Write data setup time	t_{WDS}	5	—		
Write data hold time	t_{WDH}	20	—		
Wait setup time	t_{WTS}	40	—		Fig. 16-5
Wait hold time	t_{WTH}	10	—		

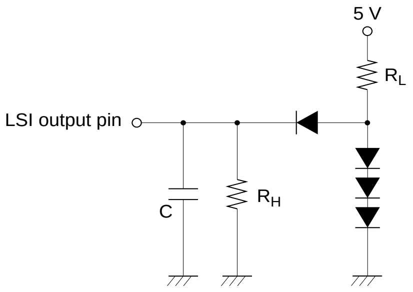
Note: * Values at maximum operating frequency

Table 16-5 Control Signal Timing

Condition: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, ø = 4.0 MHz to maximum operating frequency, Ta = –20°C to +75°C

		10 MHz			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
$\overline{\text{RES}}$ setup time	t_{RESS}	200	—	ns	Fig. 16-6
$\overline{\text{RES}}$ pulse width	t_{RESW}	10	—	t_{cyc}	
$\overline{\text{NMI}}$ setup time ($\overline{\text{NMI}}$, $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_2}$, $\overline{\text{IRQ}_6}$)	t_{NMIS}	150	—	ns	Fig. 16-7
$\overline{\text{NMI}}$ hold time ($\overline{\text{NMI}}$, $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_2}$, $\overline{\text{IRQ}_6}$)	t_{NMIH}	10	—		
Interrupt pulse width for recovery from software standby mode ($\overline{\text{NMI}}$, $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_2}$, $\overline{\text{IRQ}_6}$)	t_{NMIW}	200	—		
Crystal oscillator settling time (reset)	t_{OSC1}	20	—	ms	Fig. 16-8
Crystal oscillator settling time (software standby)	t_{OSC2}	8	—		Fig. 16-9

Measurement Conditions for AC Characteristics



C = 90 pF: P1, P2, P3, $P4_6$, P6, P7
30 pF: P4 (except $P4_6$), P5
R_L = 2.4 kΩ
R_H = 12 kΩ

Input/output timing measurement levels
Low: 0.8 V
High: 2.0 V

Figure 16-3 Test Conditions for AC Characteristics

Table 16-6 Timing Conditions of On-Chip Supporting Modules **— Preliminary —**

Condition: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, ø = 4.0 MHz to maximum operating frequency, Ta = –20°C to +75°C

			10 MHz			
Item		**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
FRT	Timer output delay time	t_{FTOD}	—	100	ns	Fig. 16-10
	Timer input setup time	t_{FTIS}	50	—		
	Timer clock input setup time	t_{FTCS}	50	—		Fig. 16-11
	Timer clock pulse width	t_{FTCWH} t_{FTCWL}	1.5	—	t_{cyc}	
TMR	Timer output delay time	t_{TMOD}	—	100	ns	Fig. 16-12
	Timer reset input setup time	t_{TMRS}	50	—		Fig. 16-14
	Timer clock input setup time	t_{TMCS}	50	—		Fig. 16-13
	Timer clock pulse width (single edge)	t_{TMCWH} t_{TMCWL}	1.5	—	t_{cyc}	
	Timer clock pulse width (both edges)		2.5	—		
SCI	Input clock cycle (Async)	t_{Scyc}	4	—	t_{cyc}	Fig. 16-15
	(Sync)		6	—		
	Transmit data delay time (Sync)	t_{TXD}	—	100	ns	
	Receive data setup time (Sync)	t_{RXS}	100	—		
	Receive data hold time (Sync)	t_{RXH}	100	—		
	Input clock pulse width	t_{SCKW}	0.4	0.6	t_{Scyc}	Fig. 16-16

Table 16-6 Timing Conditions of On-Chip Supporting Modules (cont) — Preliminary —

Condition: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, ø = 4.0 MHz to maximum operating frequency,
Ta = –20°C to +75°C

			10 MHz			
Item		**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
PORT	Output data delay time	t_{PWD}	—	100	ns	Fig. 16-17
	Input data setup time	t_{PRS}	50	—		
	Input data hold time	t_{PRH}	50	—		
HIF read cycle	$\overline{\text{CS}}$/HA$_0$ setup time	t_{HAR}	10	—	ns	Fig. 16-18
	$\overline{\text{CS}}$/HA$_0$ hold time	t_{HRA}	10	—		
	$\overline{\text{IOR}}$ pulse width	t_{HRPW}	120	—		
	HDB delay time	t_{HRD}	—	100		
	HDB hold time	t_{HRF}	0	25		
	HIRQ delay time	t_{HIRQ}	—	120		
HIF write cycle	$\overline{\text{CS}}$/HA$_0$ setup time	t_{HAW}	10	—	ns	Fig. 16-19
	$\overline{\text{CS}}$/HA$_0$ hold time	t_{HWA}	10	—		
	$\overline{\text{IOW}}$ pulse width	t_{HWPW}	60	—		
	HDB setup time	t_{HDW}	30	—		
	HDB hold time	t_{HWD}	15	—		
	GA$_{20}$ delay time	t_{HGA}	—	90		

Table 16-7 External Clock Output Settling Delay Time — Preliminary —

Conditions: V_{CC} = 4.5 V to 5.5 V, V_{SS} = 0 V, Ta = –20°C to +75°C

Item	**Symbol**	**Min**	**Max**	**Unit**	**Notes**
External clock output settling delay time	t_{DEXT}*	500	—	µs	Figure 16-20

Note: * t_{DEXT} includes a 10 t_{cyc} $\overline{\text{RES}}$ pulse width (t_{RESW}).

16.3 MCU Operational Timing

This section provides the following timing charts:

16.3.1	Bus Timing	Figures 16-4 and 16-5
16.3.2	Control Signal Timing	Figures 16-6 to 16-9
16.3.3	16-Bit Free-Running Timer Timing	Figures 16-10 and 16-11
16.3.4	8-Bit Timer Timing	Figures 16-12 to 16-14
16.3.5	Serial Communication Interface Timing	Figures 16-15 and 16-16
16.3.6	I/O Port Timing	Figure 16-17
16.3.7	Host Interface Timing	Figure 16-18 and 16-19
16.3.8	External Clock Ouptput Timing	Figure 16-20

16.3.1 Bus Timing

(1) Basic Bus Cycle (without Wait States) in Expanded Modes

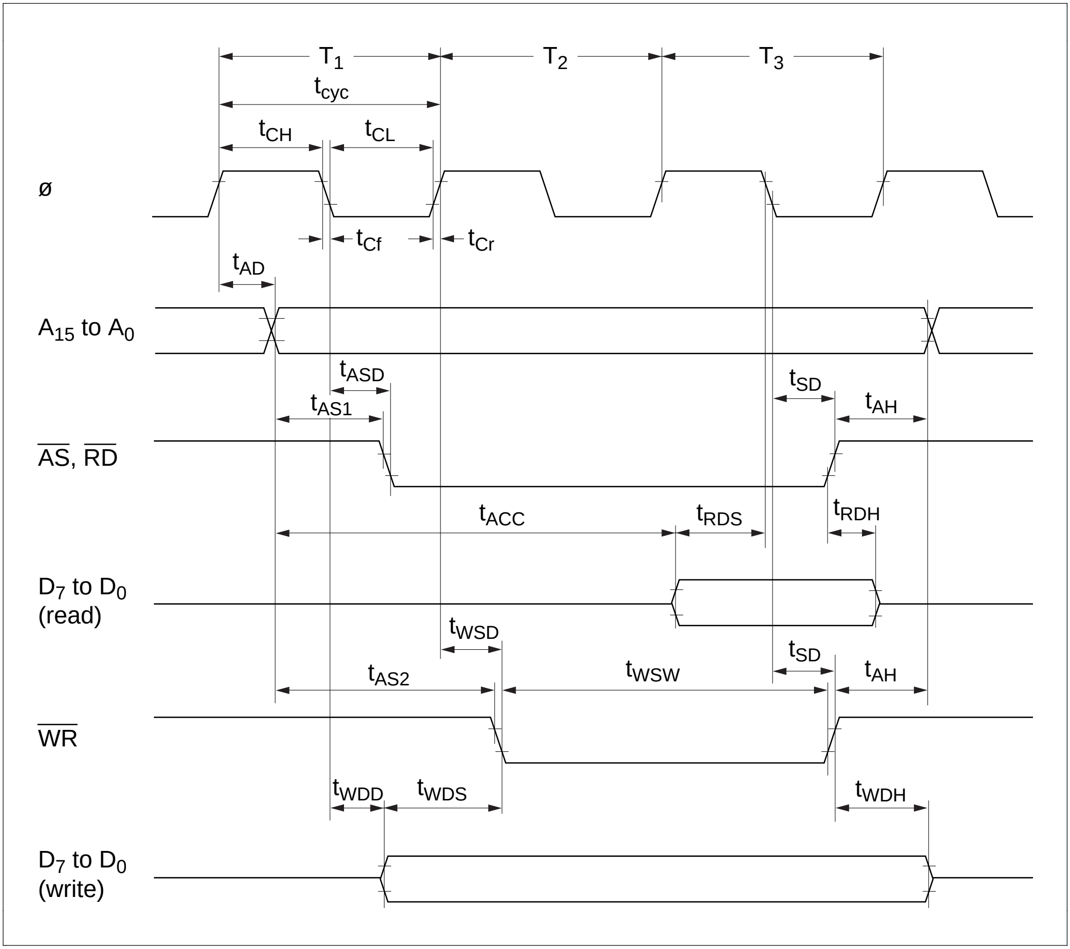


Figure 16-4 Basic Bus Cycle (without Wait States) in Expanded Modes

(2) Basic Bus Cycle (with 1 Wait State) in Expanded Modes

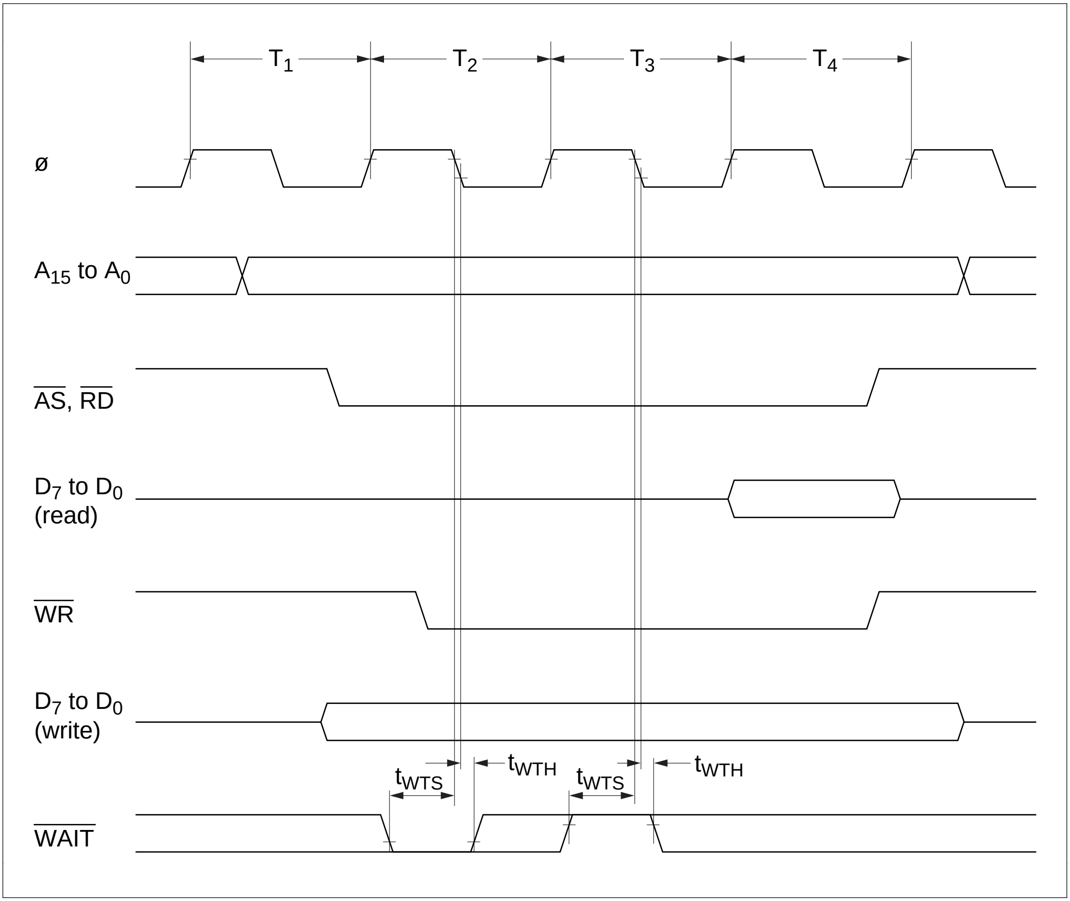


Figure 16-5 Basic Bus Cycle (with 1 Wait State) in Expanded Modes

16.3.2 Control Signal Timing

(1) Reset Input Timing

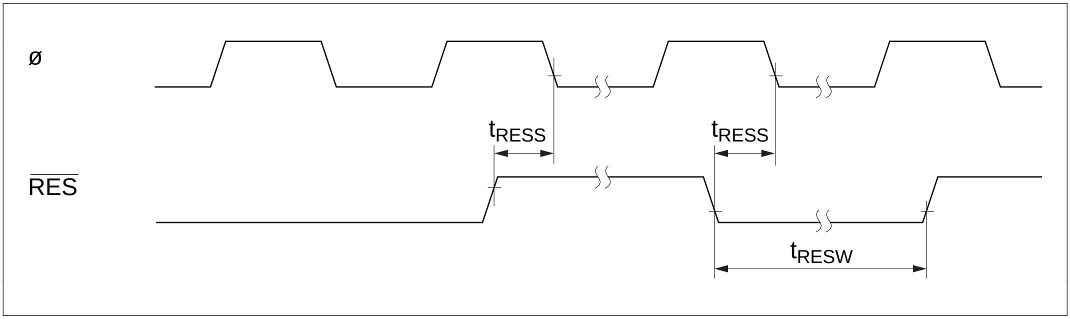
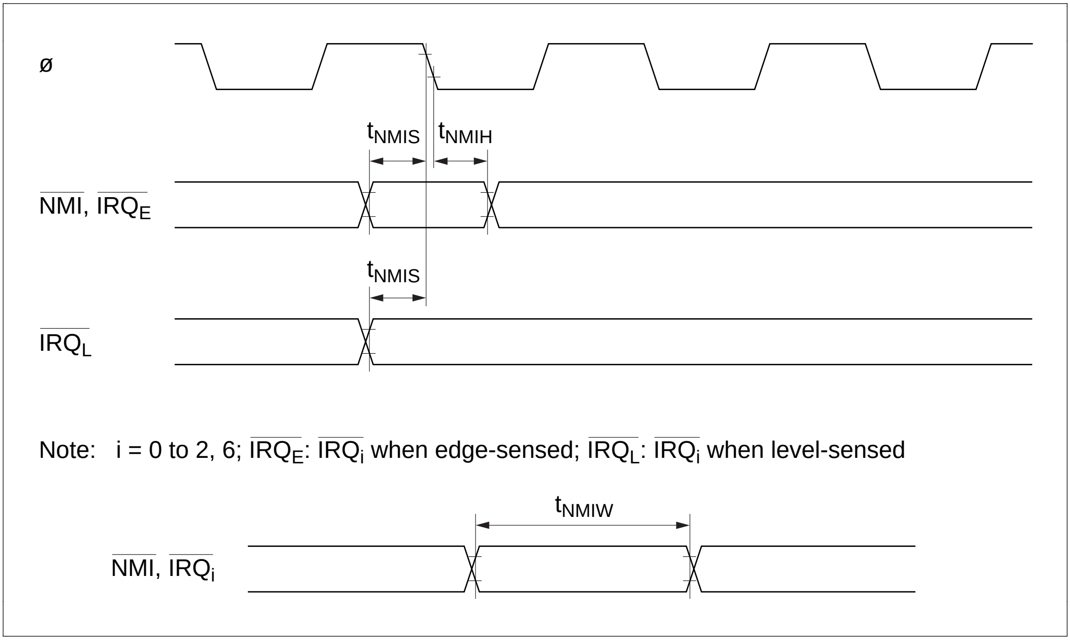


Figure 16-6 Reset Input Timing

(2) Interrupt Input Timing



ø
t_{NMIS}
t_{NMIH}
$\overline{NMI}$, $\overline{IRQ_E}$
t_{NMIS}
$\overline{IRQ_L}$
t_{NMIW}
$\overline{NMI}$, $\overline{IRQ_i}$

Note: i = 0 to 2, 6; $\overline{IRQ_E}$: $\overline{IRQ_i}$ when edge-sensed; $\overline{IRQ_L}$: $\overline{IRQ_i}$ when level-sensed

