

FR60
32-BIT MICROCONTROLLER
MB91301 Series
HARDWARE MANUAL

FR60

32-BIT MICROCONTROLLER

MB91301 Series

HARDWARE MANUAL

Be sure to refer to the “Check Sheet” for the latest cautions on development.

“Check Sheet” is seen at the following support page

URL : <http://www.fujitsu.com/global/services/microelectronics/product/micom/support/index.html>

“Check Sheet” lists the minimal requirement items to be checked to prevent problems beforehand in system development.

PREFACE

■ Objectives and Intended Reader

Thank you for using Fujitsu semiconductor products.

The MB91301 series is a standard microcontroller that has a 32-bit high-performance RISC CPU as well as built-in I/O resources and bus control mechanisms for embedded controller that requires high-performance and high-speed CPU processing. Although the MB91301 series basically uses external bus access to support a vast address space accessed by a 32-bit CPU, it has a 4 Kbytes instruction cache memory and 4 Kbytes RAM (for data) to increase the speed at which the CPU executes instructions.

The MB91301 series is most suitable for embedded applications, such as digital video cameras, navigation systems, and DVD players, that require a high level of CPU processing power.

The MB91301 series is one of the FR60 of microcontrollers, which are based on the FR30/40 of CPUs. It has enhanced bus access and is optimized for high-speed use.

This manual is intended for engineers who will develop products using the MB91301 series and describes the functions and operations of the MB91301 series. Read this manual thoroughly.

For more information on instructions, see the "Instructions Manual".

Note: FR, which is an abbreviation of FUJITSU RISC controller, is a product of FUJITSU LIMITED.

■ Trademarks

The company names and brand names herein are the trademarks or registered trademarks of their respective owners.

■ License

Purchase of Fujitsu I²C components conveys a license under the Philips I²C Patent Rights to use, these components in an I²C system provided that the system conforms to the I²C Standard Specification as defined by Philips.

■ Structure of This Manual

This manual consists of the following 20 chapters and an appendix.

CHAPTER 1 OVERVIEW

This chapter provides basic information required to understand the MB91301 series, and covers features, a block diagram, and functions.

CHAPTER 2 HANDLING THE DEVICE

This chapter provides precautions on handling the MB91301 series.

CHAPTER 3 CPU AND CONTROL UNITS

This chapter provides basic information required to understand the functions of the MB91301 series. It covers architecture, specifications, and instructions.

CHAPTER 4 EXTERNAL BUS INTERFACE

The external bus interface controller controls the interfaces with the internal bus for chips and with external memory and I/O devices.

This chapter explains each function of the external bus interface and its operation.

CHAPTER 5 I/O PORT

This chapter describes the I/O ports and the configuration and functions of registers.

CHAPTER 6 16-BIT RELOAD TIMER

This chapter describes the 16-bit reload timer, the configuration and functions of registers, and 16-bit reload timer operation.

CHAPTER 7 PPG TIMER

This chapter describes the U-TIMER, the configuration and functions of registers, and U-TIMER operation.

CHAPTER 8 U-TIMER

This chapter describes the external interrupt and NMI controller, the configuration and functions of registers, and operation of the external interrupt and NMI controller.

CHAPTER 9 EXTERNAL INTERRUPT AND NMI CONTROLLER

This chapter describes the functions and operation of the delayed interrupt module.

CHAPTER 10 DELAYED INTERRUPT MODULE

This chapter describes the interrupt controller, the configuration and functions of registers, and interrupt controller operation. It also presents an example of using the hold request cancellation request function.

CHAPTER 11 INTERRUPT CONTROLLER

This chapter describes the A/D converter, the configuration and functions of registers, and A/D converter operation.

CHAPTER 12 A/D CONVERTER

This chapter describes the UART, the configuration and functions of registers, and UART operation.

CHAPTER 13 UART

This chapter describes the I²C interface, the configuration and functions of registers, and I²C interface operation.

CHAPTER 14 DMA CONTROLLER (DMAC)

This chapter describes the DMA controller (DMAC), the configuration and functions of registers, and DMAC operation.

CHAPTER 15 BIT SEARCH MODULE

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

CHAPTER 16 I²C INTERFACE

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

CHAPTER 17 16-BIT FREE RUN TIMER

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

CHAPTER 18 INPUT CAPTURE

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

CHAPTER 19 PROGRAM LOADER MODE (SUPPORTED ONLY BY THE MB91302A (IPL INTEGRATED MODEL))

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

CHAPTER 20 REAL-TIME OS EMBEDDED MB91302A-010 USER'S GUIDE

This chapter describes the bit search module, the configuration and functions of registers, and bit search module operation.

APPENDIX

This appendix consists of the following parts: I/O map, interrupt vector, pin states in the CPU state, notes on using a little endian area, and instruction lists. The appendix contains detailed information that could not be included in the main text and reference material for programming.

- The contents of this document are subject to change without notice.
Customers are advised to consult with FUJITSU sales representatives before ordering.
- The information, such as descriptions of function and application circuit examples, in this document are presented solely for the purpose of reference to show examples of operations and uses of Fujitsu semiconductor device; Fujitsu does not warrant proper operation of the device with respect to use based on such information. When you develop equipment incorporating the device based on such information, you must assume any responsibility arising out of such use of the information. Fujitsu assumes no liability for any damages whatsoever arising out of the use of the information.
- Any information in this document, including descriptions of function and schematic diagrams, shall not be construed as license of the use or exercise of any intellectual property right, such as patent right or copyright, or any other right of Fujitsu or any third party or does Fujitsu warrant non-infringement of any third-party's intellectual property right or other right by using such information. Fujitsu assumes no liability for any infringement of the intellectual property rights or other rights of third parties which would result from the use of information contained herein.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
Please note that Fujitsu will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.

How to Read This Manual

■ Terms Used in This Manual

The following defines principal terms used in this manual.

Term	Meaning
I-bus	32 bit bus for internal instructions. In the FR family, which is based on an internal Harvard architecture, independent buses are used for instructions and data. A bus converter is connected to the I-bus.
D-bus	Internal 32-bit data bus. An internal resource is connected to the D-bus.
F-bus	Internal instructions and data are multiplexed on a Princeton bus. The F-bus is connected to the I-bus and D-bus via a switch. The F-bus is connected to built-in resources such as ROM and RAM.
X-bus	External interface bus. The X-bus is connected to the external interface module. Data and instructions are multiplexed on an external bus.
R-bus	Internal 16-bit data bus. The R-bus is connected to the F-bus via an adapter. An I/O, clock generator, and interrupt controller are connected to the R-bus. Since addresses and data are multiplexed on an R-bus that is 16 bits wide, more than one cycle is required for the CPU to access these resources.
E-unit	Execution unit for operations.
CLKP	System clock. Clock generated by the clock generator for each of the internal resources connected to the R-bus. This clock has the same frequency as the source oscillation at its maximum, but becomes a 1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, ...1/16 (or 1/2, 1/4, 1/6, ...1/32) frequency clock as determined by the divide-by rate specified by the B3 to B0 bits in the clock generator DIVR0 register.
CLKB	System clock. Operating clock for the CPU and each of the other resources connected to a bus other than the R-bus and X-bus. This clock has the same frequency as the source oscillation at its maximum, but becomes a 1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, ..., 1/16 (or 1/2, 1/4, 1/6, ..., 1/32) frequency clock as determined by the divided-by rate specified by the P3 to P0 bits in the clock generator DIVR0 register.
CLKT	System clock. Operating clock for the external resources connected to the X-bus. This clock has the same frequency as the source oscillation at its maximum, but becomes a 1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, ..., 1/16 (or 1/2, 1/4, 1/6, ..., 1/32) frequency clock as determined by the divided-by rate specified by the T3 to T0 bits in the clock generator DIVR1 register.

CONTENTS

CHAPTER 1 OVERVIEW	1
1.1 Features of the MB91301 Series	2
1.2 Block Diagram	7
1.3 Package Dimensions	8
1.4 Pin Layout	11
1.5 Pin No. Table	13
1.6 List of Pin Functions	15
1.7 I/O Circuit Types	26
CHAPTER 2 HANDLING THE DEVICE	31
2.1 Precautions on Handling the Device	32
2.2 Precautions on Handling Power Supplies	39
CHAPTER 3 CPU AND CONTROL UNITS	41
3.1 Memory Space	42
3.2 Internal Architecture	45
3.3 Instruction Cache	50
3.3.1 Configuration of the Instruction Cache	51
3.3.2 Configuration of the Control Registers	54
3.3.3 Instruction Cache Statuses and Settings	58
3.3.4 Setting up the Instruction Cache before Use	60
3.4 Dedicated Registers	63
3.4.1 Program Status (PS) Register	66
3.5 General-Purpose Registers	70
3.6 Data Structure	71
3.7 Word Alignment	72
3.8 Memory Map	73
3.9 Branch Instructions	74
3.9.1 Operation of Branch Instructions with Delay Slot	75
3.9.2 Operation of Branch Instruction without Delay Slot	78
3.10 EIT (Exception, Interrupt, and Trap)	79
3.10.1 EIT Interrupt Levels	80
3.10.2 Interrupt Control Register (ICR)	82
3.10.3 System Stack Pointer (SSP)	83
3.10.4 Table Base Register (TBR)	84
3.10.5 Multiple EIT Processing	88
3.10.6 EIT Operations	90
3.11 Reset (Device Initialization)	94
3.11.1 Reset Levels	95
3.11.2 Reset Sources	96
3.11.3 Reset Sequence	98
3.11.4 Oscillation Stabilization Wait Time	99
3.11.5 Reset Operation Modes	101

3.12	Clock Generation Control	103
3.12.1	PLL Controls	104
3.12.2	Oscillation Stabilization Wait Time and PLL Lock Wait Time	105
3.12.3	Clock Distribution	106
3.12.4	Clock Division	108
3.12.5	Block Diagram of Clock Generation Controller	109
3.12.6	Register of Clock Generation Controller	110
3.12.7	Peripheral Circuits of Clock Controller	125
3.12.8	Smooth Startup and Stop of Clock	128
3.13	Device State Control	131
3.13.1	Device States and State Transitions	133
3.13.2	Low-power Modes	136
3.14	Operating Modes	140
CHAPTER 4	EXTERNAL BUS INTERFACE	143
4.1	Overview of the External Bus Interface	144
4.2	External Bus Interface Registers	149
4.2.1	Area Select Registers 0 to 7(ASR0 to ASR7)	150
4.2.2	Area Configuration Registers 0 to 7 (ACR0 to ACR7)	152
4.2.3	Area Wait Register (AWR0 to AWR7)	158
4.2.4	Memory setting register (MCRA for SDRAM/FCRAM auto-precharge OFF mode)	167
4.2.5	Memory setting register (MCRB for FCRAM auto-precharge ON mode)	169
4.2.6	I/O Wait Registers for DMAC (IOWR0, IOWR1)	170
4.2.7	Chip Select Enable Register (CSER)	172
4.2.8	Cache Enable Register (CHER)	174
4.2.9	Pin/Timing Control Register (TCR)	175
4.2.10	Refresh Control Register (RCR)	177
4.3	Setting Example of the Chip Select Area	181
4.4	Endian and Bus Access	182
4.4.1	Big Endian Bus Access	184
4.4.2	Little Endian Bus Access	191
4.4.3	Comparison of Big Endian and Little Endian External Access	196
4.5	Operation of the Ordinary Bus Interface	202
4.5.1	Basic Timing	203
4.5.2	Operation of \overline{WRn} + Byte Control Type	204
4.5.3	Read -> Write Operation	206
4.5.4	Write -> Write Operation	207
4.5.5	Auto-Wait Cycle	208
4.5.6	External Wait Cycle	209
4.5.7	Synchronous Write Enable Output	210
4.5.8	\overline{CSn} Delay Setting	212
4.5.9	$\overline{CSn} \rightarrow \overline{RD}/\overline{WRn}$ Setup and $\overline{RD}/\overline{WRn} \rightarrow \overline{CSn}$ Hold Setting	213
4.5.10	DMA Fly-By Transfer (I/O -> Memory)	214
4.5.11	DMA Fly-By Transfer (Memory -> I/O)	215
4.6	Burst Access Operation	216
4.7	Address/data Multiplex Interface	218
4.8	Prefetch Operation	221

4.9	SDRAM/FCRAM Interface Operation	224
4.9.1	Self Refresh	228
4.9.2	Power-on Sequence	229
4.9.3	Connecting SDRAM/FCRAM to Many Areas	230
4.9.4	Address Multiplexing Format	231
4.9.5	Memory Connection Example	232
4.10	DMA Access Operation	236
4.10.1	DMA Fly-By Transfer (I/O -> Memory)	237
4.10.2	DMA Fly-By Transfer (Memory -> I/O)	239
4.10.3	DMA Fly-By Transfer (I/O -> SDRAM/FCRAM)	241
4.10.4	DMA Fly-By Transfer (SDRAM/FCRAM -> I/O)	243
4.10.5	2-Cycle Transfer (Internal RAM -> External I/O, RAM)	247
4.10.6	2-Cycle Transfer (External -> I/O)	248
4.10.7	2-Cycle Transfer (I/O -> External)	249
4.10.8	2-Cycle Transfer (I/O -> SDRAM/FCRAM)	250
4.10.9	2-Cycle Transfer (SDRAM/FCRAM -> I/O)	251
4.11	Bus Arbitration	252
4.12	Procedure for Setting a Register	254
4.13	Notes on Using the External Bus Interface	255
CHAPTER 5	I/O PORT	257
5.1	Overview of the I/O Port	258
5.2	I/O Port Registers	260
CHAPTER 6	16-BIT RELOAD TIMER	269
6.1	Overview of the 16-bit Reload Timer	270
6.2	16-bit Reload Timer Registers	271
6.2.1	Control Status Register (TMCSR)	272
6.2.2	16-bit Timer Register (TMR:TMR2 to TMR0)	275
6.2.3	16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0)	276
6.3	16-bit Reload Timer Operation	277
6.4	Operating States of the Counter	279
6.5	Precautions on Using the 16-bit Reload Timer	280
CHAPTER 7	PPG TIMER	281
7.1	Overview of PPG Timer	282
7.2	Block Diagram of PPG Timer	283
7.3	Registers of PPG Timer	285
7.3.1	Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0)	286
7.3.2	PPG Cycle Set Register (PCSR:PCSR3 to PCSR0)	290
7.3.3	PPG Duty Set Register (PDUT:PDUT3 to PDUT0)	291
7.3.4	PPG Timer Register (PTMR:PTMR3 to PTMR0)	292
7.3.5	General Control Register 10 (GCN10)	293
7.3.6	General Control Register 20 (GCN20)	296
7.4	PPG Operation	297
7.5	One-shot Operation	299
7.6	PPG Timer Interrupt Source and Timing Chart	301

7.7	Activating Multiple Channels by Using the General Control Register	303
7.8	Notes on Use of the PPG Timer	305
CHAPTER 8 U-TIMER		307
8.1	Overview of the U-TIMER	308
8.2	U-TIMER Registers	309
8.3	U-TIMER Operation	313
CHAPTER 9 EXTERNAL INTERRUPT AND NMI CONTROLLER		315
9.1	Overview of the External Interrupt and NMI Controller	316
9.2	External Interrupt and NMI Controller Registers	317
9.2.1	Interrupt Enable Register (ENIR)	318
9.2.2	External Interrupt Request Register (EIRR)	319
9.2.3	External Interrupt Request Level Setting Register (ELVR)	320
9.3	Operation of the External Interrupt and NMI Controller	321
CHAPTER 10 DELAYED INTERRUPT MODULE		325
10.1	Overview of the Delayed Interrupt Module	326
10.2	Delayed Interrupt Module Registers	327
10.3	Operation of the Delayed Interrupt Module	328
CHAPTER 11 INTERRUPT CONTROLLER		329
11.1	Overview of the Interrupt Controller	330
11.2	Interrupt Controller Registers	332
11.2.1	Interrupt Control Register (ICR)	334
11.2.2	Hold Request Cancellation Request Level Setting Register (HRCL)	336
11.3	Interrupt Controller Operation	337
11.4	Example of Using the Hold Request Cancellation Request Function (HRCR)	343
CHAPTER 12 A/D CONVERTER		345
12.1	Overview of the A/D Converter	346
12.2	A/D Converter Registers	348
12.2.1	Control Status Register (ADCS)	349
12.2.2	Data Register (ADCR)	354
12.2.3	Conversion result register (ADCR0 to ADCR3)	355
12.3	A/D Converter Operation	356
12.4	Precautions on the Using A/D Converter	358
CHAPTER 13 UART		359
13.1	Overview of the UART	360
13.2	UART Registers	362
13.2.1	Serial Mode Register (SMR)	363
13.2.2	Serial Control Register (SCR)	365
13.2.3	Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)	368
13.2.4	Serial Status Register (SSR)	369
13.3	UART Operation	373
13.3.1	Asynchronous (Start-stop Synchronization) Mode	374

13.3.2	CLK Synchronous Mode	375
13.3.3	Occurrence of Interrupts and Timing for Setting Flags	377
13.4	Example of Using the UART	380
13.5	Example of Setting Baud Rates and U-TIMER Reload Values	382
CHAPTER 14 DMA CONTROLLER (DMAC)	383	
14.1	Overview of the DMA Controller (DMAC)	384
14.2	DMA Controller (DMAC) Registers	386
14.2.1	Control/Status Registers A (DMACA0 to DMACA4)	388
14.2.2	Control/Status Registers B (DMACB0 to DMACB4)	393
14.2.3	Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)	400
14.2.4	DMAC All-Channel Control Register (DMACR)	402
14.2.5	Other Functions	404
14.3	DMA Controller (DMAC) Operation	405
14.3.1	Setting a Transfer Request	408
14.3.2	Transfer Sequence	410
14.3.3	General Aspects of DMA Transfer	414
14.3.4	Addressing Mode	416
14.3.5	Data Types	417
14.3.6	Transfer Count Control	418
14.3.7	CPU Control	419
14.3.8	Hold Arbitration	420
14.3.9	Operation from Starting to End/Stopping	421
14.3.10	DMAC Interrupt Control	425
14.3.11	Channel Selection and Control	426
14.3.12	Supplement on External Pin and Internal Operation Timing	428
14.4	Operation Flowcharts	432
14.5	Data Bus	435
14.6	DMA External Interface	438
14.6.1	Input Timing of the DREQx Pin	439
14.6.2	FR30 Compatible Mode of DACK	441
CHAPTER 15 BIT SEARCH MODULE	443	
15.1	Overview of the Bit Search Module	444
15.2	Bit Search Module Registers	445
15.3	Bit Search Module Operation	447
CHAPTER 16 I²C INTERFACE	449	
16.1	Overview of the I ² C Interface	450
16.2	I ² C Interface Registers	451
16.3	Block Diagram of I ² C Interface	453
16.4	Detailed on Registers of the I ² C Interface	454
16.5	I ² C Interface Operation	468
16.6	Operation Flowcharts	473

CHAPTER 17 16-BIT FREE RUN TIMER	477
17.1 Overview of 16-bit Free Run Timer	478
17.2 Registers of the 16-bit Free Run Timer	479
17.3 Block Diagram of the 16-bit Free Run Timer	480
17.4 Details on Registers of the 16-bit Free Run Timer	481
17.5 Operation of the 16-bit Free Run Timer	485
17.6 Precautions on Using the 16-bit Free Run Timer	487
CHAPTER 18 INPUT CAPTURE	489
18.1 Overview of Input Capture	490
18.2 Input Capture Registers	491
18.3 Block Diagram of Input Capture	492
18.4 Details on Registers of Input Capture	493
18.5 Operation of Input Capture	495
CHAPTER 19 PROGRAM LOADER MODE (SUPPORTED ONLY BY THE MB91302A (IPL INTEGRATED MODEL))	497
19.1 Overview of the Program Loader Mode	498
19.2 Setting the Program Loader	499
19.3 Operations in the Program Loader Mode	500
19.4 Example of Using the Program Loader Mode to Write to Flash Memory	511
CHAPTER 20 REAL-TIME OS EMBEDDED MB91302A-010 USER'S GUIDE	515
20.1 Overview	516
20.2 Memory Map	517
20.3 Specifications for REALOS/FR Embedded in MB91302A-010	518
20.4 Section Allocation	520
20.5 Startup Routine	521
20.6 Initial Settings for SOFTUNE Workbench and REALOS/FR	522
20.7 Mode Pins, Mode Vectors, and Reset Vectors	530
20.8 Chip Evaluation System	534
APPENDIX	535
APPENDIX A I/O MAP	536
APPENDIX B INTERRUPT VECTOR	548
APPENDIX C PIN STATE IN EACH CPU STATE	552
APPENDIX D NOTES ON USING A LITTLE ENDIAN AREA	564
D.1 C Compiler (fcc911)	565
D.2 Assembler (fasm911)	568
D.3 Linker (flnk911)	570
D.4 Debugger (sim911, eml911, mon911)	571
APPENDIX E INSTRUCTION LISTS	572
E.1 How to Read the Instruction Lists	573
E.2 FR Family Instruction Lists	578
INDEX.....	597

Main changes in this edition

Page	Changes (For details, refer to main body.)
-	Part number is deleted. (MB91301, MB91V301)
2	Table is changed in ■ Features of the MB91301 Series. (MB91301 is changed.)
6	MB91301 and MB91V301 are deleted in ■ Product Line-up is changed.
7	Figure 1.2-1 Block Diagram is changed.
8	Summary is changed in 1.3 Package Dimensions. (The MB91301 series is available in one type of package. → The MB91301 series is available in three types of package.)
12	Figure 1.4-2 Pin Layout of the MB91302A is changed. (FPT-144P-M08 is added.)
13	Pin Name is changed in Table 1.5-1 MB91V301A Pin No. Table (Package: PGA-179C-A03) (1 / 2). No.49(LABA → LBA)
17	Pin name and Function are changed in the Pin No. 42 of Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (3 / 10). (SWR → SWE)
32	■ Quartz Oscillation Circuit is changed. (Added the description; Please ask the crystal maker to evaluate the oscillational characteristics of the crystal and this device.)
33	○ Notes on during operation of PLL clock mode is changed.
34	○ Low-power consumption mode is changed. (Deleted the description; In addition, please set I flag, ILM, and ICR to diverge to the interruption handler that is the return factor after the standby returns.) (1) is changed in ○ Low-power consumption mode.
43	Figure 3.1-1 Memory Map is changed. (Figure is renewed.)
60	○ Initialization is changed. (ldi #0x000003C7, r0 → ldi #0x000003E7, r0)
85, 87	Table 3.10-4 Vector Table is changed. (Instruction break exception → Reserved for system) (Operand break trap → Reserved for system) (Deleted the description; * : Reserved for system (MB91301, MB91V301))
97	■ Watchdog Reset is changed. (Unless A5H/5AH is written to the watchdog reset postpone register (WPR) → Unless A5H/5AH is written to the timebase counter clear register (CTBR))

Page	Changes (For details, refer to main body.)
103	<p>■ Clock Generation Control is changed. (The followings sentences are added. The following section describes the generation and control of each clock. For detailed information about the registers and flags described below, see "3.12.5 Block Diagram of Clock Generation Controller" and "3.12.6 Register of Clock Generation Controller". The symbol "ϕ", which is shown in Table 3.12-4, Table 3.12-9, Table 3.12-13, Table 3.12-16, Table 3.12-20, Table 3.12-21 and Table 3.12-22, indicates the basic clock that is achieved by dividing the source clock by two or performing PLL oscillation. Therefore, the system base clock is the clock which is generated where the above base clock is generated.)</p>
109	Figure 3.12-1 Block Diagram of Clock Generation Controller is changed. ([Watchdog controller] is changed.)
112	Note is added.
114	Table 3.12-9 Oscillation Stabilization Wait Settings is changed. (CS1 → OS1) (CS0 → OS0)
116	[bit9] SYNCR (SYNChronous Reset enable) is changed.
117	<p>■ Time Base Counter Clear Register (CTBR) is changed. (The followings sentences are added. The bits are automatically cleared when the CPU is not in operation in such cases as stop/sleep mode and DMA transfer. Therefore, if the above conditions occur, the watchdog reset is delayed automatically. However, it will not be delayed, if a request to hold an external bus (BRQ) has been accepted. For this reason, select the sleep mode before entering a hold request (BRQ), when intending to hold the external bus for a long period.)</p>
(119)	■ Watchdog Reset Postpone Register (WPR) is deleted.
125	[Postponing a watchdog reset] is changed. (Once the watchdog timer is started, the program must write {A5H} and {5AH} in this order to the watchdog reset postpone register (WPR). → Once the watchdog timer is started, the program must write "A5H" and "5AH" in this order to the time base counter clear register (CTBR).)
136	[Sleep mode transition] is added.
137	○ Synchronous standby operations is changed. (○ Normal and synchronous standby operations → ○ Synchronous standby operations)
138	<p>■ Stop Mode is changed. ([Stop mode transition] is added.)</p>
141	Table 3.14-1 Mode Settings is changed. (The description in * is deleted.)
142	<p>Table 3.14-2 Function of internal ROM enable is changed. (The description in *1 is deleted.) Table 3.14-3 Settings of the Initial Bus Width is changed. (Only for MB91302A and MB91V301A is deleted.)</p>
151	<p>■ Functions of Bits in the Area Select Registers (ASR0 to ASR7) is changed. (The area select registers (ASR0 to ASR7) specify the start address of each chip select area ($\overline{CS0}$ to $\overline{CS7}$). is added.) (ASR0 to 7 registers. → ACR0 to ACR7 registers.)</p>
153	Figure 4.2-2 Configuration of Area Configuration Registers 0 to 7 (ACR0 to ACR7) is changed. (Note is added.)
181	○ Example of setting ASRs and ASZ3 to ASZ0 is changed. (10000000H → 0FFFFFFFH) (00200000H → 001FFFFFH)

Page	Changes (For details, refer to main body.)
205	<ul style="list-style-type: none"> ■ Operation Timing of the \overline{WRn} + Byte Control Type is changed. (• For write access, data output to D31-16 starts at the timing at which \overline{WRn} is asserted. → • For write access, data output to D31 to D00 starts at the timing at which $\overline{WR0}$ is asserted.)
254	<ul style="list-style-type: none"> ■ Procedure for Setting a Register is changed. (Added the description; Set ACR after setting ASR if ASR and ACR are accessed by halfword.)
260	<ul style="list-style-type: none"> ■ Configuration of the Port Data Registers (PDR) is changed. (Note:MB91301 and MB91V301 do not have PFR61 register. is deleted.)
262	<p>Figure 5.2-3 Configuration of the Pull-up Resistor Control Registers (PCR) is changed. (PCRG and PCRJ are deleted.)</p> <ul style="list-style-type: none"> ■ Configuration of the Pull-up Resistor Control Registers (PCR) is changed. (PCR0-PCR2, PCR6, PCR8-PCRB, PCRG, PCRH and PCRJ control the pull-up resistor of the corresponding I/O port. → PCR0 to PCR2, PCR6, PCR8 to PCRB, and PCRH control the pull-up resistor of the corresponding I/O port.) (Note:MB91302A and MB91V301A do not have PCRG register and PCRJ register. is deleted.)
270	<p>Figure 6.1-1 Block Diagram of the 16-bit Reload Timer is changed. (16-bit down counter(TMR) UF → 16-bit timer register (TMR) UF)</p>
309	Note is added.
313	Note is added.
321	<ul style="list-style-type: none"> ■ Operating Procedure for an External Interrupt is changed. (Added the description; 1) Set the general-purpose I/O port served dual use as the pin for the external interrupt input to input port.)
322	<ul style="list-style-type: none"> ■ External Interrupt Request Level is changed. (Added the description; If the request input level is a level setting, the pulse width must be at least 3 machine cycles. Also, as long as the interrupt input pin retains the active level, the interrupt request is continuously made to the interrupt controller, even if the source register is cleared.)
323	<ul style="list-style-type: none"> ■ Notes on Returning from STOP State Using External Interrupt is added.
324	<ul style="list-style-type: none"> ■ Return Operation from STOP State is added.
352	Notes is changed in the [bit7, bit6] MD1, MD0 (A/D converter MoDe set).
353	<p>Notes are changed in the [bit2, bit1, bit0]ANE2, ANE1, ANE0 (ANalog End channel set) Setting of A/D conversion end channel. (The followings sentences are added. After setting the start channel to the A/D conversion start channel selection bit (ANS2, ANS1, ANS0), please set neither the A/D conversion mode selection bit (MD1, MD0) nor the A/D conversion end channel selection bit (ANE2, ANE1, ANE0) by the read-modify-write type instruction. The last conversion channel is read from the ANS2, ANS1, ANS0 bits until the A/D conversion operating starts. Therefore, when the MD1, MD0 bits and ANE2, ANE1, ANE0 bits are set by the read-modify-write type instruction after setting the start channel to the ANS2, ANS1, ANS0 bits, the value of ANE2, ANE1, ANE0 bits may be re-written.)</p>
356	<ul style="list-style-type: none"> ■ Single-shot Conversion Mode is changed. (Note is added.)
358	<ul style="list-style-type: none"> ■ Precautions on Using the A/D Converter is changed. (If STS1 and STS0 are set, set $\overline{ATG}=1$ input and reload timer (channel 2)=0 output. → If STS1 and STS0 are set, set $\overline{ATG}=1$ input and reload timer (channel 1)=0 output.) (○ Restart of the A/D conversion is added.)
360	<ul style="list-style-type: none"> ■ Features of the UART is changed. (Deleted the description; • The DMAC interrupt source is cleared if the DRCL register is written to.)

Page	Changes (For details, refer to main body.)
362	"Figure 13.2-1 Configuration of UART Registers" and "Figure 13.2-2 UART Registers" are changed. (DRCL is deleted.)
(372)	13.2.5 DRCL Register is deleted.
385	Figure 14.1-1 Block Diagram of the DMA Controller (DMAC) is changed. (DSAD 2-stage register → DMASA 2-stage register) (DDAD 2-stage register → DMADA 2-stage register)
386	Figure 14.2-1 DMA Controller (DMAC) Registers is changed. (All-channel control register (DMACR) → DMAC all-channel control register (DMACR))
388	[bit31] DENB (Dma ENaBle): DMA operation enable bit is changed. (PUAS bit → PAUS bit)
391	Notes is changed in the [bit28 to bit24] IS4 to IS0 (Input Select): Transfer source selection.
394	[bit25] SADM (Source-ADdr. count-Mode select) : Transfer source address count mode specification is changed. (make the address count width (SAAZ and DASZ) equal to 0. → make the address count width (SASZ and DASZ) equal to "0".)
411	■ Demand Transfer 2-Cycle Transfer is changed. (Select the level with IS[3:0] of DMACA. → Select the level with IS[4:0] of DMACA.)
428 to 430	■ Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped is changed.
453	Figure 16.3-1 Block Diagram of the I ² C Interface is changed. (FNSB → ENSB)
457	[bit12] MSS (Master Slave Select) is changed. (Note is added.)
468	○ START Condition is changed. (not (IBCR MSS=1) is deleted.) Notes is changed. ((DBL=1:IDAR or EN=0:ICCR) → (DBL=1: IDBL or EN=0: ICCR))
503	○ Main program flowchart is changed. (* About setting UART0 → * About setting UART1)
511	19.4 Example of Using the Program Loader Mode to Write to Flash Memory is changed. (■ Allocation of Flash Memory is added.)
537, 538, 543	Register DRCL0, DRCL1, DRCL2, WPR in Table A-1 I/O Map is renewed.
547	Table A-1 I/O Map is changed. (The followings sentence is added. *2: No register in 000420 _H and 000423 _H in MB91302A and MB91V301A.) (The followings sentence is added. *4: Reserved register. Access is disabled.)
548	Table B-1 Interrupt Vectors (1 / 4) is changed. (Instruction break exception → Reserved for system) (Operand break trap → Reserved for system)
553 to 555	Table C-1 Pin States in External Bus 32-Bit Mode is changed.

The vertical lines marked in the left side of the page show the changes.

CHAPTER 1 OVERVIEW

This chapter provides basic information required to understand the MB91301 series, and covers features, a block diagram, and functions.

- 1.1 Features of the MB91301 Series
- 1.2 Block Diagram
- 1.3 Package Dimensions
- 1.4 Pin Layout
- 1.5 Pin No. Table
- 1.6 List of Pin Functions
- 1.7 I/O Circuit Types

1.1 Features of the MB91301 Series

The MB91301 series is a standard single-chip microcontroller that has a 32-bit high-performance RISC CPU (FR family) as well as built-in I/O resources and bus control mechanisms for embedded controller requiring high-performance and high-speed CPU processing. Although the MB91301 series basically uses external bus access to support a vast address space accessed by a 32-bit CPU, it has a 4-Kbyte instruction cache memory and 4-Kbyte RAM to increase the speed at which the CPU executes instructions.

This model is an FR60 model that is based on the FR30/40 of CPUs. It has enhanced bus access and is optimized for high-speed use.

The MB91301 series is most suitable for embedded applications, such as digital video cameras, navigation systems, and DVD players, that require a high level of CPU processing power.

■ Features of the MB91301 Series

The MB91301 series has the line-up of series embedded the each of program in built-in ROM.

ROM variation Products	Real time OS internal ROM version	IPL (internal program loader) internal version	User ROM version	No ROM version
MB91302A	O	O	O	O

■ FR CPU

- 32-bit RISC, load/store architecture, five pipelines
- Operating frequency of 68 MHz (Internal maximum value), 68 MHz (External maximum value) [PLL used, original oscillation at 17 MHz]
- 32-bit general-purpose register x 16
- 16-bit fixed-length instructions (basic instructions), one instruction per cycle
- Memory-to-memory transfer, bit processing, instructions, including barrel shift, etc.; instructions appropriate for embedded applications
- Function entry and exit instructions, multi load/store instructions--instructions compatible with high-level languages
- Register interlock function to facilitate assembly-language coding
- Branch instruction with a delay slot allowing a decrease in overhead for branch processing
- Built-in multiplier/instruction-level support
 - Signed 32-bit multiplication: 5 cycles
 - Signed 16-bit multiplication: 3 cycles
- Interrupts (saving of PC and PS): 6 cycles, 16 priority levels

■ Bus Interface

- Maximum operating frequency of 68 MHz (at using SDRAM)
- 24-bit address can be fully output (16-Mbyte space)
- 8-bit, 16-bit and 32-bit data I/O
- Prefetch buffer installed
- Unused data and address pins can be used as general-purpose I/O ports.
- Totally independent 8-area chip select output that can be defined at a minimum of 64K bytes
- Support of interfaces for various memory modules
 - Asynchronous SRAM, asynchronous ROM/FLASH
 - Page-mode ROM/FLASHROM (a page-size of 1, 2, 4, or 8 can be selected)
 - Burst-mode ROM/FLASH (MBM29BL160D/161D/162D, etc.)
 - SDRAM (or FCRAM type, CAS Latency1 to 8, 2/4 bank product)
 - Address/data multiplexed bus (8-bit/16-bit width only)
- Basic bus cycle: 2 cycles
- Automatic wait cycle generator (Max 15 cycles) that can be programmed for each area and can insert waits
- External wait cycles due to RDY input
- Endian setting of byte ordering (big/little) $\overline{CS0}$ are, however, is only big endian
- Write disable setting (read only data)
- Enable/disable set of capturing to the built-in cache
- Enable/disable set of prefetch function
- Supports fly-by DMA transfer that enables independent I/O wait control
- External bus arbitration using BRQ and \overline{BGRNT} is enabled

■ Built-in Memory

- DATA RAM: 4K bytes
- ROM: 4K bytes (MB91302A)

Built-in 8-Kbyte DATA RAM, 8-Kbyte DATA/instruction RAM and 8-Kbyte emulation RAM in MB91V301A

■ Instruction Cache

- Capacity of 4K bytes
- 2 way set associative
- 128 block/way, 4 entry (4 words)/block
- Lock function allows specific program codes to stay resident in cache
- Instruction RAM function: A part of the instruction cache not in use can be used as RAM

■ DMAC (DMA Controller)

- 5 channels (2 channels for external to request)
- 3 transfer sources (external pins, internal peripherals, software)

CHAPTER 1 OVERVIEW

- Internal peripheral can be selected at each channel as the transfer factor
- Addressing mode with 32-bit full address specifications (increase, decrease, fixed)
- Transfer modes (demand transfer, burst transfer, step transfer, block transfer)
- Fly-by transfer supported (three channels between external I/O and external memory)
- Transfer data size that can be selected from 8, 16, and 32 bits

■ Bit Search Module

Searches for the position of the first bit varying between 1 and 0 in the MSB of a word

■ Reload Timer

- 16-bit timer: 3 channels
- Internal clock: 2-clock cycle resolution, selectable from 2, 8 or 32 divided frequency

■ UART

- UART full-duplex double buffer
- Independent 3 channels
- Data length: 7 to 9 bits (no parity), 8 to 8 bits (parity)
- Either asynchronous (start-stop synchronization) or CLK synchronous communication can be selected.
- Multi processor mode
- Built-in 16-bit timer (U-TIMER) as baud rate generator: generating arbitrary baud rates
- An external clock can be used as the transfer clock.
- Error detection functions (parity, frame, overrun)

■ Interrupt Controller

- Total of 9 external interrupts (one unmaskable pin (NMI) and eight regular interrupt pins (INT7 to INT0))
- Internal interrupt source: UART, DMAC, A/D, UTIMER, delay interrupt, I²C, free run timer and ICU
- The I²C, free run timer, and ICU are sources unique to the MB91302A and MB91V301A.
- Priority level can be defined as programmable (16 levels) except for the unmaskable pin.

■ A/D Converter (Sequential Conversion Type)

- 10-bit resolution, 4 channels
- Sequential comparison and conversion type: peripheral clock (CLKP) 140 clock cycle conversion time (about 4.1 µs/ch at 34MHz operating)
- Built-in sample and hold circuit
- Conversion modes (single-shot conversion mode, scan conversion mode, and repeat conversion mode)
- Causes of startup (select from software, external triggers, and internal timer)

■ I²C Bus Interface

- 2-channel master/slave transmission and reception of I²C bus interface
- Arbitration function and clock synchronization function of I²C bus interface

■ Free Run Timer

- 16-bit 1channel
- Input capture 4 channels

■ Other Interval Timers

- 16-bit timer: 3 channels (U-TIMER)
- PPG timer: 4 channels
- Watchdog timer: 1 channel

■ Other Features

- Reset factor: Watchdog timer/software reset/external reset (\overline{INIT} pin)
- Low-power consumption mode: sleep/stop mode
- Clock control
 - Allows arbitrary different operating clock frequencies to be set for the CPU and peripherals. The gear clock factor can be selected from among 16 options: 1/1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8, ..., 1/16. Also, PLL multiplication can be selected. Note that the maximum operating frequency of peripherals is 34 MHz.
- Packages: MB91302A FPT-144P-M12, FPT-144P-M08,
MB91V301A PGA-179C-A03
- CMOS technology
 - 0.25 μ m
- Power voltages
 - Power supply (analog power supply): 3.3 V \pm 0.3 V (at using internal regulator)
- On chip Device Support Unit (DSU4) is installed in MB91V301A.

CHAPTER 1 OVERVIEW

■ Product Line-up

	MB91302A	MB91V301A
Type	Mask ROM product (for volume production)	Evaluation version (For evaluation and development)
RAM	4K bytes (only for data)	16K bytes (data 8 Kbyte+8 Kbyte)
ROM	4K bytes ROM has non-ROM model, the optimal real time OS internal model ^{*1} , and the IPL (Internal Program Loader) internal model ^{*2} by adding the user ROM model.	8K bytes (RAM)
DSU	-	DSU4
Package	LQFP-144 (0.4 mm pitch, 0.5 mm pitch)	PGA-179
Other	Currently in production	Currently available

*1: The Fujitsu product of real time OS REALOS/FR by conforming to the µITRON 3.0 is stored and optimized with the MB91302A. For details of built-in service call type and the specification of user task, see "CHAPTER 20 REAL-TIME OS EMBEDDED MB91302A-010 USER'S GUIDE" and following manuals;

- FR FAMILY CONFORMING to µITRON3.0 SPECIFICATIONS SOFTUNE REALOS/ FR USER'S GUIDE
- FR FAMILY CONFORMING to µITRON3.0 SPECIFICATIONS SOFTUNE REALOS/ FR KERNEL MANUAL
- FR/ F²MC FAMILY CONFORMING to µITRON SPECIFICATIONS SOFTUNE REALOS/ FR/907/896 CONFIGURATOR MANUAL
- FR-V/ FR/ F²MC FAMILY CONFORMING to µITRON3.0 SPECIFICATIONS SOFTUNE REALOS ANALYZER MANUAL

*2: The ROM stores the IPL (Internal Program Loader). Loading various programs can be executed from the external system by the internal UART/SIO.

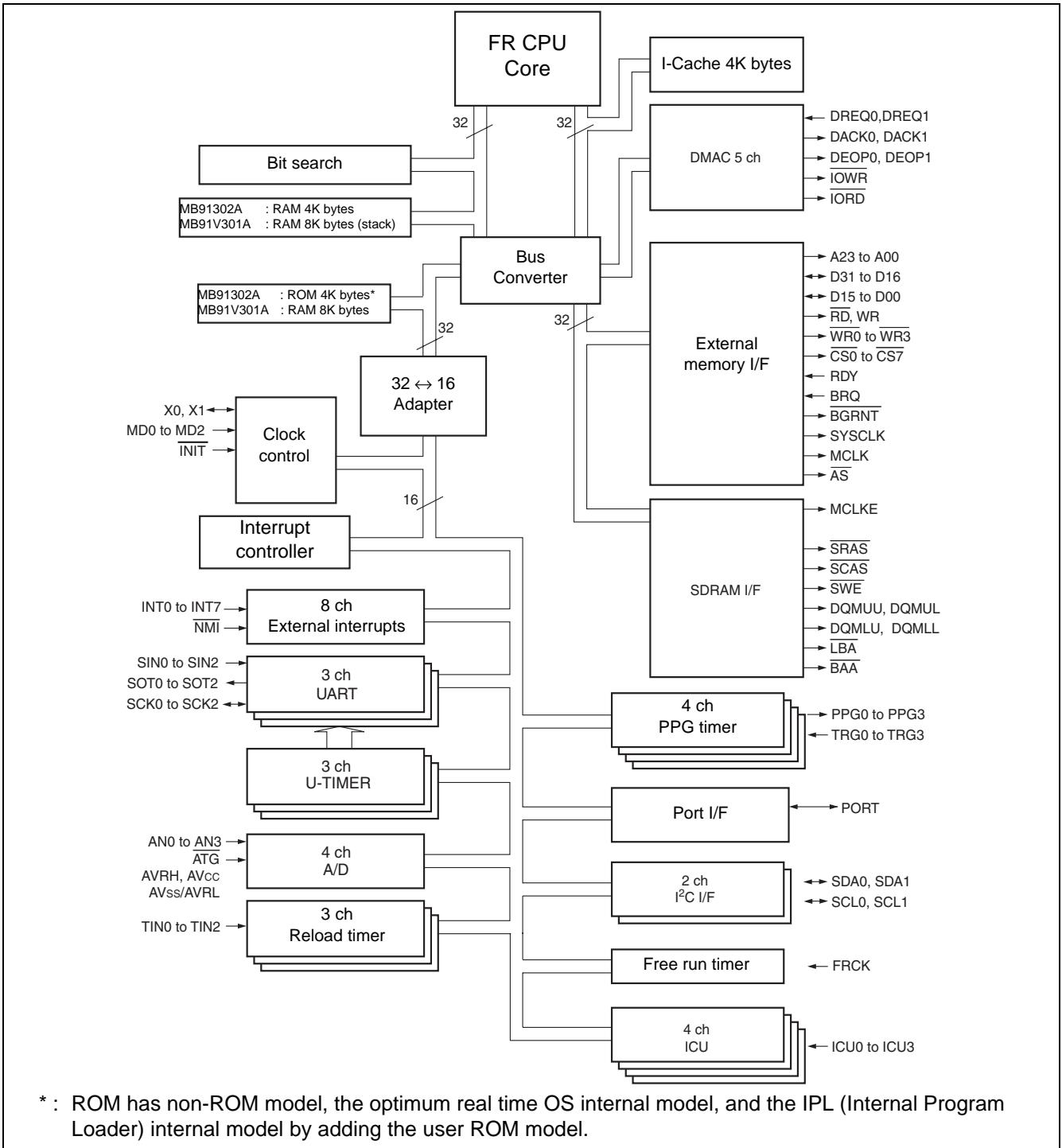
Using this function, for example, writing on board to the Flash memory connected to the external can be executed.

1.2 Block Diagram

Figure 1.2-1 are the block diagram of the MB91301 series.

■ Block Diagram

Figure 1.2-1 Block Diagram

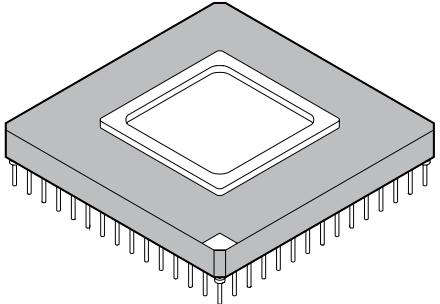


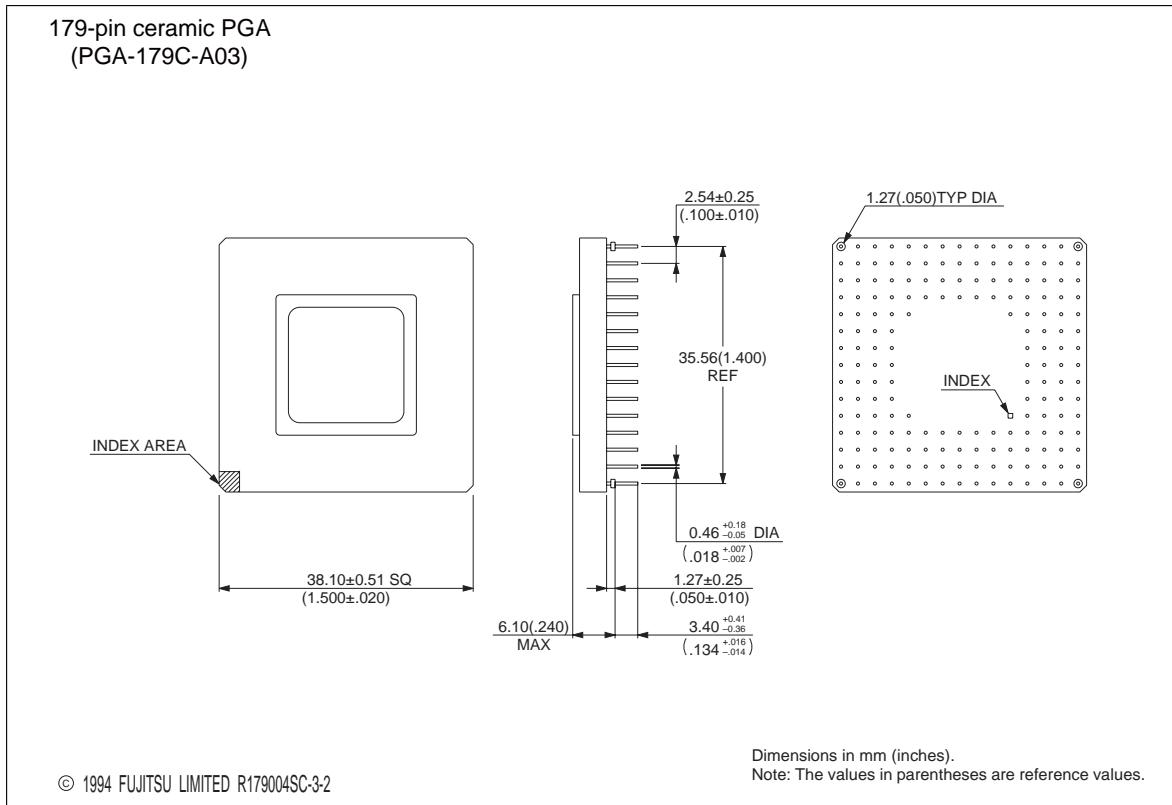
1.3 Package Dimensions

The MB91301 series is available in three types of package.

■ Package Dimension of PGA-179C-A03

Figure 1.3-1 PGA-179C-A03

179-pin ceramic PGA  (PGA-179C-A03)	Lead pitch 2.54mm(100mil) Pin matrix 15 Sealing method Metal seal

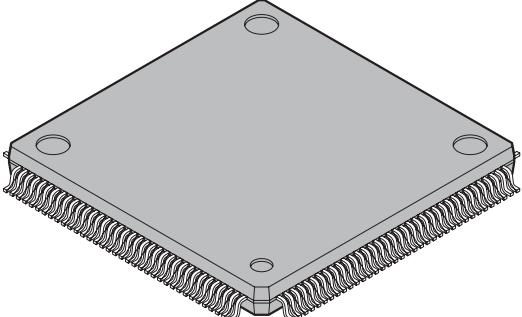


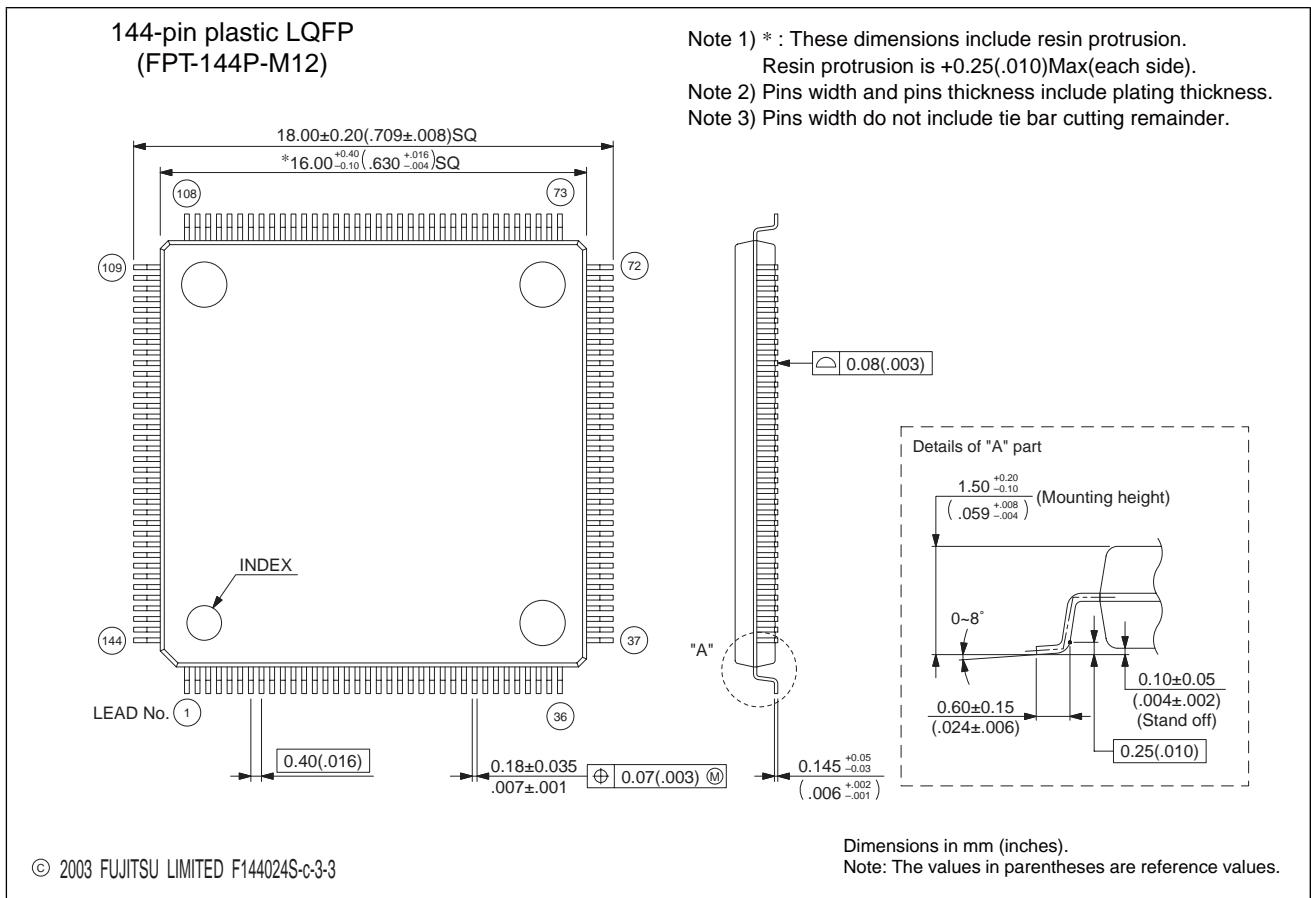
Please confirm the latest Package dimension by following URL.

<http://edevice.fujitsu.com/fj/DATASHEET/ef-ovpkv.html>

■ Package Dimension of FPT-144P-M12

Figure 1.3-2 FPT-144P-M12

 144-pin plastic LQFP (FPT-144P-M12)	Lead pitch 0.40 mm
Package width × package length	16.0 × 16.0 mm
Lead shape	Gullwing
Sealing method	Plastic mold
Mounting height	1.70 mm MAX
Weight	0.88 g
Code (Reference)	P-LFQFP144-16×16-0.40



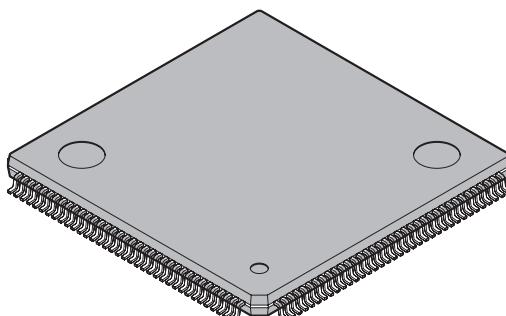
Please confirm the latest Package dimension by following URL.

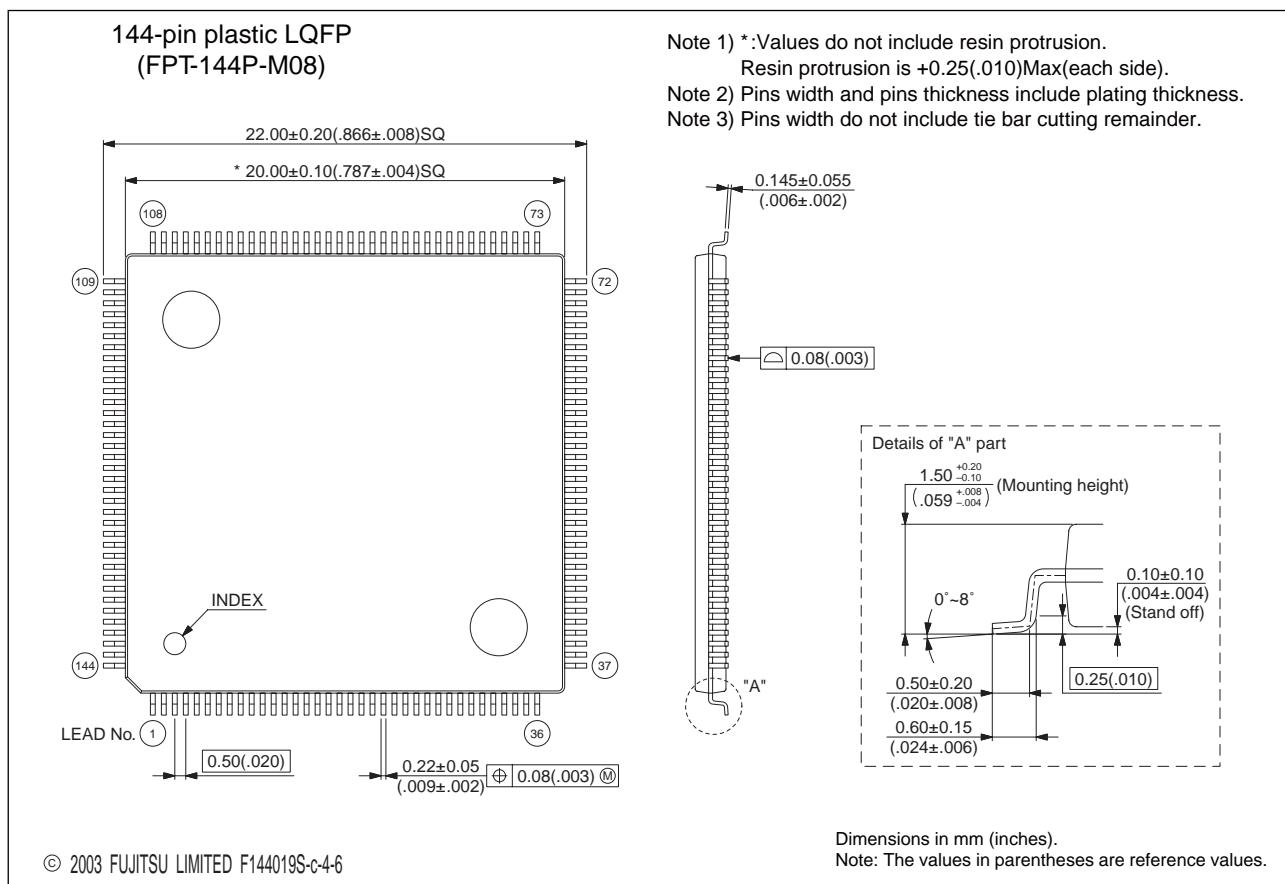
<http://edevice.fujitsu.com/fj/DATASHEET/ef-ovpkv.html>

CHAPTER 1 OVERVIEW

■ Package Dimension of FPT-144P-M08

Figure 1.3-3 FPT-144P-M08

 (FPT-144P-M08)	Lead pitch	0.50 mm
	Package width x package length	20.0 x 20.0 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm MAX
	Weight	1.20g
	Code (Reference)	P-LFQFP144-20x20-0.50



Please confirm the latest Package dimension by following URL.

<http://edevice.fujitsu.com/fj/DATASHEET/ef-ovpkv.html>

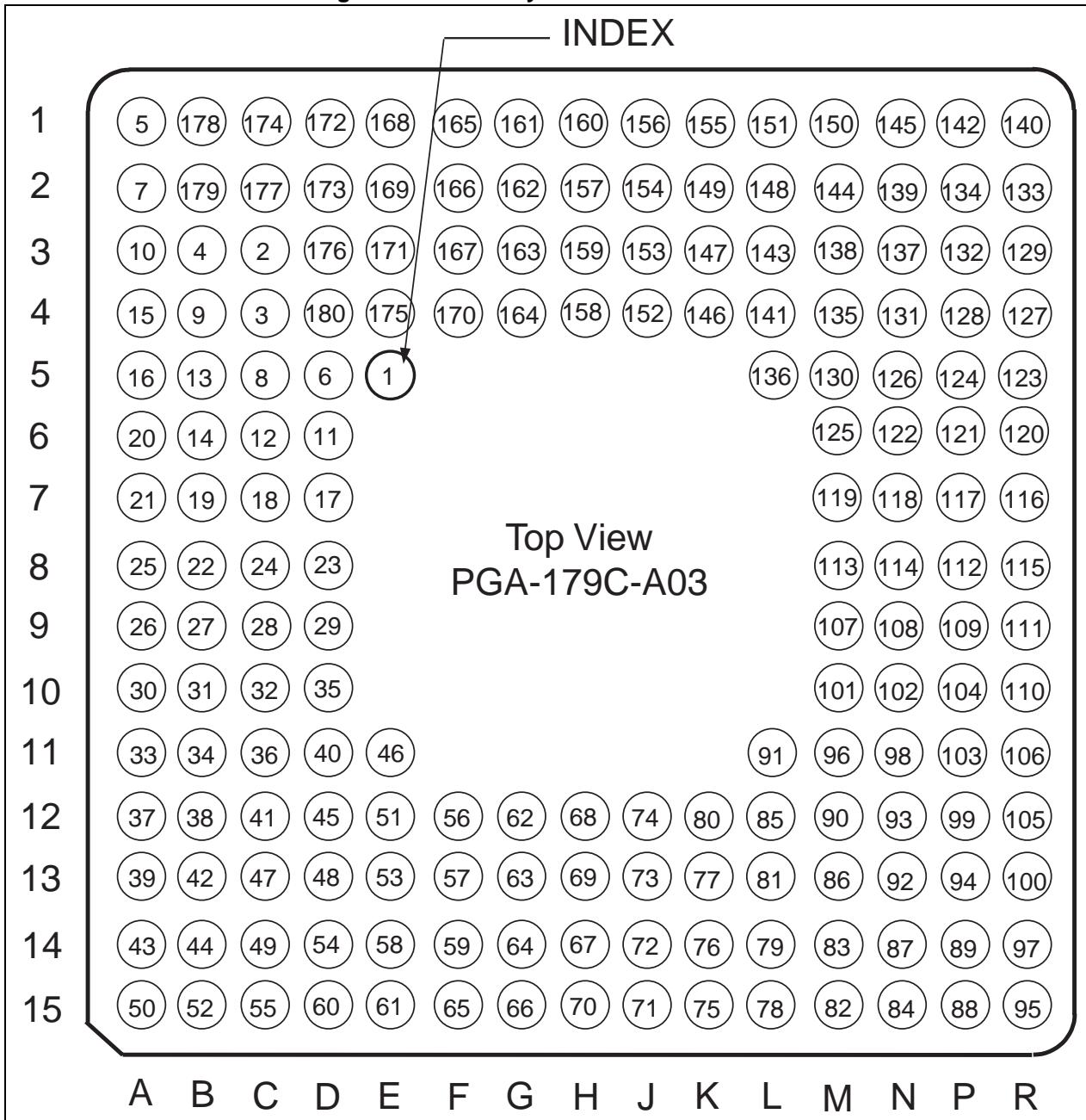
1.4 Pin Layout

This section shows the pin layout of the MB91V301A, MB91302A.

■ Pin Layout of the MB91V301A

Figure 1.4-1 is a diagram of the pin layout of the MB91V301A.

Figure 1.4-1 Pin Layout of the MB91V301A

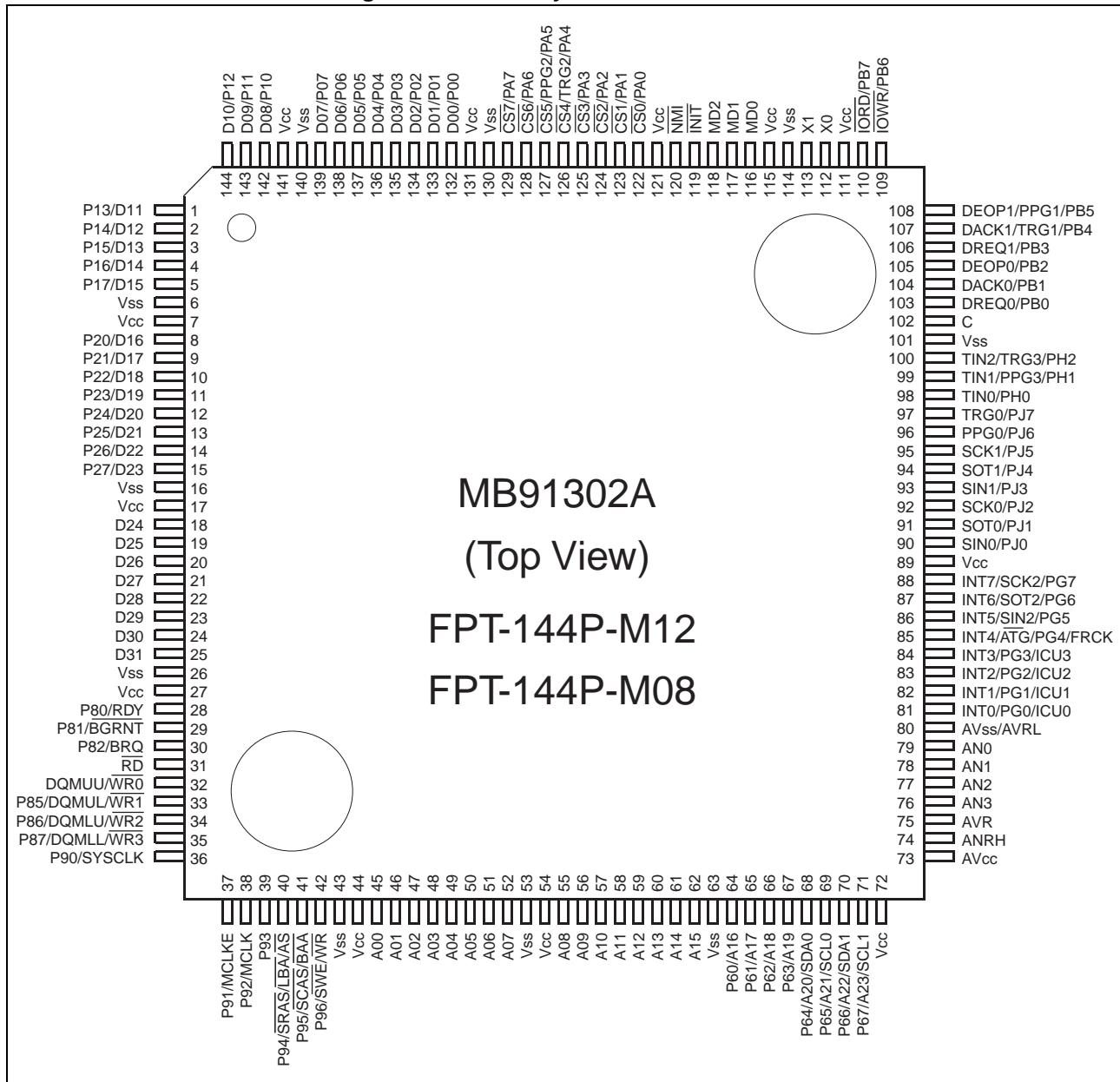


CHAPTER 1 OVERVIEW

■ Pin Layout of the MB91302A

Figure 1.4-2 is a diagram of the pin layout of the MB91302A.

Figure 1.4-2 Pin Layout of the MB91302A



1.5 Pin No. Table

The pin No. table of the MB91V301A is shown.

■ Pin No. Table

Table 1.5-1 MB91V301A Pin No. Table (Package: PGA-179C-A03) (1 / 2)

No.	PIN	Pin Name	No.	PIN	Pin Name	No.	PIN	Pin Name
1	E5	N.C.	31	B10	V _{SS}	61	E15	A07
2	C3	P13/D11	32	C10	V _{CC}	62	G12	V _{SS}
3	C4	V _{SS}	33	A11	P80/RDY	63	G13	V _{CC}
4	B3	V _{CC}	34	B11	P81/GRN _T	64	G14	A08
5	A1	P14/D12	35	D10	P82/BRQ	65	F15	A09
6	D5	P15/D13	36	C11	RD	66	G15	A10
7	A2	P16/D14	37	A12	DQMUU/WR0	67	H14	A11
8	C5	P17/D15	38	B12	P85/DQMUL/WR1	68	H12	A12
9	B4	V _{SS}	39	A13	P86/DQMLU/WR2	69	H13	A13
10	A3	V _{CC}	40	D11	P87/DQMLL/WR3	70	H15	A14
11	D6	P20/D16	41	C12	V _{SS}	71	J15	A15
12	C6	P21/D17	42	B13	V _{CC}	72	J14	V _{SS}
13	B5	P22/D18	43	A14	P90/SYSLCK	73	J13	V _{CC}
14	B6	P23/D19	44	B14	P91/MCLKE	74	J12	P60/A16
15	A4	P24/D20	45	D12	P92/MCLK	75	K15	P61/A17
16	A5	P25/D21	46	E11	P93	76	K14	P62/A18
17	D7	P26/D22	47	C13	V _{SS}	77	K13	P63/A19
18	C7	P27/D23	48	D13	V _{CC}	78	L15	SDA0/P64/A20
19	B7	V _{SS}	49	C14	P94/SRAS/LBA/AS	79	L14	SCL0/P65/A21
20	A6	V _{CC}	50	A15	P95/SCAS/BAA	80	K12	SDA1/P66/A22
21	A7	D24	51	E12	P96/SWE/WR	81	L13	SCL1/P67/A23
22	B8	D25	52	B15	V _{SS}	82	M15	V _{CC}
23	D8	D26	53	E13	V _{CC}	83	M14	V _{CC}
24	C8	D27	54	D14	A00	84	N15	EWR3
25	A8	V _{SS}	55	C15	A01	85	L12	EWR2
26	A9	V _{CC}	56	F12	A02	86	M13	EWR1
27	B9	D28	57	F13	A03	87	N14	EWR0
28	C9	D29	58	E14	A04	88	P15	ECS
29	D9	D30	59	F14	A05	89	P14	EMRAM
30	A10	D31	60	D15	A06	90	M12	ICD3

CHAPTER 1 OVERVIEW

Table 1.5-1 MB91V301A Pin No. Table (Package: PGA-179C-A03) (2 / 2)

No.	PIN	Pin Name	No.	PIN	Pin Name	No.	PIN	Pin Name
91	L11	ICD2	121	P6	SOT0/PJ1	151	L1	V _{CC}
92	N13	ICD1	122	N6	SCK0/PJ2	152	J4	<u>INIT</u>
93	N12	ICD0	123	R5	SIN1/PJ3	153	J3	<u>NMI</u>
94	P13	V _{SS}	124	P5	SOT1/PJ4	154	J2	V _{SS}
95	R15	V _{CC}	125	M6	SCK1/PJ5	155	K1	V _{CC}
96	M11	BREAK	126	N5	PPG0/PJ6	156	J1	<u>CS0/PA0</u>
97	R14	ICLK	127	R4	TRG0/PJ7	157	H2	<u>CS1/PA1</u>
98	N11	ICS2	128	P4	TIN0/PH0	158	H4	<u>CS2/PA2</u>
99	P12	ICS1	129	R3	TIN1/PPG3/PH1	159	H3	<u>CS3/PA3</u>
100	R13	ICS0	130	M5	TIN2/TRG3/PH2	160	H1	<u>CS4/TRG2/PA4</u>
101	M10	<u>TRST</u>	131	N4	V _{SS}	161	G1	<u>CS5/PPG2/PA5</u>
102	N10	C	132	P3	C	162	G2	<u>CS6/PA6</u>
103	P11	AV _{CC}	133	R2	DREQ0/PB0	163	G3	<u>CS7/PA7</u>
104	P10	AVRH	134	P2	DACK0/PB1	164	G4	V _{SS}
105	R12	AVR	135	M4	DEOP0/PB2	165	F1	V _{CC}
106	R11	AN3	136	L5	DREQ1/PB3	166	F2	D00/P00
107	M9	AN2	137	N3	DACK1/TRG1/PB4	167	F3	D01/P01
108	N9	AN1	138	M3	DEOP1/PPG1/PB5	168	E1	D02/P02
109	P9	AN0	139	N2	<u>IOWR/PB6</u>	169	E2	D03/P03
110	R10	AV _{SS} /AVRL	140	R1	<u>IORD/PB7</u>	170	F4	V _{SS}
111	R9	INT0/PG0/ICU0	141	L4	V _{CC}	171	E3	V _{CC}
112	P8	INT1/PG1/ICU1	142	P1	V _{SS}	172	D1	D04/P04
113	M8	INT2/PG2/ICU2	143	L3	X0	173	D2	D05/P05
114	N8	INT3/PG3/ICU3	144	M2	X1	174	C1	D06/P06
115	R8	INT4/ <u>ATG</u> /PG4/FRCK	145	N1	V _{SS}	175	E4	D07/P07
116	R7	INT5/SIN2/PG5	146	K4	V _{CC}	176	D3	V _{SS}
117	P7	INT6/SOT2/PG6	147	K3	MD0	177	C2	V _{CC}
118	N7	INT7/SCK2/PG7	148	L2	MD1	178	B1	D08/P10
119	M7	V _{CC}	149	K2	MD2	179	B2	D09/P11
120	R6	SIN0/PJ0	150	M1	V _{CC}	180	D4	D10/P12

1.6 List of Pin Functions

This section describes the pin functions of the MB91V301A, MB91302A.

■ Description of Pin Functions

Table 1.6-1 lists the pin of the MB91V301A, MB91302A and their functions.

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (1 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
132 to 139	166 to 169, 172 to 175	D00 to D07	J	External data bus bits 0 to 7. It is available in the external bus mode.
		P00 to P07		Can be used as ports in 8-bit or 16-bit external bus mode.
142 to 144, 1 to 5	178 to 180, 2, 5 to 8	D08 to D15	J	External data bus bits 8 to 15. It is available in the external bus mode.
		P10 to P17		Can be used as ports in 8-bit or 16-bit external bus mode.
8 to 15	11 to 18	D16 to D23	J	External data bus bits 16 to 23. It is available in the external bus mode.
		P20 to P27		Can be used as ports in 8-bit external bus mode.
18 to 25	21 to 24, 27 to 30	D24 to D31	C	External data bus bits 24 to 31. It is available in the external bus mode.
28	33	RDY	J	External ready input. The pin has this function when external ready input is enabled. Active level is "H".
		P80		General purpose input/output port. The pin has this function when external ready input is disabled.
29	34	BGRNT	J	Acknowledge output for external bus release. Outputs "L" when the external bus is released. The pin has this function when output is enabled.
		P81		General purpose input/output port. The pin has this function when output is disabled for external bus release acknowledge.
30	35	BRQ	J	External bus release request input. Input "1" to request release of the external bus. The pin has this function when input is enabled. The active level is "H".
		P82		General purpose input/output port. The pin has this function when the external bus release request input is disabled.
31	36	RD	C	External bus read strobe output. This pin is enabled at external bus mode.
32	37	WR0/ DQMUU	C	External bus write strobe output. This pin is enabled at external bus mode. When WR0 is used as the write strobe, this becomes the byte-enable pin (UUB).

CHAPTER 1 OVERVIEW

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (2 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
33	38	WR1/ DQMUL	J	External bus write strobe output. The pin has this function when WR1 output is enabled. When WR1 is used as the write strobe, this becomes the byte-enable pin (ULB).
		P85		General purpose input/output port. The pin has this function when the external bus write-enable output is disabled.
34	39	WR2/ DQMLU	J	External bus write strobe output. The pin has this function when WR2 output is enabled. When WR2 is used as the write strobe, this becomes the byte-enable pin (LUB).
		P86		General purpose input/output port. The pin has this function when the external bus write-enable output is disabled.
35	40	WR3/ DQMLL	J	External bus write strobe output. The pin has this function when WR3 output is enabled. When WR3 is used as the write strobe, this becomes the byte-enable pin (LLB).
		P87		General purpose input/output port. The pin has this functions when the external bus write-enable output is disabled.
36	43	SYSCLK	C	System clock output. The pin has this function when system clock output is enabled. This outputs the same clock as the external bus operating frequency. (Output halts in stop mode.)
		P90		General purpose input/output port. The pin has this function when system clock output is disabled.
37	44	MCLKE	J	Clock enable signal for memory.
		P91		General purpose input/output port. The pin has this function when clock enable output is disabled.
38	45	MCLK	C	Memory clock output. The pin has this function when memory clock output is enabled. This outputs the same clock as the external bus operating frequency. (Output halts in stop and sleep mode.)
		P92		General purpose input/output port. The pin has this function when memory clock output is disabled.
39	46	P93	C	General purpose input/output port.

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (3 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
40	49	AS	J	Address strobe output. The pin has this function without EDRAM area, when ASXE bit of port function register 9 is enabled.
		LBA		Address strobe output for burst flash ROM. The pin has this function in normal accessed area that is set over "1", when ASXE bit of port function register 9 is enabled.
		SRAS		RAS signal for SDRAM. This pin has this function for accessing to SDRAM area, when ASXE bit of port function register 9 is enabled.
		P94		General purpose input/output port. The pin has this function, when ASXE bit of port function register 9 is set as the general purpose port.
41	50	BAA	J	Address advance output for burst Flash ROM. The pin has this function when BAAE bit of port function register is enabled.
		SCAS		CAS signal for SDRAM. This pin has this function in SDRAM area, when BAAE bit of port function register is enabled.
		P95		General purpose input/output port. The pin has this function when BAAE bit of port function register is general purpose port.
42	51	WR	J	Memory write strobe output. This pin has this function when WEXE bit of port function register is enabled.
		SWE		Write output for SDRAM. This pin has this function when WEXE bit of port function register is enabled.
		P96		General purpose input/output port. This pin has this function when WEXE bit of port function register is general purpose port.
45 to 52	54 to 61	A00 to A07	C	External address bit 0 to 7.
55 to 62	64 to 71	A08 to A15	C	External address bit 8 to 15.
64 to 67	74 to 77	A16 to A19	J	External address bit 16 to 19. It can be used as ports when external address bus is unused.
		P60 to P63		Can be used as ports when external bus is 8-bit mode.

CHAPTER 1 OVERVIEW

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (4 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
68	78	SDA0	J, T	Data I/O pin for I ² C bus. This function is enable when typical operation of I ² C is enable. The port output must remain off unless intentionally turned on. (Open drain output) (This function is only for MB91302A, MB91V301A.)
		A20		External address bus bit 20. This function is enable during prohibited I ² C operation and using external bus.
		P64		General-purpose I/O port. This function is enable during prohibited I ² C and nonused external address bus.
69	79	SCL0	J, T	CLK I/O pin for I ² C bus. This function is enable when typical operation of I ² C is enable. The port output must remain off unless intentionally turned on. (open drain output) (This function is only for MB91302A, MB91V301A.)
		A21		External address bit 21. This function is enable during prohibited I ² C operation and using external bus.
		P65		General-purpose I/O port. This function is enable during prohibited I ² C and unused external address bus.
70	80	SDA1	J, T	DATA I/O pin for I ² C bus. This function is enable when typical operation of I ² C is enable. The output must remains off unless intentionally turned on. (open drain output) (This function is only for MB91302A, MB91V301A.)
		A22		External address bit 20. This function is enable during prohibited I ² C operation and using external bus.
		P66		General-purpose I/O port. This function is enable during prohibited I ² C and unused external address bus.
71	81	SCL1	J, T	CLK I/O pin for I ² C bus. This function is enable when typical operation of I ² C is enable. The port output must remains off unless intentionally turned on. (open drain output) (This function is only for MB91302A, MB91V301A.)
		A23		External address bit 21. This function is enable during prohibited I ² C operation and unusing external address bus.
		P67		General-purpose I/O port. This function is enable during prohibited I ² C operation and nonused external address bus.
76 to 79	106 to 109	AN3 to AN0	D	Analog input pin.

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (5 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
81 to 84	111 to 114	INT0 to INT3	L	External interrupt inputs. These inputs are used continuously when the corresponding external interrupt is enabled. In this case, do not output to these ports unless doing so intentionally. Refer to "9.2.3 External Interrupt Request Level Setting Register (ELVR)" for active level setting.
		PG0 to PG3		General purpose input/output ports.
		ICU0 to ICU3		Input capture input pins. These inputs are used continuously when selected as input capture inputs. In this case, do not output to these ports unless doing so intentionally. (This function is only for MB91302A and MB91V301A.)
85	115	INT4	L, V	External interrupt input. These inputs are used continuously when the corresponding external interrupt is enabled. In this case, do not output to these ports unless doing so intentionally. Refer to "9.2.3 External Interrupt Request Level Setting Register (ELVR)" for active level setting.
		ATG		External trigger input for A/D converter. This input is used continuously when selected as the A/D converter start trigger. In this case, do not output to this port unless doing so intentionally.
		PG4		General purpose input/output ports.
		FRCK		[FRCK] External clock input pin of free run timer. These inputs are used continuously when using as external clock input pin of free run timer. In this case, do not output to these ports unless doing so intentionally (This function is only for MB91302A and MB91V301A).
86	116	INT5	L, V	External interrupt input. These inputs are used continuously when the corresponding external interrupt is enabled. In this case, do not output to these ports unless doing so intentionally. Refer to "9.2.3 External Interrupt Request Level Setting Register (ELVR)" for active level setting.
		SIN2		Input pin for UART2 data. This pin is used continuously when UART2 is performing input. In this case, do not output to this port unless doing so intentionally.
		PG5		General purpose input/output port.

CHAPTER 1 OVERVIEW

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (6 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
87	117	INT6	L, V	External interrupt input. This input is used continuously when the corresponding external interrupt is enabled. In this case, do not output to these ports unless doing so intentionally. Refer to "9.2.3 External Interrupt Request Level Setting Register (ELVR)" for active level setting.
		SOT2		Output pin for UART2 data. The pin has this function when UART2 data output is enabled.
		PG6		General purpose input/output port.
88	118	INT7	L, V	External interrupt input. This input is used continuously when the corresponding external interrupt is enabled. In this case, do not output to these ports unless doing so intentionally. Refer to "9.2.3 External Interrupt Request Level Setting Register (ELVR)" for active level setting.
		SCK2		Input/output pin for UART2 clock. The pin has this function when UART2 clock output is enabled.
		PG7		General purpose input/output port.
90	120	SIN0	K, U	Input pin for UART0 data. This input is used continuously when UART0 is performing input. In this case, do not output to this port unless doing so intentionally.
		PJ0		General purpose input/output port.
91	121	SOT0	J, U	Output pin for UART0 data. The pin has this function when UART0 data output is enabled.
		PJ1		General purpose input/output port.
92	122	SCK0	K, U	Input/output pin for UART0 clock. The pin has this function when UART0 clock output is enabled.
		PJ2		General purpose input/output port.
93	123	SIN1	K, U	Input pin for UART1 data. This input is used continuously when UART1 is performing input. In this case, do not output to this port unless doing so intentionally.
		PJ3		General purpose input/output port.
94	124	SOT1	J, U	Output pin for UART1 data. The pin has this function when UART1 data output is enabled.
		PJ4		General purpose input/output port.
95	125	SCK1	K, U	Input/output pin for UART1 clock. The pin has this function when UART1 clock output is enabled.
		PJ5		General purpose input/output port.
96	126	PPG0	J, U	PPG timer output. This pin has this function when PPG0 output is enabled.
		PJ6		General purpose input/output port.

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (7 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
97	127	TRG0	J, U	External trigger input for PPG timer. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port unless doing so intentionally. ^{*1}
		PJ7		General purpose input/output port.
98	128	TIN0	J	Reload timer input. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port unless doing so intentionally. ^{*2}
		PH0		General purpose input/output port.
99	129	TIN1	J	Reload timer input. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port unless doing so intentionally. ^{*2}
		PPG3		PPG timer output. The pin has this function when PPG3 output is enabled.
		PH1		General purpose input/output port.
100	130	TIN2	J	Reload timer input. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port unless doing so intentionally. ^{*2}
		TRG3		External trigger input for PPG timer. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port unless doing so intentionally. ^{*1}
		PH2		General purpose input/output port.
103	133	DREQ0	J	External input for DMA transfer requests. This input is used continuously when the corresponding external input for DMA transfer requests are enabled. In this case, do not output to this port unless doing so intentionally. Refer to "14.3.1 Setting a Transfer Request" for active level setting.
		PB0		General purpose input/output port.
104	134	DACK0	J	External acknowledge output for DMA transfer requests. The pin has this function when external acknowledge output for DMA transfer requests is enabled.
		PB1		General purpose input/output port.
105	135	DEOP0	J	Completion output for DMA external transfer. The pin has this function when completion output for DMA external transfer is enabled.
		PB2		General purpose input/output port.

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (8 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
106	136	DREQ1	J	External input for DMA transfer requests. This input is used continuously when external input for DMA transfer request is enabled. In this case, do not output to this port unless doing so intentionally. Refer to "14.3.1 Setting a Transfer Request" for active level.
		PB3		General purpose input/output port.
107	137	DACK1	J	External acknowledge output for DMA transfer requests. The pin has this function when external acknowledge output for DMA transfer requests is enabled.
		TRG1		External trigger input for PPG timer. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to this port and external acknowledge output for DMA transfer request unless doing so intentionally.*1
		PB4		General purpose input/output port.
108	138	DEOP1	J	Completion output for DMA external transfer. The pin has this function when completion output for DMA external transfer is enabled.
		PPG1		PPG timer output. The pin has this function when PPE1 bit is enabled.
		PB5		General purpose input/output port.
109	139	<u>IOWR</u>	J	Write strobe output for DMA fly-by transfer. The pin has this function when outputting a write strobe for DMA fly-by transfer is enabled.
		PB6		General purpose input/output port. The pin has this function when outputting a write strobe for DMA fly-by transfer is disabled.
110	140	<u>IORD</u>	J	Read strobe output for DMA fly-by transfer. The pin has this function when outputting a read strobe for DMA fly-by transfer is enabled.
		PB7		General purpose input/output port. The pin has this function when outputting a read strobe for DMA fly-by transfer is disabled.
112	143	X0	A	Clock (oscillation) input.
113	144	X1	A	Clock (oscillation) output.
116 to 118	147 to 149	MD0 to MD2	G	Mode pins to 0 to 2. The levels applied to these pins set the basic operating mode. Connect V _{CC} or V _{SS} .
119	152	<u>INIT</u>	B	External reset input (Reset to initialize settings) ("L" active)
120	053	<u>NMI</u>	M	NMI (Non Maskable Interrupt) input ("L" active)

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (9 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
122	156	$\overline{\text{CS0}}$	J	Chip select 0 output. The pin has this function when CS0 area of CSER (Chip Select Enable Register) is enabled and the specified CS0XE bit of port function register is enabled.
		PA0		General purpose input/output port. The pin has this function when CS0XE bit of port function register is general purpose port.
123	157	$\overline{\text{CS1}}$	J	Chip select 1 output. The pin has this function when CS1 area of CSER is enabled and the specified CS1XE bit of port function register is enabled.
		PA1		General purpose input/output port. The pin has this function when chip select 1 output is disabled.
124	158	$\overline{\text{CS2}}$	J	Chip select 2 output. The pin has this function when CS2 area of CSER is enabled and the specified CS2XE bit of port function register is enabled.
		PA2		General purpose input/output port. The pin has this function when chip select 2 output is disabled.
125	159	$\overline{\text{CS3}}$	J	Chip select 3 output. The pin has this function when CS3 area of CSER is enabled and the specified CS3XE bit of port function register is enabled.
		PA3		General purpose input/output port. The pin has this function when chip select 3 output is disabled.
126	160	$\overline{\text{CS4}}$	J	Chip select 4 output. The pin has this function when CS4 area of CSER is enabled and the specified CS4XE bit of port function register is enabled.
		TRG2		External trigger input for PPG timer. This input is used continuously when the corresponding timer input is enabled. In this case, do not output to chip select and this port unless doing so intentionally. *1
		PA4		General purpose input/output port. The pin has this function when chip select 4 output is disabled.
127	161	$\overline{\text{CS5}}$	J	Chip select 5 output. The pin has this function when CS5 area of CSER is enabled and the specified CS5XE bit of port function register is enabled.
		PPG2		PPG timer output. The pin has this function when PPE2 bit is enabled.
		PA5		General purpose input/output port. The pin has this function when chip select 5 output and PPG timer output are disabled.