Figure 16-7 Interrupt Input Timing

(3) Clock Settling Timing

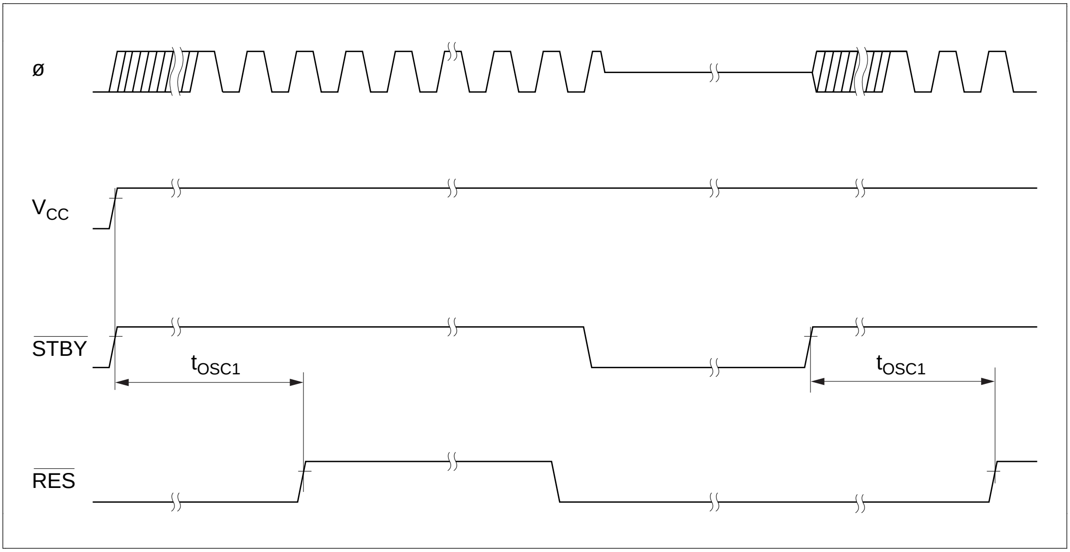


Figure 16-8 Clock Settling Timing

(4) Clock Settling Timing for Recovery from Software Standby Mode

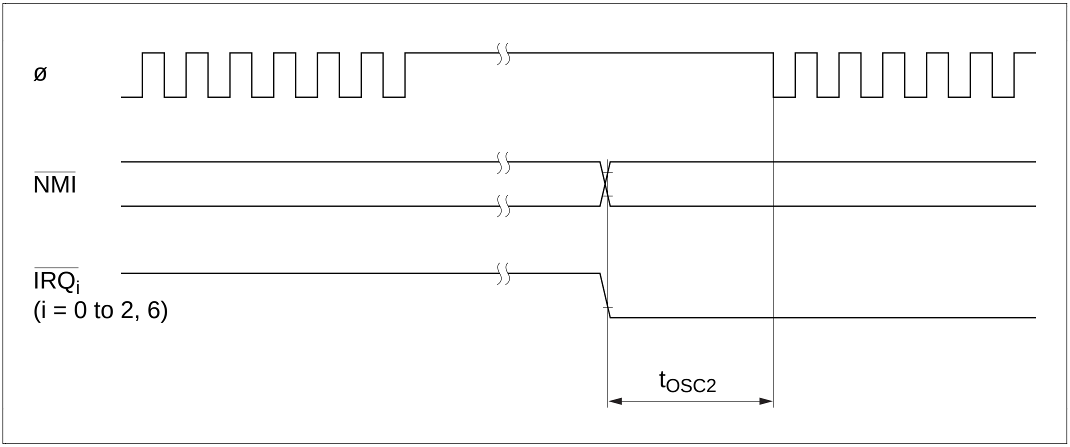


Figure 16-9 Clock Settling Timing for Recovery from Software Standby Mode

16.3.3 16-Bit Free-Running Timer Timing

(1) Free-Running Timer Input/Output Timing

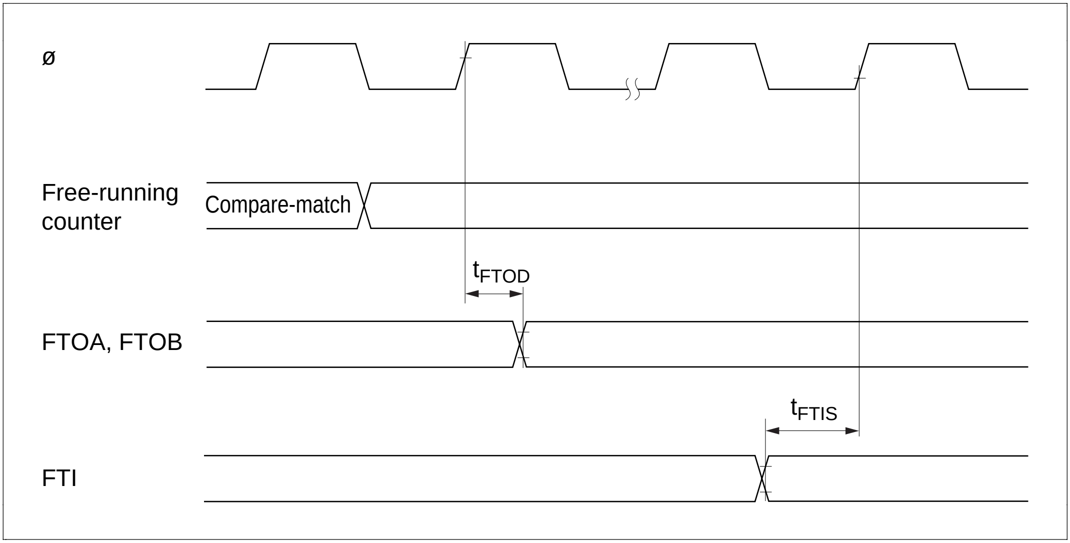


Figure 16-10 Free-Running Timer Input/Output Timing

(2) External Clock Input Timing for Free-Running Timer

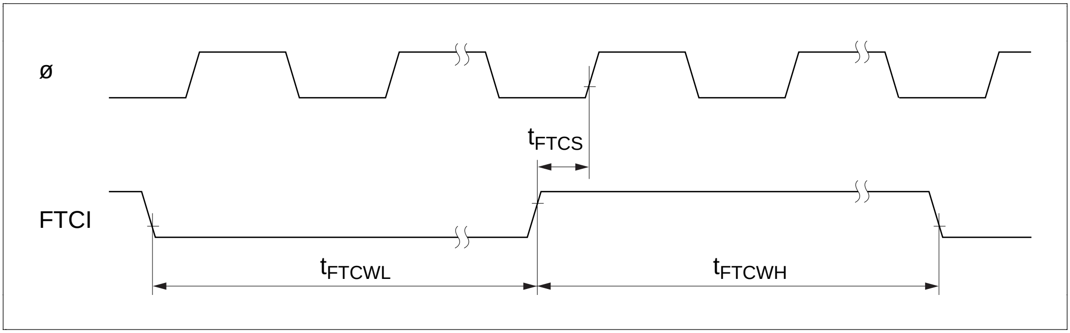


Figure 16-11 External Clock Input Timing for Free-Running Timer

16.3.4 8-Bit Timer Timing

(1) 8-Bit Timer Output Timing

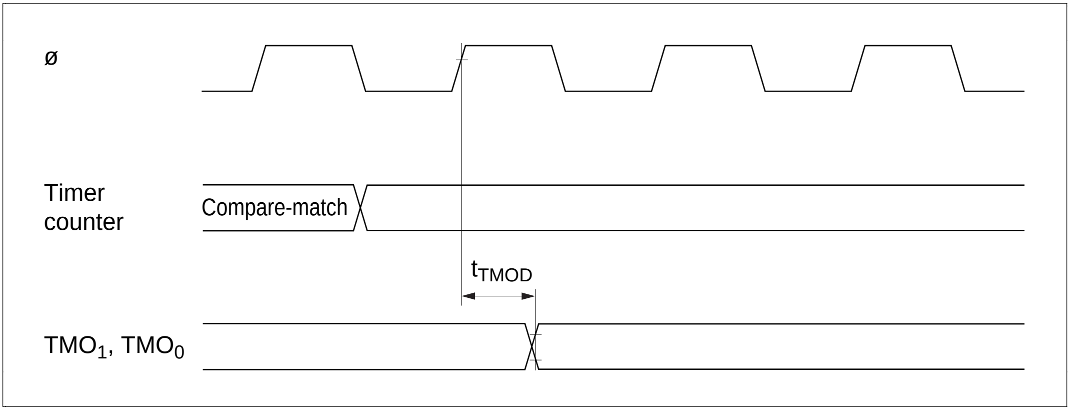


Figure 16-12 8-Bit Timer Output Timing

(2) 8-Bit Timer Clock Input Timing

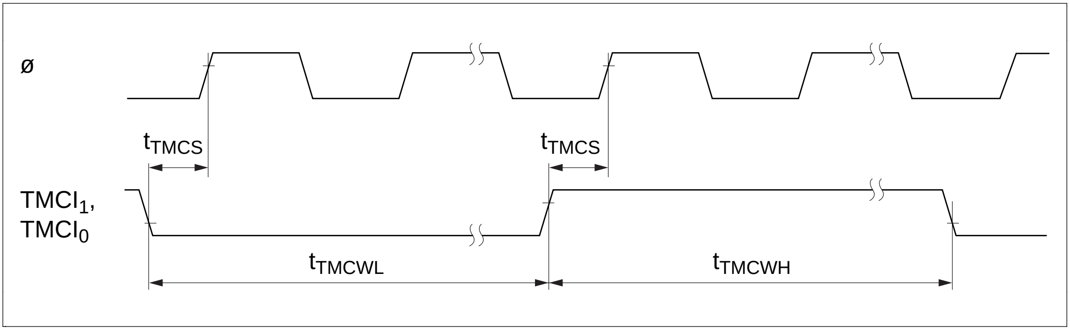


Figure 16-13 8-Bit Timer Clock Input Timing

(3) 8-Bit Timer Reset Input Timing

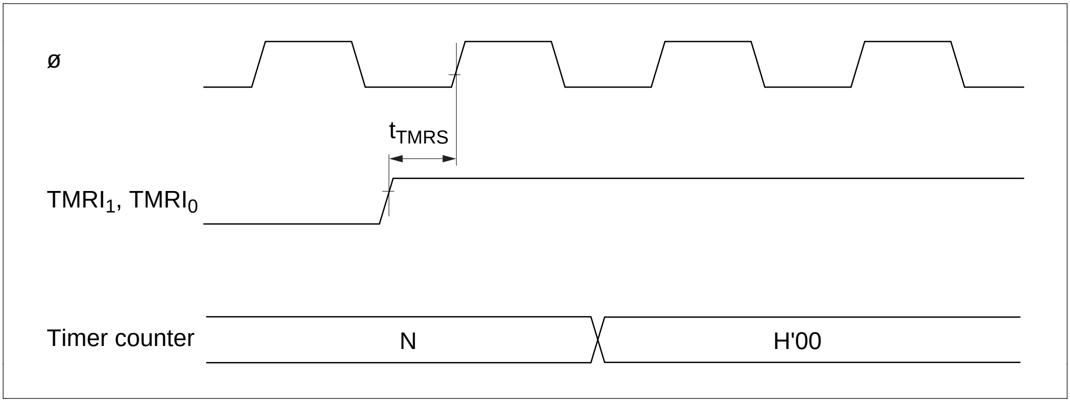


Figure 16-14 8-Bit Timer Reset Input Timing

16.3.5 Serial Communication Interface Timing

(1) SCI Input/Output Timing

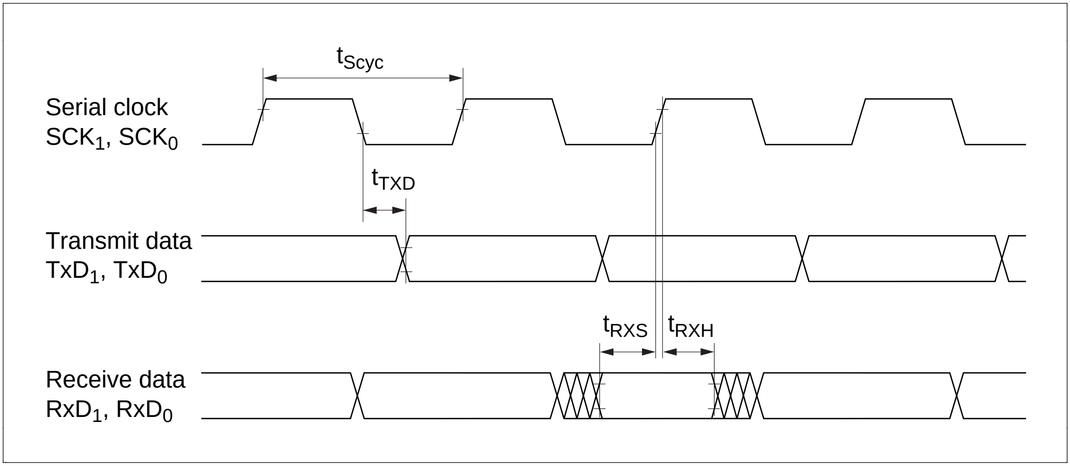


Figure 16-15 SCI Input/Output Timing (Synchronous Mode)

(2) SCI Input Clock Timing

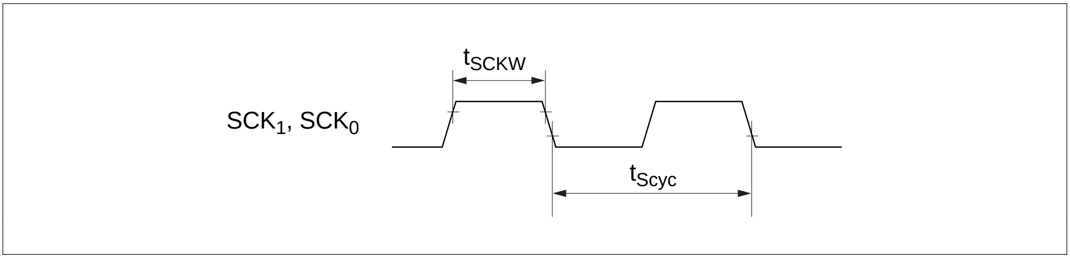
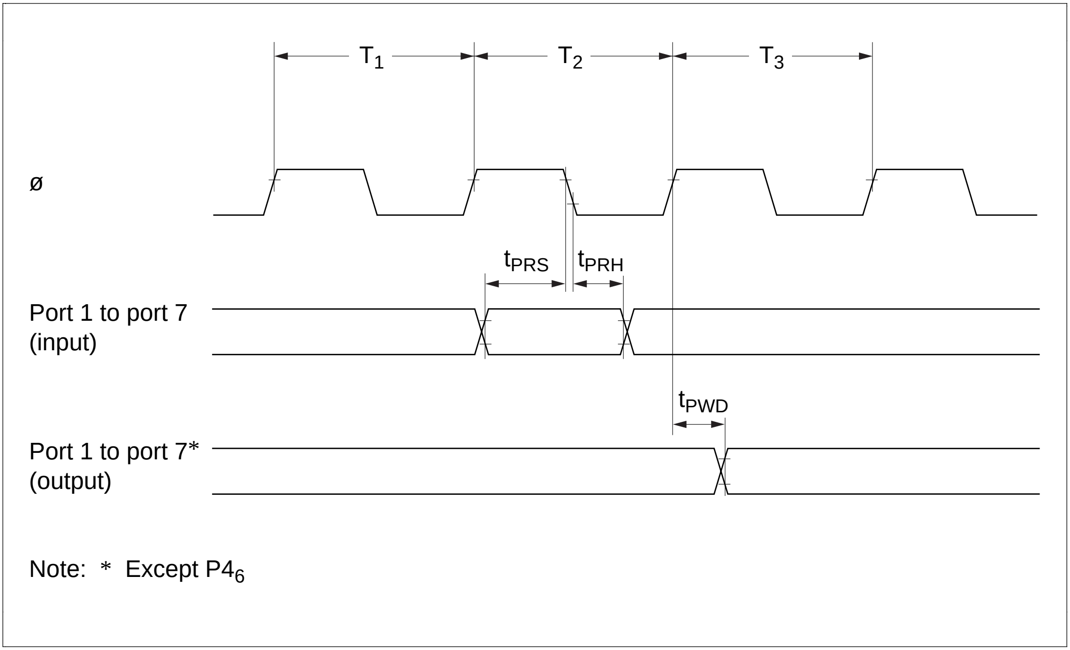


Figure 16-16 SCI Input Clock Timing

16.3.6 I/O Port Timing

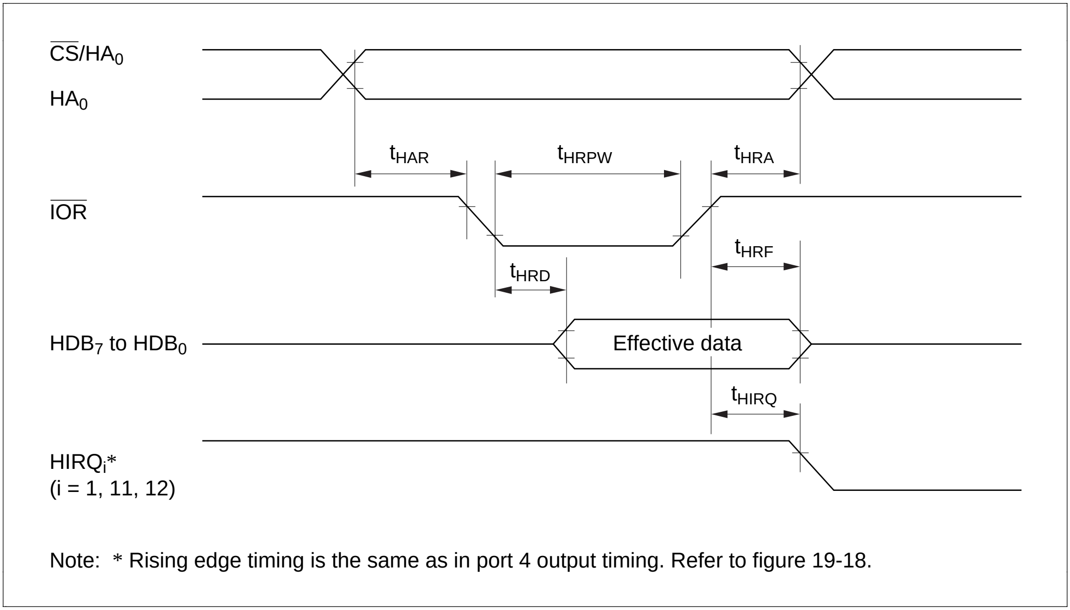


Note: * Except $P4_6$

Figure 16-17 I/O Port Input/Output Timing

16.3.7 Host Interface Timing

(1) Host Interface Read Timing



Note: * Rising edge timing is the same as in port 4 output timing. Refer to figure 19-18.

Figure 16-18 Host Interface Read Timing

(2) Host Interface Write Timing

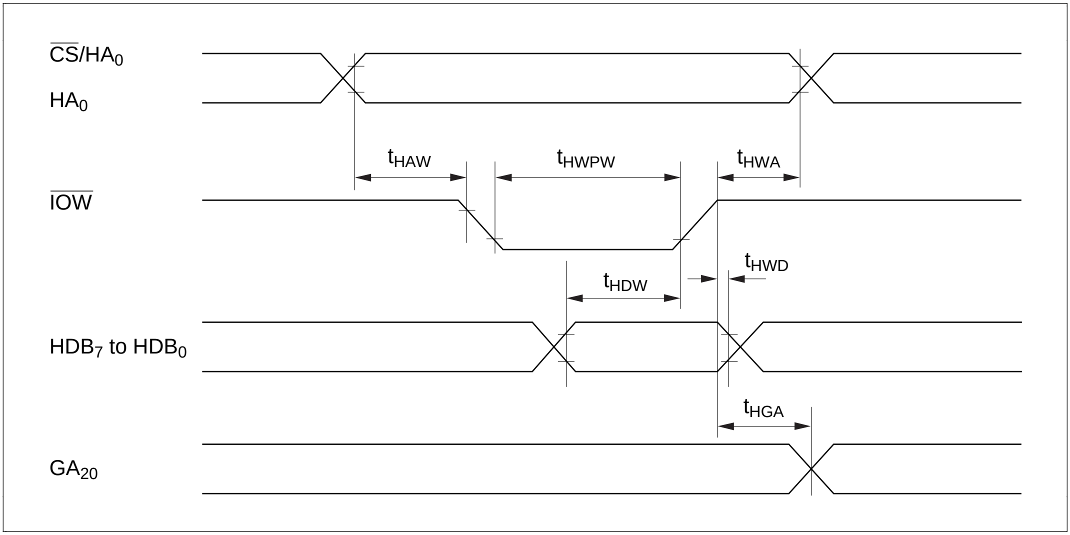
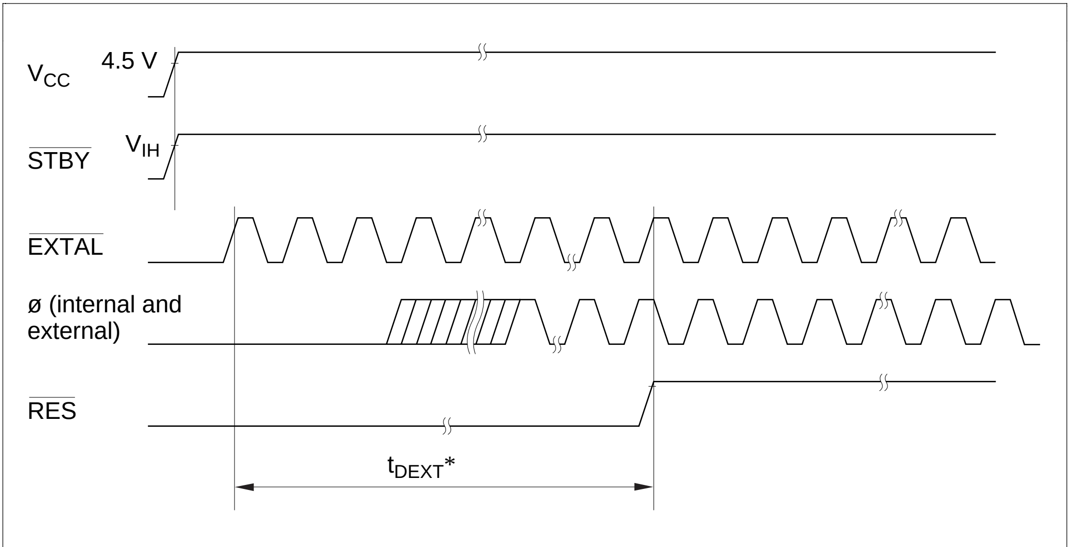


Figure 16-19 Host Interface Write Timing

16.3.8 External Clock Output Timing



Note: * t_{DEXT} includes a 10 t_{cyc} $\overline{RES}$ pulse width (t_{RESW}).

Figure 16-20 External Clock Output Settling Delay Timing

Appendix A Instruction Set

A.1 Instruction List

Operation Notation

Rd8/16	General register (destination) (8 or 16 bits)
Rs8/16	General register (source) (8 or 16 bits)
Rn8/16	General register (8 or 16 bits)
CCR	Condition code register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#xx:3/8/16	Immediate data (3, 8, or 16 bits)
d:8/16	Displacement (8 or 16 bits)
@aa:8/16	Absolute address (8 or 16 bits)
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
→	Move
—	Not

Condition Code Notation

↕	Modified according to the instruction result
*	Undetermined (unpredictable)
0	Always cleared to 0
—	Not affected by the instruction result

Table A-1 Instruction Set

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
MOV.B #xx:8, Rd	B	#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
MOV.B Rs, Rd	B	Rs8 → Rd8		2								—	—	↕	↕	0	—	2
MOV.B @Rs, Rd	B	@Rs16 → Rd8			2							—	—	↕	↕	0	—	4
MOV.B @(d:16, Rs), Rd	B	@(d:16, Rs16)→ Rd8				4						—	—	↕	↕	0	—	6
MOV.B @Rs+, Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					—	—	↕	↕	0	—	6
MOV.B @aa:8, Rd	B	@aa:8 → Rd8						2				—	—	↕	↕	0	—	4
MOV.B @aa:16, Rd	B	@aa:16 → Rd8						4				—	—	↕	↕	0	—	6
MOV.B Rs, @Rd	B	Rs8 → @Rd16			2							—	—	↕	↕	0	—	4
MOV.B Rs, @(d:16, Rd)	B	Rs8 → @(d:16, Rd16)				4						—	—	↕	↕	0	—	6
MOV.B Rs, @–Rd	B	Rd16–1 → Rd16 Rs8 → @Rd16					2					—	—	↕	↕	0	—	6
MOV.B Rs, @aa:8	B	Rs8 → @aa:8						2				—	—	↕	↕	0	—	4
MOV.B Rs, @aa:16	B	Rs8 → @aa:16						4				—	—	↕	↕	0	—	6
MOV.W #xx:16, Rd	W	#xx:16 → Rd16	4									—	—	↕	↕	0	—	4
MOV.W Rs, Rd	W	Rs16 → Rd16		2								—	—	↕	↕	0	—	2
MOV.W @Rs, Rd	W	@Rs16 → Rd16			2							—	—	↕	↕	0	—	4
MOV.W @(d:16, Rs), Rd	W	@(d:16, Rs16) → Rd16				4						—	—	↕	↕	0	—	6
MOV.W @Rs+, Rd	W	@Rs16 → Rd16 Rs16+2 → Rs16					2					—	—	↕	↕	0	—	6
MOV.W @aa:16, Rd	W	@aa:16 → Rd16						4				—	—	↕	↕	0	—	6
MOV.W Rs, @Rd	W	Rs16 → @Rd16			2							—	—	↕	↕	0	—	4
MOV.W Rs, @(d:16, Rd)	W	Rs16 → @(d:16, Rd16)				4						—	—	↕	↕	0	—	6
MOV.W Rs, @–Rd	W	Rd16–2 → Rd16 Rs16 → @Rd16					2					—	—	↕	↕	0	—	6
MOV.W Rs, @aa:16	W	Rs16 → @aa:16						4				—	—	↕	↕	0	—	6
POP Rd	W	@SP → Rd16 SP+2 → SP					2					—	—	↕	↕	0	—	6

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					—	—	↕	↕	0	—	6
MOVFPE @aa:16, Rd	B	Not supported																
MOVTPE Rs, @aa:16	B	Not supported																
ADD.B #xx:8, Rd	B	Rd8+#xx:8 → Rd8	2									—	↕	↕	↕	↕	↕	2
ADD.B Rs, Rd	B	Rd8+Rs8 → Rd8		2								—	↕	↕	↕	↕	↕	2
ADD.W Rs, Rd	W	Rd16+Rs16 → Rd16		2								—	(1)	↕	↕	↕	↕	2
ADDX.B #xx:8, Rd	B	Rd8+#xx:8+C → Rd8	2									—	↕	↕	(2)	↕	↕	2
ADDX.B Rs, Rd	B	Rd8+Rs8+C → Rd8		2								—	↕	↕	(2)	↕	↕	2
ADDS.W #1, Rd	W	Rd16+1 → Rd16		2								—	—	—	—	—	—	2
ADDS.W #2, Rd	W	Rd16+2 → Rd16		2								—	—	—	—	—	—	2
INC.B Rd	B	Rd8+1 → Rd8		2								—	—	↕	↕	↕	—	2
DAA.B Rd	B	Rd8 decimal adjust →Rd8		2								—	*	↕	↕	*	(3)	2
SUB.B Rs, Rd	B	Rd8–Rs8 → Rd8		2								—	↕	↕	↕	↕	↕	2
SUB.W Rs, Rd	W	Rd16–Rs16 → Rd16		2								—	(1)	↕	↕	↕	↕	2
SUBX.B #xx:8, Rd	B	Rd8–#xx:8–C → Rd8	2									—	↕	↕	(2)	↕	↕	2
SUBX.B Rs, Rd	B	Rd8–Rs8–C → Rd8		2								—	↕	↕	(2)	↕	↕	2
SUBS.W #1, Rd	W	Rd16–1 → Rd16		2								—	—	—	—	—	—	2
SUBS.W #2, Rd	W	Rd16–2 → Rd16		2								—	—	—	—	—	—	2
DEC.B Rd	B	Rd8–1 → Rd8		2								—	—	↕	↕	↕	—	2
DAS.B Rd	B	Rd8 decimal adjust →Rd8		2								—	*	↕	↕	*	—	2
NEG.B Rd	B	0–Rd8→ Rd8		2								—	↕	↕	↕	↕	↕	2
CMP.B #xx:8, Rd	B	Rd8–#xx:8	2									—	↕	↕	↕	↕	↕	2
CMP.B Rs, Rd	B	Rd8–Rs8		2								—	↕	↕	↕	↕	↕	2
CMP.W Rs, Rd	W	Rd16–Rs16		2								—	(1)	↕	↕	↕	↕	2
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16		2								—	—	—	—	—	—	14

Table A-1 Instruction Set (cont)

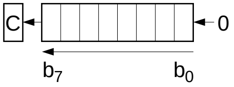
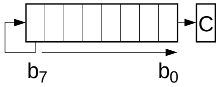
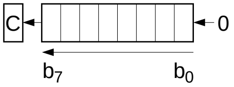
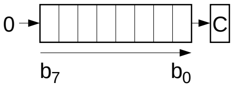
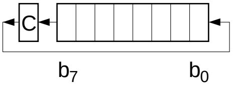
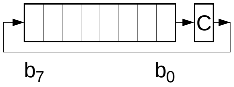
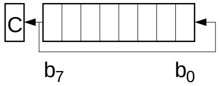
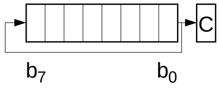
Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
DIVXU.B Rs, Rd	B	Rd16÷Rs8 → Rd16 (RdH: remainder, RdL: quotient)		2								—	—	(6)	(7)	—	—	14
AND.B #xx:8, Rd	B	Rd8∧#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
AND.B Rs, Rd	B	Rd8∧Rs8 → Rd8		2								—	—	↕	↕	0	—	2
OR.B #xx:8, Rd	B	Rd8∨#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
OR.B Rs, Rd	B	Rd8∨Rs8 → Rd8		2								—	—	↕	↕	0	—	2
XOR.B #xx:8, Rd	B	Rd8⊕#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
XOR.B Rs, Rd	B	Rd8⊕Rs8 → Rd8		2								—	—	↕	↕	0	—	2
NOT.B Rd	B	$\overline{\text{Rd8}}$ → Rd8		2								—	—	↕	↕	0	—	2
SHAL.B Rd	B			2								—	—	↕	↕	↕	↕	2
SHAR.B Rd	B			2								—	—	↕	↕	0	↕	2
SHLL.B Rd	B			2								—	—	↕	↕	0	↕	2
SHLR.B Rd	B			2								—	—	0	↕	0	↕	2
ROTXL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTXR.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTR.B Rd	B			2								—	—	↕	↕	0	↕	2

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← $(\overline{\text{\#xx:3 of Rd8}})$		2								—	—	—	—	—	—	2
BNOT #xx:3, @Rd	B	(#xx:3 of @Rd16) ← $(\overline{\text{\#xx:3 of @Rd16}})$			4							—	—	—	—	—	—	8
BNOT #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← $(\overline{\text{\#xx:3 of @aa:8}})$						4				—	—	—	—	—	—	8
BNOT Rn, Rd	B	(Rn8 of Rd8) ← $(\overline{\text{Rn8 of Rd8}})$		2								—	—	—	—	—	—	2
BNOT Rn, @Rd	B	(Rn8 of @Rd16) ← $(\overline{\text{Rn8 of @Rd16}})$			4							—	—	—	—	—	—	8
BNOT Rn, @aa:8	B	(Rn8 of @aa:8) ← $(\overline{\text{Rn8 of @aa:8}})$						4				—	—	—	—	—	—	8
BTST #xx:3, Rd	B	$(\overline{\text{\#xx:3 of Rd8}})$ → Z		2								—	—	—	↕	—	—	2
BTST #xx:3, @Rd	B	$(\overline{\text{\#xx:3 of @Rd16}})$ → Z			4							—	—	—	↕	—	—	6
BTST #xx:3, @aa:8	B	$(\overline{\text{\#xx:3 of @aa:8}})$ → Z						4				—	—	—	↕	—	—	6
BTST Rn, Rd	B	$(\overline{\text{Rn8 of Rd8}})$ → Z		2								—	—	—	↕	—	—	2

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
BTST Rn, @Rd	B	($\overline{\text{Rn8 of @Rd16}}$) → Z			4							—	—	—	↕	—	—	6
BTST Rn, @aa:8	B	($\overline{\text{Rn8 of @aa:8}}$) → Z						4				—	—	—	↕	—	—	6
BLD #xx:3, Rd	B	(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BLD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BLD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BILD #xx:3, Rd	B	($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BILD #xx:3, @Rd	B	($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6
BILD #xx:3, @aa:8	B	($\overline{\text{\#xx:3 of @aa:8}}$) → C						4				—	—	—	—	—	↕	6
BST #xx:3, Rd	B	C → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BST #xx:3, @Rd	B	C → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BST #xx:3, @aa:8	B	C → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BIST #xx:3, Rd	B	$\overline{\text{C}}$ → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BIST #xx:3, @Rd	B	$\overline{\text{C}}$ → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BIST #xx:3, @aa:8	B	$\overline{\text{C}}$ → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BAND #xx:3, Rd	B	C∧(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BAND #xx:3, @Rd	B	C∧(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BAND #xx:3, @aa:8	B	C∧(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIAND #xx:3, Rd	B	C∧($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BIAND #xx:3, @Rd	B	C∧($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6
BIAND #xx:3, @aa:8	B	C∧($\overline{\text{\#xx:3 of @aa:8}}$) → C						4				—	—	—	—	—	↕	6
BOR #xx:3, Rd	B	C∨(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BOR #xx:3, @Rd	B	C∨(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIOR #xx:3, Rd	B	C∨($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BIOR #xx:3, @Rd	B	C∨($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6
BIOR #xx:3, @aa:8	B	C∨($\overline{\text{\#xx:3 of @aa:8}}$) → C						4				—	—	—	—	—	↕	6

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation		Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			Branching Condition	#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
BXOR #xx:3, Rd	B	C⊕(#xx:3 of Rd8) → C			2								—	—	—	—	—	↕	2
BXOR #xx:3, @Rd	B	C⊕(#xx:3 of @Rd16) → C				4							—	—	—	—	—	↕	6
BXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C							4				—	—	—	—	—	↕	6
BIXOR #xx:3, Rd	B	C⊕($\overline{\text{\#xx:3 of Rd8}}$) → C			2								—	—	—	—	—	↕	2
BIXOR #xx:3, @Rd	B	C⊕($\overline{\text{\#xx:3 of @Rd16}}$) → C				4							—	—	—	—	—	↕	6
BIXOR #xx:3, @aa:8	B	C⊕($\overline{\text{\#xx:3 of @aa:8}}$) → C							4				—	—	—	—	—	↕	6
BRA d:8 (BT d:8)	—	PC ← PC+d:8								2			—	—	—	—	—	—	4
BRN d:8 (BF d:8)	—	PC ← PC+2								2			—	—	—	—	—	—	4
BHI d:8	—	If condition is true then PC ← PC+d:8 else next;	C ∨ Z = 0							2			—	—	—	—	—	—	4
BLS d:8	—		C ∨ Z = 1							2			—	—	—	—	—	—	4
BCC d:8 (BHS d:8)	—		C = 0							2			—	—	—	—	—	—	4
BCS d:8 (BLO d:8)	—		C = 1							2			—	—	—	—	—	—	4
BNE d:8	—		Z = 0							2			—	—	—	—	—	—	4
BEQ d:8	—		Z = 1							2			—	—	—	—	—	—	4
BVC d:8	—		V = 0							2			—	—	—	—	—	—	4
BVS d:8	—		V = 1							2			—	—	—	—	—	—	4
BPL d:8	—		N = 0							2			—	—	—	—	—	—	4
BMI d:8	—		N = 1							2			—	—	—	—	—	—	4
BGE d:8	—		N⊕V = 0							2			—	—	—	—	—	—	4
BLT d:8	—		N⊕V = 1							2			—	—	—	—	—	—	4
BGT d:8	—		Z ∨ (N⊕V) = 0							2			—	—	—	—	—	—	4
BLE d:8	—		Z ∨ (N⊕V) = 1							2			—	—	—	—	—	—	4
JMP @Rn	—	PC ← Rn16				2							—	—	—	—	—	—	4
JMP @aa:16	—	PC ← aa:16							4				—	—	—	—	—	—	6
JMP @@aa:8	—	PC ← @aa:8									2		—	—	—	—	—	—	8

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
BSR d:8	—	SP–2 → SP PC → @SP PC ← PC+d:8							2			—	—	—	—	—	—	6
JSR @Rn	—	SP–2 → SP PC → @SP PC ← Rn16			2							—	—	—	—	—	—	6
JSR @aa:16	—	SP–2 → SP PC → @SP PC ← aa:16						4				—	—	—	—	—	—	8
JSR @@aa:8	—	SP–2 → SP PC → @SP PC ← @aa:8								2		—	—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP									2	—	—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	—	Transition to power-down state									2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs, CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR, Rd	B	CCR → Rd8		2								—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8, CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8, CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	—	PC ← PC+2									2	—	—	—	—	—	—	2

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States*
			#xx:8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa:8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
EEPMOV	—	if R4L≠0 Repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L Until R4L=0 else next;									4	—	—	—	—	—	—	(4)

Notes: The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.

(1) Set to 1 when there is a carry or borrow from bit 11; otherwise cleared to 0.

(2) If the result is zero, the previous value of the flag is retained: otherwise the flag is cleared to 0.

(3) Set to 1 if decimal adjustment produces a carry; otherwise cleared to 0.

(4) The number of states required for execution is 4n + 8 (n = value of R4L).

(5) These instructions are not supported by the H8/3502.

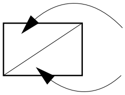
(6) Set to 1 if the divisor is negative: otherwise cleared to 0.

(7) Cleared to 0 if the divisor is not zero; set to 1 if the divisor is zero.

A.2 Operation Code Map

Table A-2 is a map of the operation codes contained in the first byte of the instruction code (bits 15 to 8 of the first instruction word).

Some pairs of instructions have identical first bytes. These instructions are differentiated by the first bit of the second byte (bit 7 of the first instruction word).



Instruction when first bit of byte 2 (bit 7 of first instruction word) is 0.

Instruction when first bit of byte 2 (bit 7 of first instruction word) is 1.

Table A-2 Operation Code Map

High \ Low	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SLEEP	STC	LDC	ORC	XORC	ANDC	LDC	ADD		INC	ADDS	MOV		ADDX	DAA
1	SHLL SHAL	SHLR SHAR	ROTXL ROTL	ROTXR ROTR	OR	XOR	AND	NOT NEG	SUB		DEC	SUBS	CMP		SUBX	DAS
2	MOV															
3																
4	BRA*2	BRN*2	BHI	BLS	BCC*2	BCS*2	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU	▨	▨	RTS	BSR	RTE	▨	▨	JMP			▨	JSR		
6	BSET	BNOT	BCLR	BTS	▨	▨	▨	BST BIST	MOV*1							
7					BOR BIOR	BXOR BIXOR	BAND BIAND	BLD BILD	▨	MOV	▨	EEPMOV	Bit manipulation instructions			
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

Notes: 1. The PUSH and POP instructions are identical in machine language to MOV instructions.
2. The BT, BF, BHS, and BLO instructions are identical in machine language to BRA, BRN, BCC, and BCS, respectively.

A.3 Number of States Required for Execution

The tables below can be used to calculate the number of states required for instruction execution. Table A-3 indicates the number of states required for each cycle (instruction fetch, branch address read, stack operation, byte data access, word data access, internal operation). Table A-4 indicates the number of cycles of each type occurring in each instruction. The total number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: Mode 1 (on-chip ROM disabled), stack located in external memory, 1 wait state inserted in external memory access.

1. BSET #0, @FFC7
   From table A-4: $I = L = 2$, $J = K = M = N= 0$
   From table A-3: $S_I = 8$, $S_L = 3$
   Number of states required for execution: $2 \times 8 + 2 \times 3 =22$

2. JSR @@30
   From table A-4: $I = 2$, $J = K = 1$, $L = M = N = 0$
   From table A-3: $S_I = S_J = S_K = 8$
   Number of states required for execution: $2 \times 8 + 1 \times 8 + 1 \times 8 = 32$

Table A-3 Number of States Taken by Each Cycle in Instruction Execution

Execution Status (Instruction Cycle)		Access location		
		On-Chip Memory	On-Chip Reg. Field	External Memory
Instruction fetch	S_I	2	—	6 + 2m
Branch address read	S_J			
Stack operation	S_K			
Byte data access	S_L		3	3 + m
Word data access	S_M		6	6 + 2m
Internal operation	S_N	1	1	1

Note: m: Number of wait states inserted in access to external device.

Table A-4 Number of Cycles in Each Instruction

Instruction	Mnemonic	Instruction Fetch I	Branch Address Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ADD	ADD.B #xx:8, Rd	1					
	ADD.B Rs, Rd	1					
	ADD.W Rs, Rd	1					
ADDS	ADDS.W #1/2, Rd	1					
ADDX	ADDX.B #xx:8, Rd	1					
	ADDX.B Rs, Rd	1					
AND	AND.B #xx:8, Rd	1					
	AND.B Rs, Rd	1					
ANDC	ANDC #xx:8, CCR	1					
BAND	BAND #xx:3, Rd	1					
	BAND #xx:3, @Rd	2			1		
	BAND #xx:3, @aa:8	2			1		
Bcc	BRA d:8 (BT d:8)	2					
	BRN d:8 (BF d:8)	2					
	BHI d:8	2					
	BLS d:8	2					
	BCC d:8 (BHS d:8)	2					
	BCS d:8 (BLO d:8)	2					
	BNE d:8	2					
	BEQ d:8	2					
	BVC d:8	2					
	BVS d:8	2					
	BPL d:8	2					
	BMI d:8	2					
	BGE d:8	2					
	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					

Table A-4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Address Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BCLR	BCLR #xx:3, Rd	1					
	BCLR #xx:3, @Rd	2			2		
	BCLR #xx:3, @aa:8	2			2		
	BCLR Rn, Rd	1					
	BCLR Rn, @Rd	2			2		
	BCLR Rn, @aa:8	2			2		
BIAND	BIAND #xx:3, Rd	1					
	BIAND #xx:3, @Rd	2			1		
	BIAND #xx:3, @aa:8	2			1		
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @Rd	2			1		
	BILD #xx:3, @aa:8	2			1		
BIOR	BIOR #xx:3, Rd	1					
	BIOR #xx:3, @Rd	2			1		
	BIOR #xx:3, @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @Rd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @Rd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @Rd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @Rd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @Rd	2			2		
	BNOT Rn, @aa:8	2			2		

Table A-4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Address Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @Rd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @Rd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @Rd	2			2		
	BSET Rn, @aa:8	2			2		
BSR	BSR d:8	2		1			
BST	BST #xx:3, Rd	1					
	BST #xx:3, @Rd	2			2		
	BST #xx:3, @aa:8	2			2		
BTST	BTST #xx:3, Rd	1					
	BTST #xx:3, @Rd	2			1		
	BTST #xx:3, @aa:8	2			1		
	BTST Rn, Rd	1					
	BTST Rn, @Rd	2			1		
	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @Rd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP.B #xx:8, Rd	1					
	CMP.B Rs, Rd	1					
	CMP.W Rs, Rd	1					
DAA	DAA.B Rd	1					
DAS	DAS.B Rd	1					
DEC	DEC.B Rd	1					
DIVXU	DIVXU.B Rs, Rd	1					12
EEPMOV	EEPMOV	2			2n + 2*		1

Table A-4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Address Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
INC	INC.B Rd	1					
JMP	JMP @Rn	2					
	JMP @aa:16	2					2
	JMP @@aa:8	2	1				2
JSR	JSR @Rn	2		1			
	JSR @aa:16	2		1			2
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @Rs, Rd	1			1		
	MOV.B @(d:16,Rs), Rd	2			1		
	MOV.B @Rs+, Rd	1			1		2
	MOV.B @aa:8, Rd	1			1		
	MOV.B @aa:16, Rd	2			1		
	MOV.B Rs, @Rd	1			1		
	MOV.B Rs, @(d:16, Rd)	2			1		
	MOV.B Rs, @–Rd	1			1		2
	MOV.B Rs, @aa:8	1			1		
	MOV.B Rs, @aa:16	2			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @Rs, Rd	1				1	
	MOV.W @(d:16, Rs), Rd	2				1	
	MOV.W @Rs+, Rd	1				1	2
	MOV.W @aa:16, Rd	2				1	
	MOV.W Rs, @Rd	1				1	
	MOV.W Rs, @(d:16, Rd)	2				1	
	MOV.W Rs, @–Rd	1				1	2
	MOV.W Rs, @aa:16	2				1	

Table A-4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Address Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MOVFPE	MOVFPE @aa:16, Rd	Not supported					
MOVTPE	MOVTPE Rs, @aa:16	Not supported					
MULXU	MULXU.B Rs, Rd	1					12
NEG	NEG.B Rd	1					
NOP	NOP	1					
NOT	NOT.B Rd	1					
OR	OR.B #xx:8, Rd	1					
	OR.B Rs, Rd	1					
ORC	ORC #xx:8, CCR	1					
POP	POP Rd	1			1		2
PUSH	PUSH Rd	1			1		2
ROTL	ROTL.B Rd	1					
ROTR	ROTR.B Rd	1					
ROTXL	ROTXL.B Rd	1					
ROTXR	ROTXR.B Rd	1					
RTE	RTE	2		2			2
RTS	RTS	2		1			2
SHAL	SHAL.B Rd	1					
SHAR	SHAR.B Rd	1					
SHLL	SHLL.B Rd	1					
SHLR	SHLR.B Rd	1					
SLEEP	SLEEP	1					
STC	STC CCR, Rd	1					
SUB	SUB.B Rs, Rd	1					
	SUB.W Rs, Rd	1					
SUBS	SUBS.W #1/2, Rd	1					
SUBX	SUBX.B #xx:8, Rd	1					
	SUBX.B Rs, Rd	1					

Table A-4 Number of Cycles in Each Instruction (cont)

			Branch		Byte	Word	
		Instruction	Address	Stack	Data	Data	Internal
		Fetch	Read	Operation	Access	Access	Operation
Instruction	Mnemonic	I	J	K	L	M	N
XOR	XOR.B #xx:8, Rd	1					
	XOR.B Rs, Rd	1					
XORC	XORC #xx:8, CCR	1					

Note: All values left blank are zero.

* n: Initial value in R4L. Source and destination are accessed n + 1 times each.