CHAPTER 1 OVERVIEW

Table 1.6-1 List of Pin Function (except for Power Supply, and GND Pins) (10 / 10)

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
128	162	CS6	J	Chip select 6 output. The pin has this function when CS6 area of CSER is enabled and the specified CS6XE bit of port function register is enabled.
		PA6		General purpose input/output port. The pin has this function when chip select 6 output are disabled.
129	163	CS7	J	Chip select 7 output. The pin has this function when CS7 area of CSER is enabled and the specified CS7XE bit of port function register is enabled.
		PA7		General purpose input/output port. The pin has this function when chip select 7 output is disabled.

*1 : Refer to "[bit7, bit6] EGS1, EGS0: Trigger input edge selection bit" in "CHAPTER 7 PPG TIMER" for active level setting.

*2 : Refer to "[bit9, bit8, bit7] MOD2, MOD1, MOD0 (MODE): Setting of operation mode" in "CHAPTER 6 16-BIT RELOAD TIMER" for active level.

Table 1.6-2 Power Supply and GND Pins

Pin no.		Pin name	Function
MB91302A	MB91V301A		
6, 16, 26, 43, 53, 63, 101, 114, 130, 140	3, 9, 19, 25, 31, 41, 47, 52, 62, 72, 94, 131, 142, 145, 154, 164, 170, 176	V _{SS}	GND pins. Connect all pins at the same potential.
7, 17, 27, 44, 54, 72, 89, 111, 121, 115, 131, 141	4, 10, 20, 26, 32, 42, 48, 53, 63, 73, 82, 83, 95, 119, 141, 146, 150, 151, 155, 165, 171, 177	V _{CC}	3 V power supply pins. Connect all pins at the same potential.
73	103	A _V _{CC}	Analog power supply pin for A/D converter
74	104	A _V R _H	Reference power supply pin for A/D converter
75	105	A _V R	Capacitor coupling pin for the A/D converter
80	110	A _V _{SS} /A _V _{RL}	Analog GND pin for A/D converter
-	1	N.C.	N.C. pin. Use at open
102	102, 132	C	Capacitor coupling pin for the internal regulator

Table 1.6-3 Tool Pins

Pin no.		Pin name	I/O circuit type	Function
MB91302A	MB91V301A			
-	97	ICLK	S	Clock output
-	101	<u>TRST</u>	Q	Tool reset
-	98 to 100	ICS2 to ICS0	N	Device status output (during TRC) DSU4 operation status output (during EML)
-	90 to 93	ICD3 to ICD0	R	Trace information output (during TRC) Program/data I/O (during EML)
-	96	BREAK	P	DSU4 break request input
-	89	EMRAM	O	Emulation memory detection
-	88	<u>ECS</u>	N	Chip select for emulation memory
-	84 to 87	<u>EWR3</u> to <u>EWR0</u>	N	Write strobe for emulation memory

1.7 I/O Circuit Types

This section describes the I/O circuit types.

■ I/O Circuit Types

Table 1.7-1 I/O Circuit Types (1 / 4)

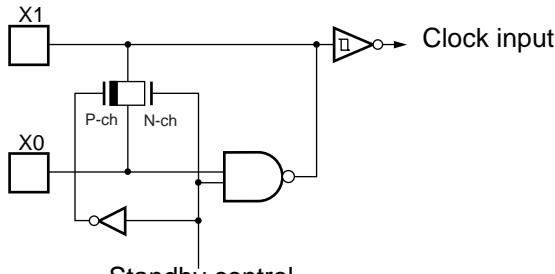
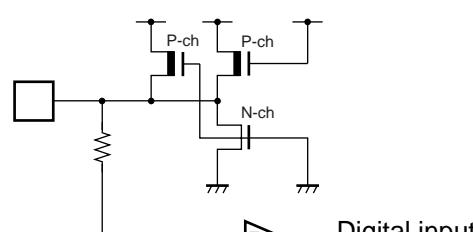
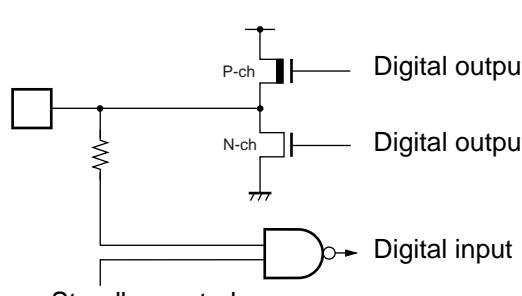
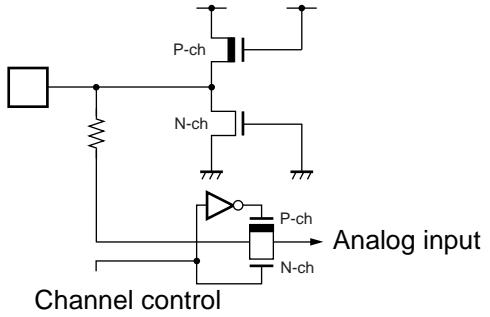
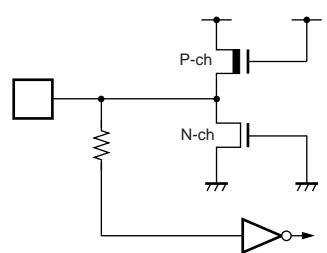
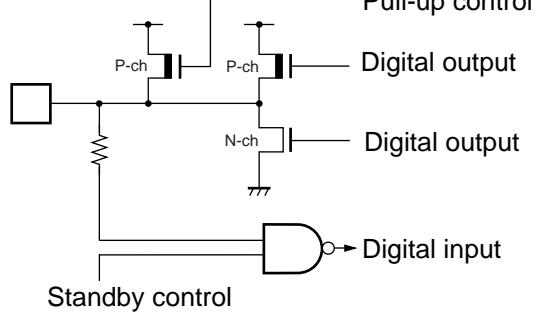
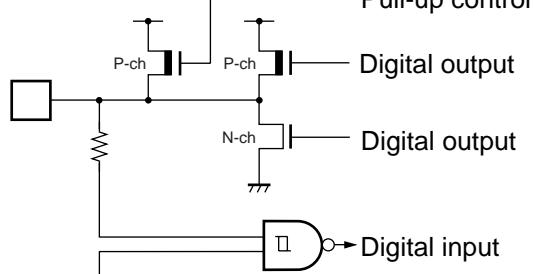
Type	Circuit	Remarks
A	 Standby control	Oscillation feedback resistance approx. 1 MΩ
B	 Digital input	CMOS hysteresis input with pull-up resistor Pull-up resistor value = approx. 25 kΩ (Typ)
C	 Digital output Digital output Standby control Digital input	CMOS level I/O with standby control $I_{OL} = 4 \text{ mA}$

Table 1.7-1 I/O Circuit Types (2 / 4)

Type	Circuit	Remarks
D	 <p>Channel control</p>	Analog input with switch
G		CMOS level output without standby control
J	 <p>Pull-up control Digital output Digital output Standby control</p>	CMOS level I/O with standby control and Pull-up control Pull-up resistor value = approx. 25 kΩ (Typ) $I_{OL} = 4 \text{ mA}$
K	 <p>Pull-up control Digital output Digital output Standby control</p>	CMOS level output CMOS level hysteresis input with standby control and Pull-up control Pull-up resistor value = approx. 25 kΩ (Typ) $I_{OL} = 4 \text{ mA}$

CHAPTER 1 OVERVIEW

Table 1.7-1 I/O Circuit Types (3 / 4)

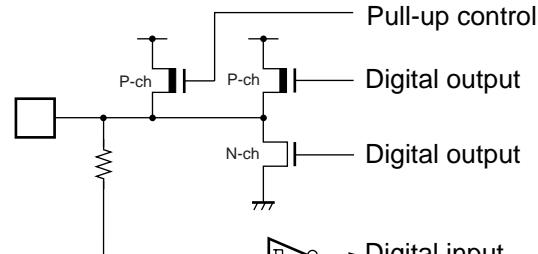
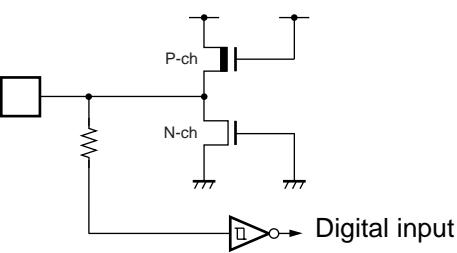
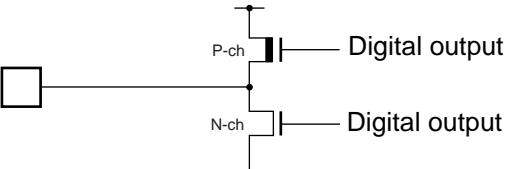
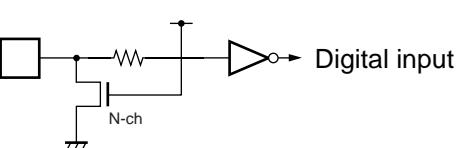
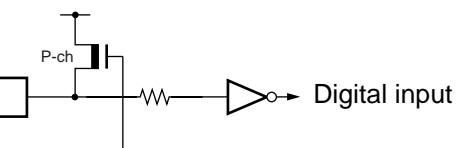
Type	Circuit	Remarks
L	 <p>Pull-up control Digital output Digital output Digital input</p>	CMOS level output CMOS level hysteresis input with Pull-up control no standby control Pull-up resistor value = approx. 25 kΩ (Typ) $I_{OL} = 4 \text{ mA}$
M	 <p>Digital input</p>	CMOS level hysteresis input no standby control
N	 <p>Digital output Digital output</p>	Output buffer CMOS level output $I_{OL} = 4 \text{ mA}$
O	 <p>Digital input</p>	Input buffer CMOS level input
P	 <p>Digital input</p>	Input buffer with pull-down Pull-down resistor value = approx. 25 kΩ (Typ)
Q	 <p>Digital input</p>	Input buffer with Pull-up Pull-up resistor value = approx. 25 kΩ (Typ)

Table 1.7-1 I/O Circuit Types (4 / 4)

Type	Circuit	Remarks
R	<p>Digital output</p> <p>Digital output</p> <p>Digital input</p>	I/O buffer with pull-down CMOS level output $I_{OL} = 4 \text{ mA}$ Pull-up resistor value = approx. $25 \text{ k}\Omega$ (Typ)
S	<p>Digital output</p> <p>Digital output</p> <p>Digital input</p>	I/O buffer CMOS level output $I_{OL} = 4 \text{ mA}$
T	<p>Pull-up control</p> <p>Digital output with open-drain control</p> <p>Digital output</p> <p>Digital input</p> <p>Standby Control</p>	N-ch open-drain output CMOS level I/O with standby control and Pull-up control Pull-up register value = approx. $25 \text{ k}\Omega$ (Typ) $I_{OL} = 4 \text{ mA}$
U	<p>Digital output</p> <p>Digital output</p> <p>Digital input</p> <p>Standby Control</p>	CMOS level output CMOS level hysteresis input with standby control 5 V tolerant $I_{OL} = 4 \text{ mA}$
V	<p>Digital output</p> <p>Digital output</p> <p>Digital input</p>	CMOS level output CMOS level hysteresis input without standby control 5 V tolerant $I_{OL} = 4 \text{ mA}$

CHAPTER 1 OVERVIEW

CHAPTER 2 HANDLING THE DEVICE

This chapter provides precautions on handling the MB91301 series.

- 2.1 Precautions on Handling the Device
- 2.2 Precautions on Handling Power Supplies

2.1 Precautions on Handling the Device

This section contains information on preventing a latch up and on the handling of pins.

■ Preventing a Latch Up

A latch up can occur if, on a CMOS IC, a voltage higher than V_{CC} or a voltage lower than V_{SS} is applied to an input or output pin or a voltage higher than the rating is applied between V_{CC} pin and V_{SS} pin. A latch up, if it occurs, significantly increases the power supply current and may cause thermal destruction of an element. When you use a CMOS IC, be very careful not to exceed the maximum rating.

■ Unused Input Pins

Do not leave an unused input pin open, since it may cause a malfunction. Handle by using a pull-up or pull-down resistor.

■ Power Supply Pins

If more than one V_{CC} or V_{SS} pin exist, those that must be kept at the same potential are designed to be connected to one other inside the device to prevent malfunctions such as latch up. Be sure to connect the pins to a power supply and ground external to the device to minimize undesired electromagnetic radiation, prevent strobe signal malfunctions due to an increase in ground level, and conform to the total output current rating. Given consideration to connecting the current supply source to V_{CC} or V_{SS} of the device at the lowest impedance possible.

It is also recommended that a ceramic capacitor of around $0.1 \mu F$ be connected between V_{CC} and V_{SS} at circuit points close to the device as a bypass capacitor.

■ Quartz Oscillation Circuit

Noise near the X0 or X1 pin may cause the device to malfunction. Design printed circuit boards so that X0, X1, the quartz oscillator (or ceramic oscillator), and the bypass capacitor to ground are located as near to one another as possible.

It is strongly recommended that printed circuit board artwork that surrounds the X0 and X1 pins with ground be used to increase the expectation of stable operation.

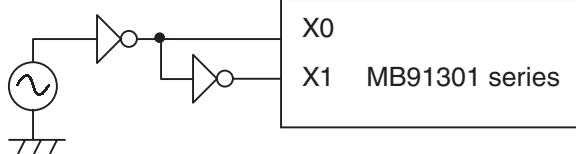
Please ask the crystal maker to evaluate the oscillational characteristics of the crystal and this device.

■ External Clock

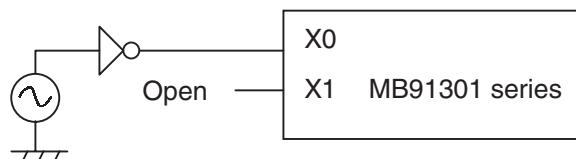
When using an external clock, in general supply it to the X0 pin while also supplying a reverse-phase clock to the X1 pin simultaneously. In this case, do not use the STOP mode (oscillation stop mode), since the X1 pin stops with "H" level output in STOP mode.

Additionally, the X0 pin can be used only if an external clock is supplied at 12.5 MHz.

Figure 2.1-1 and Figure 2.1-2 show examples of using an external clock.

Figure 2.1-1 Using an External Clock (Normal)

Note: Stop mode (oscillation stop mode) cannot be used.

Figure 2.1-2 Using an External Clock (Less than 12.5 MHz)

■ Treatment of NC and OPEN Pins

Pins marked as "NC" or "OPEN" must be left open-circuit.

■ Mode Pins (MD0 to MD2)

These pins must be directly connected to V_{CC} or V_{SS} when they are used. Keep the pattern length between a mode pin on a printed circuit board and V_{CC} or V_{SS} as short as possible so that they can be connected at a low impedance.

■ Precautions on Use

- **MB91301 series**

- **Clock controller**

Reserve a regulator wait time or an oscillation stabilization wait time when an "L" level signal is input to INIT.

- **Notes on during operation of PLL clock mode**

On this microcontroller, if in case the crystal oscillator breaks off or an external reference clock input stops while the PLL clock mode is selected, a self-oscillator circuit contained in the PLL may continue its operation at its self-running frequency. However, Fujitsu will not guarantee results of operations if such failure occurs.

- **MCLK and SYSCLK**

MCLK is stopped in sleep and stop modes, and SYSCLK is stopped only in stop mode. Use MCLK and SYSCLK appropriately according to the purpose of use.

- **Pull-up control**

If a pull-up resistor is connected to a pin to be used as an external bus pin, the AC ratings cannot be guaranteed.

Even a port that already has a pull-up resistor is invalid in stop mode (HIZ = 1) and hardware standby mode.

- **Bit search module**

Only word access is allowed for the BSD0, BSD1, and BSDC registers.

○ Low-power consumption mode

- (1) Be sure to use the following sequences after using the synchronous standby mode (TBCR: Set by time base counter control register bit8 SYNCs bit) when putting in the standby mode.

```
/* Write to STCR */
ldi    #_STCR, r0      ; STCR register (481H)
ldi    #Val_of_Stby, rl ; "Val_of_Stby" is the data to be written to STCR.
stb    rl, @r0          ; Write to STCR.

/* Write to CTBR */
ldi    #_CTBR, r2      ; CTBR register (483H)
ldi    #0xA5, rl        ; Clear command (1)
stb    rl, @r2          ; Write A5 to CTBR.
ldi    #0x5A, rl        ; Clear command (2)
stb    rl, @r2          ; Write 5A to CTBR.

/* Clear the time base counter at this point */
ldub   @r0, rl          ; Read from STCR.

/* Start moving to synchronous standby mode */
ldub   @r0, rl          ; Dummy read from STCR.
nop               ; NOP for timing adjustment x 5
nop
nop
nop
nop
```

- (2) Do not do the following when the monitor debugger is used.

- Set the break point to the above-mentioned instruction row.
- Execute the step for the above-mentioned instruction row.

○ Prefetch

When allowing prefetch from an area that has been set as a little endian area, limit access to the area to word access (i.e., access in units of 32 bits).

The area cannot be accessed correctly by byte or half-word accesses.

○ I/O port access

Only byte accesses are allowed to I/O ports.

○ Switching the function of a common port

Use the port function register (PFR) to switch the function of a pin which also serves as a port. However, use an external bus setting to switch the function of a bus pin.

○ D-bus memory

Do not set a code area in D-bus memory.

No instruction fetch is performed to the D-bus.

Instruction fetches to the D-bus area result in incorrect data interpreted as code, which can cause the microcontroller to lose control.

Do not set a data area in I-bus memory.

○ I-bus memory

Do not set a stack area or vector table in I-bus memory.

It may cause a hang during EIT processing (including RETI).

Recovery from the hang requires a reset.

Do not perform DMA transfer to I-bus memory.

○ Notes on the PS register

Since some instructions manipulate the PS register earlier, the following exceptions may cause the interrupt handler to break or the PS flag to update its display setting when the debugger is being used. As the microcontroller is designed to carry out reprocessing correctly upon returning from such an EIT event, it performs operations before and after the EIT as specified in either case.

- The following operations may be performed when the instruction immediately followed by a DIV0U/DIV0S instruction is (a) halted by a user interrupt or NMI, (b) single-stepped, or (c) breaks in response to a data event or emulator menu:
 1. D0 and D1 flags are updated earlier.
 2. The EIT handler (user interrupt/NMI or emulator) is executed.
 3. Upon returning from the EIT, the DIV0U/DIV0S instruction is executed and the D0 and D1 flags are updated to the same values as those in (1) above.
- The following operations are performed when the OR CCR/ST ILM/MOV Ri and PS instructions are executed to enable interruptions when a user interrupt or NMI trigger event has occurred.
 1. The PS register is updated earlier.
 2. The EIT handler (user interrupt/NMI) is executed.
 3. Upon returning from the EIT, the above instructions are executed and the PS register is updated to the same value as that in (1) above.

○ R15 (General purpose register)

When any of the following instructions is executed, the SSP* or USP* value is not used as R15, resulting in an incorrect value written to memory.

AND	R15, @Ri	ANDH	R15, @Ri	ANDB	R15, @Ri
OR	R15, @Ri	ORH	R15, @Ri	ORB	R15, @Ri
EOR	R15, @Ri	EORH	R15, @Ri	EORB	R15, @Ri
XCHB	@Rj, R15				

* : R15 is a virtual register. When a program attempts to access R15, the SSP or USP is accessed depending on the status of the "S" flag as an SP flag. When coding the above ten instructions using an assembler, specify a general-purpose register other than R15.

CHAPTER 2 HANDLING THE DEVICE

○ RETI instruction

Do not neither control register of the instruction cache nor the data access to RAM of the instruction cache immediately before the instruction of RETI.

○ Watchdog timer function

The watchdog timer function of this model monitors whether a program holds over a reset within a specified time. It also resets the CPU if the reset is not held over because of uncontrollable program operation. After the watchdog timer function is enabled, it keeps operating until a reset occurs.

The watchdog timer function usually holds over CPU reset automatically when program execution by the CPU stops. For the relevant exception conditions, see "3.12.7 Peripheral Circuits of Clock Controller".

The reset by the watchdog timer function might not occur if the above status is caused by uncontrollable system operation. If it might occur, a reset (INIT) request must be input from the external INIT pin.

○ A/D converter

When the device is turned on or returns from a reset or stop, it takes time for the external capacitor to be charged, requiring the A/D converter to wait for at least 10 ms.

■ Unique to the Evaluation Chip MB91V301A

○ Tool reset

On an evaluation board, use the chip with INIT and TRST connected together.

○ Single-stepping the RETI instruction

If an interrupt occurs frequently during single stepping, execute only the relevant processing routine repeatedly after single-stepping RETI. This will prevent the main routine and low-interrupt-level programs from being executed. Do not single-step the RETI instruction for avoidance purposes. When the debugging of the relevant interrupt routine becomes unnecessary, perform debugging with that interrupt disabled.

○ Simultaneous occurrences of a software break and a user interrupt/NMI

When a software break and a user interrupt /NMI take place at the same time, the emulator debugger can cause the following phenomena:

- The debugger stops pointing to a location other than the programmed breakpoints.
- The halted program is not re-executed correctly.

If these phenomena occur, use a hardware break instead of the software break. If the monitor debugger has been used, avoid setting any break at the relevant location.

○ Operand break

A stack pointer placed in an area set for a DSU operand break can cause a malfunction. Do not apply a data event break to access to the area containing the address of a system stack pointer.

○ ICE startup sequence

When using the ICE, when you start debugging, ensure that the bus configuration is set correctly for the area being used before downloading. After turning on the power to the target, the states of the \overline{RD} and $\overline{WR0}$ to $\overline{WR3}$ pins are undefined until you perform the above setting. Accordingly, include enabling pull-up as part of the startup sequence. If using these pins as general-purpose ports, set as output ports to prevent conflict with the output signals during the time the pin states are undefined.

Table 2.1-1 Pins which require handling when using the ICE

External bus width Pin name	32 bit	16 bit	8 bit
\overline{RD}	Pull-up	Pull-up	Pull-up
$\overline{WR0}$	Pull-up	Pull-up	Pull-up
$\overline{WR1}$ (P85)	Pull-up	Pull-up	*
$\overline{WR2}$ (P86)	Pull-up	*	*
$\overline{WR3}$ (P87)	Pull-up	*	*

* : Use as output ports.

CHAPTER 2 HANDLING THE DEVICE

○ Configuration batch file

The example batch file below sets the mode vector and sets up the CS0 configuration register for the download area. Use values appropriate to the hardware in the wait, timing, and other settings.

```
#-----
# Set MODR (0x7fd) =Enable In memory+16 bit External Bus
set mem/byte 0x7fd=0x5
#-----
# Set ASR0 (0x640) ; 0x0010_0000-0x002f_ffff
set mem/halfword 0x640=0x0010
#-----
# Set ACR0 (0x642)
#           ; ASZ [3:0]=0101:2M bytes
#           ; DBW [1:0]=01:16 bit width, automatically set from MODR
#           ; BST [1:0]=00:1 burst (16 bit x 2)
#           ; SREN=0:Disable BRQ
#           ; PFEN=1:Enable Pre fetch buffer
#           ; WREN=1:Enable Write operation
#           ; LEND=0: Big endian
#           ; TYPE [3:0]=0010:WEX: Disable RDY
set mem/halfword 0x642=0x5462
#-----
# Set AWR0 (0x660)
#           ; W15 to W12=0010:auto wait=2
#           ; WR07, WR06=01:RD, WR delay=1cycle
#           ; W05, W04=01:WR->WR delay=1cycle (for WEX)
#           ; W03 =1:MCLK->RD/WR delay=0.5cycle
#           ;       :for async Memory
#           ; W02 =0:ADR->CS delay=0
#           ; W01 =0:ADR->RD/WR setup 0cycle
#           ; W00 =RD/WR->ADR hold 0cycle
set mem/halfword 0x660=0x2058
#-----
```

○ Emulation memory

If SRAM as the emulation memory is built on target board, SRAM accessed by RD, WR signal, and +BYTE control signal can not be used. (The external bus is initialized to the bus mode for accessing RD, WRn after reset.)

2.2 Precautions on Handling Power Supplies

This section provides precautions on power supplies with regard to pin handling and processing when power is turned on.

■ Processing after Power-on

Immediately after power-on, be sure to apply a reset that initializes settings (INIT) from the $\overline{\text{INIT}}$ pin.

To provide for an oscillation stabilization wait time and regulator stabilization wait time immediately after power-on, continue to input the "L" level to the $\overline{\text{INIT}}$ pin as long as the oscillation stabilization wait time required by the oscillating circuit. (Initialization by INIT from the $\overline{\text{INIT}}$ pin sets the oscillation stabilization wait time to the minimum value.)

■ External Clock Input after Power-on

After power-on, be sure to input an external clock until the oscillation stabilization wait is canceled.

■ Indeterminate Output when the Power Is Turned On

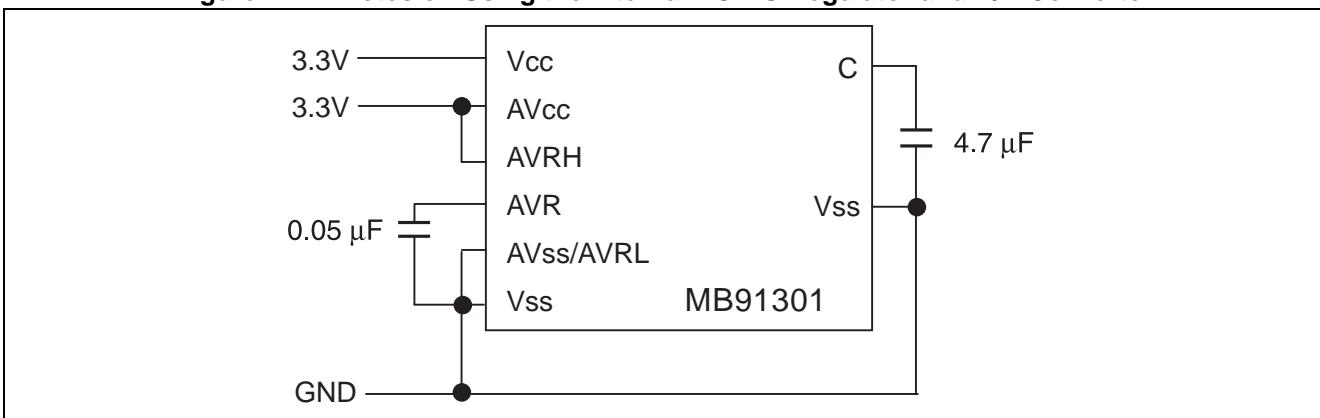
When the power is turned on, the output pin may remain unstable until the internal power supply becomes stable.

■ Notes on Using the Internal DC-DC Regulator and A/D Converter

The MB91301 series contains a regulator. Be sure to supply power to the V_{CC} pin at 3.3 V and add a bypass capacitor of about $4.7 \mu\text{F}$ for the regulator to the C pin.

The regulator contains an A/D converter and supplies power to AV_{CC} at 3.3 V. Be sure to insert a capacitor of at least $0.05 \mu\text{F}$ between the AVR and AV_{SS}/AV_{RL} pins.

Figure 2.2-1 Notes on Using the Internal DC-DC Regulator and A/D Converter



CHAPTER 2 HANDLING THE DEVICE

CHAPTER 3 CPU AND CONTROL UNITS

This chapter provides basic information required to understand the functions of the FR family. It covers architecture, specifications, and instructions.

- 3.1 Memory Space
- 3.2 Internal Architecture
- 3.3 Instruction Cache
- 3.4 Dedicated Registers
- 3.5 General-Purpose Registers
- 3.6 Data Structure
- 3.7 Word Alignment
- 3.8 Memory Map
- 3.9 Branch Instructions
- 3.10 EIT (Exception, Interrupt, and Trap)
- 3.11 Reset (Device Initialization)
- 3.12 Clock Generation Control
- 3.13 Device State Control
- 3.14 Operating Modes

3.1 Memory Space

The FR family has a logical address space of 4 G bytes (2^{32} addresses), which the CPU accesses linearly.

■ Direct Addressing Area

The areas in the address space listed below are used for input-output.

These areas are called the direct addressing area. The address of an operand can be directly specified in an instruction.

The size of the direct addressing area varies according to the size of data to be accessed:

- Byte data access: 000_H to $0FF_H$
- Halfword data access: 000_H to $1FF_H$
- Word data access: 000_H to $3FF_H$

■ Memory Map

Figure 3.1-1 shows the memory map of the MB91301 series.

Figure 3.1-1 Memory Map

	(MB91302A) (Single chip mode)	(MB91302A) Internal ROM External bus mode	(MB91302A) External ROM External bus mode	(MB91V301A) Internal ROM External bus mode (MODR register at ROM=1)	(MB91 V301A) External ROM External bus mode
0000 0000H	I/O	Direct addresssing area	I/O	Direct addresssing area	I/O
0000 0400H	I/O	see "■I/O MAP" I/O	I/O	see "■I/O MAP" I/O	I/O
0001 0000H	I-RAM *1	I-RAM *1	I-RAM *1	I-RAM *1	I-RAM *1
0002 0000H	Access prohibited	Access prohibited	Access prohibited	Access prohibited	Access prohibited
0003 E000H	Internal RAM 4K bytes	Internal RAM 4K bytes	Internal RAM 4K bytes	Internal RAM 8K bytes	Internal RAM 8K bytes
0004 0000H	Access prohibited	External area	External area	Internal RAM 8K bytes	External area
0006 0000H	Access prohibited	Access prohibited	External area	Access prohibited	External area
000F E000H		Internal ROM 4K bytes*2		Internal RAM (8K bytes) ROM emulation	
0010 0000H	Access prohibited	External area	External area	External area	External area
FFFF FFFFH					

In addition to user ROM model, MB91302A has model without ROM, optimum real time OS internal model, and IPL (Internal Program Loader) model.

*1 : On specific area between 10000H and 20000H, 4-Kbyte RAM can be used. Refer to "■INSTRUCTION CACHE".

When cache function is ON, all access areas including internal ROM/RAM are object of cache.

*2 : The real time OS internal model stores the real time OS kernel. The program loader internal model stores the program loader.

Note: Internal ROM emulation: only MB91V301A

Note:

Each mode is set depending on the mode vector fetch after $\overline{\text{INIT}}$ is negated. (For mode setting, see “■MODE SETTINGS”.)

3.2 Internal Architecture

The FR family CPU is a high-performance core based on RISC architecture and advanced instructions for embedded applications.

■ Features

- **RISC architecture used**
Basic instruction: One instruction per cycle
- **32-bit architecture**
General-purpose register: 32 bits x 16
- **4 G bytes linear memory space**
- **Multiplier installed**
 - 32-bit by 32-bit multiplication: 5 cycles
 - 16-bit by 16-bit multiplication: 3 cycles
- **Enhanced interrupt processing function**
 - Quick response speed: 6 cycles
 - Support of multiple interrupts
 - Level mask function: 16 levels
- **Enhanced instructions for I/O operations**
 - Memory-to-memory transfer instruction
 - Bit-processing instructions
- **Efficient code**
Basic instruction word length: 16 bits
- **Low-power consumption**
Sleep and stop modes
- **Gear function**

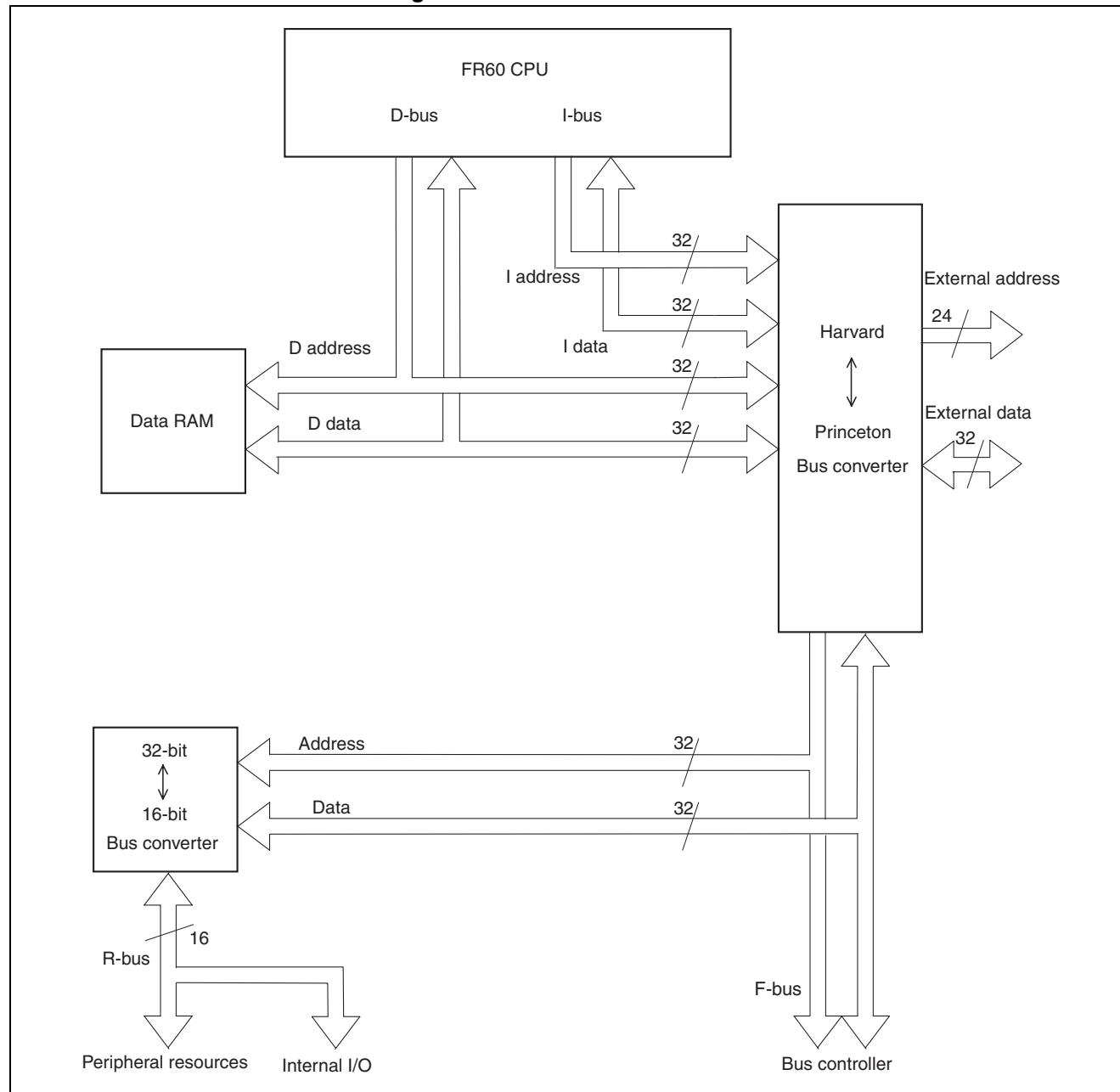
CHAPTER 3 CPU AND CONTROL UNITS

■ Internal Architecture

The FR family CPU uses the Harvard architecture, which has separate buses for instructions and data. A 32-bit/16-bit bus converter is connected to the 32-bit bus (F-bus), providing an interface between the CPU and peripheral resources. A Harvard/Princeton bus converter is connected to both the I-bus and D-bus, providing an interface between the CUP and bus controllers.

Figure 3.2-1 shows connections in the internal architecture.

Figure 3.2-1 Internal Architecture



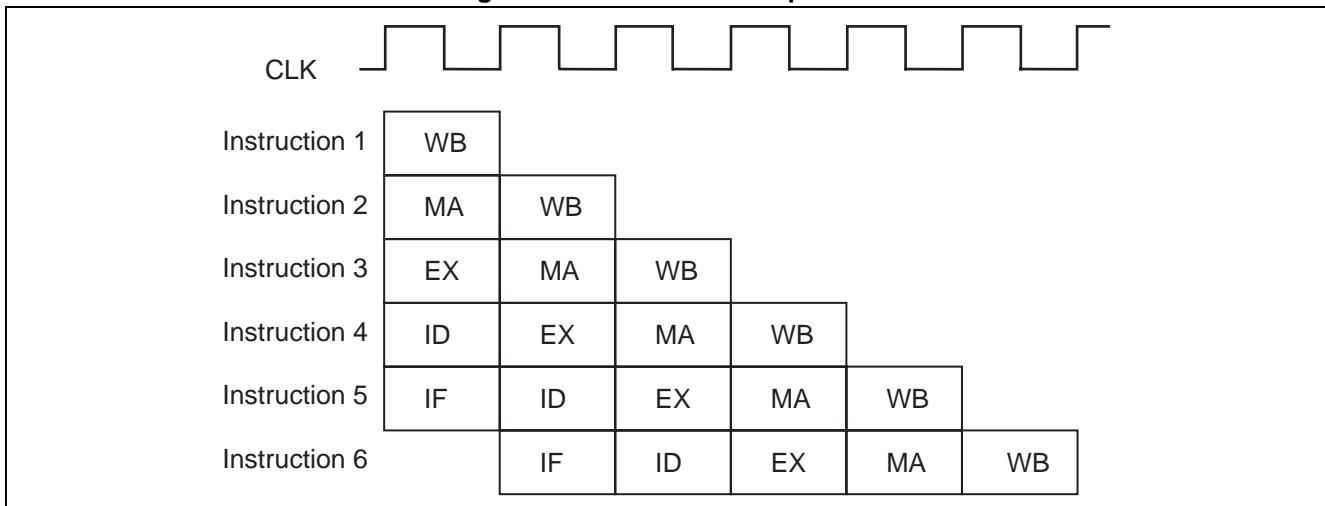
○ CPU

The CPU is a compact implementation of the 32-bit RISC FR family architecture.

Five instruction pipe lines are used to execute one instruction per cycle. A pipeline consists of the following stages:

- Instruction fetch (IF): Outputs an instruction address to fetch an instruction.
- Instruction decode (ID): Decodes a fetched instruction. Also reads a register.
- Execution (EX): Executes an arithmetic operation.
- Memory access (MA): Performs a load or store access to memory.
- Write-back (WB): Writes an operation result (or loaded memory data) to a register

Figure 3.2-2 Instruction Pipelines



Instructions are never executed randomly. If Instruction A enters a pipeline before Instruction B, it always reaches the write-back stage before Instruction B.

In general, one instruction is executed per cycle. However, multiple cycles are required to execute a load/store instruction with a memory wait, a branch instruction without a delay slot, or a multiple-cycle instruction. The execution of instructions slows down if the instructions are not supplied fast enough.

○ 32-bit/16-bit bus converter

The 32-bit/16-bit bus converter provides an interface between the F-bus accessed with 32-bit width and the R-bus accessed with 16-bit width and enables data access from the CPU to built-in peripheral circuits.

If the CPU performs one 32-bit access to the R-bus, the 32-bit/16-bit bus converter translates the access into two 16-bit accesses. Some of the built-in peripheral circuits have limitations on the access width.

○ Harvard/Princeton bus converter

The Harvard/Princeton bus converter coordinates instruction and data accesses of the CPU to provide a smooth interface between it and external buses.

The CPU has a Harvard architecture with separate buses for instructions and data. On the other hand, the bus controller that performs control of external buses has a Princeton architecture with a single bus. The Harvard/Princeton bus converter assigns priorities to instruction and data accesses from the CPU to control accesses to the bus controller. This function allows the order of external bus accesses to be permanently optimized.

CHAPTER 3 CPU AND CONTROL UNITS

■ Overview of Instructions

The FR family supports the general RISC instructions as well as logical operation, bit manipulation, and direct addressing instructions optimized for embedded applications. Each instruction is 16 bits long (some instructions 32 and 48 bits long), resulting in superior efficiency of memory use. For a list of instruction sets, see the appendix E.

An instruction set is classified into the following function groups:

- Arithmetic operation
- Load and store
- Branch
- Logical operation and bit manipulation
- Direct addressing
- Other

○ Arithmetic operation

Arithmetic operation instructions include standard arithmetic operation instructions (addition, subtraction, and comparison) and shift instructions (logical shift and arithmetic shift). The addition and subtraction instructions include an operation with carries for use with multiple-word-length operations and an operation that does not change flag values, a convenience in address calculations.

Furthermore, 32-bit-by-32-bit and 16-bit-by-16-bit multiplication instructions and a 32-bit-by-32-bit step division instruction are provided.

Additionally, an immediate data transfer instruction that sets immediate data in a register and a register-to-register transfer instruction are provided.

An arithmetic operation instruction is executed using the general-purpose registers and the multiplication and division registers in the CPU.

○ Load and store

Load and store instructions read and write to external memory. They are also used to read and write to a peripheral circuit (I/O) on the chip.

Load and store instructions have three access lengths: byte, halfword, and word. In addition to indirect memory addressing via general registers, indirect memory addressing via registers with displacement and via registers with register incrementing or decrementing are provided for some instructions.

○ Branch

The branch group includes branch, call, interrupt, and return instructions. Some branch instructions have delay slots while others do not. These may be optimized according to the application. For more information about the branch instructions, see Section "3.9 Branch Instructions".

○ Logic operation and bit manipulation

Logic operation instructions perform the AND, OR, and EOR logic operations between general-purpose registers or a general-purpose register and memory (and I/O). Bit manipulation instructions directly manipulate the contents of memory (and I/O). They access memory using general register indirect addressing.

○ Direct addressing

Direct addressing instructions are used for access between an I/O and a general-purpose register or between an I/O and the memory. High-speed and high-efficiency access can be achieved since an I/O address is directly specified in an instruction instead of using register indirect addressing. Indirect memory addressing via registers with register incrementing or decrementing are provided for some instructions.

○ Other types of instructions

Other types of instructions include instructions that provide flag setting, stack manipulation, sign/zero extension, and other functions in the PS register. Also, function entry and exit instructions that support high-level languages and register multi-load/store instructions are provided.

3.3 Instruction Cache

This section describes the instruction cache in detail.

■ Overview

The instruction cache is temporary storage memory. When low-speed external memory accesses an instruction code, the instruction cache internally stores the code already accessed once time to increase the access speed for subsequent uses.

The instruction cache data RAM enables software-based direct read access and write access when RAM mode is set. To turn the instruction cache on and then off, be sure to use the subroutine described in the precautions in Section "3.3.4 Setting up the Instruction Cache before Use".

3.3.1 Configuration of the Instruction Cache

This section describes the configuration of the instruction cache.

■ Overview of Specifications

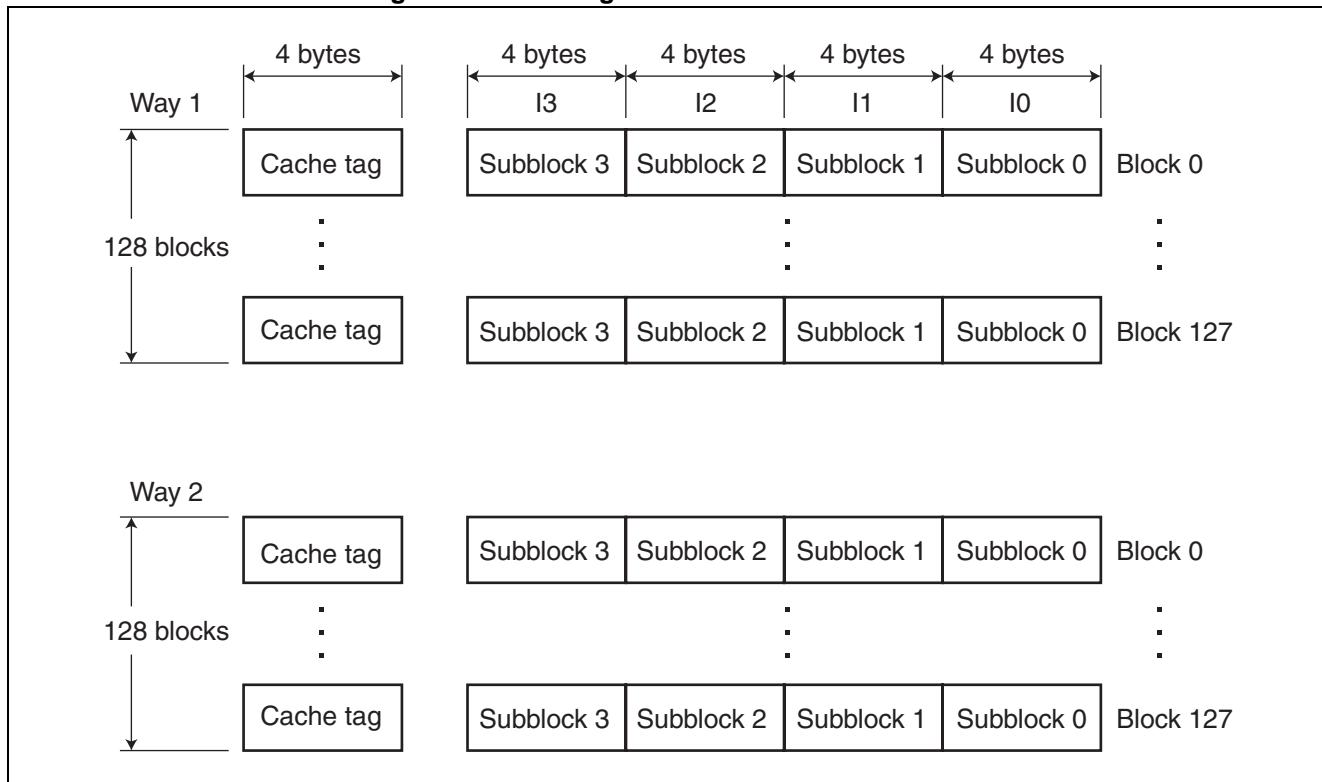
The following is an overview of the instruction cache specifications:

- FR family basic instruction length: 2 bytes
- Block layout method: 2-way set associative
- Block: One way consists of 128 blocks.
One block consists of 16 bytes (= 4 sub blocks).
One sub block consists of 4 bytes (= 1 bus access unit)

■ Configuration of Instruction Cache

Figure 3.3-1 shows the configuration of the instruction cache.

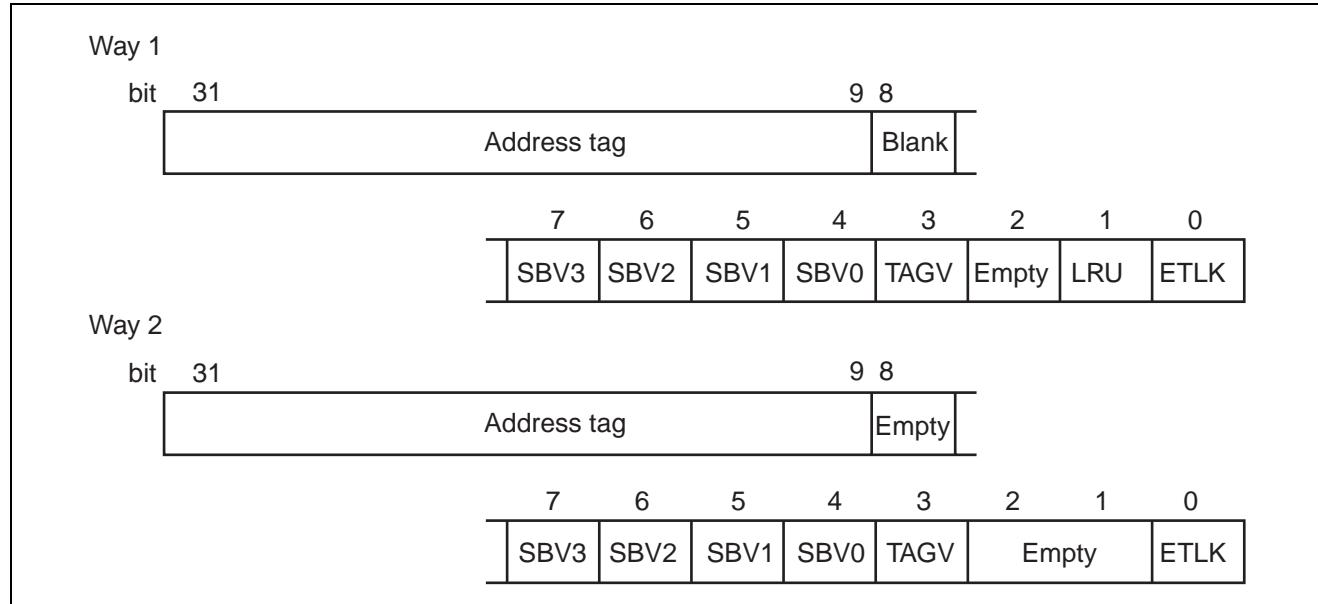
Figure 3.3-1 Configuration of Instruction Cache



■ Instruction Cache Tags

Figure 3.3-2 shows the configuration of the instruction cache tags.

Figure 3.3-2 Configuration of Instruction Cache Tags



The following describes the functions of the instruction cache tag bits.

[bit31 to bit9] Address tag

In the address tag, the high-order 23 bits of the memory address of an instruction cached in a corresponding block are stored. The instruction data stored in Sub block k of Block i has Memory Address IA, which is calculated as

$$IA = \text{address-tag} \times 2^9 + i \times 2^4 + k \times 2^2$$

The address tag is used to check the matching of an instruction address requested for the access by the CPU. Based on the result of the tag check, one of the following operations occurs:

- If the requested instruction data exists in the cache (hit)
The data is transferred from the cache to the CPU within the cycle.
- If the requested instruction data does not exist in the cache (miss)
The data acquired via external access is acquired by the CPU and the cache simultaneously.

[bit7 to bit4] Sub block valid

If SBV3 to SBV0=1, the instruction data at the address indicated by the tag has been entered in the corresponding sub block. Normally, two instructions can be stored in a sub block (except for an immediate data transfer instruction).

[bit3] TAG valid bit

Indicates whether the address tag value is valid. If this bit is "0", the block becomes invalid regardless of the sub block valid bit (when flushed).

[bit1] LRU (only for Way 1)

Exists only in the instruction cache tag of Way 1. Indicates whether, in a selected set, the entry last accessed was Way 1 or Way 2. Indicates that the last accessed entry of the set belongs to Way 1 if LRU=1 or Way 2 if LRU=0.

[bit0] ETLK (Entry lock)

Locks into the cache all the entries in the block corresponding to the tag. The entries are locked if ETLK=1 (there is no updating) if a cache miss occurs. However, invalid sub blocks are updated. If, for both Ways 1 and 2, a cache miss occurs while the entries are locked, one cycle required for the cache miss decision is lost and then external memory is accessed.

3.3.2 Configuration of the Control Registers

Control registers include the cache size register (ISIZE) and the instruction cache register (ICHCR).

This section describes the functions of these registers.

■ Configuration of Cache Size Register (ISIZE)

Figure 3.3-3 shows the configuration of the cache size register (ISIZE) bits.

Figure 3.3-3 Configuration of the Control Register (ISIZE) Bits

bit	7	6	5	4	3	2	1	0	Initial value
00000307H	-	-	-	-	-	-	SIZE1	SIZE0	-----10B

R/W R/W

The following describes the functions of the cache size register (ISIZE) bits.

[bit1, bit0] SIZE1, SIZE0

These bits set the capacity of the instruction cache. Depending on the setting, the cache size, IRAM capacity, and address map used in RAM mode vary as shown in Figure 3.3-5. If you have changed the cache capacity, be sure to flash the cache and unlock the entries before turning on the cache.

Table 3.3-1 Cache Size Registers

SIZE1	SIZE0	Capacity
0	0	1 Kbyte
0	1	2 Kbytes
1	0	4 Kbytes (Initial value)
1	1	Setting prohibited

■ Instruction Cache Control Register (ICHCR)

The instruction cache control register (ICHCR: I-Cache Control Register) controls instruction cache operation.

Writing to the ICHCR does not affect the cache operation of an instruction fetched during the subsequent three cycles.

Figure 3.3-4 shows the configuration of the instruction cache control register.

Figure 3.3-4 Configuration of Instruction Cache Control Register (ICHCR) Bits

bit	7	6	5	4	3	2	1	0	Initial value
000003E7H	RAM	-	GBLK	ALFL	EOLK	ELKR	FLSH	ENAB	0-000000B

R/W - R/W R/W R/W R/W R/W R/W

The following describes the functions of the instruction cache control register (ICHCR) bits.

[bit7] RAM (RAM mode)

If this bit is "1", RAM mode is set.

In RAM mode, set the ENAB bit to "0" to turn off the instruction cache.

[bit5] GBLK (Global lock)

This bit locks all the current entries to the instruction cache. If a miss occurs when GBLK=1, a valid entry in the instruction cache is not updated. However, invalid subblocks are updated. The instruction data fetch operation at this time is the same as when the entries are not locked.

[bit4] ALFL (Autolock fail)

This bit (ALFL) is set to "1" if locking is attempted on an entry that is already locked. If, during entry autolock, an entry update is attempted on an entry that is already locked, no new entry is locked in the instruction cache regardless of what the user intends. Reference this bit for debugging of a program or similar purpose.

Clear this bit by writing "0" to it.

[bit3] EOLK (Entry autolock)

This bit either enables or disables an autolock setting on an entry in the instruction cache. An entry accessed if this bit (EOLK) is "1" (only if a miss occurs) is locked when the hardware sets the entry lock bit in the instruction cache tag to "1". After this point, a locked entry is not subject to update when an instruction cache miss occurs. However, invalid subblocks are updated. To ensure that an entry is locked, flush the cache and set this bit.

[bit2] ELKR (Entry lock clear)

This bit specifies clearing of the entry lock bit in all the instruction cache tags. In the cycle following the one in which this bit (ELKR) is set to "1", the entry lock bit in all the cache tags is cleared to "0". However, the content of this bit is held only for one clock cycle and the bit is cleared to "0" in the second and later clock cycles.

[bit1] FLSH (Flash)

This bit specifies flushing of the instruction cache. Set this bit (FLSH) to "1" to flush the instruction cache. However, the content of this bit is held only for one clock cycle and the bit is cleared to "0" in the second and later clock cycles.

[bit0] ENAB (Enable)

This bit either enables or disables the instruction cache. If this bit (ENAB) is "0", the instruction cache is disabled and an instruction access from the CPU becomes external directly without going through the instruction cache. In the disabled state, the contents of the instruction cache are maintained.

CHAPTER 3 CPU AND CONTROL UNITS

Figure 3.3-5 Address Map of RAM

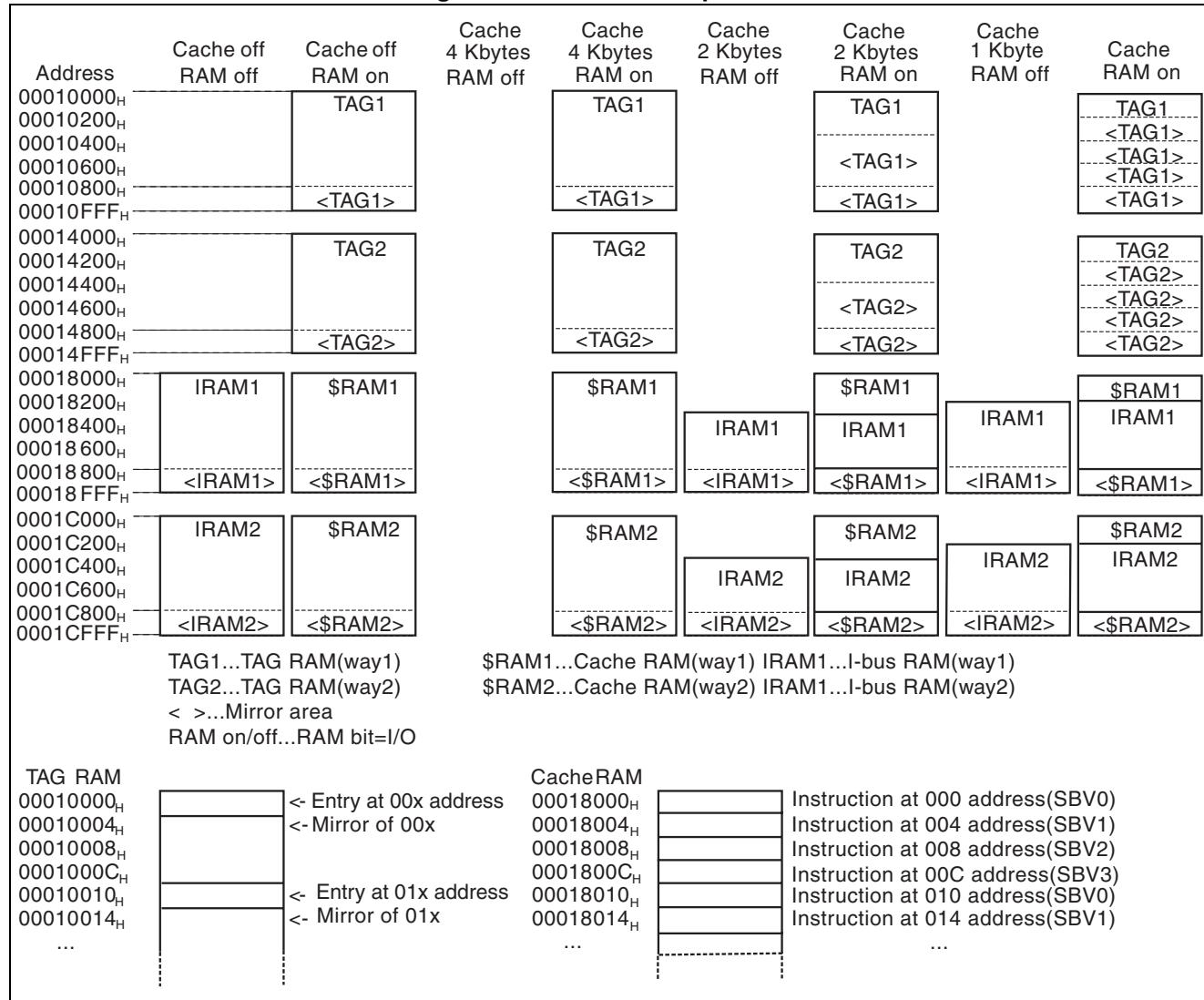


Figure 3.3-6 Memory Allocation by Cache Size

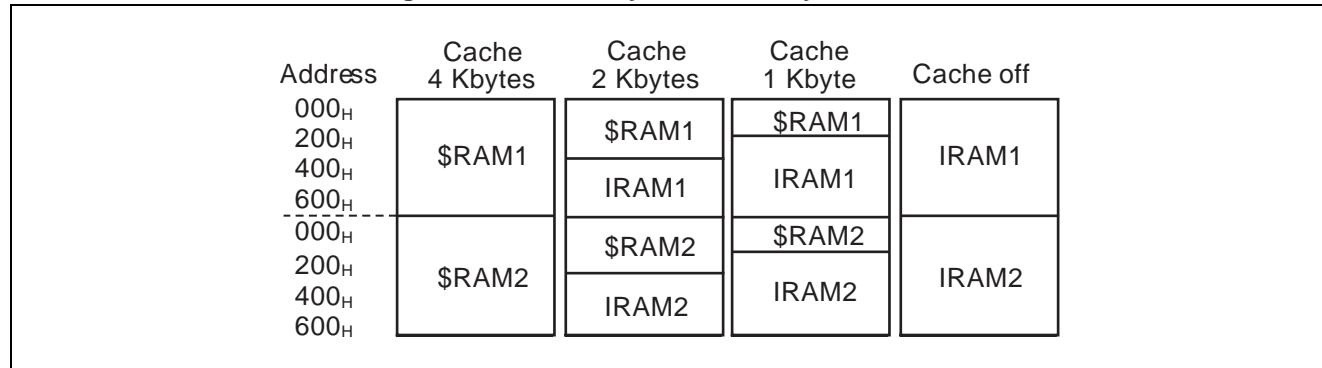
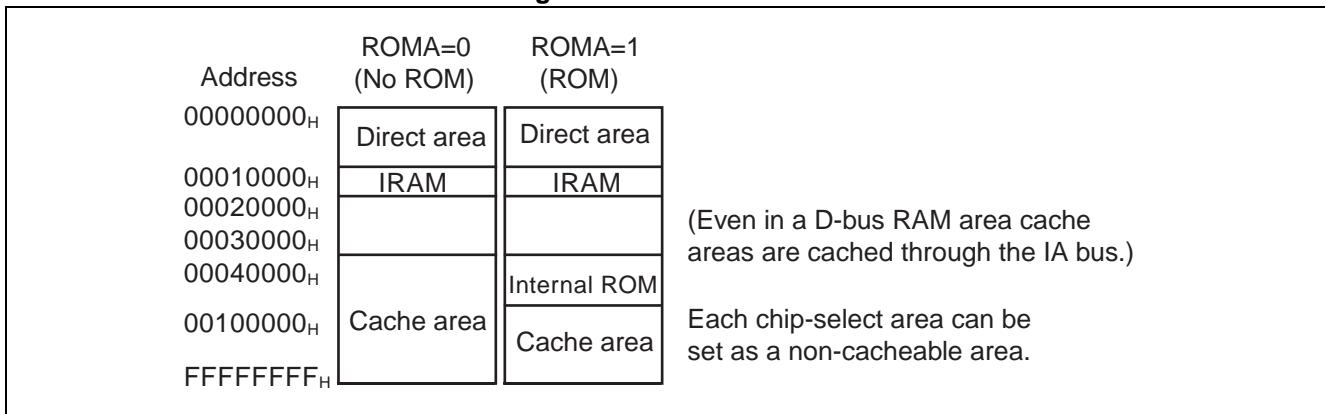


Figure 3.3-7 Cache Area



3.3.3 Instruction Cache Statuses and Settings

This section describes the state of the instruction cache in each operating modes and how to set up the instruction cache.

■ Instruction Cache Status in Each Operating Mode

Table 3.3-2 shows the state of the instruction cache in each operating mode.

The disable and flush states are encountered if a bit manipulation or similar instruction has changed only the related bit.

Table 3.3-2 Status of Instruction Cache in Each Operating Mode

		Just after reset	Disable	Flash
Cache memory	Contents undefined	Previous state maintained Not rewritable while disabled	Previous state maintained	
Tag	Address tag	Contents undefined	Previous state maintained Not rewritable while disabled	Previous state maintained
	Subblock valid bit	Contents undefined	Previous state maintained Not rewritable while disabled	Previous state maintained
	LRU	Contents undefined	Previous state maintained Not rewritable while disabled	Previous state maintained
	Entry lock bit	Contents undefined	Previous state maintained Not rewritable while disabled	Entry lock cleared
	TAG valid bit	Contents undefined	Previous state maintained Flashable while disabled	All entries invalid
RAM	Normal mode	Previous state maintained Flashable while disabled	Previous state maintained	
Control register	Global lock	Unlocked	Previous state maintained Rewritable while disabled	Previous state maintained
	Autolock fail	No fail	Previous state maintained Rewritable while disabled	Previous state maintained
	Entry autolock	Unlocked	Previous state maintained Rewritable while disabled	Previous state maintained
	Entry lock clear	No clearing	Previous state maintained Rewritable while disabled	Previous state maintained
	Enable	Disabled	Disabled	Previous state maintained
	Flash	Not flushed	Previous state maintained Rewritable while disabled	Flashed in the cycle following memory access Returned to "0" thereafter

■ Updating Entries in the Instruction Cache

Entries in the instruction cache are updated as shown in Table 3.3-3.

Table 3.3-3 Updating of Entries in the Instruction Cache

	Unlock	Lock
Hit	Not updated	Not updated
Miss	Loads the memory and updates the contents of entries in the instruction cache.	Not updated for a tag miss. Updated for subblock invalid.

■ Areas Cacheable by the Instruction Cache

- The instruction cache can cache only the internal F-bus space (RAM 8K bytes) and external bus space.
- Even if the contents of external memory are updated for DMA transfer, coherence to cached instructions is not maintained. In this case, maintain cache coherence by flushing the cache.
- Each chip select area can be set as a non-cacheable area. Setting a non-cacheable area, however, carries a penalty of one more cycle than when with the instruction cache turned off.

3.3.4 Setting up the Instruction Cache before Use

This section describes how to set up the instruction cache before it is used.

■ Setup Procedure

Before using the instruction cache, set it up as follows:

○ Initialization

Before the instruction cache is used, it must be cleared.

Set the FLSH and ELKR bits of the register to "1" to delete past data.

```
| ldi    #0x000003E7, r0    // I-Cache control register address
| ldi    #0B00000110, r1    // FLSH bit (Bit 1)
|           // ELKR bit (Bit 2)
| stb    r1, @r0          // Write to the register
```

This initializes the instruction cache.

○ Enabling the instruction cache (ON)

To enable the instruction cache, set the ENAB bit to "1".

```
| ldi    #0x000003E7, r0    // I-Cache control register address
| ldi    #0B00000001, r1    // ENAB bit (Bit 0)
| stb    r1, @r0          // Write to the register
```

Any subsequent instruction access is loaded into the instruction cache.

The instruction cache can be enabled at the same time it is initialized.

```
| ldi    #0x000003E7, r0    // I-Cache control register address
| ldi    #0B00000111, r1    // ENAB bit (Bit 0)
|           // FLSH bit (Bit 1)
|           // ELKR bit (Bit 2)
| stb    r1, @r0          // Write to the register
```

○ Disabling the instruction cache (OFF)

To disable the instruction cache, set the ENAB bit to "0".

```
ldi    #0x000003E7, r0    // I-Cache control register address
ldi    #0B00000000, r1    // ENAB bit (Bit 0)
stb    r1, @r0            // Write to the register
```

In this state maintained (which is the same as after reset), the instruction cache virtually does not exist and thus does nothing.

It may be a good idea to turn off the instruction cache if overhead seems to be a problem.

○ Locking the complete contents of the cache

Lock the instruction cache so that all the instructions it contains are removed, leaving nothing in it.

Set the GBLK bit of the register to "1". Also set the ENAB bit to "1", since otherwise the instruction cache is turned off and no locked instructions in the instruction cache are used.

```
ldi    #0x000003E7, r0    // I-Cache control register address
ldi    #0B00100001, r1    // ENAB bit (Bit 0)
                  // GBLK bit (Bit 5)
stb    r1, @r0            // Write to the register
```

○ Locking a specific instruction to the instruction cache

To lock a specific group of instructions (subroutines, etc.) to the instruction cache, set the EOLK bit to "1" before executing the instructions. A locked instruction is accessed as if it is in high-speed internal ROM.

```
ldi    #0x000003E7, r0    // I-Cache control register address  
ldi    #0B00001001, r1    // ENAB bit (Bit 0)  
                  // EOLK bit (Bit 3)  
stb    r1, @r0            // Write to the register
```

Depending on the number of memory waits, this bit becomes valid with the next or a later instruction following the stb instruction.

When locking of the group of instructions is completed, set the EOLK bit to "0".

```
ldi    #0x000003E7, r0    // I-Cache control register address  
ldi    #0B00000001, r1    // ENAB bit (Bit 0)  
                  // EOLK bit (Bit 3)  
stb    r1, @r0            // Write to the register
```

○ Clearing an instruction cache lock

Clear the lock information for an instruction locked with the EOLK bit described above.

```
ldi    #0x000003E7, r0    // I-Cache control register address  
ldi    #0B00000000, r1    // Cache disable  
stb    r1, @r0            // Write to the register  
ldi    #0B00000101, r1    // ELKR bit (Bit 2)  
stb    r1, @r0            // Write to the register
```

Only the lock information is cleared. Locked instructions are replaced sequentially with new instructions depending on the state maintained of the LRU bit.

3.4 Dedicated Registers

Use the dedicated registers for specific purposes. A program counter (PC), program status (PS), table base register (TBR), return pointer (RP), system stack pointer (SSP), user stack pointer (USP), and multiply and divide registers (MDH/MDL) are provided.

■ List of Dedicated Registers

A register consists of 32 bits.

Figure 3.4-1 shows the dedicated registers.

Figure 3.4-1 Dedicated Registers

		bit31	bit0
Program counter	PC	[]	
Program status	PS	- ILM - SCR CCR	
Table base register	TBR	[]	
Return pointer	RP	[]	
System stack pointer	SSP	[]	
User stack pointer	USP	[]	
Multiply and divide registers	MDH	[]	
	MDL	[]	

■ Program Counter (PC)

This section describes the functions of the program counter (PC: Program Counter).

The program counter (PC) consists of 32 bits. Figure 3.4-2 shows the configuration of program counter (PC) bit.

Figure 3.4-2 Configuration of Program Counter Bit

PC	bit31	bit0 [Initial value]
	[]	XXXXXXH

The program counter indicates the address of the instruction being executed.

If the PC is updated when an instruction is executed, bit0 is set to "0". Bit0 can be set to "1" only if an odd-number address is specified as the branch address.

If bit0 is set to "1", however, bit0 is invalid and an instruction must be placed at the address that is a multiple of "2".

The initial value upon reset is undefined.

CHAPTER 3 CPU AND CONTROL UNITS

■ Table Base Register (TBR)

This section describes the functions of the table base register (TBR: Table Base Register).

The table base register (TBR) consists of 32 bits. Figure 3.4-3 shows the configuration of table base register (TBR) bit.

Figure 3.4-3 Configuration of Table Base Register Bit

TBR	bit31	bit0	[Initial value]
			000FFC00H

The table base register holds the first address of the vector table to be used during EIT processing.

The initial value upon reset is 000FFC00H.

■ Return Pointer (RP)

This section describes the functions of the return pointer (RP: Return Pointer).

The return pointer (RP) consists of 32 bits. Figure 3.4-4 shows the configuration of return pointer (RP) bit.

Figure 3.4-4 Configuration of Return Pointer Bit

RP	bit31	bit0	[Initial value]
			XXXXXXXXH

The return pointer holds the return address from a subroutine.

When the CALL instruction is executed, the value of the PC is transferred to the RP.

When the RET instruction is executed, the contents of the RP are transferred to the PC.

The initial value upon reset is undefined.

■ System Stack Pointer (SSP)

This section describes the functions of the system stack pointer (SSP: System Stack Pointer).

The system stack pointer (SSP) consists of 32 bits. Figure 3.4-5 shows the configuration of system stack pointer (SSP) bit.

Figure 3.4-5 Configuration of System Stack Pointer Bit

SSP	bit31	bit0	[Initial value]
			00000000H

The SSP is the system stack pointer.

This register is used as an R15 general-purpose register if the S flag of the condition code register (CCR) is "0".

The SSP can also be specified explicitly.

This register is also used as a stack pointer that specifies a stack on which the contents of the PS and PC are to be saved if an EIT occurs.

The initial value upon reset is 00000000H.

■ User Stack Pointer (USP)

This section describes the functions of the user stack pointer (USP: User Stack Pointer).

The user stack pointer (USP) consists of 32 bits. Figure 3.4-6 shows the configuration of user stack pointer (USP) bit.

Figure 3.4-6 Configuration of User Stack Pointer Bit

USP		bit31	bit0	[Initial value]
				XXXXXXXX_H

The USP is the user stack pointer.

This register is used as an R15 general-purpose register if the S flag of the condition code register (CCR) is "1".

The USP can also be specified explicitly.

The initial value upon reset is undefined.

This register cannot be used by the RETI instruction.

■ Multiply and Divide Registers (MDH/MDL)

This section describes the functions of the multiply and divide registers (MDH/MDL: Multiply & Divide register).

The multiply and divide registers (MDH/MDL) consist of 32 bits. Figure 3.4-7 shows the configuration of multiply and divide registers (MDH/MDL) bit.

Figure 3.4-7 Configuration of Multiply and Divide Registers Bit

MDH		bit31	bit0
MDL			

MDH and MDL are the multiply and divide registers. Each register is 32 bits long.

The initial value upon reset is undefined.

○ Functions when multiplication is executed

For a 32-bit-by-32-bit multiplication, the 64-bit-long operation result is stored in the multiply and divide registers as follows:

- MDH: High-order 32 bits
- MDL: Low-order 32 bits

For a 16-bit-by-16-bit multiplication, the result is stored in one of the multiply and divide registers as follows:

- MDH: Undefined
- MDL: 32-bit result

○ Functions when division is executed

When a calculation is started, the dividend is stored in MDL.

When any of the DIV0S/DIV0U, DIV1, DIV2, DIV3, and DIV4S instructions is executed to perform division, the result is stored in MDH and MDL as follows:

- MDH: Remainder
- MDL: Quotient

3.4.1 Program Status (PS) Register

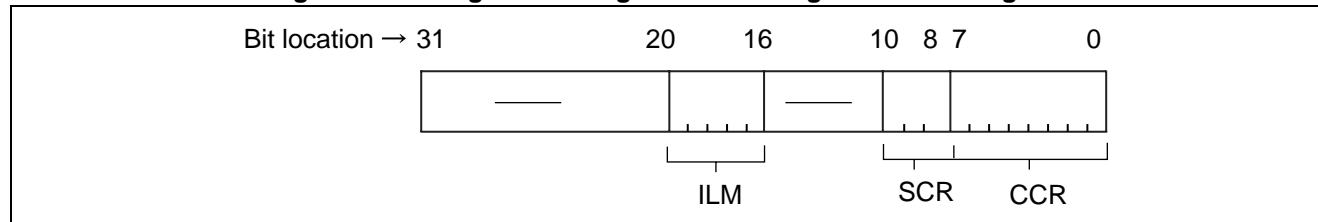
The program status register (PS: Program Status) holds the program status. The PS register consists of three parts: ILM, SCR, and CCR. All undefined bits are reserved. During reading, 0 is always read. Writing is disabled.

■ Program Status (PS) Register

○ Program Status Register (PS)

The program status (PS) register consists of the condition code register (CCR), system condition code register (SCR), and interrupt level mask (ILM) register. Figure 3.4-8 shows the register configuration of the program status register.

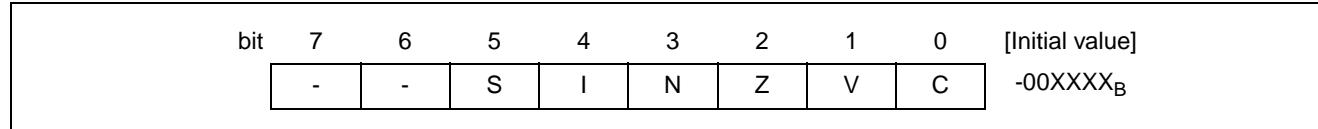
Figure 3.4-8 Register Configuration of Program Status Register



○ Condition Code Register (CCR)

Figure 3.4-9 shows the configuration of the condition code register (CCR: Condition Code Register).

Figure 3.4-9 Register Configuration of Condition Code Register



The following describes the functions of these bits.

[bit5] S (Stack flag)

This bit specifies the stack pointer to be used as general-purpose register R15.

Table 3.4-1 shows the settings of this bit.

Table 3.4-1 Functions of Stack Flag (S)

Value	Description
0	The system stack pointer (SSP) is used as general-purpose register R15. When an EIT occurs, this bit is automatically set to "0". Note that a value saved on the stack is the value before it is cleared.
1	The user stack pointer (USP) is used as general-purpose register R15.

This bit is cleared to "0" by a reset.

Set this bit to "0" when the RETI instruction is executed.

[bit4] I (Interrupt enable flag)

This bit enables or disables a user interrupt request.

Table 3.4-2 shows the settings of this bit.

Table 3.4-2 Functions of Interrupt Enable Flag (I)

Value	Description
0	User interrupts disabled. When the INT instruction is executed, this bit is cleared to "0". Note that a value saved on the stack is the value before it is cleared.
1	User interrupts enabled. The mask processing of a user interrupt request is controlled by the value held by the ILM register.

This bit is cleared to "0" by a reset.

[bit3] N (Negative flag)

This bit indicates the sign used when the operation result is handled as an integer expressed as the two's complement.

Table 3.4-3 shows the settings of this bit.

Table 3.4-3 Functions of Negative Flag (N)

Value	Description
0	Indicates that the operation result is a positive value.
1	Indicates that the operation result is a negative value.

The initial state of this bit upon reset is undefined.

[bit2] Z (Zero flag)

This bit indicates whether the operation result is "0".

Table 3.4-4 shows the settings of this bit.

Table 3.4-4 Functions of Zero Flag (Z)

Value	Description
0	Indicates that the operation result is not "0".
1	Indicates that the operation result is "0".

The initial state of this bit upon reset is undefined.

[bit1] V (Overflow flag)

This bit indicates whether an overflow has occurred as a result of the operation when the operand used in the operation is handled as an integer expressed as the two's complement.

Table 3.4-5 shows the settings of this bit.

Table 3.4-5 Functions of Overflow Flag (V)

Value	Description
0	Indicates that no overflow has occurred as a result of the operation.
1	Indicates that an overflow has occurred as a result of the operation.

The initial state of this bit upon reset is undefined.

[bit0] C (Carry flag)

This bit indicates whether a carry or a borrow has occurred from the highest bit in the operation.

Table 3.4-6 shows the settings of this bit.

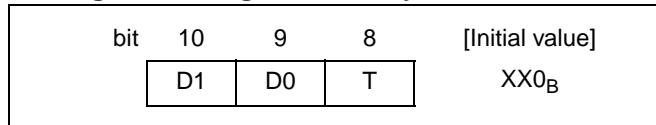
Table 3.4-6 Functions of Carry Flag (C)

Value	Description
0	Indicates that no carry and borrow have occurred.
1	Indicates that a carry or borrow has occurred.

The initial state of this bit upon reset is undefined.

○ System condition code register (SCR)

Figure 3.4-10 shows the configuration of the system condition code register (SCR: System Condition code Register).

Figure 3.4-10 Register Configuration of System Condition Code Register

The following describes the functions of the system condition code register (SCR) bits.

[bit10, bit9] D1, D0 (Step division flag)

These bits hold the intermediate data obtained when step division is executed.

Do not change these bits while division processing is being executed.

To perform other processing while executing a step division, save and restore the value of the PS register to ensure that the step division is restarted.

The initial state of this bit upon reset is undefined.

To set these bits, execute the DIV0S instruction with the dividend and the divisor to be referenced.

To forcibly clear these bits, execute the DIV0U instruction.

- Simultaneous acceptance of DIV0S/DIV0U instruction, user interrupt and NMI
Do not perform the process expecting the D0 and D1 bits in the PS register before EIT branching in the EIT processing routine.
- The D0 and D1 bits in the PS register may not indicate the correct value if the operation is stopped by break or step execution immediately before the DIV0S/DIV0U instruction.

Note, however, that the correct value will be calculated after return.

[bit8] T (Step trace trap flag)

This bit specifies whether the step trace trap is to be enabled.

Table 3.4-7 shows the settings of this bit.

Table 3.4-7 Functions of Step Trace Trap Flag (T)

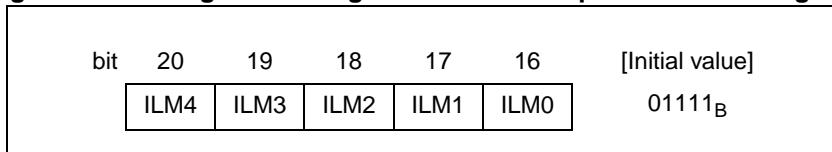
Value	Description
0	The step trace trap is disabled.
1	The step trace trap is enabled. With this setting, all the user NMI and user interrupts are prohibited.

This bit is initialized to "0" by a reset.

The step trace trap function is used by an emulator. When an emulator is used, this function cannot be used in a user program.

- **Interrupt level mask (ILM) register**

Figure 3.4-11 shows the configuration of the interrupt level mask (ILM) register.

Figure 3.4-11 Register Configuration of Interrupt Level Mask Register

The interrupt level mask (ILM) register holds an interrupt level mask value. The value held in ILM register is used as a level mask.

The CPU accepts only interrupt requests sent to it with an interrupt level higher than the level indicated by the ILM.

The highest level is 0 (00000_B) and the lowest level is 31 (11111_B).

Values that can be set by a program have a limit. If the original value is between 16 and 31, the new value must be between 16 and 31. If an instruction that sets a value between 0 and 15 is executed, the specified value plus 16 is transferred.

If the original value is between 0 and 15, an arbitrary value between 0 and 31 may be set.

This register is initialized to 15 (01111_B) by a reset.

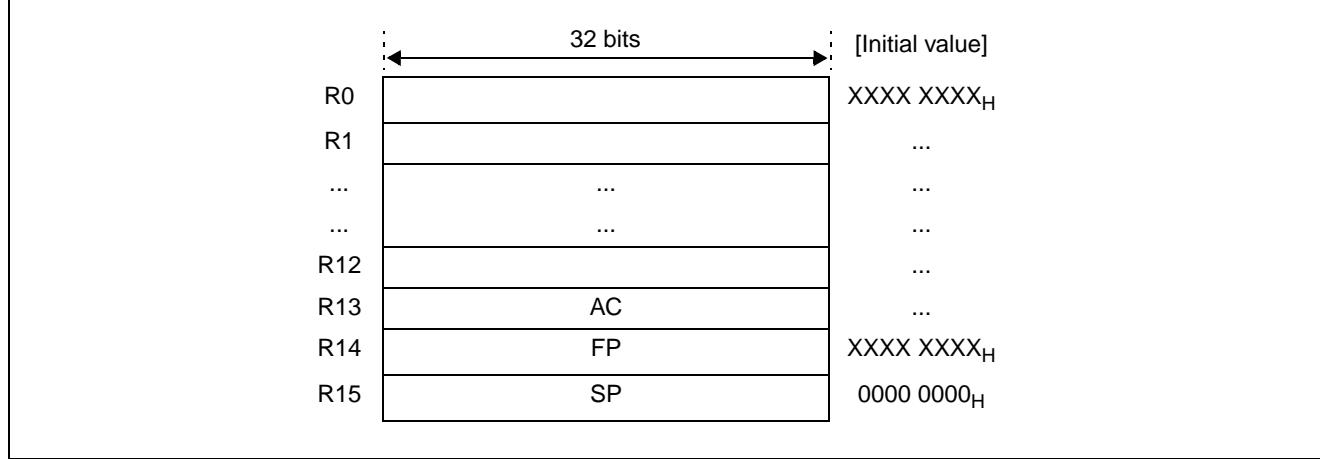
3.5 General-Purpose Registers

Registers R0 to R15 are general-purpose registers. These registers are used as an accumulator in an operation or a pointer in a memory access.

■ General-purpose Registers

Figure 3.5-1 shows the configuration of the general-purpose registers.

Figure 3.5-1 Configuration of General-purpose Registers



Of these 16 registers, the following are intended for special applications and therefore enhanced instructions are provided for them:

- R13: Virtual accumulator (AC)
- R14: Frame pointer (FP)
- R15: Stack pointer (SP)

The initial value upon reset is undefined for R0 through R14 and is 00000000_H (SSP value) for R15.

3.6 Data Structure

The FR family uses the following two data ordering methods:

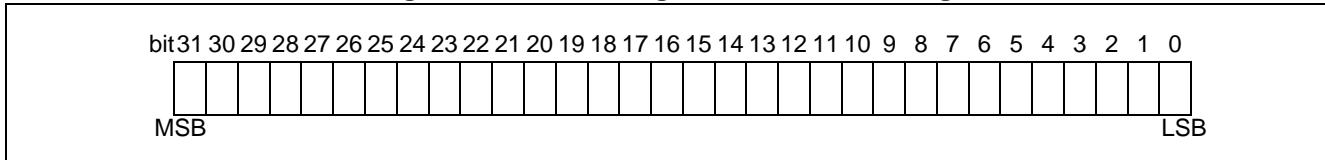
- Bit ordering
- Byte ordering

■ Bit Ordering

The FR family uses the little endian method for bit ordering.

Figure 3.6-1 shows the bit configuration in bit ordering.