Appendix B Internal I/O Register

B.1 Addresses

B.1.1 I/O Registers

Address (Last Byte)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'80										External memory (in expanded modes)
H'81										
H'82										
H'83										
H'84										
H'85										
H'86										
H'87										
H'88										
H'89										
H'8A										
H'8B										
H'8C										
H'8D										
H'8E										
H'8F										
H'90	TCR	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0	FRT
H'91	TCSR	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA	
H'92	FRCH									
H'93	FRCL									
H'94	OCRAH									
H'95	OCRAL									
H'96	OCRBH									
H'97	OCRBL									
H'98	ICRH									
H'99	ICRL									

Address (Last Byte)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'9A										
H'9B										
H'9C										
H'9D										
H'9E										
H'9F										
H'A0										
H'A1										
H'A2										
H'A3										
H'A4										
H'A5										
H'A6										
H'A7										
H'A8										
H'A9										
H'AA	TCSR/ TCNT	OVF	WT/$\overline{\text{IT}}$	TME	—	RST/ $\overline{\text{NMI}}$	CKS2	CKS1	CKS0	WDT
H'AB	TCNT									
H'AC	P1PCR	P1$_7$PCR	P1$_6$PCR	P1$_5$PCR	P1$_4$PCR	P1$_3$PCR	P1$_2$PCR	P1$_1$PCR	P1$_0$PCR	Port 1
H'AD	P2PCR	P2$_7$PCR	P2$_6$PCR	P2$_5$PCR	P2$_4$PCR	P2$_3$PCR	P2$_2$PCR	P2$_1$PCR	P2$_0$PCR	Port 2
H'AE	P3PCR	P3$_7$PCR	P3$_6$PCR	P3$_5$PCR	P3$_4$PCR	P3$_3$PCR	P3$_2$PCR	P3$_1$PCR	P3$_0$PCR	Port 3
H'AF										
H'B0	P1DDR	P1$_7$DDR	P1$_6$DDR	P1$_5$DDR	P1$_4$DDR	P1$_3$DDR	P1$_2$DDR	P1$_1$DDR	P1$_0$DDR	Port 1
H'B1	P2DDR	P2$_7$DDR	P2$_6$DDR	P2$_5$DDR	P2$_4$DDR	P2$_3$DDR	P2$_2$DDR	P2$_1$DDR	P2$_0$DDR	Port 2
HB2	P1DR	P1$_7$	P1$_6$	P1$_5$	P1$_4$	P1$_3$	P1$_2$	P1$_1$	P1$_0$	Port 1
H'B3	P2DR	P2$_7$	P2$_6$	P2$_5$	P2$_4$	P2$_3$	P2$_2$	P2$_1$	P2$_0$	Port 2
H'B4	P3DDR	P3$_7$DDR	P3$_6$DDR	P3$_5$DDR	P3$_4$DDR	P3$_3$DDR	P3$_2$DDR	P3$_1$DDR	P3$_0$DDR	Port 3
H'B5	P4DDR	P4$_7$DDR	P4$_6$DDR	P4$_5$DDR	P4$_4$DDR	P4$_3$DDR	P4$_2$DDR	P4$_1$DDR	P4$_0$DDR	Port 4
H'B6	P3DR	P3$_7$	P3$_6$	P3$_5$	P3$_4$	P3$_3$	P3$_2$	P3$_1$	P3$_0$	Port 3
H'B7	P4DR	P4$_7$	P4$_6$	P4$_5$	P4$_4$	P4$_3$	P4$_2$	P4$_1$	P4$_0$	Port 4

Address (Last Byte)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'B8	P5DDR	—	—	P5$_5$DDR	P5$_4$DDR	P5$_3$DDR	P5$_2$DDR	P5$_1$DDR	P5$_0$DDR	Port 5
H'B9	P6DDR	—	P6$_6$DDR	P6$_5$DDR	P6$_4$DDR	P6$_3$DDR	P6$_2$DDR	P6$_1$DDR	P6$_0$DDR	Port 6
H'BA	P5DR	—	—	P5$_5$	P5$_4$	P5$_3$	P5$_2$	P5$_1$	P5$_0$	Port 5
H'BB	P6DR	—	P6$_6$	P6$_5$	P6$_4$	P6$_3$	P6$_2$	P6$_1$	P6$_0$	Port 6
H'BC	P7DDR	P7$_7$DDR	P7$_6$DDR	P7$_5$DDR	P7$_4$DDR	P7$_3$DDR	P7$_2$DDR	P7$_1$DDR	P7$_0$DDR	Port 7
H'BD	—	—	—	—	—	—	—	—	—	—
H'BE	P7DR	P7$_7$	P7$_6$	P7$_5$	P7$_4$	P7$_3$	P7$_2$	P7$_1$	P7$_0$	Port 7
H'BF	—	—	—	—	—	—	—	—	—	—
H'C0										
H'C1										
H'C2	WSCR	—	—	CLKDBL	—	WMS1	WMS0	WC1	WC0	
H'C3	STCR	(IICS)	(IICX1)	(IICX0)	(SYNCE)	(PWCKE)	(PWCKS)	ICKS1	ICKS0	
H'C4	SYSCR	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME	
H'C5	MDCR	—	—	—	—	—	—	MDS1	MDS0	
H'C6	ISCR	—	IRQ6SC	—	—	—	IRQ2SC	IRQ1SC	IRQ0SC	
H'C7	IER	—	IRQ6E	—	—	—	IRQ2E	IRQ1E	IRQ0E	
H'C8	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR0
H'C9	TCSR	CMFB	CMFA	OVF	PWME	OS3	OS2	OS1	OS0	
H'CA	TCORA									
H'CB	TCORB									
H'CC	TCNT									
H'CD										
H'CE										
H'CF										
H'D0	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR1
H'D1	TCSR	CMFB	CMFA	OVF	PWME	OS3	OS2	OS1	OS0	
H'D2	TCORA									
H'D3	TCORB									
H'D4	TCNT									
H'D5										
H'D6										
H'D7										

Address (Last Byte)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'D8	SMR	C/$\overline{\text{A}}$	CHR	PE	O/$\overline{\text{E}}$	STOP	MP	CKS1	CKS0	SCI0
H'D9	BRR									
H'DA	SCR	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'DB	TDR									
H'DC	SSR	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT	
H'DD	RDR									
H'DE	SCMR	—	—	—	—	SDIR	SINV	—	SMIF	
H'DF	—	—	—	—	—	—	—	—	—	
H'E0	SMR	C/$\overline{\text{A}}$	CHR	PE	O/$\overline{\text{E}}$	STOP	MP	CKS1	CKS0	SCI1
H'E1	BRR									
H'E2	SCR	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'E3	TDR									
H'E4	SSR	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT	
H'E5	RDR									
H'E6	—	—	—	—	—	—	—	—	—	
H'E7	—	—	—	—	—	—	—	—	—	
H'E8										
H'E9										
H'EA										
H'EB										
H'EC										
H'ED										
H'EE										
H'EF										

Address (Last Byte)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'F0	HICR	—	—	—	—	—	IBFIE2	IBFIE1	FGA$_{20}$E	HIF
H'F1	KMIMR	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0	
H'F2	KMPCR	KM$_7$PCR	KM$_6$PCR	KM$_5$PCR	KM$_4$PCR	KM$_3$PCR	KM$_2$PCR	KM$_1$PCR	KM$_0$PCR	Port 6/ Port 7
H'F3	—	—	—	—	—	—	—	—	—	HIF
H'F4	IDR1	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0	
H'F5	ODR1	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0	
H'F6	STR1	DBU	DBU	DBU	DBU	$C/\overline{D}$	DBU	IBF	OBF	
H'F7	—	—	—	—	—	—	—	—	—	
H'F8	—	—	—	—	—	—	—	—	—	
H'F9	—	—	—	—	—	—	—	—	—	
H'FA	—	—	—	—	—	—	—	—	—	
H'FB	—	—	—	—	—	—	—	—	—	
H'FC	IDR2	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0	
H'FD	ODR2	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0	
H'FE	STR2	DBU	DBU	DBU	DBU	$C/\overline{D}$	DBU	IBF	OBF	
H'FF	—	—	—	—	—	—	—	—	—	

Notes: FRT: Free-running timer

TMR0: 8-bit timer channel 0

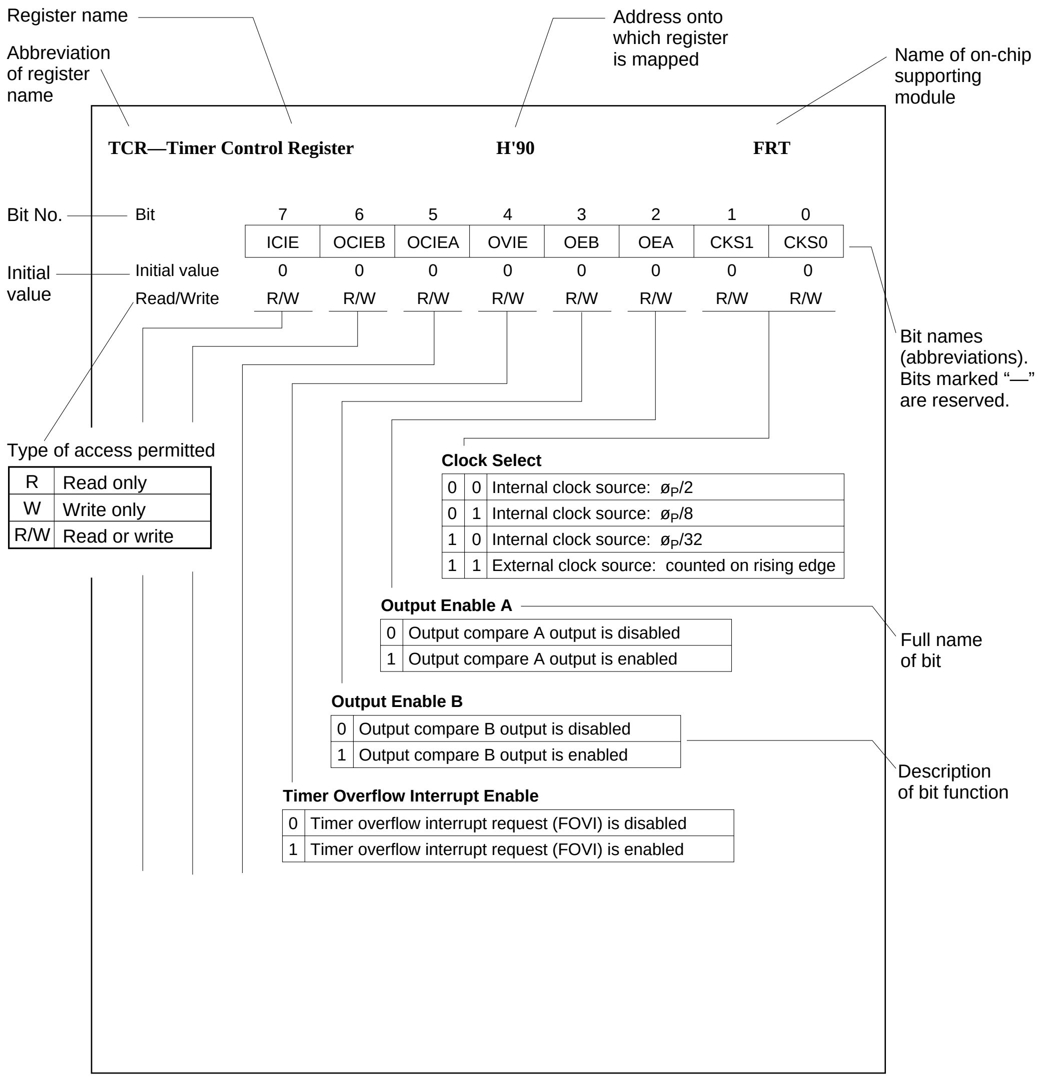
TMR1: 8-bit timer channel 1

SCI0: Serial communication interface 0

SCI1: Serial communication interface 1

HIF: Host interface

B.2 Function


Register name
Abbreviation of register name
Address onto which register is mapped
Name of on-chip supporting module
Bit No.
Initial value
Bit names (abbreviations). Bits marked "—" are reserved.
Type of access permitted
Full name of bit
Description of bit function

TCR—Timer Control Register **H'90** **FRT**

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Type of access permitted

R	Read only
W	Write only
R/W	Read or write

Clock Select

0	0	Internal clock source: $ø_P/2$
0	1	Internal clock source: $ø_P/8$
1	0	Internal clock source: $ø_P/32$
1	1	External clock source: counted on rising edge

Output Enable A

0	Output compare A output is disabled
1	Output compare A output is enabled

Output Enable B

0	Output compare B output is disabled
1	Output compare B output is enabled

Timer Overflow Interrupt Enable

0	Timer overflow interrupt request (FOVI) is disabled
1	Timer overflow interrupt request (FOVI) is enabled

TCR—Timer Control Register H'90 FRT

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Select

0	0	Internal clock source: ϕ_P/2
0	1	Internal clock source: ϕ_P/8
1	0	Internal clock source: ϕ_P/32
1	1	External clock source: counted on rising edge

Output Enable A

0	Output compare A output is disabled
1	Output compare A output is enabled

Output Enable B

0	Output compare B output is disabled
1	Output compare B output is enabled

Timer Overflow Interrupt Enable

0	Timer overflow interrupt request (FOVI) is disabled
1	Timer overflow interrupt request (FOVI) is enabled

Output Compare Interrupt Enable A

0	Output compare interrupt request A (OCIA) is disabled
1	Output compare interrupt request A (OCIA) is enabled

Output Compare Interrupt Enable B

0	Output compare interrupt request B (OCIB) is disabled
1	Output compare interrupt request B (OCIB) is enabled

Input Capture Interrupt Enable

0	Input capture interrupt request (ICI) is disabled
1	Input capture interrupt request (ICI) is enabled

TCSR—Timer Control/Status Register H'91 FRT

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/W	R/W	R/W	R/W

Counter Clear A

0	The FRC is not cleared
1	The FRC is cleared at compare-match A

Input Edge Select

0	FRC contents are transferred to ICR on the falling edge of FTI
1	FRC contents are transferred to ICR on the rising edge of FTI

Output Level A

0	A 0 logic level is output for compare-match A
1	A 1 logic level is output for compare-match A

Output Level B

0	A 0 logic level is output for compare-match B
1	A 1 logic level is output for compare-match B

Timer Overflow

0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when FRC changes from H'FFFF to H'0000

Output Compare Flag A

0	To clear OCFA, the CPU must read OCFA after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when FRC = OCRA

Output Compare Flag B

0	To clear OCFB, the CPU must read OCFB after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when FRC = OCRB

Input Capture Flag

0	To clear ICF, the CPU must read ICF after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when an FTI input signal causes the FRC value to be copied to the ICR

Note: * Software can write a 0 in bits 7 to 4 to clear the flags, but cannot write a 1 in these bits.

FRC (H and L)—Free-Running Counter H'92, H'93 FRT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

OCRA (H and L)—Output Compare Register A H'94, H'95 FRT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OCRA is constantly compared with the FRC value, and the OCFA bit is set to 1 when OCRA = FRC

OCRB (H and L)—Output Compare Register B H'96, H'97 FRT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OCRB is constantly compared with the FRC value, and the OCFB bit is set to 1 when OCRB = FRC

ICR (H and L)—Input Capture Register H'98, H'99 FRT

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Contains FRC count captured on FTI input

TCSR/TCNT—Timer Control/Status Register H'AA WDT

Bit	7	6	5	4	3	2	1	0
	OVF	WT/$\overline{\text{IT}}$	TME	—	RST/$\overline{\text{NMI}}$	CKS2	CKS1	CKS0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/(W)*	R/W	R/W	—	R/W	R/W	R/W	R/W

Clock Select 2 to 0

CKS2	CKS1	CKS0	Clock Source	Overflow Interval ($ø_P$ = 10 MHz)
0	0	0	$ø_P/2$	51.2 µs (Initial value)
0	0	1	$ø_P/32$	819.2 µs
0	1	0	$ø_P/64$	1.6 ms
0	1	1	$ø_P/128$	3.3 ms
1	0	0	$ø_P/256$	6.6 ms
1	0	1	$ø_P/512$	13.1 ms
1	1	0	$ø_P/2048$	52.4 ms
1	1	1	$ø_P/4096$	104.9 ms

Reset or NMI Select

0	NMI function enabled (Initial value)
1	Reset function enabled

Timer Enable

0	TCNT is initialized to H'00 and stopped (Initial value)
1	TCNT runs and requests a reset or interrupt when it overflows

Timer Mode Select

0	Interval timer mode (OVF request) (Initial value)
1	Watchdog timer mode (reset or NMI request)

Overflow Flag

0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit (Initial value)
1	Set to 1 when TCNT changes from H'FF to H'00

TCNT—Timer Counter　　H'AB (read)
H'AA (write)　　WDT

TCNT—Timer Counter | **H'AB (read)** **H'AA (write)** | **WDT**

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

P1PCR—Port 1 Pull-Up MOS Control Register　　H'AC　　P1

Bit	7	6	5	4	3	2	1	0
	$P1_7$PCR	$P1_6$PCR	$P1_5$PCR	$P1_4$PCR	$P1_3$PCR	$P1_2$PCR	$P1_1$PCR	$P1_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port 1 Input Pull-Up Control

0	Input pull-up transistor is off
1	Input pull-up transistor is on

P2PCR—Port 2 Pull-Up MOS Control Register　　H'AD　　P2

Bit	7	6	5	4	3	2	1	0
	$P2_7$PCR	$P2_6$PCR	$P2_5$PCR	$P2_4$PCR	$P2_3$PCR	$P2_2$PCR	$P2_1$PCR	$P2_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port 2 Input Pull-Up Control

0	Input pull-up transistor is off
1	Input pull-up transistor is on

P3PCR—Port 3 Pull-Up MOS Control Register H'AE P3

Bit	7	6	5	4	3	2	1	0
	P3$_7$PCR	P3$_6$PCR	P3$_5$PCR	P3$_4$PCR	P3$_3$PCR	P3$_2$PCR	P3$_1$PCR	P3$_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port 3 Input Pull-Up Control

0	Input pull-up transistor is off
1	Input pull-up transistor is on

P1DDR—Port 1 Data Direction Register H'B0 P1

Bit	7	6	5	4	3	2	1	0
	P1$_7$DDR	P1$_6$DDR	P1$_5$DDR	P1$_4$DDR	P1$_3$DDR	P1$_2$DDR	P1$_1$DDR	P1$_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 1 Input/Output Control

0	Input port
1	Output port

P1DR—Port 1 Data Register H'B2 P1

Bit	7	6	5	4	3	2	1	0
	P1$_7$	P1$_6$	P1$_5$	P1$_4$	P1$_3$	P1$_2$	P1$_1$	P1$_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2DDR—Port 2 Data Direction Register H'B1 P2

Bit	7	6	5	4	3	2	1	0
	P2$_7$DDR	P2$_6$DDR	P2$_5$DDR	P2$_4$DDR	P2$_3$DDR	P2$_2$DDR	P2$_1$DDR	P2$_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 2 Input/Output Control

0	Input port
1	Output port

P2DR—Port 2 Data Register H'B3 P2

Bit	7	6	5	4	3	2	1	0
	P2$_7$	P2$_6$	P2$_5$	P2$_4$	P2$_3$	P2$_2$	P2$_1$	P2$_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3DDR—Port 3 Data Direction Register H'B4 P3

Bit	7	6	5	4	3	2	1	0
	P3$_7$DDR	P3$_6$DDR	P3$_5$DDR	P3$_4$DDR	P3$_3$DDR	P3$_2$DDR	P3$_1$DDR	P3$_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 3 Input/Output Control

0	Input port
1	Output port

P3DR—Port 3 Data Register H'B6 P3

Bit	7	6	5	4	3	2	1	0
	P3$_7$	P3$_6$	P3$_5$	P3$_4$	P3$_3$	P3$_2$	P3$_1$	P3$_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4DDR—Port 4 Data Direction Register H'B5 P4

Bit	7	6	5	4	3	2	1	0
	$P4_7$DDR	$P4_6$DDR	$P4_5$DDR	$P4_4$DDR	$P4_3$DDR	$P4_2$DDR	$P4_1$DDR	$P4_0$DDR
Modes 1 and 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 4 Input/Output Control

0	Input port
1	Output port

P4DR—Port 4 Data Register H'B7 P4

Bit	7	6	5	4	3	2	1	0
	$P4_7$	$P4_6$	$P4_5$	$P4_4$	$P4_3$	$P4_2$	$P4_1$	$P4_0$
Initial value	0	—*	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * Depends on the state of the $P4_6$ pin.

P5DDR—Port 5 Data Direction Register H'B8 P5

Bit	7	6	5	4	3	2	1	0
	—	—	P5$_5$DDR	P5$_4$DDR	P5$_3$DDR	P5$_2$DDR	P5$_1$DDR	P5$_0$DDR
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	W	W	W	W	W	W

Port 5 Input/Output Control

0	Input port
1	Output port

P5DR—Port 5 Data Register H'BA P5

Bit	7	6	5	4	3	2	1	0
	—	—	P5$_5$	P5$_4$	P5$_3$	P5$_2$	P5$_1$	P5$_0$
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	R/W	R/W	R/W	R/W	R/W	R/W

P6DDR—Port 6 Data Direction Register H'B9 P6

Bit	7	6	5	4	3	2	1	0
	—	P6$_6$DDR	P6$_5$DDR	P6$_4$DDR	P6$_3$DDR	P6$_2$DDR	P6$_1$DDR	P6$_0$DDR
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

Port 6 Input/Output Control

0	Input port
1	Output port

P6DR—Port 6 Data Register H'BB P6

Bit	7	6	5	4	3	2	1	0
	—	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P7DDR—Port 7 Data Direction Register H'BC P7

Bit	7	6	5	4	3	2	1	0
	$P7_7$DDR	$P7_6$DDR	$P7_5$DDR	$P7_4$DDR	$P7_3$DDR	$P7_2$DDR	$P7_1$DDR	$P7_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 7 Input/Output Control

0	Input port
1	Output port

P7DR—Port 7 Data Register H'BE P7

Bit	7	6	5	4	3	2	1	0
	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
	—	—	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	1	1	0	0	1	0	0	0
Read/Write	—	—	R/W	R/W	R/W	R/W	R/W	R/W

Wait Count 1 and 0

0	0	No wait states inserted by wait state controller (Initial value)
	1	1 state inserted
1	0	2 states inserted
	1	3 states inserted

Wait Mode Select 1 and 0

0	0	Programmable wait mode
	1	No wait states inserted by wait state controller
1	0	Pin wait mode (Initial value)
	1	Pin auto-wait mode

Clock Double

0	The undivided system clock (ø) is supplied as the clock (øp) for supporting modules (Initial value)
1	The system clock (ø) is divided by two and supplied as the clock (øp) for supporting modules

STCR—Serial Timer Control Register H'C3

Bit	7	6	5	4	3	2	1	0
	(IICS)	(IICX1)	(IICX0)	(SYNCE)	(PWCKE)	(PWCKS)	ICKS1	ICKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Internal Clock Select 1 and 0
See TCSR for details.

Note: The IICS, IICX1, IICX0, SYNCE, PWCKE, and PWCKS bits must not be used (must not be set to 1).

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

RAM Enable

0	On-chip RAM is disabled
1	On-chip RAM is enabled

Host Interface Enable

0	Host interface is disabled (initial value)
1	Host interface is enabled (slave mode)

NMI Edge

0	Falling edge of $\overline{\text{NMI}}$ is detected
1	Rising edge of $\overline{\text{NMI}}$ is detected

External Reset

0	Reset was caused by watchdog timer overflow.
1	Reset was caused by external input (Initial value)

Standby Timer Select 2 to 0

0	0	0	Clock settling time = 8,192 states
0	0	1	Clock settling time = 16,384 states
0	1	0	Clock settling time = 32,768 states
0	1	1	Clock settling time = 65,536 states
1	0	—	Clock settling time = 131,072 states
1	1	—	Unused

Software Standby

0	SLEEP instruction causes transition to sleep mode
1	SLEEP instruction causes transition to software standby mode

MDCR—Mode Control Register H'C5 System Control

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	MDS1	MDS0
Initial value	1	1	1	0	0	1	—*	—*
Read/Write	—	—	—	—	—	—	R	R

Mode Select

Mode pin values

Note: * Initialized according to MD_1 and MD_0 inputs.

ISCR—IRQ Sense Control Register H'C6 System Control

Bit	7	6	5	4	3	2	1	0
	—	IRQ6SC	—	—	—	IRQ2SC	IRQ1SC	IRQ0SC
Initial value	1	0	1	1	1	0	0	0
Read/Write	—	R/W	—	—	—	R/W	R/W	R/W

IRQ_0 Sense Control

IRQOSC	Description
0	The low level of $\overline{IRQ_0}$ generates an interrupt request
1	The falling edge of $\overline{IRQ_0}$ generates an interrupt request

IRQ_1 Sense Control

IRQ1SC	Description
0	The low level of $\overline{IRQ_1}$ generates an interrupt request
1	The falling edge of $\overline{IRQ_1}$ generates an interrupt request

IRQ_2 Sense Control

IRQ2SC	Description
0	The low level of $\overline{IRQ_2}$ generates an interrupt request
1	The falling edge of $\overline{IRQ_2}$ generates an interrupt request

IRQ_6 Sense Control

IRQ6SC	Description
0	The low level of $\overline{KEYIN_0}$ to $\overline{KEYIN_7}$ generates an interrupt request
1	The falling edge of $\overline{KEYIN_0}$ to $\overline{KEYIN_7}$ generates an interrupt request

Bit	7	6	5	4	3	2	1	0
	—	IRQ6E	—	—	—	IRQ2E	IRQ1E	IRQ0E
Initial value	1	0	1	1	1	0	0	0
Read/Write	—	R/W	—	—	—	R/W	R/W	R/W

IRQ Enable

0	IRQ_6 is disabled
1	IRQ_6 is enabled

IRQ Enable

0	IRQ_0/IRQ_1/IRQ_2 is disabled
1	IRQ_0/IRQ_1/IRQ_2 is enabled

TCR—Timer Control Register H'C8 TMR0

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Select 2 to 0

Channel	TCR			STCR		Description
	Bit 2	Bit 1	Bit 0	Bit 1	Bit 0	
0	CKS2	CKS1	CKS0	ICKS1	ICKS0	
	0	0	0	—	—	No clock source (timer stopped)
	0	0	1	—	0	$ø_P$/8 internal clock source, counted on the falling edge
	0	0	1	—	1	$ø_P$/2 internal clock source, counted on the falling edge
	0	1	0	—	0	$ø_P$/64 internal clock source, counted on the falling edge
	0	1	0	—	1	$ø_P$/32 internal clock source, counted on the falling edge
	0	1	1	—	0	$ø_P$/1024 internal clock source, counted on the falling edge
	0	1	1	—	1	$ø_P$/256 internal clock source, counted on the falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on the rising edge
	1	1	0	—	—	External clock source, counted on the falling edge
	1	1	1	—	—	External clock source, counted on both the rising and falling edges
1	0	0	0	—	—	No clock source (timer stopped)
	0	0	1	0	—	$ø_P$/8 internal clock source, counted on the falling edge
	0	0	1	1	—	$ø_P$/2 internal clock source, counted on the falling edge
	0	1	0	0	—	$ø_P$/64 internal clock source, counted on the falling edge
	0	1	0	1	—	$ø_P$/128 internal clock source, counted on the falling edge
	0	1	1	0	—	$ø_P$/1024 internal clock source, counted on the falling edge
	0	1	1	1	—	$ø_P$/2048 internal clock source, counted on the falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on the rising edge
	1	1	0	—	—	External clock source, counted on the falling edge
	1	1	1	—	—	External clock source, counted on both the rising and falling edges

Counter Clear 1 and 0

0	0	Not cleared
0	1	Cleared on compare-match A
1	0	Cleared on compare-match B
1	1	Cleared on rising edge of external reset input signal

Timer Overflow Interrupt Enable

0	The timer overflow interrupt request (OVI) is disabled
1	The timer overflow interrupt request (OVI) is enabled

Compare-Match Interrupt Enable A

0	Compare-match interrupt request A (CMIA) is disabled
1	Compare-match interrupt request A (CMIA) is enabled

Compare-Match Interrupt Enable B

0	Compare-match interrupt request B (CMIB) is disabled
1	Compare-match interrupt request B (CMIB) is enabled

TCSR—Timer Control/Status Register H'C9 TMR0

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	PWME	OS3[*1]	OS2[*1]	OS1[*1]	OS0[*1]
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)[*2]	R/(W)[*2]	R/(W)[*2]	R/W	R/W	R/W	R/W	R/W

Output Select 1 and 0

0	0	No change when compare-match A occurs
0	1	Output changes to 0 when compare-match A occurs
1	0	Output changes to 1 when compare-match A occurs
1	1	Output inverts (toggles) when compare-match A occurs

Output Select 3 and 2

0	0	No change when compare-match B occurs
0	1	Output changes to 0 when compare-match B occurs
1	0	Output changes to 1 when compare-match B occurs
1	1	Output inverts (toggles) when compare-match B occurs

PWM Mode Enable

0	Normal timer mode	(Initial value)
1	PWM mode	

Timer Overflow Flag

0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when TCNT changes from H'FF to H'00

Compare-Match Flag A

0	To clear CMFA, the CPU must read CMFA after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when TCNT = TCORA

Compare-Match Flag B

0	To clear CMFB, the CPU must read CMFB after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when TCNT = TCORB

Notes: [*1] When all four output select bits (bits OS3 to OS0) are cleared to 0, the timer output signal is disabled.

[*2] Software can write a 0 in bits 7 to 5 to clear the flags, but cannot write a 1 in these bits.

TCORA—Time Constant Register A H'CA TMR0

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The CMFA bit is set to 1 when TCORA = TCNT

TCORB—Time Constant Register B H'CB TMR0

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The CMFB bit is set to 1 when TCORB = TCNT

TCNT—Timer Counter H'CC TMR0

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

TCR—Timer Control Register H'D0 TMR1

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

TCSR—Timer Status Control Register H'D1 TMR1

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	PWME	OS3*1	OS2*1	OS1*1	OS0*1
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*2	R/(W)*2	R/(W) *2	R/W	R/W	R/W	R/W	R/W

Notes: Bit functions are the same as for TMR0.

*1. When all four output select bits (bits OS3 to OS0) are cleared to 0, the timer output signal is disabled.

*2. Software can write a 0 in bits 7 to 5 to clear the flags, but cannot write a 1 in these bits.

TCORA—Time Constant Register A H'D2 TMR1

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

TCORB—Time Constant Register B H'D3 TMR1

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

TCNT—Timer Counter H'D4 TMR1

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

Bit	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	MP	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Select

0	0	ø clock
0	1	ø$_P$/4 clock
1	0	ø$_P$/16 clock
1	1	ø$_P$/64 clock

Multiprocessor Mode

0	Multiprocessor function disabled
1	Multiprocessor format selected

Stop Bit Length

0	One stop bit
1	Two stop bits

Parity Mode

0	Even parity
1	Odd parity

Parity Enable

0	Transmit: No parity bit is added Receive: Parity is not checked
1	Transmit: A parity bit is added Receive: Parity is checked

Character Length

0	8 bits per character
1	7 bits per character

Communication Mode

0	Asynchronous communication
1	Synchronous communication

BRR—Bit Rate Register H'D9 SCI0

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets the bit rate

TDR—Transmit Data Register 0 H'DB SCI0

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Stores transmit data

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Enable 0

0	The SCK pin is not used by the SCI
1	The SCK pin is used for serial clock output

Clock Enable 1

0	Internal clock source is selected
1	External clock source is selected

Transmit End Interrupt Enable

0	TSR-empty interrupt request is disabled
1	TSR-empty interrupt request is enabled

Multiprocessor Interrupt Enable

0	Multiprocessor receive interrupt function is disabled
1	Multiprocessor receive interrupt function is enabled

Receive Enable

0	The receive function is disabled
1	The receive function is enabled

Transmit Enable

0	The transmit function is disabled
1	The transmit function is enabled

Receive Interrupt Enable

0	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are disabled
1	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are enabled

Transmit Interrupt Enable

0	The TDR-empty interrupt request (TXI) is disabled
1	The TDR-empty interrupt request (TXI) is enabled

SSR—Serial Status Register H'DC SCI0

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Multiprocessor Bit Transfer

0	Multiprocessor bit = 0 in transmit data
1	Multiprocessor bit = 1 in transmit data

Multiprocessor Bit

0	Multiprocessor bit = 0 in receive data
1	Multiprocessor bit = 1 in receive data

Transmit End

0	Cleared by reading TDRE = 1, then writing 0 in TDRE
1	Set to 1 when TE = 0, or when TDRE = 1 at the end of character transmission

Parity Error

0	To clear PER, the CPU must read PER after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when a parity error occurs (the parity of the received data does not match the parity selected by the O/$\overline{E}$ bit in SMR)

Framing Error

0	To clear FER, the CPU must read FER after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 if a framing error occurs (stop bit = 0)

Overrun Error

0	To clear OER, the CPU must read OER after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 if reception of the next character ends while the receive data register is still full (RDRF = 1)

Receive Data Register Full

0	To clear RDRF, the CPU must read RDRF after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 when one character is received without error and transferred from RSR to RDR

Transmit Data Register Empty

0	To clear TDRE, the CPU must read TDRE after it has been set to 1, then write a 0 in this bit
1	This bit is set to 1 at the following times: 1. When TDR contents are transferred to TSR 2. When the TE bit in SCR is cleared to 0

Note: * Software can write a 0 in bits 7 to 3 to clear the flags, but cannot write a 1 in these bits.

RDR—Receive Data register H'DD SCI0

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Stores receive data

SCMR—Serial Communication Mode Register H'DE SCI0

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	SDIR	SINV	—	SMIF
Initial value	1	1	1	1	0	0	1	0
Read/Write	—	—	—	—	R/W	R/W	—	R/W

Serial Communication Mode Select

0	Normal SCI mode (Initial value)
1	Reserved mode

Data Invert

0	TDR contents are transmitted as they are (Initial value) TDR contents are stored in RDR as they are
1	TDR contents are inverted before being transmitted Receive data is stored in RDR in inverted form

Data Transfer Direction

0	TDR contents are transmitted LSB-first (Initial value) Receive data is stored in RDR LSB-first
1	TDR contents are transmitted MSB-first Receive data is stored in RDR MSB-first

SMR—Serial Mode Register 1 H'E0 SCI1

Bit	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	MP	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for SCI0.

BRR—Bit Rate Register H'E1 SCI1

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for SCI0.

SCR—Serial Control Register H'E2 SCI1

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for SCI0.

TDR—Transmit Data Register H'E3 SCI1

Bit	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for SCI0.

SSR—Serial Status Register 1 H'E4 SCI1

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: Bit functions are the same as for SCI0.

RDR—Receive Data register 1 H'E5 SCI1

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Note: Bit functions are the same as for SCI0.

HICR—Host Interface Control Register H'F0 HIF

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	IBFIE2	IBFIE1	FGA20E
Initial value	1	1	1	1	1	0	0	0
Slave Read/Write	—	—	—	—	—	R/W	R/W	R/W
Host Read/Write	—	—	—	—	—	—	—	—

Fast Gate A_{20} Enable

0	Disables fast A_{20} gate function (Initial value)
1	Enables fast A_{20} gate function

Input Buffer Full Interrupt Enable 1

0	IDR1 input buffer full interrupt is disabled (Initial value)
1	IDR1 input buffer full interrupt is enabled

Input Buffer Full Interrupt Enable 2

0	IDR2 input buffer full interrupt is disabled (Initial value)
1	IDR2 input buffer full interrupt is enabled

KMIMR—Keyboard Matrix Interrupt Mask Register H'F1 System Control

Bit	7	6	5	4	3	2	1	0
	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Keyboard Matrix Interrupt Mask

0	Key-sense input interrupt request is enabled
1	Key-sense input interrupt request is disabled (Initial value)

KMPCR—Key-Sense MOS Pull-Up Control Register H'F2 P6/P7

Bit	7	6	5	4	3	2	1	0
	KM$_7$PCR	KM$_6$PCR	KM$_5$PCR	KM$_4$PCR	KM$_3$PCR	KM$_2$PCR	KM$_1$PCR	KM$_0$PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port 6 Input MOS Pull-Up Control

0	The input MOS pull-up is off
1	The input MOS pull-up is on

IDR1—Input Data Register H'F4 HIF

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R	R	R	R	R	R	R	R
Host Read/Write	W	W	W	W	W	W	W	W

ODR1—Output Data Register H'F5 HIF

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Host Read/Write	R	R	R	R	R	R	R	R

STR1—Status Register H'F6 HIF

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	$C/\overline{D}$	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

Output Buffer Full

0	This bit is cleared when the host processor read ODR1 (Initial value)
1	This bit is set when the slave processor writes to ODR1

Input Buffer Full

0	This bit is cleared when the slave processor reads IDR1 (Initial value)
1	This bit is set when the host processor writes to IDR1

Command/Data

0	Contents of IDR1 are data (Initial value)
1	Contents of IDR1 are a command

Defined by User

The user can use these bits as necessary

IDR2—Input Data Register H'FC HIF

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R	R	R	R	R	R	R	R
Host Read/Write	W	W	W	W	W	W	W	W

ODR2—Output Data Register H'FD HIF

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Host Read/Write	R	R	R	R	R	R	R	R

STR2—Status Register H'FE HIF

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	C/$\overline{D}$	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

Output Buffer Full

0	This bit is cleared when the host processor read ODR1 (Initial value)
1	This bit is set when the slave processor writes to ODR1

Input Buffer Full

0	This bit is cleared when the slave processor reads IDR1 (Initial value)
1	This bit is set when the host processor writes to IDR1

Command/Data

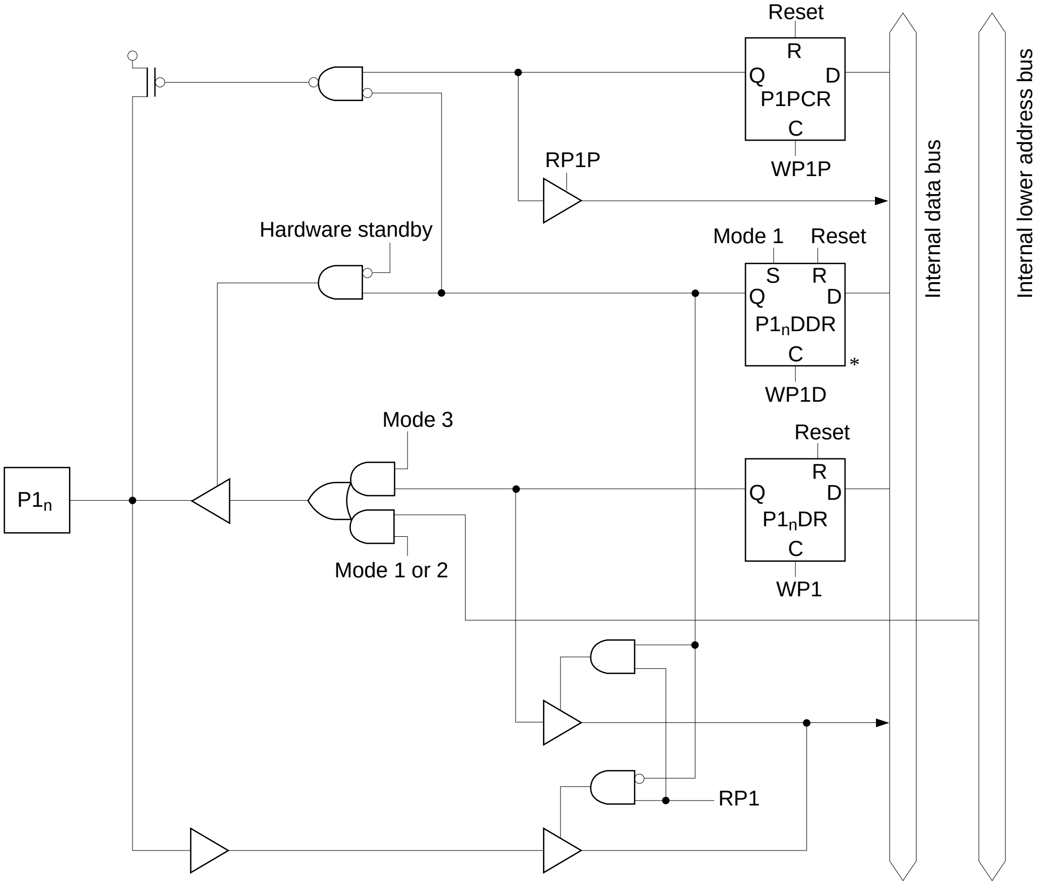
0	Contents of IDR1 are data (Initial value)
1	Contents of IDR1 are a command

Defined by User

The user can use these bits as necessary

Appendix C I/O Port Block Diagrams

C.1 Port 1 Block Diagram

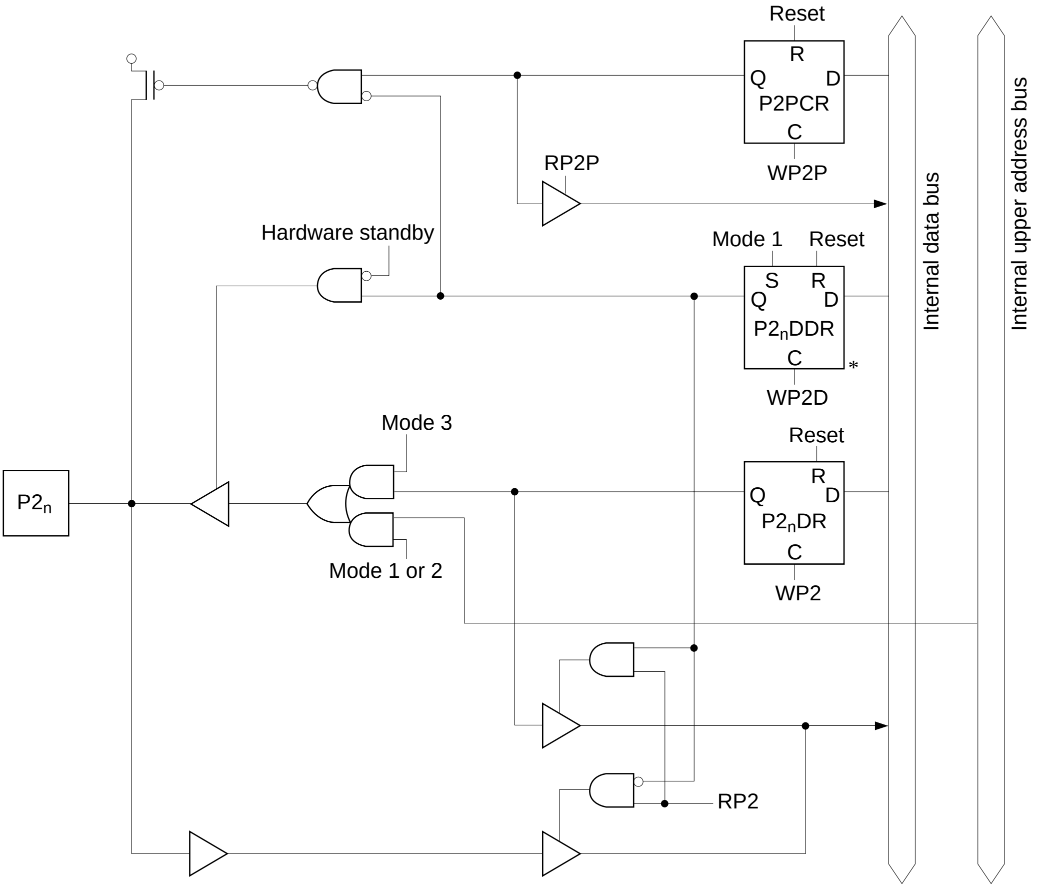


WP1P: Write to P1PCR
WP1D: Write to P1DDR
WP1: Write to port 1
RP1P: Read P1PCR
RP1: Read port 1
n = 0 to 7

Note: * Set priority

Figure C-1 Port 1 Block Diagram

C.2 Port 2 Block Diagram



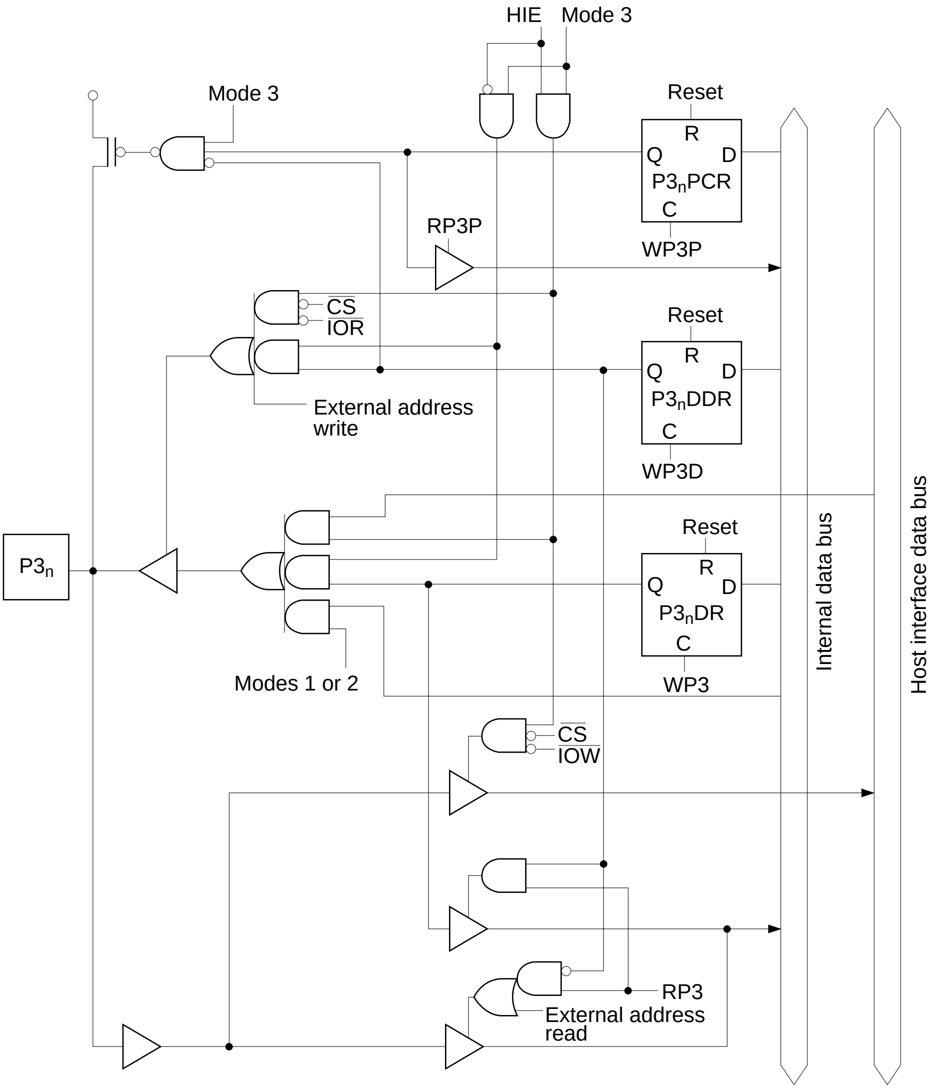
WP2P:	Write to P2PCR
WP2D:	Write to P2DDR
WP2:	Write to port 2
RP2P:	Read P2PCR
RP2:	Read port 2

n = 0 to 7

Note: * Set priority

Figure C-2 Port 2 Block Diagram

C.3 Port 3 Block Diagram



WP3P: Write to P3PCR
WP3D: Write to P3DDR
WP3: Write to port 3
RP3P: Read P3PCR
RP3: Read port 3
n = 0 to 7