Figure 3.6-1 Bit Configuration in Bit Ordering

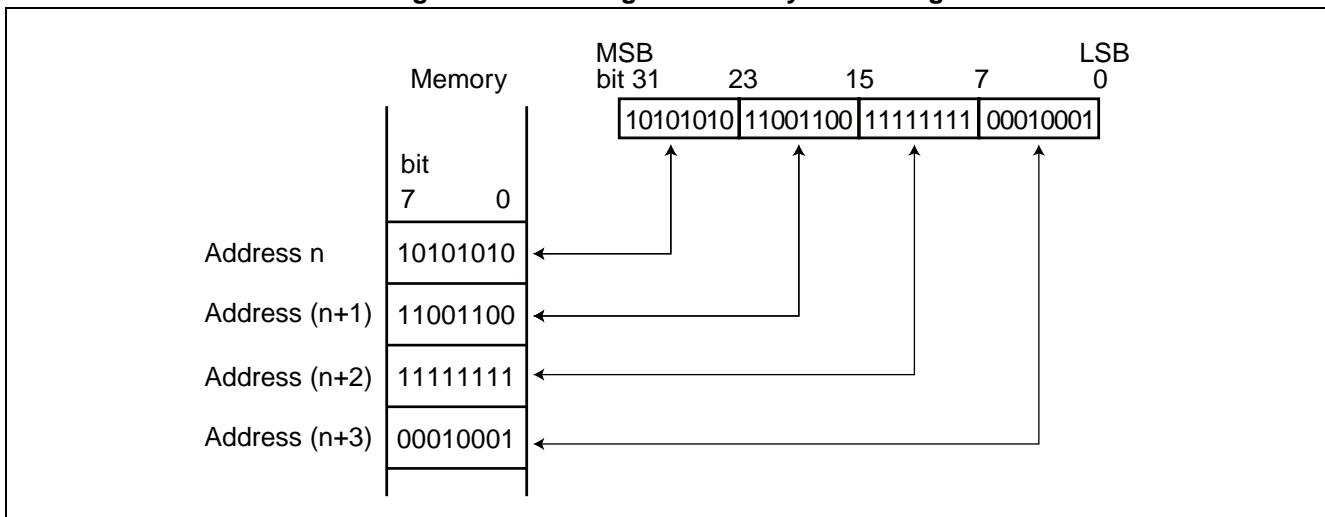


■ Byte Ordering

The FR family uses the big endian method for byte ordering.

Figure 3.6-2 shows the configuration of byte ordering.

Figure 3.6-2 Configuration of Byte Ordering



3.7 Word Alignment

Since instructions and data are accessed in byte units, the addresses at which they are placed depend on the instruction length or the data width.

■ Program Access

A program for the FR family must be placed at an address that is a multiple of "2".

Bit0 of the program counter (PC) is set to "0" if the PC is updated when an instruction is executed.

Bit0 can be set to "1" only if an odd-number address is specified as the branch address.

If bit0 is set to "1", however, bit0 is invalid and an instruction must be placed at the address that is a multiple of "2".

No odd-number address exception exists.

■ Data Access

If data in the FR family is accessed, forced alignment is applied to the address based on the width.

- Word access: An address must be a multiple of "4". (The lowest-order 2 bits are forcibly set to "00".)
- Halfword access: An address must be a multiple of "2". (The lowest-order bit is forcibly set to "0".)
- Byte access: -

During word or halfword data access, some of the bits in the result of calculating an effective address are forcibly set to "0". For example, in @(R13, R2) addressing mode, the register before addition is used without change in the calculation (even if the lowest-order bit is "1") and the low-order bits are masked. A register before calculation is not masked.

[Example] LD @(R13, R2), R0

R13	00002222H
R2	00000003H
+)	
Addition result	00002225H
	↓ Lower 2 bits forcibly masked
Address pin	00002224H

3.8 Memory Map

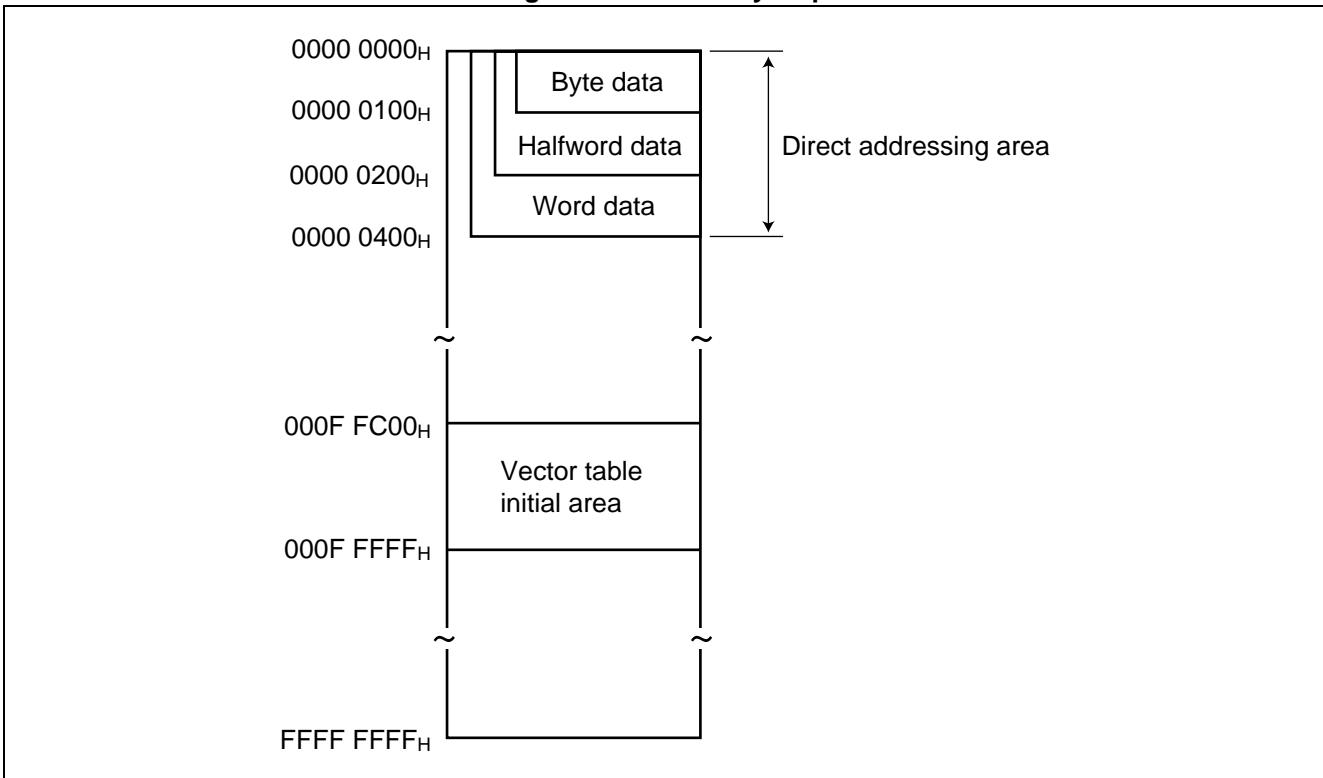
This section shows the memory map for the FR family.

■ Memory Map

The address space of memory is 32 bits linear.

Figure 3.8-1 shows the memory map.

Figure 3.8-1 Memory Map



○ Direct addressing area

The following areas in the address space are the areas for I/O. When direct addressing is used in these areas, an operand address can be directly specified in an instruction.

The size of an address area for which an address can be directly specified varies is determined by the data length as follows:

- Byte data (8 bits): 000_H to 0FF_H
- Halfword data (16 bits): 000_H to 1FF_H
- Word data (32 bits): 000_H to 3FF_H

○Vector table initial area

The area from 000FFC00_H to 000FFFFF_H is the initial EIT vector table area.

You can place the vector table that will be used during EIT processing at any address by rewriting the TBR. Initialization by a reset places the table at this address.

3.9 Branch Instructions

An operation with or without a delay slot can be specified for a branch instruction used in the FR family.

■ Branch Instructions with Delay Slot

Instructions written as follows perform a branch operation with a delay slot:

JMP:D	@Ri	CALL:D	label12	CALL:D	@Ri	RET:D
BRA:D	label9	BNO:D	label9	BEQ:D	label9	BNE:D
BC:D	label9	BNC:D	label9	BN:D	label9	BP:D
BV:D	label9	BNV:D	label9	BLT:D	label9	BGE:D
BLE:D	label9	BGT:D	label9	BLS:D	label9	BHI:D

3.9.1 Operation of Branch Instructions with Delay Slot

In operation with a delay slot, the instruction located just after a branch instruction (placed in a "delay slot") is executed before the instruction that branches is executed.

■ Operation of Branch Instruction with Delay Slot

Since an instruction in the delay slot is executed before the branch operation, the apparent execution speed is one cycle. However, a NOP instruction must be placed in the delay slot if there is no valid instruction put there.

[Example]

```
; List of instructions
    ADD      R1,     R2      ;
    BRA:D   LABEL    ; Branch instruction
    MOV      R2,     R3      ; Delay slot ... Executed before branch
    ...
LABEL : ST       R3,     @R4      ; Branch destination
```

If a conditional branch instruction is used, an instruction placed in the delay slot is executed whether or not the condition for branching is met.

If a delay branch instruction is used, the order of execution for some instructions seems to be reversed. However, this occurs only for updating the PC and the instructions are executed in the specified order for other operations (register update and reference, etc.)

The following is a concrete example.

○ JMP:D @Ri / CALL:D @Ri instruction

Ri referenced by the JMP:D @Ri / CALL:D @Ri instruction is not affected even though Ri is updated by the instruction in the delay slot.

[Example]

```
LDI:32   #Label,   R0      ;
JMP:D    @R0      ; Branch to Label
LDI:8    #0,       R0      ; No effect on the branch destination address
...
```

CHAPTER 3 CPU AND CONTROL UNITS

○ RET:D instruction

RP referenced by the RET:D instruction is not affected even though RP is updated by the instruction in the delay slot.

[Example]

```
RET:D          ; Branch to address defined beforehand in RP
MOV    R8,    RP  ; No effect on the return operation
...
...
```

○ Bcc:D rel instruction

The flag referenced by the Bcc:D rel instruction is not affected by the instruction in the delay slot.

[Example]

```
ADD    #1,    R0  ; Flag change
BC:D    Overflow   ; Branch to execution result of above
                    ; instruction
AND    CCR     #0  ; This flag update is not referenced by the
                    ; above branch instruction.
...
...
```

○ CALL:D instruction

If RP is referenced by an instruction in the delay slot of the CALL:D instruction, the data that has been updated by the CALL:D instruction is read.

[Example]

```
CALL:D  Label      ; Updating RP and branching
MOV    RP,    R0  ; Transferring RP, execution result of above
                  ; CALL:D
...
...
```

■ Limitations on Branch Instruction with Delay Slot**○ Instructions that can be placed in the delay slot**

Only an instruction meeting the following conditions can be executed in the delay slot.

- One-cycle instruction
- Instruction other than a branch instruction
- Instruction whose operation is not affected even though the order is changed

A one-cycle instruction is an instruction denoted in the Number of Cycles column in the list of instructions as 1, a, b, c, and d.

○ Step trace trap

A step trace trap does not occur between the execution of a branch instruction with a delay slot and the delay slot.

○ Interrupt/NMI

An interrupt/NMI is not accepted between the execution of a branch instruction with a delay slot and the delay slot.

○ Undefined instruction exception

An undefined instruction exception does not occur if there is an undefined instruction in the delay slot. If an undefined instruction is in the delay slot, it operates as a NOP instruction.

3.9.2 Operation of Branch Instruction without Delay Slot

In operation without a delay slot, instructions are executed in the order in which they are specified. An instruction immediately following a branch is never executed before it.

■ Branch Instructions without Delay Slot

Instructions written as follows perform a branch operation without a delay slot:

JMP	@Ri	CALL	label12	CALL	@Ri	RET	
BRA	label9	BNO	label9	BEQ	label9	BNE	label9
BC	label9	BNC	label9	BN	label9	BP	label9
BV	label9	BNV	label9	BLT	label9	BGE	label9
BLE	label9	BGT	label9	BLS	label9	BHI	label9

■ Operation of Branch Instruction without Delay Slot

In operation without a delay slot, instructions are executed in the order in which they are specified. An instruction immediately following a branch is never executed before it.

[Example]

```
;           List of instructions
      ADD  R1,   R2      ;
      BRA  LABEL       ; Branch instruction (without a delay slot)
      MOV  R2,   R3      ; Not executed
      ...
LABEL : ST   R3,  @R4    ; Branch destination
```

A branch instruction without a delay slot is executed in two cycles if a branch occurs and in one cycle if no branch occurs.

Since no appropriate instruction can be placed in the delay slot, this instruction results in a more efficient instruction code than a branch instruction with a delay slot and with NOP specified.

For both optimum execution speed and code efficiency, select an operation with a delay slot if a valid instruction can be placed in the delay slot; otherwise, select an operation without a delay slot.

3.10 EIT (Exception, Interrupt, and Trap)

EIT, a generic term for exception, interrupt, and trap, refers to suspending program execution if an event occurs during execution and then executing another program.

■ EIT (Exception, Interrupt, and Trap)

An exception is an event that occurs related to the execution context. Execution restarts from the instruction that caused the exception.

An interrupt is an event that occurs independently of execution context. The event is caused by hardware.

A trap is an event that occurs related to the execution context. Some traps, such as system calls, are specified in a program. Execution restarts from the instruction following the one that caused the trap.

■ Features

- Multiple interrupt is supported to the interruption.
- It is a level mask function (15 levels are available to the user) to the interruption.
- Trap instruction (INT)
- EIT (hardware/software) for emulator startup

■ EIT Causes

The following are causes of EIT:

- Reset
- User interrupt (internal resource, external interrupt)
- NMI
- Delayed interrupt
- Undefined instruction exception
- Trap instruction (INT)
- Trap instruction (INTE)
- Step trace trap
- No-coprocessor trap
- Coprocessor error trap

■ Return from EIT

Use the RETI instruction to return from EIT.

3.10.1 EIT Interrupt Levels

The interrupt levels are 0 to 31 and are managed with five bits.

■ EIT Interrupt Levels

Table 3.10-1 shows the allocation of the levels.

Table 3.10-1 EIT Interrupt Levels

Level		Interrupt source	Note
Binary	Decimal		
00000	0	(Reserved for system)	
...	
...	
00011	3	(Reserved for system)	If the original ILM value is between 16 and 31, a program cannot set a value in this ILM range.
00100	4	INTE instruction Step trace trap	
00101	5	(Reserved for system)	
...	
...	
01110	14	(Reserved for system)	
01111	15	NMI (for user)	
10000	16	Interrupt	User interrupts prohibited if ILM is set
10001	17	Interrupt	
...	
...	
11110	30	Interrupt	
11111	31	-	Interrupts prohibited if ICR is set

Operation is possible for levels 16 to 31.

The interrupt level does not affect an undefined instruction exception, no-coprocessor trap, coprocessor error trap, or an INT instruction. It does not change the ILM, either.

■ I Flag

A flag that specifies whether an interrupt is permitted or prohibited. This flag is provided as bit4 of the PS register.

Table 3.10-2 shows the functions of I flag.

Table 3.10-2 Functions of I Flag

Value	Description
0	Interrupts prohibited Cleared to "0" if the INT instruction is executed. Note that a value saved on the stack is the value before it is cleared.
1	Interrupts permitted The mask processing of an interrupt request is controlled by the value in the ILM register.

■ Interrupt Level Mask (ILM) Register

A PS register (bit20 to bit16) that holds an interrupt level mask value.

The CPU accepts only an interrupt request sent to it with an interrupt level higher than the level indicated by the ILM.

The highest level is 0 (00000_B) and the lowest level is 31 (11111_B).

Values that can be set by a program have a limit. If the original value is between 16 and 31, the new value must be between 16 and 31. If an instruction that sets a value between 0 and 15 is executed, the specified value plus 16 is transferred.

If the original value is between 0 and 15, any value between 0 and 31 may be set.

Note:

Use the ST ILM instruction to set this register.

■ Level Mask for Interrupt and NMI

If an NMI or interrupt request occurs, the interrupt level (Table 3.10-1) of the interrupt source is compared with the level mask value held in the ILM. A request meeting the following condition is masked and is not accepted:

Interrupt level of cause \geq Level mask value

3.10.2 Interrupt Control Register (ICR)

The interrupt control register (ICR: Interrupt Control Register), located in the interrupt controller, sets the level of an interrupt request. An ICR is provided for each of the interrupt request inputs. The ICR is mapped on the I/O space and is accessed from the CPU through a bus.

■ Configuration of Interrupt Control Register (ICR)

Figure 3.10-1 shows the configuration of the interrupt control register (ICR) bits.

Figure 3.10-1 Configuration of Interrupt Control Register Bit

Address	bit	7	6	5	4	3	2	1	0	[Initial value]
000440 _H to 00046F _H		-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	---111111 _B
		R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The following describes the functions of the interrupt control register (ICR) bits.

[bit4] ICR4

This bit is always set to "1".

[bit3 to bit0] ICR3 to ICR0

These bits are the low-order 4 bits of the interrupt level of the corresponding interrupt source. They can be read and written to.

Together with bit4, a value between 16 and 31 can be set in the ICR.

■ Mapping of Interrupt Control Register (ICR)

Table 3.10-3 shows the relationship between interrupt sources, interrupt control register, and interrupt vectors.

Table 3.10-3 Interrupt Sources, Interrupt Control Registers, and Interrupt Vectors

Interrupt source	Interrupt control register		Corresponding interrupt vector		
			Number		Address
			Hexadecimal	Decimal	
IRQ00	ICR00	00000440 _H	10 _H	16	TBR + 3BC _H
IRQ01	ICR01	00000441 _H	11 _H	17	TBR + 3B8 _H
IRQ02	ICR02	00000442 _H	12 _H	18	TBR + 3B4 _H
...
...
IRQ45	ICR45	0000046D _H	3D _H	61	TBR + 308 _H
IRQ46	ICR46	0000046E _H	3E _H	62	TBR + 304 _H
IRQ47	ICR47	0000046F _H	3F _H	63	TBR + 300 _H

TBR initial value: 00F FC00_H

Note: See "CHAPTER 11 INTERRUPT CONTROLLER".

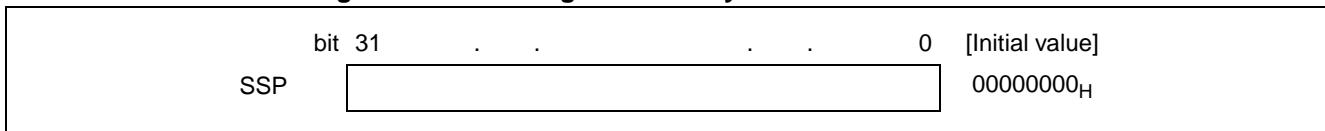
3.10.3 System Stack Pointer (SSP)

The system stack pointer (SSP) is used to point to the stack to save and restore data when EIT is accepted or a return operation occurs.

■ System Stack Pointer (SSP)

The system stack pointer (SSP: System Stack Pointer) consists of 32 bits. Figure 3.10-2 shows the configuration of system stack pointer (SSP) bits.

Figure 3.10-2 Configuration of System Stack Pointer Bit



Eight is subtracted from the register value during EIT processing and eight is added to the register value during the return operation from EIT that occurs when the RETI instruction is executed.

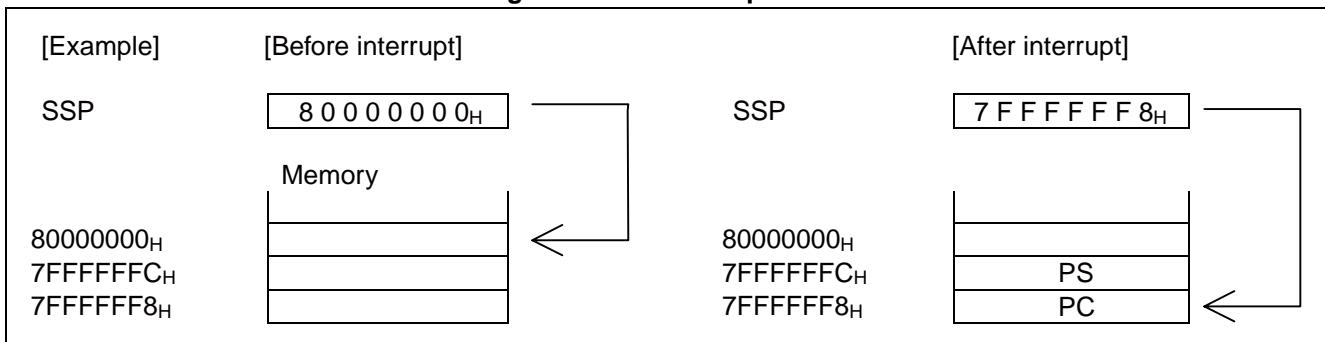
The system stack pointer (SSP) is initialized to 00000000H by a reset.

The SSP is also used as general-purpose register R15 if the S flag in the CCR is set to "0".

■ Interrupt Stack

The value in the PC or PS is saved to or restored from an area pointed to by the system stack pointer (SSP). After an interrupt occurs, the PC is stored at the address indicated by the SSP and the PS is stored at the address indicated by the SSP plus 4. This situation is shown in Figure 3.10-3.

Figure 3.10-3 Interrupt Stack



3.10.4 Table Base Register (TBR)

The table base register (TBR: Table Base Register) indicates the beginning address of the vector table for EIT.

■ Table Base Register (TBR)

The table base register (TBR) consists of 32 bits.

Figure 3.10-4 shows the configuration of table base register (TBR) bits.

Figure 3.10-4 Configuration of Table Base Register Bit

bit 31		0	[Initial value]
TBR			000FFC00 _H

Obtain a vector address by adding to the TBR the offset value predetermined for an EIT cause.

The table base register (TBR) is initialized to 000FFC00_H by a reset.

■ EIT Vector Table

A 1 Kbyte area from the address indicated in the table base register (TBR) is the vector area for EIT.

The size for each vector is 4 bytes. The relationship between a vector number and a vector address can be expressed as follows:

$$\begin{aligned} \text{vctadr} &= \text{TBR} + \text{vctofs} \\ &= \text{TBR} + (3\text{FC}_H - 4 \times \text{vct}) \end{aligned}$$

vctadr: Vector address

vctofs: Vector offset

vct: Vector number

The low-order two bits of the addition result are always handled as "00_B".

The area from 000FFC00_H to 000FFFFF_H is the initial area for the vector table upon reset.

Special functions are allocated to some of the vectors.

Table 3.10-4 shows the vector table on the architecture.

Table 3.10-4 Vector Table (1 / 3)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
Reset	0	00	-	3FC _H	000FFFFC _H	-
Mode vector	1	01	-	3F8 _H	000FFFF8 _H	-
Reserved for system	2	02	-	3F4 _H	000FFFF4 _H	-
Reserved for system	3	03	-	3F0 _H	000FFFF0 _H	-
Reserved for system	4	04	-	3EC _H	000FFFEC _H	-
Reserved for system	5	05	-	3E8 _H	000FFFE8 _H	-
Reserved for system	6	06	-	3E4 _H	000FFFE4 _H	-
No-coprocessor trap	7	07	-	3E0 _H	000FFFE0 _H	-
Coprocessor error trap	8	08	-	3DC _H	000FFFDC _H	-
INTE instruction	9	09	-	3D8 _H	000FFFD8 _H	-
Reserved for system	10	0A	-	3D4 _H	000FFFD4 _H	-
Reserved for system	11	0B	-	3D0 _H	000FFFD0 _H	-
Step trace trap	12	0C	-	3CC _H	000FFFCC _H	-
NMI request (tool)	13	0D	-	3C8 _H	000FFFC8 _H	-
Undefined instruction exception	14	0E	-	3C4 _H	000FFFC4 _H	-
NMI request	15	0F	Fixed to 15(F _H)	3C0 _H	000FFFC0 _H	-
External Interrupt 0	16	10	ICR00	3BC _H	000FFFBC _H	6
External Interrupt 1	17	11	ICR01	3B8 _H	000FFF8B _H	7
External Interrupt 2	18	12	ICR02	3B4 _H	000FFF84 _H	11
External Interrupt 3	19	13	ICR03	3B0 _H	000FFF80 _H	12
External Interrupt 4	20	14	ICR04	3AC _H	000FFFAC _H	-
External Interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H	-
External Interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H	-
External Interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H	-
Reload Timer 0	24	18	ICR08	39C _H	000FFF9C _H	8
Reload Timer 1	25	19	ICR09	398 _H	000FFF98 _H	9
Reload Timer 2	26	1A	ICR10	394 _H	000FFF94 _H	10
UART0 (reception completed)	27	1B	ICR11	390 _H	000FFF90 _H	0
UART1 (reception completed)	28	1C	ICR12	38C _H	000FFF8C _H	1

CHAPTER 3 CPU AND CONTROL UNITS

Table 3.10-4 Vector Table (2 / 3)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
UART2 (reception completed)	29	1D	ICR13	388 _H	000FFF88 _H	2
UART0 (transmission completed)	30	1E	ICR14	384 _H	000FFF84 _H	3
UART1 (transmission completed)	31	1F	ICR15	380 _H	000FFF80 _H	4
UART2 (transmission completed)	32	20	ICR16	37C _H	000FFF7C _H	5
DMAC0 (end, error)	33	21	ICR17	378 _H	000FFF78 _H	-
DMAC1 (end, error)	34	22	ICR18	374 _H	000FFF74 _H	-
DMAC2 (end, error)	35	23	ICR19	370 _H	000FFF70 _H	-
DMAC3 (end, error)	36	24	ICR20	36C _H	000FFF6C _H	-
DMAC4 (end, error)	37	25	ICR21	368 _H	000FFF68 _H	-
A/D	38	26	ICR22	364 _H	000FFF64 _H	15
PPG0	39	27	ICR23	360 _H	000FFF60 _H	13
PPG1	40	28	ICR24	35C _H	000FFF5C _H	14
PPG2	41	29	ICR25	358 _H	000FFF58 _H	-
PPG3	42	2A	ICR26	354 _H	000FFF54 _H	-
Reserved for system	43	2B	ICR27	350 _H	000FFF50 _H	-
U-TIMER0	44	2C	ICR28	34C _H	000FFF4C _H	-
U-TIMER1	45	2D	ICR29	348 _H	000FFF48 _H	-
U-TIMER2	46	2E	ICR30	344 _H	000FFF44 _H	-
Time-base timer overflow	47	2F	ICR31	340 _H	000FFF40 _H	-
I ² C I/F0	48	30	ICR32	33C _H	000FFF3C _H	-
I ² C I/F1	49	31	ICR33	338 _H	000FFF38 _H	-
Reserved for system	50	32	ICR34	334 _H	000FFF34 _H	-
Reserved for system	51	33	ICR35	330 _H	000FFF30 _H	-
16bit free run timer	52	34	ICR36	32C _H	000FFF2C _H	-
ICU0 (fetch)	53	35	ICR37	328 _H	000FFF28 _H	-
ICU1 (fetch)	54	36	ICR38	324 _H	000FFF24 _H	-
ICU2 (fetch)	55	37	ICR39	320 _H	000FFF20 _H	-
ICU3 (fetch)	56	38	ICR40	31C _H	000FFF1C _H	-
Reserved for system	57	39	ICR41	318 _H	000FFF18 _H	-
Reserved for system	58	3A	ICR42	314 _H	000FFF14 _H	-
Reserved for system	59	3B	ICR43	310 _H	000FFF10 _H	-

Table 3.10-4 Vector Table (3 / 3)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
Reserved for system	60	3C	ICR44	30C _H	000FFF0C _H	-
Reserved for system	61	3D	ICR45	308 _H	000FFF08 _H	-
Reserved for system	62	3E	ICR46	304 _H	000FFF04 _H	-
Delayed interrupt source bit	63	3F	ICR47	300 _H	000FFF00 _H	-
Reserved for system (used in REALOS)	64	40	-	2FC _H	000FFEFC _H	-
Reserved for system (used in REALOS)	65	41	-	2F8 _H	000FFEF8 _H	-
Reserved for system	66	42	-	2F4 _H	000FFEF4 _H	-
Reserved for system	67	43	-	2F0 _H	000FFEF0 _H	-
Reserved for system	68	44	-	2EC _H	000FFEEC _H	-
Reserved for system	69	45	-	2E8 _H	000FFEE8 _H	-
Reserved for system	70	46	-	2E4 _H	000FFEE4 _H	-
Reserved for system	71	47	-	2E0 _H	000FFEE0 _H	-
Reserved for system	72	48	-	2DC _H	000FFEDC _H	-
Reserved for system	73	49	-	2D8 _H	000FFED8 _H	-
Reserved for system	74	4A	-	2D4 _H	000FFED4 _H	-
Reserved for system	75	4B	-	2D0 _H	000FFED0 _H	-
Reserved for system	76	4C	-	2CC _H	000FFECC _H	-
Reserved for system	77	4D	-	2C8 _H	000FFEC8 _H	-
Reserved for system	78	4E	-	2C4 _H	000FFEC4 _H	-
Reserved for system	79	4F	-	2C0 _H	000FFEC0 _H	-
Used in INT instruction	80 to 255	50 to FF	-	2BC _H to 000 _H	000FFEB _H to 000FFC00 _H	-

3.10.5 Multiple EIT Processing

If multiple EIT causes occur at the same time, the CPU repeats the operation of selecting and accepting one of the EIT causes, executing the EIT sequence, and then detecting EIT causes again. If there are no more EIT causes to be accepted while the CPU is detecting EIT causes, the CPU executes the handler instruction of the last accepted EIT cause. As a result, the order of executing handlers for multiple EIT causes that occur at the same time is determined according to the following two elements:

- Priority of EIT causes to be accepted
 - How other causes can be masked when one cause is accepted
-

■ Priority of EIT Causes to Be Accepted

The priority of EIT causes to be accepted is the order of causes for which the EIT sequence is to be executed (that is, saving the PS and PC, updating the PC, and masking other causes, if required). The handler of a cause accepted earlier is not necessarily executed earlier.

Table 3.10-5 shows the priority of EIT causes to be accepted.

Table 3.10-5 Priority of EIT Causes to be Accepted and Masking of Other Causes

Priority of acceptance	Cause	Masking of other causes
1	Reset	Other causes are abandoned.
2	Undefined instruction exception	Canceled
3	INT instruction	I flag=0
4	No-coprocessor trap Coprocessor error trap	-
5	User interrupt	ILM=level of cause accepted
6	NMI (for users)	ILM=15
7	(INTE instruction)	ILM=4 *
8	NMI (for emulators)	ILM=4
9	Step trace trap	ILM=4
10	INTE instruction	ILM=4

*: The priority is "6" only if the INTE instruction and the NMI for emulators occur at the same time. The NMI for emulators is used in the MB91301 series for breaks due to data access.

In consideration of masking other causes after an EIT cause is accepted, the handlers of EIT causes that occur at the same time are executed in the order shown in Table 3.10-6.

Table 3.10-6 Order of Executing EIT Handlers

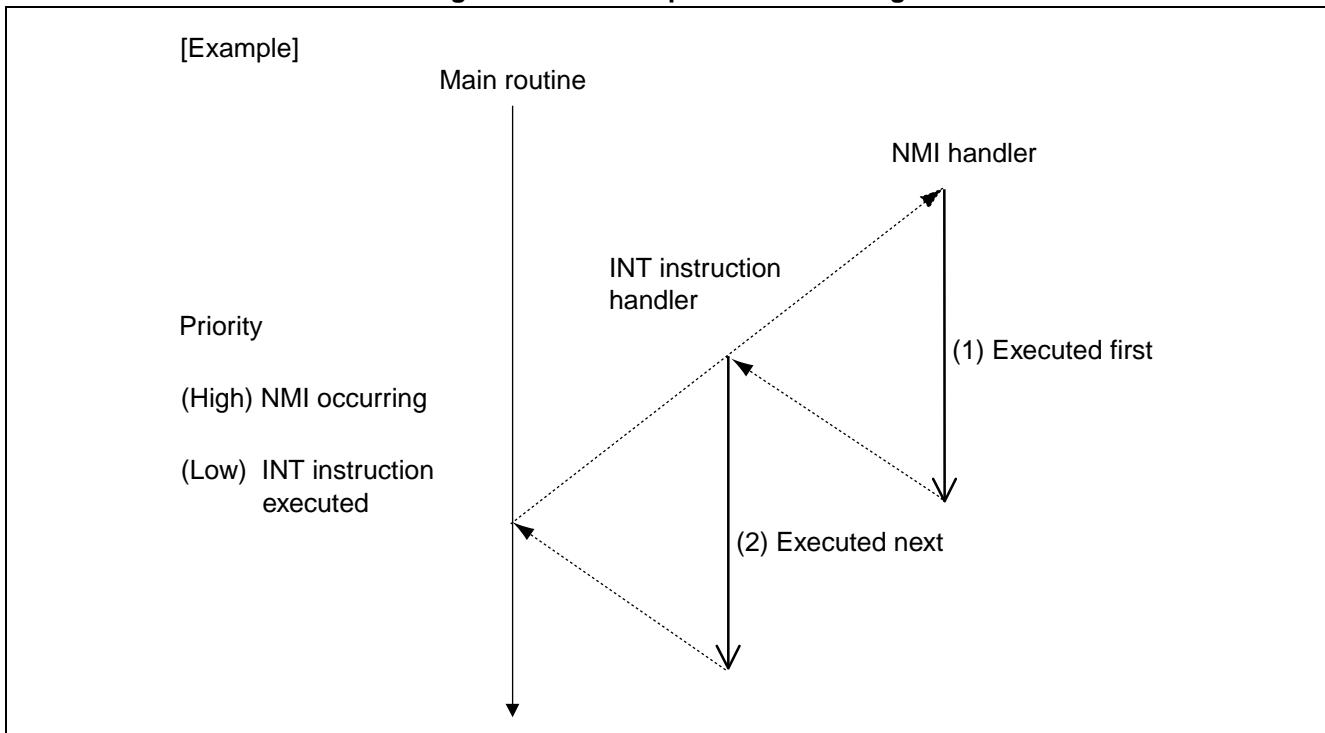
Order of executing handlers	Cause
1	Reset ^{*1}
2	Undefined instruction exception
3	Step trace trap ^{*2}
4	INTE instruction ^{*2}
5	NMI (for users)
6	INT instruction
7	User interrupt
8	No-coprocessor trap, coprocessor error trap

*1: Other causes are abandoned.

*2: If the INTE instruction is executed in steps, only a step trace trap EIT occurs. An INTE cause is ignored.

Figure 3.10-5 shows an example of multiple EIT processing.

Figure 3.10-5 Multiple EIT Processing



3.10.6 EIT Operations

This section describes EIT operations.

■ EIT Operations

In the following, it is assumed that the destination source PC indicates the address of the instruction that detected an EIT cause.

In addition, "address of the next instruction" means that the instruction that detected EIT is as follows:

- If LDI is 32: PC + 6
- If LDI is 20 and COPOP, COPLD, COPST, and COPSV are used: PC + 4
- Other instructions: PC + 2

■ Operation of User Interrupt/NMI

If an interrupt request for a user interrupt or a user NMI occurs, whether the request can be accepted is determined with the following procedure:

- 1) Compare the interrupt levels of requests that have occurred simultaneously and select the request with the highest level (the smallest value). As levels to be compared, the value held in the corresponding ICR is used for a maskable interrupt and a predetermined constant is used for an NMI.
- 2) If multiple interrupt requests with the same level occur, select the interrupt request with the smallest interrupt number.
- 3) Mask and do no accept an interrupt request with an interrupt level greater than or equal to the level mask value. Go to Step 4) if the interrupt level is less than the level mask value.
- 4) Mask and do not accept the selected interrupt request if it is maskable and the I flag is set to "0". Go to Step 5) if the I flag is "1". If the selected interrupt request is an NMI, go to Step 5) regardless of the I flag value.
- 5) If the above conditions are met, the interrupt request is accepted at a break in the instruction processing.

If a user interrupt or NMI request is accepted when EIT requests are detected, the CPU operates as follows, using an interrupt number corresponding to the accepted interrupt request. Parentheses show an address indicated by the register.

[Operation]

- 1) (TBR + Vector offset of accepted interrupt request) --> TMP
- 2) SSP-4 --> SSP
- 3) PS --> (SSP)
- 4) SSP-4 --> SSP
- 5) Address of next instruction --> (SSP)
- 6) Interrupt level of accepted request --> ILM
- 7) "0" --> S flag
- 8) TMP --> PC

A new EIT is detected before the execution of the first instruction in the handler, once the interrupt sequence is completed.

If an acceptable EIT has occurred at this point, the CPU enters the EIT processing sequence.

An instruction such as ORCCR, STILM, MOV Ri and PS may be executed twice before and after the interrupt handler, if it is executed to enable an interrupt while a user interrupt or NMI source is occurring. Note, however, that this does not affect the operation as it only means that the same value is set twice in the CPU registers.

Do not perform the process expecting the content of the PS register before EIT branching in the EIT processing routine.

■ Operation of INT Instruction

The INT #u8 instruction operates as shown below.

A branch to the interrupt handler for the vector indicated by u8 generation.

[Operation]

- 1) (TBR + 3FC_H-4 x u8) --> TMP
- 2) SSP-4 --> SSP
- 3) PS --> (SSP)
- 4) SSP-4 --> SSP
- 5) PC + 2 --> (SSP)
- 6) "0" --> I flag
- 7) "0" --> S flag
- 8) TMP --> PC

■ Operation of INTE Instruction

The INTE instruction operates as shown below.

A branch to the interrupt handler for the vector indicated by vector number #9 generation.

[Operation]

- 1) (TBR+3D8_H) --> TMP
- 2) SSP-4 --> SSP
- 3) PS --> (SSP)
- 4) SSP-4 --> SSP
- 5) PC + 2 --> (SSP)
- 6) "00100_B" --> ILM
- 7) "0" --> S flag
- 8) TMP --> PC

Do not use the INTE instruction in the processing routine of the INTE instruction or a step trace trap.

During step execution, no EIT due to INTE generation.

CHAPTER 3 CPU AND CONTROL UNITS

■ Operation of Step Trace Trap

Set the T flag in the SCR of the PS to enable the step trace function. A trap and a break then occur every time an instruction is executed. A step trace trap is detected under the following conditions:

[Conditions of step trace trap detection]

- T flag =1
- There is no delayed branch instruction.
- A processing routine other than the INTE instruction or a step trace trap is in progress.

If the above conditions are met, a break occurs between instruction operations.

[Operation]

- 1) (TBR+3CC_H) --> TMP
- 2) SSP-4 --> SSP
- 3) PS --> (SSP)
- 4) SSP-4 --> SSP
- 5) Address of next instruction --> (SSP)
- 6) "00100_B" --> ILM
- 7) "0" --> S flag
- 8) TMP --> PC

Set the T flag to enable the step trace trap to prohibit a user NMI and a user interrupt. No EIT occurs due to the INTE instruction.

A trap occurs in the FR family in the instruction following the one in which the T flag has been set.

■ Operation of Undefined Instruction Exception

If, during instruction decode, an undefined instruction is detected, an undefined instruction exception occurs.

An undefined instruction exception is detected under the following conditions:

- An undefined instruction is detected during instruction decode.
- The instruction is not located in the delay slot (it does not immediately follow the delayed branch instruction).

If the above conditions are met, an undefined instruction exception and a break occur.

[Operation]

- 1) (TBR+3C4_H) - -> TMP
- 2) SSP-4 --> SSP
- 3) PS --> (SSP)
- 4) SSP-4 --> SSP
- 5) PC --> (SSP)
- 6) "0" --> S flag
- 7) TMP --> PC

The PC value to be saved is the address of an instruction that detected an undefined instruction exception.

■ No-coprocessor Trap

If a coprocessor instruction using a coprocessor that is not installed is executed, a no-coprocessor trap occurs.

[Operation]

- 1) $(TBR+3E0_H) \rightarrow TMP$
- 2) $SSP-4 \rightarrow SSP$
- 3) $PS \rightarrow (SSP)$
- 4) $SSP-4 \rightarrow SSP$
- 5) Address of next instruction $\rightarrow (SSP)$
- 6) "0" $\rightarrow S$ flag
- 7) $TMP \rightarrow PC$

■ Coprocessor Error Trap

If an error occurs while a coprocessor is being used and then a coprocessor instruction that operates on the coprocessor is executed, a coprocessor error trap occurs.

[Operation]

- 1) $(TBR+3DC_H) \rightarrow TMP$
- 2) $SSP-4 \rightarrow SSP$
- 3) $PS \rightarrow (SSP)$
- 4) $SSP-4 \rightarrow SSP$
- 5) Address of next instruction $\rightarrow (SSP)$
- 6) "0" $\rightarrow S$ flag
- 7) $TMP \rightarrow PC$

■ Operation of RETI Instruction

The RETI instruction specifies return from the EIT processing routine.

[Operation]

- 1) $(R15) \rightarrow PC$
- 2) $R15+4 \rightarrow R15$
- 3) $(R15) \rightarrow PS$
- 4) $R15+4 \rightarrow R15$

The RETI instruction must be executed while the S flag is set to "0".

■ Precaution on Delay Slot

A delay slot for a branch instruction has restrictions regarding EIT.

See Section "3.9 Branch Instructions".

3.11 Reset (Device Initialization)

This section describes a reset (that is, device initialization) of the FR family.

■ Reset (Device Initialization)

If a reset source occurs, the device stops all the programs and hardware operations and completely initializes the state. This state is called the reset state.

When a reset source no longer exists, the device starts programs and hardware operations from their initial state. The series of operations from the reset state to the start of operations is called the reset sequence.

3.11.1 Reset Levels

The reset operations of the FR family device are classified into two levels, each of which has different causes and initialization operations. This section describes these reset levels.

■ Settings Initialization Reset (INIT)

The highest-level reset, which initializes all settings, is called a settings initialization reset (INIT).

A settings initialization reset (INIT) mainly performs the following initialization:

- **Items initialized in a settings initialization reset (INIT)**

- Device operation mode (setting of bus mode and external bus width)
- All internal clock settings (clock source selection, PLL control, and divide-by setting)
- All external bus extended interface settings
- All settings on pin statuses other than the above settings
- All sections initialized by an operation initialization reset (RST)

For more information, see the description of each of these functions.

Note:

After power-on, be sure to apply the settings initialization reset (INIT) at the $\overline{\text{INIT}}$ pin.

■ Operation Initialization Reset (RST)

A normal-level reset that initializes the operation of a program is called an operation initialization reset (RST).

If a settings initialization reset (INIT) occurs, an operation initialization reset (RST) also occurs.

An operation initialization reset (RST) mainly initializes the following items:

- **Items initialized by an operation initialization reset (RST)**

- Program operation
- CPU and internal buses
- Register settings of peripheral circuits
- I/O port settings
- All CS0 area settings of external buses
- Device operation mode (setting of bus mode and external bus width)

For more information, see the description of each of these functions.

3.11.2 Reset Sources

This section describes the reset sources and the reset levels in the FR family device. To determine reset sources that have occurred in the past, read the RSRR (reset source register). For more information about registers and flags described in this section, see Section "3.12.5 Block Diagram of Clock Generation Controller" and "3.12.6 Register of Clock Generation Controller".

■ INIT Pin Input (Settings Initialization Reset Pin)

The INIT pin, which is an external pin, is used as the settings initialization reset pin.

A settings initialization reset (INIT) request is generated while the "L" level is being input to this pin.

Input the "H" level to this pin to clear a settings initialization reset (INIT) request.

If a settings initialization reset (INIT) is generated in response to a request from this pin, bit15 (INIT bit) of the RSRR (reset source register) is set.

Because a settings initialization reset (INIT) in response to a request from this pin has the highest interrupt level among all reset sources, it has precedence over any other input, operation, or state.

Immediately after power-on, be sure to apply a settings initialization reset (INIT) at the INIT pin. To assure the oscillation stabilization wait time for the oscillation circuit immediately after power-on, input the "L" level to the INIT pin for the stabilization wait time required by the oscillation circuit. INIT at the INIT pin initializes the oscillation stabilization wait time to the minimum value.

- Reset source: "L" level input to the external INIT pin
- Source of clearing: "H" level input to the external INIT pin
- Reset level: Settings initialization reset (INIT)
- Corresponding flag: bit15 (INIT)

■ Software Reset (STCR: SRST Bit Writing)

If "0" is written to bit4 (SRST bit) of the standby control register (STCR), a software reset request occurs. A software reset request is an operation initialization reset (RST) request.

When the request is accepted and an operation initialization reset (RST) is generated, the software reset request is cleared.

If an operation initialization reset (RST) is generated due to a software reset request, a bit11 (SRST bit) in the RSRR (reset source register) is set.

An operation initialization reset (RST) is generated due to a software reset request only after all bus access has stopped and if bit7 (SYNCR bit) of the time base counter control register (TBCR) has been set (synchronization reset mode). Thus, depending on the bus usage status, a long time is required before an operation initialization reset (RST) occurs.

- Reset source: Writing "0" to bit4 (SRST) of the standby control register (STCR)
- Source of clearing: Generation of an operation initialization reset (RST)
- Reset level: Request of operation initialization reset (RST)
- Corresponding flag: bit11(SRST)

■ Watchdog Reset

Writing to the watchdog timer control register (RSRR) starts the watchdog timer. Unless A5_H/5A_H is written to the timebase counter clear register (CTBR) within the cycle specified in bit9 and bit8 (WT1 and WT0 bits) in the RSRR, a watchdog reset request occurs.

A watchdog reset request is a settings initialization reset (INIT) request. If, after the request is accepted, a settings initialization reset (INIT) occurs or an operation initialization reset (RST) occurs, the watchdog reset request is cleared.

If a settings initialization reset (INIT) is generated due to a watchdog reset request, bit13 (WDOG bit) in the reset source register (RSRR) is set.

Note that, if a settings initialization reset (INIT) is generated due to a watchdog reset request, the oscillation stabilization wait time is not initialized.

- Reset source: Setting cycle of the watchdog timer elapses
- Source of clearing: Generation of a settings initialization reset (INIT) or an operation initialization reset (RST)
- Reset level: Settings initialization reset (INIT)
- Corresponding flag: bit13 (WDOG)

3.11.3 Reset Sequence

When a reset source no longer exists, the device starts to execute the reset sequence.

A reset sequence has different operations depending on the reset level.

This section describes the operations of the reset sequence for different reset levels.

■ Setting Initialization Reset (INIT) Clear Sequence

If a settings initialization reset (INIT) request is cleared, the following operations are performed one step at a time for the device.

- 1) Clear the settings initialization reset (INIT) and enter the oscillation stabilization wait state.
- 2) For the oscillation stabilization wait time (set with bit3 and bit2 [OS1 and OS0 bits] in the STCR), maintain the operation initialization reset (RST) state and stop the internal clock.
- 3) In the operation initialization reset (RST) state, start internal clock operation.
- 4) Clear the operation initialization reset (RST) and enter the normal operating state.
- 5) Read the mode vector from address 000FFFF8_H.
- 6) Write the mode vector to the MODR (mode register) at address 000007FD_H.
- 7) Read the reset vector from address 000FFFC_H.
- 8) Write the reset vector to the program counter (PC).
- 9) The program starts execution from the address loaded in the program counter (PC).

■ Operation Initialization Reset (RST) Clear Sequence

If an operation initialization reset (RST) request is cleared, the following operations are performed one step at a time for the device.

- 1) Clear the operation initialization reset (RST) and enter the normal operating state.
- 2) Read the mode vector from address 000FFFF8_H
- 3) Write the mode vector to the MODR (mode register) at address 000007FD_H
- 4) Read the reset vector from address 000FFFC_H.
- 5) Write the reset vector to the program counter (PC).
- 6) The program starts execution from the address loaded in the program counter (PC).

3.11.4 Oscillation Stabilization Wait Time

If a device returns from the state in which the original oscillation was or may have been stopped, the device automatically enters the oscillation stabilization wait state. This function prevents the use of oscillator output after starting before oscillation has stabilized.

For the oscillation stabilization wait time, neither an internal nor an external clock is supplied; only the built-in time base counter runs until the stabilization wait time set in the standby control register (STCR) has elapsed.

This section describes the oscillation stabilization wait operation.

■ Sources of an Oscillation Stabilization Wait

The following lists sources of an oscillation stabilization wait.

- **Clearing of a settings initialization reset (INIT)**

The device enters the oscillation stabilization wait state if a settings initialization reset (INIT) is cleared for a variety of reasons.

When the oscillation stabilization wait time has elapsed, the device enters the operation initialization reset (RST) state.

- **Returning from stop mode**

The device enters the oscillation stabilization wait state immediately after stop mode is cleared.

However, if it is cleared by a settings initialization reset (INIT) request, the device enters the settings initialization reset (INIT) state. Then, after the settings initialization reset (INIT) is cleared, the device enters the oscillation stabilization wait state.

When the oscillation stabilization wait time has elapsed, the device enters the state corresponding to the source that cleared stop mode:

- Return due to input of a valid external interrupt request (including NMI):
The device enters the normal operating state.
- Return due to a settings initialization reset (INIT) request:
The device enters the operation initialization reset (INIT) state.
- Return due to an operation initialization reset (RST) request:
The device enters the operation initialization reset (RST) state.

- **Returning from an abnormal state when PLL is selected**

If, while the device is operating with PLL as the source clock, an abnormal condition* occurs in PLL control, the device automatically enters an oscillation stabilization wait to assure the PLL lock time.

When the oscillation stabilization wait time has elapsed, the device enters the normal operating state.

*: The multiply-by rate is changed while PLL is working, or an incorrect bit such as a bit equivalent to PLL operation enable bit is generated.

■ Selecting an Oscillation Stabilization Wait Time

The oscillation stabilization wait time is measured with the built-in time base counter.

If a source for an oscillation stabilization wait occurs and the device enters the oscillation stabilization wait state, the built-in time base counter is initialized and then it starts to measure the oscillation stabilization wait time.

Using bit3 and bit2 (OS1 and OS0 bits) of the standby control register (STCR), select and set one of the four types of oscillation stabilization wait time.

Once selected, a setting is initialized only if a settings initialization reset (INIT) is generated due to the external $\overline{\text{INIT}}$ pin. The oscillation stabilization wait time that has been set before a reset is maintained if a settings initialization reset (INIT) is generated or an operation initialization reset (RST) is generated due to a watchdog reset or hardware standby condition.

The four types of oscillation stabilization wait time settings are designed for the following four types of use:

- OS1, OS0=00_B: No oscillation stabilization wait time (if neither PLL nor the oscillator should stop in stop mode)
- OS1, OS0=01_B: PLL lock wait time (if an external clock will be input or the oscillator should not stop in stop mode)
- OS1, OS0=10_B: Oscillation stabilization wait time (intermediate) (if an oscillator that stabilizes quickly, such as a ceramic vibrator, is used)
- OS1, OS0=11_B: Oscillation stabilization wait time (long) (if an ordinary quartz oscillator will be used)

Immediately after power-on, be sure to apply the settings initialization reset (INIT) at the $\overline{\text{INIT}}$ pin.

To assure the oscillation stabilization wait time of the oscillation circuit immediately after power-on, maintain "L" level input to the $\overline{\text{INIT}}$ pin for the stabilization wait time required by the oscillation circuit and the regulator (INIT generated due to the $\overline{\text{INIT}}$ pin initializes the oscillation stabilization wait time setting to the minimum value).

3.11.5 Reset Operation Modes

Two modes for an operation initialization reset (RST) are provided: normal (asynchronous) reset mode and synchronous reset mode. The operation initialization reset mode is selected with bit7 (SYNCR bit) of the time base counter control register (TBCR). This mode setting is initialized only by a settings initialization reset (INIT). A settings initialization reset always results in an asynchronous reset.

This section describes the operation of these modes.

■ Normal Reset Operation

Normal reset operation refers to entering the operation initialization reset (RST) state immediately after an operation initialization reset (RST) request occurs.

If, in this mode, a reset (RST) request is accepted, the device immediately enters the reset (RST) state regardless of the operating state of the internal bus.

In this mode, the result of bus access performed prior to each status transition is not guaranteed. However, these requests can certainly be accepted.

If bit7 (SYNCR bit) of the time base counter control register (TBCR) is set to "0", normal reset mode is selected. The initial value after a settings initialization reset (INIT) is normal reset mode.

■ Synchronous Reset Operation

Synchronous reset operation refers to entering the operation initialization reset (RST) state after all bus access has stopped when an operation initialization reset (RST) request occurs.

If, in this mode, a reset (RST) request is accepted, the device does not enter the reset (RST) state while internal bus access is in progress.

If the above request is accepted, a sleep request is issued to the internal buses. If all the buses stop and enter the sleep state, the device enters the operation initialization reset (RST) state.

In this mode, the result of all bus accesses is guaranteed because all bus access is stopped prior to each status transition.

If bus access does not stop for some reason, no requests can be accepted while the bus access is in progress. Even in this case, the settings initialization reset (INIT) is immediately valid.

Bus access may not stop in the following cases:

- A bus release request (BRQ) continues to be input to the external extended bus interface, bus release acknowledge (BGRNT) is valid, and a new bus access request arrives from an internal bus.
- A ready request (RDY) continues to be input to the external extended bus interface and bus wait is valid. In the following cases, the device eventually enters another state but only after a long time.
- When self-refreshing in sleep mode has been set with the SDRAM interface activated (State transition does not occur until the self-refresh mode setting is completed.)

Reference:

The DMA controller, which stops transfer when a request is accepted, does not delay transition to another state. If bit7 (SYNCR bit) of the time base counter control register (TBCR) is set to "1", synchronous reset mode is selected. The initial value after a settings initialization reset (INIT) is normal reset mode.

3.12 Clock Generation Control

This section describes clock generation and control.

■ Clock Generation Control

The internal operating clock of the FR family device is generated as follows:

- Selection of a source clock: Select a clock supply source.
- Generation of a base clock: Divide the source clock by two or perform PLL oscillation to generate a base clock.
- Generation of an internal clock: Divide the base clock and generate three types of operating clocks, which are supplied to each section.

The following section describes the generation and control of each clock. For detailed information about the registers and flags described below, see "3.12.5 Block Diagram of Clock Generation Controller" and "3.12.6 Register of Clock Generation Controller".

The symbol “ ϕ ”, which is shown in Table 3.12-4, Table 3.12-9, Table 3.12-13, Table 3.12-16, Table 3.12-20, Table 3.12-21 and Table 3.12-22, indicates the basic clock that is achieved by dividing the source clock by two or performing PLL oscillation. Therefore, the system base clock is the clock which is generated where the above base clock is generated.

■ Source Clock

○ Self-induced oscillation mode (X0/X1 pin input)

In this mode, an oscillator is connected to external oscillation pins and the original oscillation generated by the built-in oscillation circuit is used as the source clock.

The source for supply of all clocks, including the external bus clock, is the MB91301 series itself.

The main clock, generated from the X0/X1 pins, is intended to be used as a high-speed clock.

The main clock is multiplied by the built-in main PLL, each of which can be independently controlled.

Generate an internal base clock by selecting one of the following source clocks:

- Main clock divided by 2
- Main clock multiplied in the main PLL

Select a source clock by setting the clock source control register (CLKR).

3.12.1 PLL Controls

Operation (oscillation) enable and disable and the multiply-by rate setting can be independently controlled for each of the PLL oscillation circuits corresponding to the main source clock. Each control is set in the clock source control register (CLKR).
This section describes each control.

■ PLL Operation Enable

To enable or disable the main PLL oscillation, set bit10 (PLL1EN bit) of the clock source control register (CLKR).

○ PLL operation control

In self-induced oscillation mode, either the operation enable/disable bit or the multiply-by rate setting bit is initialized to "0" after a settings initialization reset (INIT), causing the PLL oscillation to stop. While it is stopped, PLL output cannot be selected as the source clock.

When the program operation starts, set the multiply-by rate of the PLL to be used as the clock source, enable it, and switch the source clock after the PLL lock wait time elapses. For the PLL lock wait time, use of a time-base timer interrupt is recommended.

While PLL output is selected as the source clock, the PLL cannot be stopped (writing to the register is disabled). To stop a PLL upon transition to stop mode, reselect as the source clock the main clock divided by two before stopping the PLL.

If bit0 (OSCD1 bit) of the standby control register (STCR) is set to stop oscillation in stop mode, the corresponding PLL automatically stops when the device enters stop mode. As a result, you do not need to set operation stop. When the device returns from stop mode later, the PLL automatically restarts the oscillation operation. If oscillation is not set to stop in stop mode, the PLL does not automatically stop. In this case, set operation stop before transition to stop mode as required.

■ PLL Multiply-by Rate

Set the multiply-by rate of the main PLL in bit14 to bit12 (PLL1S2, PLL1S1, and PLL1S0 bits) of the clock source control register (CLKR).

After a settings initialization reset (INIT), all bits are initialized to "0".

○ PLL multiply-by rate setting in self-induced oscillation mode

To change the PLL multiply-by rate setting from the initial value in self-induced oscillation mode, do so before or as soon as the PLL is enabled after the program has started execution. After changing the multiply-by rate, switch the source clock after the lock wait time elapses. For the PLL lock wait time, use of a time-base timer interrupt is recommended.

To change the PLL multiply-by rate setting during operation, switch the source clock to a clock other than the PLL in question before making the change. After changing the multiply-by rate, switch the source clock after the lock wait time has elapsed, as described above.

You can also change the PLL multiply-by rate setting while using a PLL. In this case, however, the program stops running after the device automatically enters the oscillation stabilization wait state after the multiply-by rate setting is rewritten and does not resume execution until the specified oscillation stabilization wait time has elapsed.

The program does not stop running if the clock source is switched to a clock other than a PLL.

3.12.2 Oscillation Stabilization Wait Time and PLL Lock Wait Time

If a clock selected as the source clock is not already stabilized, an oscillation stabilization wait time is required (See Section "3.11.4 Oscillation Stabilization Wait Time").

For a PLL, a lock wait time is required after operation starts until the output stabilizes to the specified frequency.

This section describes the wait time used in various situations.

- **Wait time after power-on**

After power-on, an oscillation stabilization wait time for the main clock oscillation circuit is required.

Since the oscillation stabilization wait time setting is initialized to the minimum value due to INIT pin input (settings initialization reset pin), assure the oscillation stabilization wait time by using the time during which the "L" level is sent to the INIT pin input.

In this state, since no PLL is enabled, no lock wait time needs to be considered.

- **Wait time after setting initialization**

If a settings initialization reset (INIT) is cleared, the device enters the oscillation stabilization wait state. In this case, the specified oscillation stabilization wait is internally generated. In the first oscillation stabilization wait state after input from the INIT pin, the setting time is initialized to the minimum value, soon ending this state, and the device enters the operation initialization reset (RST) state.

If, after a program starts running, a settings initialization reset (INIT) is generated for a reason other than INIT pin input and is then cleared, the oscillation stabilization wait time specified in the program is internally generated.

In these states, since no PLL is enabled, no lock wait time needs to be considered.

- **Wait time after changing the PLL multiply-by rate**

If you change the multiply-by rate setting of a running PLL after a program starts execution, use the PLL output only after lock wait time elapses.

If the PLL is not selected as the source clock, the program can run even during the lock wait time.

For the PLL lock wait time, use of a time-base timer interrupt is recommended.

- **Wait time after returning from stop mode**

If, after a program starts execution, the device enters stop mode and then stop mode is cleared, the oscillation stabilization wait time specified in the program is internally generated. If the clock oscillation circuit selected as the source clock is set to stop in stop mode, the oscillation stabilization wait time of the oscillation circuit or the lock wait time of the PLL in use, whichever is longer, is required. Set the oscillation stabilization wait time before entering stop mode.

If the clock oscillation circuit selected as the source clock is not set to stop in stop mode, the PLL does not automatically stop. No oscillation stabilization wait time is required unless the PLL has stopped. Setting the oscillation stabilization wait time to the minimum value before stop mode is entered is recommended.

3.12.3 Clock Distribution

An operating clock for each function is generated based on the base clock generated from the source clock. A total of four internal operating clocks are provided. A divide-by rate can be set independently for each of them.

This section describes these internal operating clocks.

■ CPU Clock (CLKB)

This clock is used for the CPU, internal memory, and internal buses.

It is used by the following circuits:

- CPU
- Built-in RAM
- Bit search module
- I-bus, D-bus, X-bus, and F-bus
- DMA controller
- DSU (development tool interface circuit)

Since 68 MHz is the upper-limit frequency for operation, do not set a combination of multiply-by rate and divide-by rate that results in a frequency exceeding this limit.

■ Peripheral Clock (CLKP)

This clock is used for peripheral circuits and peripheral buses.

It is used by the following circuits:

- Peripheral bus
- Clock controller (only for the bus interface)
- Interrupt controller
- Peripheral I/O ports
- I/O port bus
- External interrupt input
- UART
- 16-bit timer
- A/D converter
- I²C interface

Since 34 MHz is the upper-limit frequency for operation, do not set a combination of multiply-by rate and divide-by rate that results in a frequency exceeding this limit.

■ External Bus Clock (CLKT)

This clock is used for external extended bus interfaces.

It is used by the following circuits:

- External extended bus interface
- External CLK output
- Bus interface port

Since 68 MHz is the upper-limit frequency for operation, do not set a combination of multiply-by rate and divide-by rate that results in a frequency exceeding this limit.

3.12.4 Clock Division

A divide-by rate can be set independently for each of the internal operating clocks. With this function, an optimum operating frequency can be set for each circuit.

■ Clock Division

Set a divide-by rate in Basic Clock Division Setting Register 0 (DIVR0) and Basic Clock Division Setting Register 1 (DIVR1). Each of these registers has four setting bits and (Register setting value + 1) is the divide-by rate of the clock in relation to the base clock. Even if the divide-by rate setting is an odd number, the duty is always 50%.

If the setting value is changed, the new divide-by rate becomes valid at the leading edge of the next clock after the setting is made.

The divide-by rate setting is not initialized if an operation initialization reset (RST) occurs and the setting made before the reset occurs is retained. The divide-by rate setting is initialized only if a settings initialization reset (INIT) occurs. In the initial state, all clocks other than the peripheral clock (CLKP) have a divide-by rate of "1". Thus, be sure to set the divide-by rate before changing the source clock to a faster clock.

An upper-limit frequency for the operation is set for each clock. If you set a combination of source clock, PLL multiply-by rate setting, and divide-by rate setting that results in a frequency exceeding this upper-limit frequency, operation is not guaranteed. Be extra careful of the order in which you change settings to select the source clock and to configure the associated setting items.

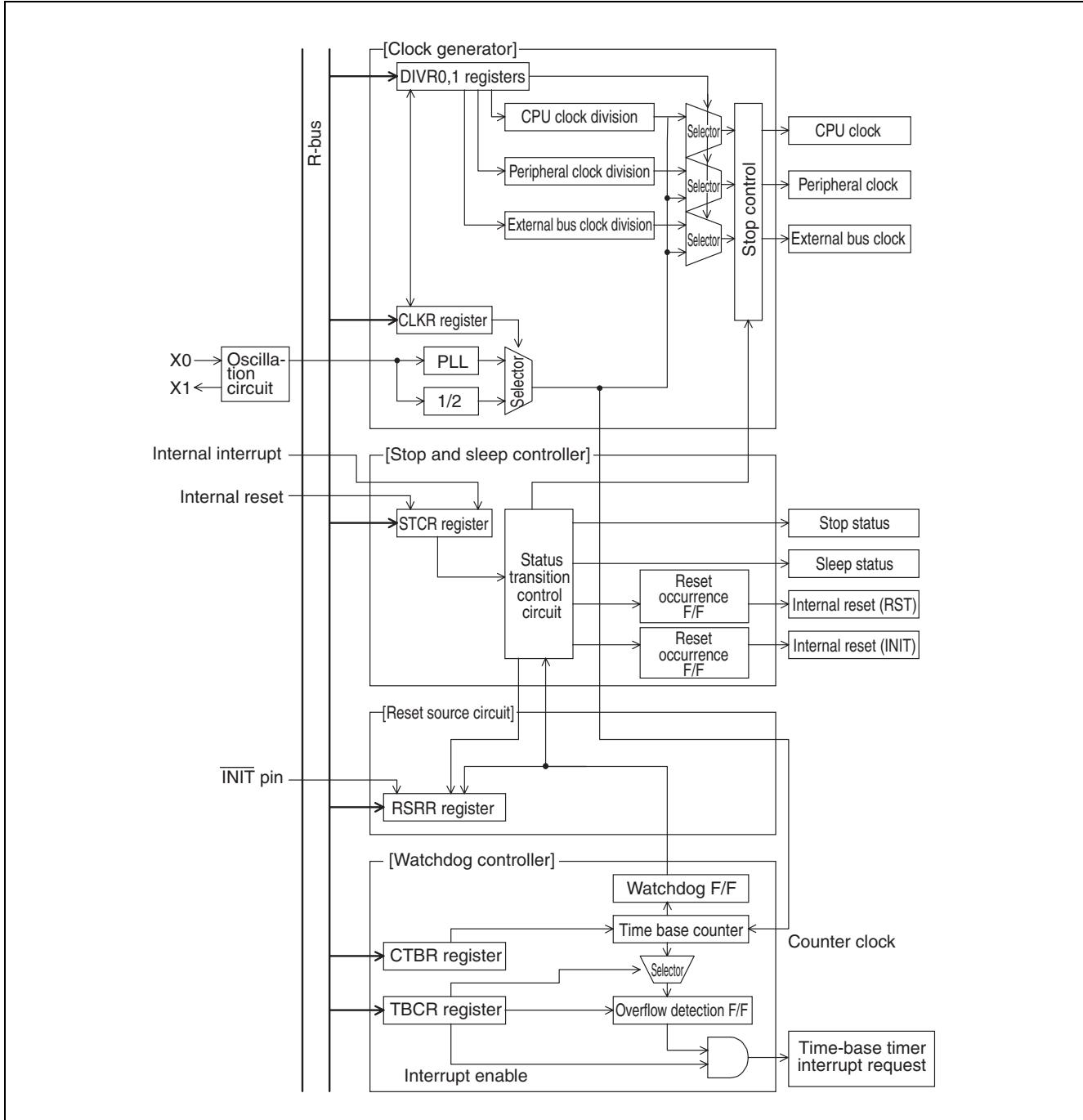
3.12.5 Block Diagram of Clock Generation Controller

This section provides a block diagram of the clock generation controller.

■ Block Diagram

Figure 3.12-1 shows a block diagram of the clock generation controller. Refer to "3.12.6 Register of Clock Generation Controller" for the detail of registers in Figure 3.12-1.

Figure 3.12-1 Block Diagram of Clock Generation Controller



3.12.6 Register of Clock Generation Controller

This section describes the functions of registers to be used in the clock generation controller.

■ Reset Source Register/Watchdog Timer Control Register (RSRR)

Figure 3.12-2 shows the configuration of the reset source register/watchdog timer control register (RSRR).

Figure 3.12-2 Reset Source Register/Watchdog Timer Control Register (RSRR)

bit	15	14	13	12	11	10	9	8
Address: 00000480 _H	INIT	-	WDOG	-	SRST	-	WT1	WT0
	R	R	R	R	R	R	R/W	R/W
Initial value ($\overline{\text{INIT}}$ pin)	1	0	0	0	0	0	0	0
Initial value (INIT)	-	0	-	X	X	-	0	0
Initial value (RST)	X	X	X	-	-	X	0	0

–: Varies according to the source.
x: Not initialized

This register holds the source of the last reset that occurred as well as the interval setting and startup control for the watchdog timer. If the timer is read, the reset source that has been held is cleared after it is read. If more than one reset is generated before this register is read, reset source flags are accumulated and the multiple flags are set.

Writing to this register starts the watchdog timer. Thereafter, the watchdog timer continues running until a reset (RST) occurs.

The following describes the functions of the reset source register/watchdog timer control register (RSRR) bits.

[bit15] INIT (INITialize reset occurred)

This bit indicates whether a reset (INIT) occurred due to $\overline{\text{INIT}}$ pin input.

Table 3.12-1 INIT Function

INIT	Function
0	No reset (INIT) occurred due to $\overline{\text{INIT}}$ pin input.
1	A reset (INIT) occurred due to $\overline{\text{INIT}}$ pin input.

- This bit is initialized to "0" after it is read.
- This bit is readable; writing to the bit has no effect on the bit value.

[bit14] (Reserved)

This bit is reserved.

[bit13] WDOG (WatchDOG reset occurred)

This bit indicates whether a reset (INIT) occurred due to the watchdog timer.

Table 3.12-2 WDOG Function

WDOG	Function
0	No reset (INIT) occurred due to the watchdog timer.
1	A reset (INIT) occurred due to watchdog timer.

- This bit is initialized to "0" after a reset (INIT) due to $\overline{\text{INIT}}$ pin input or just after it is read.
- This bit is readable; writing to the bit has no effect on the bit value.

[bit12] (Reserved)

This bit is reserved.

[bit11] SRST (Software ReSeT occurred)

This bit indicates whether a reset (RST) occurred due to writing to the SRST bit of the STCR register (a software reset).

Table 3.12-3 SRST Function

SRST	Function
0	No reset (RST) occurred due to a software reset.
1	A reset (RST) occurred due to a software reset.

- This bit is initialized to "0" after a reset (INIT) due to $\overline{\text{INIT}}$ pin input or just after it is read.
- This bit is readable; writing to the bit has no effect on the bit value.

[bit10] (Reserved)

This bit is reserved.

[bit9, bit8] WT1, WT0 (Watchdog interval Time select)

This bit sets the interval of the watchdog timer.

The values written to these bits determine the interval of the watchdog timer, which can be selected from the four types shown in Table 3.12-4.

Table 3.12-4 Interval Setting of Watchdog Timer

WT1	WT0	Minimum required interval for writing to the CTBR to suppress a watchdog reset	Time from writing the last $5A_H$ to the CTBR until a watchdog reset occurs
0	0	$\phi \times 2^{16}$ (initial value)	$\phi \times 2^{16}$ to $\phi \times 2^{17}$
0	1	$\phi \times 2^{18}$	$\phi \times 2^{18}$ to $\phi \times 2^{19}$
1	0	$\phi \times 2^{20}$	$\phi \times 2^{20}$ to $\phi \times 2^{21}$
1	1	$\phi \times 2^{22}$	$\phi \times 2^{22}$ to $\phi \times 2^{23}$

Note: ϕ : Frequency of the system base clock

- These bits are initialized to " 00_B " after a reset (RST).
- These bits are readable, but are writable only once after a reset (RST). Any further writing is disabled.

CHAPTER 3 CPU AND CONTROL UNITS

■ Standby Control Register (STCR)

Figure 3.12-3 shows the configuration of the standby control register (STCR).

Figure 3.12-3 Configuration of Standby Control Register (STCR) Bits

bit	7	6	5	4	3	2	1	0
Address: 00000481H	STOP	SLEEP	HIZ	SRST	OS1	OS0	-	OSCD1
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (<u>INIT</u> pin)	0	0	1	1	0	0	1	1
Initial value (INIT)	0	0	1	1	X	X	1	1
Initial value (RST)	0	0	X	1	X	X	X	X

The standby control register (STCR) controls the operating mode of the device.

This register controls the transition to the two standby modes of stop and sleep, pins when in stop mode, and the stopping of oscillation stop. It also sets the oscillation stabilization wait time and issues software resets.

Note:

Use the following sequences after using the synchronous standby mode (TBCR:Set by time base counter control register bit8 SYNCS bit) when putting in the standby mode.

```
(LDI #value_of_standby, R0)      ; value_of_standby is the writing data to STCR.
(LDI #_STCR, R12)                ; _STCR is the address (481H) of STCR.
STB   R0, @R12                  ; Writing in standby control register (STCR)
LDUB  @R12, R0                  ; STCR read for synchronous standby
LDUB  @R12, R0                  ; Dummy re-read of STCR
NOP                            ; five NOPs for timing adjustment
NOP
NOP
NOP
NOP
```

Furthermore, set the I flag, ILM and ICR to ensure that branching into the interrupt handler which is a return source occurs after returning to the standby mode.

The following describes the functions of the standby control register (STCR) bits.

[bit7] STOP (STOP mode)

This bit specifies entry into stop mode. If "1" is written to both bit6 (SLEEP bit) and this bit, this bit (STOP) has precedence and the device enters stop mode.

Table 3.12-5 STOP Function

STOP	Function
0	Stop mode not entered (initial value)
1	Stop mode entered

- This bit is initialized to "0" by a reset (RST) and by a stop return source.
- This bit is readable and writable.

[bit6] SLEEP (SLEEP mode)

This bit specifies entry into stop mode. If "1" is written to both bit7 (STOP bit) and this bit, bit7 (STOP) has precedence and the device enters stop mode.

Table 3.12-6 SLEEP Function

SLEEP	Function
0	Sleep mode not entered (initial value)
1	Sleep mode entered

- This bit is initialized to "0" by a reset (RST) and by a sleep return source.
- This bit is readable and writable.

[bit5] HIZ (HIZ mode)

This bit controls the pin state in stop mode.

Table 3.12-7 HIZ Function

HIZ	Function
0	The pin state before stop mode entered is maintained.
1	Pin output is set to high-impedance state in stop mode (initial value).

- This bit is initialized to "0" by a reset (INIT).
- This bit is readable and writable.

[bit4] SRST (Software ReSeT)

This bit specifies issuing of a software reset (RST).

Table 3.12-8 SRST Function

SRST	Function
0	A software reset is issued.
1	A software reset is not issued (initial value).

- This bit is initialized to "1" by a reset (RST).
- This bit is readable and writable. The read value is always "1".

[bit3, bit2] OS1, OS0 (Oscillation Stabilization time select)

These bits set the oscillation stabilization wait time used after a reset (INIT), return from stop mode, etc.

The values written to these bits determine the interval of the watchdog timer, which can be selected from the four types shown in Table 3.12-9.

Table 3.12-9 Oscillation Stabilization Wait Settings

OS1	OS0	Oscillation stabilization wait time	If the source oscillation is 17 MHz
0	0	$\phi \times 2^1$ (initial value)	0.238 [μs]
0	1	$\phi \times 2^{11}$	240.9 [μs]
1	0	$\phi \times 2^{16}$	7.71 [ms]
1	1	$\phi \times 2^{22}$	493 [ms]

Note: ϕ :Frequency of the system base clock; in this case, twice the cycle of the source oscillation input

- These bits are initialized to "00_B" by a reset (INIT) generated due to $\overline{\text{INIT}}$ pin input.
- These bits are readable and writable.

[bit1] (Reserved)

- This bit is initialized to "1" by a reset (INIT).
- Write "1" when writing.

[bit0] OSCD1 (OSCillation Disable mode for XIN1)

This bit controls stopping of main oscillation input (XIN1) in stop mode.

Table 3.12-10 OSCD1 Function

OSCD1	Function
0	Main oscillation does not stop in stop mode.
1	Main oscillation stops in stop mode (initial value).

- This bit is initialized to "1" by a reset (INIT).
- This bit is readable and writable.

■ Time Base Counter Control Register (TBCR)

Figure 3.12-4 shows the configuration of the time base counter control register (TBCR) bits.

Figure 3.12-4 Configuration of Time Base Counter Control Register (TBCR) Bits

bit	7	6	5	4	3	2	1	0
Address: 00000482H	TBIF	TBIE	TBC2	TBC1	TBC0	-	SYNCR	SYNCS
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	X	X	X	X	0	0
Initial value (RST)	0	0	X	X	X	X	X	X

The time base counter control register (TBCR) controls time-base timer interrupts, among other things.

This register enables time-base timer interrupts, selects an interrupt interval time, and sets an optional function for the reset operation.

The following describes the functions of the time base counter control register (TBCR) bits.

[bit15] TBIF (Time-base timer Interrupt Flag)

This bit is the time-base timer interrupt flag. It indicates that the interval time (TBC2 to TBC0 bits, which are bit13 to bit11) specified by the time base counter has elapsed.

A time-base timer interrupt request is generated if this bit is set to "1" when interrupts are enabled by bit14 (TBIE bit, TBIE=1).

Table 3.12-11 Function of Time-base timer Interrupt Flag (TBIF)

TBIF	Function
Clear source	An instruction writes "0".
Set source	The specified interval time elapses (the trailing edge of the time base counter is detected).

- This bit is initialized to "0" by a reset (RST).
- This bit is readable and writable, although only "0" can be written to it. Writing "1" does not change the bit value. The value read by a read-modify-write instruction is always "1".

[bit14] TBIE (Time-base Timer Interrupt Enable)

This bit is the time-base timer interrupt request output enable bit.

It controls output of an interrupt request when the interval time of the time base counter has elapsed. A time-base timer interrupt request is generated if the TBIF bit is set to "1" when this bit is set to "1".

Table 3.12-12 Function of time-base timer interrupt request output enable bit (TBIE)

TBIE	Function
0	Time-base timer interrupt request output disabled (initial value)
1	Time-base timer interrupt request output enabled

- This bit is initialized to "0" by a reset (RST).
- This bit is readable and writable.

[bit13 to bit11] TBC2, TBC1, TBC0 (time-base timer Counting time select)

These bits set the interval time of the time base counter that is used for the time-base timer.

The values written to these bits determine the interval time, which can be selected from the eight types shown in Table 3.12-13.

Table 3.12-13 Interval Settings

TBC2	TBC1	TBC0	Timer interval time	If the source oscillation is 17 MHz and PLL is multiplied by 4
0	0	0	$\phi \times 2^{11}$	30.1 [μs]
0	0	1	$\phi \times 2^{12}$	60.2 [μs]
0	1	0	$\phi \times 2^{13}$	120.5 [μs]
0	1	1	$\phi \times 2^{22}$	61.7 [ms]
1	0	0	$\phi \times 2^{23}$	123.4 [ms]
1	0	1	$\phi \times 2^{24}$	246.7 [ms]
1	1	0	$\phi \times 2^{25}$	493.4 [ms]
1	1	1	$\phi \times 2^{26}$	986.9 [ms]

Note: ϕ : Frequency of the system base clock

- The initial value is undefined. Be sure to set a value before enabling an interrupt.
- These bits are readable and writable.

[bit10] (Reserved)

This bit is reserved. The read value is undefined. Writing to this bit has no effect on operation.

[bit9] SYNCR (SYNChronous Reset enable)

This bit is the synchronous reset enable bit.

It is used to select one of the following operations, which is to be used if an operation initialization reset (RST) request occurs:

- Performing a normal rest for Immediate reset (RST)
- Performing a synchronous reset after stopping all bus access for operation initial reset (RST)

Table 3.12-14 Function of synchronous reset operation enable bit (SYNCR)

SYNCR	Function
0	Normal reset operation (initial value)
1	Synchronous reset operation

- This bit is initialized to "0" by a reset (INIT).
- This bit is readable and writable.

[bit8] SYNCs (SYNChronous Standby enable)

This bit is the synchronous standby enable bit.

Be sure to set "1" into this bit when using the standby mode (sleep or stop mode).

Table 3.12-15 Function of synchronous standby operation enable bit (SYNCs)

SYNCs	Function
0	Normal standby operation (initial value)
1	Synchronous standby operation

- This bit is initialized to "0" by a reset (INIT).
- This bit is readable and writable.

■ Time Base Counter Clear Register (CTBR)

Figure 3.12-5 shows the configuration of the time base counter clear register (CTBR) bits.

Figure 3.12-5 Configuration of Time Base Counter Clear Register (CTBR) Bits

bit	7	6	5	4	3	2	1	0
Address: 000483H	D7	D6	D5	D4	D3	D2	D1	D0
	W	W	W	W	W	W	W	W
Initial value (INIT)	X	X	X	X	X	X	X	X
Initial value (RST)	X	X	X	X	X	X	X	X

The time base counter clear register (CTBR) initializes the time base counter.

If "A5H" and "5AH" are written successively to this register, all the bits in the time base counter are cleared to "0" as soon as "5AH" is written. There is no time limit between writing of "A5H" and "5AH". However, if data other than "5AH" is written after "A5H" is written, "A5H" must be written again before "5AH" is written. Otherwise, a clear operation will not occur.

The bits are automatically cleared when the CPU is not in operation in such cases as stop/sleep mode and DMA transfer. Therefore, if the above conditions occur, the watchdog reset is delayed automatically. However, it will not be delayed, if a request to hold an external bus (BRQ) has been accepted. For this reason, select the sleep mode before entering a hold request (BRQ), when intending to hold the external bus for a long period.

The value read from this register is undefined.

Note:

If the time base counter is cleared using this register, the oscillation stabilization wait interval, watchdog timer interval, and time-base timer interval temporarily vary.

■ Clock Source Control Register (CLKR)

Figure 3.12-6 shows the configuration of the clock source control register (CLKR) bits.

Figure 3.12-6 Configuration of Clock Source Control Register (CLKR) Bits

bit	15	14	13	12	11	10	9	8
Address: 000484 _H	Reserved	PLL1S2	PLL1S1	PLL1S0	Reserved	PLL1EN	CLKS1	CLKS0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	0	0	0	0	0	0
Initial value (RST)	X	X	X	X	X	X	X	X

The clock source control register (CLKR) is used to select the clock source that will be used as the base clock of the system and controls the PLL. Use this register to select one of two clock sources. This register also enables the main PLL and each of the sub-PLLs and selects the multiply-by rate for them.

The following describes the functions of the clock source control register (CLKR) bits.

[bit15] (Reserved)

This bit is reserved.

[bit14 to bit12] PLL1S2, PLL1S1, PLL1S0 (PLL1 ratio Select 2 to 0)

These bits are the multiply-by selection bits for the main PLL. Select one of the eight multiply-by rates (the MB91301 series supports only four of these) shown in Table 3.12-16.

Rewriting of these bits is disabled while the main PLL is selected as the clock source.

Table 3.12-16 Main PLL Multiply-By Rate Settings

PLL1S2	PLL1S1	PLL1S0	Main PLL multiply-by rate	If the source oscillation is 17 MHz
0	0	0	x 1 (equal)	*2
0	0	1	x 2 (multiplied by 2)	*2
0	1	0	x 3 (multiplied by 3)	$\phi^{*1}=19.6[\text{ns}](51.0[\text{MHz}])$
0	1	1	x 4 (multiplied by 4)	$\phi^{*1}=14.7[\text{ns}](68.0[\text{MHz}])$
1	0	0	x 5 (multiplied by 5)	*2
1	0	1	x 6 (multiplied by 6)	Do not make a setting
1	1	0	x 7 (multiplied by 7)	Do not make a setting
1	1	1	x 8 (multiplied by 8)	Do not make a setting

*1: ϕ : Frequency of the system base clock

*2: Not supported by the MB91301 series (supported when the range of PLL output is 48 MHz to 68 MHz).

- These bits are initialized to "000_B" by a reset (INIT).
- These bits are readable and writable.

Note:

The upper-limit frequency for operation is 68 MHz. Do not make a setting exceeding this frequency.

[bit11] (Reserved)

This bit is reserved.

[bit10] PLL1EN (PLL1 ENable)

This bit is the enable bit of the main PLL.

Rewriting of this bit is disabled while the main PLL is selected as the clock source.

Selection of the main PLL as the clock source is disabled while this bit is set to "0" (because of the setting of bit9 and bit8, which are the CLKS1 and CLK0 bits).

Table 3.12-17 Function of the main PLL operation enable bit (PLL1EN)

PLL1EN	Function
0	Main PLL stopped (initial value)
1	Main PLL enabled

- This bit is initialized to "0" by a reset (INIT).
- This bit is readable and writable.

[bit9, bit8] CLKS1, CLKS0 (CLocK source Select)

These bits set the clock source that will be used by the MB91301 series.

The values written to these bits determine the clock source, which can be selected from the two types shown in Table 3.12-18.

Table 3.12-18 Clock Source Settings

CLKS1	CLKS0	Clock source setting
0	0	Source oscillation input from X0/X1 divided by 2 (initial value)
0	1	Source oscillation input from X0/X1 divided by 2
1	0	Main PLL
1	1	Do not make a setting

Table 3.12-19 shows the combinations of the CLKS1 and CLKS0 bits that cannot be changed and those that can.

Table 3.12-19 Combinations of CLKS1 and CLKS0 Bits that Can and Cannot Be Changed

Cannot be changed	Can be changed
"00 _B " --> "11 _B "	"00 _B " --> "01 _B " or "10 _B "
"01 _B " --> "10 _B "	"01 _B " --> "11 _B " or "00 _B "
"10 _B " --> "01 _B " or "11 _B "	"10 _B " --> "00 _B "
"11 _B " --> "00 _B " or "10 _B "	"11 _B " --> "01 _B "

- These bits are initialized to "00_B" by a reset (INIT).
- These bits are readable and writable.

■ Base Clock Division Setting Register 0 (DIVR0)

Figure 3.12-7 shows the configuration of the Base Clock Division Setting Register 0 (DIVR0) bits.

Figure 3.12-7 Configuration of Base Clock Division Setting Register 0 (DIVR0) Bits

bit	15	14	13	12	11	10	9	8
Address: 000486 _H	B3	B2	B1	B0	P3	P2	P1	P0
	R/W							
Initial value (INIT)	0	0	0	0	0	0	1	1
Initial value (RST)	X	X	X	X	X	X	X	X

Base Clock Division Setting Register 0 (DIVR0) controls the divide-by rate of an internal clock in relation to the base clock. This register sets the divide-by rates of the CPU clock, the clocks of an internal bus (CLKB) and a peripheral circuit, and the peripheral bus clock (CLKP).

An upper-limit frequency for the operation is set for each clock. If you set a combination of source clock, PLL multiply-by rate setting, and divide-by rate setting that results in a frequency exceeding this upper-limit frequency, operation is not guaranteed. Be extra careful of the order in which you change settings to select the source clock and to configure the associated setting items.

If the setting in this register is changed, the new divide-by rate takes effect for the clock rate following the one in which the setting was made.

[bit15 to bit12] B3, B2, B1, B0 (clkB divide select 3 to 0)

These bits are the clock divide-by rate setting bits of the CPU clock (CLKB). Set the clock divide-by rate of the CPU, internal memory, and internal bus clock (CLKB). The values written to these bits determine the divide-by rate (clock frequency) of the CPU and internal bus clock in relation to the base clock, which can be selected from the 16 types shown in Table 3.12-20.

Note:

The upper-limit frequency for operation is 68 MHz. Do not set a divide-by rate that results in a frequency exceeding this limit.

Table 3.12-20 Clock Divide-By Rate (CPU Clock) Settings

B3	B2	B1	B0	Clock divide-by rate	Clock frequency: if the source oscillation is 17 [MHz] and the PLL is multiplied by 4
0	0	0	0	ϕ	68 [MHz] (initial value)
0	0	0	1	$\phi \times 2$ (divided by 2)	34 [MHz]
0	0	1	0	$\phi \times 3$ (divided by 3)	22.7 [MHz]
0	0	1	1	$\phi \times 4$ (divided by 4)	17 [MHz]
0	1	0	0	$\phi \times 5$ (divided by 5)	13.6 [MHz]
0	1	0	1	$\phi \times 6$ (divided by 6)	11.3 [MHz]
0	1	1	0	$\phi \times 7$ (divided by 7)	9.71 [MHz]
0	1	1	1	$\phi \times 8$ (divided by 8)	8.5 [MHz]
...
1	1	1	1	$\phi \times 16$ (divided by 16)	4.25 [MHz]

Note: ϕ : Frequency of the system base clock

- These bits are initialized to "0000_B" by a reset (INIT).
- These bits are readable and writable.

[bit11 to bit8] P3, P2, P1, P0 (clkP divide select 3 to 0)

These bits are the clock divide-by rate setting bits of the peripheral clock (CLKP). Set the clock divide-by rate of the peripheral circuit and the peripheral bus clock (CLKP). The values written to these bits determine the divide-by rate (clock frequency) of the peripheral circuit and the peripheral bus clock in relation to the base clock, which can be selected from the 16 types shown in Table 3.12-21.

Table 3.12-21 Clock Divide-by Rate (Peripheral Clock) Settings

P3	P2	P1	P0	Clock divide-by rate	Clock frequency: if the source oscillation is 17 [MHz] and the PLL is multiplied by 4
0	0	0	0	ϕ	68 [MHz]
0	0	0	1	$\phi \times 2$ (divided by 2)	34 [MHz]
0	0	1	0	$\phi \times 3$ (divided by 3)	22.7 [MHz]
0	0	1	1	$\phi \times 4$ (divided by 4)	17 [MHz] (initial value)
0	1	0	0	$\phi \times 5$ (divided by 5)	13.6 [MHz]
0	1	0	1	$\phi \times 6$ (divided by 6)	11.3 [MHz]
0	1	1	0	$\phi \times 7$ (divided by 7)	9.71 [MHz]
0	1	1	1	$\phi \times 8$ (divided by 8)	8.5 [MHz]
...
1	1	1	1	$\phi \times 16$ (divided by 16)	4.25 [MHz]

Note: ϕ : Frequency of the system base clock

- These bits are initialized to "0011_B" by a reset (INIT).
- These bits are readable and writable.

Note:

The upper-limit frequency for operation is 34 MHz. Do not set a divide-by rate that results in a frequency exceeding this limit.

■ Base Clock Division Setting Register 1 (DIVR1)

Figure 3.12-8 shows the configuration of the Base Clock Division Setting Register 1 (DIVR1) bits.

Figure 3.12-8 Configuration of Base Clock Division Setting Register 1 (DIVR1) Bits

bit	15	14	13	12	11	10	9	8
Address: 00000487 _H	T3	T2	T1	T0	-	-	-	-
	R/W							
Initial value (INIT)	0	0	0	0	0	0	0	0
Initial value (RST)	X	X	X	X	X	X	X	X

Base Clock Division Setting Register 1 (DIVR1) controls the divide-by rate of an internal clock in relation to the base clock. This register sets the divide-by rates of the external extended bus interface clock (CLKT). An upper-limit frequency for operation is set for each clock. If you set a combination of source clock, PLL multiply-by rate setting, and divide-by rate setting that results in a frequency exceeding this upper-limit frequency, operation is not guaranteed. Be extra careful of the order in which you change settings to select the source clock and to configure the associated setting items.

If the setting in this register is changed, the new divide-by rate takes effect for the clock rate following the one in which the setting was made.

[bit7 to bit4] T3, T2, T1, T0 (clkT divide select 3 to 0)

These bits are the clock divide-by rate setting bits of the external bus clock (CLKT). Set the clock divide-by rate of the external extended bus interface clock (CLKT). The values written to these bits determine the divide-by rate (clock frequency) of the external extended bus interface clock in relation to the base clock, which can be selected from the 16 types shown in Table 3.12-22.

Table 3.12-22 Clock Divide-By Rate (External Bus Clock) Settings

T3	T2	T1	T0	Clock divide-by rate	Clock frequency: if the source oscillation is 17[MHz] and the PLL is multiplied by 4
0	0	0	0	ϕ	68 [MHz] (initial value)
0	0	0	1	$\phi \times 2$ (divided by 2)	34 [MHz]
0	0	1	0	$\phi \times 3$ (divided by 3)	22.7 [MHz]
0	0	1	1	$\phi \times 4$ (divided by 4)	17 [MHz]
0	1	0	0	$\phi \times 5$ (divided by 5)	13.6 [MHz]
0	1	0	1	$\phi \times 6$ (divided by 6)	11.3 [MHz]
0	1	1	0	$\phi \times 7$ (divided by 7)	9.71 [MHz]
0	1	1	1	$\phi \times 8$ (divided by 8)	8.5 [MHz]
...
1	1	1	1	$\phi \times 16$ (divided by 16)	4.25 [MHz]

Note: ϕ : Frequency of the system base clock

- These bits are readable and writable.

[bit3 to bit0] (Reserved)

These bits are reserved.

Note:

The upper-limit frequency for operation is 68 MHz. Do not set a divide-by rate that results in a frequency exceeding this limit.

3.12.7 Peripheral Circuits of Clock Controller

This section describes the peripheral circuit functions of the clock controller.

■ Time Base Counter

The clock controller has a 26-bit time base counter that runs on the system base clock.

The time base counter is used to measure the oscillation stabilization wait time in addition to having the uses listed below (For more information about the oscillation stabilization wait time, see Section "3.11.4 Oscillation Stabilization Wait Time").

- Watchdog timer

The watchdog timer, which is used to detect a system runaway, measures time using the bit output of the time base counter.

- Time-base timer

The time-base timer generates an interval interrupt using output from the time base counter.

The following describes these functions.

○ Watchdog timer

The watchdog timer detects a runaway using output from the time base counter. If a program runaway results in a watchdog reset no longer being postponed for a specified interval, a settings initialization reset (INIT) request is generated as a watchdog reset.

[Startup and interval setting of the watchdog timer]

The watchdog timer is started when the reset source register and the watchdog timer control register (RSRR) are written to for the first time after a reset (RST). At this time, the interval time of the watchdog timer is set in bit9 and bit8 (WT1 and WT0 bits). Only the time defined in this first write is valid as the interval time setting. Any further writing is ignored.

[Postponing a watchdog reset]

Once the watchdog timer is started, the program must write "A5H" and "5AH" in this order to the time base counter clear register (CTBR). This operation initializes the watchdog reset generation flag.

[Generation of a watchdog reset]

The watchdog reset generation flag is set at the trailing edge of the time base counter output of the specified interval. If the flag has already been set when a trailing edge is detected a second time, a settings initialization reset (INIT) request is generated as a watchdog reset.

[Stopping the watchdog timer]

The watchdog timer, once started, cannot be stopped until an operation initialization reset (RST) occurs.

In the following states, when an operation initialization reset (RST) occurs, the watchdog timer is stopped and remains inoperative until a program starts it.

- Operation initialization reset (RST) state
- Settings initialization reset (INIT) state
- Oscillation stabilization wait reset (RST) state

[Suspending the watchdog timer (automatic postponement)]

If program operation stops on the CPU, the watchdog reset generation flag is initialized and generation of a watchdog reset is postponed. Stopping of program operation specifically refers to the following statuses:

- Sleep state
- Stop state
- Oscillation stabilization wait RUN state
- During fetching instructions to the D-bus of D-bus RAM, etc.
- During breaking an emulator debugger and a monitor debugger in use
- Period of execution from INTE Instruction to RETI Instruction
- Step Trace Trap (the break by each I instruction caused by T Flag = 1 in the PS Register)
- During Data Access operation of cache memory at Instruction cache control register (ISIZE, ICHCR) or RAM Mode

If the time base counter is cleared, the watchdog reset generation flag is initialized at the same time, postponing generation of a watchdog reset.

If system falls into the condition mentioned above because of system runaway, the watchdog reset cannot be executed. In that case, execute initialization reset (INIT) from external INIT pin.

■ Time-base Timer

The time-base timer generates an interval using output from the time base counter. This timer is appropriate for measurements that require a relatively long time (for example, a maximum interval of {base clock $\times 2^{27}$ } cycles such as for the PLL lock wait time or a sub clock).

If the trailing edge of the time base counter output for the specified interval is detected, a time-base timer interrupt request is generated.

[Startup and interval settings of the time-base timer]

For the time-base timer, the interval time is set in bit13 to bit11 (TBC2, TBC1, and TBC0 bits) of the time base counter control register (TBCR). The trailing edge of the time base counter output for the specified interval is always detected. Thus, after setting the interval time, clear bit15 (TBIF bit) and then set bit14 (TBIE bit) to "1" to enable output of an interrupt request.

Before changing the interval time, set bit14 (TBIE bit) to "0" to disable interrupt request output.

Since the time base counter always counts regardless of these settings, before enabling interrupts, clear the time base counter to obtain an accurate interval interrupt time. Otherwise, an interrupt request may be generated immediately after an interrupt is enabled.

[Clearing of the time base counter due to a program]

If "A5_H" and "5A_H" are written in this order to the time base counter clear register (CTBR), all bits of the time base counter are cleared to "0" immediately after "5A_H" is written. There is no time limit between writing of "A5_H" and "5A_H". However, if data other than "5A_H" is written after "A5_H" is written, "A5_H" must be written again before "5A_H" is written. Otherwise, no clear operation occurs.

If the time base counter is cleared, the watchdog reset generation flag is initialized at the same time, postponing generation of a watchdog reset.

[Clearing of the time base counter due to the device state]

All bits of the time base counter are cleared to "0" at the same time if the device enters one of the following states:

- Stop state
- Settings initialization reset (INIT) state

Especially in the stop state, an interval interrupt of the time-base timer may unintentionally be generated because the time base counter is used to measure the oscillation stabilization wait time. Before setting stop mode, therefore, disable time-base timer interrupts to prevent the time-base timer from being used.

In any other state, time-base timer interrupts are automatically disabled because an operation initialization reset (RST) occurs.

3.12.8 Smooth Startup and Stop of Clock

This section explains the method to control the internal voltage effect and voltage surge.

■ Smooth Startup and Stop of Clock

Connect the bypass capacitor of about $5.0\mu\text{F}$ to the C pin to control the internal voltage effect or voltage surge so that the change of the internal voltage is controlled drastically. Also, switch all clocks (CPU, internal bus clock (CLKB), external bus clock (CLKT), peripheral circuit, and peripheral bus clock (CLKP)) gradually instead of switching to a desired frequency from the low frequency suddenly.

If returning to the operation in the low frequency, change it gradually and start and shut down the clock as follows.

When changing the operation from high frequency to low frequency, perform the same procedure.

○ Start up

- 1) Enable the PLL operation. (Set the PLL1EN bit of the clock source control register (CLKR) to "1".)
- 2) Oscillation stabilization wait time
- 3) Divide the CLKB, CLKT, and CLKP by 16. (Set the DIVR0 and DIVR1 registers.)
- 4) Set the multiply-by rate of the PLL and switch the X0 to PLL side. (Set the clock source control register (CLKR).)
- 5) Reduce the divide-by rate of the CLKB, CLKT, and CLKP by degrees. Insert the wait loop between the division steps.

○ Shut down

- 1) Divide the CLKB, CLKT, and CLKP gradually (number of steps depends on the frequency setting) up to the maximum division coefficient and insert the wait loop between the division steps. (Set the DIVR0 and DIVR1 registers.)
- 2) Change from the PLL to the source oscillation of the X0/X1. (Set the clock source control register (CLKR).)
- 3) Disable the PLL. (Set the PLL1EN bit of the clock source control register (CLKR) to "0".)

■ Program Example of Smooth Startup and Stop of Clock

○ Procedure of startup

```

#macro wait_loop loop_number
#local _wait64_loop
    ldi #loop_number,r0
    _wait64_loop:
        add   #-1,r0
        bne   _wait64_loop
#endm

smooth_up_start3:
    ldi    #_DIVR0,r1      // Division register for CLKB and CLKP
    ldi    #_DIVR1,r2      // Division register for CLKT
    ldi    #_CLKR,r3       // CLKR register
    ldi    #0xff,r4
    ldi    #0x11,r5
    ldi    #0x33,r6
    ldi    #0x77,r7
    ldi    #0x01,r8
    ldi    #0x34,r12
    ldi    #0x36,r13
    nop
    nop
    nop
    stb   r12,@r3          // PLL → X0 PLL operation enable
//Divide CLKB, CLKT, and CLKP by 16.
    stb   r4,@r1          // Divide CLKB and CLKP by 16
    stb   r4,@r2          // Divide CLKT by 16
    wait_loop 4
    stb   r13,@r3          //Switch X0 to PLL side.
    wait_loop 4

//Reduce the divide-by rate of CLKB and CLKP gradually.
    stb   r7,@r1          // Divide CLKB and CLKP by 16 → 8
    wait_loop 8
    stb   r6,@r1          // Divide CLKB and CLKP by 8 → 4
    wait_loop 8
    stb   r5,@r1          // Divide CLKB and CLKP by 4 → 2
    wait_loop 16
    stb   r8,@r1          // Divide CLKB by 2 → no division, divide CLKP by 2 → 2
    wait_loop 16

//Reduce the divide-by rate of CLKT gradually.
    stb   r7,@r2          // Divide CLKT by 16 → 8
    wait_loop 8
    stb   r6,@r2          // Divide CLKT by 8 → 4
    wait_loop 8
    stb   r5,@r2          // Divide CLKT by 4 → 2
    wait_loop 16
    stb   r8,@r2          // No CLKT division
    wait_loop 16

```

○ Procedure of shut down

```

#macro wait_loop loop_number
#local _wait64_loop
    ldi    #loop_number,r0
    _wait64_loop:
        add   #-1,r0
        bne  _wait64_loop
#endm

smooth_down_start3:
    ldi    #_DIVR0,r1      // Division register for CLKB and CLKP
    ldi    #_DIVR1,r2      // Division register for CLKT
    ldi    #_CLKR,r3       // CLKR register
    ldi    #0x11,r5
    ldi    #0x3f,r6
    ldi    #0xff,r8
    ldi    #0x04,r9
    ldi    #0x33,r10
    ldi    #0xff,r12
    ldi    #0x00,r13
    ldi    #0x1f,r14
    nop
    nop
    nop
//Increase the divide-by rate of CLKT gradually.
    stb   r14,@r2    // No CLKT division → 2
    wait_loop 16
    stb   r6,@r2    // Divide CLKT by 2 → 4
    wait_loop 8
    stb   r8,@r2    // Divide CLKT by 4 → 16
    wait_loop 4

//Increase the divide-by rate of CLKB and CLKP gradually.
    stb   r5,@r1    // No CLKB division → by 2, divide CLKP by 2 → 2
    wait_loop 16
    stb   r10,@r1   // Divide CLKB and CLKP by 2 → 4
    wait_loop 8
    stb   r12,@r1   //Divide CLKB and CLKP by 4 → 16
    wait_loop 4

//Reduce the multiply-by rate of the PLL gradually.

    stb   r9,@r3    // Change from the PLL to the source oscillation of the X0/X1.
    stb   r13,@r3   // PLL off
    nop
    nop
    nop
    ret

```

3.13 Device State Control

This section describes the states of the FR family and their control. It also describes low-power mode.

■ Device States

The FR family has the operating states listed below.

For more information about these states, see Section "3.13.1 Device States and State Transitions".

- RUN state (normal operation)
- Sleep state
- Stop state
- Oscillation stabilization wait RUN state
- Oscillation stabilization wait reset (RST) state
- Operation initialization reset (RST) state
- Settings initialization reset (INIT) state

■ Low-power Modes

The following two low-power modes are provided.

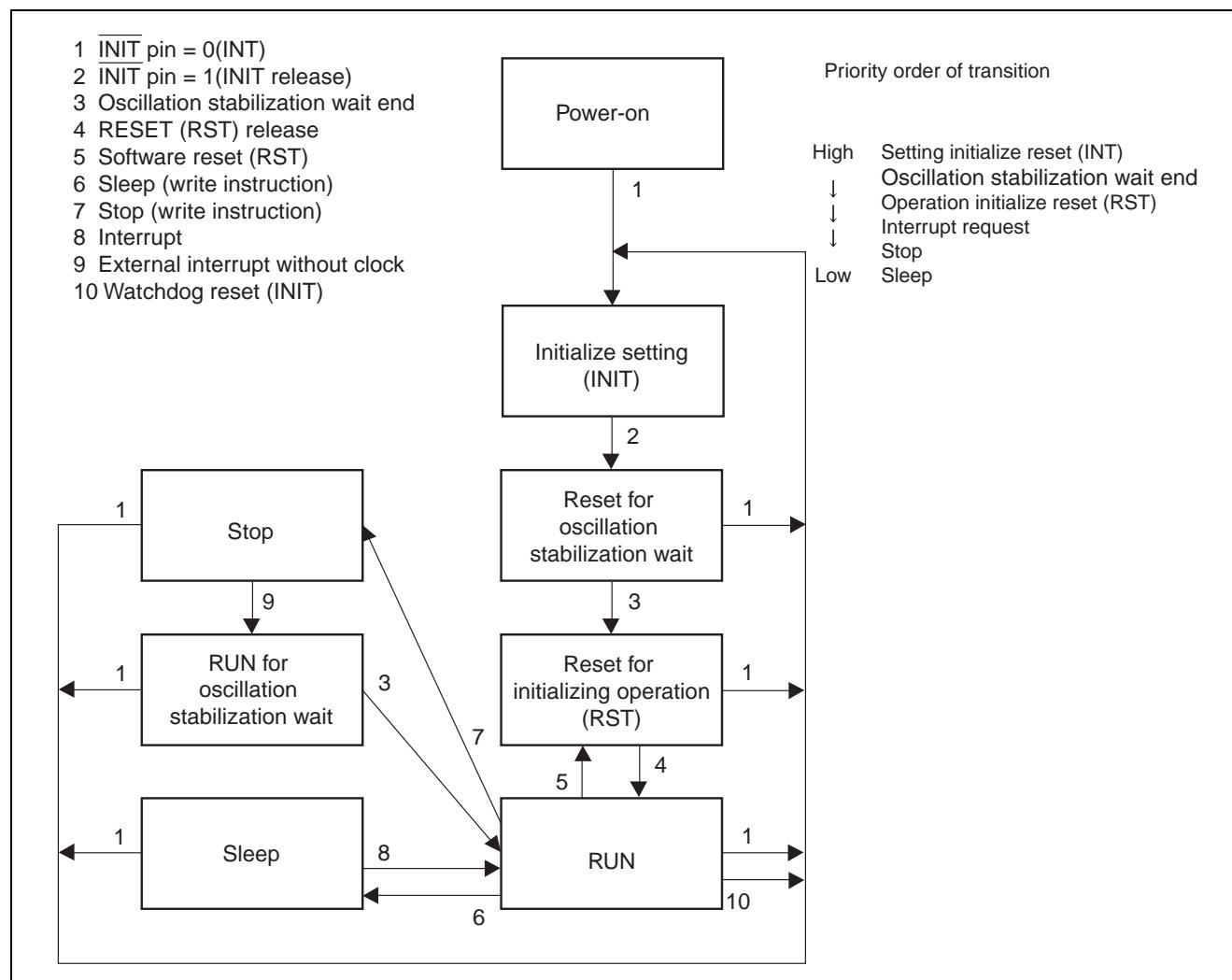
For more information, see Section "3.13.2 Low-power Modes".

- Sleep mode
- Stop mode

CHAPTER 3 CPU AND CONTROL UNITS

■ State of Device and Each Transition

The following shows the device state transitions of the MB91301 series.



3.13.1 Device States and State Transitions

This section describes device operating states and the transition between operating states.

■ RUN State (Normal Operation)

In the RUN state, a program is being executed. All internal clocks are supplied and all circuits are enabled.

For the 16-bit peripheral bus, however, only the bus clock is stopped, when it is not being accessed.

In this state, a state transition request is accepted. If synchronous reset mode is selected, however, state transition operations different from normal reset mode are used for some requests. For more information, see "Synchronous Reset Operation" in Section "3.11.5 Reset Operation Modes".

■ Sleep State

In the sleep state, a program is stopped. Program operation causes a transition to this state.

Only execution of the program on the CPU is stopped; peripheral circuits are enabled. The instruction cache is stopped and the built-in memory modules and the internal and external buses are stopped unless the DMA controller issues a request.

- If a valid interrupt request occurs, the state is cleared and the RUN state (normal operation) is entered.
- If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is entered.
- If an operation initialization reset (RST) request occurs, the operation initialization reset (RST) state is entered.

■ Stop State

In the stop state, the device is stopped. Program operation causes a transition to this state.

All internal circuits are stopped. All internal clocks are stopped and the oscillation circuit and PLL can be stopped if set to do so. Be sure to set PLL1EN=0 before entering the stop state.

In addition, the external pins (except some) can be set to high impedance via settings.

- If a specific valid interrupt request (no clock required) occurs, the oscillation stabilization wait RUN state is entered.
- If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is entered.
- If an operation initialization reset (RST) request occurs, the oscillation stabilization wait reset (RST) state is entered.

■ Oscillation Stabilization Wait RUN State

In the oscillation stabilization wait RUN state, the device is stopped. This state occurs after a return from the stop state.

All internal circuits except the clock generation controller (time base counter and device state controller) are stopped. All internal clocks are stopped, but the oscillation circuit and the PLL that has been enabled are running.

- High impedance control of external pins in the stop or other state is cleared.
- If the specified oscillation stabilization wait time elapses, the RUN state (normal operation) is entered.
- If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is entered.
- If an operation initialization reset (RST) request occurs, the oscillation stabilization wait reset (RST) state is entered.

■ Oscillation Stabilization Wait Reset (RST) Status

In the oscillation stabilization wait reset (RST) state, the device is stopped. This state occurs after a return from the stop state or the settings initialization reset (INIT) state. All internal circuits except the clock generation controller (time base counter and device state controller) are stopped. All internal clocks are stopped, but the oscillation circuit and the PLL that has been enabled are running.

- High impedance control of external pins in the stop state, etc., is cleared.
- An operation initialization reset (RST) is output to the internal circuits.
- If the specified oscillation stabilization wait time elapses, the oscillation stabilization wait reset (RST) state is entered.
- If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is entered.

■ Operation Initialization Reset (RST) State

In the operation initialization reset (RST) state, a program is initialized. This state occurs if an operation initialization reset (RST) request is accepted or the oscillation stabilization wait reset (RST) state is ended.

Execution of a program on the CPU is stopped and the program counter is initialized. Most peripheral circuits are initialized. All internal clocks are stopped, but the oscillation circuit and the PLL that has been enabled are running.

- An operation initialization reset (RST) is output to the internal circuits.
- If an operation initialization reset (RST) no longer exists, the RUN state (normal operation) is entered and the operation initialization reset sequence is executed. After a return from the settings initialization reset (INIT), the settings initialization reset sequence is executed.
- If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is entered.

■ Settings Initialization Reset (INIT) State

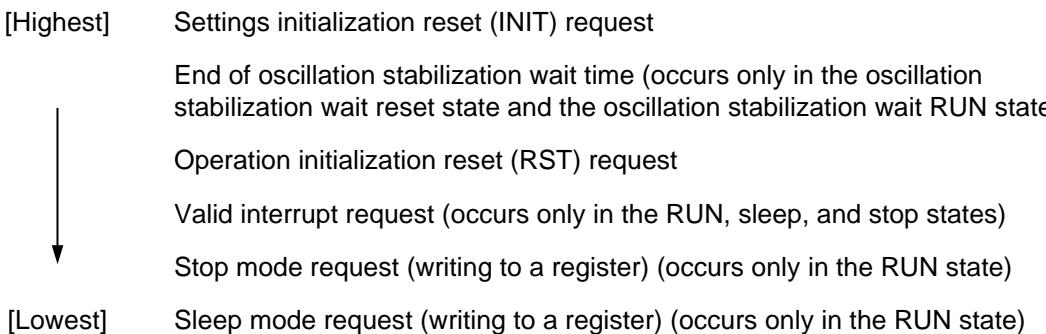
In the settings initialization reset (INIT) state, all settings are initialized. This state occurs if a settings initialization reset (INIT) is accepted.

Execution of a program on the CPU is stopped and the program counter is initialized. All peripheral circuits are initialized. The oscillation circuit runs, but the PLL stops running. All internal clocks are stopped while the "L" level is input to the external $\overline{\text{INIT}}$ pin; otherwise, they run.

- A settings initialization reset (INIT) and an operation initialization reset (RST) are output to the internal circuits.
- If a settings initialization reset (INIT) no longer exists, the state is cleared and the oscillation stabilization wait reset (RST) state is entered. Then, the operation initialization reset (RST) state is entered and the settings initialization reset sequence is executed.

■ Priority of State Transition Requests

In any state, state transition requests conform to the priority listed below. However, some requests that occur only in a specific state are valid only in that state.



3.13.2 Low-power Modes

This section describes the low-power modes, some MB91301 series states, and how to use the low-power modes.

■ Low-power Modes

The FR family has the following low-power modes:

- Sleep mode: The device enters the sleep state due to writing to a register.
- Stop mode: The device enters the stop state due to writing to a register.

These modes are described below.

■ Sleep Mode

If "1" is set for bit6 (SLEEP bit) of the standby control register (STCR), sleep mode is initiated and the device enters the sleep state. The sleep state is maintained until a source for return from the sleep state is generated.

If "1" is set for both bit7 (STOP bit) and bit6 of the standby control register (STCR), bit7 (STOP bit) has precedence and the device enters the stop state.

For more information about the sleep state, see "Sleep State" in Section "3.13.1 Device States and State Transitions".

[Sleep mode transition]

Use the following sequences after using the synchronous standby mode (TBCR:Set by time base counter control register bit8 SYNCS bit) when putting in the sleep mode.

```
/* Write to STCR */
ldi    #_STCR, r0      ; STCR register (481H)
ldi    #Val_of_Stby, r1 ; "Val_of_Stby" is the data to be written to STCR.
stb    r1, @r0          ; Write to STCR.

/* Write to CTBR */
ldi    #_CTBR, r2      ; CTBR register (483H)
ldi    #0xA5, r1        ; Clear command (1)
stb    r1, @r2          ; Write A5 to CTBR.
ldi    #0x5A, r1        ; Clear command (2)
stb    r1, @r2          ; Write 5A to CTBR.

/* Clear the time base counter at this point */
ldub   @r0, r1          ; Read from STCR.

/* Start moving to synchronous standby mode */
ldub   @r0, r1          ; Dummy read from STCR.
nop               ; NOP for timing adjustment x 5
nop
nop
nop
nop
```

○ **Circuits that stop in the sleep state**

- Program execution on the CPU
- Bit search module (enabled if DMA transfer occurs)
- Various built-in memory (enabled if DMA transfer occurs)
- Internal types of and external buses (enabled if DMA transfer occurs. A bus request is enabled.)

○ **Circuits that do not stop in the sleep state**

- Oscillation circuit
- PLL that has been enabled
- Clock generation controller
- Interrupt controller
- Peripheral circuit
- DMA controller

○ **Sources of return from the sleep state**

- Generation of a valid interrupt request

If the ICR register is not set to "11111_B" and an interrupt request occurs, sleep mode is cleared and the RUN state (normal operation) is entered. If the ICR register is set to "11111_B" and an interrupt request occurs, sleep mode is not cleared.

- Generation of a settings initialization reset (INIT) request

If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) state is unconditionally entered.

- Generation of an operation initialization reset (RST)

If an operation initialization reset (RST) occurs, the operation initialization reset (RST) state is unconditionally entered.

For information about the priority of sources, see "Priority of State Transition Requests" in Section "3.13.1 Device States and State Transitions".

○ **Synchronous standby operations**

If "1" is set for bit8 (SYNCS bit) of the time base counter control register (TBCR), synchronous standby operation is enabled. In this case, simply writing to the SLEEP bit does not cause a transition to the sleep state. Instead, writing to the SLEEP bit and then reading the STCR register causes a transition to the sleep state.

Always use the sequences described in "■ Sleep Mode" in "3.13.2 Low-power Modes" when using the sleep mode.

■ Stop Mode

If "1" is set for bit7 (STOP bit) of the standby control register (STCR), stop mode is initiated and the device enters the stop state. The stop state is maintained until a source for return from the stop state occurs.

If "1" is set for both bit6 (SLEEP bit) and bit7 bit of the standby control register (STCR), bit7 (STOP bit) has precedence and the device enters the stop state.

For more information about the stop state, see "Stop State" in Section "3.13.1 Device States and State Transitions".

[Stop mode transition]

Use the following sequences after using the synchronous standby mode (TBCR:Set by time base counter control register bit8 SYNC8 bit) when putting in the stop mode.

```
(LDI #value_of_standby, R0) ; value_of_standby is the writing data to STCR.  
(LDI #_STCR, R12) ; _STCR is the address (481H) of STCR.  
STB R0, @R12 ; Writing in standby control register (STCR)  
LDUB @R12, R0 ; STCR read for synchronous standby  
LDUB @R12, R0 ; Dummy re-read of STCR  
NOP ; five NOPs for timing adjustment  
NOP  
NOP  
NOP  
NOP
```

Furthermore, set the I flag, ILM and ICR to ensure that branching into the interrupt handler which is a return source occurs after returning to the standby mode.

○ Circuits that stop in the stop state

- Oscillation circuits set to stop
- If "1" is set for bit0 (OSCD1 bit) of the standby control register (STCR), the main clock oscillation circuit in the stop state is stopped.
- PLL connected to the oscillation circuit that is either disabled or set to stop
- If "1" is set for bit0 (OSCD1 bit) of the standby control register (STCR) and "1" is set for bit10 (PLL1EN bit) of the clock source control register (CLKR), the main clock PLL in the stop state is stopped.
- All internal circuits except those, described below, that do not stop in the stop state

○ Circuits that do not stop in the stop state

- Oscillation circuits that are set not to stop
 - If "0" is set for bit0 (OSCD1 bit) of the standby control register (STCR), the main clock oscillation circuit in the stop state is not stopped.
- PLL connected to the oscillation circuit that is enabled and is not set to stop
 - If "0" is set for bit0 (OSCD1 bit) of the standby control register (STCR) and "1" is set for bit10 (PLL1EN bit) of the clock source control register (CLKR), the main clock PLL in the stop state is not stopped.

○ **High impedance control of a pin in the stop state**

- If "1" is set for bit5 (HIZ bit) of the standby control register (STCR), the output of a pin in the stop state is set to the high impedance state. For information about pins subject to this control, see the appendix, "STATUS OF PINS IN THE CPU STATES".
- If "0" is set for bit5 (HIZ bit) of the standby control register (STCR), the output of a pin in the stop state maintains the value before transition to the stop state. For more information, see the appendix, "STATUS OF PINS IN THE CPU STATES".

○ **Sources of return from the stop state**

- Generation of a specific valid interrupt request (no clock required)

The external interrupt input pins (INT0 to INT7 pins) are enabled. If the ICR register is not set to " 11111_B " and an interrupt request occurs, stop mode is cleared and the RUN state (normal operation) is entered. If the ICR register is set to " 11111_B " and an interrupt request occurs, stop mode is not cleared.

- Generation of a settings initialization reset (INIT) request

If a settings initialization reset (INIT) request occurs, the settings initialization reset (INIT) is unconditionally entered.

For information about the priority of sources, see "Priority of State Transition Requests" in Section "3.13.1 Device States and State Transitions".

○ **Selecting a clock source in stop mode**

In self-induced oscillation mode, select the main clock divided by 2 as the source clock before setting stop mode. For more information, see Section "3.12 Clock Generation Control" especially Section "3.12.1 PLL Controls".

The same limitations as in the normal operation apply to the setting of a divide-by rate.

○ **Normal and synchronous standby operations**

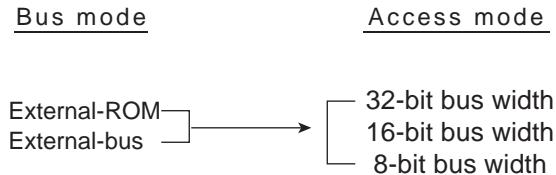
If "1" is set for bit8 (SYNCS bit) of the time base counter control register (TBCR), synchronous standby operation is enabled. In this case, simply writing to the SLEEP bit does not cause a transition to the sleep state. Instead, writing to the SLEEP bit and then reading the STCR register causes a transition to the sleep state.

Always use the sequences described in "■ Sleep Mode" in "3.13.2 Low-power Modes" when using the sleep mode.

3.14 Operating Modes

Two operating modes are provided: bus mode and access mode. This section describes these modes.

■ Operating Modes



○ Bus mode

Bus mode refers to a mode in which the operations of internal ROM and the external access function are controlled. A bus mode is specified using the setting pins (MD2, MD1, and MD0) and the ROMA bit in the mode data.

○ Access mode

An access mode refers to a mode that controls the external data bus width. An access mode is specified using the WTH1 and WTH0 bits in the mode register and the DBW1 and DBW0 bits in ACR0 to ACR7 (Area Configuration Register0 to 7).

■ Bus Modes

The FR family has the following three modes: bus mode 0 (single chip mode), bus mode 1 (internal-ROM/external-bus mode), and bus mode 2 (external-ROM/external-bus mode). The bus mode 2 (external-ROM/external-bus mode) can only be set for MB91V301A.

All bus modes can be set for MB91302A.

For more information, see Section "3.1 Memory Space".

○ Bus mode 0 (single chip mode)

The internal I/O, D-busRAM, F-busRAM, and F-busROM are valid, while access to any other areas is invalid under this mode.

The external pins serve as peripherals or general-purpose ports. The pin does not work as a bus pin.

○ Bus Mode 1 (internal-ROM/external-bus mode)

In this mode, the internal I/O, D-busRAM, F-busRAM, and F-busROM are valid. Access to an area that enables external access is handled as access to an external space. Some external pins serve as bus pins.

○ Bus Mode 2 (external-ROM/external-bus mode)

In this mode, the internal I/O, and D-busRAM are valid. All accesses are handled as access to an external space. Some external pins serve as bus pins.

■ Mode Settings

For the FR family, set the operating mode using the mode pins (MD2, MD1, and MD0) and the mode register (MODR).

○ Mode pins

Use the three mode pins (MD2, MD1, and MD0) to specify mode vector fetch and to set a test mode.

Table 3.14-1 Mode Settings

Mode pin			Mode name	Reset vector access area	Remarks
MD2	MD1	MD0			
0	0	0	Internal ROM mode vector	Internal	Single chip mode
0	0	1	External ROM mode vector	External	Set the bus width using the mode register.

Note that any setting other than those listed in the table is not allowed.

○ Mode register (MODR)

The mode register (MODR: MODE Register) determines the operating mode.

Figure 3.14-1 shows the configuration of the mode register (MODR).

Figure 3.14-1 Configuration of the Mode Register (MODR)

Address	bit	Operation mode setting bit							Initial value
		23	22	21	20	19	18	17	16
0007FD _H	-	-	-	-	-	ROMA	WTH1	WTH0	XXXXXXX _B

The data to be written to the mode register using a mode vector fetch (see "3.11.3 Reset Sequence") is called "mode data".

Once this data is written in the mode register (MODR), the operation is performed according to the operation mode specified by this register. The mode register (MODR) is set by all reset factors.

In addition, the data cannot be written from a user program.

- Details of mode data

Figure 3.14-2 details the mode data to be set in the mode vector.

Figure 3.14-2 Details of mode data

Address	bit	Operation mode setting bit							Initial value
		31	30	29	28	27	26	25	24
0007FD _H	-	-	-	-	-	ROMA	WTH1	WTH0	XXXXXXX _B

The following explains the functions of the bits in the mode register (MODR).

[bit31 to bit27] (Reserved)

These bits are reserved. Be sure to set them to "00000_B". If you set the other value of "00000_B", the operation is not guaranteed.

[bit26] ROMA (Internal ROM enable bit)

This bit sets whether to making built-in F-bus RAM and F-bus ROM region effective.

Table 3.14-2 Function of internal ROM enable

ROMA	Function	Remark
0	External ROM mode	Built-in F-bus region (40000_H to 100000_H) becomes an external region.
1	Internal ROM mode	Built-in F-bus region (40000_H to 100000_H) becomes access prohibited.*

*: EVA product has the built-in 8K bytes RAM. So the EVA product can be set. For details, see "Memory Map" in "3.1 Memory Space".

[bit25, bit24] WTH1, WTH0 (Bus width setting bit)

These bits specify the bus width to be used in external bus mode.

In external bus mode, this value is set in the DBW1 and DBW0 bits in ACR0H (the CS0 area).

Table 3.14-3 shows the settings for the initial bus width.

Table 3.14-3 Settings of the Initial Bus Width

WTH1	WTH0	Bus width	Remarks
0	0	8 bits	External bus mode
0	1	16 bits	External bus mode
1	0	32 bits	External bus mode
1	1	Single chip mode	

Note:

The mode data for setting mode vector should be set at $000FFFF8_H$ as byte data. FR family's byte endian uses big endian. Please set to the upper byte of bit31 to bit24.

Figure 3.14-3 Note on Mode Data

False	000FFFF8 _H	bit 31	24 23	16 15	8 7	0
		XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	Mode Data	
True	000FFFF8 _H	bit 31	24 23	16 15	8 7	0
		Mode Data	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXX	
	000FFFC _H	Reset Vector				

CHAPTER 4 EXTERNAL BUS INTERFACE

The external bus interface controller controls the interfaces with the internal bus for chips and with external memory and I/O devices.

This chapter explains each function of the external bus interface and its operation.

- 4.1 Overview of the External Bus Interface
- 4.2 External Bus Interface Registers
- 4.3 Setting Example of the Chip Select Area
- 4.4 Endian and Bus Access
- 4.5 Operation of the Ordinary Bus Interface
- 4.6 Burst Access Operation
- 4.7 Address/data Multiplex Interface
- 4.8 Prefetch Operation
- 4.9 SDRAM/FCRAM Interface Operation
- 4.10 DMA Access Operation
- 4.11 Bus Arbitration
- 4.12 Procedure for Setting a Register
- 4.13 Notes on Using the External Bus Interface

4.1 Overview of the External Bus Interface

This section explains the features, block diagram, I/O pins, and registers of the external bus interface.

■ Features

The external bus interface has the following features:

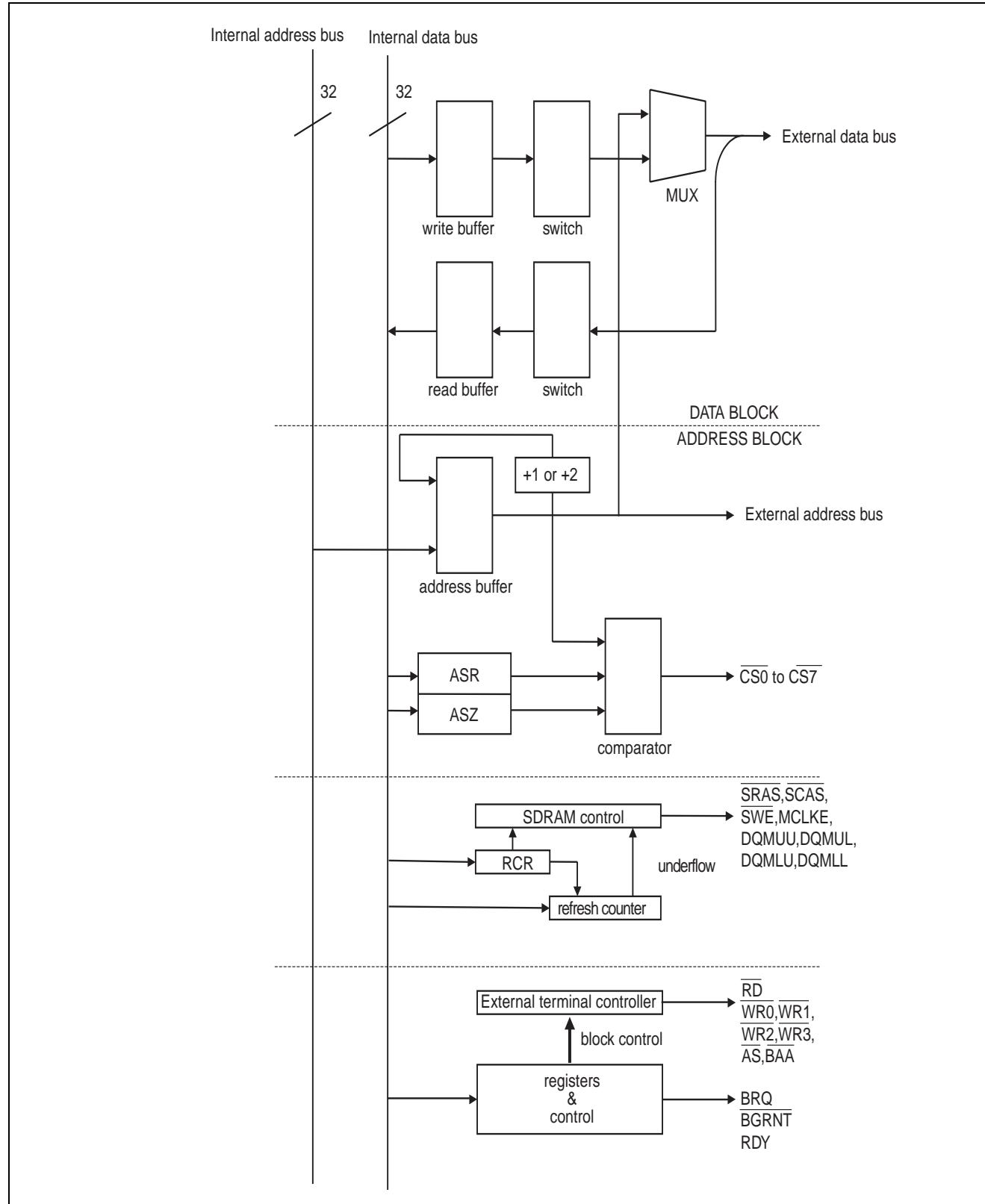
- Addresses of up to 32 bits (4 Gbytes space) can be output.
- Various kinds of external memory (8-bit/16-bit/32-bit modules) can be directly connected and multiple access timings can be mixed and controlled.
 - Asynchronous SRAM and asynchronous ROM/FLASH memory (multiple write strobe method or byte enable method)
 - Page mode ROM/FLASH memory (Page sizes 2, 4, and 8 can be used)
 - Burst mode ROM/FLASH memory
 - SDRAM (FCRAM modules are also supported, including two- and four-bank types with CAS latency 1 to 8)
 - Address/data multiplex bus (8-bit/16-bit width only)
 - Synchronous memory (such as ASIC built-in memory) (Synchronous SRAM cannot be directly connected)
- Eight independent banks (chip select areas) can be set, and chip select corresponding to each bank can be output.
 - The size of each area can be set in multiples of 64K bytes (64K bytes to 2G bytes for each chip select area).
 - An area can be set at any location in the logical address space (Boundaries may be limited depending on the size of the area.)
- In each chip select area, the following functions can be set independently:
 - Enabling and disabling of the chip select area (Disabled areas cannot be accessed)
 - Setting of the access timing type to support various kinds of memory (SDRAM can be connected to the $\overline{CS6}$ and $\overline{CS7}$ areas only.)
 - Detailed access timing setting (individual setting of the access type such as the wait cycle)
 - Setting of the data bus width (8-bit/16-bit/32-bit)
 - Setting of the order of bytes (big endian or little endian) (Only big endian can be set for the $\overline{CS0}$ area)
 - Setting of write disable (read-only area)
 - Enabling and disabling of fetches from the built-in cache
 - Enabling and disabling of the prefetch function
 - Maximum burst length setting (1, 2, 4, 8)

- **A different detailed timing can be set for each access timing type.**
 - For the same type of access timing, a different setting can be made in each chip select area.
 - Auto-wait can be set to up to 15 cycles (asynchronous SRAM, ROM, Flash, and I/O area).
 - The bus cycle can be extended by external RDY input (asynchronous SRAM, ROM, Flash, and I/O area).
 - The first access wait and page wait can be set (burst, page mode, and ROM/FLASH area).
 - Various kinds of idle/recovery cycles and setting delays can be inserted.
 - Capable of setting timing values such as the CAS latency and RAS-CAS delay (SDRAM area)
 - Capable of controlling the distributed/centralized auto-refresh, self-refresh, and other refresh timings (SDRAM area)
- **Fly-by transfer by DMA can be performed.**
 - Transfer between memory (including SDRAM) and I/O can be performed in a single access operation.
 - The memory wait cycle can be synchronized with the I/O wait cycle in fly-by transfer.
 - The hold time can be secured by only extending transfer source access.
 - Idle/recovery cycles specific to fly-by transfer can be set.
- **External bus arbitration using BRQ and BGRNT can be performed.**
- **Pins that are not used by the external interface can be used as general-purpose I/O ports through settings.**

CHAPTER 4 EXTERNAL BUS INTERFACE

■ Block Diagram

Figure 4.1-1 Block Diagram of the External Bus Interface



■ I/O Pins

The I/O pins are external bus interface pins (Some pins have other uses).

The following lists the I/O pins for each interface:

○ Ordinary bus interface

- A31 to A00, D31 to D00 (AD15 to AD00)
- $\overline{CS_0}$, $\overline{CS_1}$, $\overline{CS_2}$, $\overline{CS_3}$, $\overline{CS_4}$, $\overline{CS_5}$, $\overline{CS_6}$, $\overline{CS_7}$
- \overline{AS} , SYSCLK, MCLK
- \overline{RD}
- \overline{WR} , $\overline{WR0}$ (UUBX), $\overline{WR1}$ (ULBX), $\overline{WR2}$ (ULBX), $\overline{WR3}$ (LLBX)
- RDY, BRQ, \overline{BGRNT}

○ Memory interface

- MCLK, MCLKE
- $\overline{LBA} (= \overline{AS})$, \overline{BAA} (for burst ROM/FLASH)
- \overline{SRAS} , \overline{SCAS} , \overline{SWE} ($= \overline{WR}$) (for SDRAM)
- DQMUU,DQMUL,DQMLU,DQMLL (for SDRAM ($= \overline{WR0}$, $\overline{WR1}$, $\overline{WR2}$, $\overline{WR3}$))

○ DMA interface

- \overline{IOWR} , \overline{IORD}
- DACK0, DACK1
- DREQ0, DREQ1
- DEOP0, DEOP1

CHAPTER 4 EXTERNAL BUS INTERFACE

■ Register List

Figure 4.1-2 shows the registers used by the external bus interface:

Figure 4.1-2 List of External Bus Interface Registers

bit	31	24 23	16 15	8 7	0	
		ASR0		ACR0		
		ASR1		ACR1		
		ASR2		ACR2		
		ASR3		ACR3		
		ASR4		ACR4		
		ASR5		ACR5		
		ASR6		ACR6		
		ASR7		ACR7		
		AWR0		AWR1		
		AWR2		AWR3		
		AWR4		AWR5		
		AWR6		AWR7		
	MCRA	MCRB	Reserved	Reserved		Memory Setting Register (For mode without SDRAM/FCRAM auto pre-charge) (MCRA)
	Reserved	Reserved	Reserved	Reserved		Memory Setting Register (For mode with FCRAM auto pre-charge) (MCRB)
	IOWR0	IOWR1	Reserved	Reserved		DMAC I/O Wait Register (IOWR0, IOWR1)
	Reserved	Reserved	Reserved	Reserved		Chip Select Area Enable Register (CSER)
	CSER	CHER	Reserved	TCR		Cache Enable Register (CHER)
	Reserved	Reserved	Reserved	Reserved		Pin/Timing Control Register (TCR)
	Reserved	Reserved	Reserved	Reserved		Refresh Control Register (RCR)
	Reserved	Reserved	Reserved	Reserved		
	Reserved	(MODR)	Reserved	Reserved		

Note: Reserved indicates a reserved register. Be sure to set "0".
MODR cannot be accessed from user programs.

4.2 External Bus Interface Registers

This section explains the registers used in the external bus interface.

■ Register Types

The following registers are used by the external bus interface:

- Area select registers (ASR0 to ASR7)
- Area configuration registers (ACR0 to ACR7)
- Area wait registers (AWR0 to AWR7)
- Memory configuration register (MCRA for SDRAM/FCRAM auto-precharge OFF mode)
- Memory configuration register (MCRB for FCRAM auto-precharge ON mode)
- I/O wait registers for DMAC (IOWR0, IOWR1)
- Chip select enable register (CSER)
- Cache enable register (CHER)
- Pin/timing control register (TCR)
- Refresh control register (RCR)

4.2.1 Area Select Registers 0 to 7(ASR0 to ASR7)

This section explains the configuration and functions of area select registers 0 to 7 (ASR0 to ASR7).

■ Configuration of Area Select Registers 0 to 7 (ASR0 to ASR7)

The area select registers (ASR0 to ASR7: Area Select Registers 0 to 7) specify the start address of each chip select area of $\overline{CS0}$ to $\overline{CS7}$.

Figure 4.2-1 shows the configuration of area select registers 0 to 7 (ASR0 to ASR7: Area Select Register).

Figure 4.2-1 Configuration of the Area Select Registers (ASR0 to ASR7)

										Initial value		
ASR0	bit	15	14	13	12	...	2	1	0	INIT	RST	Access
	Address	000640H	A31	A30	A29	...	A18	A17	A16	0000H	0000H	R/W
ASR1	bit	15	14	13	12	...	2	1	0			
	Address	000644H	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR2	bit	15	14	13	12	...	2	1	0			
	Address	000648H	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR3	bit	15	14	13	12	...	2	1	0			
	Address	00064CH	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR4	bit	15	14	13	12	...	2	1	0			
	Address	000650H	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR5	bit	15	14	13	12	...	2	1	0			
	Address	000654H	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR6	bit	15	14	13	12	...	2	1	0			
	Address	000658H	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W
ASR7	bit	15	14	13	12	...	2	1	0			
	Address	00065CH	A31	A30	A29	...	A18	A17	A16	XXXXH	XXXXH	R/W

■ Functions of Bits in the Area Select Registers (ASR0 to ASR7)

The area select registers (ASR0 to ASR7) specify the start address of each chip select area (CS0 to CS7).

The start address can be set in the high-order 16 bits (bits A31 to A16). Each chip select area starts with the address set in this register and covers the range set by the four bits ASZ3 to ASZ0 of the ACR0 to ACR7 registers.

The boundary of each chip select area obeys the setting of the four bits ASZ3 to ASZ0 of the ACR0 to ACR7 registers. For example, if an area of 1 Mbyte is set by the four bits ASZ3 to ASZ0, the low-order four bits of the ASR0 to ASR7 registers are ignored and only bits A31 to A20 are valid.

The ASR0 register is initialized to 0000_H by INIT and RST. ASR1 to ASR7 are not initialized by INIT and RST, and are therefore undefined. After starting chip operation, be sure to set the corresponding ASR register before enabling each chip select area with the CSER register.

4.2.2 Area Configuration Registers 0 to 7 (ACR0 to ACR7)

This section explains the configuration and functions of area configuration registers 0 to 7 (ACR0 to ACR7).

■ Configuration of Area Configuration Registers 0 to 7 (ACR0 to ACR7)

The area configuration registers 0 to 7 (ACR0 to ACR7: FArea Configuration Register 0 to 7) set the function of each chip select area.

Figure 4.2-2 shows the configuration of area configuration registers 0 to 7 (ACR0 to ACR7).

Figure 4.2-2 Configuration of Area Configuration Registers 0 to 7 (ACR0 to ACR7)

										Initial value										
ACR0H	bit	15	14	13	12	11	10	9	8	INIT	RST	Access								
Address	000642H	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	1111XX00 _B	1111XX00 _B	R/W								
ACR0L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	0	TYP3	TYP2	TYP1	TYP0	00000000 _B	00000000 _B	R/W
Address	000643H																			
ACR1H	bit	15	14	13	12	11	10	9	8	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000646H																			
ACR1L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000647H																			
ACR2H	bit	15	14	13	12	11	10	9	8	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00064AH																			
ACR2L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00064BH																			
ACR3H	bit	15	14	13	12	11	10	9	8	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00064EH																			
ACR3L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00064FH																			
ACR4H	bit	15	14	13	12	11	10	9	8	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000652H																			
ACR4L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000653H																			
ACR5H	bit	15	14	13	12	11	10	9	8	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000656H																			
ACR5L	bit	7	6	5	4	3	2	1	0	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000657H																			

(Continued)

(Continued)

										Initial value		
ACR6H	bit	15	14	13	12	11	10	9	8	INIT	RST	Access
Address 00065A _H		ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXX _B	XXXXXXXX _B	R/W
ACR6L	bit	7	6	5	4	3	2	1	0			
Address 00065B _H		SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXX _B	XXXXXXXX _B	R/W
ACR7H	bit	15	14	13	12	11	10	9	8			
Address 00065E _H		ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	XXXXXXXX _B	XXXXXXXX _B	R/W
ACR7L	bit	7	6	5	4	3	2	1	0			
Address 00065F _H		SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	XXXXXXXX _B	XXXXXXXX _B	R/W

Note:

Please set ASR and ACR by word access at the same time.

Please set ACR after setting ASR if ASR and ACR are accessed by halfword.

The following explains the function of each bit:

[bit15 to bit12] ASZ3 to ASZ0 (Area Size Bits 3 to 0)

These bits set the area size. Table 4.2-1 shows their settings.

Table 4.2-1 Area Size Settings

ASZ3	ASZ2	ASZ1	ASZ0	Size of each chip select area
0	0	0	0	64 Kbytes (00010000 _H byte, ASR A[31:16] bits are valid)
0	0	0	1	128 Kbytes (00020000 _H byte, ASR A[31:17] bits are valid)
0	0	1	0	256 Kbytes (00040000 _H byte, ASR A[31:18] bits are valid)
0	0	1	1	512 Kbytes (00080000 _H byte, ASR A[31:19] bits are valid)
0	1	0	0	1 Mbyte (00100000 _H byte, ASR A[31:20] bits are valid)
0	1	0	1	2 Mbytes (00200000 _H byte, ASR A[31:21] bits are valid)
0	1	1	0	4 Mbytes (00400000 _H byte, ASR A[31:22] bits are valid)
0	1	1	1	8 Mbytes (00800000 _H byte, ASR A[31:23] bits are valid)
1	0	0	0	16 Mbytes (01000000 _H byte, ASR A[31:24] bits are valid)
1	0	0	1	32 Mbytes (02000000 _H byte, ASR A[31:25] bits are valid)
1	0	1	0	64 Mbytes (04000000 _H byte, ASR A[31:26] bits are valid)
1	0	1	1	128 Mbytes (08000000 _H byte, ASR A[31:27] bits are valid)
1	1	0	0	256 Mbytes (10000000 _H byte, ASR A[31:28] bits are valid)
1	1	0	1	512 Mbytes (20000000 _H byte, ASR A[31:29] bits are valid)
1	1	1	0	1024 Mbytes (40000000 _H byte, ASR A[31:30] bits are valid)
1	1	1	1	2048 Mbytes (80000000 _H byte, ASR A[31] bit is valid)

CHAPTER 4 EXTERNAL BUS INTERFACE

ASZ3 to ASZ0 are used to set the size of each area by modifying the number of comparison of bits to ASR. Thus, an ASR contains bits that are not compared. Bits ASZ3 to ASZ0 of ACR0 are initialized to 1111_B ($0F_H$) by RST. Despite this setting, however, the CS0 area just after RST is executed is specially set from 00000000_H to $FFFFFFFFFF_H$ (setting of entire area). The entire-area setting is reset after the first write to ACR0 and an appropriate size is set as indicated in Table 4.2-1.

If SDRAM/FCRAM is connected to an area set by ASR6 and ASR7, set it 128M bytes (1011_H) or less.

[bit11, bit10] DBW1, DBW0 (Data Bus Width 1, 0)

These bits set the data bus width of each chip select area as indicated in Table 4.2-2:

Table 4.2-2 Setting of the Data Bus Width of Each Chip Select Area

DBW1	DBW0	Data bus width
0	0	8 bits (byte access)
0	1	16 bits (halfword access)
1	0	32 bits (word access)
1	1	Reserved Setting disabled

The same values as those of the WTH bits of the mode vector are written automatically to bits DBW1, DBW0 of ACR0 during the reset sequence.

Use the same data bus width setting for all the SDRAM/FCRAM-connected areas, using these bits.

[bit9, bit8] BST1, BST0 (Burst Size 1, 0)

These bits set the maximum burst length of each chip select area as indicated in Table 4.2-3.

Table 4.2-3 Setting of the Maximum Burst Length of Each Chip Select

BST1	BST0	Maximum burst length
0	0	1 (single access)
0	1	2 bursts (address boundary: 1 bit)
1	0	4 bursts (address boundary: 2 bits)
1	1	8 bursts (address boundary: 3 bits)

In areas for which a burst length other than the single access is set, continuous burst access is performed within the address boundary determined by the burst length only when prefetch access is performed or data having a size exceeding the bus width is read.

The maximum burst length for the area with a bus length of 32 bits is 4 bursts. It is recommended to select 2 bursts or shorter.

Setting of 2 bursts or less as the maximum burst length in the bus width 16-bit area is recommended.

RDY input is ignored in areas for which any burst length other than the single access is set.

Use the same burst length for all the SDRAM/FCRAM-connected areas, using these bits.

[bit7] SREN (ShaRed Enable)

This bit sets enabling or disabling of sharing of each chip select area by BRQ/BGRNT as indicated in the following table.

Table 4.2-4 Enabling or disabling of sharing of each chip select area by BRQ/BGRNT

SREN	Sharing enable/disable
0	Disable sharing by BRQ/BGRNT(CSn cannot be high impedance)
1	Enable sharing by BRQ/BGRNT(CSn can be high impedance)

In areas where sharing is enabled, chip select output (CSn) is set to high impedance while the bus is open (during BGRNT= "L" output). In areas where sharing is disabled, chip select output (CSn) is not set to high impedance even though the bus is open (during BGRNT= "L" output).

Access strobe output (AS, BAA, RD, WR0, WR1, WR2, WR3, WR, MCLK, MCLKE) is set to high impedance only if sharing of all areas enabled by CSER is enabled.

[bit6] PFEN (PreFetch Enable)

This bit sets enabling and disabling of prefetching of each chip select area as indicated in the following table.

Table 4.2-5 Enabling and disabling of prefetching of each chip select area

PFEN	Prefetch enable/disable
0	Disable prefetch
1	Enable prefetch

When reading from an area for which prefetching is enabled, the subsequent address is read in advance and stored in the built-in prefetch buffer. When the stored address is accessed from the internal bus, the lookahead data in the prefetch buffer is returned without performing external access.

For more information, see Section "4.8 Prefetch Operation".

[bit5] WREN (WRite Enable)

This bit sets enabling and disabling of writing to each chip select area.

Table 4.2-6 Enabling and disabling of writing to each chip select area

WREN	Write enable/disable
0	Disable write
1	Enable write

If an area for which write operations are disabled is accessed for a write operation from the internal bus, the access is ignored and no external access at all is performed. Set the WREN bit of areas for which write operations are required, such as data areas, to "1".

[bit4] LEND (Little ENDian select)

This bit sets the order of bytes of each chip select area as indicated in the following table.

Table 4.2-7 Setting of byte ordering to each chip select area

LEND	Order of bytes
0	Big endian
1	Little endian

Be sure to set the LEND bit of ACR0 to "0". CS0 supports only the big endian method.

[bit3 to bit0] TYP3 to TYP0 (TYPe select)

These bits set the access type of each chip select area as indicated in Table 4.2-8.

Table 4.2-8 Access Type Settings for Each Chip Select Area

TYP3	TYP2	TYP1	TYP0	Access type
0	0	x	x	Normal access (asynchronous SRAM, I/O, and single/page/burst-ROM/FLASH)
0	1	x	x	Address data multiplex access (8/16-bit bus width only)
0	x	x	0	Disable WAIT insertion by the RDY pin.
0	x	x	1	Enable WAIT insertion by the RDY pin (disabled during bursts).
0	x	0	x	Use the $\overline{WR0}$ to $\overline{WR3}$ pins as write strobes (\overline{WRn} is always H).
0	x	1	x	Use the \overline{WRn} pin as the write strobe. *1
1	0	0	0	Memory type A: SDRAM/FCRAM *2 (The auto precharge is not used)
1	0	0	1	Memory type B: FCRAM *2 (The auto precharge is used)
1	0	1	0	Setting disabled
1	0	1	1	Setting disabled
1	1	0	0	Setting disabled
1	1	0	1	Setting disabled
1	1	1	0	Setting disabled
1	1	1	1	Mask area setting (The access type is the same as that of the overlapping area) *3

*1: If this setting is made, $\overline{WR0}$ to $\overline{WR3}$ can be used as the enable of each bit.

*2: Only the ACR6 and ACR7 registers are valid. The ACR0, ACR1, ACR2, ACR3, ACR4, and ACR5 registers are disabled.

*3: See the "CS area mask setting function" (next bullet).

Set the access type as the combination of all bits.

For details of the operations of each access type, see the explanation of operation in the section on and after "4.5 Operation of the Ordinary Bus Interface".

○ CS area mask setting function

If you want to set an area some of whose operation settings are changed for a certain CS area (referred to as the base setting area), you can set TYPE3 to TYPE0 of ARC in another CS area to "1111_B" so that the area can function as a mask setting area.

If you do not use the mask setting function, disable any overlapping area settings for multiple CS areas.

Access operations to the mask setting area are as follows:

- $\overline{\text{CSn}}$ corresponding to a mask setting area is not asserted.
- $\overline{\text{CSn}}$ corresponding to a base setting area is asserted.
- For the following ACR settings, the settings on the mask setting area side are valid:
 - bit11, bit10 (DBW1, DBW0): Bus width setting
 - bit9, bit8 (BST1, BST0): Burst length setting
 - bit7 (SREN): Sharing-enable setting
 - bit6 (PFEN): Prefetch-enable setting
 - bit5 (WREN): Write-enable setting (For this setting only, only a setting that is the same as that of the base setting area is allowed)
 - bit4 (LEND): Little endian setting
- For the following ACR setting, the setting on the base setting area side is valid:
 - bit3 to bit0 (TYPE3 to TYPE0): Access type setting
- For the AWR settings, the settings on the mask setting area side are valid.
- For the CHER settings, the settings on the mask setting area side are valid.

A mask setting area can be set for only part of another CS area (base setting area). You cannot set a mask setting area for an area without a base setting area. In addition, the mask setting area must not be duplicated. Use care when setting ASR and bits ASZ3 to ASZ0 of ACR.

The following restrictions apply when using these bits:

- A write-enable setting cannot be implemented by a mask.
- Write-enable settings in the base CS area and the mask setting area must be identical.
- If write operations to a mask setting area are disabled, the area is not masked and operates as a base CS area.
- If write operations to the base CS area are disabled but are enabled to the mask setting area, the area has no base, resulting in malfunctions.

4.2.3 Area Wait Register (AWR0 to AWR7)

This section explains the configuration and functions of the area wait registers (AWR0 to AWR7).

■ Configuration of the Area Wait Registers (AWR0 to AWR7)

The area wait registers (AWR0 to AWR7: Area Wait Register 0 to 7) specify various kinds of waits for each chip select area.

Figure 4.2-3 shows the configuration of the area wait registers (AWR0 to AWR7).

Figure 4.2-3 Configuration of the Area Wait Registers (AWR0 to AWR7)

										Initial value		
	bit	31	30	29	28	27	26	25	24	INIT	RST	Access
AWR0H	bit	31	30	29	28	27	26	25	24	01111111 _B	01111111 _B	R/W
Address	000660 _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR0L	bit	23	22	21	20	19	18	17	16	11111111 _B	11111011 _B	R/W
Address	000661 _H	W07	W06	W05	W04	W03	W02	W01	W00			
AWR1H	bit	15	14	13	12	11	10	9	8	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000662 _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR1L	bit	7	6	5	4	3	2	1	0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000663 _H	W07	W06	W05	W04	W03	W02	W01	W00			
AWR2H	bit	31	30	29	28	27	26	25	24	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000664 _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR2L	bit	23	22	21	20	19	18	17	16	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000665 _H	W07	W06	W05	W04	W03	W02	W01	W00			
AWR3H	bit	15	14	13	12	11	10	9	8	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000666 _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR3L	bit	7	6	5	4	3	2	1	0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000667 _H	W07	W06	W05	W04	W03	W02	W01	W00			
AWR4H	bit	31	30	29	28	27	26	25	24	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000668 _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR4L	bit	23	22	21	20	19	18	17	16	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	000669 _H	W07	W06	W05	W04	W03	W02	W01	W00			
AWR5H	bit	15	14	13	12	11	10	9	8	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00066A _H	W15	W14	W13	W12	W11	W10	W09	W08			
AWR5L	bit	7	6	5	4	3	2	1	0	XXXXXXXXX _B	XXXXXXXXX _B	R/W
Address	00066B _H	W07	W06	W05	W04	W03	W02	W01	W00			

(Continued)

(Continued)

										Initial value			
	bit	31	30	29	28	27	26	25	24	INIT	RST	Access	
AWR6H	bit	31	30	29	28	27	26	25	24	XXXXXXXX _B	XXXXXXXX _B	R/W	
Address 00066C _H		W15	W14	W13	W12	W11	W10	W09	W08				
AWR6L	bit	23	22	21	20	19	18	17	16	XXXXXXXX _B	XXXXXXXX _B	R/W	
Address 00066D _H		W07	W06	W05	W04	W03	W02	W01	W00				
AWR7H	bit	15	14	13	12	11	10	9	8	XXXXXXXX _B	XXXXXXXX _B	R/W	
Address 00066E _H		W15	W14	W13	W12	W11	W10	W09	W08				
AWR7L	bit	7	6	5	4	3	2	1	0	XXXXXXXX _B	XXXXXXXX _B	R/W	
Address 00066F _H		W07	W06	W05	W04	W03	W02	W01	W00				

The function of each bit changes according to the access type (TYP3 to TYP0 bits) setting of the ACR0 to ACR7 registers.

○ Normal access and address/data multiplex access

A chip select area determined by either of the following settings becomes the area for normal access or a address/data multiplex access operation.

Table 4.2-9 Setting of access type (Typ3 to Typ0 bits)

TYP3	TYP2	TYP1	TYP0	Access type
0	0	x	x	Normal access (asynchronous SRAM, I/O, and single/page/burst-ROM/FLASH)
0	1	x	x	Address data multiplex access (8/16-bit bus width only)

The following lists the functions of each AWR0 to AWR7 bit for a normal access or address/data multiplex access area. Since the initial values of registers other than AWR0 are undefined, set them to their initial values before enabling each area with the CSER register.

The following explains the functions of the bits in the area wait registers (AWR0 to AWR7).

[bit15 to bit12] W15 to W12 (First Access Wait Cycle)

These bits set the number of auto-wait cycles to be inserted into the first access cycle of each cycle. Except for the burst access cycles, only this wait setting is used.

The initial value of the CS0 area is set to 7 waits. The initial value of any other area is undefined.

Table 4.2-10 lists the settings for the number of auto-wait cycles during first access.

Table 4.2-10 Settings for the Number of Auto-Wait Cycles (During First Access)

W15	W14	W13	W12	First access wait cycle
0	0	0	0	Auto-wait cycle 0
0	0	0	1	Auto-wait cycle 1
...				...
1	1	1	1	Auto-wait cycle 15

[bit11 to bit8] W11 to W08 (Inpage Access Wait Cycle)

These bits set the number of auto-wait cycles to be inserted into the inpage access cycle during burst access. They are valid only for burst cycles.

Table 4.2-11 lists the settings for the number of auto-wait cycles during burst access.

Table 4.2-11 Settings for the Number of Auto-Wait Cycles (During Burst Access)

W11	W10	W09	W08	Inpage access wait cycle
0	0	0	0	Auto-wait cycle 0
0	0	0	1	Auto-wait cycle 1
...				...
1	1	1	1	Auto-wait cycle 15

If the same value is set for the first access wait cycle and inpage access wait cycle, the access time for the address in each access cycle is not the same. This is because the inpage access cycle contains an address output delay.

[bit7, bit6] W07, W06 (Read -> Write Idle Cycle)

The read -> write idle cycle is set to prevent collision of read data and write data on the data bus when a write cycle follows a read cycle. During an idle cycle, all chip select signals are negated and the data terminals maintain the high impedance state. If a write cycle follows a read cycle or an access operation to another chip select area occurs after a read cycle, the specified idle cycle is inserted. Table 4.2-12 lists the settings for idle cycles.

Table 4.2-12 Settings of the Idle Cycle

W07	W06	Read -> write idle cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

[bit5, bit4] W05, W04 (Write Recovery Cycle)

The write recovery cycle is set if a device that limits the access period after write access is to be controlled. During a write recovery cycle, all chip select signals are negated and the data pins maintain the high impedance state. If the write recovery cycle is set to "1" or more, a write recovery cycle is always inserted after write access.

Table 4.2-13 lists the settings for the number of write recovery cycles.

Table 4.2-13 Settings for the Number of Write Recovery Cycles

W05	W04	Write recovery cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

[bit3] W03 (WR0 to WR3, WRn Output Timing Selection)

The $\overline{WR0}$ to $\overline{WR3}$, \overline{WRn} output timing setting selects whether to use write strobe output as an asynchronous strobe or synchronous write enable. The asynchronous strobe setting corresponds to normal memory/ I/O. The synchronous enable setting corresponds to clock-synchronized memory/ I/O (such as the memory in an ASIC).

Table 4.2-14 Setting of $\overline{WR0}$ to $\overline{WR3}$, \overline{WRn} output timing

W03	$\overline{WR0}$ to $\overline{WR3}$, \overline{WRn} output timing selection
0	MCLK synchronous write enable output (valid from $\overline{AS} = "L"$)
1	Asynchronous write strobe output (normal operation)

If synchronous write enable (W03 bit of AWR is "0") is used, operations are as follows:

- The timing of synchronous write enable output assumes that the output is captured by the rising edge of MCLK output of an external memory access clock. This timing is different from the asynchronous strobe output timing.
- The $\overline{WR0}$ to $\overline{WR3}$ and \overline{WRn} terminal output asserts synchronous write enable output at the timing at which \overline{AS} pin output is asserted. For a write to an external bus, the synchronous write enable output is "L". For a read from an external bus, the synchronous write enable output is "H".
- Write data is output from the external data output pin in the clock cycle following the cycle in which synchronous write enable output is asserted.
- Read strobe output (\overline{RD}) functions as an asynchronous read strobe regardless of the setting of the $\overline{WR0}$ to $\overline{WR3}$ and \overline{WRn} output timing. Use it as is for controlling the data I/O direction.

If synchronous write enable output is used, the following restrictions apply:

- Do not make the following additional wait settings:
 - $\overline{CSn} \rightarrow \overline{RD}/\overline{WRn}$ setup (Always set "0" for the W01 bit of AWR)
 - First wait cycle setting (Always set "0000_B" for the W15 to W12 bits of AWR)
- Do not make the following access type settings (TYPE3 to TYPE0 bits in the ACR register (bit3 to bit0))
 - Address/data multiplex bus setting (Always set "0" for the TYPE2 bit of ACR)
 - Setting to use $\overline{WR0}$ to $\overline{WR3}$ as a strobe (Always set "0" for the TYPE1 bit of ACR)
 - RDY input enable setting (Always set "0" for the TYPE0 bit of ACR)
- For synchronous write enable output, always set 1 ("00_B" for bits BST1, BST0 of ACR) as the burst length.

[bit2] W02 (Address -> CSn Delay)

The address -> CSn delay setting is made when a certain type of setup is required for the address when CSn falls or CSn edges are needed for successive accesses to the same chip select area.

Set the address and set the delay from AS output to CS0 to CS7 output.

Table 4.2-15 Setting of address -> CSn delay

W02	Address -> CSn delay
0	Delay
1	No delay

If no delay is selected by setting "0", assertion of CS0 to CS7 starts at the same timing that AS is asserted. If, at this point, successive accesses are made to the same chip select area, assertion of CS0 to CS7 without change between two access operations may continue.

If delay is specified by selecting "1", assertion of CS0 to CS7 starts when the external clock memory MCLK output rises. If, at this point, successive accesses are made to the same chip select area, CS0 to CS7 are negated at a timing between two access operations. If CSn delay is selected, one setup cycle is inserted before asserting the read/write strobe after assertion of the delayed CSn (operation is the same as the CSn -> RD/WRn setup setting of W01).

The address -> CSn delay setting works for DACKX signal (basic mode) output to the same area in the same way. DACKX output in basic mode has the same waveforms as those of CSn output to the same area.

[bit1] W01 (CSn -> RD/WRn Setup Extension Cycle)

The CSn -> RD/WRn setup extension cycle is set to extend the period before the read/write strobe is asserted after CSn is asserted. At least one setup extension cycle is inserted before the read/write strobe is asserted after CSn is asserted.

Table 4.2-16 Function of CSn -> RD/WRn setup delay cycle

W01	CSn -> RD/WRn setup delay cycle
0	0 cycle
1	1 cycle

If 0 cycle is selected by setting "0", RD/WR0 to WR3/WR are output at the earliest when external clock MCLK output rises just after CSn is asserted. WR0 to WR3/WR may be delayed one cycle or more depending on the internal bus state.

If 1 cycle is selected by setting "1", RD/WR0 to WR3/WR are always output 1 cycle or more later.

When successive accesses are made within the same chip select area without negating CSn, a setup extension cycle is not inserted. If a setup extension cycle for determining the address is required, set the W02 bit and insert the address -> CSn delay. Since CSn is negated for each access operation, the setup extension cycle is enabled.

If the CSn delay set by W02 is inserted, this setup cycle is always enabled regardless of the setting of the W01 bit.

[bit0] W00 (RD/WRn -> CSn Hold Extension Cycle)

The $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold extension cycle is set to extend the period before negating $\overline{\text{CSn}}$ after the read/write strobe is negated. One hold extension cycle is inserted before $\overline{\text{CSn}}$ is negated after the read/write strobe is negated.

Table 4.2-17 Function of $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold extension cycle

W00	$\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold extension cycle
0	0 cycle
1	1 cycle

If 0 cycle is selected by setting "0", $\overline{\text{CS}0}$ to $\overline{\text{CS}7}$ are negated after the hold delay after it starts on the rising edge of external memory clock MCLK output after $\overline{\text{RD}}/\overline{\text{WR}0}$ to $\overline{\text{WR}3}/\overline{\text{WR}}$ are negated.

If 1 cycle is selected by setting "1", $\overline{\text{CS}0}$ to $\overline{\text{CS}7}$ are negated one cycle later.

When making successive accesses within the same chip select area without negating $\overline{\text{CSn}}$, the hold extension cycle is not inserted. If a hold extension cycle for determining the address is required, set the W02 bit and insert the address -> $\overline{\text{CSn}}$ delay. Since $\overline{\text{CSn}}$ is negated for each access operation, this hold extension cycle is enabled.

○ Memory type A(SDRAM/FCRAM) and Memory type B(FCRAM)

The chip select areas for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 4.2-18 serve for SDRAM/FCRAM access.

Table 4.2-18 lists the access type settings (TYP3 to TYP0 bits).

Table 4.2-18 Access Type Settings (TYP3 to TYP0 Bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	0	Memory type A: SDRAM/FCRAM (Auto-precharge is not used.)

The following explains those functions of individual bits in AWR6 and AWR7 which apply to SDRAM access areas. As the initial value is undefined, set the access type before each area is enabled by the chip select area enable register (CSER).

For all the areas connected to SDRAM/FCRAM, use the same settings for this type of registers.

The following summarizes the functions of individual bits in the area wait registers (AWR6 and AWR7).

[bit15] W15: Reserved bit

Be sure to set this bit to "0".

[bit14 to bit12] W14 to W12 (RAS-CAS delay Cycle): RAS-CAS delay cycles

Set these bits to the number of cycles from RAS output to CAS output.

Table 4.2-19 lists the settings for the number of cycles from RAS output to CAS output.

Table 4.2-19 Setting the Number of Cycles from RAS Output to CAS Output

W14	W13	W12	RAS-CAS delay cycle
0	0	0	1 cycle
0	0	0	2 cycles
...			...
1	1	1	8 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS-CAS delay cycle.

[bit11] W11: Reserved bit

Be sure to set this bit to "0".

[bit10 to bit8] W10 to W08 (CAS latency Cycle): CAS latency

Set these bits to the CAS latency.

Table 4.2-20 lists the settings for the CAS latency.

Table 4.2-20 CAS Latency Setting

W10	W09	W08	CAS latency
0	0	0	1 cycle
0	0	0	2 cycles
...			...
1	1	1	8 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same CAS latency.

[bit7, bit6] W07, W06 (Read ->Write Cycle): Read-to-write cycle

Set these bits to the minimum number of cycles from the last read data input cycle to the write command issuance. Set the minimum number of cycles taken until issuance.

Table 4.2-21 lists the settings for the read-to-write cycle.

Table 4.2-21 Read-to-write Cycle

W07	W06	Read-to-write cycle
0	0	1 cycle
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same read-to-write cycle.

The number of read-to-write idle cycles is one smaller than the number of cycles set by this bit.

[bit5, bit4] W05, W04 (Write Recovery Cycle): Write recovery cycle

Set these bits to the minimum number of cycles from the last write data output to the next read command issuance.

Table 4.2-22 lists the settings for the write recovery cycle.

Table 4.2-22 Write Recovery Cycle

W05	W04	Write recovery cycle
0	0	Prohibited
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same write recovery cycle.

CHAPTER 4 EXTERNAL BUS INTERFACE

[bit3, bit2] W03, W02 (RAS Active time): RAS active time

Set these bits to the minimum number of cycles for RAS active time.

Table 4.2-23 lists the settings for RAS active time.

Table 4.2-23 RAS Active Time

W03	W02	RAS active time
0	0	1 cycle
0	1	2 cycles
1	0	5 cycles
1	1	6 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS active time.

[bit1, bit0] W01, W00 (RAS precharge cycle): RAS precharge cycles

Set these bits to the number of RAS precharge cycles.

Table 4.2-24 lists the settings for the RAS precharge cycle.

Table 4.2-24 RAS Precharge Cycle

W03	W02	RAS precharge cycle
0	0	1 cycle
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS precharge cycle.

4.2.4 Memory Setting Register (MCRA for SDRAM/FCRAM auto-precharge OFF mode)

This section describes the configuration and the function of memory setting register (MCRA for SDRAM/FCRAM auto-precharge OFF mode).

■ Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)

Memory setting register (MCRA for SDRAM/FCRAM auto-precharge OFF mode)

The memory setting register (MCRA: Memory Setting Register for extend type-A for SDRAM/FCRAM auto-precharge OFF mode) is used to make various settings for SDRAM/FCRAM connected to the chip select area.

Figure 4.2-4 shows the bit configuration of the memory setting register (MCRA for SDRAM/FCRAM auto-precharge OFF mode).

Figure 4.2-4 Configuration of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)

Address 000670H	bit	31	30	29	28	27	26	25	24	Initial value
	Reserved	PSZ2	PSZ1	PSZ0	WBST	BANK	ABS1	ABS0		XXXXXXXX _B (INIT) XXXXXXXX _B (RST)
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The register serves as the area for making various settings for SDRAM/FCRAM connected to the chip select area for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 4.2-25.

Table 4.2-25 lists the access type settings (TYP3 to TYP0 bits).

Table 4.2-25 Access Type Settings (TYP3 to TYP0 Bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	0	Memory type A:SDRAM/FCRAM (not used auto precharge)

MCRB shares register hardware with MCRA. Updating the MCRA therefore updates the MCRB accordingly.

The following summarizes the functions of individual bits in the memory setting register (MCRA for SDRAM/FCRAM auto-precharge OFF mode).

[bit31] (Reserved)

Be sure to set this bit to "0".

[bit30 to bit28] PSZ2 to PSZ0 (Page SiZe): Page size

Set these bits to the page size of SDRAM to be connected.

Table 4.2-26 lists the settings for the page size of SDRAM connected.

Table 4.2-26 Settings for the Page Size of SDRAM

PSZ2	PSZ1	PSZ0	Page size of SDRAM
0	0	0	8-bit column address:A0 to A7(256 memory words)
0	0	1	9-bit column address:A0 to A8(512 memory words)
0	1	0	10-bit column address:A0 to A9(1024 memory words)
0	1	1	11-bit column address:A0 to A9, A11(2048 memory words)
1	X	X	Prohibited

[bit27] WBST (Write BurST enable): Write burst setting

Set this bit to select whether to burst-write for write access.

Table 4.2-27 lists the settings for burst write.

Table 4.2-27 Settings for Burst Write

WBST	Settings for burst write
0	Single write
1	Burst write

For connecting FCRAM, be sure to set the bit to "1".

FCRAM supports neither burst read nor single write mode.

[bit26] BANK (BANK type select): Bank number setting

Set this bit to the number of banks of SDRAM to be connected.

Table 4.2-28 lists the settings for bank number.

Table 4.2-28 Settings for Bank Number

BANK	Settings for bank number
0	2 banks
1	4 banks

[bit25, bit24] ABS1, ABS0 (Active Bank Select): Setting of active bank number

Set these bits to the maximum number of banks to be made active simultaneously.

Table 4.2-29 lists the settings for the number of active banks.

Table 4.2-29 Settings for the Number of Active Banks

ABS1	ABS0	Number of active banks
0	0	1 bank
0	1	2 banks
1	0	3 banks
1	1	4 banks

4.2.5 Memory Setting Register (MCRB for FCRAM auto-precharge ON mode)

This section describes the memory setting register (MCRB for FCRAM auto-precharge ON mode).

■ Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode)

Settings for Memory configuration register (MCRB: Memory Configuration Register for extend type-B for FCRAM auto-precharge ON mode) is used to make various settings for FCRAM connected to the chip select area.

Figure 4.2-5 shows the bit configuration of the memory setting register (MCRB for FCRAM auto-precharge ON mode).

Figure 4.2-5 Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode)

Address 000671 _H	bit 23	22	21	20	19	18	17	16	Initial value	Access
	Reserved	PSZ2	PSZ1	PSZ0	WBST	BANK	ABS1	ABS0	XXXXXXXX _B (INIT) XXXXXXXX _B (RST)	R/W

The register serves as the area for making various settings for FCRAM connected to the chip select area for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 4.2-30.

Table 4.2-30 lists the access type settings (TYP3 to TYP0 bits).

Table 4.2-30 Access Type Settings (TYP3 to TYP0 Bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	1	Memory type B: FCRAM (used auto precharge)

MCRB shares register hardware with MCRA. Updating the MCRB therefore updates the MCRA accordingly.

The functions are the same as MCRA. Note, however, that the function of the WBST bit is not available to this TYP setting.

(FCRAM supports neither burst read nor single write mode.)

4.2.6 I/O Wait Registers for DMAC (IOWR0, IOWR1)

This section explains the configuration and functions of the I/O wait registers for DMAC (IOWR0, IOWR1).

■ Configuration of the I/O Wait Registers for DMAC (IOWR0, IOWR1)

The I/O wait registers for DMAC (IOWR0, IOWR1: I/O Wait Register for DMAC0, DMAC1) set various kinds of waits during DMA fly-by access.

Figure 4.2-6 shows the configuration of the I/O wait registers for DMAC (IOWR0, IOWR1).

Figure 4.2-6 Configuration of the I/O Wait Registers for DMAC (IOWR0, IOWR1)

IOWR0	bit	31	30	29	28	27	26	25	24	Initial value	Access
	Address	000678H	RYE0	HLD0	WR01	WR00	IW03	IW02	IW01	IW00	XXXXXXXXB (INIT) XXXXXXXXB (RST)
IOWR1	bit	23	22	21	20	19	18	17	16		
	Address	000679H	RYE1	HLD1	WR11	WR10	IW13	IW12	IW11	IW10	XXXXXXXXB (INIT) XXXXXXXXB (RST)

■ Functions of Bits in the I/O Wait Registers for DMAC (IOWR0, IOWR1)

The following explains the functions of the bits in the I/O wait registers for DMAC.

[bit31, bit23] RYE0, RYE1 (RDY enable 0, RDY enable 1)

These bits set the wait control, using RDY, of channels 0 and 1 during DMAC fly-by access.

Table 4.2-31 RDY function setting

RYEn	RDY function setting
0	Disable RDY input for I/O access.
1	Enable RDY input for I/O access.