Figure C-3 Port 3 Block Diagram

C.4 Port 4 Block Diagrams

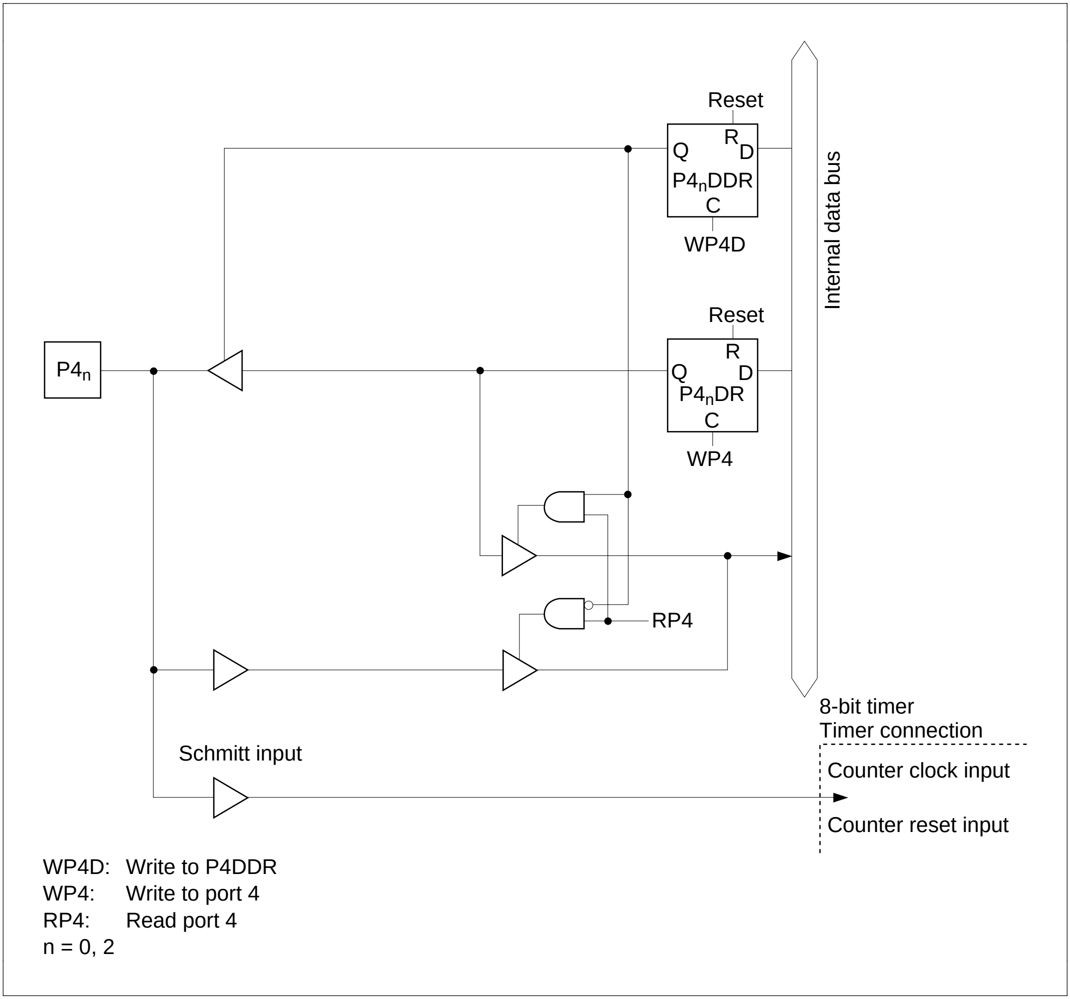


Figure C-4 (a) Port 4 Block Diagram (Pins $P4_0$, $P4_2$)

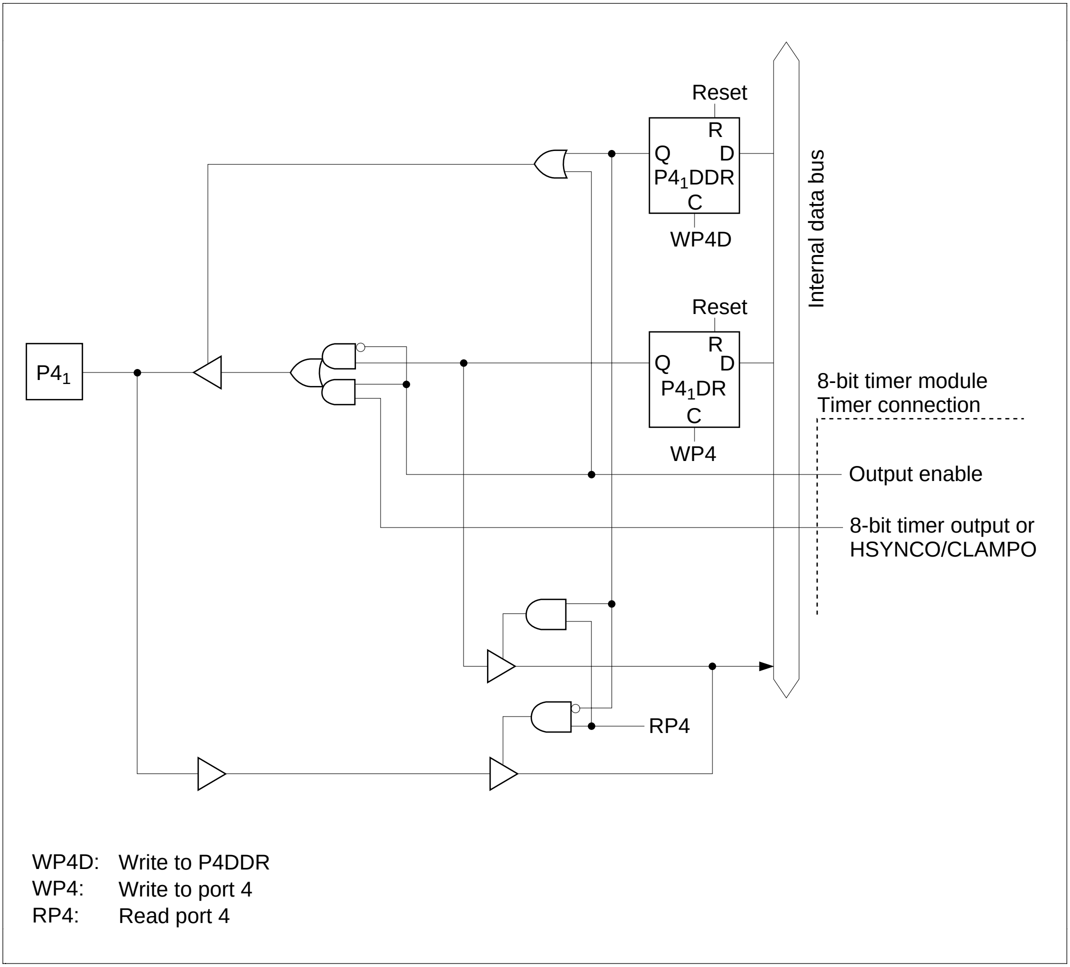


Figure C-4 (b) Port 4 Block Diagram (Pin $P4_1$)

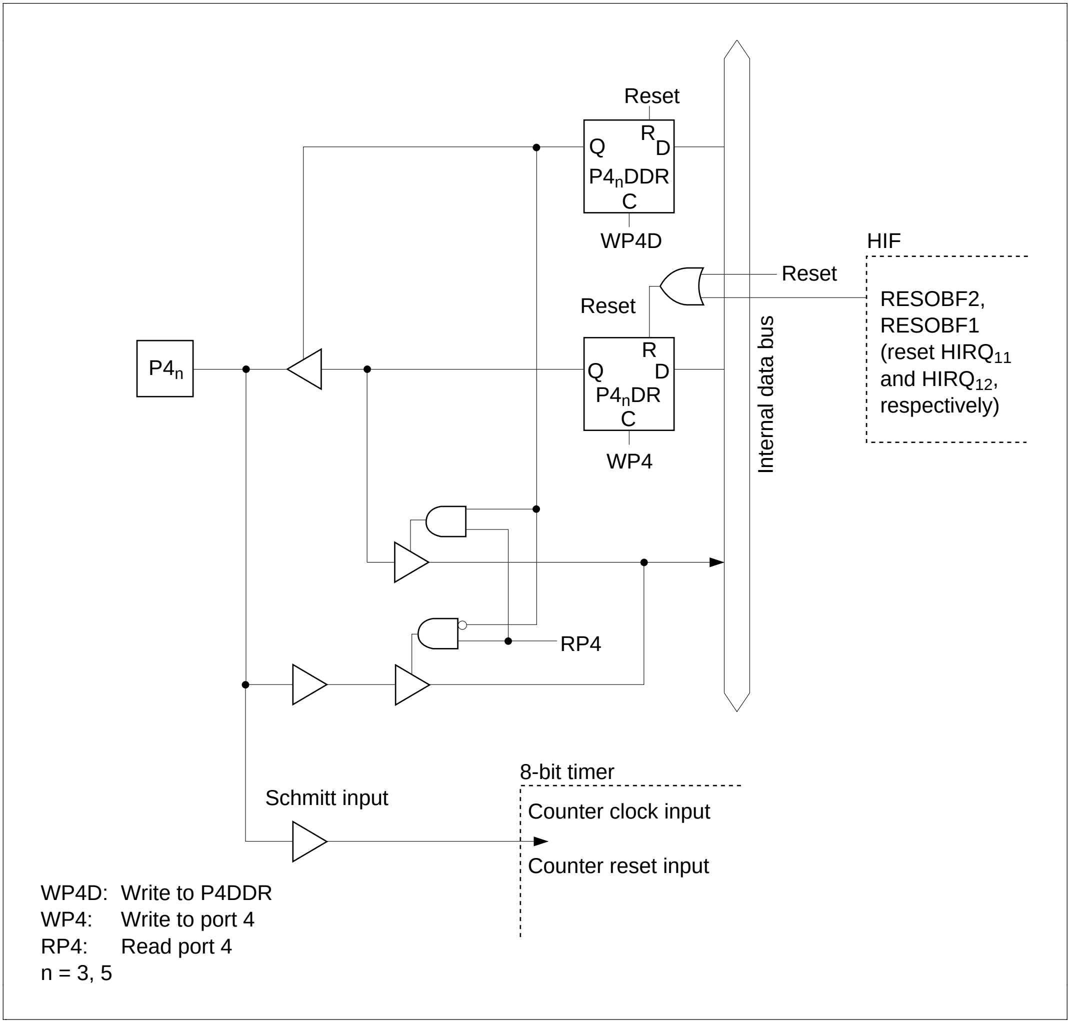
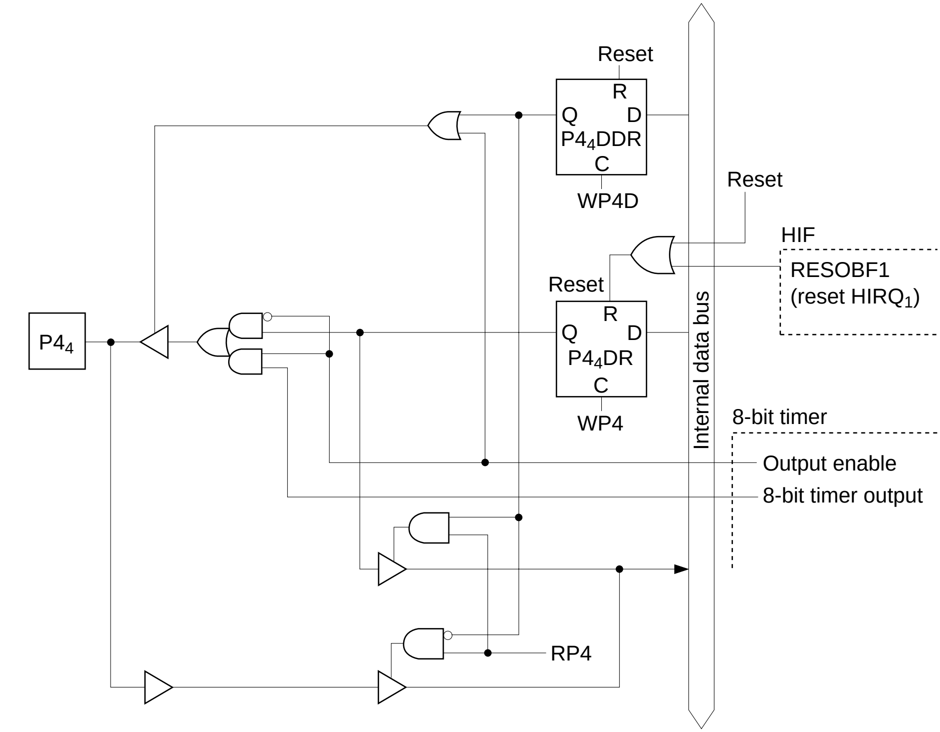
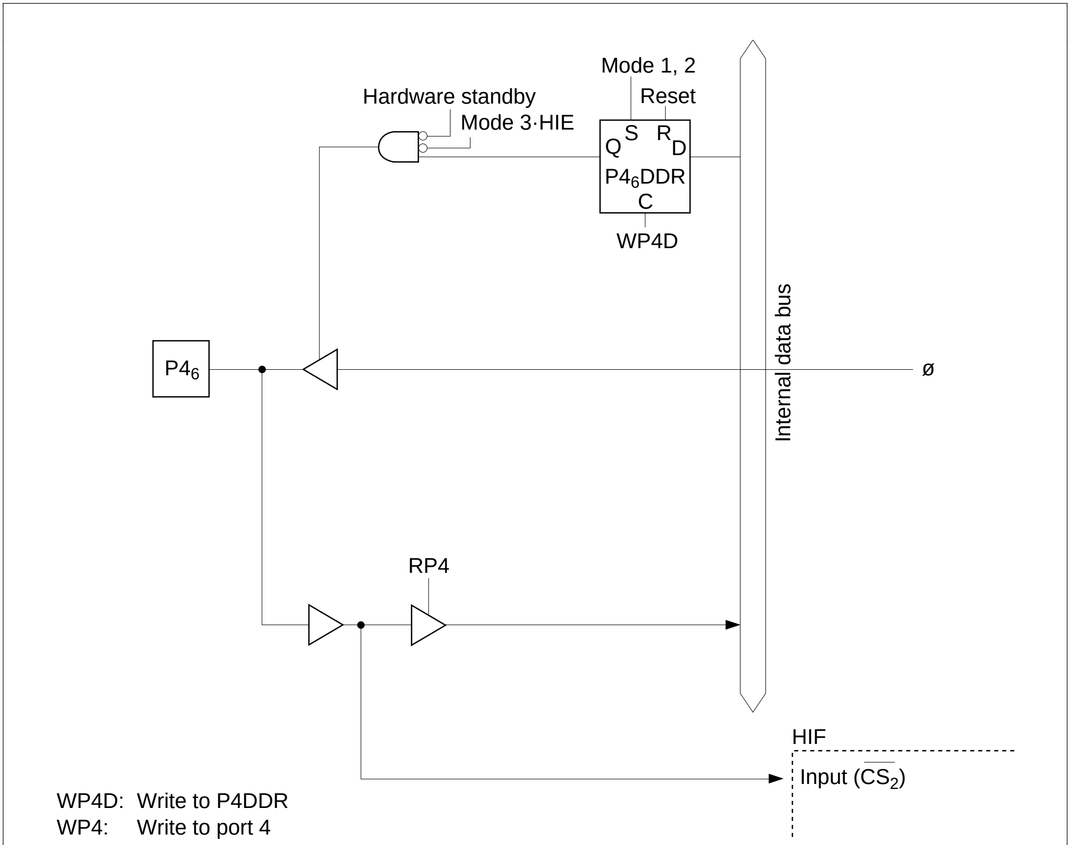


Figure C-4 (c) Port 4 Block Diagram (Pins $P4_3$, $P4_5$)



WP4D: Write to P4DDR
WP4: Write to port 4
RP4: Read port 4

Figure C-4 (d) Port 4 Block Diagram (Pin $P4_4$)



WP4D: Write to P4DDR
WP4: Write to port 4
RP4: Read port 4

Figure C-4 (e) Port 4 Block Diagram (Pin $P4_6$)

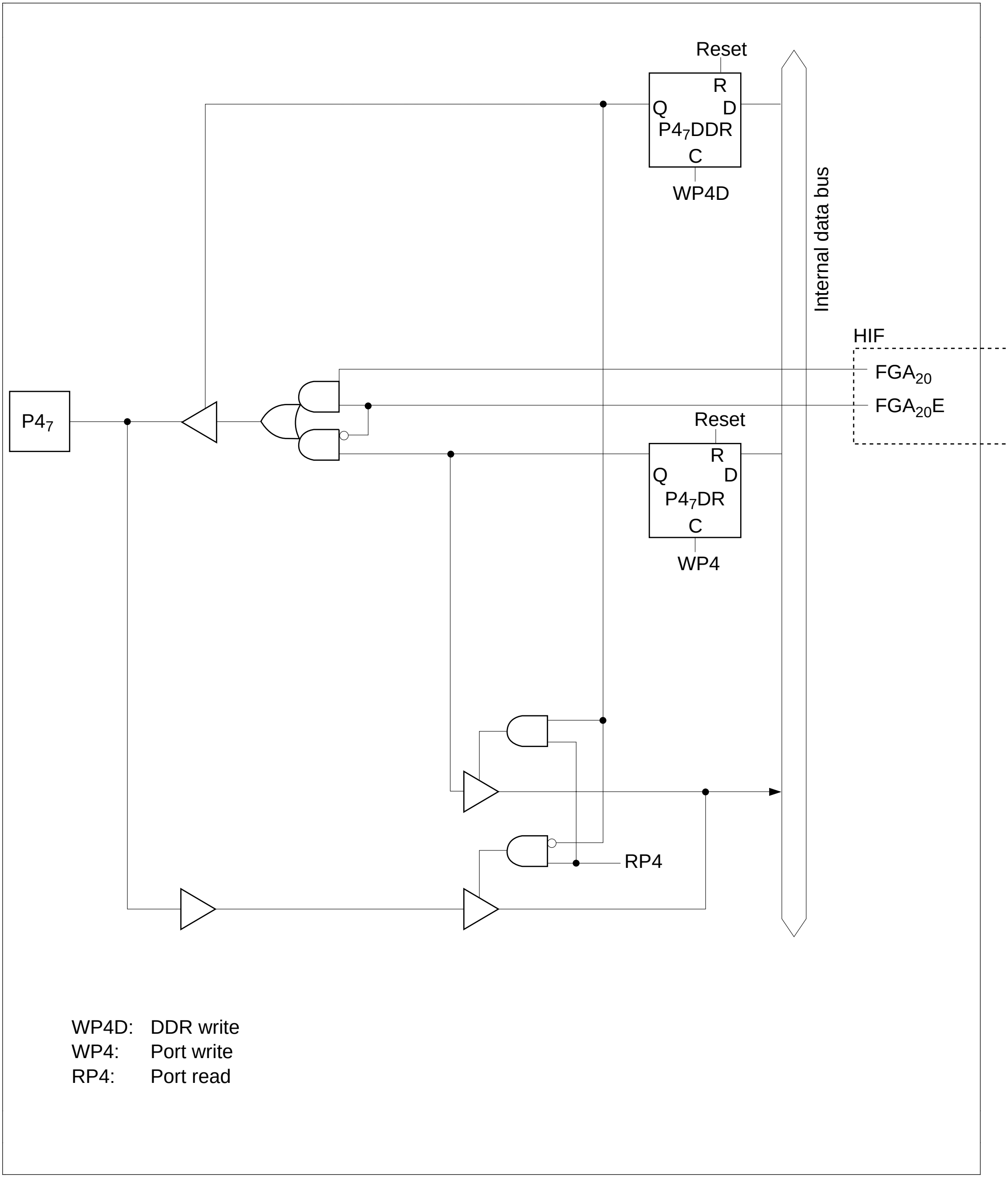
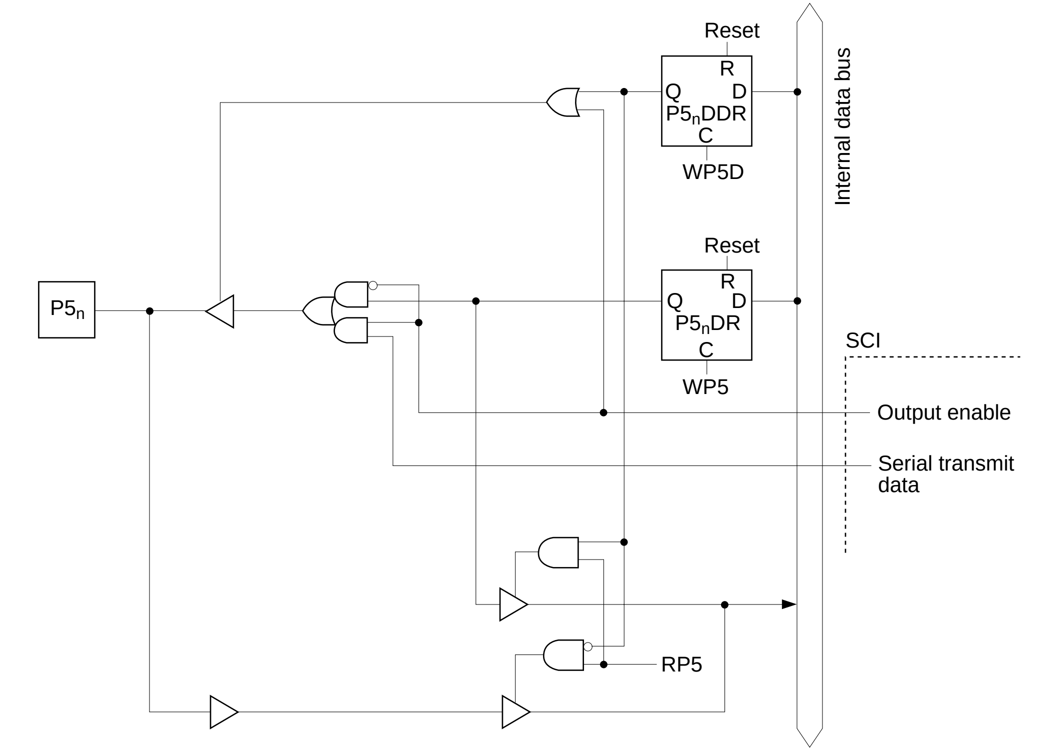