When "1" is set, wait insertion by the RDY pin can be performed during fly-by transfer on the relevant channel. IOWR and IORD are extended until the RDY pin is enabled. Also, RD/WR0 to WR3/WR on the memory side are extended synchronously. If the chip select area of the fly-by transfer destination is set to RDY-enabled in the ACR register, wait insertion by the RDY pin can be performed regardless of the RYEn bit of IOWR. When the chip select area of the fly-by transfer destination is set to RDY-disabled in the ACR register, wait insertion by the RDY pin can only be performed during fly-by access if the area is set to RDY-enabled by the RYEn bit on the IOWR side.

[bit30, bit22] HLD0, HLD1 (Hold Wait Control)

These bits control the hold cycle of the read strobe signal on the transfer source access side during DMA fly-by access.

Table 4.2-32 Hold wait setting

HLDn	Hold wait setting
0	Do not insert a hold extension cycle.
1	Insert a hold extension cycle to extend the read cycle by one cycle.

If "0" is set, the read strobe signal (\overline{RD} for memory \rightarrow I/O and \overline{IORD} for I/O \rightarrow memory) and the write strobe signal (\overline{IOWR} for memory \rightarrow I/O and $\overline{WR0}$ to $\overline{WR3}$ and \overline{WR} for I/O \rightarrow memory) on the transfer source access side are output at the same timing.

If "1" is set, the read strobe signal is output one cycle longer than the write strobe signal to secure a hold time for data at the transfer source access side when sending it to the transfer destination.

[bit29, bit28, bit21, bit20] WR01/WR00, WR11/WR10 (I/O Idle Wait) :

Setting the number of I/O Idle Cycles

These bits set the number of idle cycles for continuous access during DMA fly-by access. Table 4.2-33 lists the settings for the number of I/O idle cycles.

Table 4.2-33 Settings for the Number of I/O Idle Cycles

WRn1	WRn0	Setting of the number of I/O idle cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

If one or more cycles is set as the number of idle cycles, cycles equal to the number specified are inserted after I/O access during DMA fly-by access. During the idle cycles, all \overline{CSn} and strobe output is negated and the data pin is set to the high impedance state.

[bit27 to bit24, bit19 to bit16, bit11 to bit8] IW03 to IW00,IW13 to IW10 (I/O Access Wait) :

Setting the number of I/O Access Wait Cycles

These bits set the number of auto-wait cycles for I/O access during DMA fly-by access.

Table 4.2-34 lists the settings for the number of I/O wait cycles.

Table 4.2-34 Settings for the Number of I/O Wait Cycles

IWn3	IWn2	IWn1	IWn0	Number of I/O wait cycles
0	0	0	0	0 cycle
0	0	0	1	1 cycle
...				...
1	1	1	1	15 cycles

Because data is synchronized between the transfer source and transfer destination, the I/O side setting of the IWnn bits and the wait setting for the fly-by transfer destination (such as memory), whichever is larger, is used as the number of wait cycles to be inserted. Consequently, more wait cycles than specified by the IWnn bits may be inserted.

4.2.7 Chip Select Enable Register (CSER)

The chip select enable register (CSER) set up the access permit of each chip select area.

■ Configuration of the Chip Select Enable Register (CSER)

The chip select enable register (CSER: Chip Select Enable register) enables and disables each chip select area.

Figure 4.2-7 shows the configuration of the chip select enable register (CSER).

Figure 4.2-7 Configuration of the Chip Select Enable Register (CSER)

Address	bit 31	30	29	28	27	26	25	24	Initial value	Access
000680 _H	CSE7	CSE6	CSE5	CSE4	CSE3	CSE2	CSE1	CSE0	11111111 _B (INIT)	R/W
000681 _H									11111111 _B (RST)	

■ Functions of Bits in the Chip Select Enable Register (CSER)

The following explains the functions of the bits in the chip select enable register (CSER).

[bit31 to bit24] CSE7 to CSE0 (Chip Select Enable 0 to Chip Select Enable 7)

These bits are the chip select enable bits for $\overline{CS0}$ to $\overline{CS7}$.

The initial value is 00000001_B, which enables only the CS0 area.

When "1" is written, a chip select area operates according to the settings of ASR0 to ASR7, ACR0 to ACR7, and AWR0 to AWR7.

Before setting this register, be sure to make all settings required for the corresponding chip select areas. However, make sure to execute the power-on sequence by the PON bit in the refresh control register (RCR) after enabling the chip select areas using the chip select area enable register (CSER).

The power-on sequence is invalid to SDRAM/FCRAM connected to the area which has not been enabled by the chip select area enable register (CSER).

Table 4.2-35 shows the function of chip select enable 0 to 7.

Table 4.2-35 Function of chip select enable 0 to 7

CSE7 to CSE0	Area control
0	Disable
1	Enable

Table 4.2-36 lists the corresponding $\overline{CS_n}$ for the chip select enable bits.

Table 4.2-36 $\overline{CS_n}$ Corresponding to the Chip Select Enable Bits

CSE bit	Corresponding $\overline{CS_n}$
bit24: CSE0	$\overline{CS0}$
bit25: CSE1	$\overline{CS1}$
bit26: CSE2	$\overline{CS2}$
bit27: CSE3	$\overline{CS3}$
bit28: CSE4	$\overline{CS4}$
bit29: CSE5	$\overline{CS5}$
bit30: CSE6	$\overline{CS6}$
bit31: CSE7	$\overline{CS7}$

4.2.8 Cache Enable Register (CHER)

This section explains the configuration and functions of the cache enable register (CHER).

■ Configuration of the Cache Enable Register (CHER)

The cache enable register (CHER: CacHe Enable Register) controls the transfer of data read from each chip select area.

Figure 4.2-8 shows the configuration of the cache enable register (CHER).

Figure 4.2-8 Configuration of the Cache Enable Register (CHER)

Address	bit 23	22	21	20	19	18	17	16	Initial value	Access
000681 _H	CHE7	CHE6	CHE5	CHE4	CHE3	CHE2	CHE1	CHE0	11111111 _B (INIT) 11111111 _B (RST)	R/W

■ Functions of Bits in the Cache Enable Register (CHER)

The following explains the functions of the bits in the cache enable register (CHER).

[bit23 to bit17] CHE7 to CHE0 (Cache Enable 7 to Cache Enable 0)

These bits enable and disable each chip select area for transfers to the built-in cache.

Table 4.2-37 Cache area setting

CHEn	Cache area setting
0	Not a cache area (data read from the applicable area is not saved in the cache)
1	Cache area (data read from the applicable area is saved in the cache)

4.2.9 Pin/Timing Control Register (TCR)

This section explains the configuration and functions of the pin/timing control register and its function.

■ Configuration of the Pin/Timing Control Register (TCR)

The pin/timing control register (TCR: Terminal and Timing Control Register) controls the functions related to the general external bus interface controller, such as the setting of common pin functions and timing control.

Figure 4.2-9 shows the configuration of the pin/timing control register (TCR).

Figure 4.2-9 Configuration of the Pin/Timing Control Register (TCR)

Address	bit	7	6	5	4	3	2	1	0	Initial value	Access
000683H		BREN	PSUS	PCLR	Reserved	OHT1	OHT0	RDW1	RDW0	00000000B (INIT) 0000XXXXB (RST)	R/W

■ Functions of Bits in the Pin/Timing Control Register (TCR)

The following explains the functions of the bits in the pin/timing control register (TCR).

[bit7] BREN (BRQ Enable)

This bit enables BRQ pin input and external bus sharing.

Table 4.2-38 BRQ input enable setting

BREN	BRQ input enable setting
0	No bus sharing by BRQ/BGRNT. BRQ input is disabled.
1	Bus sharing by BRQ/BGRNT. BRQ input is enabled.

In the initial state ("0"), BRQ input is ignored. When "1" is set, the bus is made open (control with high impedance) and BGRNT is activated ("L" level is output) when the bus is ready to be made open after the BRQ input becomes "H" level.

[bit6] PSUS (Prefetch suspend)

This bit controls temporary stopping of prefetch to all areas.

Table 4.2-39 Function of prefetch control

PSUS	Prefetch control
0	Enable prefetch
1	Suspend prefetch

If "1" is set, no new prefetch operation is performed before "0" is written. Since during this time the contents of the prefetch buffer are not deleted unless a prefetch buffer occurs, clear the prefetch buffer using the PCLR bit function (bit5) before restarting prefetch.

[bit5] PCLR (Prefetch buffer clear)

This bit completely clears the prefetch buffer.

Table 4.2-40 Function of prefetch buffer control

PCLR	Prefetch buffer control
0	Normal state
1	Clear the prefetch buffer.

If "1" is written, the prefetch buffer is cleared completely. When clearing is completed, the bit value automatically returns to "0". Interrupt (set to "1") the prefetch by the PSUS bit and then clear the buffer (It is also possible to write 11_B to both the PSUS and PCLR bits).

[bit4 to bit2] (Reserved)

These bits are reserved. Be sure to set it to "0".

[bit1, bit0] RDW1, RDW0 (Reduce Wait cycle)

These bits instruct all chip select areas and fly-by I/O channels to reduce only the number of auto-wait cycles in the auto-access cycle wait settings uniformly while the AWR register settings are retained unchanged. The settings for idle cycles, recovery cycles, setup, and hold cycles are not affected. They do not function in the SDRAM control areas either. Table 4.2-41 lists the settings for the wait cycle reduction for combinations of these bits.

Table 4.2-41 Settings for Wait Cycle Reduction

RDW1	RDW0	Wait cycle reduction
0	0	Normal wait (AWR0 to AWR7 settings)
0	1	1/2 (1-bit shift to the right) of the AWR0 to AWR7 settings
1	0	1/4 (2-bit shift to the right) of the AWR0 to AWR7 settings
1	1	1/8 (3-bit shift to the right) of the AWR0 to AWR7 settings

The purpose of this function is to prevent an excessive access cycle wait during operation on a low-speed clock (for example, when the base clock is switched to low speed or the frequency division ratio setting of the external bus clock is large).

To reset the wait cycle in these cases, each of the AWRs must usually be rewritten one at a time. However, when the RDW1/RDW0 bit function is used, the access cycle wait is reduced for all of the AWRs in a single operation while all of the other high-speed clock settings in each register are retained.

Before returning the clock to high speed, be sure to reset the RDW1/RDW0 bits to " 00_B ".

4.2.10 Refresh Control Register (RCR)

This section describes the bit configuration and functions of the refresh control register (RCR).

■ Structure of the Refresh Control Register (RCR)

The refresh control register (RCR) is used to make various refresh control settings for SDRAM.

The setting of this register is meaningless as long as SDRAM control is not set for any area, in that case the register value must not be updated from the initial state.

When read by a read-modify-write instruction, the SELF, RRLD, and PON bits always return to "0".

Figure 4.2-10 shows the bit configuration of the refresh control register (RCR).

Figure 4.2-10 Structure of the Refresh Control Register (RCR)

RCRH	bit	31	30	29	28	27	26	25	24	Initial value	Access
Address 000684H		SELF	RRLD	RFINT5	RFINT4	RFINT3	RFINT2	RFINT1	RFINT0	00XXXXXX _B (INIT) 00XXXXXX _B (RST)	R/W
RCRL	bit	23	22	21	20	19	18	17	16	XXXX0XXX _B (INIT) XXXX0XXX _B (RST)	R/W
Address 000685H		BRST	RFC2	RFC1	RFC0	PON	TRC2	TRC1	TRC0		

■ Bit Functions of the Refresh Control Register (RCR)

The following summarizes the functions of individual bits in the refresh control register (RCR).

[bit31] SELF (SELF refresh assert): Self-refresh control

This bit is used to control the self-refresh mode for memory that supports the self-refresh mode.

Table 4.2-42 lists the settings for self-refresh control.

Table 4.2-42 Settings for Self-refresh Control

SELF	Self-refresh control
0	Auto-refresh or power-down
1	Transition to self-refresh mode

Setting the bit to "1" performs a self-refresh after issuing the SELF command. Writing "0" terminates the self-refresh mode.

To hold the contents of SDRAM when putting the LSI into stop mode, use this bit to enter the self-refresh mode before entering the stop mode. At this time, centralized refreshing is performed before transition to the self-refresh mode. External access requests generated before it is completed are put on hold. The mode transits to the stop mode.

The device is released from the self-refresh mode either when "0" is written to this bit or access to SDRAM occurs. At this time, centralized refreshing is performed immediately after the release. If external access such as SDRAM access is attempted, therefore, the external access request is kept on hold and the CPU stops operation for a while. An attempt to put the LSI into the stop mode when it cannot enter the self-refresh mode causes it to directly enter the power save mode, resulting in corruption of data in SDRAM.

When read by a Read-modify-Write instruction, the SELF, RRLD, and PON bits always return to "0".

[bit30] RRLD (Refresh counter ReLoaD): Refresh counter start control

This bit is used to start and reload the refresh counter.

Table 4.2-43 shows the function of refresh counter startup control.

Table 4.2-43 Function of Refresh Counter Startup Control

RRLD	Refresh counter startup control
0	Disable (no operation)
1	Execute auto-refreshing once and reload the RFINT value.

The refresh counter is inactive in the initial state.

If this bit is set to "1" in this state, all the SDRAM areas currently enabled in the CSER are auto-refreshed either once in distributed refresh mode or the RFC-specified number of times in centralized refresh mode. After that, the values in the RFINT5 to RFINT0 bits are reloaded.

From then on, the refresh counter starts being decremented. Whenever the counter causes an underflow from "000000_B", repeatedly, the values in the RFINT5 to RFINT0 bits are reloaded while at the same time auto-refreshing is performed once.

The bit returns to "0" upon completion of reloading.

To stop auto-refreshing, write "000000_B" to the RFINT5 to RFINT0 bits.

When read by a Read-modify-Write instruction, the bit always returns "0".

[bit29 to bit24] RFINT5 to RFINT0 (ReFresh INTerval): Auto-refresh interval

Set these bits to the interval for automatic refreshing.

The auto-refresh interval can be obtained for distributed refresh mode $\{(REFINT5 - REFINT0) \times 32 \times (\text{external bus clock cycle})\}$ or for centralized refresh mode $\{(REFINT5 - REFINT0) \times 32 \times (\text{RFC specified number of times}) \times (\text{external bus clock cycle})\}$

Calculate the design value in consideration of the maximum RAS active time.

To stop the auto refresh function, write "000000_B" to the RFINT5 to RFINT0 bits.

The refresh counter keeps on being decremented even while the auto-refresh command is being issued.

[bit23] BRST (BuRST refresh select): Burst refresh control

This bit is used to control the operation mode for auto-refreshing.

Table 4.2-44 shows the function of burst refresh control.

Table 4.2-44 Function of Burst Refresh Control

BRST	Burst refresh control
0	Distributed refresh (Auto-refresh is activated at intervals.)
1	Burst refresh (Auto-refresh is activated repeatedly at one time.)

When distributed refreshing is set, the auto-refresh command is issued once at every refresh interval.

When burst refreshing is set, the auto-refresh command is issued continuously for the number of times set in the refresh counter at every refresh interval.

[bit22 to bit20] RFC2, RFC1, RFC0 (ReFresh Count): Refresh count

Set these bits to the number of times a refresh must be performed to refresh all SDRAM.

Table 4.2-45 shows the number of times to refresh.

Table 4.2-45 Number of Times to Refresh

RFC2	RFC1	RFC0	Number of times to refresh
0	0	0	256
0	0	1	512
0	1	0	1024
0	1	1	2048
1	0	0	4096
1	0	1	8192
1	1	0	Setting prohibited
1	1	1	Refresh prohibited

The number of times to refresh specified here is the number of times centralized refreshing is performed before and after transition to the self-refresh mode. When burst refreshing has been selected with the BRST bit, the number of times to refresh is also the number of times the refresh command is issued at every refresh interval.

CHAPTER 4 EXTERNAL BUS INTERFACE

[bit19] PON (Power ON): Power-on control

This bit is used to control the SDRAM (FCRAM) power-on sequence.

Table 4.2-46 shows the function of power-on control.

Table 4.2-46 Function of Power-on Control

PON	Power-on control
0	Disabled (no-operation)
1	Start power-on sequence

Writing "1" to the PON bit starts the SDRAM power-on sequence.

Before starting the power-on sequence, be sure to set the relevant registers such as AWR, MCRA(B), and CSER.

This bit returns to "0" as soon as the power-on sequence is started.

When enabling the PON bit, set RFINT and enable RRLD to activate the refresh counter.

Refreshing is not performed only with the PON bit.

Do not enable this bit along with the SELF bit.

When read by a Read-modify-Write instruction, the bit always returns "0".

[bit18 to bit16] TRC2, TRC1, TRC0 (Time of Refresh Cycle): Refresh cycle (t_{RC})

These bits set the refresh cycle (t_{RC}).

Table 4.2-47 lists the settings for the refresh cycle (t_{RC}).

Table 4.2-47 Settings for the Refresh Cycle (t_{RC})

TRC2	TRC1	TRC0	Refresh cycle (t_{RC})
0	0	0	4
0	0	1	5
0	1	0	6
0	1	1	7
1	0	0	8
1	0	1	9
1	1	0	10
1	1	1	11

4.3 Setting Example of the Chip Select Area

In the external bus interface, a total of eight chip select areas can be set. This section presents an example of setting the chip select area.

■ Example of Setting the Chip Select Area

The address space of each area can be placed, in units of a minimum of 64K bytes, anywhere in the 4 Gbytes space using ASR0 to ASR7 (Area Select Registers) and ACR0 to ACR7 (Area Configuration Registers). When bus access is made to an area specified by these registers, the corresponding chip select signals ($\overline{CS0}$ to $\overline{CS7}$) are activated ("L" output) during the access cycle.

○ Example of setting ASRs and ASZ3 to ASZ0

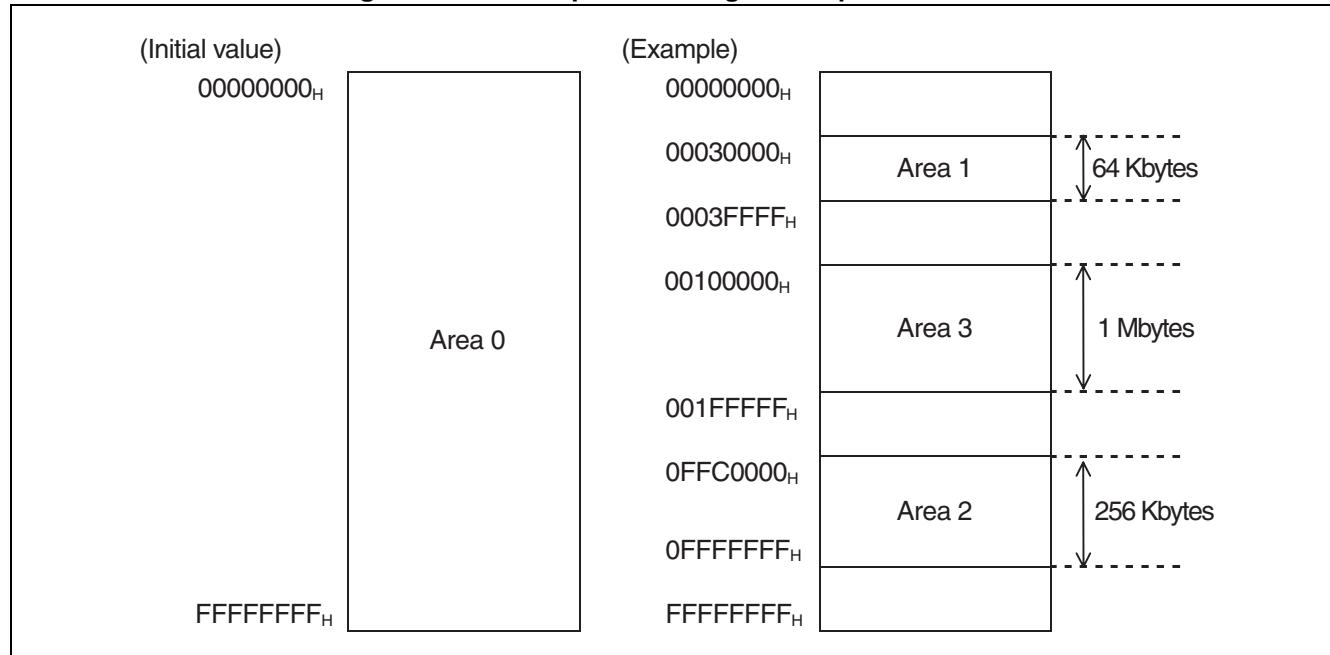
- ASR1=0003_H ACR1 ASZ3 to ASZ0=0000_B: Chip select area 1 is assigned to 00030000_H to 0003FFFF_H.
- ASR2=0FFC_H ACR2 ASZ3 to ASZ0=0010_B: Chip select area 2 is assigned to 0FFC0000_H to 0FFFFFFF_H.
- ASR3=0011_H ACR3 ASZ3 to ASZ0=0100_B: Chip select area 3 is assigned to 00100000_H to 001FFFFF_H.

Since at this point 1M bytes is set for bits ASZ3 to ASZ0 of the ACR, the unit for boundaries 1 Mbyte and bit19 to bit16 of ASR3 are ignored. Before there is any writing to ACR0 after a reset, 00000000_H to FFFFFFFF_H is assigned to chip select area 0.

Set the chip select areas so that there is no overlap.

Figure 4.3-1 shows an example of setting the chip select area.

Figure 4.3-1 Example of Setting the Chip Select Area



4.4 Endian and Bus Access

There is a one-to-one correspondence between the $\overline{WR0}$ to $\overline{WR3}$ control signal and the byte location regardless of the data bus width. The following summarizes the location of bytes on the data bus of the FR family used according to the specified data bus width and the corresponding control signal for each bus mode.

■ Relationship between Data Bus Width and Control Signal

This section summarizes the location of bytes on the data bus used according to the specified data bus width and the corresponding control signal for each bus mode.

○ Ordinary bus interface

Figure 4.4-1 Data Bus Width and Control Signal on the Ordinary Bus Interface

a) 32-bit bus width	b) 16-bit bus width	c) 8-bit bus width			
Databus	Control signal	Databus	Control signal	Databus	Control signal
D31	$\overline{WR0}$ (UUB)		$\overline{WR0}$ (UUB)		$\overline{WR0}$ (UUB)
	$\overline{WR1}$ (ULB)		$\overline{WR1}$ (ULB)	-	-
	$\overline{WR2}$ (LUB)	-	-	-	-
	$\overline{WR3}$ (LLB)	-	-	-	-
D0					
		(D15 to D0 are not used)		(D23 to D0 are not used)	

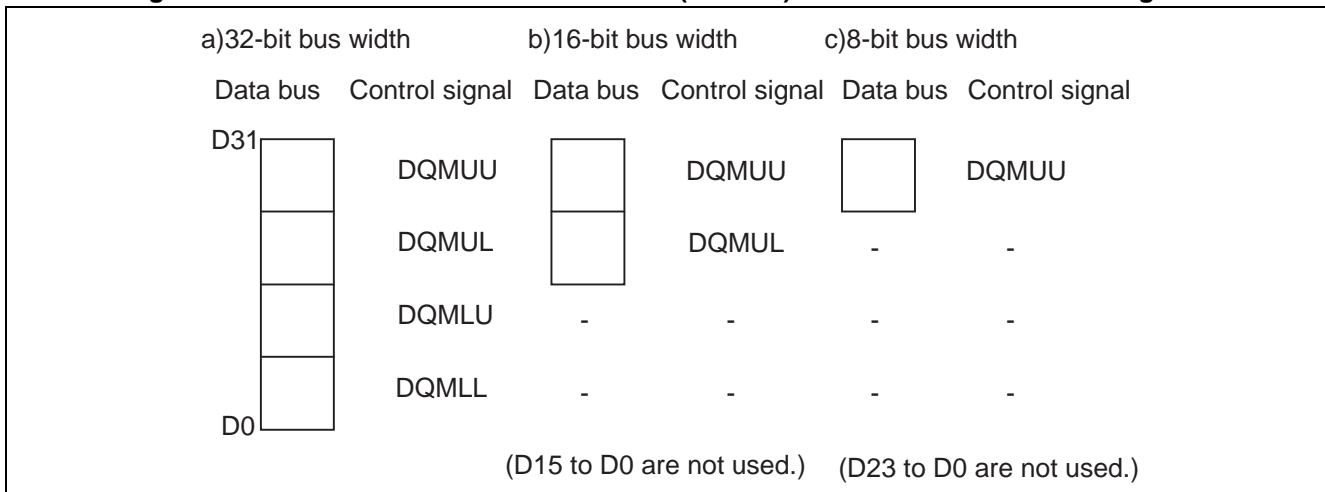
○ Time division I/O interface

Figure 4.4-2 Data Bus Width and Control Signal in the Time Division I/O Interface

a) 16-bit bus width	b) 8-bit bus width				
Databus	Output address	Control signal	Databus	Output address	Control signal
D31	A15 to A8	$\overline{WR0}$		A7 to A0	$\overline{WR0}$
D16	A7 to A0	$\overline{WR1}$	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
(D15 to D0 are not used)			(D23 to D0 are not used)		

- SDRAM (FCRAM) Interface

Figure 4.4-3 Data Bus Width of the SDRAM (FCRAM) Interface and Its Control Signals



4.4.1 Big Endian Bus Access

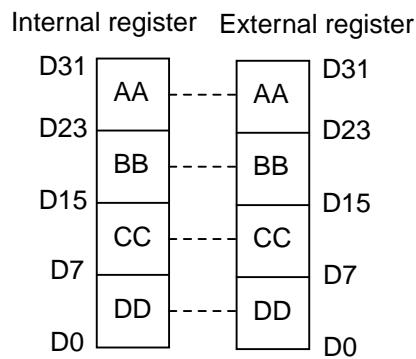
With the exception of the CS0 area of the FR family, either the big endian method or the little endian method can be selected for each chip select area. If "0" is set for the LEND bit of the ACR register, the area is treated as big endian. The FR family is normally big endian and performs external bus access.

■ Data Format

The relationship between the internal register and the external data bus is as follows:

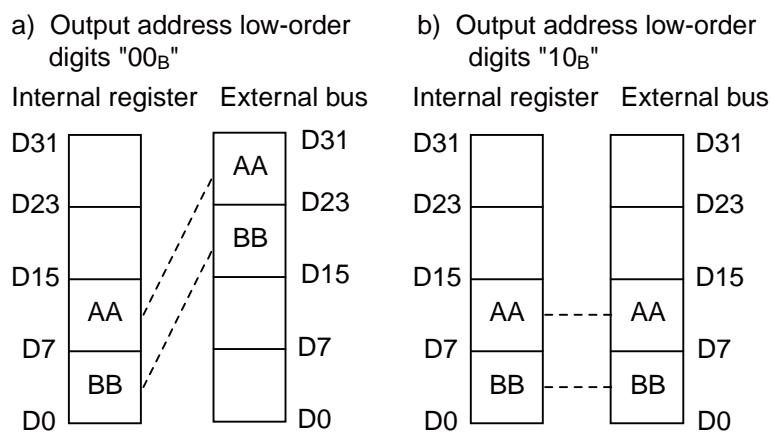
- Word access (when LD/ST instruction executed)

Figure 4.4-4 Relationship between Internal Register and External Data Bus for Word Access



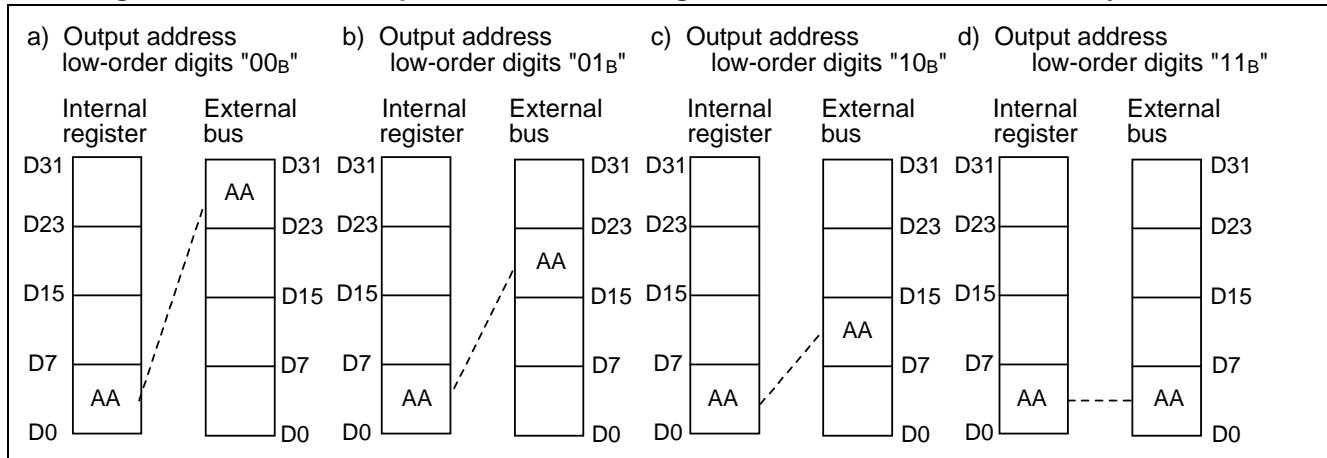
- Halfword access (LDUH/STH instruction executed)

Figure 4.4-5 Relationship between the Internal Register and External Data Bus for Halfword Access



- Byte access (LDUB/STB instruction executed)

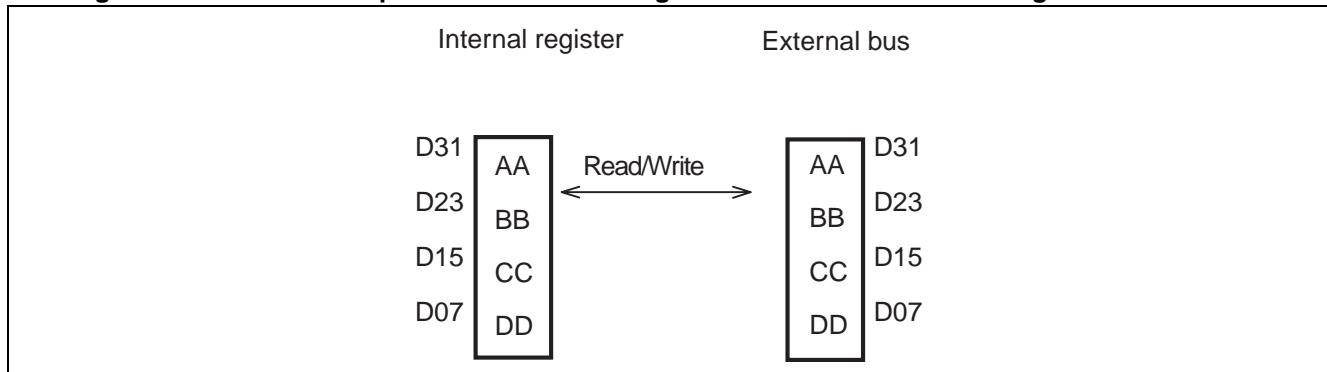
Figure 4.4-6 Relationship between Internal Register and External Data Bus for Byte Access



■ Data Bus Width

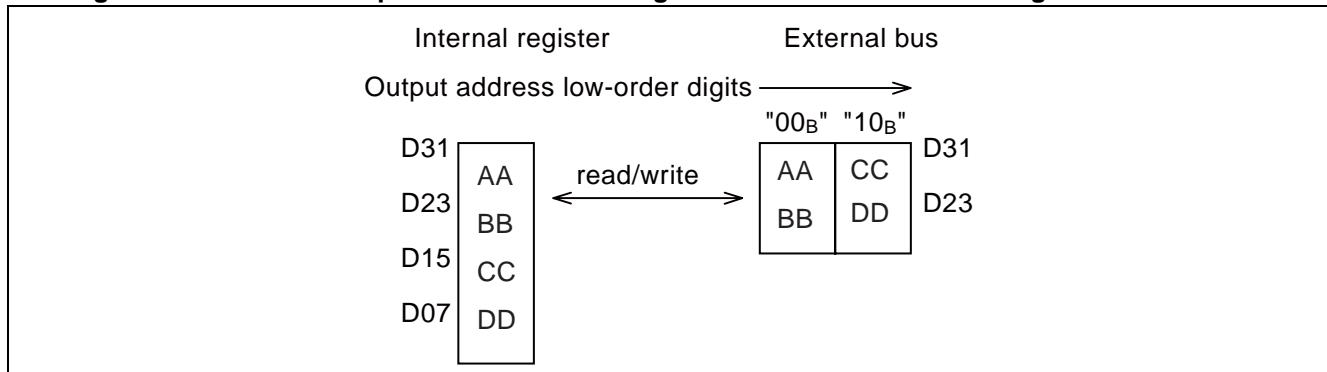
- 32-bit bus width

Figure 4.4-7 Relationship between Internal Register and External Bus Having 32-bit Bus Width



- 16-bit bus width

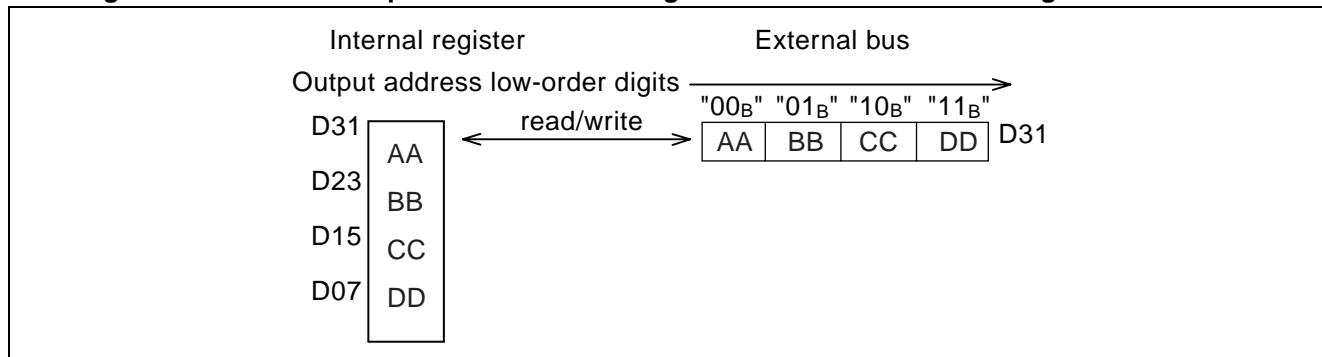
Figure 4.4-8 Relationship between Internal Register and External Bus Having 16-Bit Bus Width



CHAPTER 4 EXTERNAL BUS INTERFACE

- 8-bit bus width

Figure 4.4-9 Relationship between Internal Register and External Bus Having 8-bit Bus Width



■ External Bus Access

Figure 4.4-10, Figure 4.4-11 and Figure 4.4-12 show external bus access (32-bit/16-bit/8-bit bus width) separately for word, halfword, and byte access. The following items are included in Figure 4.4-10, Figure 4.4-11 and Figure 4.4-12:

- Access byte location
- Program address and output address
- Bus access count

PA1/PA0 : Lower 2 bits of address specified by program

Output A1/A0 : Lower 2 bits of output address

: The top byte location of output address

+ : The data byte location to access

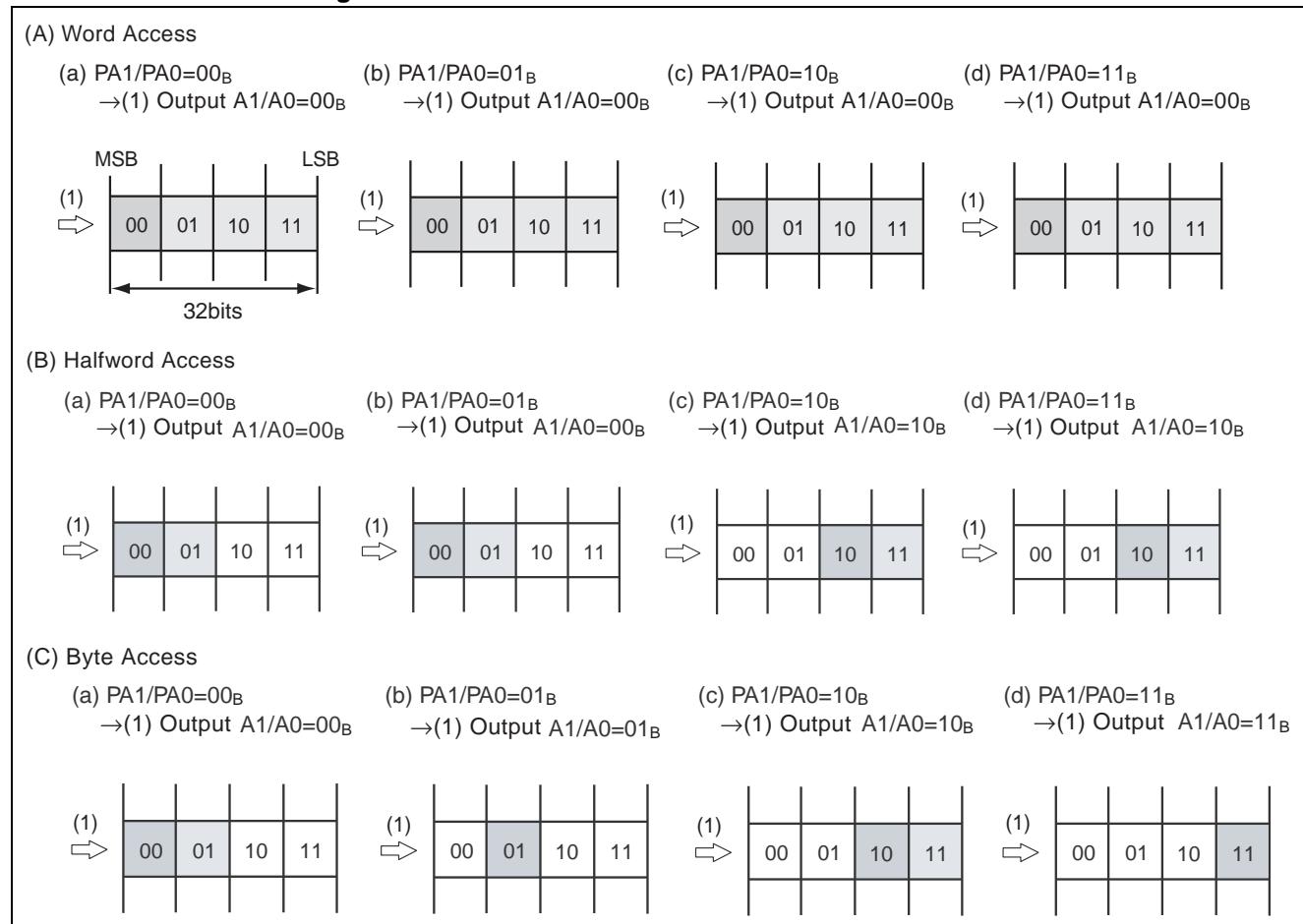
(1) to (4) : Bus access count

The FR family does not detect misalignment errors.

Therefore, for word access, the lower two bits of the output address are always "00_B" regardless of whether "00_B", "01_B", "10_B", or "11_B" is specified as the lower 2 bits by the program. For halfword access, the lower 2 bits of the output address are "00_B" if the lower two bits specified by the program are "00_B" or "01_B", and are "10_B" if "10_B" or "11_B".

○ 32-bit bus width

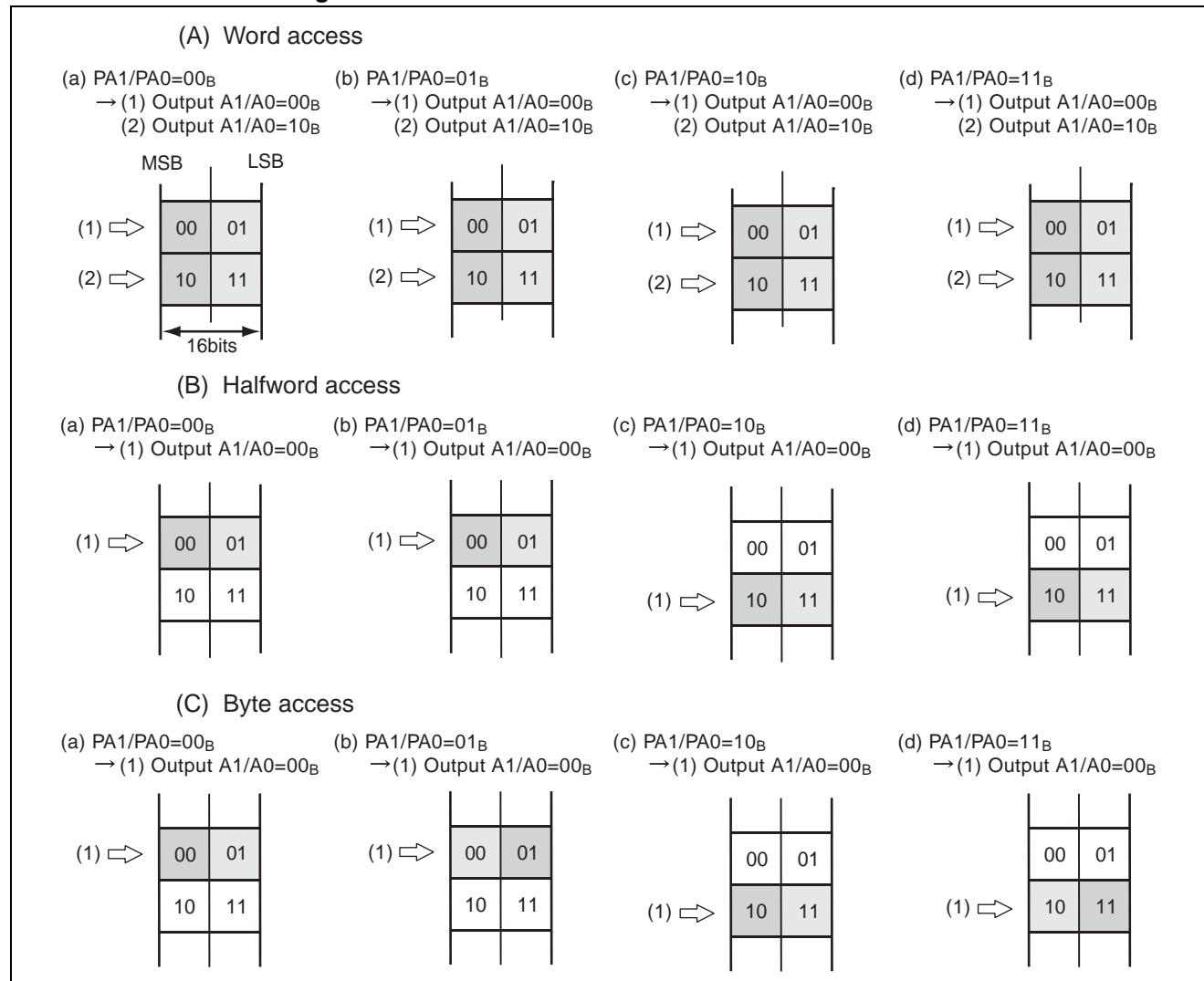
Figure 4.4-10 External Bus Access for 32-bit Bus Width



CHAPTER 4 EXTERNAL BUS INTERFACE

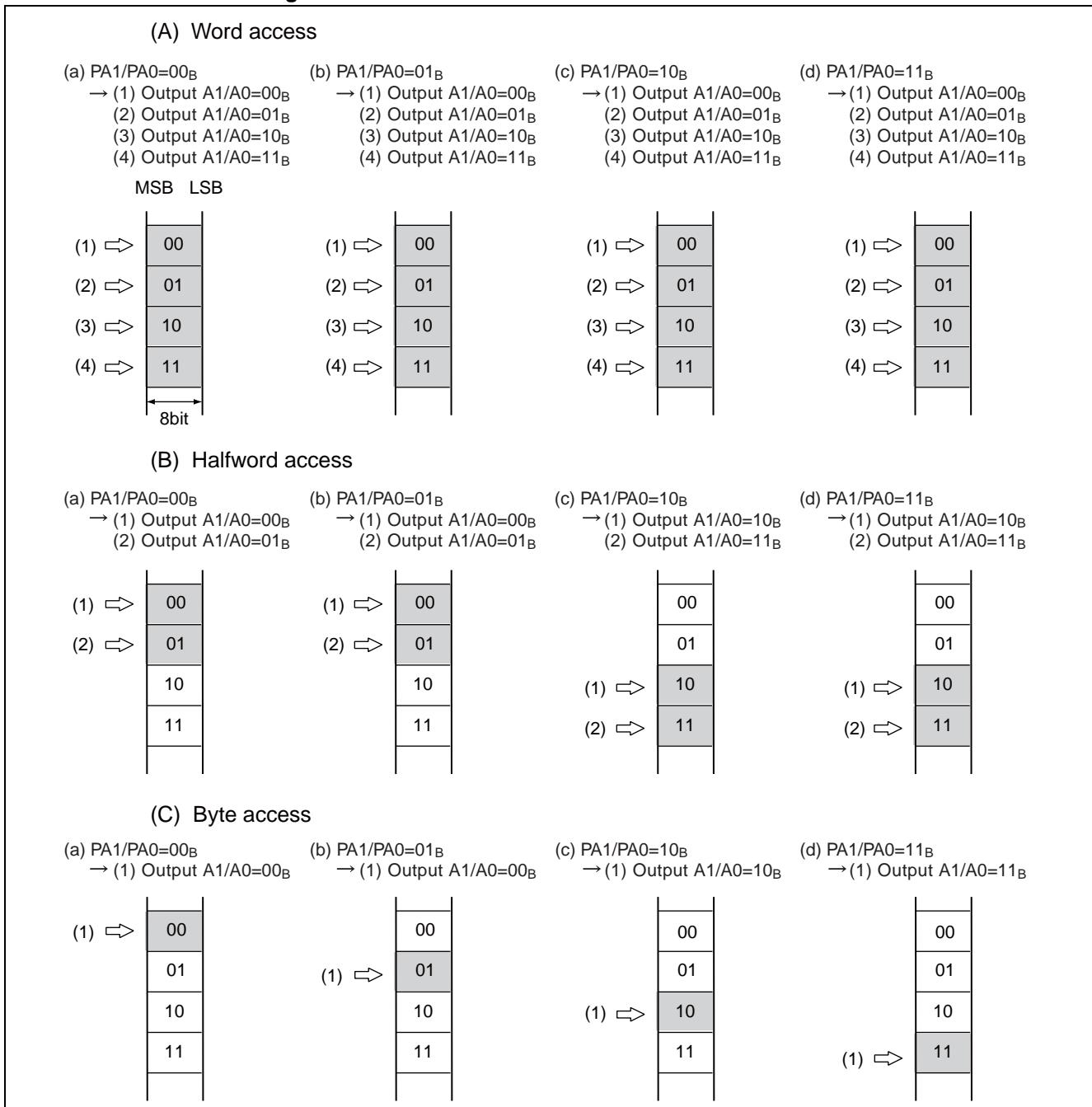
○ 16-bit bus width

Figure 4.4-11 External Bus Access for 16-bit Bus Width



○ 8-bit bus width

Figure 4.4-12 External Bus Access for 8-bit Bus Width

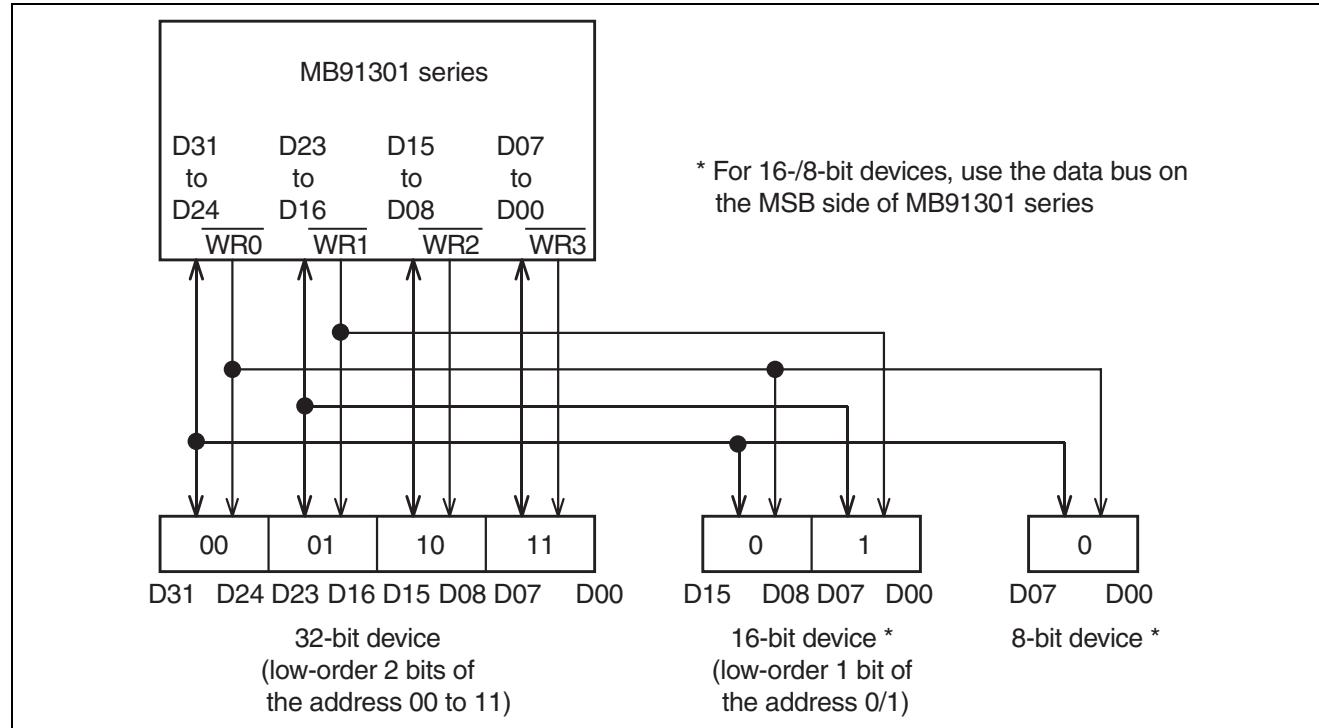


CHAPTER 4 EXTERNAL BUS INTERFACE

■ Example of Connection with External Devices

Figure 4.4-13 shows an example of connecting the FR family to external devices.

Figure 4.4-13 Example of Connecting the FR Family to External Devices



4.4.2 Little Endian Bus Access

On the FR family, each chip can switch between the big endian and little endian methods independently from others, except the CS0 area. When the LEND bit in the area configuration register (ACR) is set to "1", the corresponding area is handled in the little endian method.

Little endian bus access on the FR family is implemented by using the bus access operation used for the big endian method. Basically, the order of output addresses and control signal output are the same as for the big endian method and the byte locations on the data bus are swapped in accordance with the bus width.

Note that, when a connection is made, the big endian area and the little endian area must be kept physically separate.

■ Differences between Little Endian and Big Endian

The following explains the differences between little endian and big endian.

- The order of addresses that are output is the same for little endian and big endian.
- Word access

The byte data on the MSB side for big endian address A1 and A0 " 00_B " becomes byte data on the LSB side when the little endian method is used.

For a word access, the locations of all four bytes in the word are reversed:

- Halfword access

The byte data on the MSB side for the big endian address "A0" becomes byte data on the LSB side when the little endian method is used.

For halfword access, the byte locations of two bytes in the halfword are reversed.

- Byte access

There is no difference between little endian and big endian.

- The data bus/control signal used for 32/16/8-bit bus width is the same for little endian and big endian.

■ Restrictions on the LittleEndian Area

- If prefetch is enabled for a little endian area, always use word access to access the area. If data written to the prefetch buffer is accessed with any length other than word length, the correct endian conversion is not performed and the wrong data will be read. The reason is hardware restrictions related to the endian conversion mechanism.
- Do not place any instruction code in a little endian area.

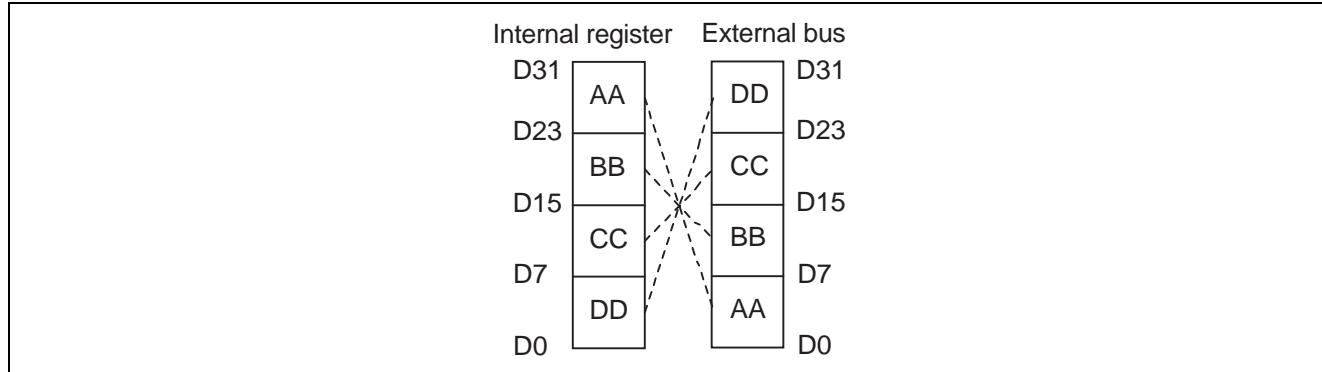
CHAPTER 4 EXTERNAL BUS INTERFACE

■ Data Format

The relationship between the internal register and external data bus is as follows:

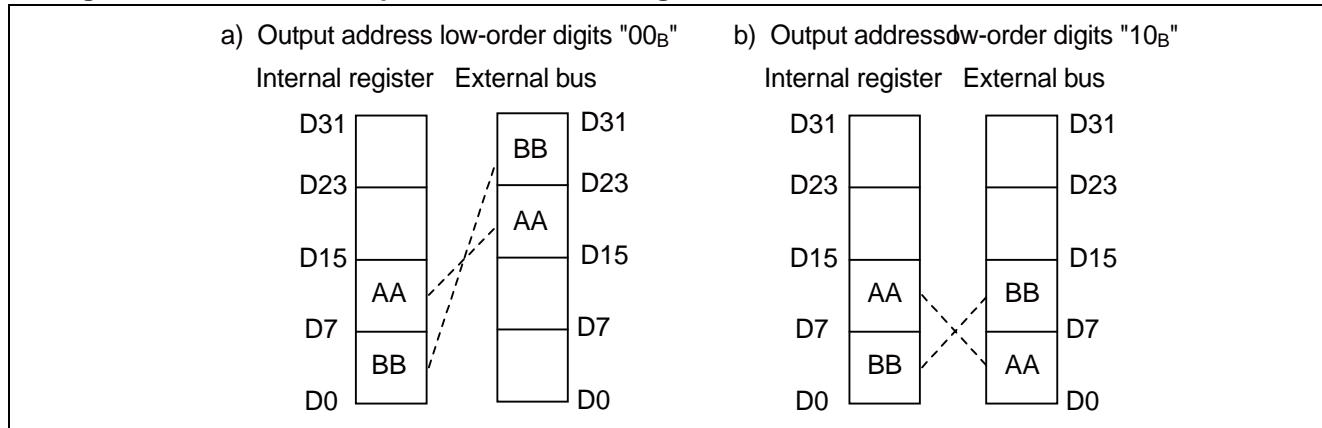
- Word access (when executing the LD/ST instructions)

Figure 4.4-14 Relationship between the Internal Register and External Data Bus for Word Access



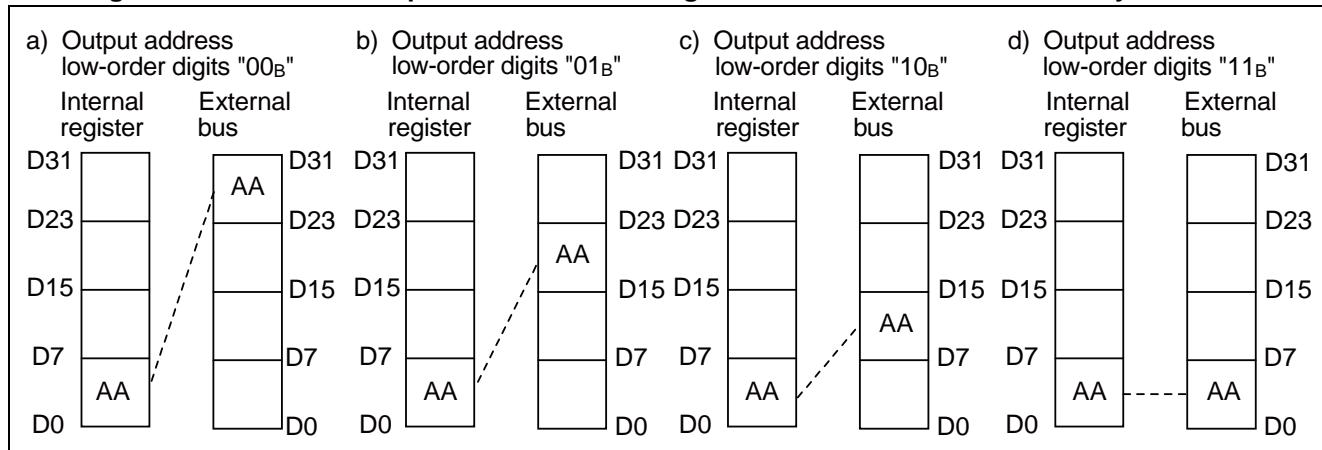
- Halfword access (when executing the LDUH/STH instructions)

Figure 4.4-15 Relationship between Internal Register and External Data Bus for Halfword Access



- Byte access (when executing the LDUB/STB instructions)

Figure 4.4-16 Relationship between Internal Register and External Data Bus for Byte Access

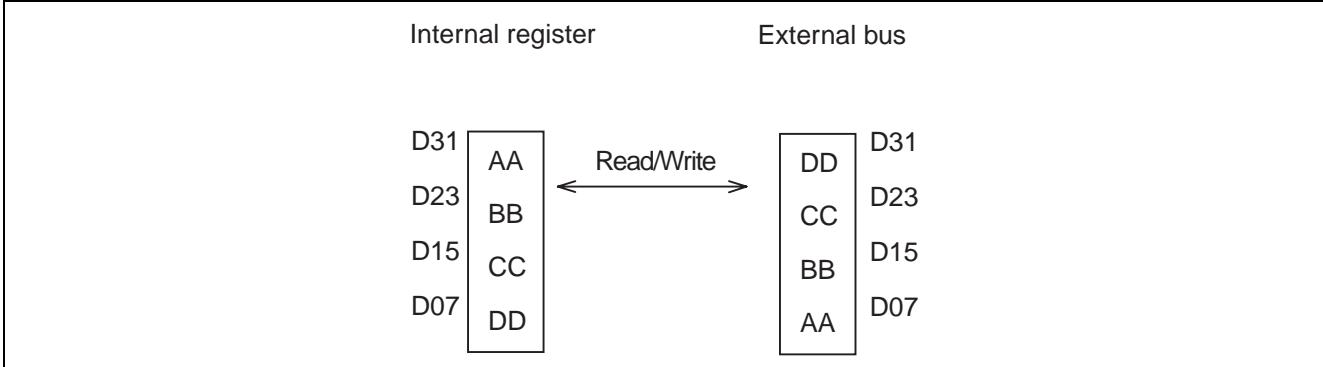


■ Data Bus Width

The following shows the relationships between the internal register and external data bus for each data bus width.

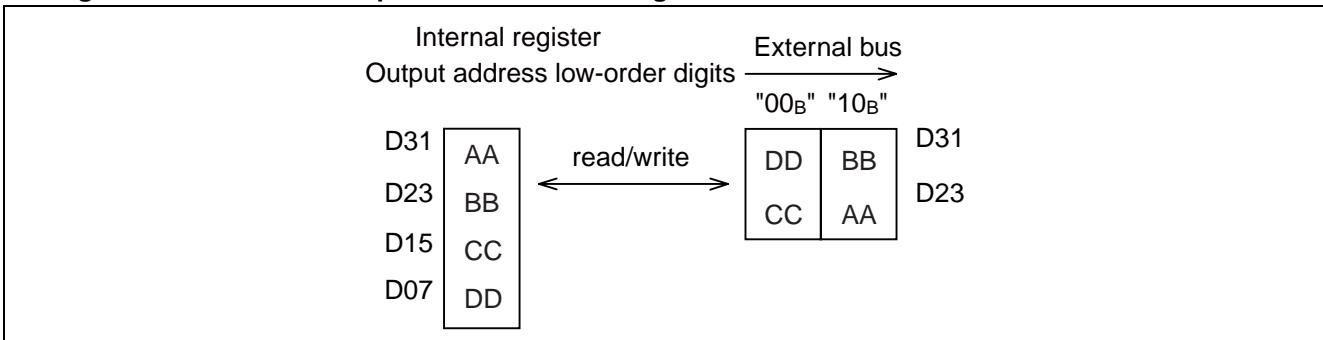
○ 32-bit bus width

Figure 4.4-17 Relationship between Internal Register and External Bus Data for 32-bit Bus Width



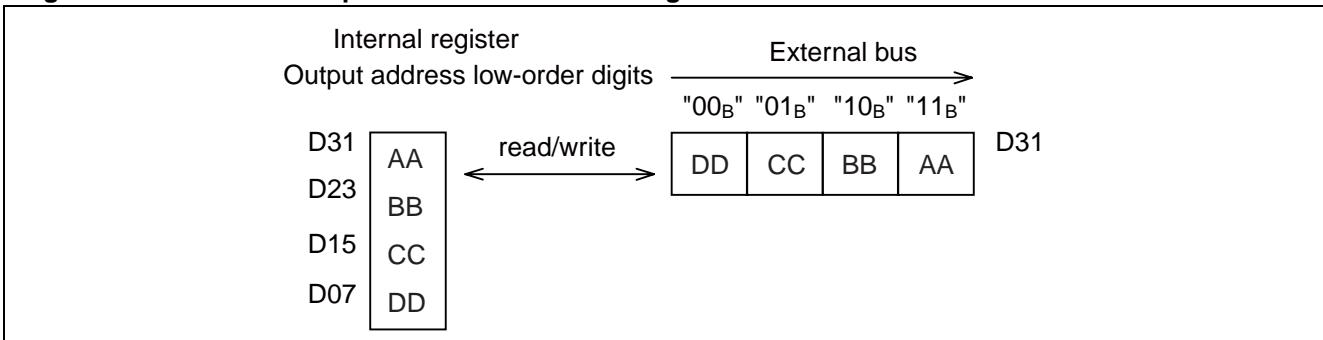
○ 16-bit bus width

Figure 4.4-18 Relationship between Internal Register and External Bus Data for 16-bit Bus Width



○ 8-bit bus width

Figure 4.4-19 Relationship between the Internal Register and External Data Bus in the 8-bit Bus Width



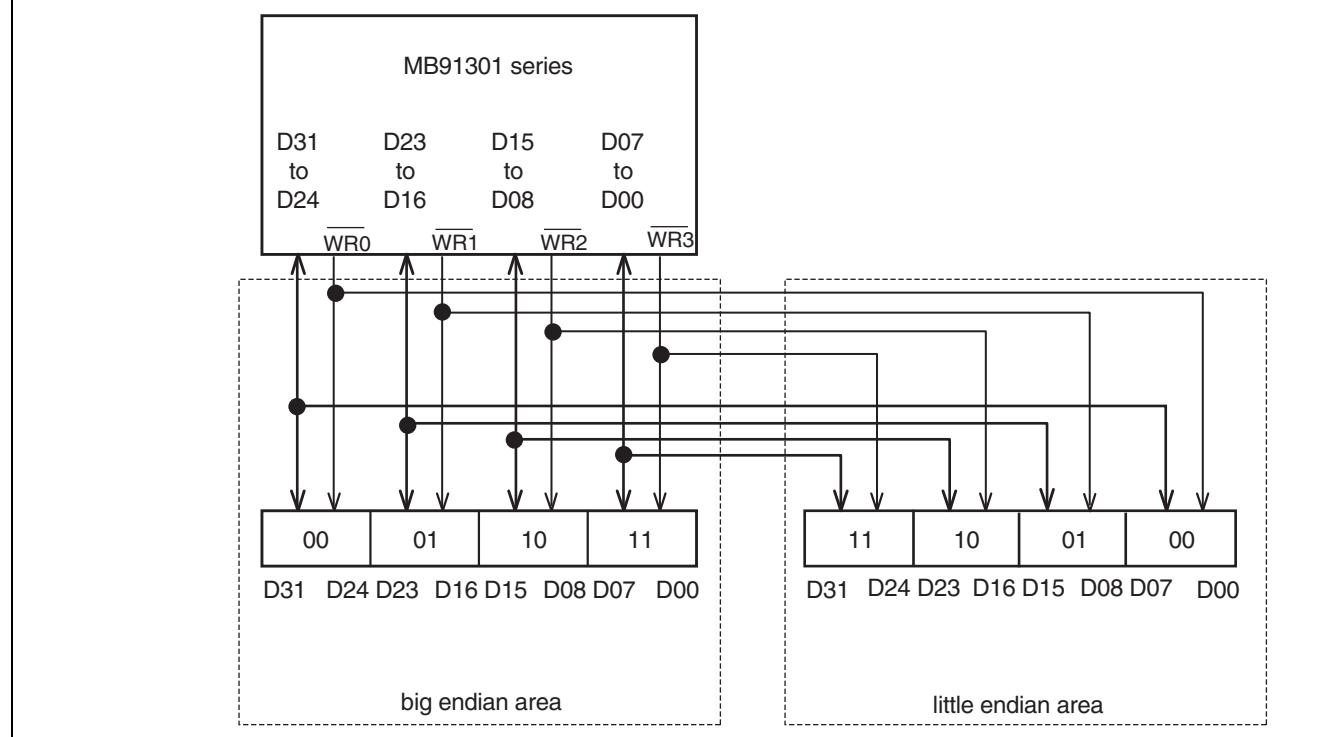
CHAPTER 4 EXTERNAL BUS INTERFACE

■ Examples of Connection with External Devices

The following shows examples of connecting the FR family to external devices for each bus width.

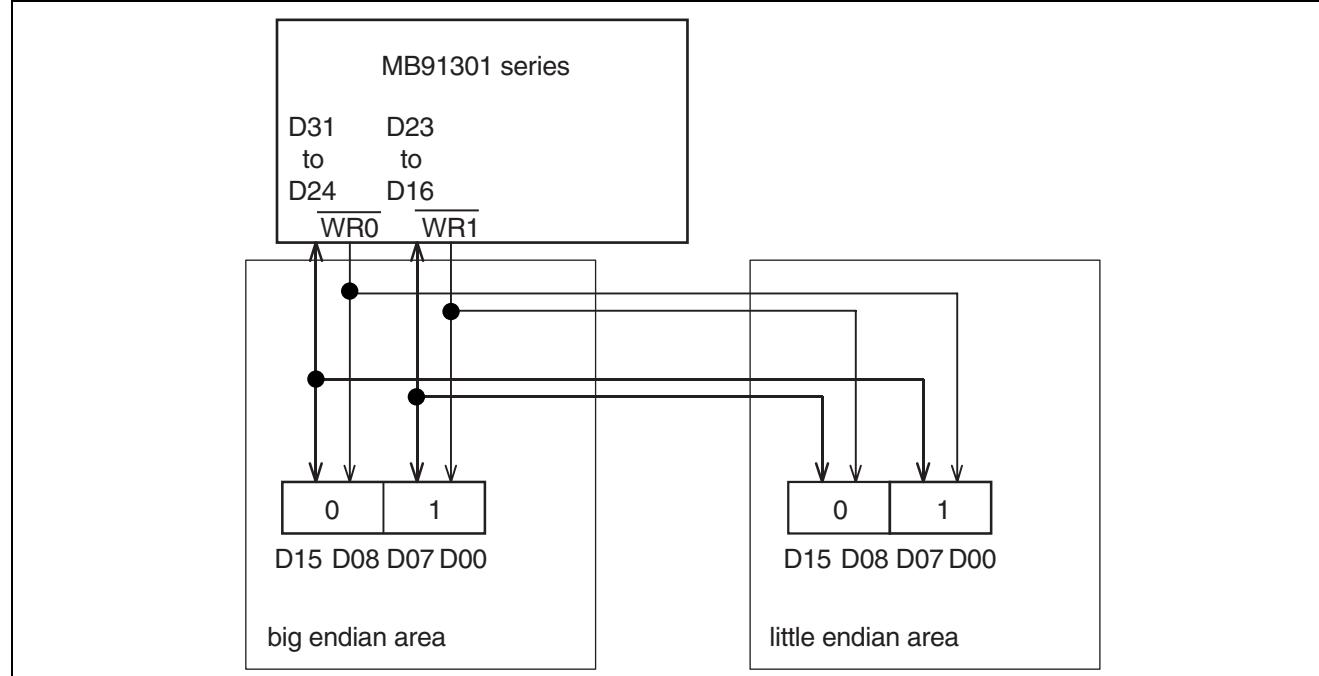
○ 32-bit bus width

Figure 4.4-20 Example of Connecting the FR Family to External Devices (32-bit Bus Width)



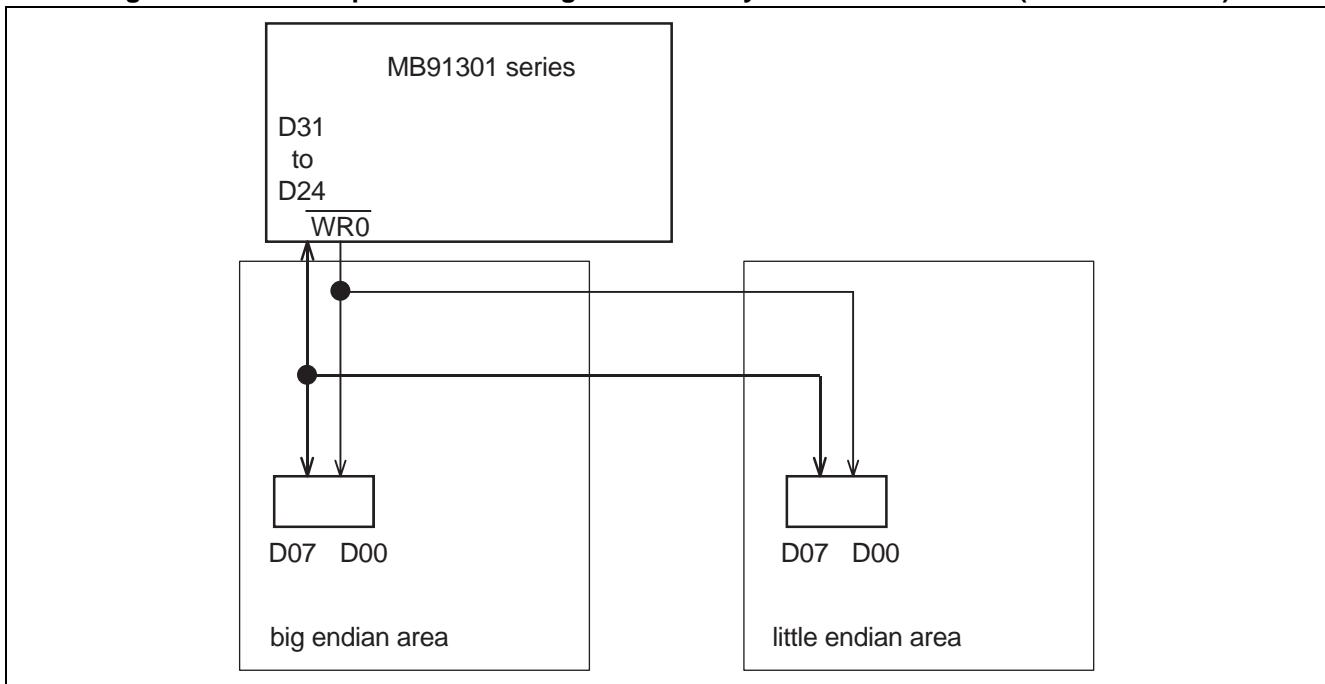
○ 16-bit bus width

Figure 4.4-21 Example of Connecting the FR Family to External Devices (16-bit Bus Width)



- 8-bit bus width

Figure 4.4-22 Example of Connecting the FR Family to External Devices (8-bit Bus Width)



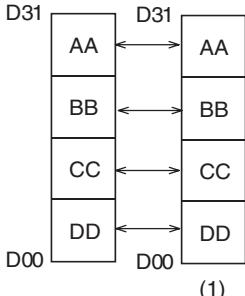
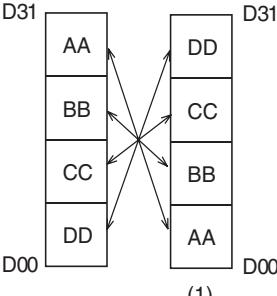
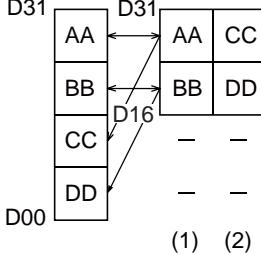
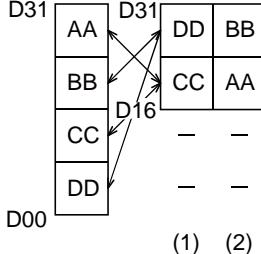
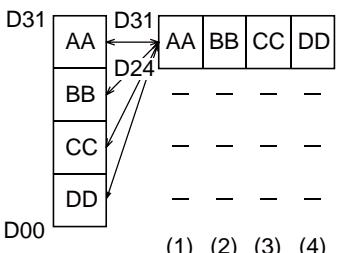
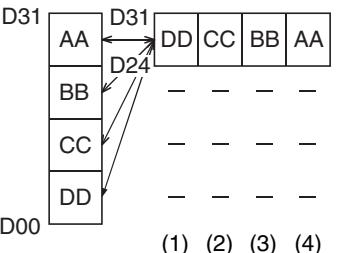
4.4.3 Comparison of Big Endian and Little Endian External Access

This section shows a comparison of big endian and little endian external access in word access, halfword access, and byte access for each bus width.

When data bus is connected according to examples of connecting to external devices as shown in Sections "4.4.1 Big Endian Bus Access" and "4.4.2 Little Endian Bus Access", all accesses become big endian for the internal register as follows.

The following figures describe big endian area and little endian area separately.

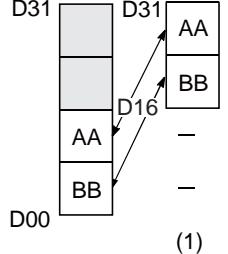
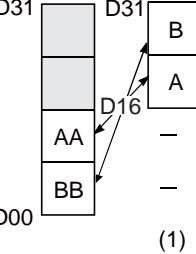
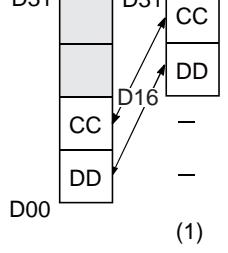
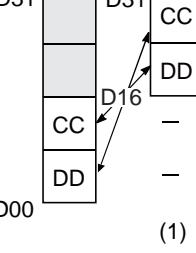
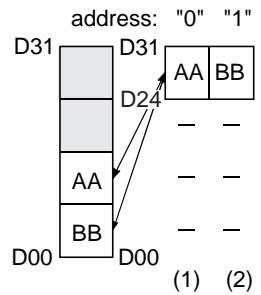
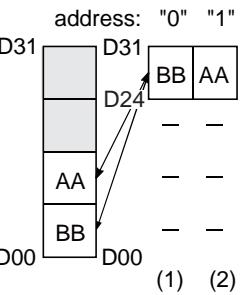
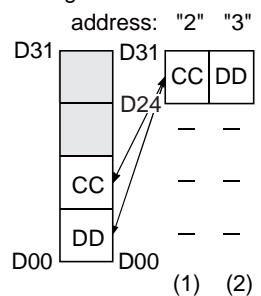
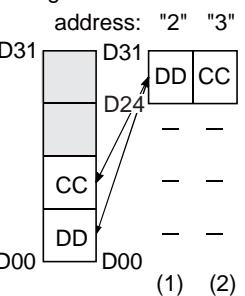
■ Word Access

	Big endian mode			Little endian mode		
32-bit bus width	Internal register External terminal address: Lower 2-bit: "0"	D31  (1)	Control terminal	Internal register External terminal address : "0"	 (1)	Control terminal
16-bit bus width	Internal register External terminal address: "0" "2"	D31  (1) (2)	Control terminal	Internal register External terminal address: "0" "2"	 (1) (2)	Control terminal
8-bit bus width	Internal register External terminal address: "0" "1" "2" "3"	D31  (1) (2) (3) (4)	Control terminal	Internal register External terminal address: "0" "1" "2" "3"	 (1) (2) (3) (4)	Control terminal

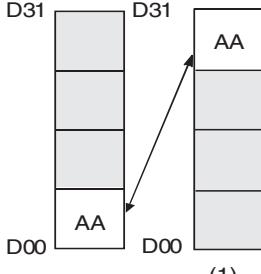
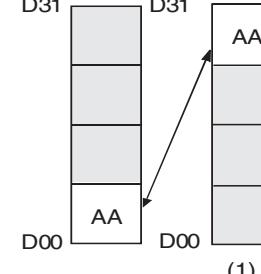
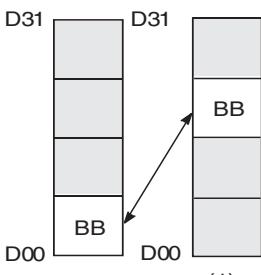
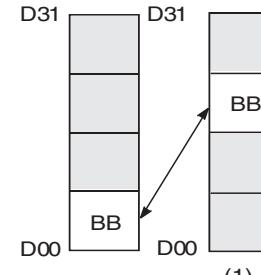
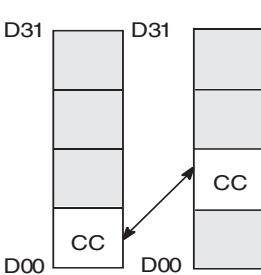
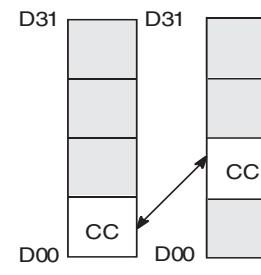
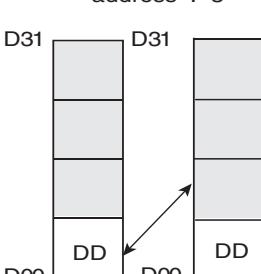
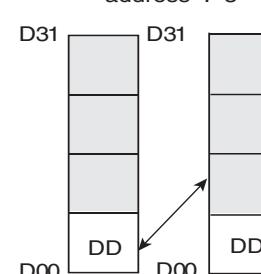
■ Halfword Access

	Big endian mode	Little endian mode
32-bit bus width	<p>Internal register address : "0"</p> <p>(1)</p>	<p>Internal register address : "0"</p> <p>(1)</p>
	<p>Internal register address : "2"</p> <p>(1)</p>	<p>Internal register address : "2"</p> <p>(1)</p>

CHAPTER 4 EXTERNAL BUS INTERFACE

	Big endian mode			Little endian mode		
	Internal register address: "0"	External terminal	Control terminal	Internal register address: "0"	External terminal	Control terminal
16-bit bus width	D31  D00 (1)	AA BB	WR0 WR1	D31  D00 (1)	B A AA BB	WR0 WR1
	D31  D00 (1)	CC DD	WR0 WR1	D31  D00 (1)	CC DD	WR0 WR1
8-bit bus width	D31  D00 (1) (2)	AA BB	WR0 WR1	D31  D00 (1) (2)	BB AA	WR0 WR1
	D31  D00 (1) (2)	CC DD	WR0 WR1	D31  D00 (1) (2)	DD CC	WR0 WR1

■ Byte Access

	Big endian mode			Little endian mode		
32-bit bus width	Internal register address : "0"	External terminal address : "0"	Control terminal	Internal register address : "0"	External terminal address : "0"	Control terminal
	D31 	AA	WR0	D31 	AA	WR0
	(1)		—	(1)		—
	Internal register address : "1"	External terminal address : "1"	Control terminal	Internal register address : "1"	External terminal address : "1"	Control terminal
	D31 	BB	WR1	D31 	BB	WR1
	(1)		—	(1)		—
	Internal register address : "2"	External terminal address : "2"	Control terminal	Internal register address : "2"	External terminal address : "2"	Control terminal
	D31 	CC	WR2	D31 	CC	WR2
	(1)		—	(1)		—
	Internal register address : "3"	External terminal address : "3"	Control terminal	Internal register address : "3"	External terminal address : "3"	Control terminal
	D31 	DD	WR3	D31 	DD	WR3
	(1)		—	(1)		—

CHAPTER 4 EXTERNAL BUS INTERFACE

	Big endian mode			Little endian mode		
16-bit bus width	Internal register address: "0"	External terminal	Control terminal	Internal register address: "0"	External terminal	Control terminal
	D31 AA D16 AA D00	D31 AA	$\overline{WR0}$ — — —	D31 AA D16 AA D00	D31 AA	$\overline{WR0}$ — — —
	Internal register address: "1"	External terminal	Control terminal	Internal register address: "1"	External terminal	Control terminal
	D31 BB D16 BB D00	D31 BB	— $\overline{WR1}$ — —	D31 BB D16 BB D00	D31 BB	— $\overline{WR1}$ — —
	Internal register address: "2"	External terminal	Control terminal	Internal register address: "2"	External terminal	Control terminal
	D31 CC D16 CC D00	D31 CC	$\overline{WR0}$ — — —	D31 CC D16 CC D00	D31 CC	$\overline{WR0}$ — — —
	Internal register address: "3"	External terminal	Control terminal	Internal register address: "3"	External terminal	Control terminal
	D31 DD D16 DD D00	D31 DD	— $\overline{WR1}$ — —	D31 DD D16 DD D00	D31 DD	— $\overline{WR1}$ — —

	Big endian mode			Little endian mode		
8-bit bus width	Internal register address: "0"	External terminal	Control terminal	Internal register address: "0"	External terminal	Control terminal
	D31 D24 D00	AA	$\overline{WR0}$ — —	D31 D24 D00	AA	$\overline{WR0}$ — —
	Internal register address: "1"	External terminal	Control terminal	Internal register address: "1"	External terminal	Control terminal
	D31 D24 D00	BB	$\overline{WR0}$ — —	D31 D24 D00	BB	$\overline{WR0}$ — —
	Internal register address: "2"	External terminal	Control terminal	Internal register address: "2"	External terminal	Control terminal
	D31 D24 D00	CC	$\overline{WR0}$ — —	D31 D24 D00	CC	$\overline{WR0}$ — —
	Internal register address: "3"	External terminal	Control terminal	Internal register address: "3"	External terminal	Control terminal
	D31 D24 D00	DD	$\overline{WR0}$ — —	D31 D24 D00	DD	$\overline{WR0}$ — —

4.5 Operation of the Ordinary Bus Interface

This section explains operation of the ordinary bus interface.

■ Ordinary Bus Interface

For the ordinary bus interface, two clock cycles are the basic bus cycles for both read access and write access.

The following operational phases of the ordinary bus interface are explained below with the use of a timing chart.

- Basic timing (for successive accesses)
- \overline{WRn} + byte control type
- Read -> write
- Write -> write
- Auto-wait cycle
- External wait cycle
- Synchronous write enable output
- \overline{CSn} delay setting
- $\overline{CSn} \rightarrow \overline{RD}/\overline{WRn}$ setup, $\overline{RD}/\overline{WRn} \rightarrow \overline{CSn}$ hold setting
- DMA fly-by transfer (I/O -> memory)
- DMA fly-by transfer (memory -> I/O)

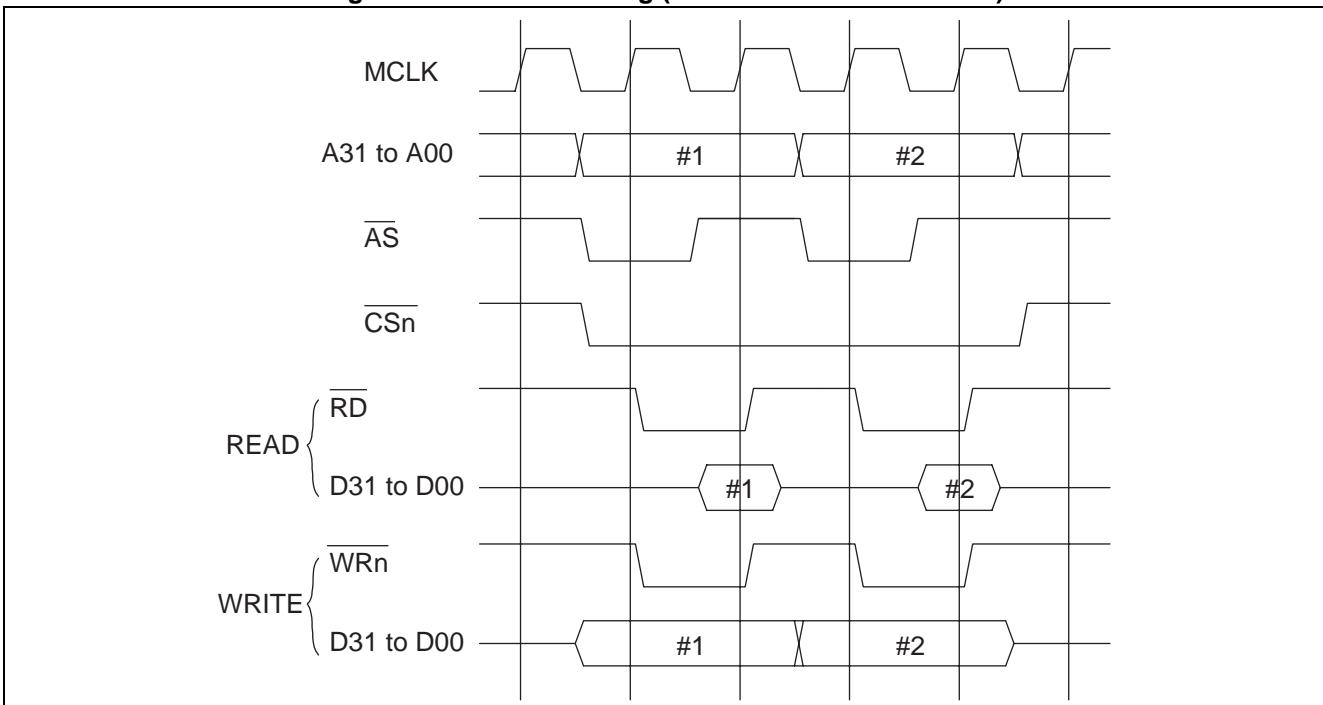
4.5.1 Basic Timing

This section shows the basic timing for successive accesses.

■ Basic Timing (For Successive Accesses)

Figure 4.5-1 shows the operation timing for (TYP3 to TYP0 = 0000_B, AWR = 0008_H).

Figure 4.5-1 Basic Timing (For Successive Accesses)



- AS is asserted for one cycle in the bus access start cycle.
- A31 to A00 continues to output the address of the location of the start byte in word/halfword/byte access from the bus access start cycle to the bus access end cycle.
- If the W02 bit of the AWR0 to AWR7 registers is "0", CS0 to CS7 are asserted at the same timing as AS. For successive accesses, CS0 to CS7 are not negated. If the W00 bit of the AWR register is "0", CS0 to CS7 are negated after the bus cycle ends. If the W00 bit is "1", CS0 to CS7 are negated one cycle after bus access ends.
- RD and WR0 to WR3 are asserted from the 2nd cycle of the bus access. Negation occurs after the wait cycle of bits W15 to W12 of the AWR register is inserted. The timing of asserting RD and WR0 to WR3 can be delayed by one cycle by setting the W01 bit of the AWR register to "1".
- If a setting is made so that WR0 to WR3 is used like TYP3 to TYP0=0x0_B, WRn is always "H".
- For read access, D31 to D00 is read when MCLK rises in the cycle in which the wait cycle ended after RD was asserted.
- For write access, data output to D31 to D00 starts at the timing at which WR0 to WR3 are asserted.

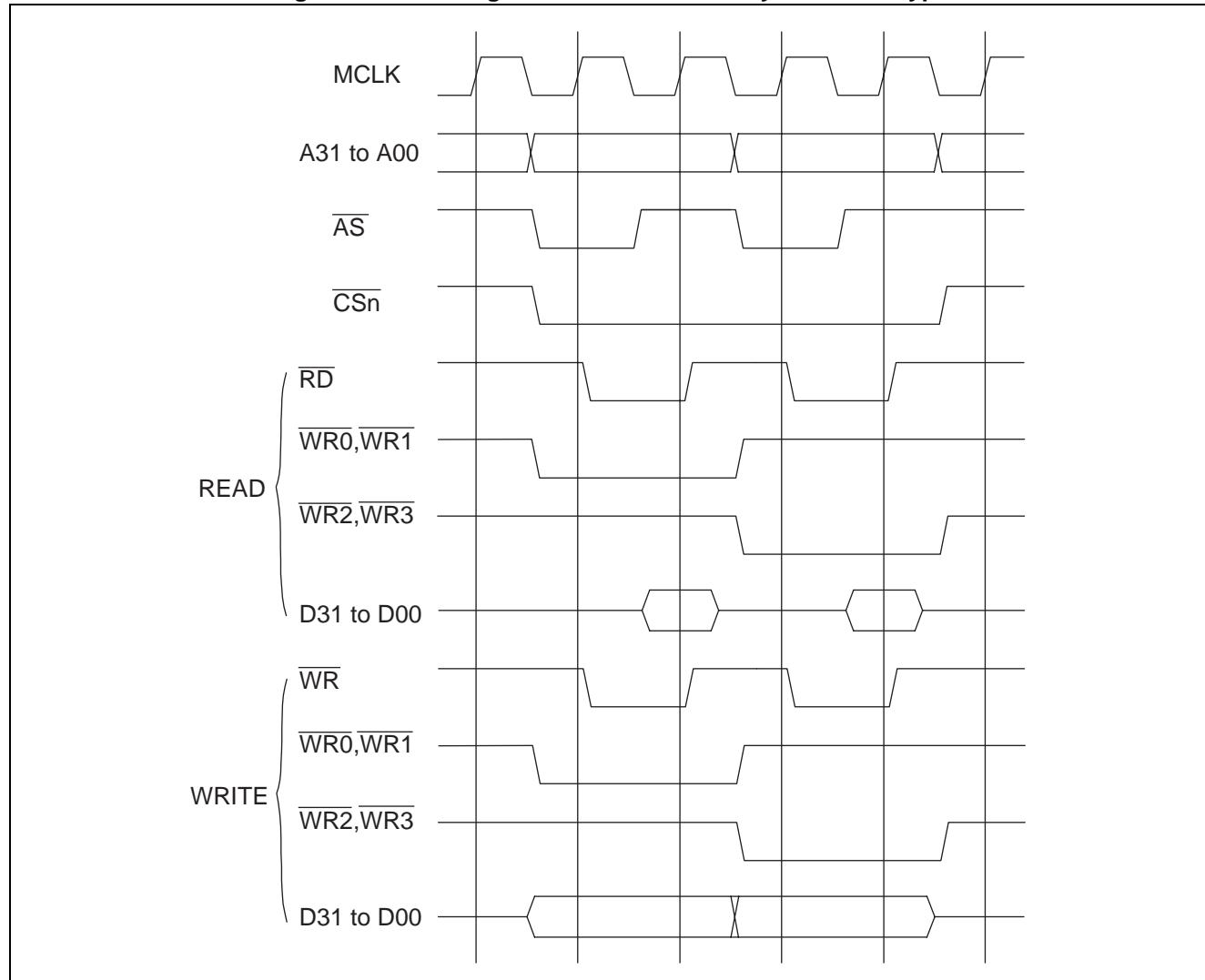
4.5.2 Operation of \overline{WRn} + Byte Control Type

This section shows the operation timing for the \overline{WRn} + byte control type.

■ Operation Timing of the \overline{WRn} + Byte Control Type

Figure 4.5-2 shows the operation timing for (TYP3 to TYP0 = 0010_B, AWR = 0008_H).

Figure 4.5-2 Timing Chart for the \overline{WRn} + Byte Control Type



- Operation of \overline{AS} , \overline{CS} , \overline{RD} , A31 to A00, and D31 to D00 is the same as that described in "4.5.1 Basic Timing".
- \overline{WR} is asserted from the 2nd cycle of the bus access. Negation occurs after the wait cycle of bits W15 to W12 of the AWR register is inserted. The timing of asserting \overline{RD} and $\overline{WR0}$ to $\overline{WR3}$ can be delayed by one cycle by setting the W01 bit of the AWR register to "1". (Operation is the same as that for $\overline{WR0}$ to $\overline{WR3}$ described in "4.5.1 Basic Timing".)

- $\overline{WR0}$ to $\overline{WR3}$ indicate the byte location expressed with negative logic when they are used for access as the byte enable signal. Assertion continues from the bus access start cycle to the bus access end cycle and changes at the same timing as the address timing. The byte location for access is indicated for both read access and write access.
- For write access, data output to D31 to D00 starts at the timing at which $\overline{WR0}$ is asserted. If the areas defined by TYP3 to TYP0=0x0xB ($\overline{WR0}$ to $\overline{WR3}$ used) and TYP3 to TYP0=0x1xB (\overline{WRn} + byte control) are mixed, be sure to make the following setting for all areas that will be used. (For details, see "4.13 Notes on Using the External Bus Interface".)
 - Set at least one read -> write idle cycle.
 - Set at least one write recovery cycle.

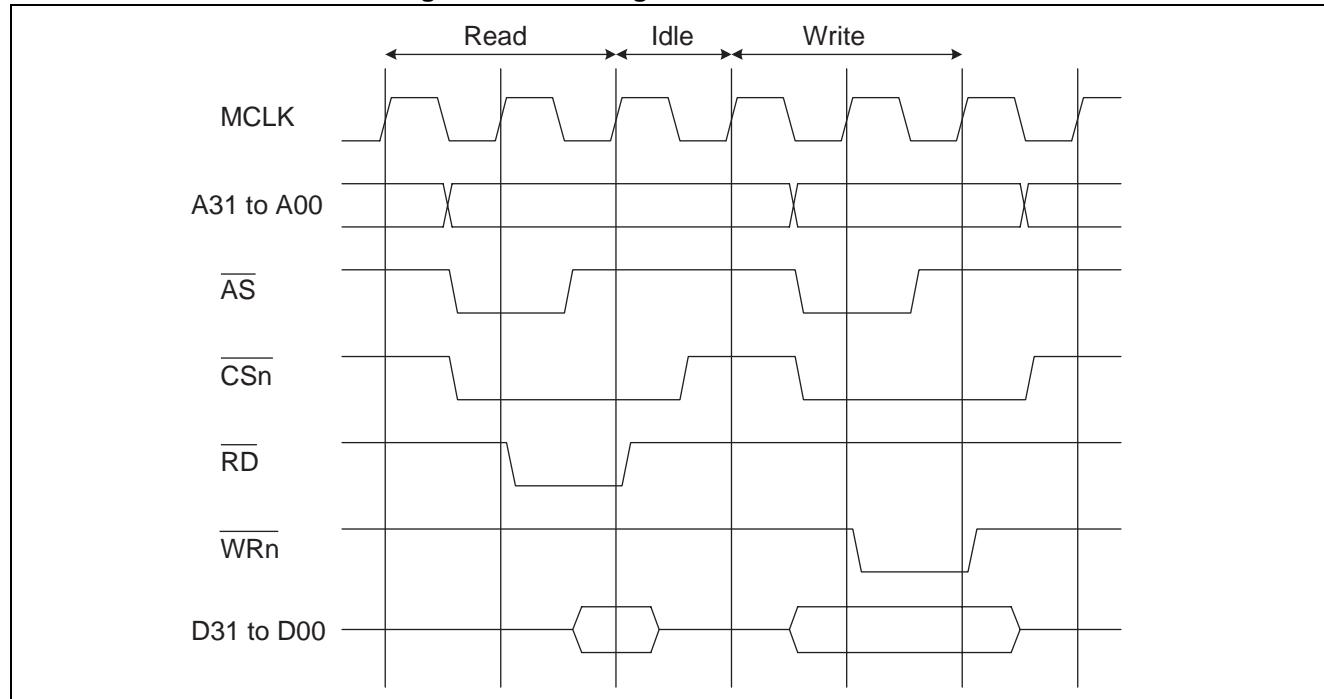
4.5.3 Read -> Write Operation

This section shows the operating timing for read -> write.

■ Operation Timing of Read -> Write

Figure 4.5-3 shows the operation timing for (TYP3 to TYP0=0000_B, AWR=0048_H).

Figure 4.5-3 Timing Chart for Read -> Write



- Setting of the W07/W06 bits of the AWR register enables 0 to 3 idle cycles to be inserted.
- Settings in the CS area on the read side are enabled.
- This idle cycle is inserted if the next access after a read access is write access or access to another area.

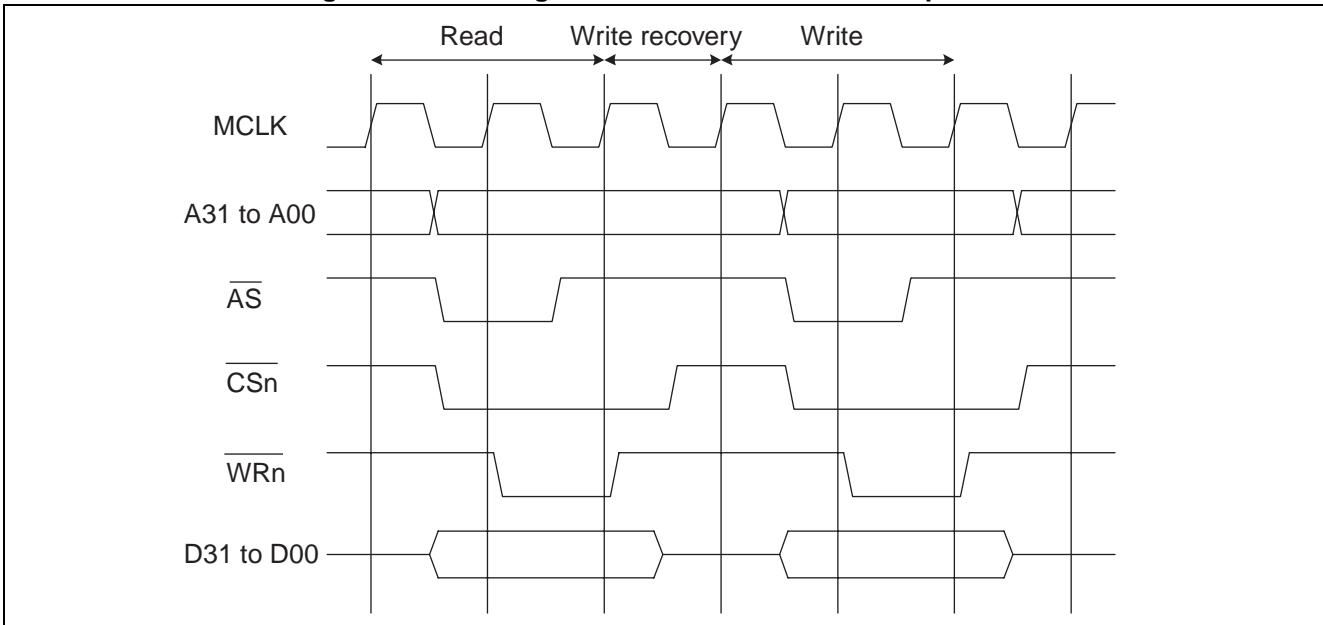
4.5.4 Write -> Write Operation

This section shows the operation timing for write -> write.

■ Write -> Write Operation

Figure 4.5-4 shows the operation timing for (TYP3 to TYP0=0000_B, AWR=0018_H).

Figure 4.5-4 Timing Chart for the Write -> Write Operation



- Setting of the W05/W04 bits of the AWR register enables 0 to 3 write recovery cycles to be inserted.
- After all of the write cycles, recovery cycles are generated.
- Write recovery cycles are also generated if write access is divided into phases for access with a bus width wider than that specified.

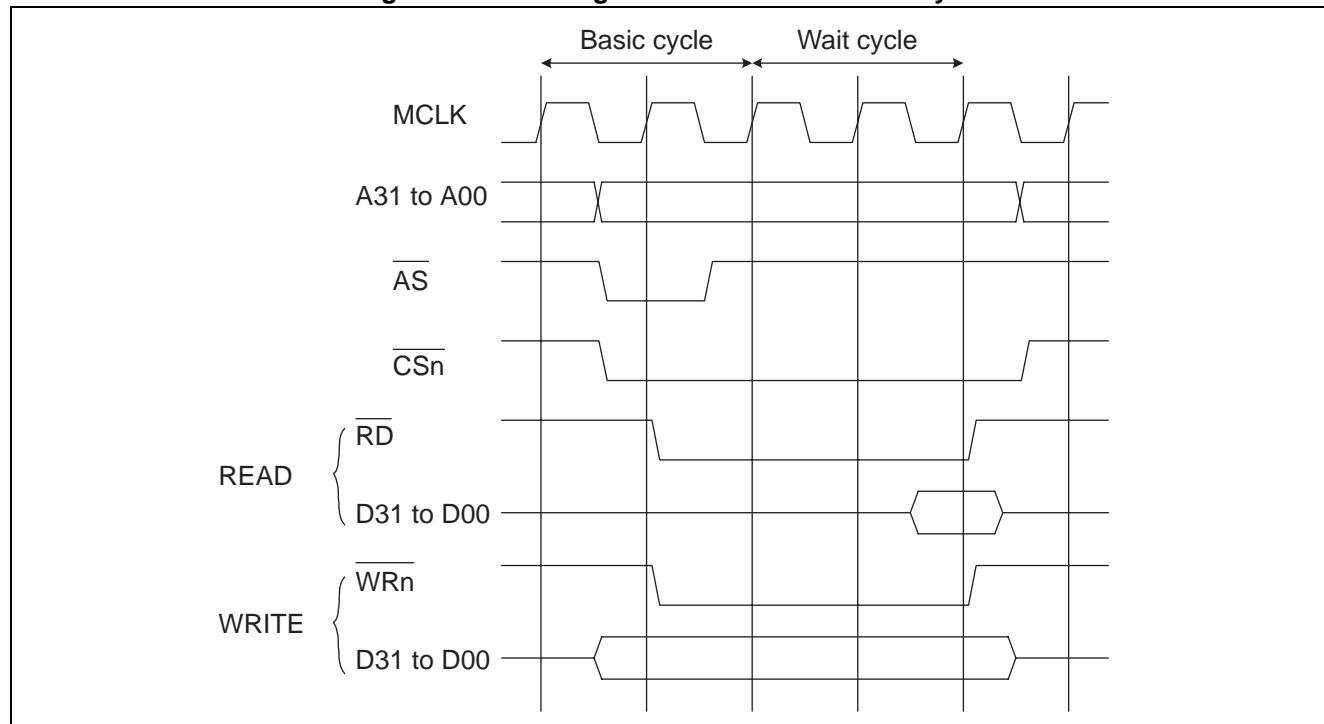
4.5.5 Auto-Wait Cycle

This section shows the operation timing for the auto-wait cycle.

■ Auto-Wait Cycle Timing

Figure 4.5-5 shows the operation timing for (TYP3 to TYP0=0000_B, AWR=2008_H).

Figure 4.5-5 Timing Chart for the Auto-Wait Cycle



- Setting of the W15 to W12 bits (first wait cycles) of the AWR register enables 0 to 15 auto-wait cycles to be set.
- In Figure 4.5-5, two auto-wait cycles are inserted, making a total of four cycles for access. If auto-wait is set, the minimum number of bus cycles is 2 cycles + (first wait cycles). For a write operation, the minimum number of bus cycles may be still longer depending on the internal state.

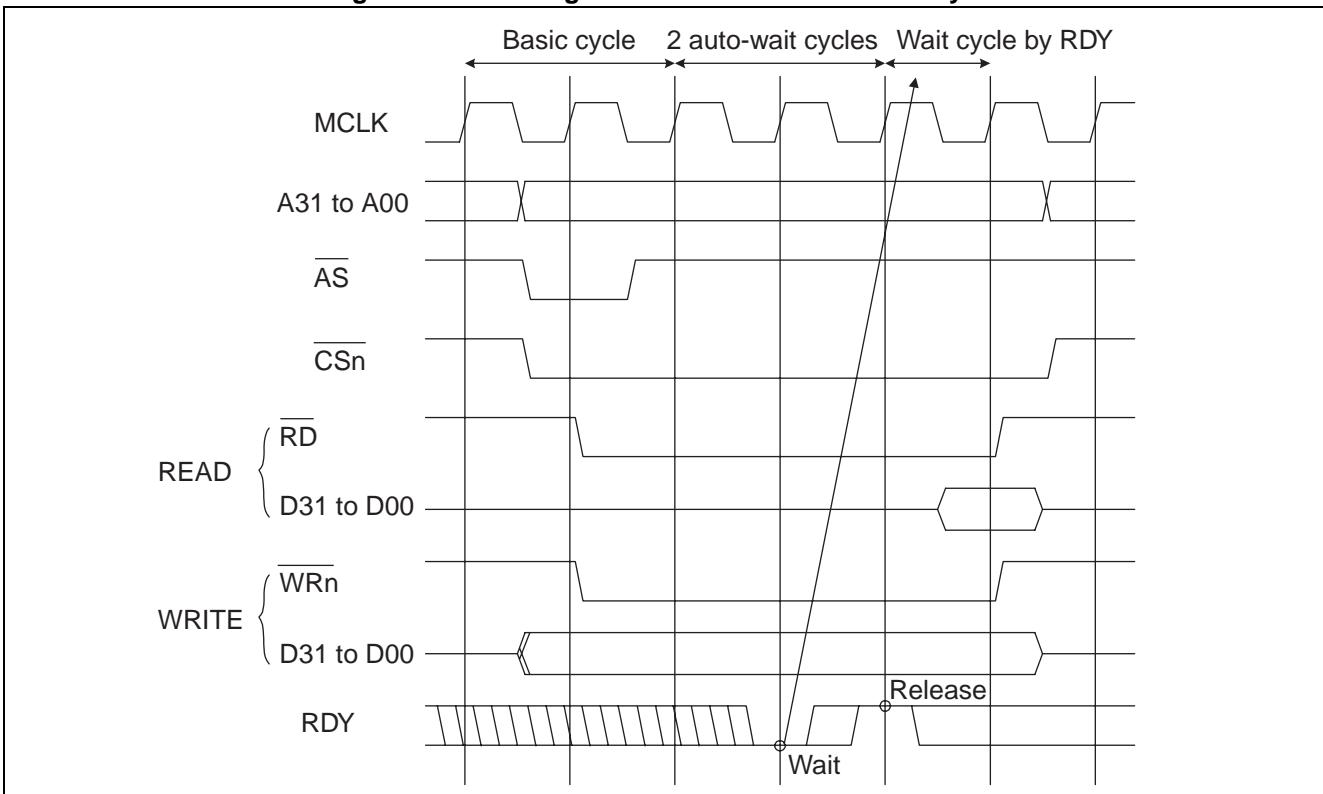
4.5.6 External Wait Cycle

This section shows the operation timing for the external wait cycle.

■ External Wait Cycle Timing

Figure 4.5-6 shows the operation timing for ($TYP3$ to $TYP0=0001_B$, $AWR=2008_H$).

Figure 4.5-6 Timing Chart for the External Wait Cycle



- Setting "1" for the $TYP0$ bit of the ACR register and enabling the external RDY input pin enables external wait cycles to be inserted.
- In Figure 4.5-6, the oblique-lined portion of the RDY pin is invalid because the wait based on the automatic wait cycle remains in effect.

The value at the RDY input pin is evaluated from the last automatic wait cycle on.

Once a wait cycle is completed, the value at the RDY input pin remains invalid until the next access cycle is started.

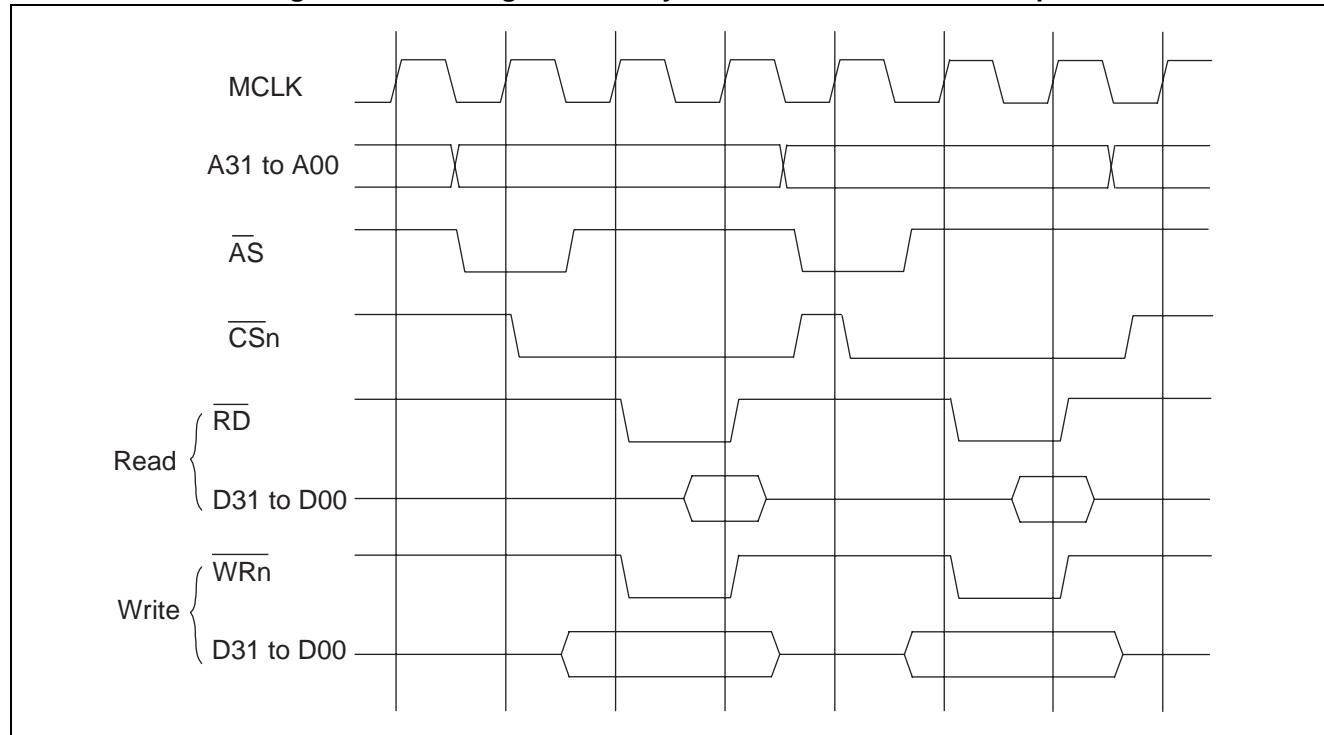
4.5.7 Synchronous Write Enable Output

This section shows the operation timing for synchronous write enable output.

■ Operation Timing for Synchronous Write Enable Output

Figure 4.5-7 shows the operation timing for (TYP3 to TYP0=0000_B, AWR=0000_H).

Figure 4.5-7 Timing Chart for Synchronous Write Enable Output



If synchronous write enable output is enabled (If the W03 bit of the AWR is "1"), operation is as follows.

- WR0 to WR3 and WR pin output asserts synchronous write enable output at the timing at which AS pin output is asserted. For a write to an external bus, the synchronous write enable output is "L". For a read from an external bus, the synchronous write enable output is "H".
- Write data is output from the external data output pin in the clock cycle following the cycle in which synchronous write enable output is asserted.
- Read strobe output (RD) functions as an asynchronous read strobe regardless of the setting of WR0 to WR3 and WR output timing. Use it as is for controlling the data I/O.

- If synchronous write enable output is used, the following restrictions apply:
Do not set the following additional wait because the timing for synchronous write enable output becomes meaningless:
 - $\overline{CS} \rightarrow \overline{RD}/\overline{WRn}$ setup (Always write "0" to the W01 bit of AWR)
 - First wait cycle setting (Always write 0000_B to bits W15 to W12 of AWR)Do not set the following access types (TYP3 to TYP0 bits (bit3 to bit0) in the ACR register) because the timing for synchronous write enable output becomes meaningless:
 - Multiplex bus setting (Always write "0" to the TYP2 bit of ACR)
 - RDY input enable setting (Always write "0" to the TYP0 bit of ACR)Always set the burst length to "1" (BST1, BST0 bits = 0) for the synchronous write enable output.

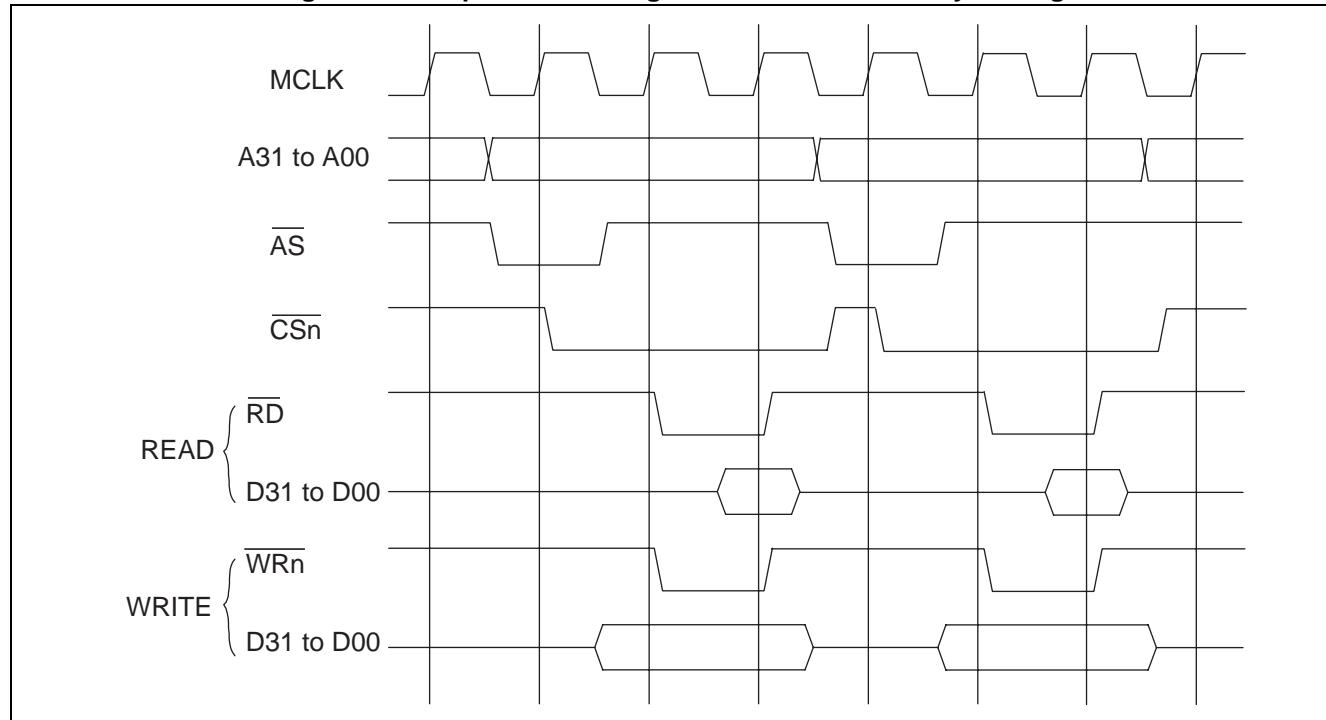
4.5.8 CSn Delay Setting

This section shows the operation timing for the CSn delay setting.

■ Operation Timing for the CSn Delay Setting

Figure 4.5-8 shows the operation timing for (TYP3 to TYP0=0000_B, AWR=000C_H).

Figure 4.5-8 Operation Timing Chart for the CSn Delay Setting



If the W02 bit is "1", assertion starts in the cycle following the cycle in which AS is asserted. For successive accesses, a negation period is inserted.

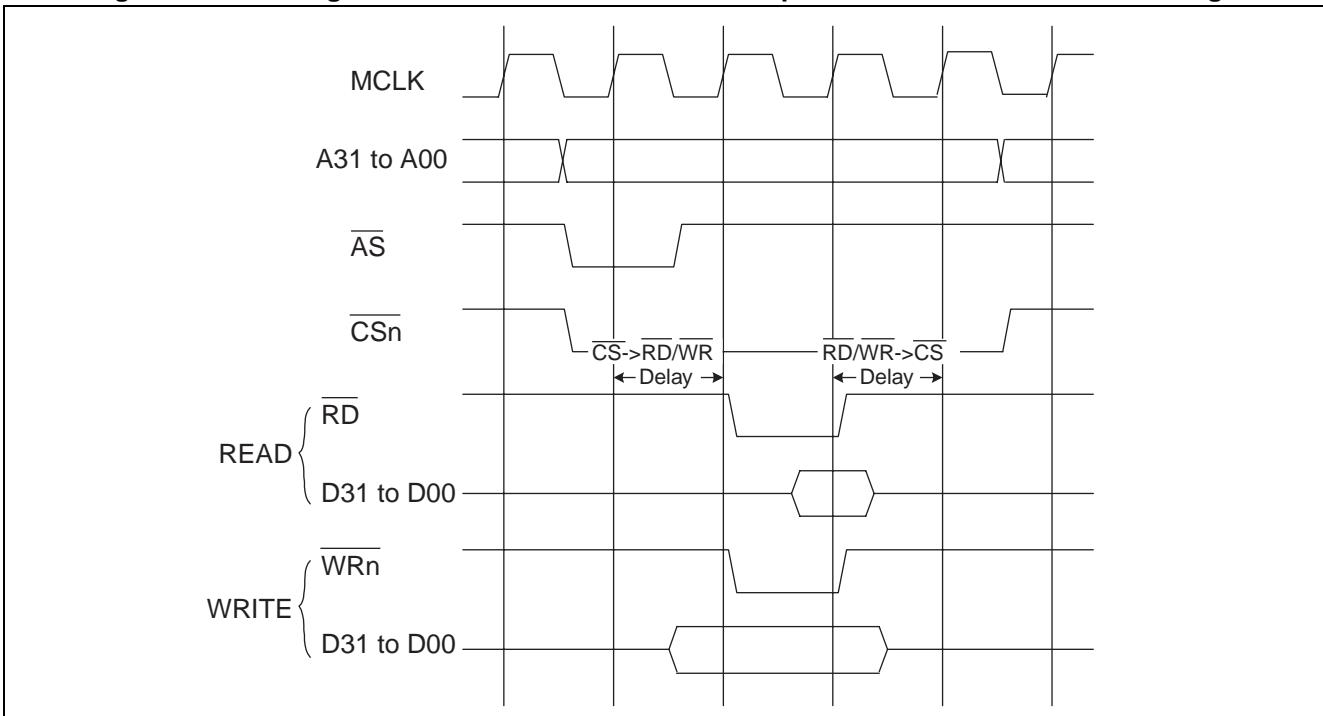
4.5.9 $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ Setup and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ Hold Setting

This section shows the operation timing for the $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ setup and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold settings.

■ Operation Timing for the $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ Setup and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ Hold Settings

Figure 4.5-9 shows the operation timing for (TYP3 to TYP0=0000_B AWR=000B_H).

Figure 4.5-9 Timing Chart for the $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ Setup and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ Hold Settings



- Setting "1" for the W01 bit of the AWR register enables the $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ setup delay to be set. Set this bit to extend the period between chip select assertion and read/write strobe.
- Setting "1" for the W00 bit of the AWR register enables the $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold delay to be set. Set this bit to extend the period between read/write strobe negation and chip select negation.
- The $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ setup delay (W01 bit) and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold delay (W00 bit) can be set independently.
- When making successive accesses within the same chip select area without negating the chip select, neither a $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ setup delay nor an $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold delay is inserted.
- If a setup cycle for determining the address or a hold cycle for determining the address is needed, set "1" for the address -> $\overline{\text{CSn}}$ delay setting (W02 bit of the AWR register).

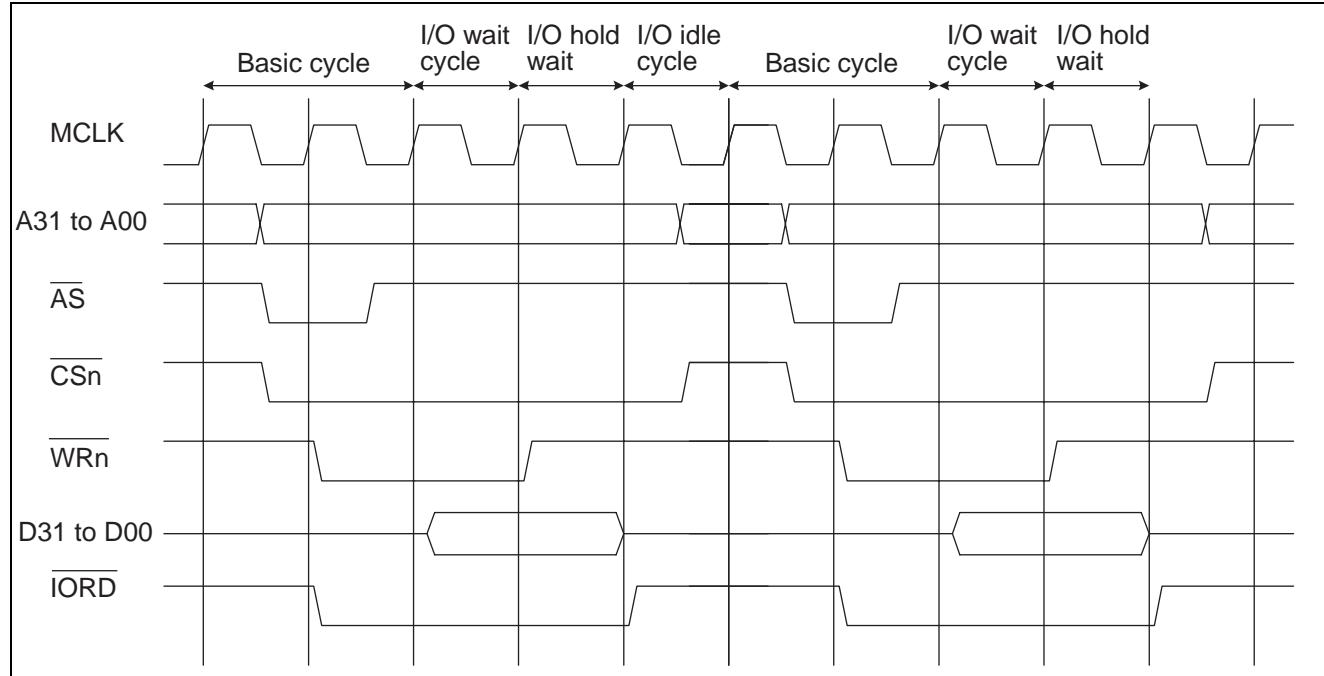
4.5.10 DMA Fly-By Transfer (I/O -> Memory)

This section shows the operation timing for DMA fly-by transfer (I/O -> memory).

■ Operation Timing for DMA Fly-By Transfer (I/O -> Memory)

Figure 4.5-10 shows the operation timing for ($TYP3 = TYP0 = 0000_B$, $AWR = 0008_H$, $IOWR = 51_H$). This timing chart shows a case in which a wait is not set on the memory side.

Figure 4.5-10 Timing Chart for DMA Fly-By Transfer (I/O -> Memory)



- Setting "1" for the HLD bit of the IOWR0 to IOWR3 registers enables the I/O read cycle to be extended by one cycle.
- Setting bits IW3 to IW0 of the IOWR0 to IOWR3 registers enables 0 to 15 wait cycles to be inserted.
- If wait is also set on the memory side (AWR15 to AWR12 is not "0"), the larger value is used as the wait cycle after comparison with the I/O wait (IW3 to IW0 bits).

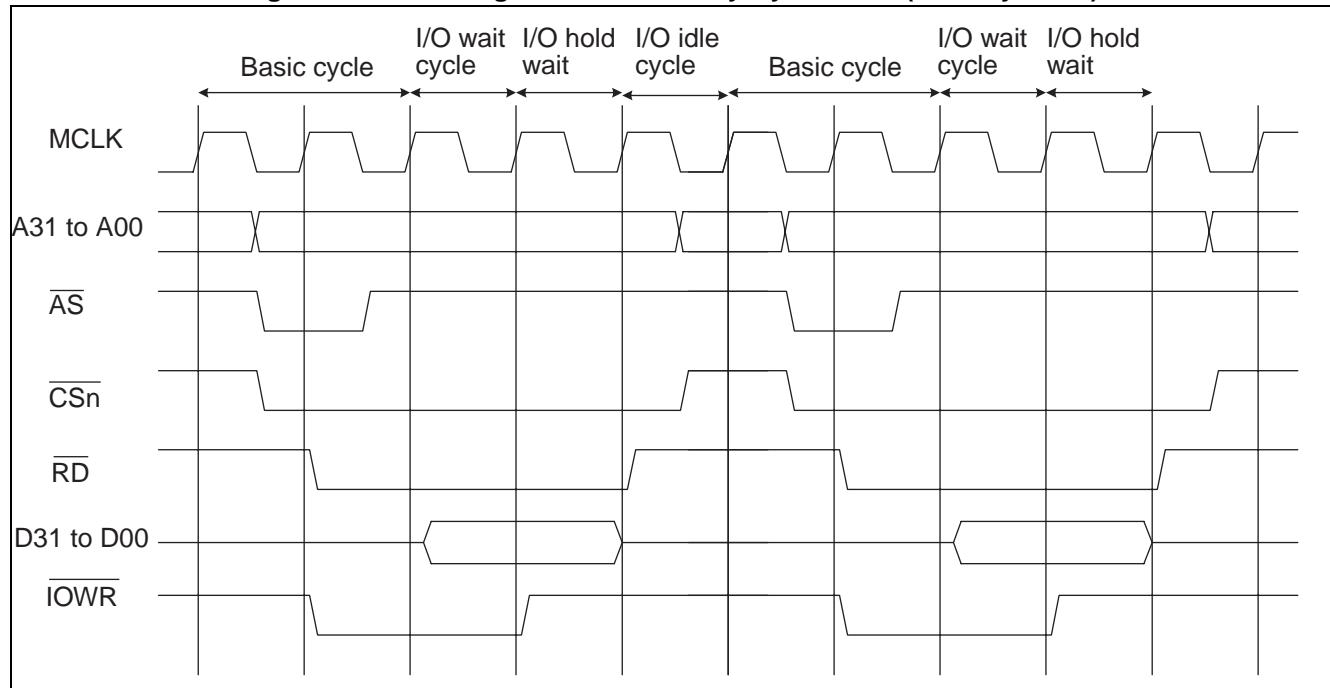
4.5.11 DMA Fly-By Transfer (Memory -> I/O)

This section shows the operation timing for DMA fly-by transfer (memory -> I/O).

■ Operation Timing for DMA Fly-By Transfer (Memory -> I/O)

Figure 4.5-11 shows the operation timing chart for ($TYP3 \rightarrow TYP0=0000_B$, $AWR=0008_H$, $IOWR=51_H$). This timing chart shows a case in which a wait is not set on the memory side.

Figure 4.5-11 Timing Chart for DMA Fly-By Transfer (Memory -> I/O)



- Setting "1" for the HLD bit of the IOWR0, IOWR1 registers enables the I/O read cycle to be extended by one cycle.
- Setting the WR1, WR0 bits of the IOWR0, IOWR1 registers enables 0 to 3 write recovery cycles to be inserted.
- If the write recovery cycle is set to "1" or more, a write recovery cycle is always inserted after write access.
- Setting bits IW3 to IW0 of the IOWR0, IOWR1 registers enables 0 to 15 wait cycles to be inserted.
- If wait is also set on the memory side ($AWR15 \rightarrow AWR12$ is not "0"), the larger value is used as the wait cycle after comparison with the I/O wait (IW3 to IW0 bits).

4.6 Burst Access Operation

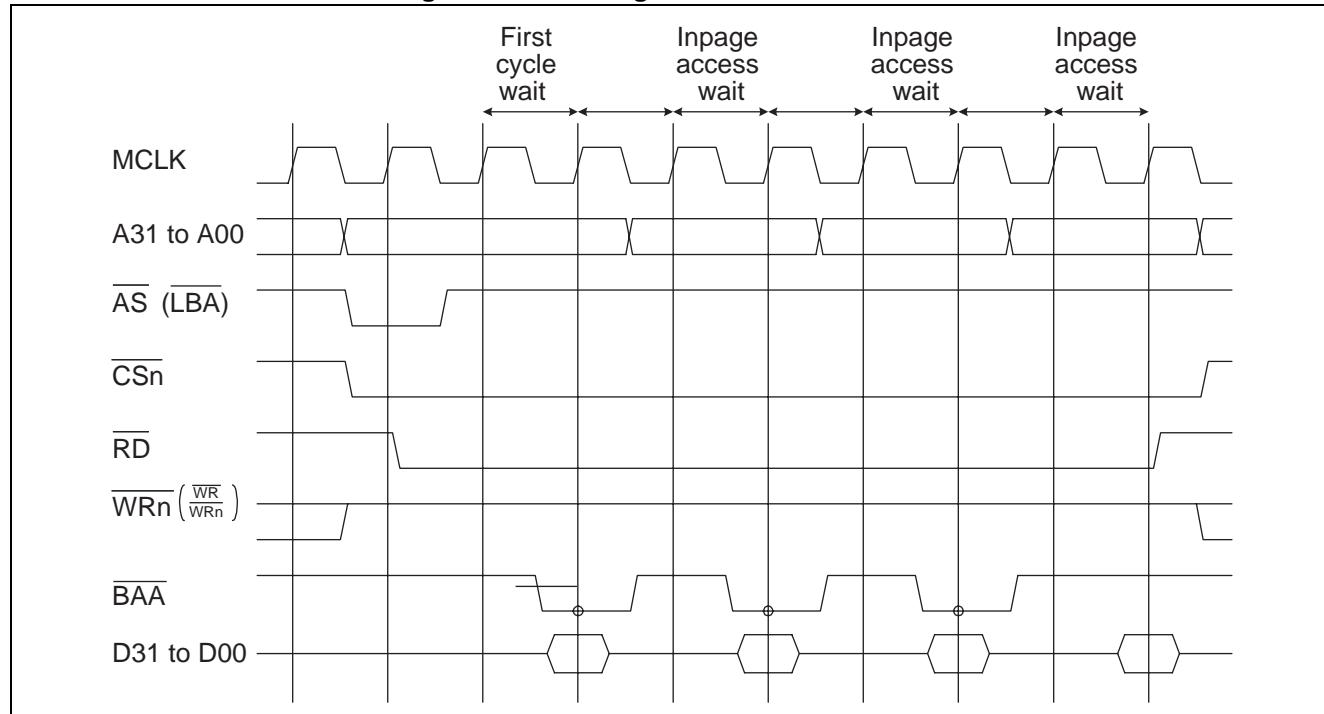
In the external bus interface, the operation that transfers successive data items in one access sequence is called burst access. The normal access cycle (that is, not burst access) is called single access. One access sequence starts with an assertion of \overline{AS} and \overline{CSn} and ends with negation of \overline{CSn} . Multiple data items two or more units of data of the unit set for the area.

This section explains burst access operation.

■ Burst Access Operation

Figure 4.6-1 shows the operation timing chart for (first wait cycle=1, inpage access wait cycle=1, TYP3 to TYP0=0000_B, AWR=1108_H).

Figure 4.6-1 Timing Chart for Burst Access



- In the external bus interface, the operation that transfers successive data items in one access sequence is called burst access. The normal access cycle (that is, not burst access) is called single access. One access sequence starts with an assertion of \overline{AS} and \overline{CSn} and ends with negation of \overline{CSn} . Multiple data items two or more units of data of the unit set for the area.
- In addition to more efficient use of access cycles when a sizable amount of data of asynchronous memory such as page mode ROM and burst flash memory is read, burst cycles can also be used for reading from normal asynchronous memory.

- The access sequence when burst cycles are used can be divided into the following two types:
 - First access cycle

The first access cycle is the start cycle for the burst access and operates in the same way as the normal single access cycle.

- Page access cycle

The page access cycle is a cycle following the first access cycle in which both \overline{CSn} and \overline{RD} (read strobe) are asserted. Wait cycles that are different from those set for a single cycle can be set. The page access cycle is repeated while access remains in the address boundary determined by the burst length setting. When access within the address boundary ends, burst access terminates and \overline{CSn} is negated.

- Setting of the W15 to W12 bits of the AWR register enable the first 0 to 15 wait cycles to be inserted. At this point, the minimum number of the first access cycles is the wait cycles + 2 cycles (three cycles in the timing chart shown in Figure 4.6-1).
- Setting of the W11 to W08 bits of the AWR register enables 0 to 15 page wait cycles to be inserted. At this point, the page access cycles can be obtained from the page wait cycles + 1 cycle (Two cycles in the timing chart shown in Figure 4.6-1).
- Setting of the BST bits of the ACR register enables the burst length to be set as "1", "2", "4", or "8". If the burst length is set to "1", single access mode is set and only the first cycle is repeated. However, if the data bus width is set to 32 bits (the BST bits of the ACR register are " 10_B "), set the burst length to "4" or less (A malfunction occurs if the burst length is set to "8").
- If burst access is enabled, burst access is used when prefetch access or transfer with a larger size than the specified data bus width is performed. For example, if word access to an area whose data bus width is set to 8 bits and burst length to "4" is performed, access of 4 bursts is performed once instead of repeating byte access four times.
- Since RDY input is ignored in areas for which burst access is set, do not set TYP3 to TYP0= $0xx1_B$.
- The \overline{LBA} and \overline{BAA} signals are designed for burst FLASH memory. \overline{LBA} indicates the start of access and \overline{BAA} indicates the address increment.
- A31 to A00 is updated after the wait cycles that were set during burst access.
- For fly-by transfer, burst access operation cannot be performed.

4.7 Address/data Multiplex Interface

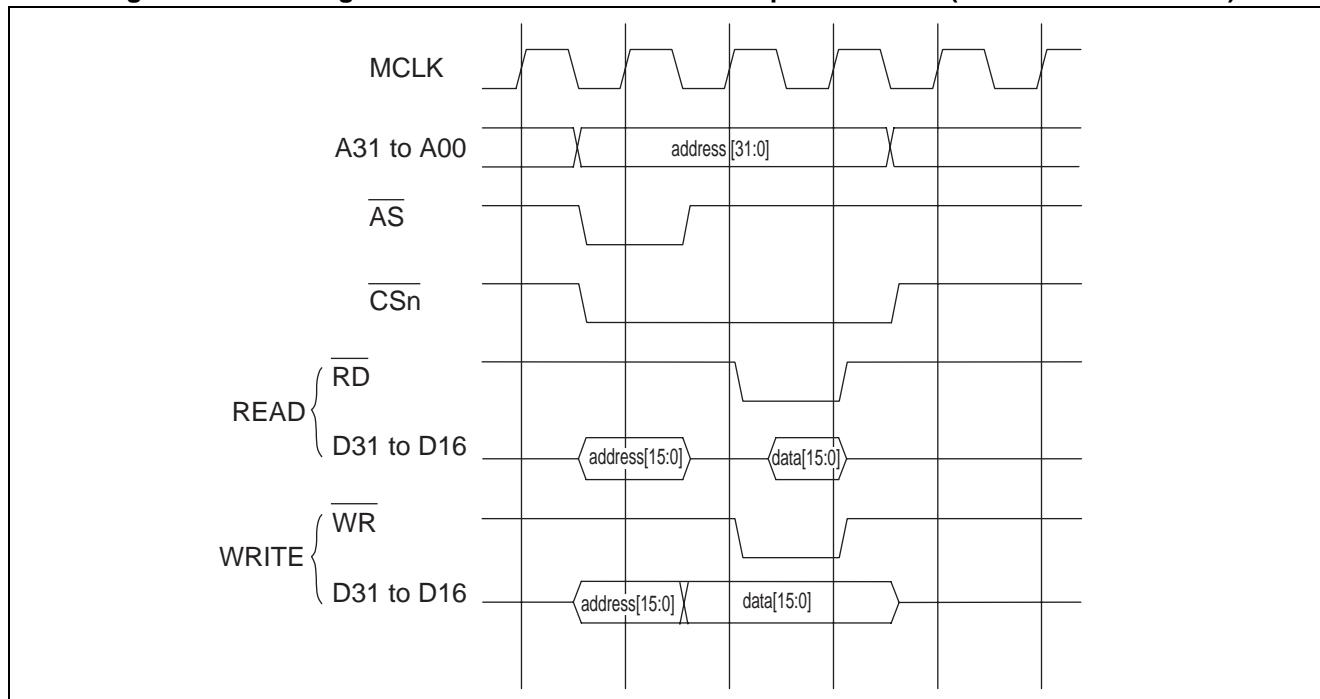
This section explains the following three cases of operation of the address/data multiplex interface:

- Without external wait
- With external wait
- CSn -> RD/WRn setup

■ Without External Wait

Figure 4.7-1 shows the operation timing chart for (TYP3 to TYP0=0100_B, AWR=0008_H).

Figure 4.7-1 Timing Chart for the Address/Data Multiplex Interface (without External Wait)



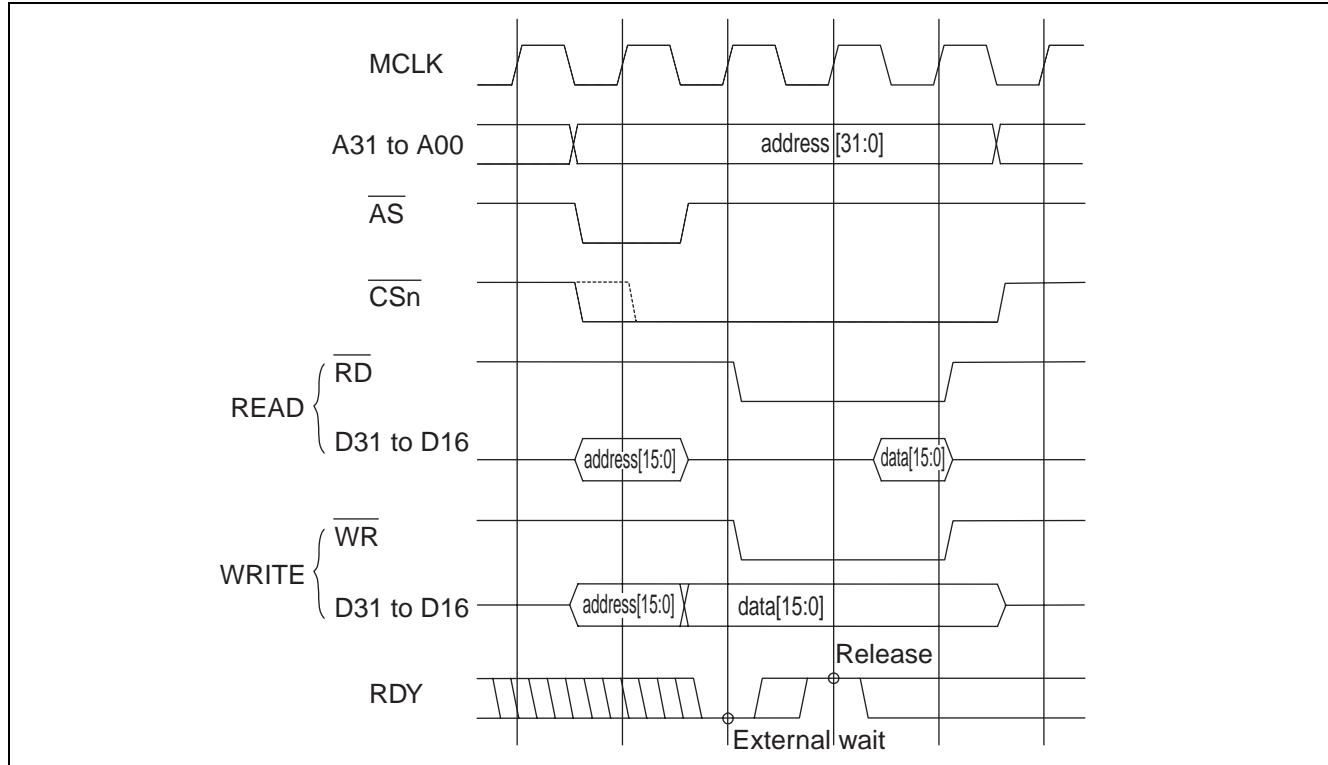
- Making a setting such as TYP3 to TYP0=01xx_B in the ACR register enables the address/data multiplex interface to be set.
- If the address/data multiplex interface is set, set 8 bits or 16 bits for the data bus width (DBW1, DBW0 bits). The 32 bit width is not supported.
- In the address/data multiplex interface, the total of 3 cycles of 2 address output cycles + 1 data cycle becomes the basic number of access cycles.
- In the address output cycles, AS is asserted as the output address latch signal. However, when CSn -> RD/WRn setup delay (AWR1) is set to "0", the multiplex address output cycle becomes only 1 cycle as shown in Figure 4.7-1, and address cannot be directly latched at the rising edge of AS. Therefore, the address is fetched at the rising edge of MCLK for the cycle which "L" is asserted to AS.

- As with a normal interface, the address indicating the start of access is output to A31 to A0 during the time division bus cycle. Use this address if you want to use an address more than 8/16 bits in the address/data multiplex interface.
- As with the normal interface, auto-wait (AWR15 to AWR12), read -> write idle cycle (AWR7, AWR6), write recovery (AWR5, AWR4), address -> $\overline{\text{CSn}}$ delay (AWR2), $\overline{\text{CSn}}$ -> $\overline{\text{RD}}/\overline{\text{WRn}}$ setup delay (AWR1), and $\overline{\text{RD}}/\overline{\text{WRn}}$ -> $\overline{\text{CSn}}$ hold delay (AWR0) can be set.
- In areas for which the address/data multiplex interface is set, set 1(DBW1, DBW0=00_B) as the burst length.

■ With External Wait

Figure 4.7-2 shows the operation timing chart for (TYP3 to TYP0=0101_B, AWR=1008_H).

Figure 4.7-2 Timing Chart for the Address/Data Multiplex Interface (with External Wait)



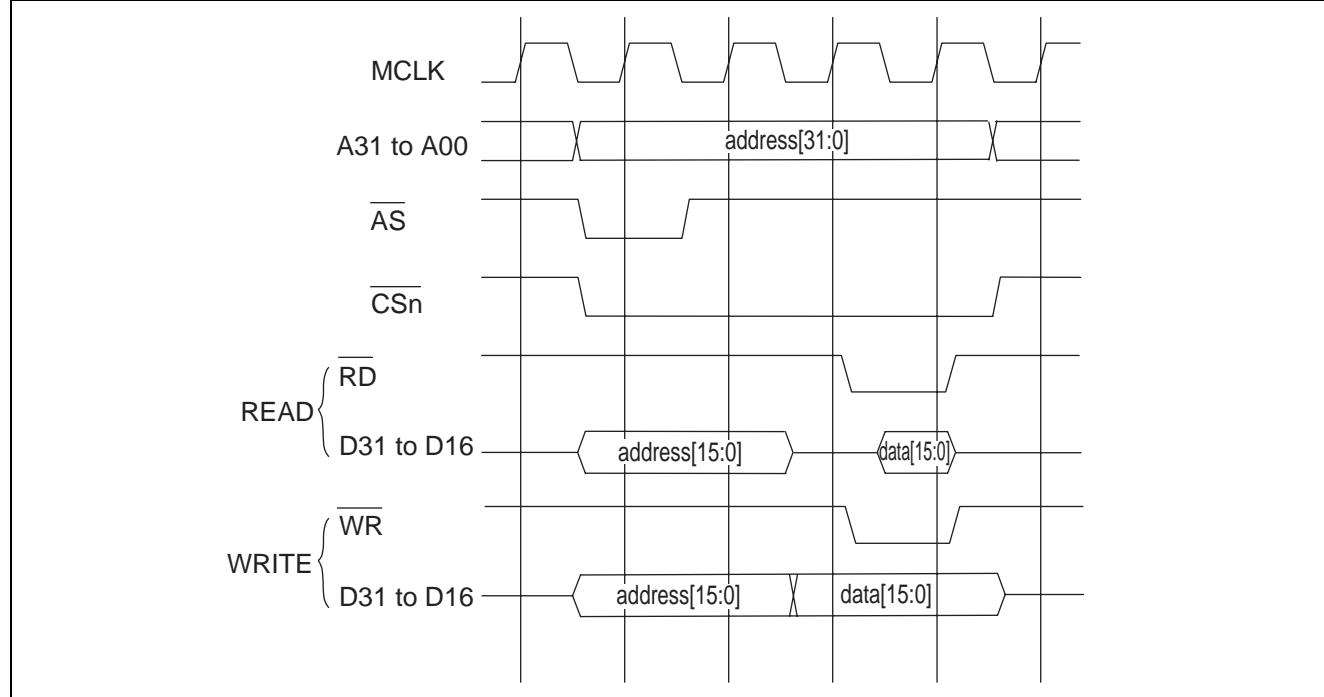
Making a setting such as TYP3 to TYP0=01x1_B in the ACR register enables RDY input in the address/data multiplex interface.

CHAPTER 4 EXTERNAL BUS INTERFACE

■ $\overline{\text{CSn}} \rightarrow \overline{\text{RD}}/\overline{\text{WRn}}$ Setup

Figure 4.7-3 shows the operation timing chart for ($\text{TYP3} \rightarrow \text{TYP0}=0101_B$, $\text{AWR}=100B_H$).

Figure 4.7-3 Timing Chart for the Address/Data Multiplex Interface ($\overline{\text{CSn}} \rightarrow \overline{\text{RD}}/\overline{\text{WRn}}$ Setup)



Setting "1" for the $\overline{\text{CSn}} \rightarrow \overline{\text{RD}}/\overline{\text{WRn}}$ setup delay (AWR1) enables the multiplex address output cycle to be extended by one cycle as shown in Figure 4.7-3, allowing the address to be latched directly to the rising edge of $\overline{\text{AS}}$. Use this setting if you want to use $\overline{\text{AS}}$ as an ALE (Address Latch Enable) strobe without using MCLK.

4.8 Prefetch Operation

This section explains the prefetch operation.

■ Prefetch Operation

The external bus interface controller contains a prefetch buffer consisting of 16 x 8 bits.

If the PSUS bit of the TCR register is "0" and read access to an area to which the PFEN bit of the ACR register is set to "1" occurs, the subsequent address is prefetched and then stored in the prefetch buffer.

If the stored address is accessed from the internal bus, the lookahead data in the prefetch buffer is returned without external access being performed. This can reduce the wait time for successive accesses to the external bus areas.

○ Basic conditions for starting external access using prefetch

External bus access using prefetch occurs when the following conditions are met:

- The PSUS bit of the TCR register is "0".
- Neither sleep mode nor stop mode is set.
- Read access by the external bus to a chip select area for which prefetch is enabled has been performed. DMA access and read access by a read-modify-write instruction, however, are excluded.
- No external bus access request (external bus area access to an area for which prefetch is not enabled or DMA transfer with an external bus area) other than the prefetch access has occurred.
- The part of the prefetch buffer for the next operation of capturing the prefetch access is completely empty.

While the above conditions are met, the prefetch access will continue. If external bus area access to an area for which prefetch is not enabled occurs after prefetch access, prefetch access to the area for which prefetch is enabled will continue as long as the prefetch buffer clear conditions are not met.

For an access that mixes multiple prefetch-enabled areas and multiple prefetch-disabled areas, the prefetch buffer always holds data of the prefetch-enabled area accessed last. Since, in this case, access to prefetch-disabled areas does not affect the prefetch buffer state at all, data in the prefetch buffer is not wasted even if prefetch-disabled data access and prefetch-enabled instruction fetch are mixed.

○ Optional clear for temporary stopping of a prefetch access

Setting "1" for the PSUS bit of the TCR register temporarily stops a prefetch. The prefetch can be restarted by setting the PSUS bit to "0". At this point, the contents of the buffer are retained if no error occurs or a buffer clear such as occurs when the PCLR bit is set does not occur.

Setting "1" for the PCLR bit of the TCR register completely clears the prefetch buffer. Clear the buffer by setting the PSUS bit when prefetch is interrupted.

Prefetch is temporarily stopped for the minimum unit (64 Kbytes) of the boundary=chip select area where the high-order 16 bits of an address change. If the boundary is crossed, first a buffer read error occurs and then prefetch starts in a new area.

Prefetch is temporarily stopped in SDRAM/FCRAM-connected areas, when the bank address is crossed. If a new bank address is accessed, first a buffer read error occurs, and then prefetch

starts at the new bank address. In SDRAM/FCRAM-connected areas, prefetch is also temporarily stopped, even if the page address is undated when write access to a prefetched area causes a page error or when access is made to another SDRAM/FCRAM area in which prefetch is not enabled.

○ Unit for one prefetch access operation

The unit for one prefetch access operation is determined by the DBW bits (bus width) and BST bits (burst length).

Prefetch access always occurs with the full size of the bus width specified by the DBW bits and access for the count of the burst length set by the BST bits in one access operation is performed. That is, if any value other than "00_B" is set for the BST bits, the prefetch always occurs in page mode/burst mode. Keep in mind whether ROM/RAM is conformable and enough access time is applicable. (Set an appropriate value bits W15 to W08 bits of the AWR register).

During burst access, successive accesses occur only within the address boundary that is determined by the burst length. Thus, if the boundary is crossed, for example, 4 bytes of free space are available in the buffer, these 4 bytes cannot be accessed in one operation (If the prefetch buffer starts at xxxx0E_H, 4 bytes of free space are available in the buffer, and two bursts are set even though the bus width is 16 bits, only 2 bytes, xxxx0E_H and xxxx0F_H, can be captured in the next prefetch access).

The following provides two examples:

- Area whose bus width is set to 16 bits and whose burst length is set to "2"

The amount of data read into the buffer in one prefetch operation is 4 bytes. In this case, prefetch access is delayed until 4 bytes of free space are available in the prefetch buffer.

- Area whose bus width is set to 8 bits and whose burst length is set to "8"

The amount of data read into the buffer in one prefetch operation is 8 bytes. In this case, prefetch access is delayed until 8 bytes of free space are available in the prefetch buffer.

○ Burst length setting and prefetch efficiency

If requests for external bus access, other than prefetch access, to or errors in the prefetch buffer occur during one operation of prefetch access as explained in the previous bullet, "Unit of one prefetch access operation", these access requests must wait until access to the prefetch buffer that is being executed is completed.

Thus, if the burst length is too long, the efficiency and reaction of bus access other than prefetch may be degraded. If, on the other hand, the burst length is set to "1", many read cycles may be wasted even if burst/page access memory is connected because single access is always performed.

If settings are made so that the amount of data read in one prefetch access operation is large, prefetch access can be started only after free space in the prefetch buffer for this amount is available. Thus, access to the prefetch buffer is infrequent, and the external bus tends to be idle. For example, if the bus width is set to 16 bits and the burst length is set to "8", the amount of data read into the buffer in one prefetch operation is 16 bytes. Thus, a new prefetch access can be started only after the prefetch buffer is completely empty.

Adjust the optimum burst length to suit use and the environment after taking the above into consideration. Generally, when connecting asynchronous memory to which burst/page access cannot be applied, it is best to set the burst length to "1" (single access). Conversely, when memory whose burst/page access cycle is short is connected, it is better to set the burst length to any value other than "1" (single access). In this case, it is best to make the setting so that 8 bytes (half of the buffer) are read in one read operation according to the bus width. However, the optimum condition varies with the frequency of external access and varies with the frequency divide-by rate setting of the external access clock.

○ **Reading from the prefetch buffer**

Data stored in the prefetch buffer is read in response to access from the internal bus if an address matches, and no external access is performed. In reading from the buffer, addresses can be hit (up to 16 bytes) if they are in the forward direction but not continuous, so that a second read from the external bus is avoided, if possible, even for a short forward branch.

If the address currently being accessed for prefetch matches during access from the internal bus, a wait signal is returned internally before data is captured after prefetch access is completed. In this case, no buffer error occurs.

If an address in the prefetch buffer matches when a read is performed for DMA transfer, data in the prefetch buffer is not used, and instead, external data is read by the external bus. In this case, a buffer error occurs. The prefetch is not continued and no prefetch access is performed until a new external access operation to a prefetch-enabled area occurs.

○ **Clearing/updating the prefetch buffer**

If either of the following conditions is met, the prefetch buffer is completely cleared:

- If "1" is written to the PCLR bit of the TCR register
- If a buffer read error occurs. A buffer read error is if any of the following events occurs:
 - When no address is found in the buffer that matches in an to read from a prefetch-enabled area. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer, but the prefetch access is started from the subsequent address to store addresses in the buffer.
 - In an access to read from a prefetch-enabled area with a read-modify-write instruction. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer. Also, no prefetch access is performed (This is because data is written to the next address).
 - In an access to read from a prefetch-enabled area for DMA transfer. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer. Also, no prefetch access is performed.
- If a buffer write hit occurs. A buffer write hit is as follows:
 - When the address of just 1 byte that matches is found in the buffer in an access to write to a prefetch-enabled area. In this case, the external bus is accessed again, but no prefetch access is performed before a new read access occurs.

Only part of the prefetch buffer is cleared when the following condition is met:

- If a buffer read hit occurs

In this case, only the part of the buffer before the hit address is cleared.

○ **Restrictions on prefetch-enabled areas**

If prefetch to a little endian area is enabled, be sure to access the area using word access. If data read into the prefetch buffer is accessed with any length other than word length, the correct endian conversion is not performed and thus the wrong data will be read. This is due to hardware restrictions related to the endian conversion mechanism.

4.9 SDRAM/FCRAM Interface Operation

This section describes the operations of the SDRAM/FCRAM interface.

■ SDRAM/FCRAM Interface

The CS6 and CS7 areas can be used as SDRAM/FCRAM space by setting the TYP3 to TYP0 bits in the area configuration register (ACR) to "100X_B".

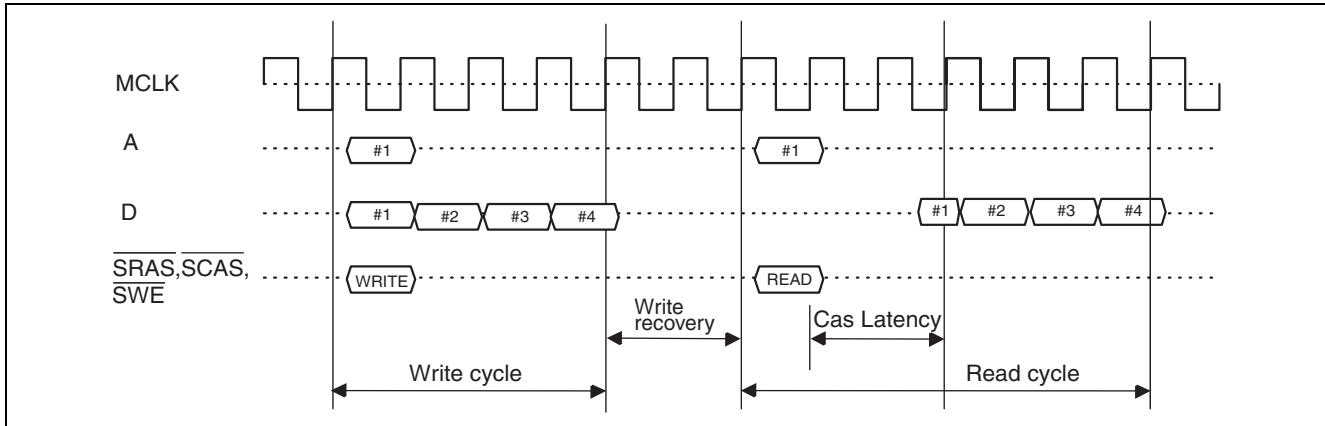
This section provides timing charts to describe the following operations of the SDRAM/FCRAM interface.

- Burst read/write (Settings: Page hit, CAS latency "2")
- Single read/write (Settings: Page hit, CAS latency "3", auto-precharge OFF)
- Single read (Settings: Page miss, CAS latency "3", auto-precharge OFF)
- Single read/write (Settings: CAS latency "1", TYP 1001_B, auto-precharge ON)
- Auto-refresh

■ Burst Read/Write Operation Timing

Figure 4.9-1 shows the operation timings assuming that page hits and CAS latency "2" are set.

Figure 4.9-1 Burst Read/Write Timing Chart

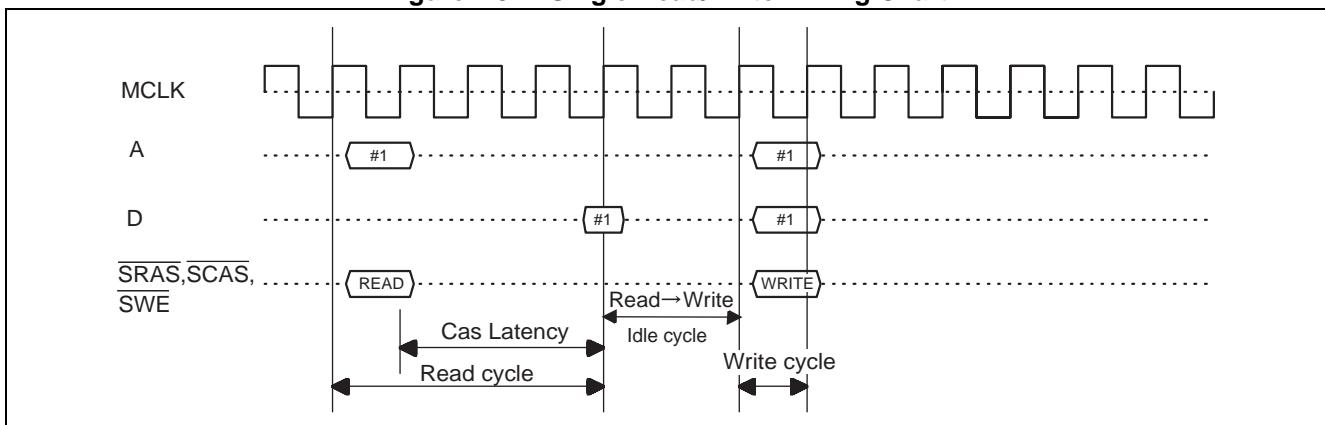


- All of the A13 to A00 pins may not be used depending on the SDRAM capacity. See Section "4.9.5 Memory Connection Example".
- The MCLK is a clock signal input to SDRAM. Signals such as addresses, data, and commands are input to SDRAM at the rise of the MCLK.
- Set the W05 and W04 bits in the area wait register (AWR) to the write recovery cycle according to the SDRAM/FCRAM standards.
- Set the W10 to W08 bits in the area wait register (AWR) to the CAS latency according to the SDRAM/FCRAM standards.
- Set the burst length using the BST bit in the area configuration register (ACR).

■ Single Read/Write Operation Timing

Figure 4.9-2 shows the operation timings assuming that page hits, CAS latency "3", and no auto-precharge are set.

Figure 4.9-2 Single Read/Write Timing Chart

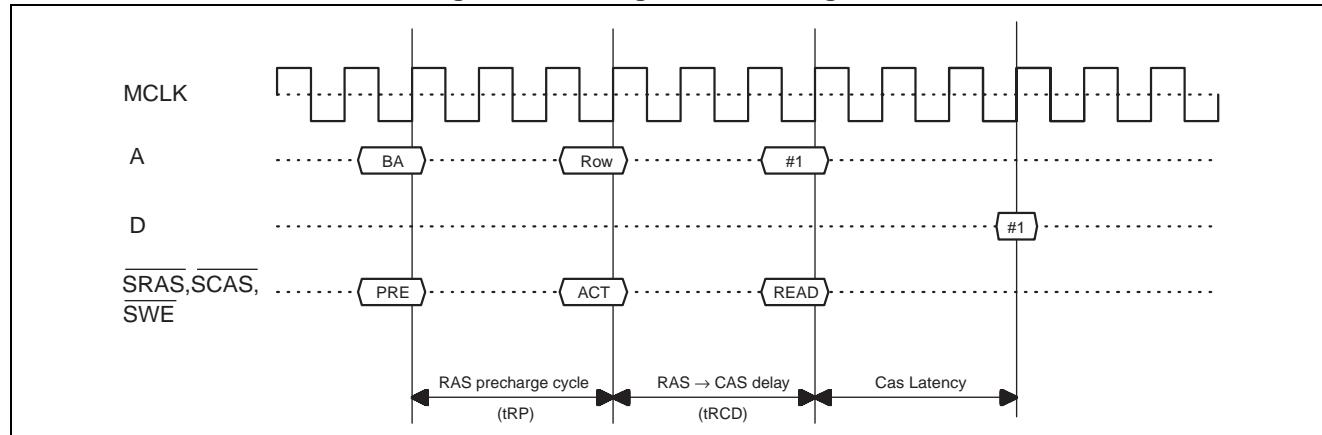


Set the W07 and W06 bits in the area wait register (AWR) to the read-to-write idle cycle according to the SDRAM/FCRAM standards.

■ Single Read Operation Timing

Figure 4.9-3 shows the operation timings assuming that page misses, CAS latency "3", and no auto-precharge are set.

Figure 4.9-3 Single Read Timing Chart

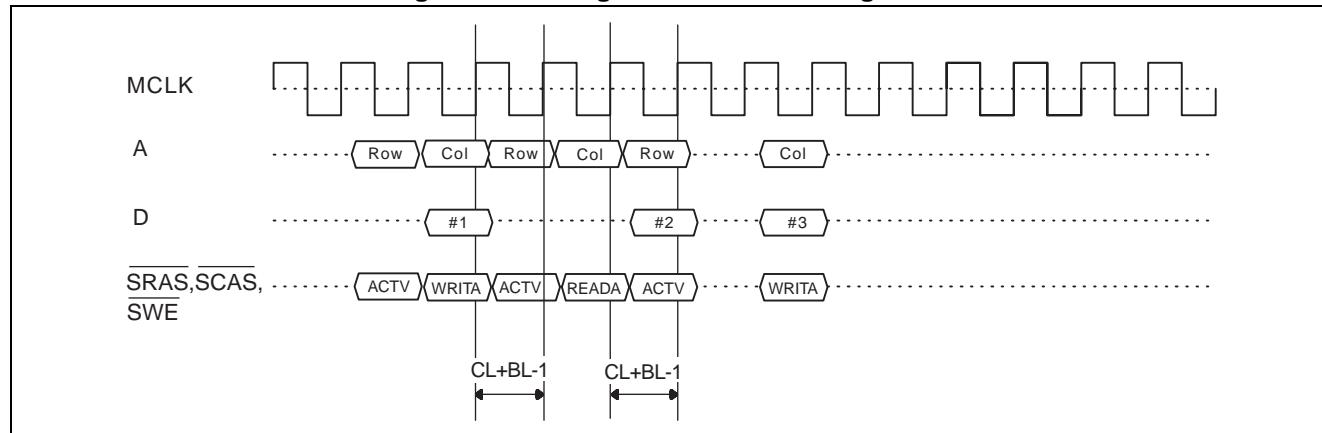


- When a page miss occurs, a read operation is performed after the PRE charge and ACTV commands are issued.
- Set the W01 and W00 bits in the area wait register (AWR) to the RAS precharge cycle (t_{RP}) according to the SDRAM/FCRAM standards.
- Set the W14 to W12 bits in the area wait register (AWR) to the RAS-to-CAS delay (t_{RCD}) according to the SDRAM/FCRAM standards.

■ Single Read/Write Operation Timing

Figure 4.9-4 shows the operation timings assuming that CAS latency "1", TYP = 1001_B, and auto-precharge are set.

Figure 4.9-4 Single Read/Write Timing Chart

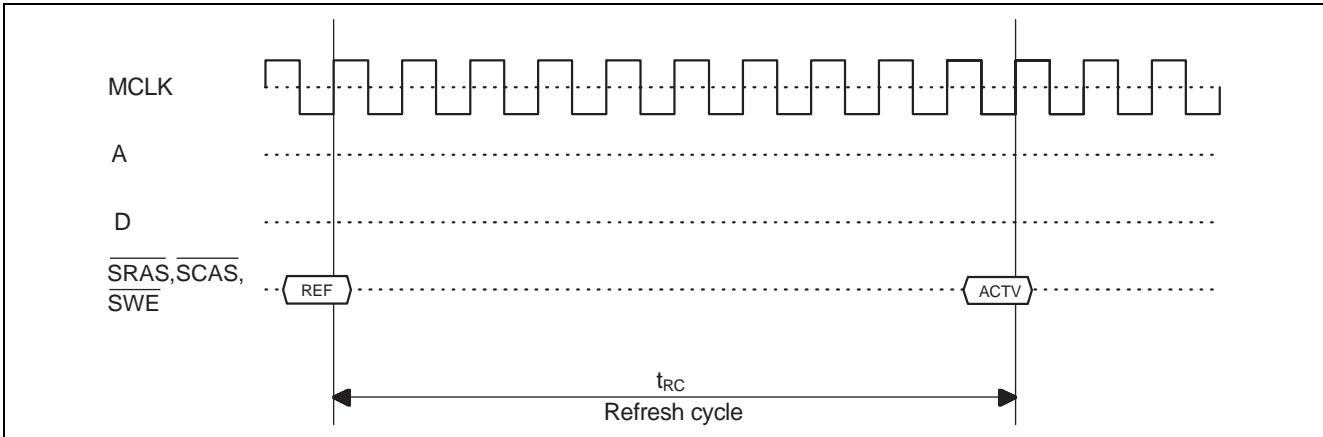


- Setting TYP to "1001_B" causes a read/write command with auto-precharge to be issued. Since the cycle from READA/WRITA issuance to ACTV issuance is fixed at CL + BL - 1, however, TYP can be set to "1001_B" only when FCRAM is connected.
- This timing is effective, for example, for recurring page misses as it eliminates the cycle for issuing the PRE command.

■ Auto-refresh Operation Timing

Figure 4.9-5 shows auto-refresh operation timings.

Figure 4.9-5 Auto-refresh Timing Chart



- The refresh command is issued every "refresh control register's (RCR's) RFINT5 to RFINT0 value x 32" cycles and access is restarted upon completion of each refresh.
- Set the TRC bit in the refresh control register (RCR) according to the SDRAM/FCRAM standards.
- Satisfy the maximum RAS active time as well.

4.9.1 Self Refresh

This section describes self-refreshing.

■ Self Refresh

Writing "1" to the SELF bit in the refresh control register (RCR) causes the SDRAM/FCRAM interface to initiate the self-refresh transition sequence.

After executing auto-refreshing the number of times set in the RFC2 to RFC0 bits, the SDRAM/FCRAM interface issues the SELF command to SDRAM/FCRAM to enter the self-refresh mode.

The device is released from the self-refresh mode either when "0" is written to the SELF bit or read/write access to SDRAM/FCRAM occurs.

The SDRAM/FCRAM interface issues the SELFX command to execute auto-refreshing the number of times set in the RFC2 to RFC0 bits upon detection of writing "0" to the SELF bit or access to SDRAM/FCRAM in the self-refresh mode.

Even when access to SDRAM/FCRAM by DMA transfer occurs after setting the self-refresh mode and putting the chip into sleep mode, the self-refresh mode is canceled.

○ Self-refresh mode transition procedure

- 1) Set SELF bit to "1".
- 2) Issue the REF command the number of times set in the RFC2 to RFC0 bits.
- 3) Issue SELF command

○ Self-refresh mode reset procedure

- 1) Set the SELF bit to "0" or access to SDRAM/FCRAM.
- 2) Issue SELFX command
- 3) Issue the REF command the number of times set in the RFC2 to RFC0 bits.
- 4) Transition to the normal access state

4.9.2 Power-on Sequence

This section describes the power-on sequence.

■ Power-on Sequence

Setting the PON bit in the refresh control register (RCR) to "1" initiates the power-on sequence.

Take the following steps to set the PON bit to "1" for transition to the power-on sequence.

- 1) Reserve the clock stabilization wait time specified in the SDRAM/FCRAM manual.
- 2) Set ACR, AWR, MCRA(B).
- 3) Set the CSER to enable the area to which SDRAM/FCRAM has been connected.
- 4) Set the PON bit to "1" while setting the RCR value.

Taking the above steps causes the SDRAM/FCRAM interface to execute the following power-on sequence.

- 5) Execute the PALL command.
- 6) Execute the REF command eight times.
- 7) The mode register is set according to the BST bit in the ACR, CL (CAS Latency) bit in the AWR, and the WBST bit in the MCRA.
- 8) Transition to the normal access state

4.9.3 Connecting SDRAM/FCRAM to Many Areas

This section shows the connecting SDRAM/FCRAM to many areas.

■ Connecting SDRAM/FCRAM to Many Areas

SDRAM/FCRAM can be set for $\overline{CS6}$ and $\overline{CS7}$ areas. When connecting SDRAM/FCRAM to 2 areas, connect the same type of modules.

More precisely, connect the modules common in the following register settings.

- Area configuration register (ACR): Set all of the DBW1 to DBW0, BST1 to BST0, and TYP3 to TYP0 bits to the same.
- Area wait register (AWR): Set all the bits to the same.
- Memory setting register (MCR): All the settings are the same as the registers are common.
- Refresh control register (RCR): All the settings are the same as the registers are common.

To enable the two areas at a time, execute the power-on sequence, auto-refresh, and self-refresh at the same time.

4.9.4 Address Multiplexing Format

This section describes the address multiplexing format.

■ Address Multiplexing Format

SDRAM/FCRAM access addresses correspond to row, bank, and column addresses differently depending on the settings of the ASZ3 to ASZ0, DBW1 and DBW0, PSZ2 to PSZ0, and BANK bits.

Addresses are arranged in the order of Column, BANK, and Row addresses, starting from the least significant bit.

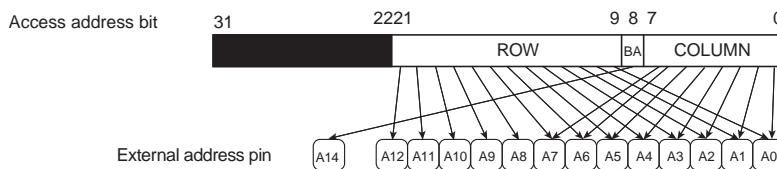
Set each bit as shown below.

- ASZ3 to ASZ0 bits: Set these bits to the total amount of SDRAM/FCRAM connected to the corresponding area. For using two modules in parallel, set the total amount. Affects the number of row addresses.
- DBW1 and DBW0 bits: Set these bits to the data bus width. (Set the bits to "16 bits" for connecting a pair of eight-bit modules in parallel.) Column addresses are shifted according to the data bus width setting. 8 bits: Do not shift. 16 bits: Shift one bit. 32 bits: Shift two bits.
- PSZ2 to PSZ0 bits: Set these bits to the number of column addresses used for SDRAM/FCRAM.
- BANK bit: Set this bit to the number of SDRAM/FCRAM bank addresses.