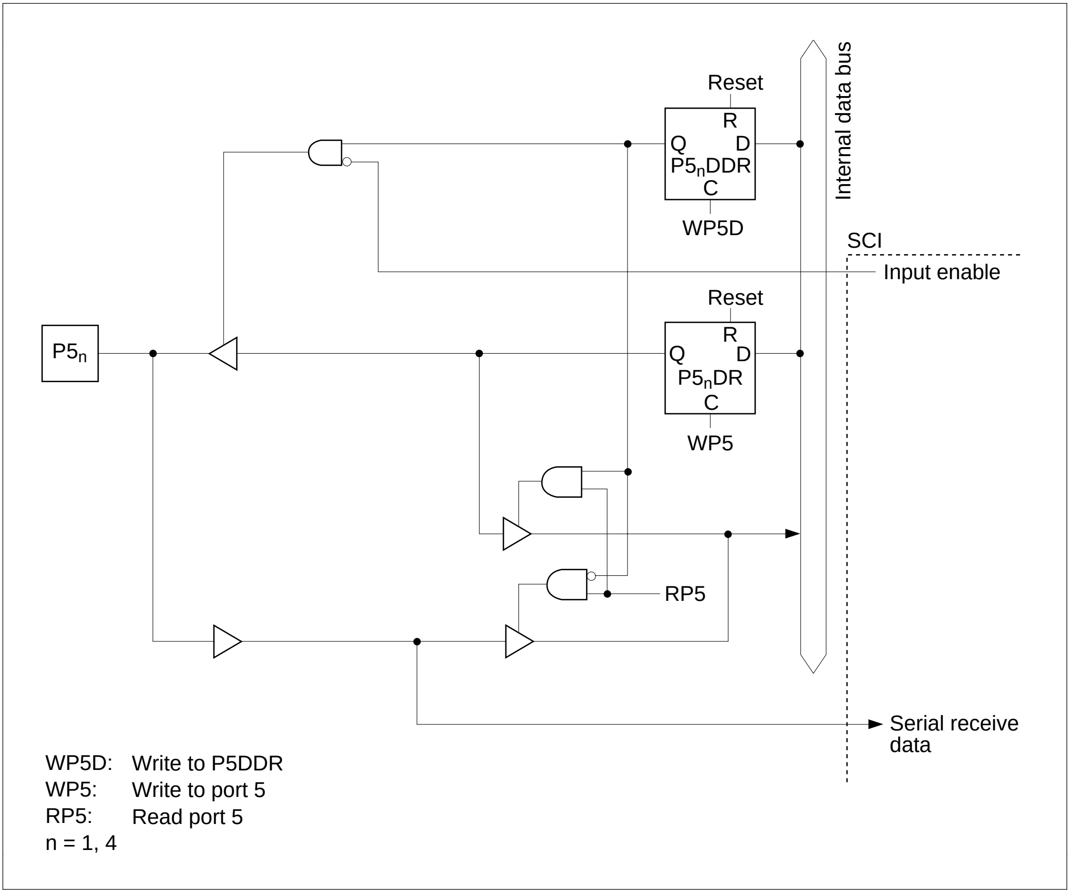
Figure C-4 (f) Port 4 Block Diagram (Pin P4$_7$)

C.5 Port 5 Block Diagrams



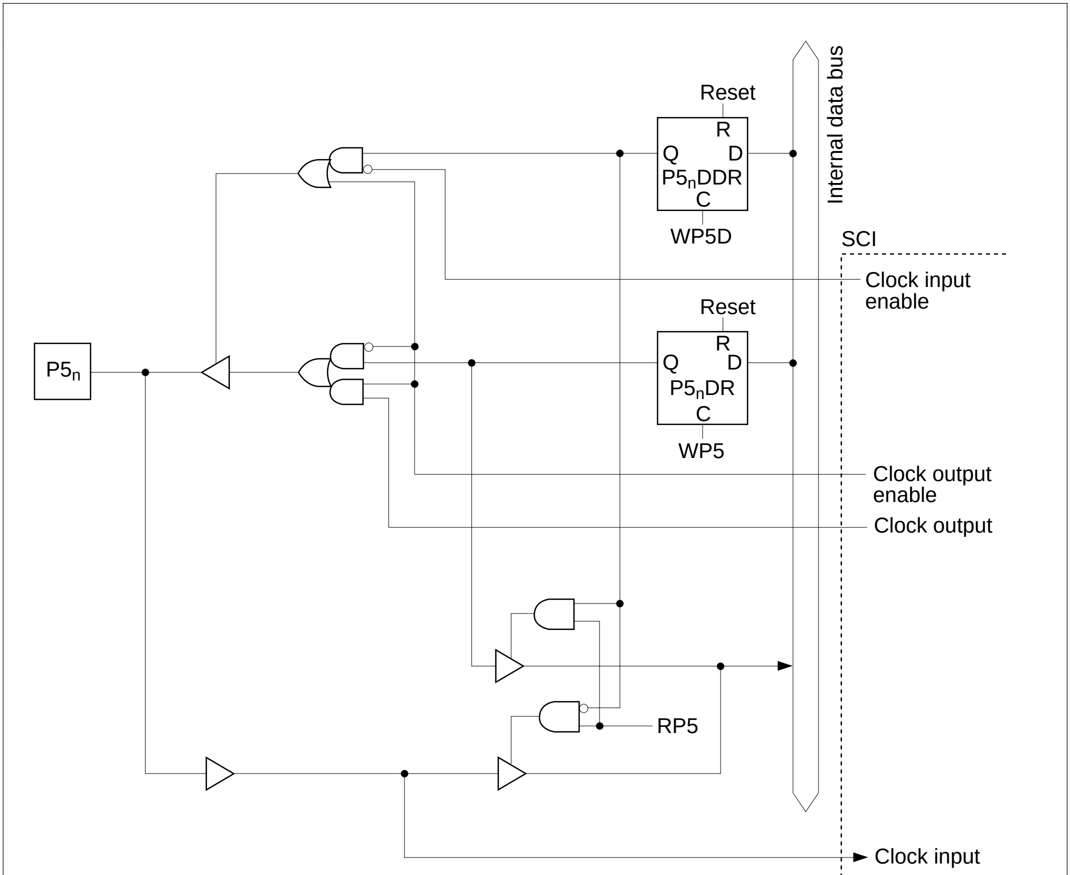
WP5D: Write to P5DDR
WP5: Write to port 5
RP5: Read port 5
n = 0, 3

Figure C-5 (a) Port 5 Block Diagram (Pins $P5_0$, $P5_3$)



WP5D: Write to P5DDR
WP5: Write to port 5
RP5: Read port 5
n = 1, 4

Figure C-5 (b) Port 5 Block Diagram (Pins $P5_1$, $P5_4$)



WP5D: Write to P5DDR
WP5: Write to port 5
RP5: Read port 5
n = 2, 5

Figure C-5 (c) Port 5 Block Diagram (Pins $P5_2$, $P5_5$)

C.6 Port 6 Block Diagrams

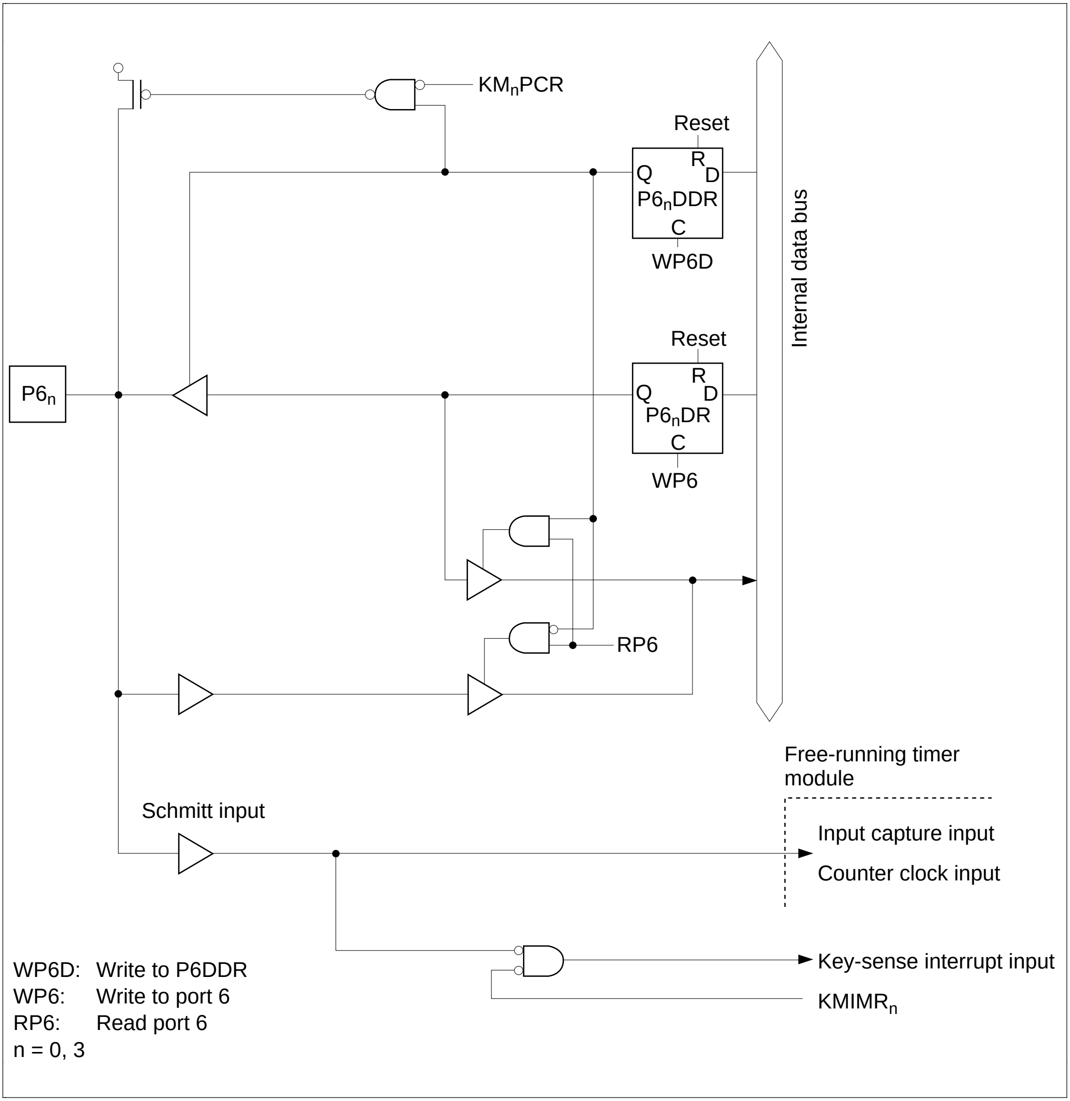
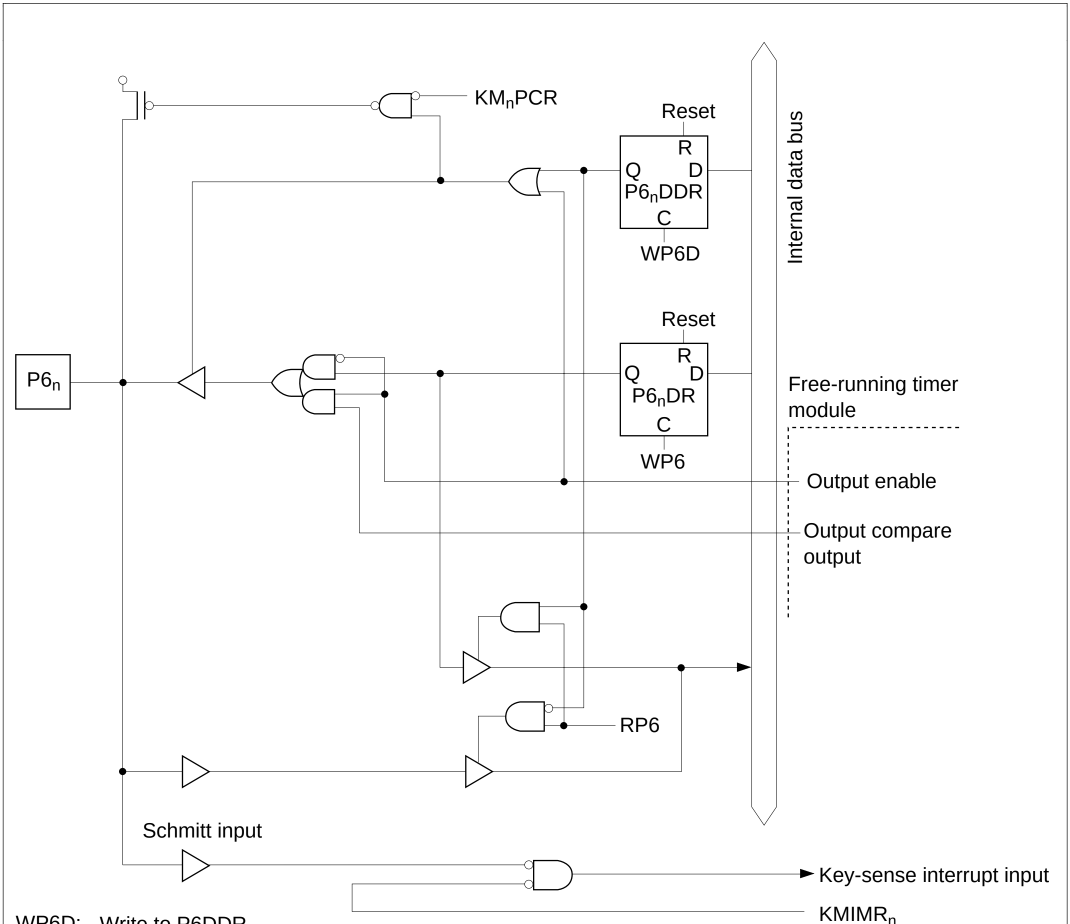


Figure C-6 (a) Port 6 Block Diagram (Pins P6$_0$, P6$_3$)



WP6D:	Write to P6DDR
WP6:	Write to port 6
RP6:	Read port 6

n = 1, 2

Figure C-6 (b) Port 6 Block Diagram (Pins $P6_1$, $P6_2$)

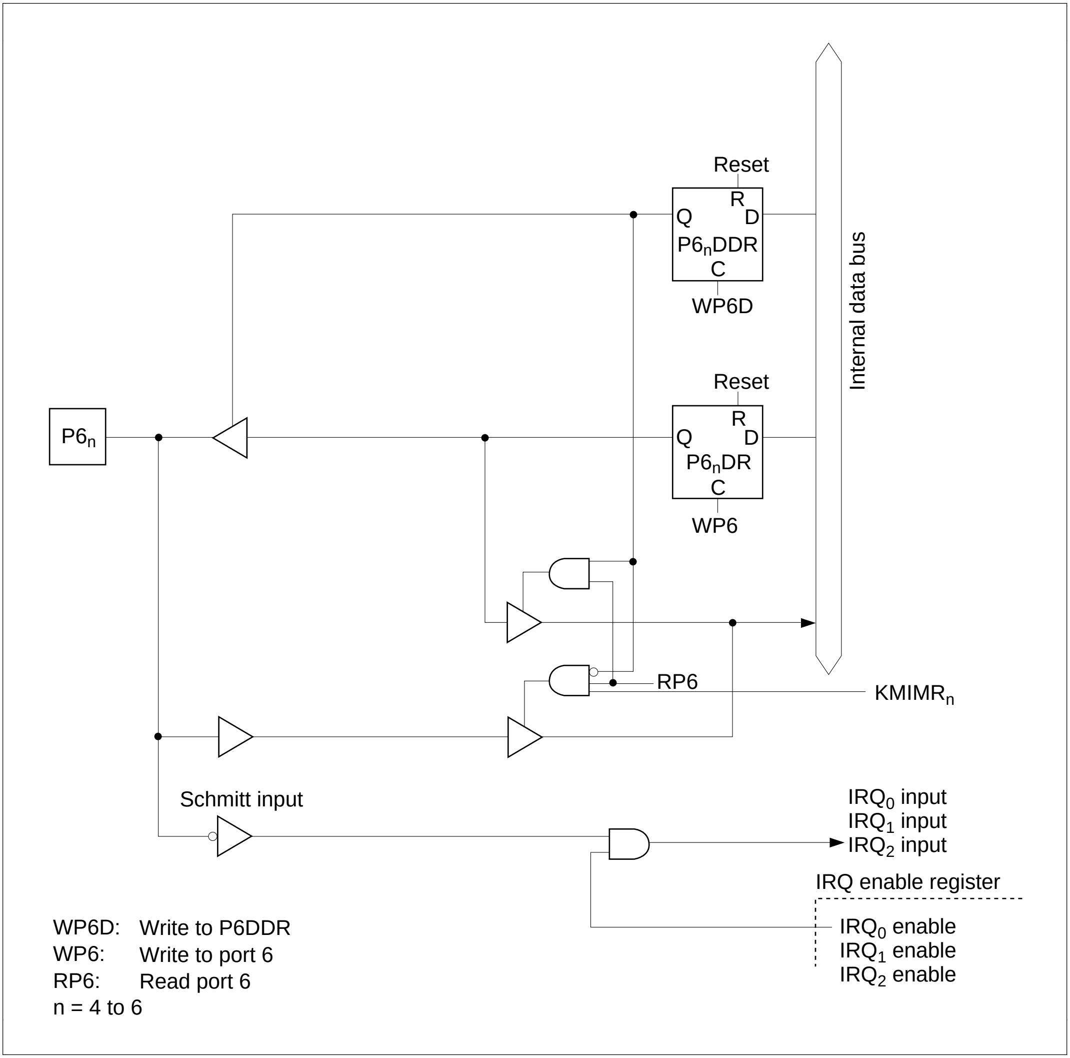


Figure C-6 (c) Port 6 Block Diagram (Pins $P6_4$, $P6_5$, $P6_6$)