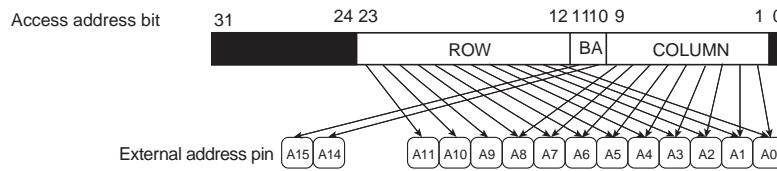
Figure 4.9-6 shows examples of combinations of access addresses and Row/BANK/Column addresses.

Figure 4.9-6 Examples of Combinations of access Addresses and Row/BANK/Column Addresses

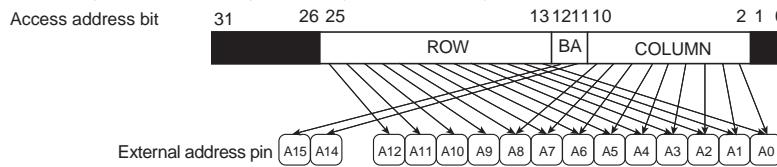
- 4M bytes (set ASZ to "0110B"), 8-bit bus width (set DBW to "00B")
256 column (set PSZ to "000B"), 2 banks (set BANK to "0")



- 16M bytes (set ASZ to "1000B"), 16-bit bus width (set DBW to "01B")
512 column (set PSZ to "001B"), 4 banks (set BANK to "1")



- 64M bytes (set ASZ to "1010B"), 32-bit bus width (set DBW to "10B")
512 column (set PSZ to "001B"), 4 banks (set BANK to "1")



4.9.5 Memory Connection Example

This section shows the memory connection example.

■ Memory Connection Example

The SDRAM/FCRAM interface is connected to SDRAM/FCRAM as shown in Table 4.9-1 in principle.

Table 4.9-1 SDRAM/FCRAM Interface to SDRAM/FCRAM Connection Table

SDRAM/ FCRAM interface pin	SDRAM/ FCRAM pin	Remarks
MCLK	CLK	
MCLKE	CKE	
SRAS (AS)	RAS	
SCAS (BAA)	CAS	
SWE (WR)	WE	
CS6 or CS7	CS	Only the CS6/CS7 area can be set as SDRAM/FCRAM space.
A00 to A09	A00 to A09	Addresses do not have to be shifted depending on the bus width.
A10/AP	A10/AP	A10 for row address output; otherwise AP
A11 to A13	A11 to A13	Connected to the address used for SDRAM/FCRAM.
A14	BA0	BA for 2 bank product
A15	BA1	The pin is not used for a two-bank module.
D31 to D00	DQ	The connection changes depending on the endian method and data bus width. For detailed connection, see Section "4.4 Endian and Bus Access".
DQMUU, DQMUL, DQMLU, DQMLL	DQM	The connection changes depending on the endian method and data bus width. For detailed connection, see Section "4.4 Endian and Bus Access".

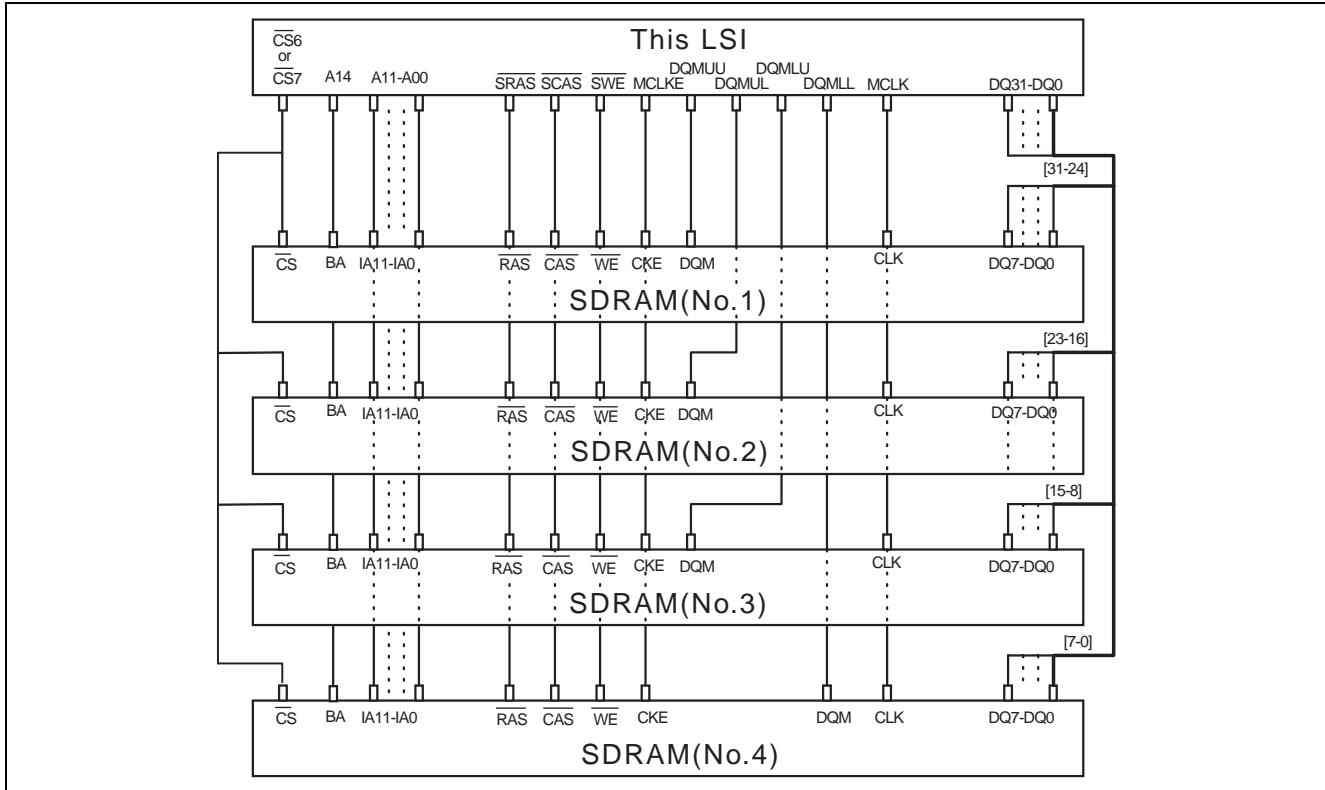
○ Using 8-bit SDRAM/FCRAM (Big endian)

Total data bus width of 32 bits: Use four SDRAM/FCRAM modules.

Total data bus width of 16 bits: Use two SDRAM/FCRAM modules.

Figure 4.9-7 shows how to use 64-Mbit SDRAM (one bank address and 12 row addresses).

Figure 4.9-7 Using 64-Mbit SDRAM



When SDRAM modules are used with a total data width of 16 bits, SDRAMs No. 3 and No. 4 are not required and DQ15 to DQ0 must be left open.

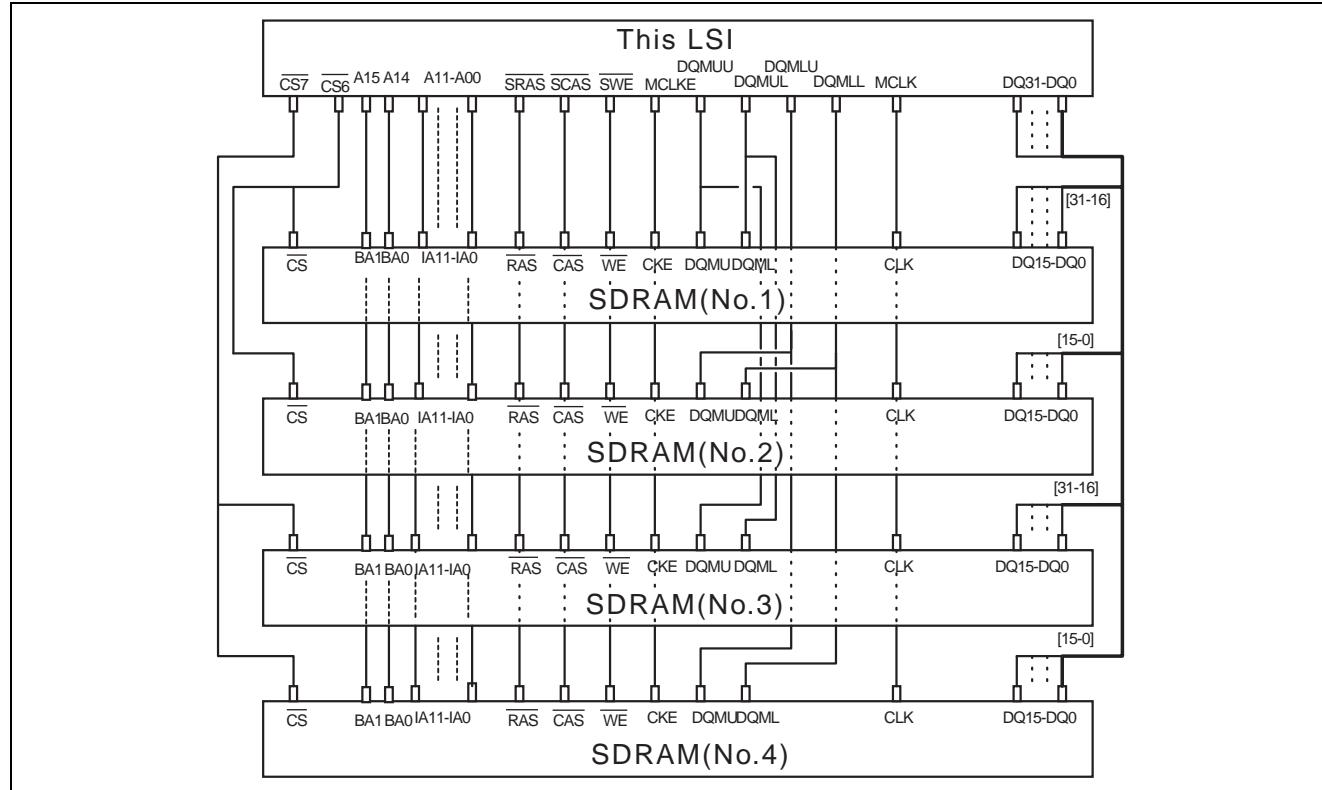
○ Using 16-bit SDRAM/FCRAM

Total data width of 32 bits: Use two or four SDRAM modules.

Total data width of 16 bits: Use one or two SDRAM modules.

Figure 4.9-8 shows how to use 64-Mbit SDRAM (two bank addresses and 12 row addresses).

Figure 4.9-8 Using 64-Mbit SDRAM



When using one SDRAM module with a data width of 16 bits, SDRAMs No. 2, No. 3, and No. 4 are not required and DQ15 to DQ0 must be left open.

When two SDRAM modules are used with a data width of 16 bits, SDRAMs No. 2 and No. 4 are not required.

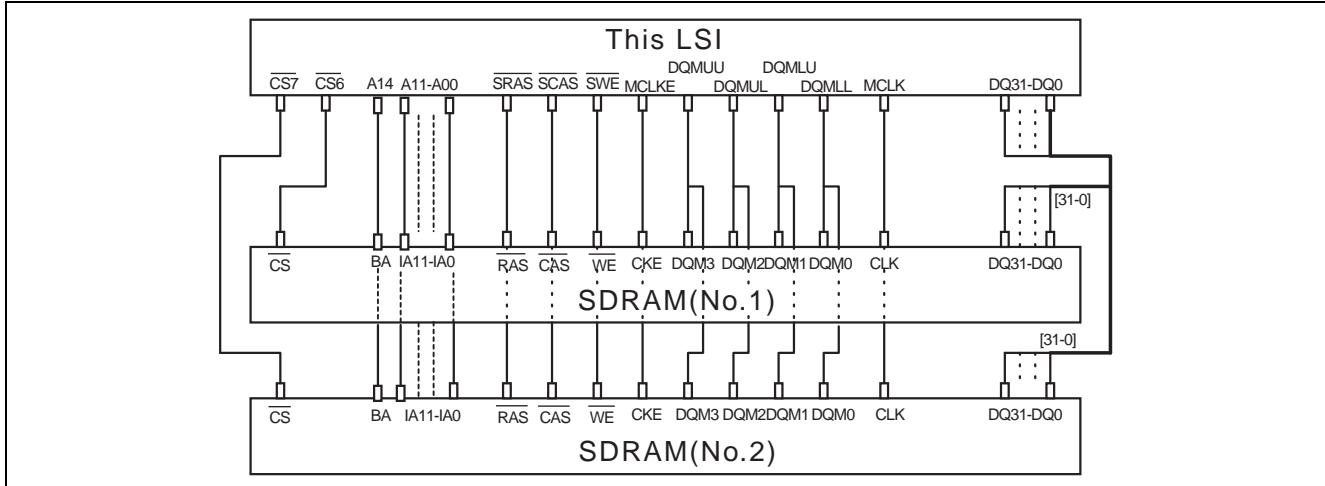
When two SDRAM modules are used with a data width of 32 bits, SDRAMs No. 3 and No. 4 are not required.

○ Using 32-bit SDRAM

When the data width is 32 bits: Use one or two SDRAM modules.

Figure 4.9-9 shows 64-Mbit SDRAM (one bank address and 12 row addresses).

Figure 4.9-9 Using 64-Mbit



SDRAM No. 2 is not required when the device is used with only one SDRAM module.

4.10 DMA Access Operation

This section explains DMA access operation.

■ DMA Access Operation

This section explains the following nine DMA operations:

- DMA fly-by transfer (I/O -> memory)
- DMA fly-by transfer (memory -> I/O)
- DMA fly-by transfer (I/O -> SDRAM/FCRAM)
- DMA fly-by transfer (SDRAM/FCRAM -> I/O)
- 2-cycle transfer (internal RAM -> external I/O, RAM)
- 2-cycle transfer (external -> I/O)
- 2-cycle transfer (I/O -> external)
- 2-cycle transfer (I/O -> SDRAM/FCRAM)
- 2-cycle transfer (SDRAM/FCRAM -> I/O)

4.10.1 DMA Fly-By Transfer (I/O -> Memory)

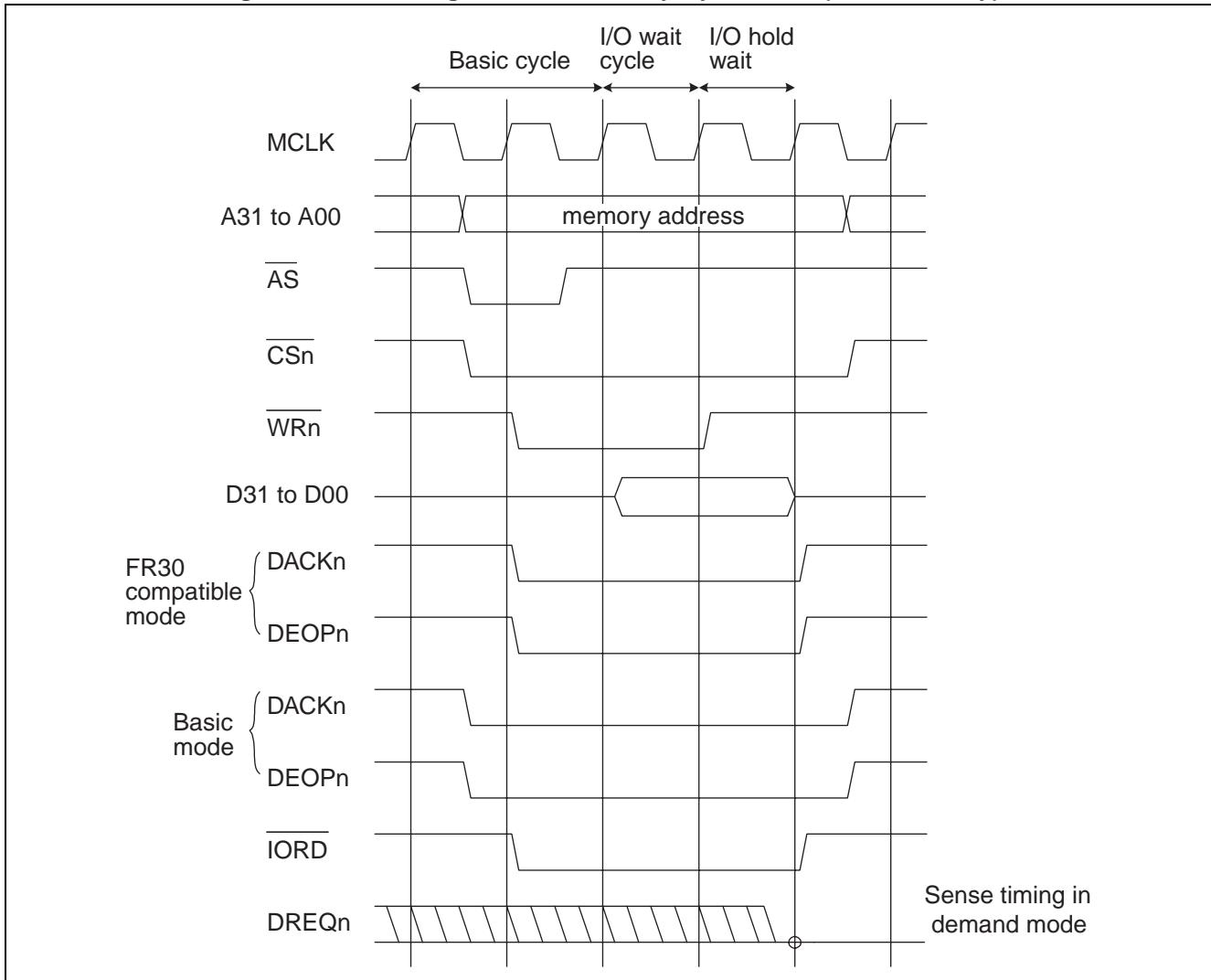
This section explains DMA fly-by transfer (I/O -> memory).

■ DMA Fly-By Transfer (I/O -> Memory)

Figure 4.10-1 shows the operation timing chart for (TYP3 to TYP0=0000_B, AWR=0008_H, IOWR=41_H).

Figure 4.10-1 shows a case when a wait is not set on the memory side.

Figure 4.10-1 Timing Chart for DMA Fly-By Transfer (I/O -> Memory)



- Setting "1" for the W01 bit of the AWR register enables the **CSn -> RD/WRn** setup delay to be set. Set this bit to extend the period between assertion of chip select and the read/write strobe.
- Setting "1" for the W00 bit of the AWR register enables the **RD/WRn -> CSn** hold delay to be set. Set this bit to extend the period between negation of the read/write strobe and negation of chip select.

CHAPTER 4 EXTERNAL BUS INTERFACE

- The $\overline{\text{CSn}}$ -> $\overline{\text{RD/WRn}}$ setup delay (W01 bit) and $\overline{\text{RD/WRn}}$ -> $\overline{\text{CSn}}$ hold delay (W00 bit) can be set independently.
- When successive accesses are made within the same chip select area without negating the chip select, neither $\overline{\text{CSn}}$ -> $\overline{\text{RD/WRn}}$ setup delay nor $\overline{\text{RD/WRn}}$ -> $\overline{\text{CSn}}$ hold delay is inserted.
- If a setup cycle for determining the address or a hold cycle for determining the address is needed, set "1" for the address -> $\overline{\text{CSn}}$ delay setting (W02 bit of the AWR register).

Reference:

For I/O on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated. For memory on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated. The I/O hold wait cycle does not affect the write strobe. However, the address and CS signal are retained until the fly-by bus access cycles end.

4.10.2 DMA Fly-By Transfer (Memory -> I/O)

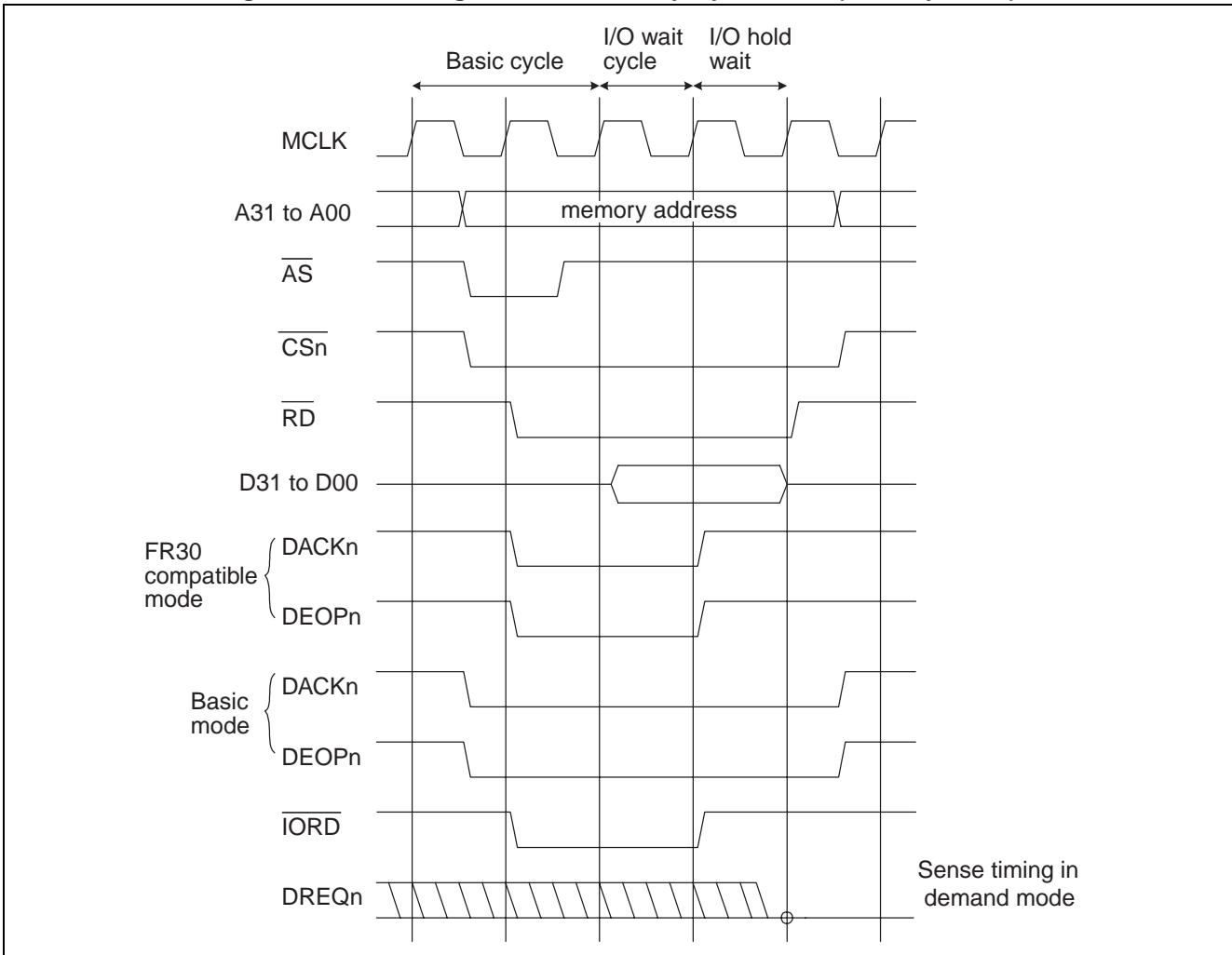
This section explains DMA fly-by transfer (memory -> I/O).

■ DMA Fly-By Transfer (Memory -> I/O)

Figure 4.10-2 shows the operation timing chart for (TYP3 to TYP0=0000_H, AWR=0008_H, IOWR=41_H).

Figure 4.10-2 shows a case in which a wait is not set on the memory side.

Figure 4.10-2 Timing Chart for DMA Fly-By Transfer (Memory -> I/O)



- Setting "1" for the HLD bit of the IOWR0 to IOWR3 registers extends the I/O read cycle by one cycle.
- Setting bits WR1, WR0 bits of the IOWR0 to IOWR3 registers enables 0 to 3 write recovery cycles to be inserted.
- If the write recovery cycle is set to "1" or more, a write recovery cycle is always inserted after write access.
- Setting bits IW3 to IW0 of the IOWR0 to IOWR3 registers enables 0 to 15 wait cycles to be inserted.

CHAPTER 4 EXTERNAL BUS INTERFACE

- If wait is also set on the memory side (AWR15 to AWR12 is not "0"), the larger value is used as the wait cycle after comparison with the I/O wait (IW3 to IW0 bits).
-

Reference:

For memory on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated. For I/O on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated. The I/O hold wait cycle does not affect the write strobe. However, the address and CS signal are retained until the fly-by bus access cycles end. Always perform fly-by transfers using the same data bus width.

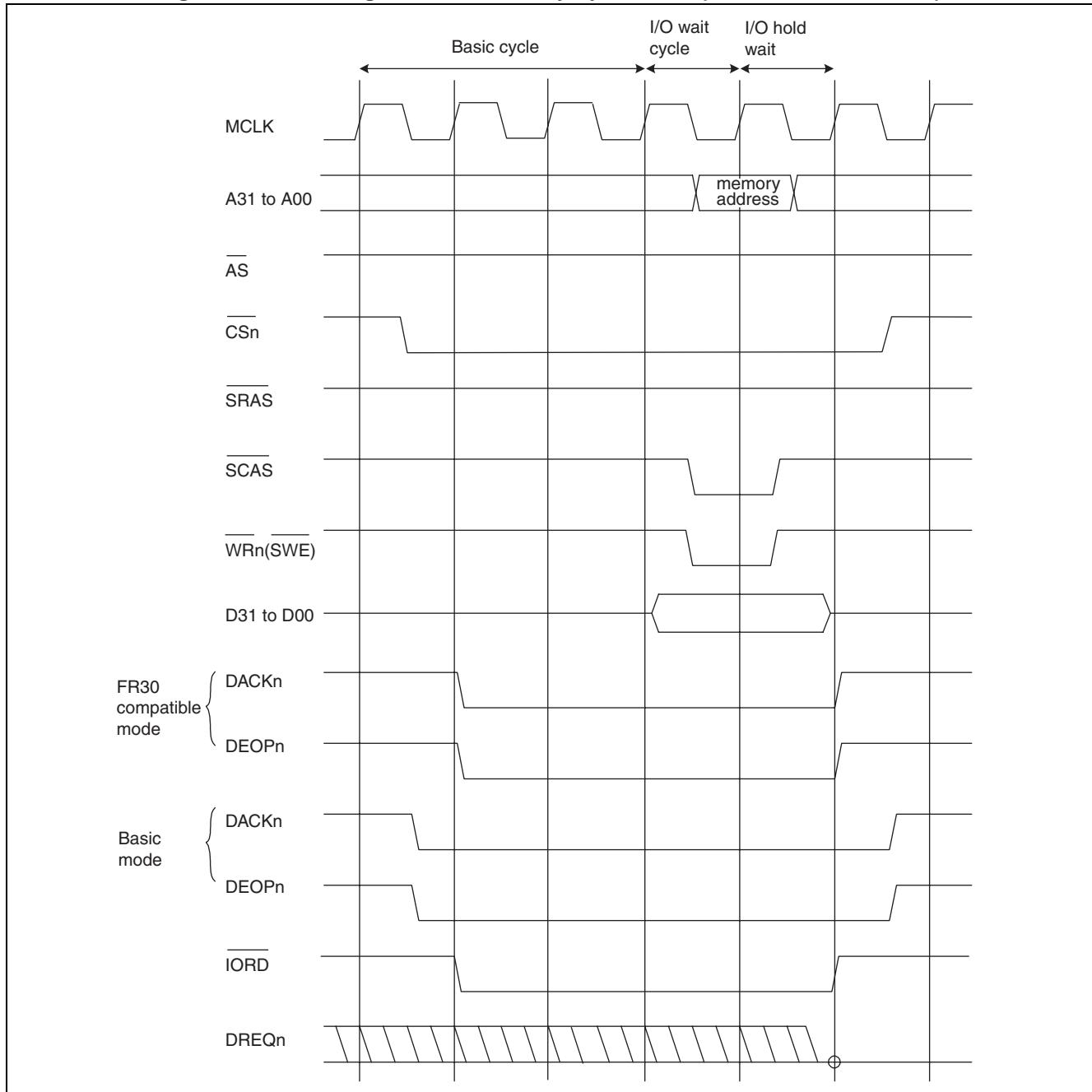
4.10.3 DMA Fly-By Transfer (I/O -> SDRAM/FCRAM)

This section describes the operation of DMA fly-by transfer (I/O device to SDRAM/FCRAM).

■ DMA Fly-By Transfer (I/O -> SDRAM/FCRAM)

Figure 4.10-3 shows an operation timing chart assuming TYP3 to TYP0 set to 1000_B , AWR set to 0051_H , and IOWR set to 41_H .

Figure 4.10-3 Timing Chart for DMA Fly-by Transfer (I/O to SDRAM/FCRAM)



CHAPTER 4 EXTERNAL BUS INTERFACE

- For the I/O device on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated.
- For SDRAM/FCRAM on the receiving side, a WRIT command is issued at the timing that allows writing after the I/O wait cycle. The I/O wait cycle may be longer depending on the SDRAM/FCRAM bank active state and SDRAM/FCRAM wait setting.
- The I/O hold wait cycle does not affect the write strobe. Note, however, that the CS signal is retained until the fly-by bus access cycles end.
- For fly-by transfer from an I/O device to SDRAM/FCRAM, be sure to set the HLD bit in the DMAC I/O wait register (IOWR) to "1" to enable the I/O hold wait cycle.
- Fly-by transfer must always be performed between data buses having the same bus width.

4.10.4 DMA Fly-By Transfer (SDRAM/FCRAM -> I/O)

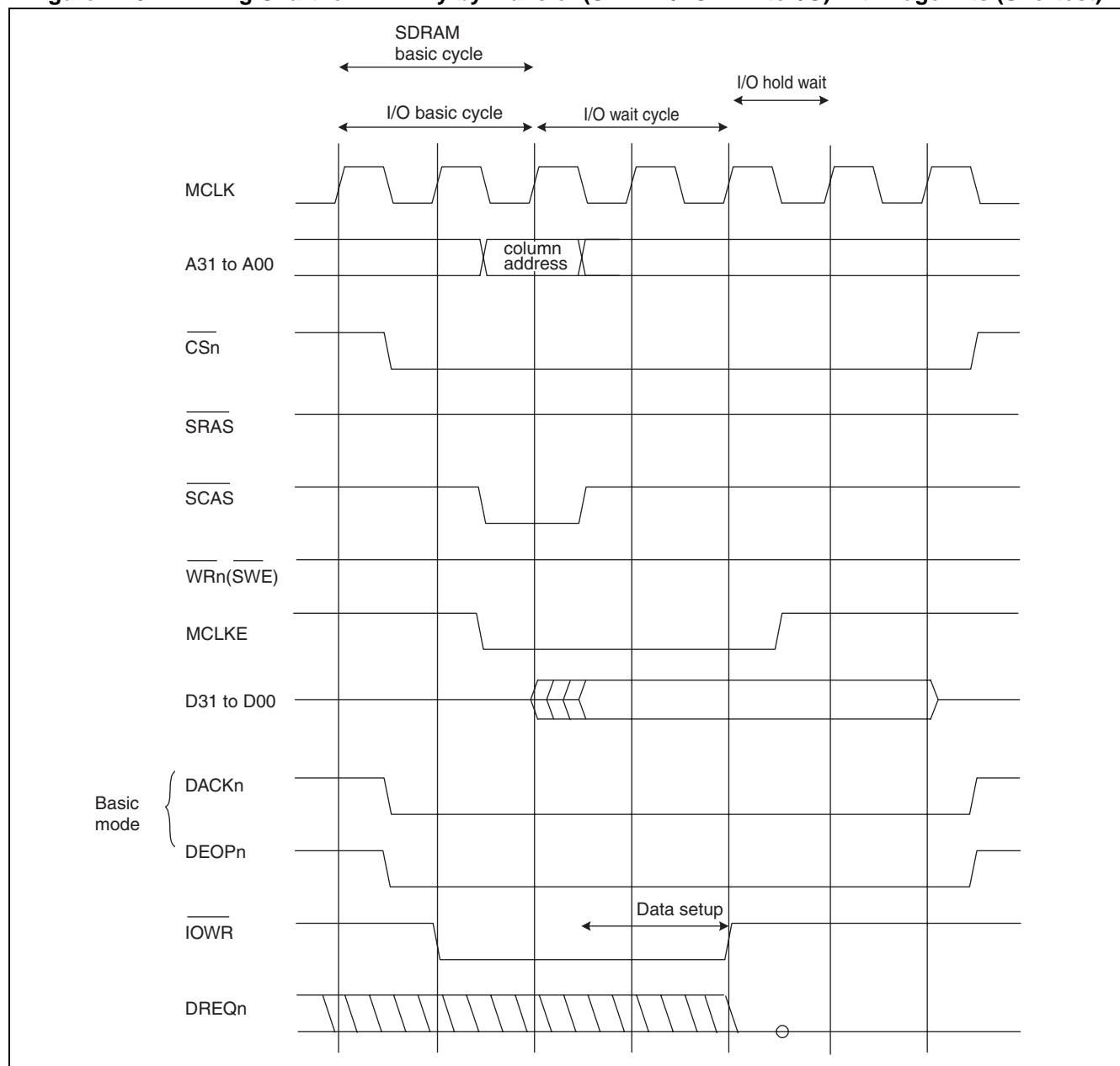
This section describes the operation of DMA fly-by transfer (SDRAM/FCRAM device to I/O).

■ DMA Fly-By Transfer (SDRAM/FCRAM -> I/O)

Figure 4.10-4 shows an operation timing chart assuming TYP3 to TYP0 set to 1000_B , AWR set to 0051_H , and IOWR set to 42_H .

○ At SDRAM page hit (Shortest)

Figure 4.10-4 Timing Chart for DMA Fly-by Transfer (SDRAM/FCRAM to I/O) with Page Hits (Shortest)

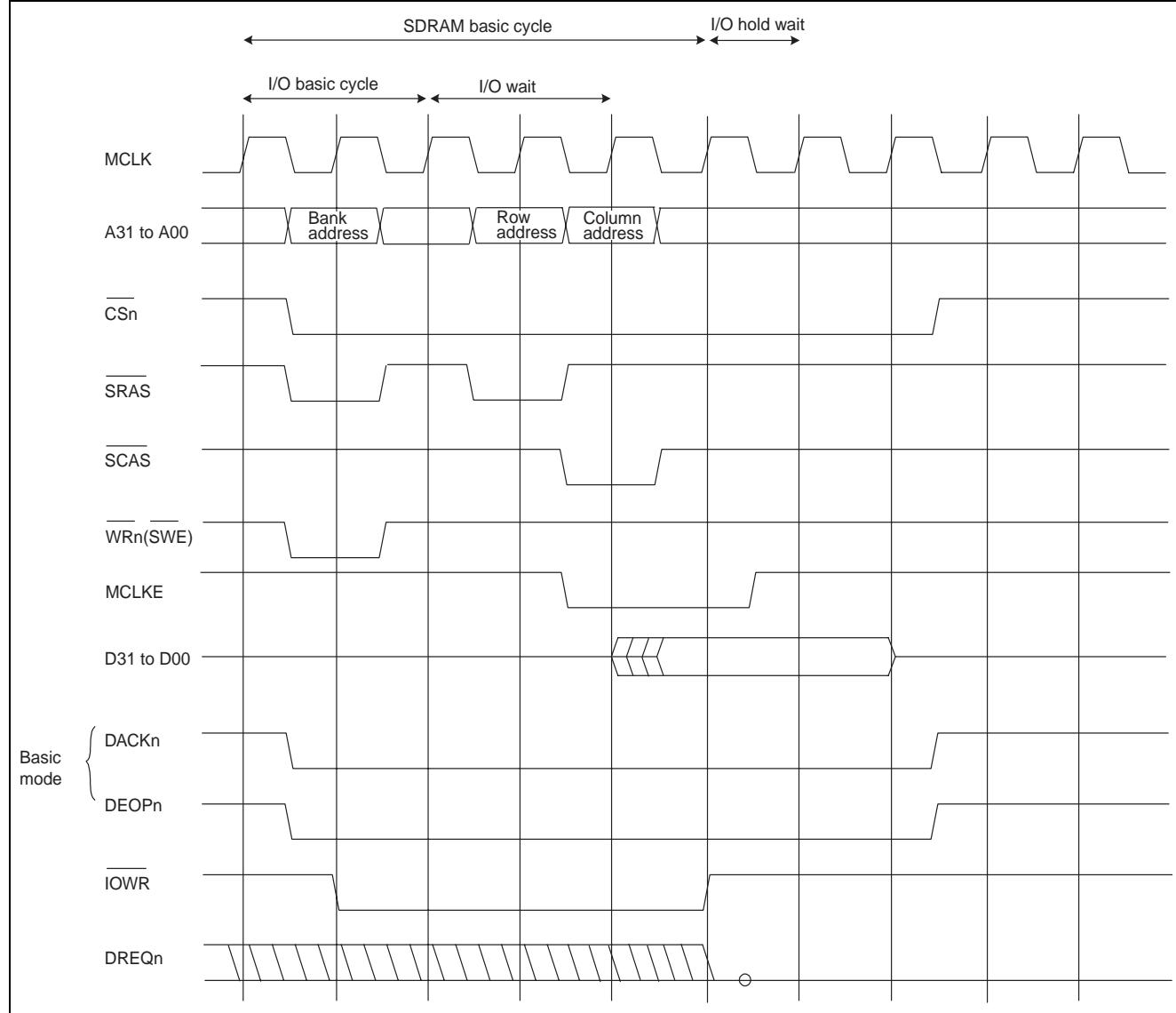


If SDRAM access is shorter than I/O access, the SDRAM access is extended by the I/O access (base access plus I/O wait).

Figure 4.10-5 shows an operation timing chart assuming TYP3 to TYP0 set to 1000_B, AWR set to 0051_H, and IOWR set to 42_H.

○ At SDRAM page misses

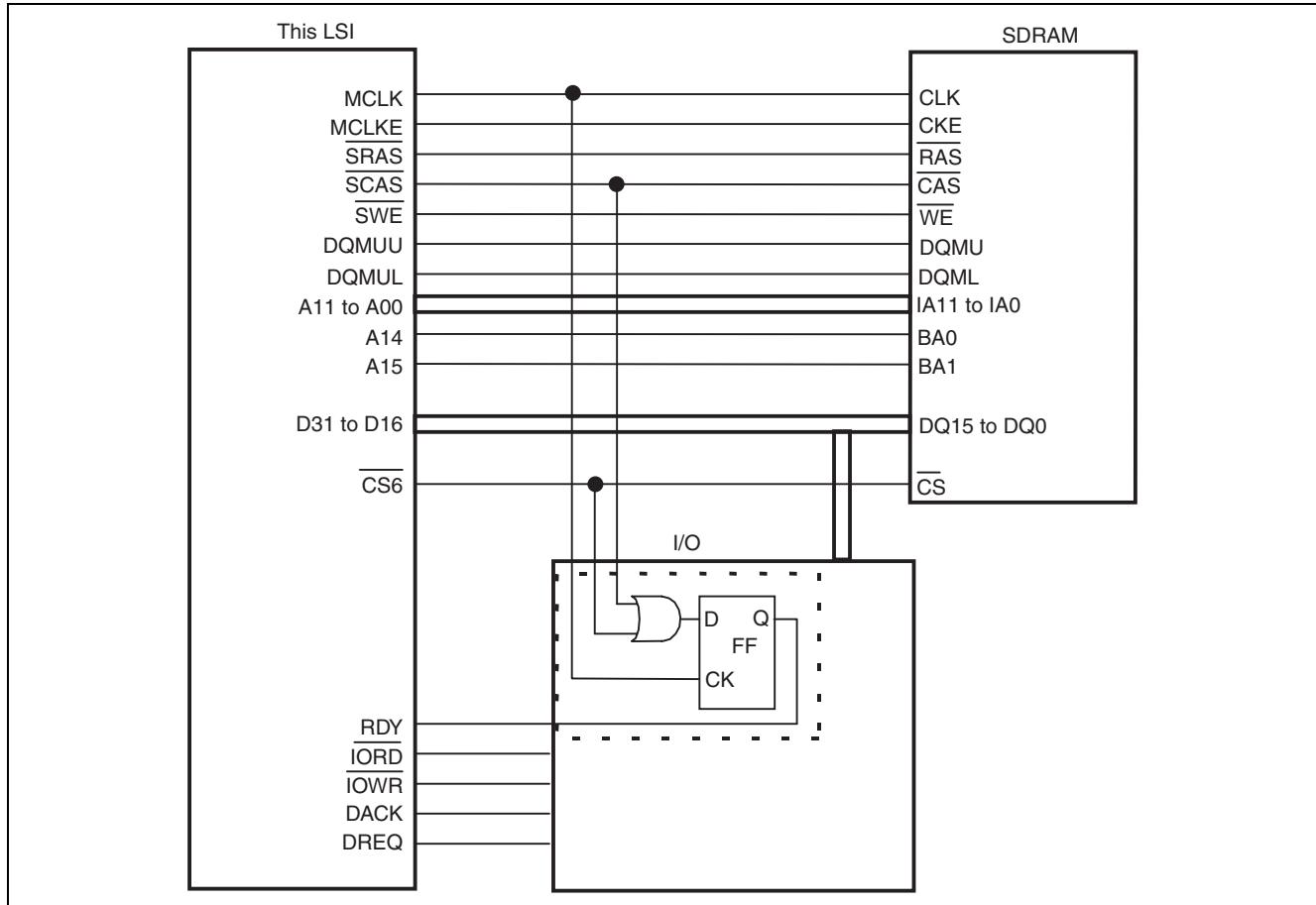
Figure 4.10-5 Timing Chart for DMA Fly-by Transfer (SDRAM/FCRAM to I/O) with Page Misses



- If SDRAM access is extended, for example, by precharging when a page miss occurs in reference to SDRAM, the SDRAM access exceeds the set I/O access, so that the I/O access is extended to be longer than the SDRAM access. When the I/O device requires data setup, therefore, the I/O wait cycle must be set such that I/O access is longer than the maximum SDRAM access cycle. For the above settings, set the number of I/O wait cycles to at least "4".
- For SDRAM/FCRAM on the data output side, a READ command is issued at the timing that satisfies the I/O wait cycle. If the I/O hold cycle has been set, then, a DESL command is issued to insert the I/O hold cycle in the cycle immediately followed by the end of the bus access cycles.

- For the I/O device on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated. The I/O hold wait cycle does not affect the write strobe. However, the CS signal is retained until the fly-by bus access cycles end.
- Fly-by transfer must always be performed between data buses having the same bus width.
- When the I/O wait cycle is used to reserve data setup time, the I/O wait value must be set according to the page miss condition. A page hit therefore generates a penalty. If this penalty generated at a page hit causes a problem, prepare an external circuit as illustrated in Figure 4.10-6 to use an external wait cycle based on the CAS signal, thereby extending I/O access to reserve data setup time.

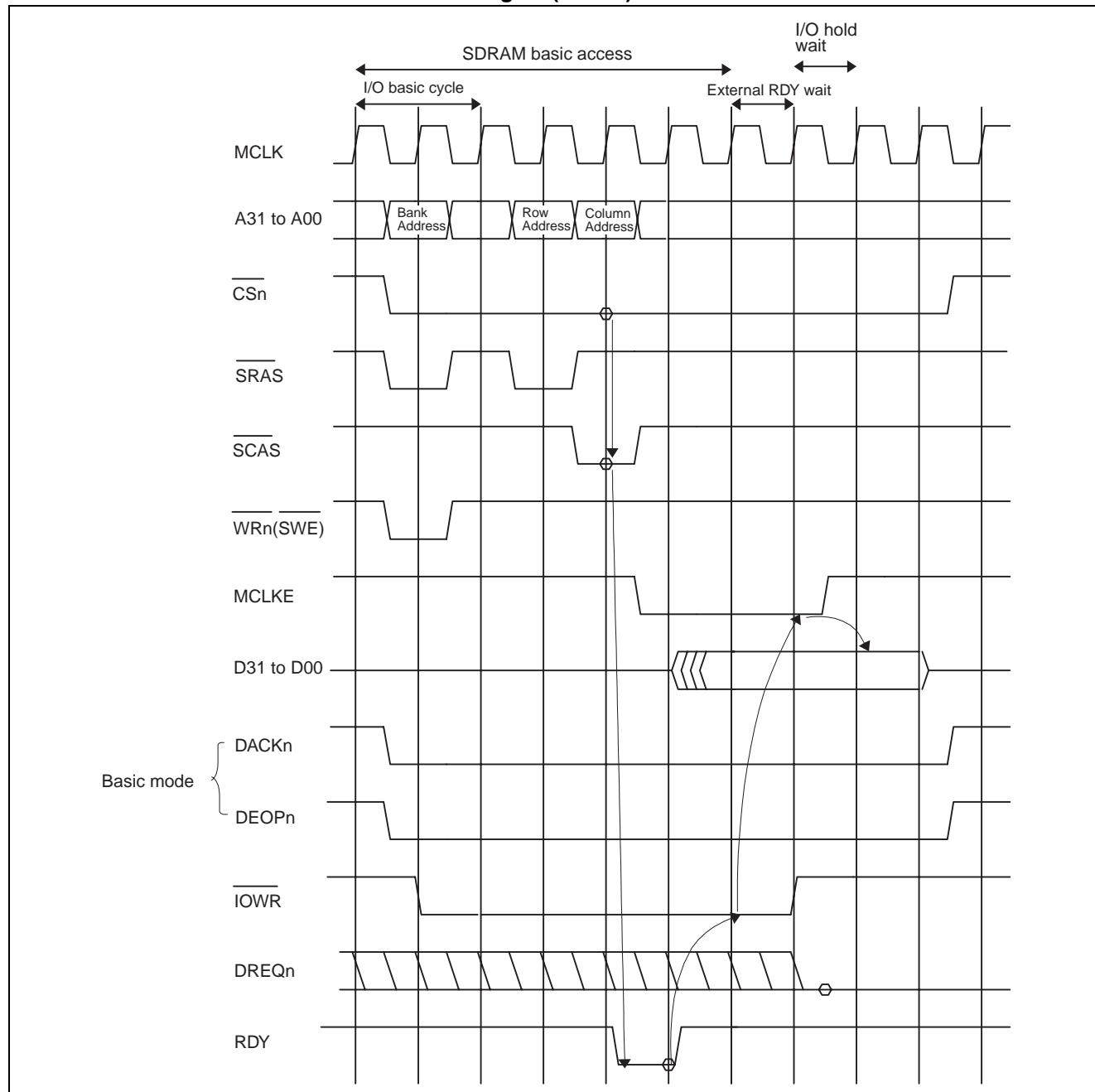
Figure 4.10-6 Sample Circuit Solving a Fly-by Penalty Using External Wait Cycles Based on the CAS Signal (CL = 2)



Notes:

- For CL = 3, provide two stages of MCLK-based FF to cause a delay of another cycle.
- If any device requires an external wait cycle, add a logic gate to the RDY signal as required.

Figure 4.10-7 Timing Chart for Fly-by Penalty Solution Using External Wait Cycles Based on the CAS Signal (CL = 2)



The rise of the IOWR signal can be delayed one cycle by extending SDRAM read access one cycle when the signal resulting from OR (negative-logic AND) operation of the CAS signal and the chip select signal for the SDRAM area subject to transfer is input to external RDY pin at the timing based on the MCLK.

As the external wait signal is generated based on the CAS signal fall timing in this case, the data setup time from the SDRAM data output to the I/O device can be reserved for one cycle, regardless of a page hit or miss in SDRAM.

Set the external wait using the RYE0 and RYE1 bits in the DMAC I/O wait register such that the RDY function of the DMA fly-by access channel to be used is enabled.

When the CAS latency is "3", SDRAM data output is delayed one cycle. Add one stage of flip flop by the MCLK to input the signal delayed one cycle from the above diagram to the RDY pin.

4.10.5 2-Cycle Transfer (Internal RAM -> External I/O, RAM)

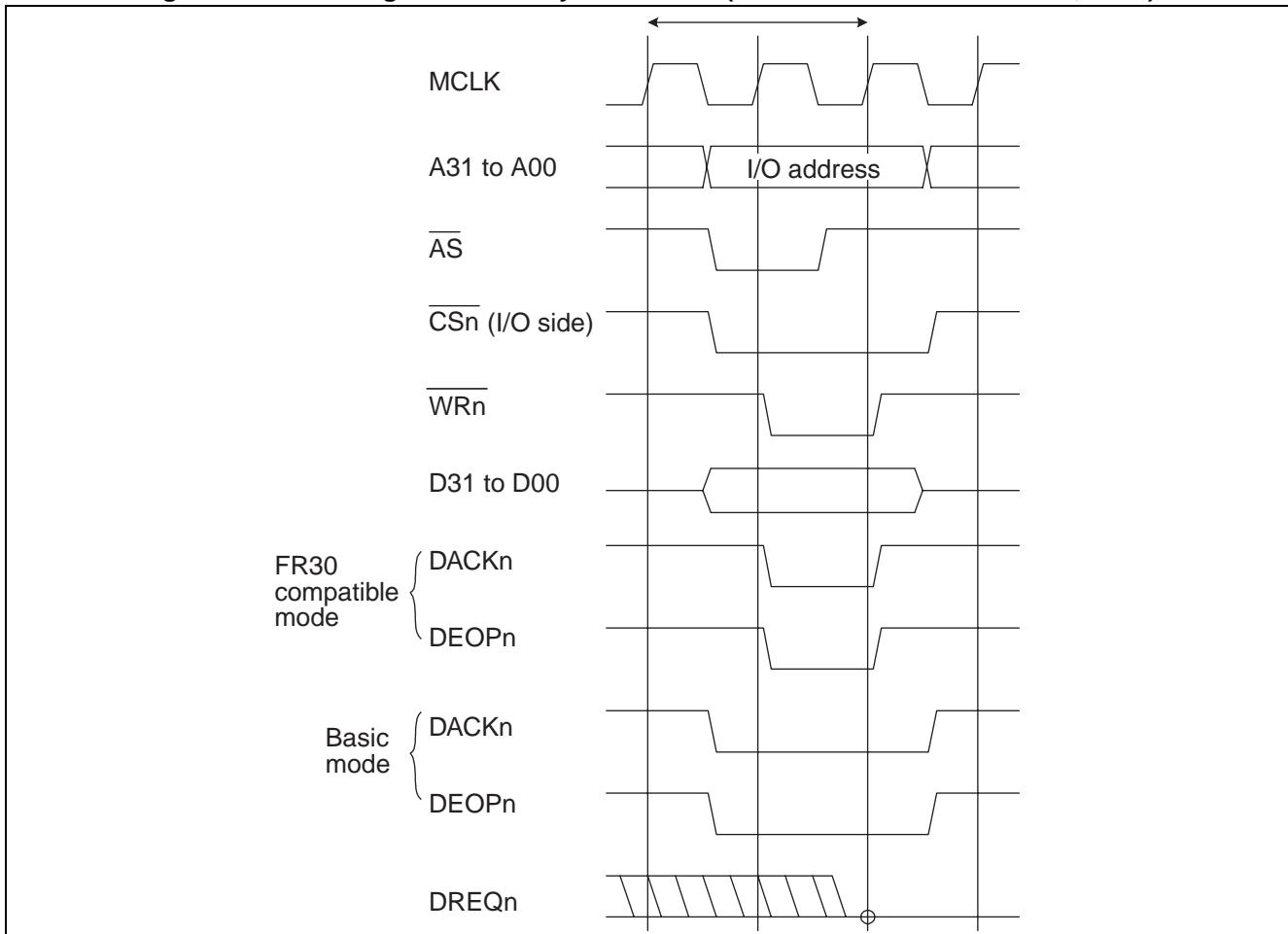
This section explains 2-cycle transfer (internal RAM -> external I/O, RAM) operation. The timing is the same as for external I/O, RAM -> internal RAM.

■ 2-Cycle Transfer (Internal RAM -> External I/O, RAM)

Figure 4.10-8 shows the operation timing chart for ($TYP3 \rightarrow TYP0=0000_B$, $AWR=0008_H$, $IOWR=00_H$).

Figure 4.10-8 shows a case in which a wait is not set on the I/O side.

Figure 4.10-8 Timing Chart for 2-cycle Transfer (Internal RAM -> External I/O, RAM)



- The bus is accessed in the same way as an interface when DMAC transfer is not performed.
- DACKn/DEOPn is not output in the internal RAM access cycles.

4.10.6 2-Cycle Transfer (External -> I/O)

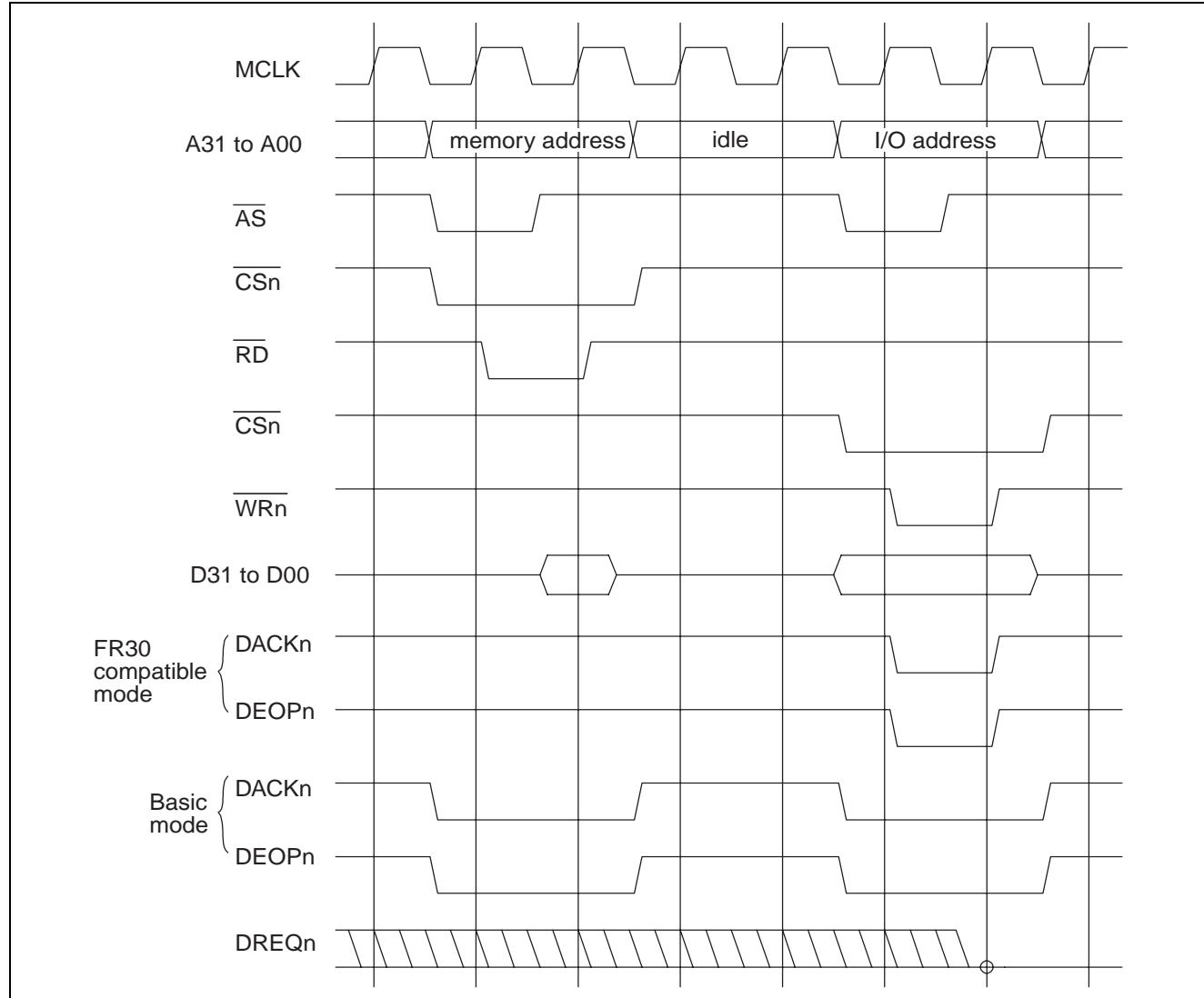
This section explains 2-cycle transfer (external -> I/O) operation.

■ 2-Cycle Transfer (External -> I/O)

Figure 4.10-9 shows the operation timing chart for (TYP3 to TYP0=0000_B, AWR=0008_H, IOWR=00_H).

Figure 4.10-9 shows a case in which a wait is not set for memory and I/O.

Figure 4.10-9 Timing Chart for 2-Cycle Transfer (External -> I/O)



- The bus is accessed in the same way as an interface when the DMAC transfer is not performed.
- In basic mode, DACKn/DEOPn is output in both transfer source bus access and transfer destination bus access.

4.10.7 2-Cycle Transfer (I/O -> External)

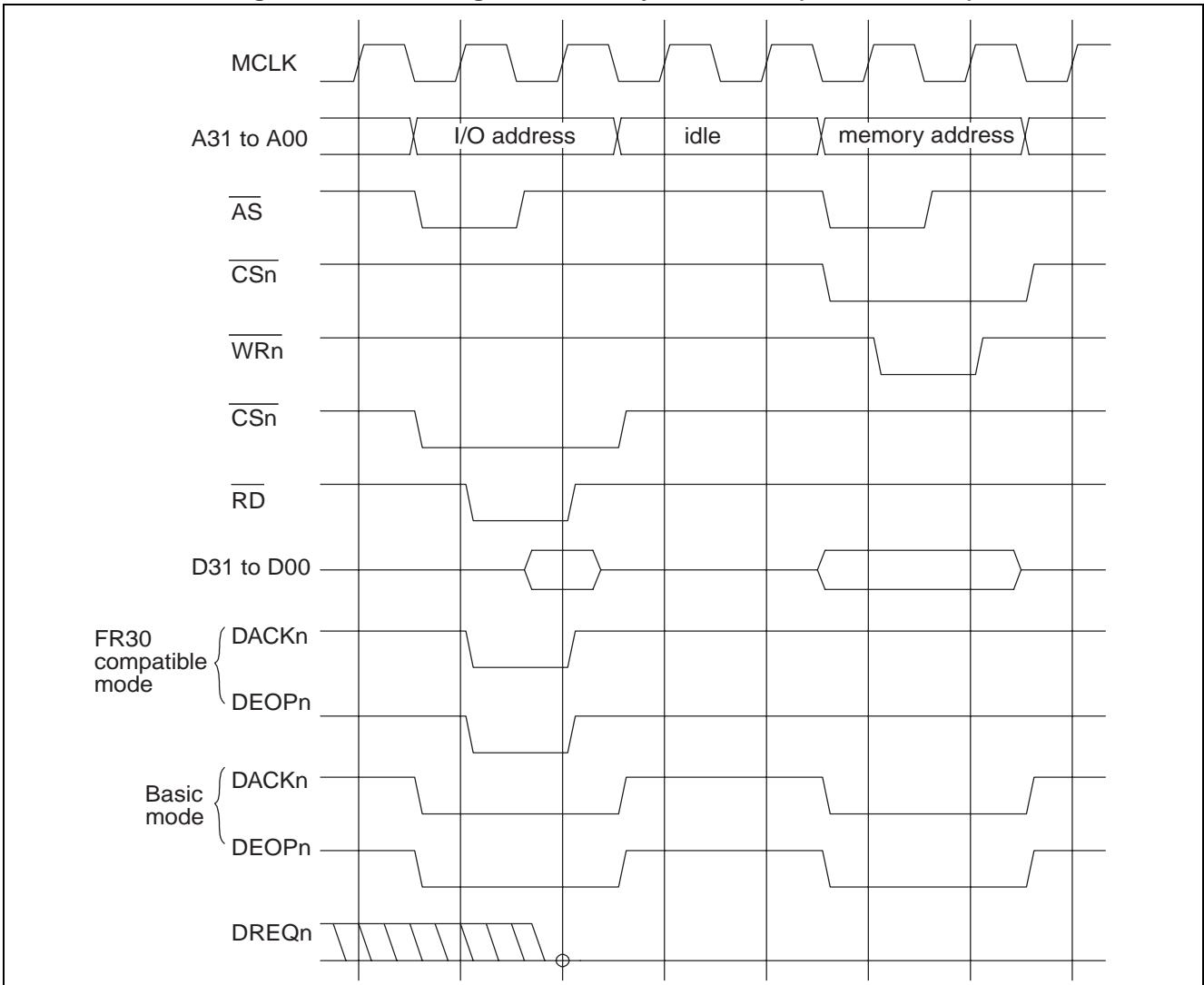
This section explains 2-cycle transfer (I/O -> external) operation.

■ 2-Cycle Transfer (I/O -> External)

Figure 4.10-10 shows the operation timing chart for ($TYP3$ to $TYP0=0000_B$, $AWR=0008_H$, $IOWR=00_H$).

Figure 4.10-10 shows a case in which a wait is not set for memory and I/O.

Figure 4.10-10 Timing Chart for 2-Cycle Transfer (I/O -> External)



- The bus is accessed in the same way as an interface when the DMAC transfer is not performed.
- In basic mode, **DACKn/DEOPn** is output both in the transfer source bus access and transfer destination bus access.

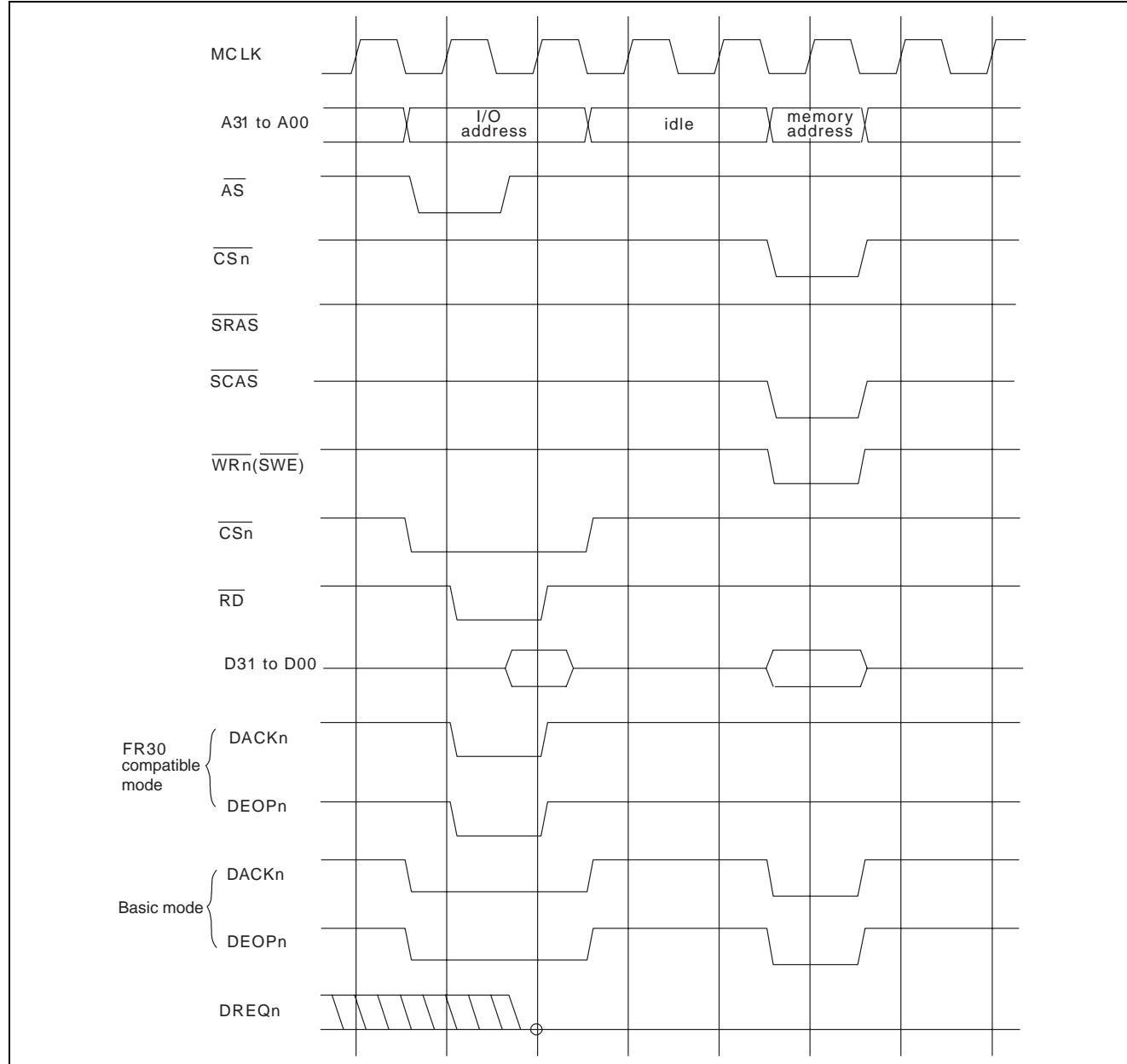
4.10.8 2-Cycle Transfer (I/O -> SDRAM/FCRAM)

This section describes the operation of 2-cycle transfer (I/O device to SDRAM/FCRAM).

■ 2-Cycle Transfer (I/O -> SDRAM/FCRAM)

Figure 4.10-11 shows an operation timing chart assuming TYP3 to TYP0 set to 1000_B, AWR set to 0051_H, and IOWR set to 00_H.

Figure 4.10-11 Timing Chart for 2-cycle Transfer (I/O to SDRAM/FCRAM)



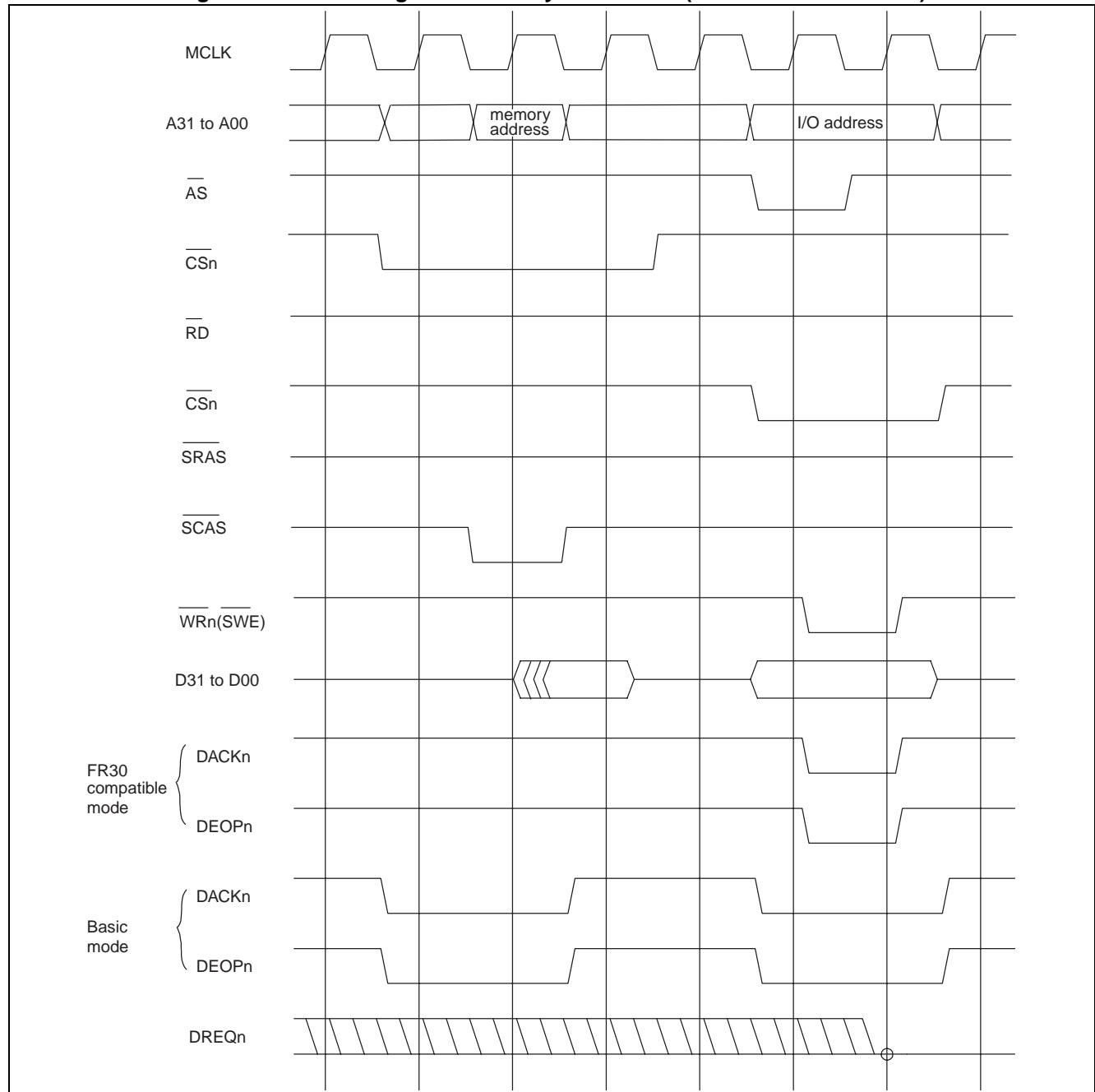
4.10.9 2-Cycle Transfer (SDRAM/FCRAM -> I/O)

This section describes the operation of 2-cycle transfer (SDRAM/FCRAM to I/O device).

■ 2-Cycle Transfer (SDRAM/FCRAM -> I/O)

Figure 4.10-12 shows a timing chart for 2-cycle transfer (SDRAM/FCRAM to I/O).

Figure 4.10-12 Timing Chart for 2-cycle Transfer (SDRAM/FCRAM to I/O)



- Bus access is the same as that of the interface for non-DMA transfer.
- In base mode, DACKn/DEOPn is output at both of transfer source bus access and transfer destination bus access.

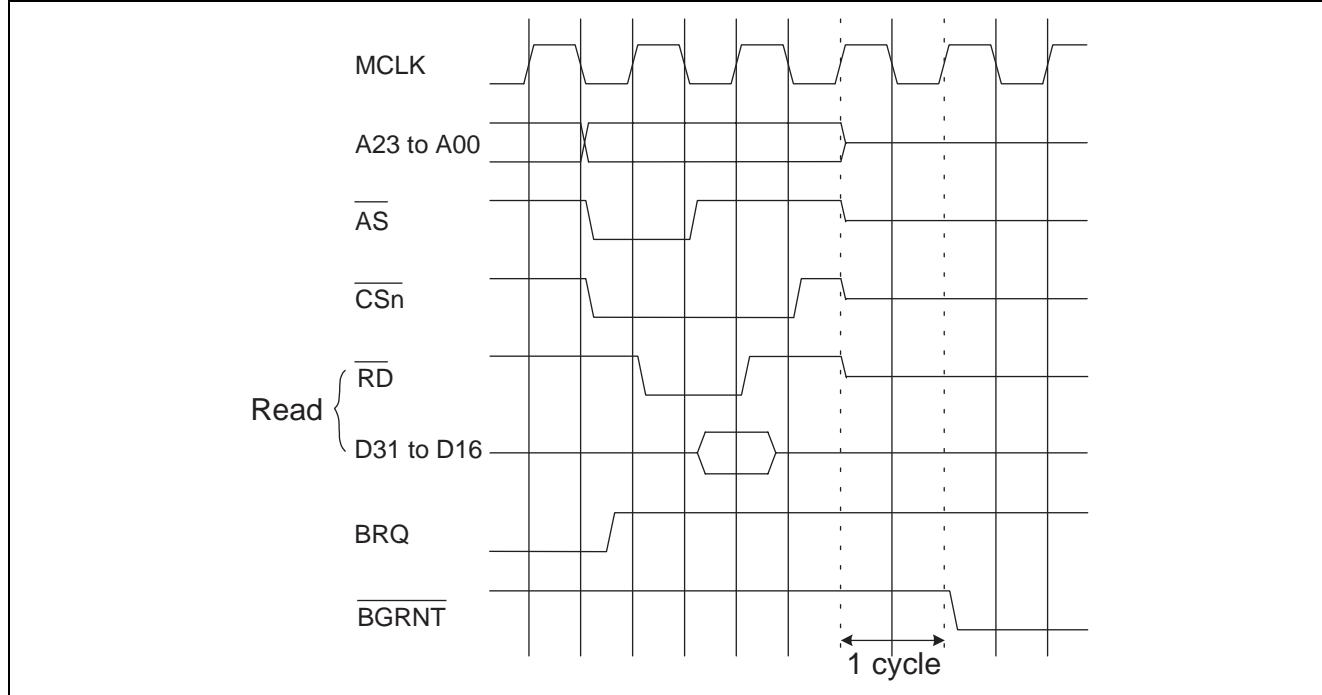
4.11 Bus Arbitration

This section shows timing charts for releasing the bus right and for acquiring the bus right.

■ Releasing the Bus Right

Figure 4.11-1 shows the timing chart for releasing the bus right.

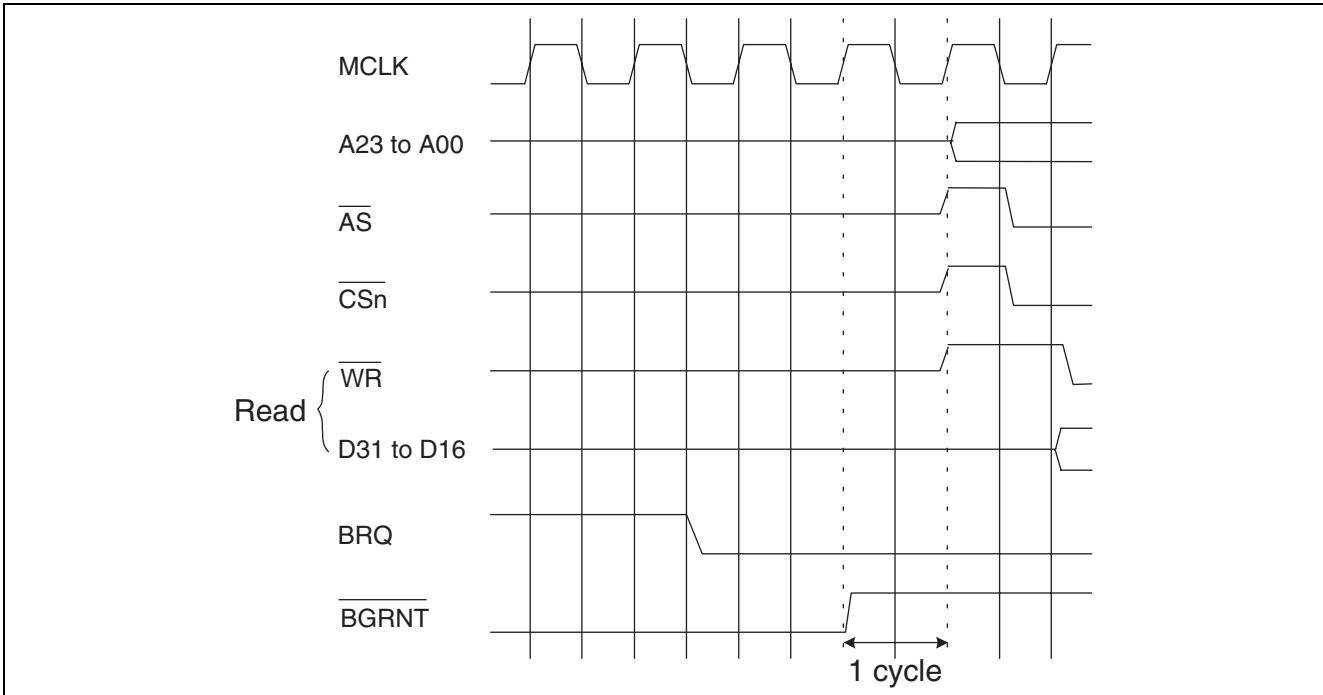
Figure 4.11-1 Timing Chart for Releasing the Bus Right



■ Acquiring the Bus Right

Figure 4.11-2 shows the timing chart for acquiring the bus right.

Figure 4.11-2 Timing Chart for Acquiring the Bus Right



- Setting "1" for the BREN bit of the TRC register enables bus arbitration by BRQ/BGRNT to be performed.
- When the bus right is released, the pin is set to high impedance and then BGRNT is asserted one cycle later.
- When the bus right is acquired, BGRNT is negated and then each pin is activated one cycle later.
- CSn is set to high impedance only if the SREN bit in the ACR0 to ACR7 registers is set.
- If all areas enabled by the CSER register are shared (the SREN bit of the ACR register is "1"), AS, BAA, RD, WR, and WR0 to WR3 are set to high impedance.

4.12 Procedure for Setting a Register

This section explains the procedure for setting a register.

■ Procedure for Setting a Register

Using the following procedures to make external bus interface settings:

1. Before rewriting the contents of a register, be sure to set the CSER register so that the corresponding area is not used (0). If you change the settings while "1" is set, access before and after the change cannot be guaranteed.
2. Use the following procedure to change a register:
 - 1) Set "0" for the CSER bit corresponding to the applicable area.
 - 2) Set both ASR and ACR at the same time using word access.
Set ACR after setting ASR if ASR and ACR are accessed by halfword.
 - 3) Set AWR.
 - 4) Set the CHER bit corresponding to the applicable area.
 - 5) Set the CSER bit corresponding to the applicable area.
3. The $\overline{CS0}$ area is enabled after a reset is released. If the area is used as a program area, the register contents need to be rewritten while the CSER bit is "1". In this case, make the settings described in 2) to 4) above in the initial state with a low-speed internal clock. Then, switch the clock to a high-speed clock.
4. Use the following procedure to change the register value in an area for which prefetch:
 - 1) Set "0" for the bit of CSER corresponding to the applicable area.
 - 2) Set "1" for both the PSUS bit and PCLR bit of the TCR register.
 - 3) Set both ASR and ACR at the same time using word access.
Set ACR after setting ASR if ASR and ACR are accessed by halfword.
 - 4) Set AWR.
 - 5) Set the CHER bit corresponding to the applicable area.
 - 6) Set "0" for both the PSUS bit and PCLR bit of the TCR register.
 - 7) Set "1" for the bit of CSER corresponding to the applicable area.

4.13 Notes on Using the External Bus Interface

This section explains some notes when using the external bus interface.

■ Notes for Use

If settings are made so that the area (TYP3 to TYP0=0x0xB) where $\overline{WR_0}$ to $\overline{WR_3}$ are used as a write strobe and the area (TYP3 to TYP0=0x1xB) where \overline{WR} is used as a write strobe are mixed, be sure to make the following setting in all areas that will be used:

- Set at least one read -> write idle cycle (other than setting AWR W07, W06=00B).
- Set at least one write recovery cycle (other than setting AWR W05, W04=00B).

However, if $\overline{WR_0}$ to $\overline{WR_3}$ are disabled (ROM only is connected) in the area (TYP3 to TYP0=0x0xB) where $\overline{WR_0}$ to $\overline{WR_3}$ are used as a write strobe, the above restriction does not apply. Also, the above restriction does not apply if both the address -> $\overline{RD}/\overline{WR_n}$ setup cycle (W01=1) and $\overline{RD}/\overline{WR_n}$ -> address hold cycle (W00=1) are set in the area (TYP3 to TYP0=0x1xB) where \overline{WE} is used as a write strobe.

CHAPTER 4 EXTERNAL BUS INTERFACE

CHAPTER 5 I/O PORT

This chapter describes the I/O ports and the configuration and functions of registers.

5.1 Overview of the I/O Port

5.2 I/O Port Registers

5.1 Overview of the I/O Port

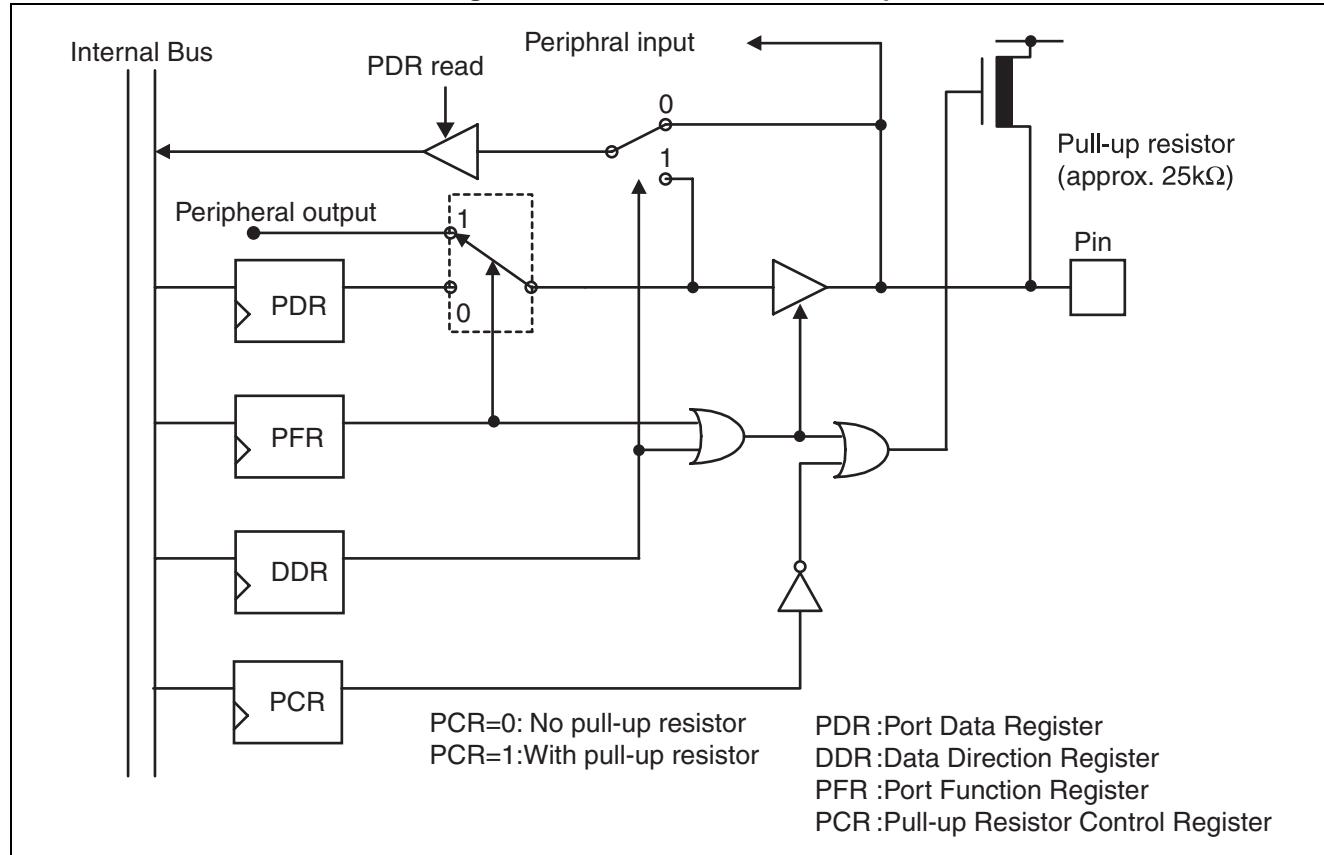
This section provides an overview of the I/O port.

■ Basic Block Diagram of the I/O Port

The MB91301 series interface can be used as an I/O port if settings are made so that the external bus interface or peripherals corresponding to pins do not use the pins as input/output pins.

Figure 5.1-1 shows the basic configuration of the I/O port.

Figure 5.1-1 Port Block with Pull-up



Note:

For port output, the pull-up resistor register is invalid regardless of the setting.

The I/O port consists of PDRs (Port Data Registers), DDRs (Data Direction Registers), PFRs (Port Function Registers) and PCR (Pull-up Resistance Control Registers).

■ I/O Port Modes

The I/O port has the following three modes:

○ **Port input mode (PFR=0 & DDR=0)**

- PDR read: Reads the level of the corresponding external pin.
- PDR write: Writes a setting value to the PDR.

○ **Port output mode (PFR=0 & DDR=1)**

- PDR read: Reads the value of the PDR.
- PDR write: Outputs the value of the PDR to the corresponding external pin.

○ **Peripheral output mode (PFR=1 & DDR=x)**

- PDR read: Reads the value of the corresponding peripheral output.
- PDR write: Writes a setting value to the PDR.

Notes:

Use byte access to access to the port.

- When a port from port 0 to port A is used as an external bus pin, the external bus function has priority. Thus, if the DDR register is rewritten while the port is functioning as an external bus pin, no input/output switching occurs. The DDR register value is enabled when the pin is switched to a general-purpose pin by changing the PFR register.
 - During stop mode ($HIZ = 0$), the setting of pull-up resistor control register has priority.
 - During stop mode ($HIZ = 1$), the setting of pull-up resistor control register is disable in the hardware standby state.
 - If port pin is used as the external bus terminal, using pull-up resistor is prohibited. Do not write "1" to the corresponding bit of pull-up resistor control register (PCR).
-

5.2 I/O Port Registers

This section describes the configuration and functions of the I/O port registers.

■ Configuration of the Port Data Registers (PDR)

Shown below is the configuration of the port data registers (PDR).