C.7 Port 7 Block Diagrams

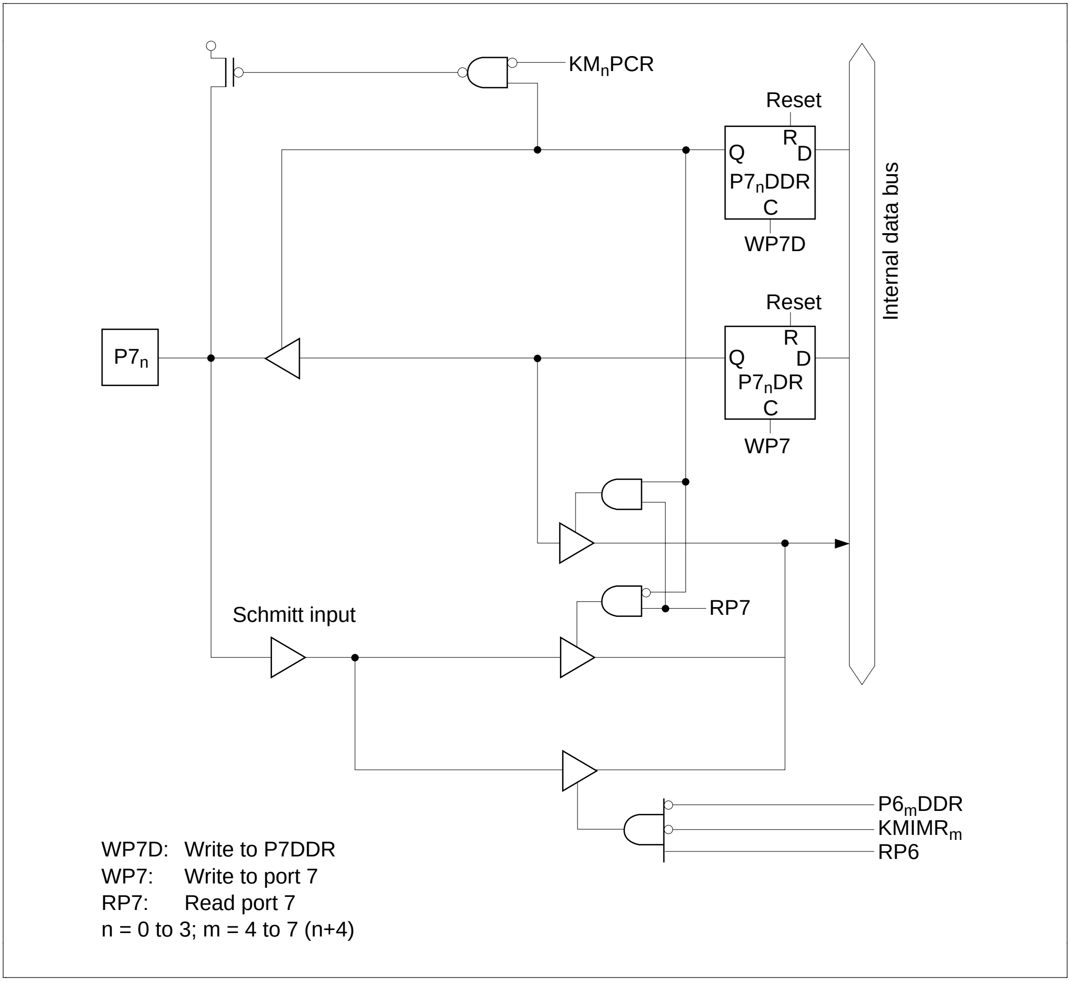
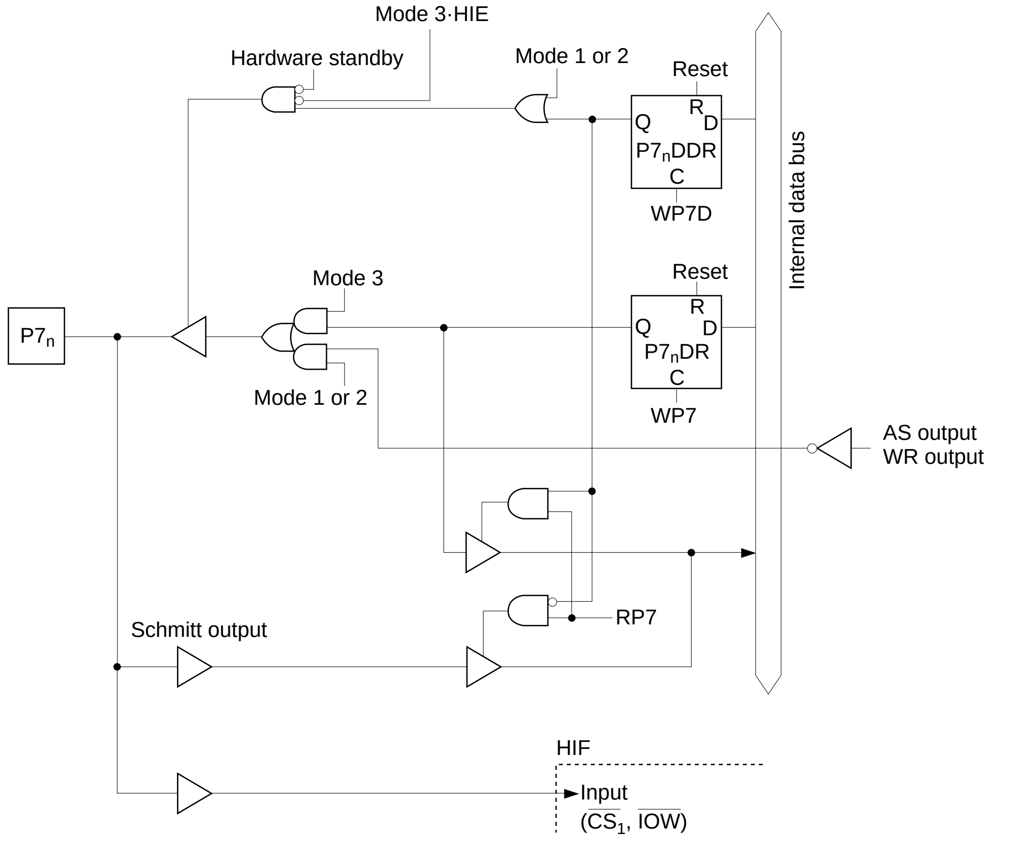


Figure C-7 (a) Port 7 Block Diagram (Pins $P7_0$, $P7_1$, $P7_2$, $P7_3$)



WP7D: Write to P7DDR
WP7: Write to port 7
RP7: Read port 7
n = 4, 5

Figure C-7 (b) Port 7 Block Diagram (Pins $P7_4$, $P7_5$)

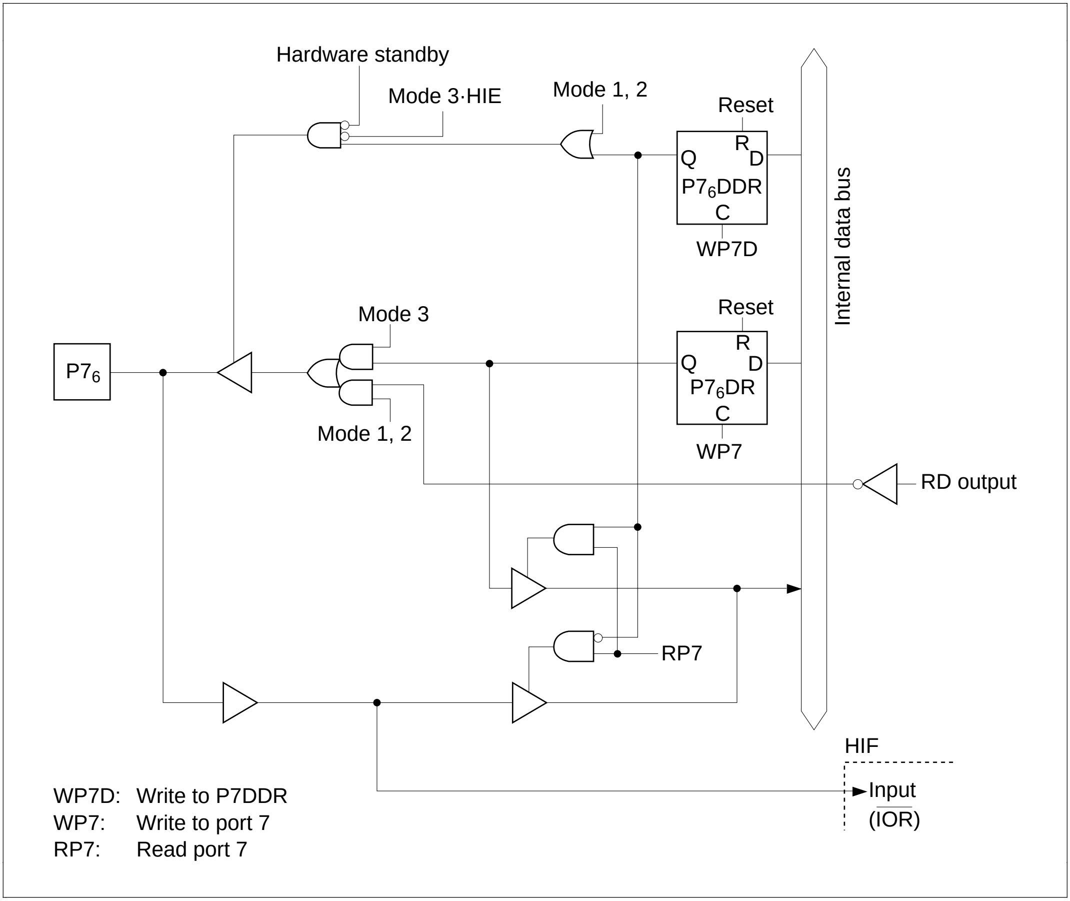


Figure C-7 (c) Port 7 Block Diagram (Pin P7$_6$)

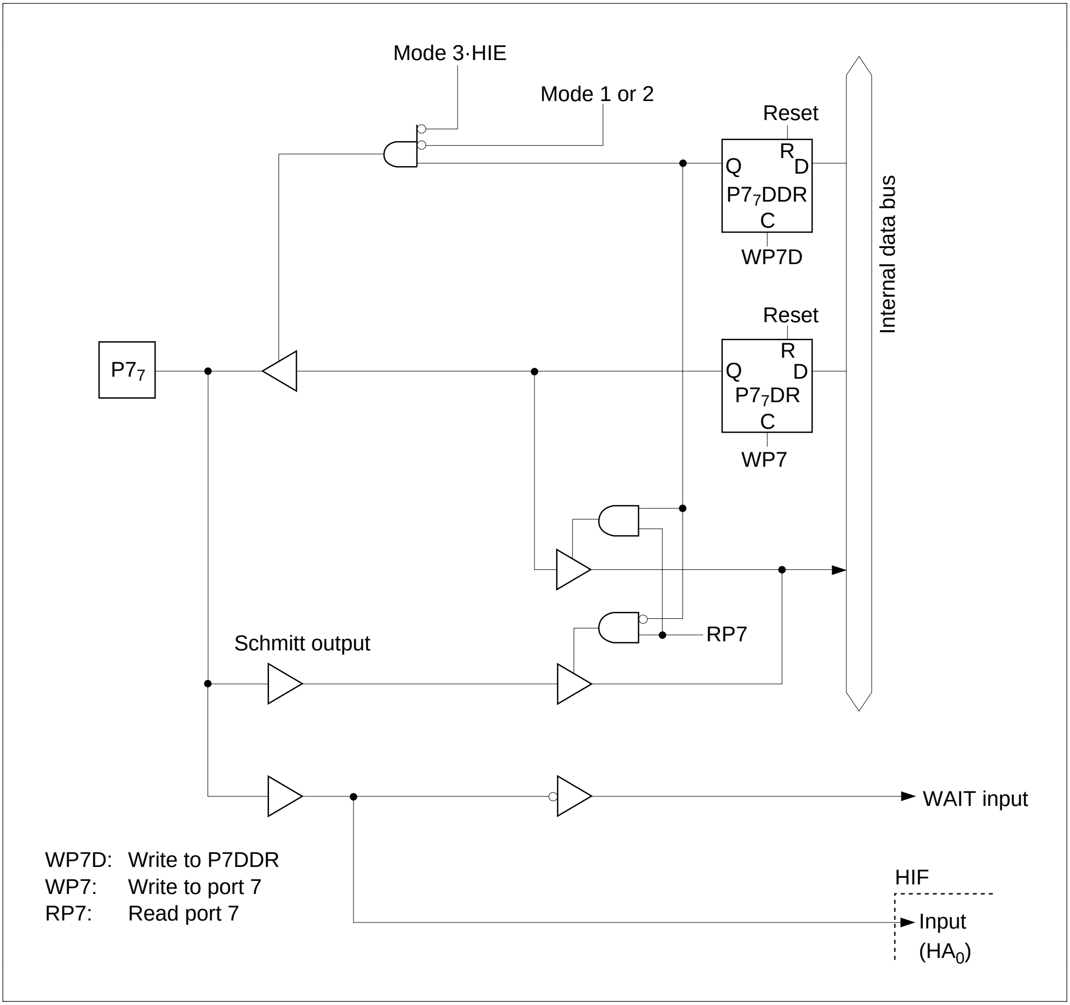


Figure C-7 (d)   Port 7 Block Diagram (Pin P7$_7$)

Appendix D Pin States

Table D-1 Port States in Each Mode

Pin Name	MCU Mode	Reset	Hardware Standby	Software Standby	Sleep Mode	Normal Operation
$P1_7$ to $P1_0$	1	Low	3-state	Low	Prev. state (Addr. output pins: last address accessed)	A_7 to A_0
A_7 to A_0	2	3-state		Low if DDR = 1, prev. state if DDR = 0		Addr. output or input port
	3			Prev. state		I/O port
$P2_7$ to $P2_0$	1	Low	3-state	Low	Prev. state (Addr. output pins: last address accessed)	A_{15} to A_8
A_{15} to A_8	2	3-state		Low if DDR = 1, prev. state if DDR = 0		Addr. output or input port
	3			Prev. state		I/O port
$P3_7$ to $P3_0$	1	3-state	3-state	3-state	3-state	D_7 to D_0
D_7 to D_0	2					
	3			Prev. state	Prev. state	I/O port
$P4_5$ to $P4_0$	1	3-state	3-state	Prev. state (note)	Prev. state	I/O port
	2					
	3					
$P4_6$/ø	1	Clock output	3-state	High	Clock output	Clock output
	2					
	3	3-state		High if DDR = 1, 3-state if DDR = 0	Clock output if DDR = 1, 3-state if DDR = 0	Clock output if DDR = 1, input port if DDR = 0
$P4_7$	1	3-state	3-state	Prev. state (note)	Prev. state	I/O port
	2					
	3					
$P5_5$ to $P5_0$	1	3-state	3-state	Prev. state (note)	Prev. state	I/O port
	2					
	3					

Table D-1 Port States in Each Mode

Pin Name	MCU Mode	Reset	Hardware Standby	Software Standby	Sleep Mode	Normal Operation
$P6_6$ to $P6_0$	1	3-state	3-state	Prev. state (note)	Prev. state	I/O port
	2					
	3					
$P7_7$/$\overline{\text{WAIT}}$	1	3-state	3-state	3-state/prev. state	3-state/prev. state	$\overline{\text{WAIT}}$/ I/O port
	2					
	3			Prev. state	Prev. state	I/O port
$P7_6$ to $P7_4$ $\overline{\text{AS}}$, $\overline{\text{WR}}$, $\overline{\text{RD}}$	1	High	3-state	High	High	$\overline{\text{AS}}$, $\overline{\text{WR}}$, $\overline{\text{RD}}$
	2					
	3	3-state		Prev. state	Prev. state	I/O port
$P7_3$ to $P7_0$	1	3-state	3-state	Prev. state (note)	Prev. state	I/O port
	2					
	3					

Legend

3-state: High-impedance state

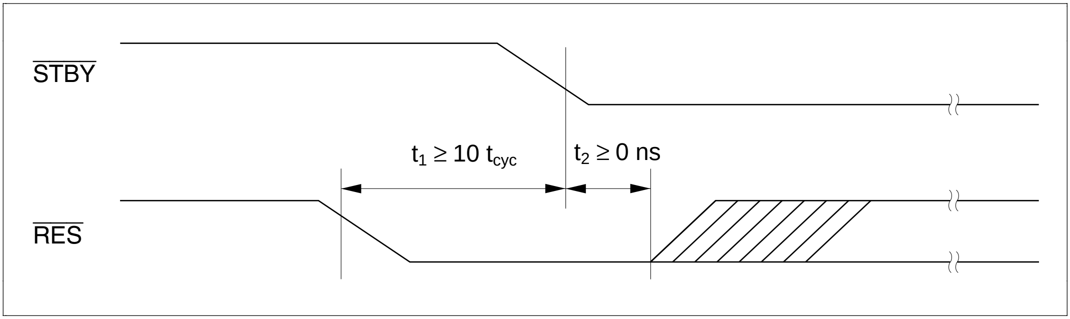
Prev. state: Input pins are in the high-impedance state (for pins with a built-in MOS input pull-up the pull-up remains on when DDR = 0 and PCR = 1); output pins retain their previous state.

Note: As on-chip supporting modules are initialized, general input or output is determined by DDR and DR.

Appendix E Timing of Transition to and Recovery from Hardware Standby Mode

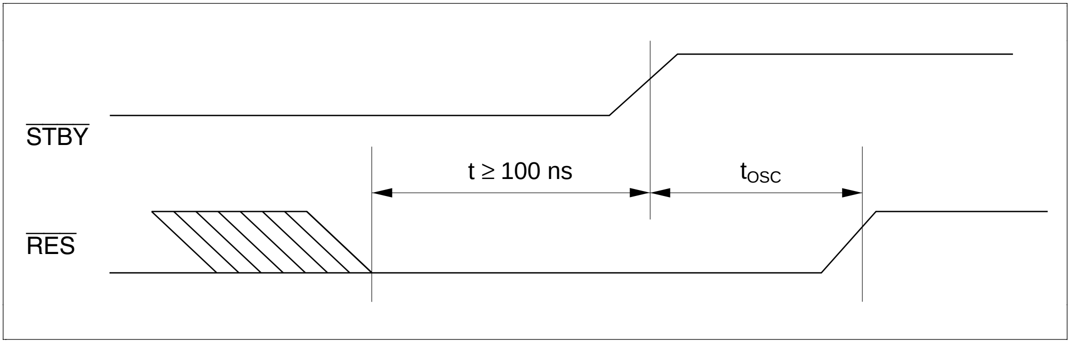
Timing of Transition to Hardware Standby Mode

(1) To retain RAM contents when the RAME bit in SYSCR is set to 1, drive the $\overline{\text{RES}}$ signal low 10 system clock cycles before the $\overline{\text{STBY}}$ signal goes low, as shown below. $\overline{\text{RES}}$ must remain low until $\overline{\text{STBY}}$ goes low (minimum delay from $\overline{\text{STBY}}$ low to $\overline{\text{RES}}$ high: 0 ns).



(2) When the RAME bit in SYSCR is cleared to 0 or when it is not necessary to retain RAM contents, $\overline{\text{RES}}$ does not have to be driven low as in (1).

Timing of Recovery From Hardware Standby Mode: Drive the $\overline{\text{RES}}$ signal low approximately 100 ns before $\overline{\text{STBY}}$ goes high.



Appendix F Product Code Lineup

Table F-1 Product Code Lineup

Product Type		Product Code	Mark Code	Package (Hitachi Package Code)
H8/3502	Mask ROM version	HD6433502P	HD6433502(***)P	64-pin shrink DIP (DP-64S)
		HD6433502F	HD6433502(***)F	64-pin QFP (FP-64A)

Note: (***) in mask versions is the ROM code.

Appendix G Package Dimensions

Figure G-1 shows the dimensions of the DP-64S package. Figure G-2 shows the dimensions of the FP-64A package.

Unit: mm

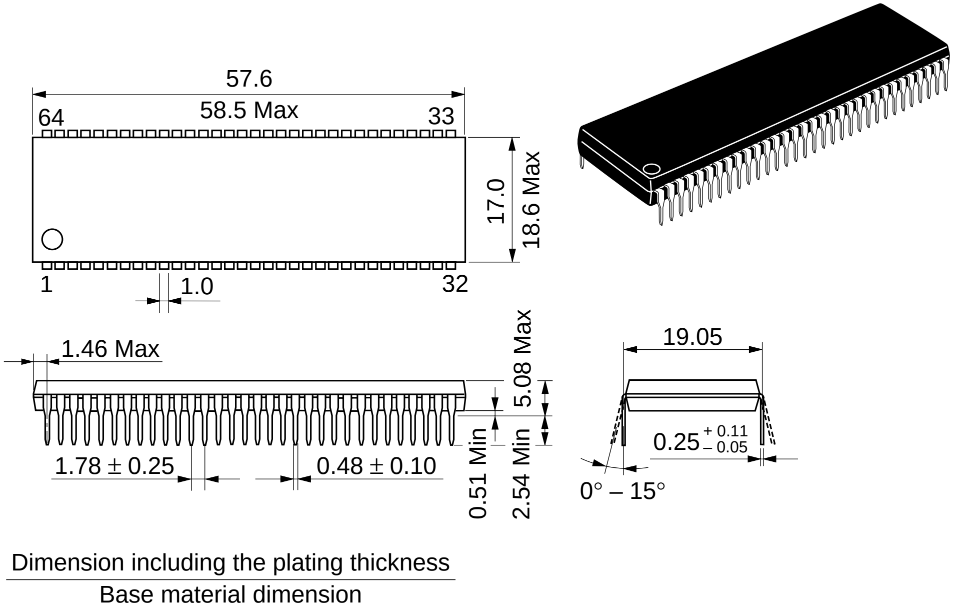


Figure G-1 Package Dimensions (DP-64S)

Unit: mm

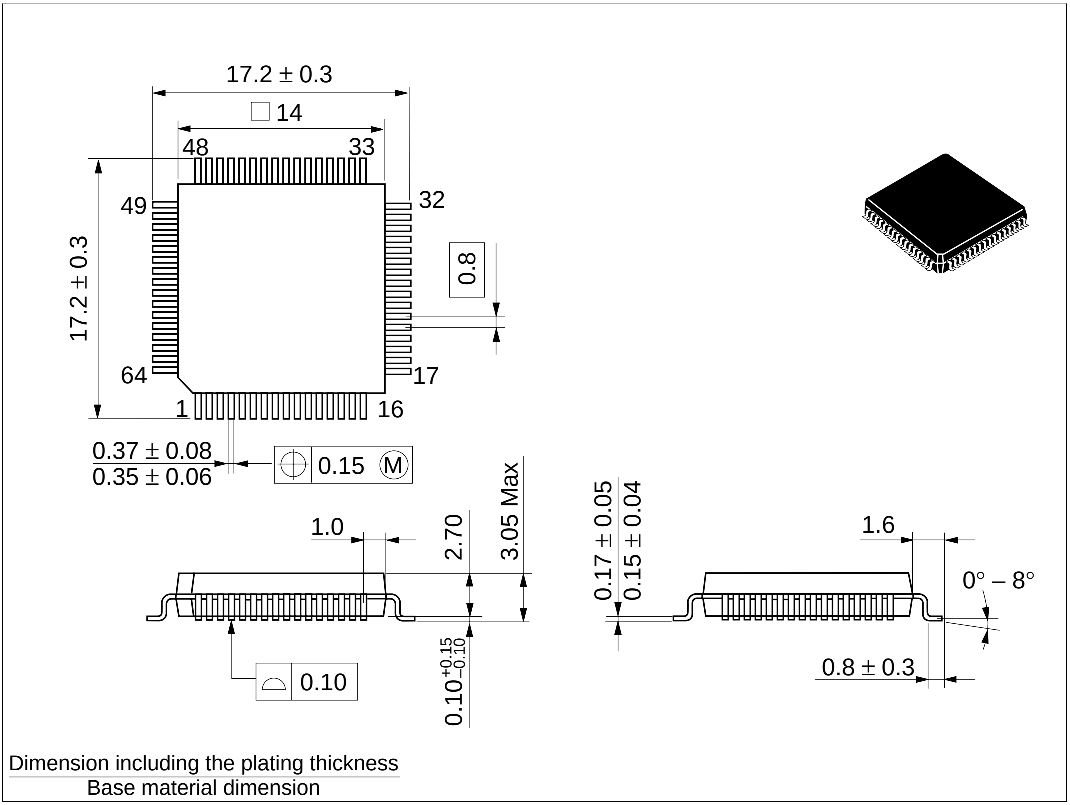


Figure G-2 Package Dimensions (FP-64A)

H8/3502 Series Hardware Manual

Publication Date: 1st Edition, September 1997
Published by: Semiconductor and IC Div.
Hitachi, Ltd.
Edited by: Technical Documentation Center
Hitachi Microcomputer System Ltd.