Figure 5.2-1 Configuration of the Port Data Registers (PDR)

PDR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000000H		P07	P06	P05	P04	P03	P02	P01	P00	XXXXXXXXX _B	R/W
PDR1	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000001H		P17	P16	P15	P14	P13	P12	P11	P10	XXXXXXXXX _B	R/W
PDR2	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000002H		P27	P26	P25	P24	P23	P22	P21	P20	XXXXXXXXX _B	R/W
PDR6	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000006H		P67	P66	P65	P64	P63	P62	P61	P60	XXXXXXXXX _B	R/W
PDR8	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000008H		P87	P86	P85	P84	P83	P82	P81	P80	XXXXXXXXX _B	R/W
PDR9	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000009H		-	P96	P95	P94	P93	P92	P91	P90	-XXXXXXXX	R/W
PDRA	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000AH		PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	XXXXXXXXX _B	R/W
PDRB	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000BH		PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	XXXXXXXXX _B	R/W
PDRG	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000010H		PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0	XXXXXXXXX _B	R/W
PDRH	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000011H		-	-	-	-	-	PH2	PH1	PH0	----XXX _B	R/W
PDRJ	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000013H		PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0	XXXXXXXXX _B	R/W

- PDR0 to PDR2, PDR6, PDR8 to PDRB, PDRG, PDRH and PDRJ are the input/output data registers for the I/O port.
 - Input/output is controlled by the corresponding DDR0 to DDR2, DDR6, DDR8 to DDRB, DDRG, DDRH and DDRJ, and PFR6, PFR8, PFR9, PFRA1, PFRB1, PFRB2, PFRA2, PFRG, PFRH and PFRJ.
 - There are not any PFR (Port Function Register) for P00 to P07, P10 to P17, P20 to P27.

■ Configuration of the Data Direction Registers (DDR)

Figure 5.2-2 shows the configuration of the data direction registers (DDR).

Figure 5.2-2 Configuration of the Data Direction Registers (DDR)

Register	Address	Bit Range	Initial Value	Access
DDR0	Address: 000600H	bit 7 6 5 4 3 2 1 0 P07 P06 P05 P04 P03 P02 P01 P00	Initial value 00000000B	Access R/W
DDR1	Address: 000601H	bit 7 6 5 4 3 2 1 0 P17 P16 P15 P14 P13 P12 P11 P10	Initial value 00000000B	Access R/W
DDR2	Address: 000602H	bit 7 6 5 4 3 2 1 0 P27 P26 P25 P24 P23 P22 P21 P20	Initial value 00000000B	Access R/W
DDR6	Address: 000606H	bit 7 6 5 4 3 2 1 0 P67 P66 P65 P64 P63 P62 P61 P60	Initial value 00000000B	Access R/W
DDR8	Address: 000608H	bit 7 6 5 4 3 2 1 0 P87 P86 P85 P84 P83 P82 P81 P80	Initial value 00000000B	Access R/W
DDR9	Address: 000609H	bit 7 6 5 4 3 2 1 0 - P96 P95 P94 P93 P92 P91 P90	Initial value -00000000B	Access R/W
DDRA	Address: 00060AH	bit 7 6 5 4 3 2 1 0 PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0	Initial value 00000000B	Access R/W
DDRB	Address: 00060BH	bit 7 6 5 4 3 2 1 0 PB7 PB6 PB5 PB4 PB3 PB2 PB1 PB0	Initial value 00000000B	Access R/W
DDRG	Address: 000400H	bit 7 6 5 4 3 2 1 0 PG7 PG6 PG5 PG4 PG3 PG2 PG1 PG0	Initial value 00000000B	Access R/W
DDRH	Address: 000401H	bit 7 6 5 4 3 2 1 0 - - - - - PH2 PH1 PH0	Initial value -----000B	Access R/W
DDRJ	Address: 000403H	bit 7 6 5 4 3 2 1 0 PJ7 PJ6 PJ5 PJ4 PJ3 PJ2 PJ1 PJ0	Initial value 00000000B	Access R/W

DDR0 to DDR2, DDR6, DDR8 to DDRB, DDRG, DDRH and DDRJ control the input/output direction of the corresponding I/O port at the bit level.

- If PFR=0
 - DDR=0: Port input
 - DDR=1: Port output
- If PFR=1
 - DDR=0: Peripheral input
 - DDR=1: Peripheral output

■ Configuration of the Pull-up Resistor Control Registers (PCR)

The configuration of the pull-up resistor control registers (PCR) is shown in Figure 5.2-3:

Figure 5.2-3 Configuration of the Pull-up Resistor Control Registers (PCR)

PCR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000620H		P07	P06	P05	P04	P03	P02	P01	P00	00000000B	R/W
PCR1	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000621H		P17	P16	P15	P14	P13	P12	P11	P10	00000000B	R/W
PCR2	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000622H		P27	P26	P25	P24	P23	P22	P21	P20	00000000B	R/W
PCR6	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000626H		P67	P66	P65	P64	P63	P62	P61	P60	00000000B	R/W
PCR8	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000628H		P87	P86	P85	P84	P83	P82	P81	P80	00000000B	R/W
PCR9	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000629H		-	P96	P95	P94	-	-	P91	-	-000--0-B	R/W
PCRA	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00062AH		PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	00000000B	R/W
PCRB	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00062BH		PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	00000000B	R/W
PCRH	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000423H		-	-	-	-	-	PH2	PH1	PH0	----000B	R/W

PCR0 to PCR2, PCR6, PCR8 to PCRB, and PCRH control the pull-up resistor of the corresponding I/O port.

- PCR=0: No pull-up resistance
- PCR=1: Pull-up resistance

■ Configuration of the Port Function Registers (PFR)

The configuration of the port function registers (PFR) is shown in Figure 5.2-4:

Figure 5.2-4 Configuration of the Port Function Registers (PFR)

PFR6	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000616 _H		AE23	AE22	AE21	AE20	AE19	AE18	AE17	AE16	11111111 _B	R/W
PFR8	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000618 _H		WR3XE	WR2XE	WR1XE	-	-	BRQE	-	-	111--0--B	R/W
PFR9	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000619 _H		-	WRXE	BAAE	ASXE	-	MCKE	MCKEE	SYSE	-0000111 _B	R/W
PFRA1	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00061A _H		CS7XE	CS6XE	CS5XE	CS4XE	CS3XE	CS2XE	CS1XE	CS0XE	11111111 _B	R/W
PFRB1	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00061B _H		DES1	AK12	AK11	AK10	DESO	AK02	AK01	AK00	00000000 _B	R/W
PFRB2	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00061C _H		DRDE	DWRE	PPE1	-	-	-	AKH1	AKH0	000---00 _B	R/W
PFRA2	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00061E _H		-	-	PPE2	-	-	-	-	-	--0-----B	R/W
PFRG	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000410 _H		SCE2	SOE2	-	-	-	-	-	-	00-----B	R/W
PFRH	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000411 _H		-	-	-	-	-	-	PPE3	-	-----0-B	R/W
PFRJ	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000413 _H		-	PPE0	SCE1	SOE1	-	SCE0	SOE0	-	-000-00-B	R/W
PFR61	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address: 000417 _H		-	-	-	-	TEST1	TEST0	I2CE1	I2CE0	---0000 _B	R/W

PFR6, PFR8, PFR9, PFRA1, PFRB1, PFRB2, PFRA2, PFRG, PFRH and PFRJ control the output of the corresponding external bus interface and peripherals at the bit level.

Be sure to set "0" for Empty bit of PFR.

■ Function of the Port Function Registers (PFR)

The following table summarizes the initial values and functions of the PFR registers.

Table 5.2-1 Functions of the Port Function Registers (PFR) (1 / 5)

Register name	Bit name	Bit value	Function	Initial value
PFR6 000616H	AE16	0	General-purpose port (P60)	1
		1	Address output	
	AE17	0	General-purpose port (P61)	1
		1	Address output	
	AE18	0	General-purpose port (P62)	1
		1	Address output	
	AE19	0	General-purpose port (P63)	1
		1	Address output	
	AE20	0	General-purpose port (P64)	1
		1	Address output	
	AE21	0	General-purpose port (P65)	1
		1	Address output	
	AE22	0	General-purpose port (P66)	1
		1	Address output	
	AE23	0	General-purpose port (P67)	1
		1	Address output	
PFR8 000618H	BRQE	0	General-purpose port (P82, P81)	0
		1	Enable at setting BREN of BRQ, \overline{BGRNT} and TCR register = 1.	
	WR1XE	0	General-purpose port (P85)	1
		1	WR1/ULBX/DQMUL	
	WR2XE	0	General-purpose port (P86)	1
		1	WR2/LUBX/DQMLU	
	WR3XE	0	General-purpose port (P87)	1
		1	WR3/LUBX/DQMLL	
PFR9 000619H	SYSE	0	General-purpose port (P90)	1
		1	Set "1" at using SYSCLK.	
	MCKEE	0	General-purpose port (P91)	1
		1	MCLKE	

Table 5.2-1 Functions of the Port Function Registers (PFR) (2 / 5)

Register name	Bit name	Bit value	Function	Initial value
PFR9 000619H	MCKE	0	General-purpose port (P92)	1
		1	Set "1" at using memory clock and MCLK	
	ASXE	0	General-purpose port (P94)	0
		1	Set "1" at using $\overline{AS}/\overline{LBA}/\overline{SRAS}$ and general/burst memory	
	BAAE	0	General-purpose port (P95)	0
		1	Set "1" at using $\overline{BAA}/\overline{SCAS}$ and burst memory	
	WRXE	0	General-purpose port (P96)	0
		1	Set "1" at using $\overline{WRn}/\overline{SWR}$ and general/burst memory	
PFRA 00061AH	CS0XE	0	General-purpose port (PA0)	1
		1	$\overline{CS0}$ output, Enable at setting CSE0 bit of CSER register to "1".	
	CS1XE	0	General-purpose port (PA1)	1
		1	$\overline{CS1}$ output, Enable at setting CSE1 bit of CSER register to "1".	
	CS2XE	0	General-purpose port (PA2)	1
		1	$\overline{CS2}$ output, Enable at setting CSE2 bit of CSER register to "1".	
	CS3XE	0	General-purpose port (PA3)	1
		1	$\overline{CS3}$ output, Enable at setting CSE3 bit of CSER register to "1".	
	CS4XE	0	General-purpose port (PA4)	1
		1	$\overline{CS4}$ output, Enable at setting CSE4 bit of CSER register to "1".	
	CS5XE	0	General-purpose port (PA5)	1
		1	$\overline{CS5}$ output, Enable at setting CSE5 bit of CSER register to "1".	
	CS6XE	0	General-purpose port (PA6)	1
		1	$\overline{CS6}$ output, Enable at setting CSE6 bit of CSER register to "1".	
	CS7XE	0	General-purpose port (PA7)	1
		1	$\overline{CS7}$ output, Enable at setting CSE7 bit of CSER register to "1".	

Table 5.2-1 Functions of the Port Function Registers (PFR) (3 / 5)

Register name	Bit name	Bit value	Function	Initial value
PFRB1 00061BH	AK02, AK01, AK00	0,0,0	General-purpose port (PB0, PB1, PB2)	000
		0,0,1	DACK0, DEOP0 output (FR30-compatible for fly-by transfer)	
		0,1,0	DACK0, DEOP0 output (FR30-compatible for two-cycle transfer RD timing)	
		0,1,1	DACK0, DEOP0 output (FR30-compatible for two-cycle transfer WRn timing)	
		1,0,0	DACK0, DEOP0 output (FR30-compatible for two-cycle transfer WE timing)	
		1,0,1	DACK0, DEOP0 output (FR30-compatible for two-cycle transfer WRn/RD timing)	
		1,1,0	DACK0, DEOP0 output (FR30-compatible for two-cycle transfer WE, RD timing)	
		1,1,1	DACK0, DEOP0 output (chip select timing)	
	DES0, PB2 (DDRB)	0,0	General-purpose port input (PB2)	00
		0,1	General-purpose port output (PB2)	
		1,0	DMAC: DSTP0 input (setting prohibited)	
		1,1	DMAC: DEOP0 output	
	AK12, AK11, AK10	0,0,0	General-purpose port (PB3, PB4, PB5)	000
		0,0,1	DACK1, DEOP1 output (FR30-compatible for fly-by transfer)	
		0,1,0	DACK1, DEOP1 output (FR30-compatible for two-cycle transfer RD timing)	
		0,1,1	DACK1, DEOP1 output (FR30-compatible for two-cycle transfer WRn timing)	
		1,0,0	DACK1, DEOP1 output (FR30-compatible for two-cycle transfer WE timing)	
		1,0,1	DACK1, DEOP1 output (FR30-compatible for two-cycle transfer WRn/RD timing)	
		1,1,0	DACK1, DEOP1 output (FR30-compatible for two-cycle transfer WE, RD timing)	
		1,1,1	DACK1, DEOP1 output (chip select timing)	
	DES1, PB5 (DDRB)	0,0	General-purpose port input (PB5)	00
		0,1	General-purpose port output (PB5)	
		1,0	DMAC: DSTP1 input (setting prohibited)	
		1,1	DMAC: DEOP1 output	

Table 5.2-1 Functions of the Port Function Registers (PFR) (4 / 5)

Register name	Bit name	Bit value	Function	Initial value
PFRB2 00061CH	AKH0	0	DACK0 output active L	0
		1	DACK0 output active H	
	AKH1	0	DACK1 output active L	0
		1	DACK1 output active H	
	PPE1	0	General-purpose port (PB5)/DEOP1 output	0
		1	PPG1 output	
	DWRE	0	General-purpose port (PB6)	0
		1	IOWR output	
	DRDE	0	General-purpose port (PB7)	0
		1	IORD output	
PFRA2 00061EH	PPE2	0	General-purpose port (PA5)/CS5 output	0
		1	PPG2 output	
PFRG 000410H	SOE2	0	General-purpose port (PG6)	0
		1	SOT2 output	
	SCE2	0	General-purpose port (PG7)	0
		1	SCK2 output	
PFRH 000411H	PPE3	0	General-purpose port (PH1)	0
		1	PPG3 output	
PFRJ 000413H	SOE0	0	General-purpose port (PJ1)	0
		1	SOT0 output	
	SCE0	0	General-purpose port (PJ2)	0
		1	SCK0 output	
	SOE1	0	General-purpose port (PJ4)	0
		1	SOT1 output	
	SCE1	0	General-purpose port (PJ5)	0
		1	SCK1 output	
	PPE0	0	General-purpose port (PJ6)	0
		1	PPG0 output	
PFR61 000617H	I ² CE0	0	General-purpose port (P65, P64)/address output (A21, A20)	0
		1	I ² C I/F, SCL0, SDA0 I/O	

Table 5.2-1 Functions of the Port Function Registers (PFR) (5 / 5)

Register name	Bit name	Bit value	Function	Initial value
PFR61 000617 _H	I ² CE1	0	General-purpose port (P67, P66)/address output (A23, A22)	0
		1	I ² C I/F, SCL1, SDA1 I/O	
	TEST0	0	Be sure to set "0".	0
		1	Test function. Setting disabled.	
	TEST1	0	Be sure to set "0".	0
		1	Test function. Setting disabled.	

- For enabled PPG1 output, set PPE1 bit = 1 and DES1 bit = 0.
- For enabled PPG2 output, set PPE2 bit = 1 and CS5XE bit = 0.
- For enabled SDA0 and SCL0 output, set I²CE0 bit = 1.
- For enabled SDA1 and SCL1 output, set I²CE1 bit = 1.
- For enabled general port (P67), set I²CE1 bit = 0 and AE23 bit of the PFR6 register = 0.
- For enabled general port (P66), set I²CE1 bit = 0 and AE22 bit of the PFR6 register = 0.
- For enabled general port (P65), set I²CE0 bit = 0 and AE21 bit of the PFR6 register = 0.
- For enabled general port (P64), set I²CE0 bit = 0 and AE20 bit of the PFR6 register = 0.
- For enabled address output (A23), set I²CE1 bit = 0 and AE23 bit of the PFR6 register = 1.
- For enabled address output (A22), set I²CE1 bit = 0 and AE22 bit of the PFR6 register = 1.
- For enabled address output (A21), set I²CE0 bit = 0 and AE21 bit of the PFR6 register = 1.
- For enabled address output (A20), set I²CE0 bit = 0 and AE20 bit of the PFR6 register = 1.

CHAPTER 6 16-BIT RELOAD TIMER

This chapter describes the 16-bit reload timer, the configuration and functions of registers, and 16-bit reload timer operation.

- 6.1 Overview of the 16-bit Reload Timer
- 6.2 16-bit Reload Timer Registers
- 6.3 16-bit Reload Timer Operation
- 6.4 Operating States of the Counter
- 6.5 Precautions on Using the 16-bit Reload Timer

6.1 Overview of the 16-bit Reload Timer

The 16-bit reload timer consists of a 16-bit down counter, a 16-bit reload register, a prescaler for creating an internal count clock, and a control register.

■ Overview of the 16-bit Reload Timer

The 16-bit reload timer consists of a 16-bit down counter, a 16-bit reload register, a prescaler for creating an internal count clock, and a control register.

The input clock can be selected from three internal clocks (machine clock divided by 2, 8, and 32) and an external clock.

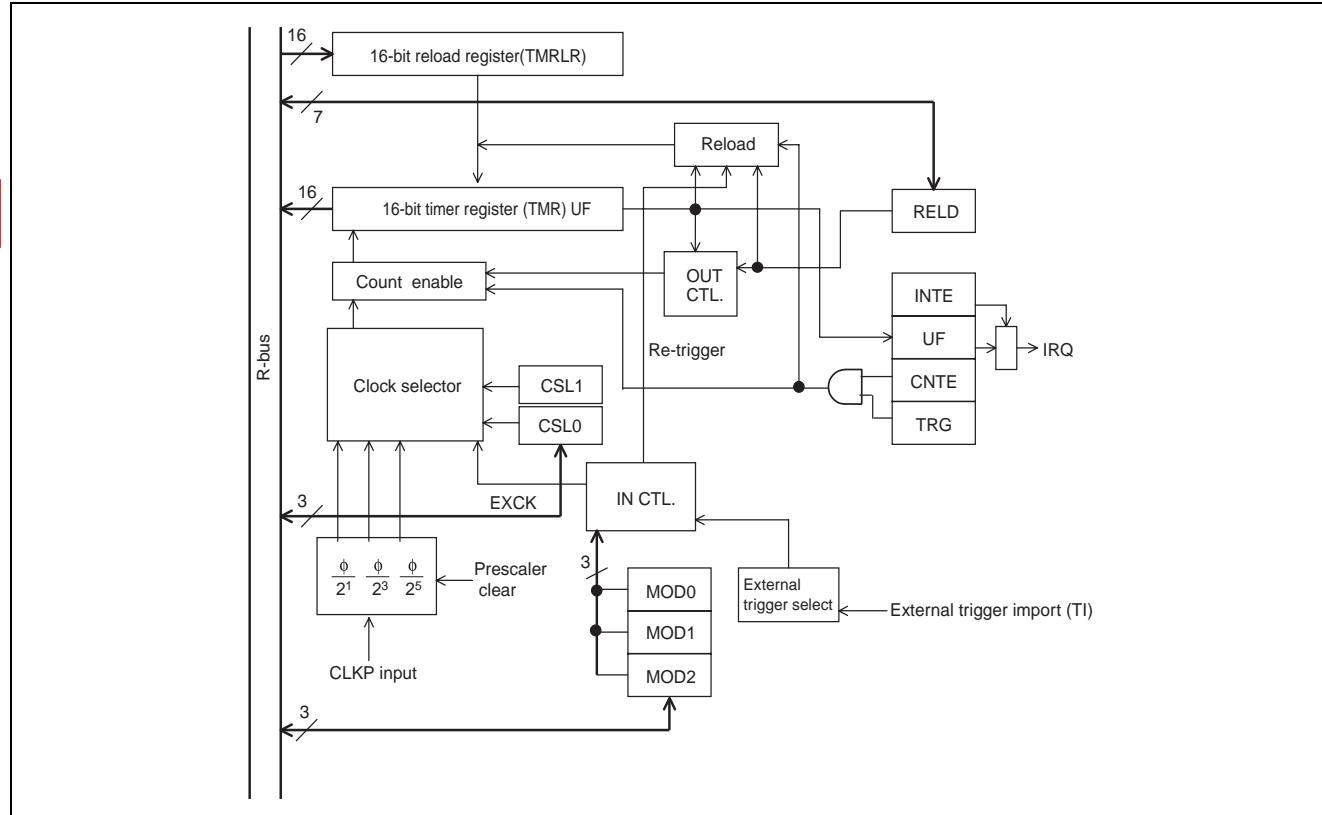
Channels 0 and 1 supports the activation of DMA transfers resulting from interrupts.

The MB91301 series has three built-in channels, for the 16-bit reload timer.

■ Block Diagram

Figure 6.1-1 is a block diagram of the 16-bit reload timer.

Figure 6.1-1 Block Diagram of the 16-bit Reload Timer



6.2 16-bit Reload Timer Registers

This section describes the configuration and functions of the registers used by the 16-bit reload timer.

■ 16-bit Reload Timer Registers

Figure 6.2-1 16-bit Reload Timer Registers

bit	15	14	13	12	11	10	9	8	
	-	-	-	-	CSL1	CSL0	MOD2	MOD1	Control status register (TMCSR)
bit	7	6	5	4	3	2	1	0	
	MOD0	-	-	RELD	INTE	UF	CNTE	TRG	
bit	15							0	16-bit timer register (TMR)
bit	15							0	16-bit reload register (TMRLR)

6.2.1 Control Status Register (TMCSR)

The control status register (TMCSR) controls the operating modes and interrupts of the 16-bit timer.

■ Bit Configuration of the Control Status Register (TMCSR)

Figure 6.2-2 Bit Configuration of the Control Status Register (TMCSR)

bit	15	14	13	12	11	10	9	8	Initial value
Address: 00004E _H	Reserved	Reserved	Reserved	Reserved	CSL1	CSL0	MOD2	MOD1	--XX0000 _B
000056 _H	-	-	(R/W)	(R/W)	R/W	R/W	R/W	R/W	00000000 _B
00005E _H									
bit	7	6	5	4	3	2	1	0	
	MOD0	Reserved	Reserved	RELD	INTE	UF	CNTE	TRG	
	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	

Rewrite bits other than the UF, CNTE, and TRG bits only when CNTE=0.

The control status register (TMCSR) supports simultaneous writing.

When write to bit5, bit12, and bit13, be sure to write "0".

■ Bit Functions of the Control Status Register (TMCSR)

The following describes the bit functions of the control status register (TMCSR).

[bit15] (Reserved)

This bit is unused.

[bit14] (Reserved)

This bit is unused.

[bit13] (Reserved)

Be sure to write "0" at write.

[bit12] (Reserved)

Be sure to write "0" at write.

[bit11, bit10] CSL1, CSL0 (Count source SeLect)

These bits are the count source select bits. Count sources can be selected from the internal clock or the external event. Table 6.2-1 shows the count sources that can be selected using these bits. Countable edges used when external event count mode are set using the MOD1 and MOD0 bits

Table 6.2-1 Count Sources Set Using the CSL Bits

CSL1	CSL0	Clock source (ϕ: Machine clock)
0	0	Internal clock $\phi/2^1$ (ch.0 to ch.2)
0	1	Internal clock $\phi/2^3$ (ch.0 to ch.2)
1	0	Internal clock $\phi/2^5$ (ch.0 to ch.2)
1	1	External clock (event) (ch.0 to ch.2)

Note:

The minimum pulse width required for an external clock is $2T$ (T : Peripheral clock machine cycle).

[bit9, bit8, bit7] MOD2, MOD1, MOD0 (MODe): Setting of operation mode

These bits set the operating modes.

These functions are switched by the count source ("internal clock" or "external event").

- Internal clock: setting reload trigger
- External event: setting count enable edge

The MOD2 bit has to be set to "0".

[Reload trigger setting at selecting internal clock]

When internal clock (CSL1, CSL0 = 00_B , 01_B , 10_B) are selected as count source, the contents of reload register are loaded after inputted enable edge by setting MOD2, MOD1, and MOD0 bits, and count function keep operating. Table 6.2-2 describes the settings of the MOD2, MOD1, and MOD0 bits.

Table 6.2-2 MOD2, MOD1, and MOD0 Bits Setting Method (in Internal Clock Mode)

MOD2	MOD1	MOD0	Valid edge
0	0	0	Software trigger
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)
1	x	x	Setting prohibited

Note: x in this table represents any value.

[Valid edge setting at selecting external event]

When external clock event (CSL1, CSL0 = 11_B) are selected as count source, the event is counted after inputted enable edge by setting MOD2 to MOD0 bits. Table 6.2-3 describes the settings of the MOD2, MOD1, and MOD0 bits.

Table 6.2-3 MOD2,MOD1, and MOD0 Bits Setting Method (in Selecting Event Count Mode)

MOD2	MOD1	MOD0	Valid edge or level
x	0	0	-
	0	1	External event (Rising edge)
	1	0	External event (Falling edge)
	1	1	External event (Both edges)

Note: x in this table represents any value.

Reload of external event is generated by underflow and software trigger.

[bit6] (Reserved)

This bit is unused.

The read value is always "0".

[bit5] (Reserved)

Be sure to write "0" at writing.

[bit4] RELD: Reload enable

This bit is the reload enable bit. If it is set to "1", reload mode is entered. As soon as the counter value underflows from 0000_H to $FFFF_H$, the contents of the reload register are loaded into the counter and the count operation is continued.

If this bit is set to "0", the count operation is stopped when the counter value underflows from 0000_H to $FFFF_H$.

[bit3] INTE: Interrupt request enable

This bit is the interrupt request enable bit. If the INTE bit is set to "1", an interrupt request is generated when the UF bit is set to "1". If it is set to "0", no interrupt request is generated.

[bit2] UF: Timer interrupt request

This bit is the timer interrupt request flag. This bit is set to "1" when the counter value underflows from 0000_H to $FFFF_H$. Write "0" to this bit to clear it.

Writing "1" to this bit is meaningless. When this bit is read by a read-modify-write instruction, "1" is always read.

[bit1] CNTE: Count enable bit of timer

This bit is the count enable bit of the timer. Write "1" to this bit to enter the start trigger wait state. Write "0" to this bit to stop the count operation.

[bit0] TRG: Software trigger

This bit is the software trigger bit. Write "1" to this bit to generate a software trigger, load the contents of the reload register into the counter, and start the count operation.

Writing "0" to this bit is meaningless. The read value is always "0".

The trigger input to this register is valid only if CNTE=1. No operation occurs if CNTE=0.

6.2.2 16-bit Timer Register (TMR:TMR2 to TMR0)

The 16-bit timer register (TMR:TMR2 to TMR0) is a register to which the count value of the 16-bit timer can be read. The initial value is undefined.

Be sure to read this register using a 16-bit data transfer instruction.

■ Bit Configuration of the 16-bit Timer Register (TMR:TMR2 to TMR0)

Figure 6.2-3 shows the bit configuration of the 16-bit timer register (TMR:TMR2 to TMR0).

Figure 6.2-3 Bit Configuration of the 16-bit Timer Register (TMR:TMR2 to TMR0)

																bit0	Initial value
Address: 00004A _H																XXXX _H	
000052 _H	R	R	R	R	...			R	R	R	R						
00005A _H	X	X	X	X	...			X	X	X	X						

6.2.3 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0)

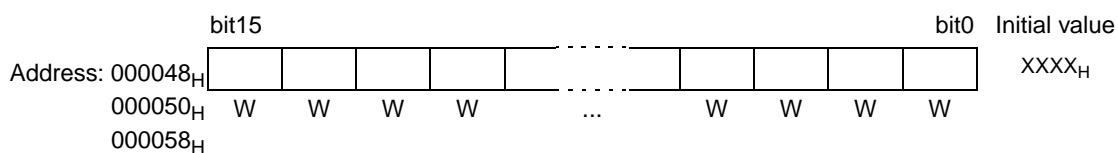
The 16-bit reload register (TMRLR:TMRLR2 to TMRLR0) holds the initial value of a counter. The initial value is undefined.

Be sure to read this register using a 16-bit data transfer instruction.

■ Bit Configuration of the 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0)

Figure 6.2-4 shows the bit configuration of the 16-bit reload register (TMRLR:TMRLR2 to TMRLR0).

Figure 6.2-4 Bit Configuration of the 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0)



6.3 16-bit Reload Timer Operation

This section describes the following operations of the 16-bit reload timer:

- Internal clock operation
- Underflow operation

■ Internal Clock Operation

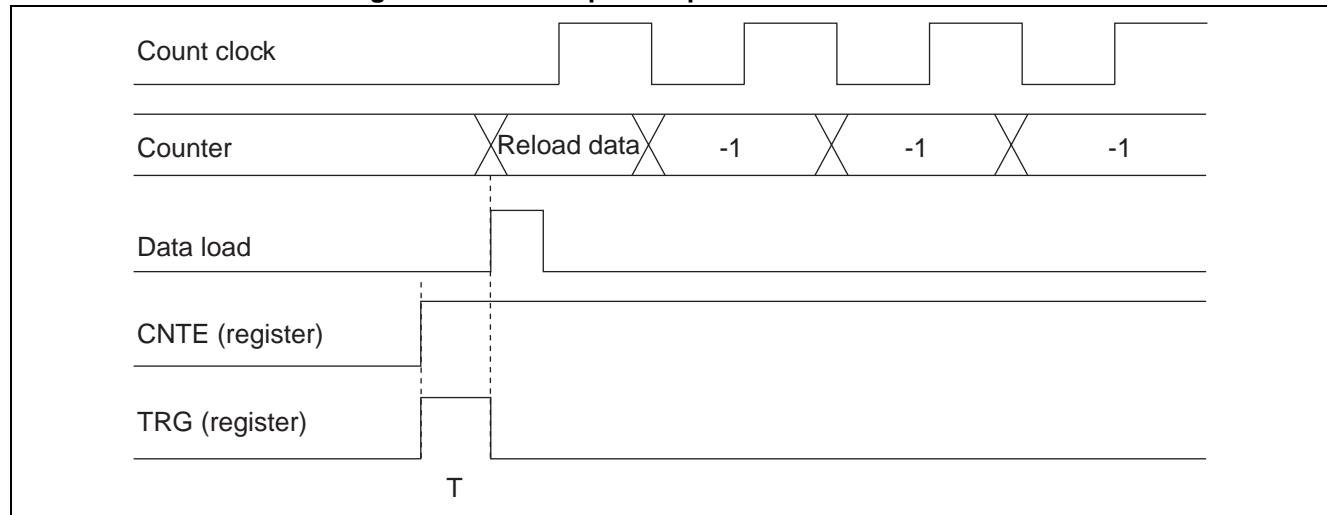
If the timer operates with a divide-by clock of the internal clock, one of the clocks created by dividing the machine clock by 2, 8, or 32 can be selected as the clock source.

To start the count operation as soon as counting is enabled, write "1" to the CNTE and TRG bits of the control status register. Trigger input occurring due to the TRG bit is always valid regardless of the operating mode while the timer is running (CNTE=1).

Figure 6.3-1 shows the startup and operations of the counter.

Time as long as T (T: peripheral clock machine cycle) is required after the counter start trigger is input and before the data of the reload register is actually loaded into the counter.

Figure 6.3-1 Startup and Operations of the Counter



■ Underflow Operation

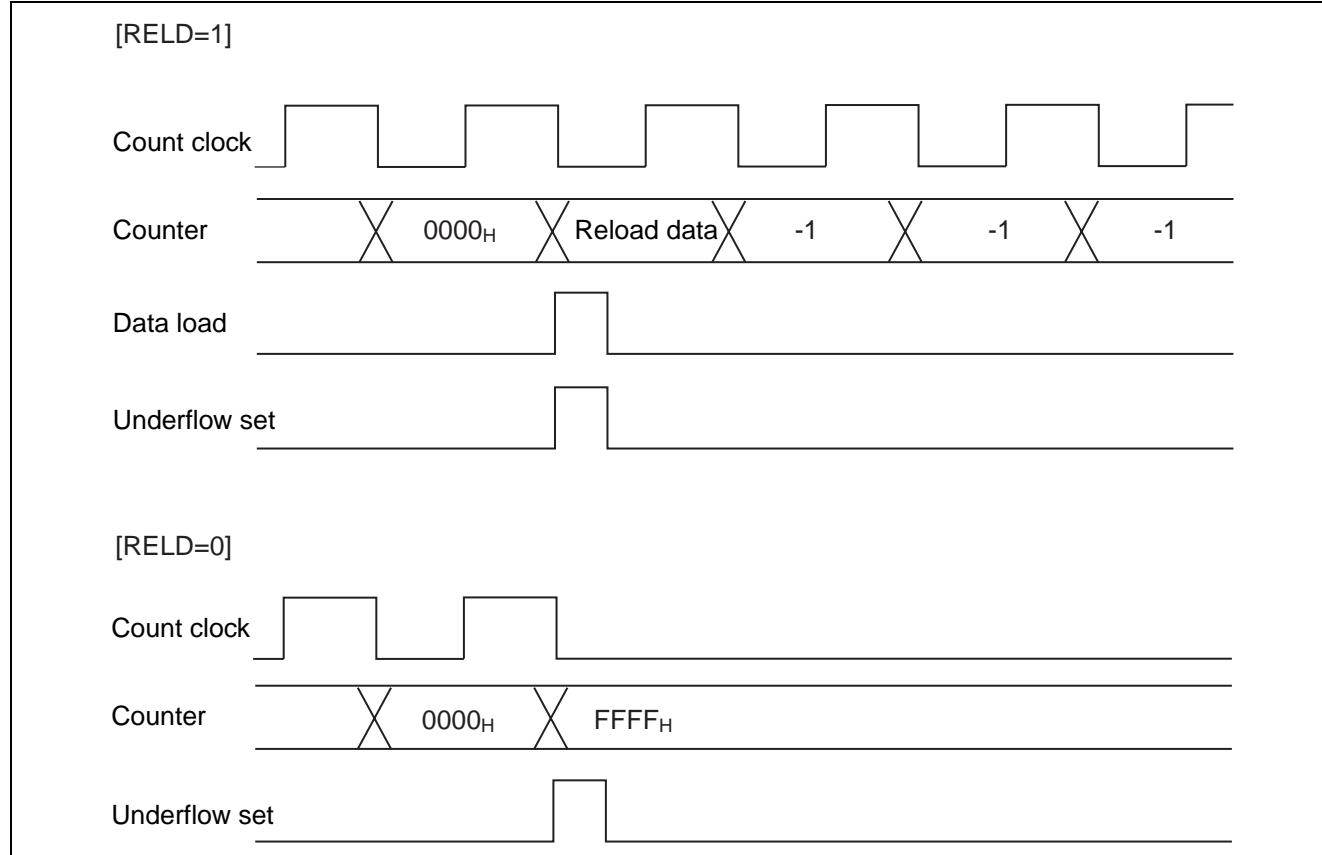
An underflow is an event in which the counter value changes from 0000_H to $FFFF_H$. Thus, an underflow occurs at the count of [Reload register setting value + 1].

If the RELD bit of the control status register (TMCSR) is set to "1" when an underflow occurs, the contents of the 16-bit reload register (TMRLR) are loaded and the count operation is continued. If the RELD bit is set to "0", the counter stops at $FFFF_H$.

An underflow sets the UF bit of the control status register (TMCSR) and, if the INTE bit is set to "1", generates an interrupt request.

Figure 6.3-2 shows the timing chart of the underflow operation.

Figure 6.3-2 Timing Chart of the Underflow Operation



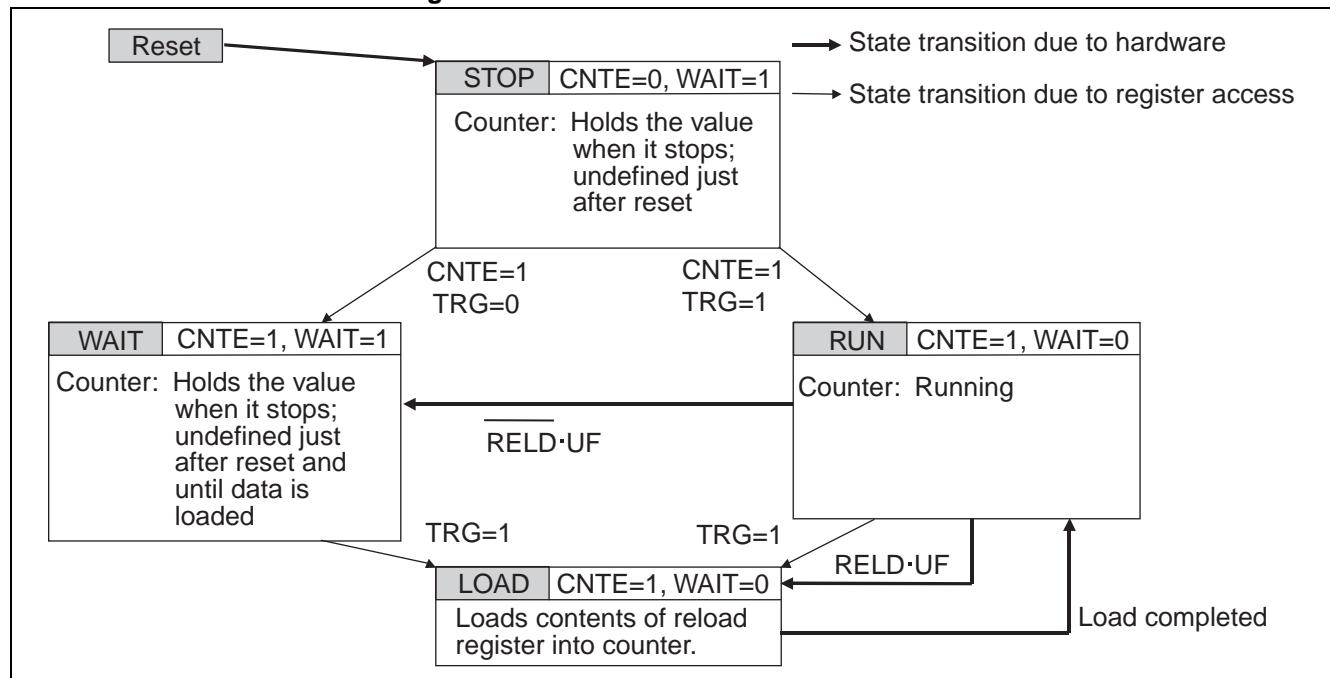
6.4 Operating States of the Counter

The counter state is determined by the CNTE bit of the control status register (TMCSR) and the WAIT signal, which is an internal signal. The states that can be set including the stop state, when CNTE=0 and WAIT=1 (STOP state); the startup trigger wait state, when CNTE=1 and WAIT=1 (WAIT status); and the operation state, when CNTE=1 and WAIT=0 (RUN state).

■ Operating States of the Counter

Figure 6.4-1 shows the state transitions.

Figure 6.4-1 Status Transitions of Counter



6.5 Precautions on Using the 16-bit Reload Timer

This section contains precautions on using the 16-bit reload timer.

■ Precautions on Using the 16-bit Reload Timer

○ Internal prescaler

The internal prescaler is enabled if a trigger (software or external trigger) is applied while bit1 (timer enable: CNTE) of the control status register (TMCSR) is set to "1".

○ Timing of setting and clearing the interrupt request flag

If the device attempts to set and clear the interrupt request flag at the same time, the flag is set and the clear operation becomes ineffective.

○ 16-bit reload register (TMRLR)

If the device attempts to write to the 16-bit reload register and reload the data into the 16-bit reload register at the same time, old data is loaded into the counter. New data is loaded into the counter only in the next reload timing.

○ 16-bit timer register (TMR)

If the device attempts to load and count the 16-bit timer register at the same time, the load (reload) operation takes precedence.

CHAPTER 7 PPG TIMER

This chapter describes the PPG timer, register configurations and functions, and PPG timer operation. The chapter also provides a block diagram of the PPG timer.

- 7.1 Overview of PPG Timer
- 7.2 Block Diagram of PPG Timer
- 7.3 Registers of PPG Timer
- 7.4 PPG Operation
- 7.5 One-shot Operation
- 7.6 PPG Timer Interrupt Source and Timing Chart
- 7.7 Activating Multiple Channels by Using the General Control Register
- 7.8 Notes on Use of the PPG Timer

7.1 Overview of PPG Timer

The PPG timer can generate PWM wave forms with great precision and efficiency.
The MB91301 series has four built-in channels for the PPG timers.

■ Features of PPG Timer

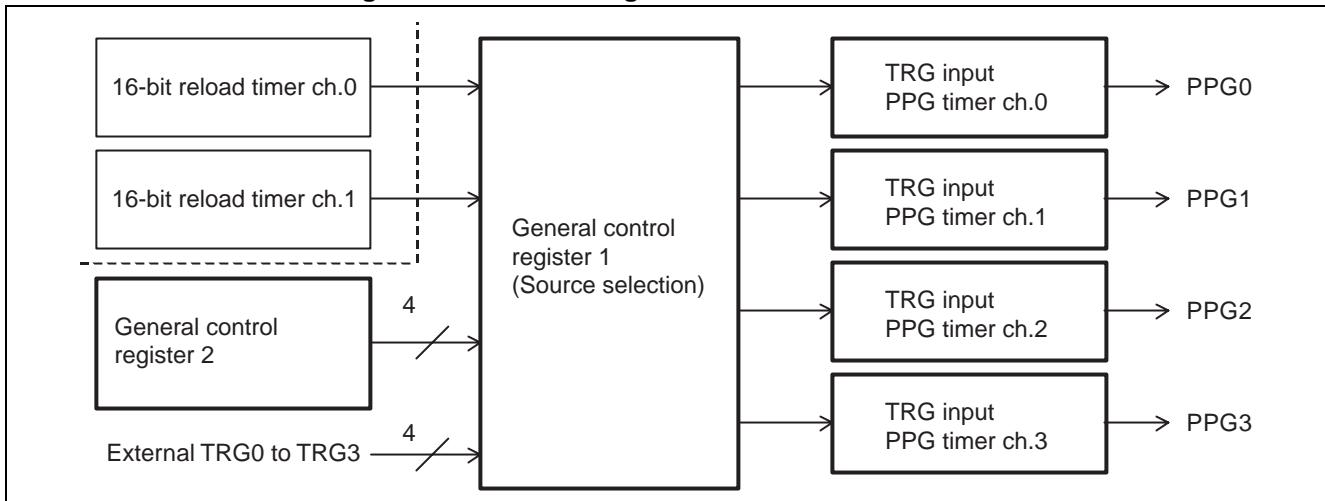
- Each channel consists of the following elements:
 - 16-bit down counter
 - 16-bit data register with cycle setting buffer
 - 16-bit compare register with duty setting buffer
 - Pin controller
- One of the following can be selected for the 16-bit down counter clock:
 - Internal clock: ϕ
 - Internal clock: $\phi/4$
 - Internal clock: $\phi/16$
 - Internal clock: $\phi/64$
- The counter value can be initialized to $FFFF_H$ by using reset and counter borrows.
- Each channel has a PPG output.
- Register
 - Cycle set register: reload data register with buffer
 - Duty set register: compare register with buffer
 - Transfer from buffers is performed by using counter borrows.
- Pin control overview
 - When a duty ratio match occurs, the counter value is set to "1". (Preferred)
 - When a counter borrow occurs, the counter value is reset to "0".
 - By using output value fix mode, all-low (or all-high) can be output easily.
 - In addition, the polarity can be specified.
- An interrupt request can be generated by the following sources. Interrupt requests can be used to start DMA transfer.
 - Start of PPG timer
 - Counter borrow (cycle match)
 - Duty cycle match
 - Counter borrow (cycle match) or duty ratio match
- Software or other interval timers can be used to specify that multiple channels are activated at the same time. In addition, restart during operation can be specified.
- Detected request level can be selected from "rising edge", "falling edge" and "both edges".

7.2 Block Diagram of PPG Timer

Figure 7.2-1 shows the block diagram of an entire PPG timer. Figure 7.2-2 shows the block diagram of one channel of the PPG timer.

■ Block Diagram of the Entire PPG Timer

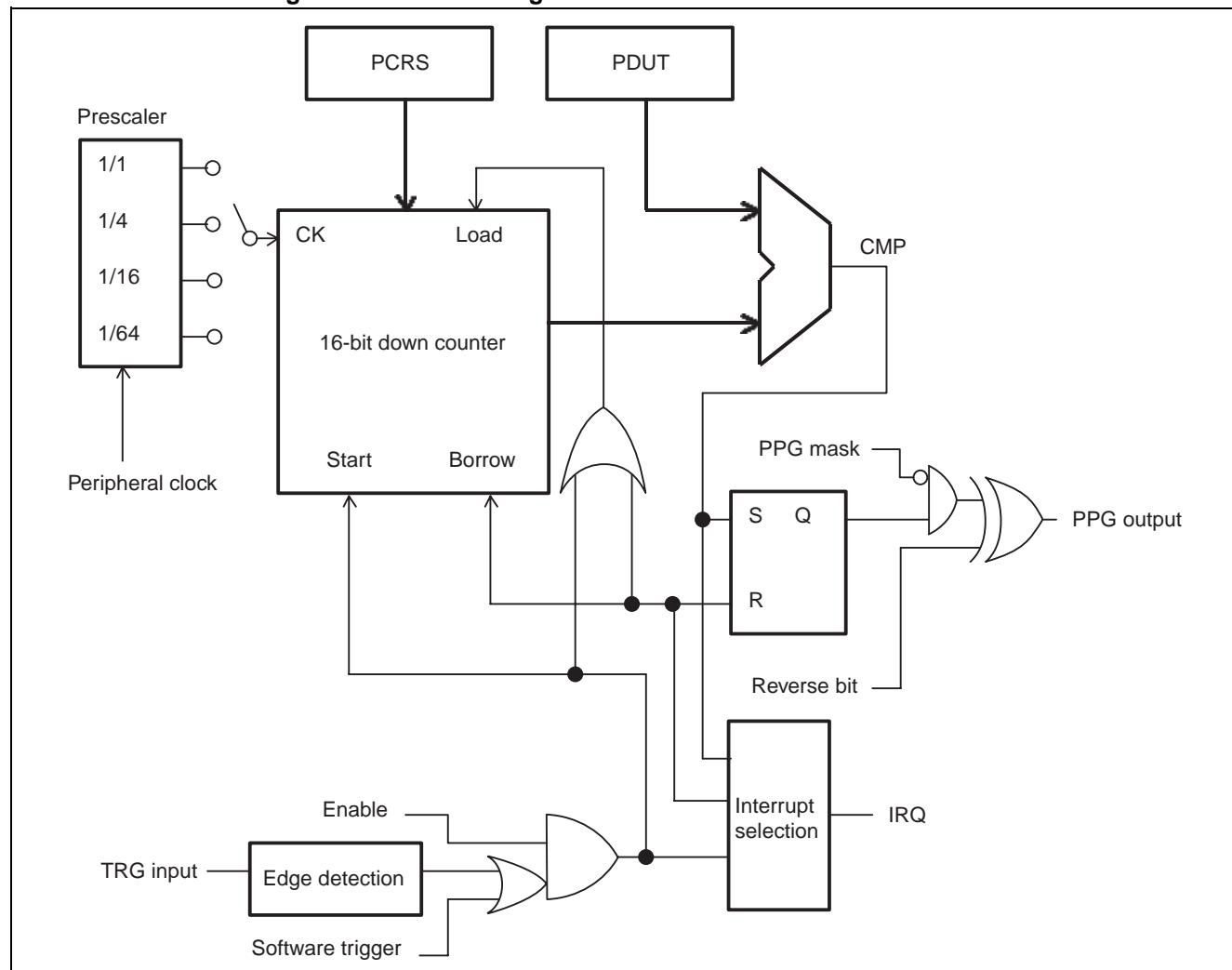
Figure 7.2-1 Block Diagram of the Entire PPG Timer



CHAPTER 7 PPG TIMER

■ Block Diagram of One Channel of the PPG Timer

Figure 7.2-2 Block Diagram of One Channel of the PPG Timer



7.3 Registers of PPG Timer

Figure 7.3-1 lists the registers of the PPG timer.

■ Register List of PPG Timer

Figure 7.3-1 Register List of PPG Timer

bit 15	0	
	GCN10	General control register 10
	GCN20	General control register 20
	PTMR0	PPG Timer register (ch.0)
	PCSR0	PPG Cycle set register (ch.0)
	PDUT0	PPG Duty set register (ch.0)
PCNH0	PCNL0	Control status register (ch.0)
	PTMR1	PPG Timer register (ch.1)
	PCSR1	PPG Cycle set register (ch.1)
	PDUT1	PPG Duty set register (ch.1)
PCNH1	PCNL1	Control status register (ch.1)
	PTMR2	PPG Timer register (ch.2)
	PCSR2	PPG Cycle set register (ch.2)
	PDUT2	PPG Duty set register (ch.2)
PCNH2	PCNL2	Control status register (ch.2)
	PTMR3	PPG Timer register (ch.3)
	PCSR3	PPG Cycle set register (ch.3)
	PDUT3	PPG Duty set register (ch.3)
PCNH3	PCNL3	Control status register (ch.3)

7.3.1 Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0)

The control status register (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0) controls the PPG timer and indicates the status of the timer. Note that some bits cannot be rewritten while the PPG timer is operating.

■ Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0)

The bit configuration of the control status registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0) is shown in Figure 7.3-2.

**Figure 7.3-2 Bit Configuration of the Control Status Register
(PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0)**

PCNH	bit	15	14	13	12	11	10	9	8	Initial value	
	Address:	ch.0 000126H	CNTE	STGR	MDSE	RTRG	CKS1	CKS0	PGMS	-	00000000B
	ch.1 00012EH	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-		
	ch.2 000136H	○	○	×	×	×	×	○	-	←Rewriting during operation	
	ch.3 00013EH										
PCNL	bit	7	6	5	4	3	2	1	0		
	Address:	ch.0 000127H	EGS1	EGS0	IREN	IRQF	IRS1	IRS0	-	OSEL	000000X0B
	ch.1 00012FH	R/W	R/W	R/W	R/W	R/W	R/W	-	R/W		
	ch.2 000137H	×	×	○	○	×	×	-	×	←Rewriting during operation	
	ch.3 00013FH										

■ Bit Function of Control Status Registers (PCNH, PCNL)

The bit function of the control status registers (PCNH, PCNL) is shown below.

[bit15] CNTE: Timer enable bit

This bit enables operation of the 16-bit down counter.

Table 7.3-1 Timer enable setting

CNTE	Function
0	Stopped (initial value)
1	Enabled

[bit14] STGR: Software trigger bit

When this bit is written to "1", a software trigger is activated. Whenever this bit is read, a value of "0" is returned.

[bit13] MDSE: Mode selection bit

This bit determines whether the PPG operation in which pulses are generated continuously or the one-shot operation in which only single pulses are generated is used.

Table 7.3-2 Mode selection setting

MDSE	Function
0	PPG operation (initial value)
1	One-shot operation

[bit12] RTRG: Restart enable bit

This bit determines whether restart through a software trigger or trigger input is allowed.

Table 7.3-3 Restart enable setting

RTRG	Function
0	Restart disabled (initial value)
1	Restart enabled

[bit11, bit10] CKS1, CKS0: Counter clock selection bit

These bits select the counter clock of the 16-bit down counter.

Table 7.3-4 Counter clock selection setting

CKS1	CKS0	Cycle
0	0	ϕ (initial value)
0	1	$\phi/4$
1	0	$\phi/16$
1	1	$\phi/64$

ϕ : Peripheral machine clock

[bit9] PGMS: PPG output mask selection bit

When this bit is written to "1", the PPG output can be masked to "0" or "1" regardless of the mode setting, cycle setting, or duty ratio setting.

PPG output when PGMS is set to "1" is shown below.

Table 7.3-5 PPG Output when PGMS is Set to "1"

Polarity	PPG output
Normal polarity	"L" output
Reverse polarity	"H" output

For output of all-"H" for normal polarity (or all-"L" for reverse polarity), write the same value to the cycle set register and the duty set register to obtain the reverse output of these mask values.

[bit8] (Reserved)

This bit is unused bit.

[bit7, bit6] EGS1, EGS0: Trigger input edge selection bit

This bit selects the valid edge for the activation source selected by the general control register 1.

When the software trigger bit is set to "1", a software trigger is enabled regardless of the mode selected.

Table 7.3-6 Setting of trigger input edge selection

EGS1	EGS0	Edge selection
0	0	Disabled (initial value)
0	1	Rising edge
1	0	Falling edge
1	1	Both edges

[bit5] IREN: Interrupt request enable bit

This bit specifies whether to enable interrupt requests.

Table 7.3-7 Interrupt request setting

IREN	Function
0	Disabled (initial value)
1	Enabled

[bit4] IRQF: Interrupt request flag

When bit5 (IREN) is set to "Enabled", and the interrupt source specified by bit3 and bit2 (IRS1 and IRS0) occurs, this bit is set and an interrupt request is issued to the CPU. In addition, when activation of DMA transfer is specified, DMA transfer is started.

This bit is cleared when a value of "0" is written or the clear signal is received from the DMAC.

The value of this bit does not change even if there is an attempt to set it to "1" via a write operation.

When this bit is read by read-modify-write instructions, "1" is returned regardless of the bit value.

[bit3, bit2] IRS1, IRS0: Interrupt source selection bit

These bits select the interrupt source that sets bit4 (IRQF).

Table 7.3-8 Setting of Interrupt Source Selection

IRS1	IRS0	Interrupt source
0	0	Software trigger or trigger input (initial value)
0	1	Counter borrow (cycle match)
1	0	Duty match
1	1	Counter borrow (cycle match) or duty match

[bit1] (Reserved)

This bit is unused bit.

[bit0] OSEL: PPG output polarity specification bit

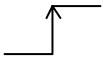
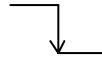
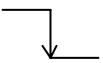
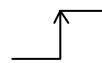
This bit specifies the polarity of the PPG output.

It becomes because of the combination with this bit and the PGMS bit of bit9 as shown in Table 7.3-9.

Table 7.3-9 PPG Output Polarity Specification Combination

PMGS	OSEL	PPG output
0	0	Normal polarity (initial value)
0	1	Reverse polarity
1	0	Fixed to "L" level
1	1	Fixed to "H" level

Table 7.3-10 PPG Output Polarity Specification

Polarity	After reset	Duty match	Counter borrow
Normal polarity	"L" output		
Reverse polarity	"H" output		

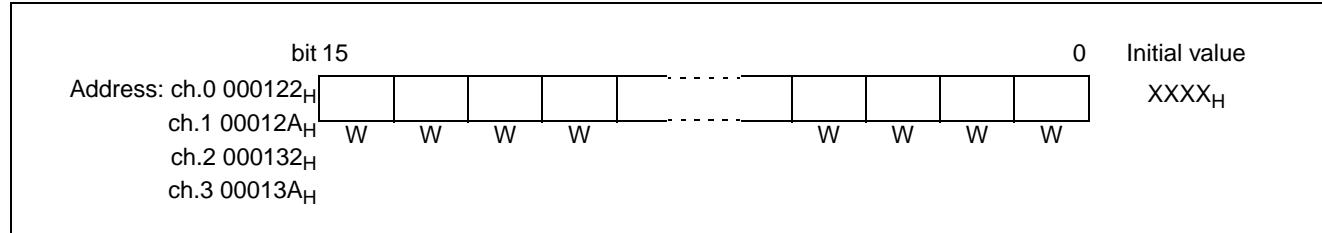
7.3.2 PPG Cycle Set Register (PCSR:PCSR3 to PCSR0)

The PCSR:PCSR3 to PCSR0 is a buffer register for setting cycles. It has a buffer. Transfers from the buffer are performed through counter borrows.

■ Bit Configuration of PPG Cycle Set Register (PCSR:PCSR3 to PCSR0)

The bit configuration of the PCSR:PCSR3 to PCSR0 is shown below.

Figure 7.3-3 Bit Configuration of the PPG Cycle Set Register (PCSR:PCSR3 to PCSR0)



After initializing or rewriting the PCSR, write to the duty set register.

This register must be accessed in 16-bit data or 32-bit data.

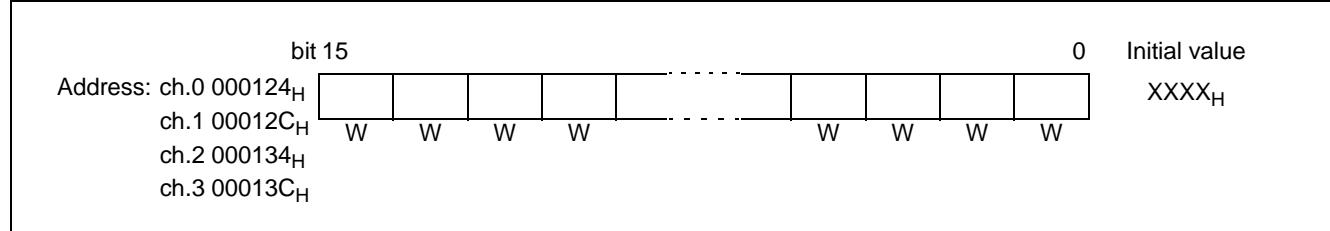
7.3.3 PPG Duty Set Register (PDUT:PDUT3 to PDUT0)

The PDUT:PDUT3 to PDUT0 is a buffer register for setting duties. It has a buffer. Transfers from the buffer are performed through counter borrows.

■ Bit Configuration of PPG Duty Set Register (PDUT:PDUT3 to PDUT0)

The bit configuration of the PDUT:PDUT3 to PDUT0 is shown below.

Figure 7.3-4 Bit Configuration of PPG Duty Set Register (PDUT:PDUT3 to PDUT0)



If the same value is written to the PCSR and PDUT, all-"H" is output for normal polarity and all-"L" is output for reverse polarity.

Do not set values so that the condition PCSR < PDUT would be met. Otherwise, the PPG output becomes undefined.

This register must be accessed in 16-bit data or 32-bit data.

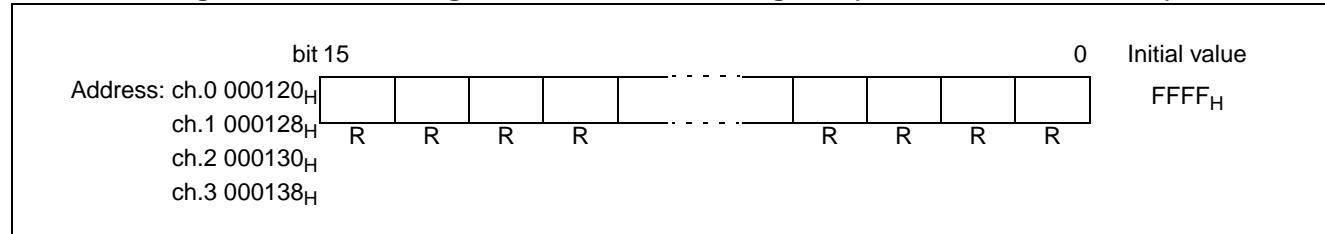
7.3.4 PPG Timer Register (PTMR:PTMR3 to PTMR0)

The PTMR:PTMR3 to PTMR0 can be used to read the 16-bit down counter.

■ Bit Configuration of PPG Timer Register (PTMR:PTMR3 to PTMR0)

The bit configuration of the PTMR (PTMR:PTMR3 to PTMR0) is shown below.

Figure 7.3-5 Bit Configuration of PPG Timer Register (PTMR:PTMR3 to PTMR0)



This register must be accessed in 16-bit data.

7.3.5 General Control Register 10 (GCN10)

The GCN10 selects the source of the PPG timer trigger input.

■ Bit Configuration of General Control Register 10 (GCN10)

The bit configuration of the GCN10 is shown below.

Figure 7.3-6 Bit Configuration of General Control Register 10 (GCN10)

bit	15	14	13	12	11	10	9	8	
Address: 000118H	TSEL[33:30]				TSEL[23:20]				
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	←Attribute
	0	0	1	1	0	0	1	0	←Initial value
bit	7	6	5	4	3	2	1	0	
000119H	TSEL[13:10]				TSEL[03:00]				
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	←Attribute
	0	0	0	1	0	0	0	0	←Initial value

■ Details of General Control Register 10 (GCN10)

[bit15 to bit12] TSEL33 to TSEL30: ch.3 trigger input selection bit

These bits are ch.3 trigger input select bits.

Table 7.3-11 Ch.3 Trigger Input Selection

TSEL33 to TSEL30				Function
0	0	0	0	EN0 bit of GCN2
0	0	0	1	EN1 bit of GCN2
0	0	1	0	EN2 bit of GCN2
0	0	1	1	EN3 bit of GCN2 (initial value)
0	1	0	0	16-bit reload timer ch.0
0	1	0	1	16-bit reload timer ch.1
0	1	1	X	Setting prohibited
1	0	0	0	External TRG0
1	0	0	1	External TRG1
1	0	1	0	External TRG2
1	0	1	1	External TRG3
1	1	X	X	Setting prohibited

[bit11 to bit8] TSEL23 to TSEL20: ch.2 trigger input selection bit

These bits are ch.2 trigger input select bits.

Table 7.3-12 Ch.2 Trigger Input Selection

TSEL23 to TSEL20				Function
0	0	0	0	EN0 bit of GCN2
0	0	0	1	EN1 bit of GCN2
0	0	1	0	EN2 bit of GCN2 (initial value)
0	0	1	1	EN3 bit of GCN2
0	1	0	0	16-bit reload timer ch.0
0	1	0	1	16-bit reload timer ch.1
0	1	1	X	Setting prohibited
1	0	0	0	External TRG0
1	0	0	1	External TRG1
1	0	1	0	External TRG2
1	0	1	1	External TRG3
1	1	X	X	Setting prohibited

[bit7 to bit4] TSEL13 to TSEL10: ch.1 trigger input selection bit

These bits are ch.1 trigger input select bits.

Table 7.3-13 Ch.1 Trigger Input Selection

TSEL13 to TSEL10				Function
0	0	0	0	EN0 bit of GCN2
0	0	0	1	EN1 bit of GCN2 (initial value)
0	0	1	0	EN2 bit of GCN2
0	0	1	1	EN3 bit of GCN2
0	1	0	0	16-bit reload timer ch.0
0	1	0	1	16-bit reload timer ch.1
0	1	1	X	Setting prohibited
1	0	0	0	External TRG0
1	0	0	1	External TRG1
1	0	1	0	External TRG2
1	0	1	1	External TRG3
1	1	X	X	Setting prohibited

[bit3 to bit0] TSEL03 to TSEL00: ch.0 trigger input selection bit

These bits are ch.0 trigger input select bits.

Table 7.3-14 Ch.0 Trigger Input Selection

TSEL03 to SEL00				ch.0 trigger input
0	0	0	0	EN0 bit of GCN2 (initial value)
0	0	0	1	EN1 bit of GCN2
0	0	1	0	EN2 bit of GCN2
0	0	1	1	EN3 bit of GCN2
0	1	0	0	16-bit reload timer ch.0
0	1	0	1	16-bit reload timer ch.1
0	1	1	X	Setting prohibited
1	0	0	0	External TRG0
1	0	0	1	External TRG1
1	0	1	0	External TRG2
1	0	1	1	External TRG3
1	1	X	X	Setting prohibited

7.3.6 General Control Register 20 (GCN20)

The GCN20 activates a start trigger through software.

■ Bit Configuration of General Control Register 20 (GCN20)

The bit configuration of the GCN20 is shown below.

Figure 7.3-7 Bit Configuration of General Control Register 20 (GCN20)

bit	7	6	5	4	3	2	1	0	
Address: 00011B _H	-	-	-	-	EN3	EN2	EN1	EN0	
	R/W 0	←Attribute ←Initial value							

When one of the EN-bits of this register is selected by the GCN10, the register value is passed to the trigger input of the PPG timer.

The PPG timers of multiple channels can be activated at the same time by generating the edge selected by the EGS1 and EGS0 bits of the control status register (PCNH, PCNL) via software.

Bit7 to bit4 of this register must be set to "0".

Bit7 to bit0 of address 00011A_H must be set to "0".

7.4 PPG Operation

The PPG operation allows continuous pulses to be output after a start trigger is detected. The cycle and duty ratio of the output pulses can be controlled by changing the values of the PCSR and PDUT, respectively.

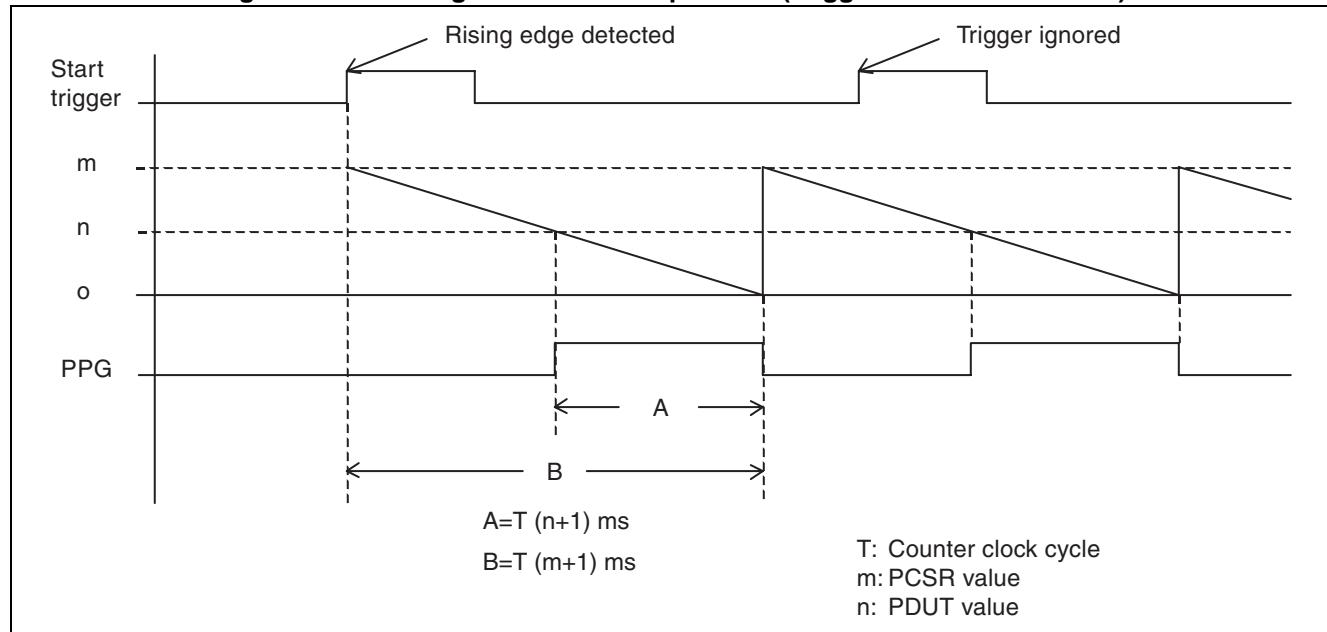
After data is written to PCSR, be sure to write to PDUT.

■ PPG Operation

- When restart is inhibited

Figure 7.4-1 shows a timing chart of the PPG operation when trigger restart is inhibited.

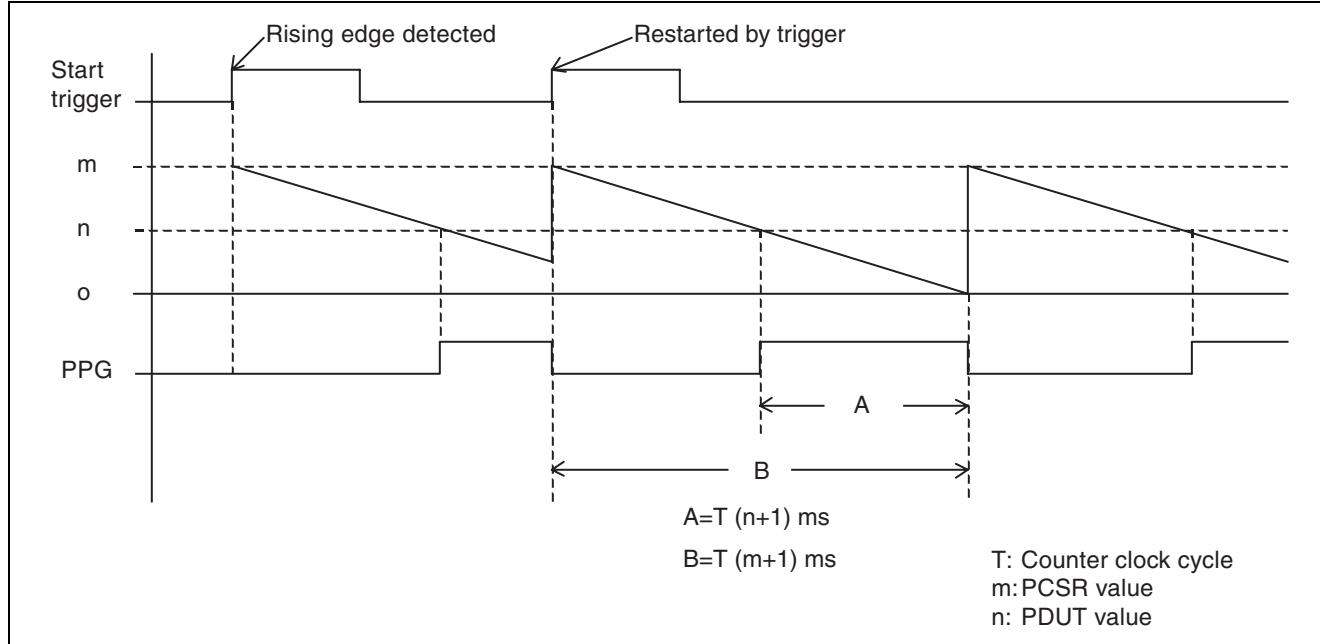
Figure 7.4-1 Timing Chart of PPG Operation (Trigger Restart Prohibited)



○ When restart is enabled

Figure 7.4-2 shows the timing chart of the PPG operation when trigger restart is enabled.

Figure 7.4-2 Timing Chart of PPG Operation (Trigger Restart Enabled)



7.5 One-shot Operation

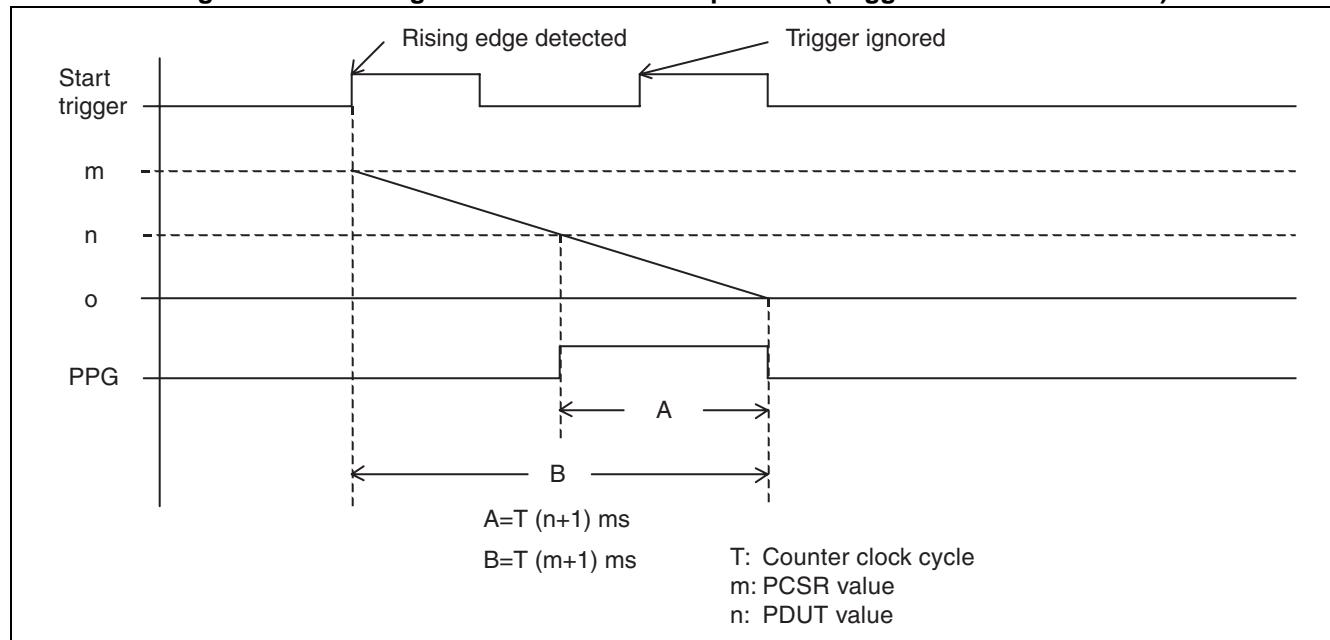
The one-shot operation allows output of a single pulse of any width through a trigger. If restart is enabled, the counter value is reloaded when the edge is detected during operation.

■ One-shot Operation

○ When restart is inhibited

Figure 7.5-1 shows the timing chart of a one-shot operation when a trigger restart is inhibited.

Figure 7.5-1 Timing Chart of a One-shot Operation (Trigger Restart Prohibited)

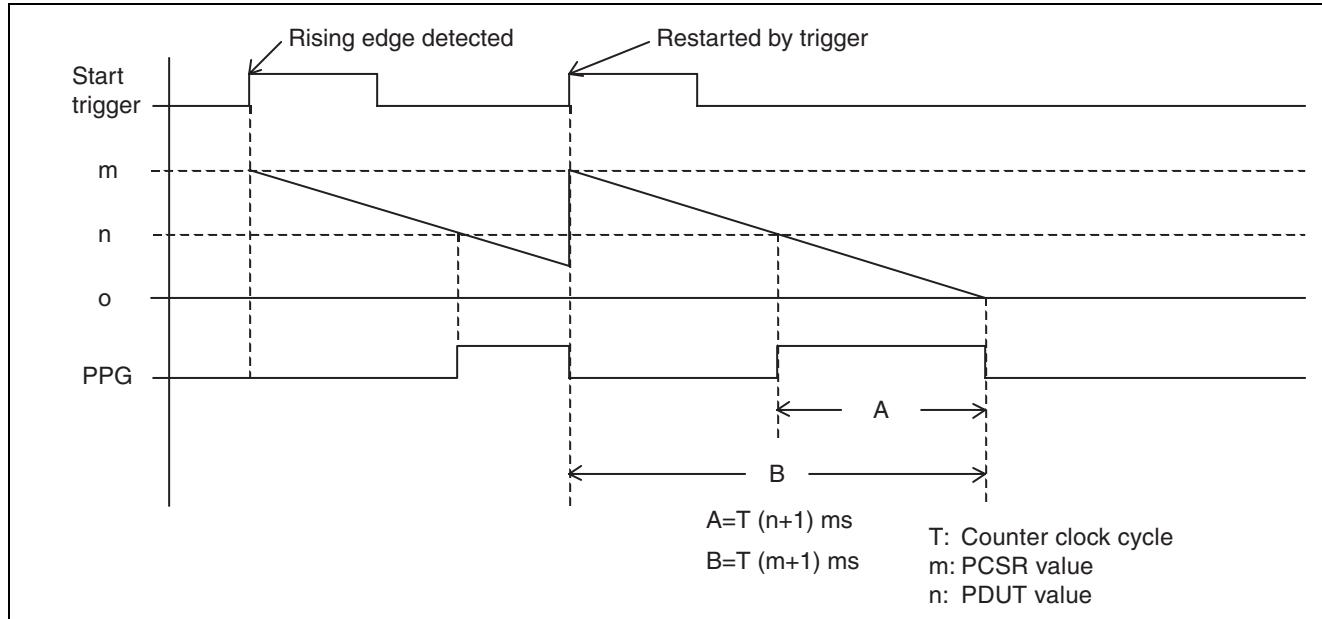


CHAPTER 7 PPG TIMER

○ When restart is enabled

Figure 7.5-2 shows the timing chart of a one-shot operation when a trigger restart is enabled.

Figure 7.5-2 Timing Chart of One-shot Operation (Trigger Restart Enabled)



7.6 PPG Timer Interrupt Source and Timing Chart

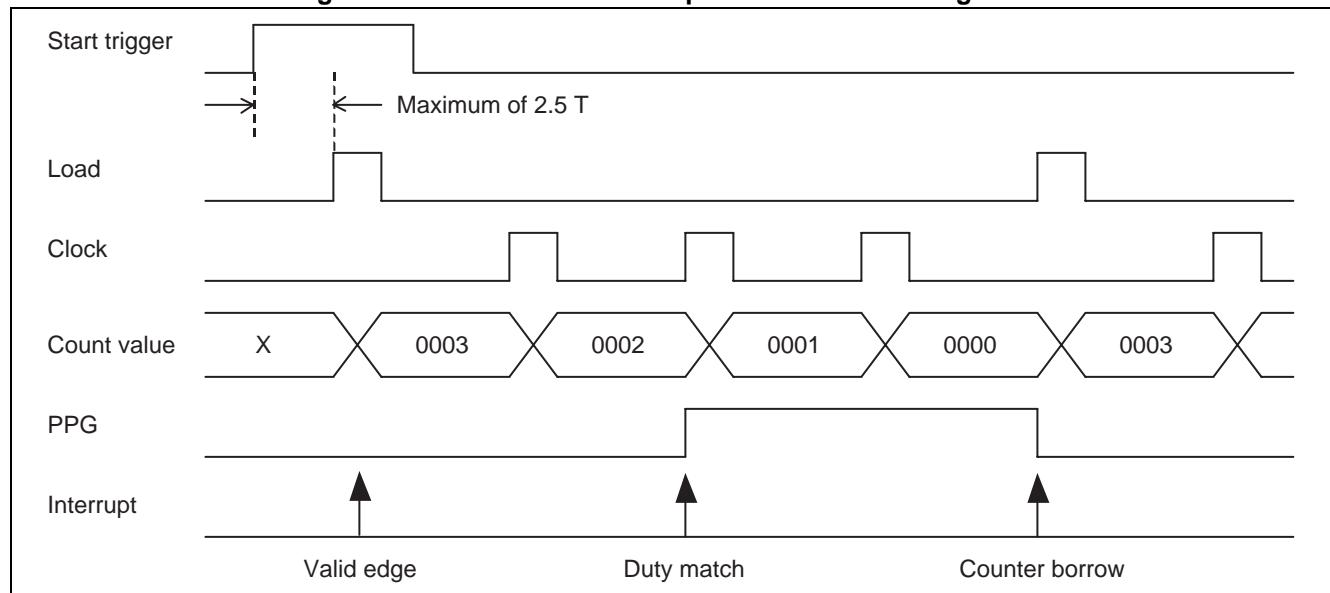
This section describes interrupt sources and provides the related timing charts.

■ Interrupt Sources and Timing Chart (PPG Output: Normal Polarity)

Figure 7.6-1 shows the PPG timer interrupt sources and a timing chart.

A maximum time of 2.5 T (T: counter clock cycle) is required from when a start trigger is activated to when the counter value is loaded.

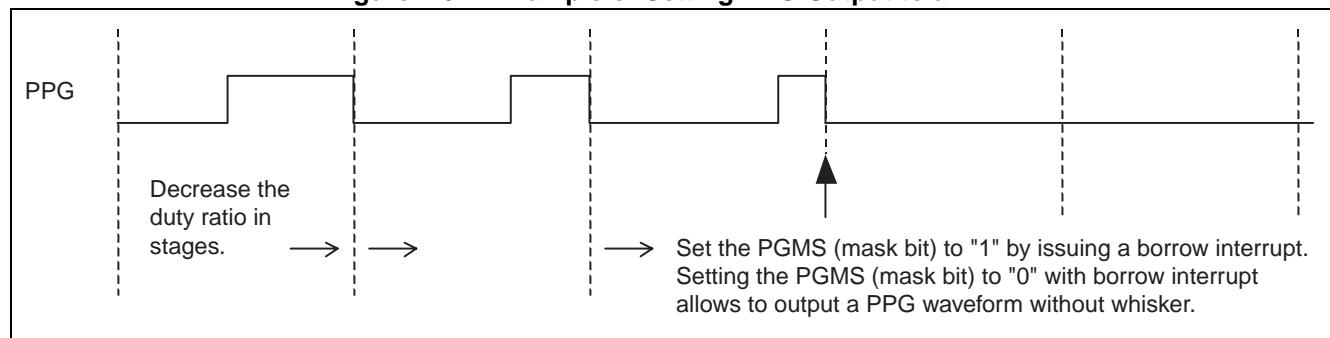
Figure 7.6-1 PPG Timer Interrupt Sources and Timing Chart



■ Examples for Setting PPG Output to all-"L" or all-"H"

Figure 7.6-2 shows how to set the PPG output to all-"L".

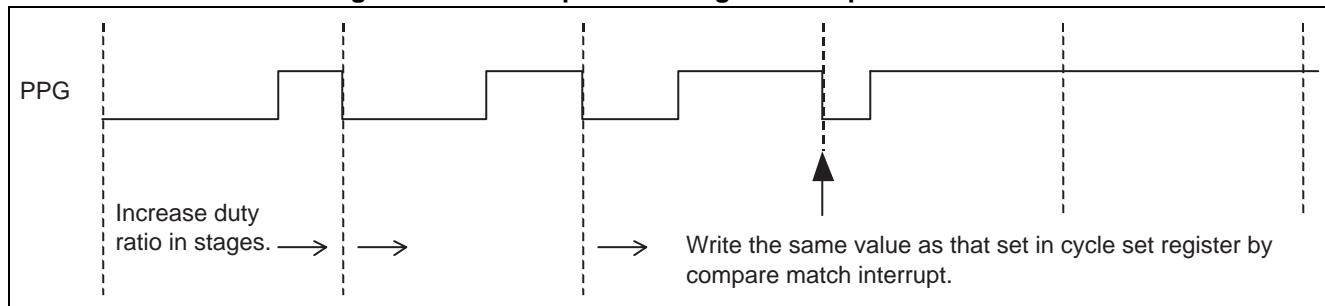
Figure 7.6-2 Example of Setting PPG Output to all-"L"



CHAPTER 7 PPG TIMER

Figure 7.6-3 shows an example of setting PPG output to all-"H".

Figure 7.6-3 Example of Setting PPG Output to all-"H"



7.7 Activating Multiple Channels by Using the General Control Register

You can activate multiple channels at the same time by selecting the start trigger with the GCN10.

This section shows an example of how GCN20 is set to activate channels via software.

■ Activating Multiple Channels with the GCN

[Setting procedure]

- 1) Set the cycle in the PCSR.
 - 2) Set the duty ratio in the PDUT.
-

Note:

The setting must follow the order of PCSR followed by PDUT.

- 3) Specify the trigger input source for the channel to be activated with GCN10.

In this case, the initial setting is kept because GCN20 is used.

(ch.0 --> EN0, ch.1 --> EN1, ch.2 --> EN2, ch.3 --> EN3)

- 4) Set the control status register for the channel to be activated.

- CNTE: 1 --> Enables timer operation.
 - STGR: 0 --> Since the channel is activated by GCN20, this bit is not set.
 - MDSE: 0 --> Selects PPG operation.
 - RTRG: 0 --> Inhibits restart.
 - CSK1, CSK0:00 --> Sets the counter clock to ϕ .
 - PGMS: 0 --> Does not mask PPG output.
 - Bits 8:0 --> Any value can be set because these bits are unused.
 - EGS1, EGS0:01 --> Activates channel at a rising edge
 - IREN: 1 --> Enables interrupt request.
 - IRQF: 0 --> Clears interrupt source.
 - IRS1, IRS0:01 --> Issues interrupt request when counter borrow occurs.
 - PPE0 to PPE3: 1 --> Enables PPG output. (setting of port function register)
 - OSEL: 0 --> Sets normal polarity.
- 5) Activate a start trigger by writing data to GCN20.

To activate ch.0 and ch.1 at the same time with the above settings, set the EN0 and EN1 bits of GCN20 to "1". A rising edge is generated and pulses are output from PPG0 and PPG1.

■ When the 16-bit Reload Timer is Used for Activation

Specify the 16-bit reload timer as a source in GCN10 (see 3) above). Start the 16-bit reload timer instead of writing data to GCN20 in 5) above.

In addition, set the control status register as follows:

- RTRG: 1 --> Enables restart.
- EGS1, EGS0:11 --> Enables activation by both edges

By setting 16-bit reload timer output to toggle mode, the PPG timer can be restarted at fixed intervals.

7.8 Notes on Use of the PPG Timer

This section gives notes on using the PPG timer.

■ Precautions when Using

- If the interrupt request flag set timing and clear timing are simultaneous, the flag setting operation overrides the flag clearing operation.
- The values in bit11 and bit10 in the PPG control register (the CKS1 and CKS0 count lock select bits) are reflected as soon as they are written. Stop the PPG timer counting when updating their setting.
- The PPG down counter (PPGC: 16-bit down counter) prefers loading to counting if they are wanted simultaneously.

CHAPTER 7 PPG TIMER

CHAPTER 8 U-TIMER

This chapter describes the U-TIMER, the configuration and functions of registers, and U-TIMER operation.

8.1 Overview of the U-TIMER

8.2 U-TIMER Registers

8.3 U-TIMER Operation

8.1 Overview of the U-TIMER

This section provides an overview and a block diagram of the U-TIMER (16-bit timer for UART baud rate generation).

■ Overview of the U-TIMER

The U-TIMER is a 16-bit timer used to generate the baud rate for the UART. Use a combination of a chip operating frequency and a reload value of the U-TIMER to specify a baud rate.

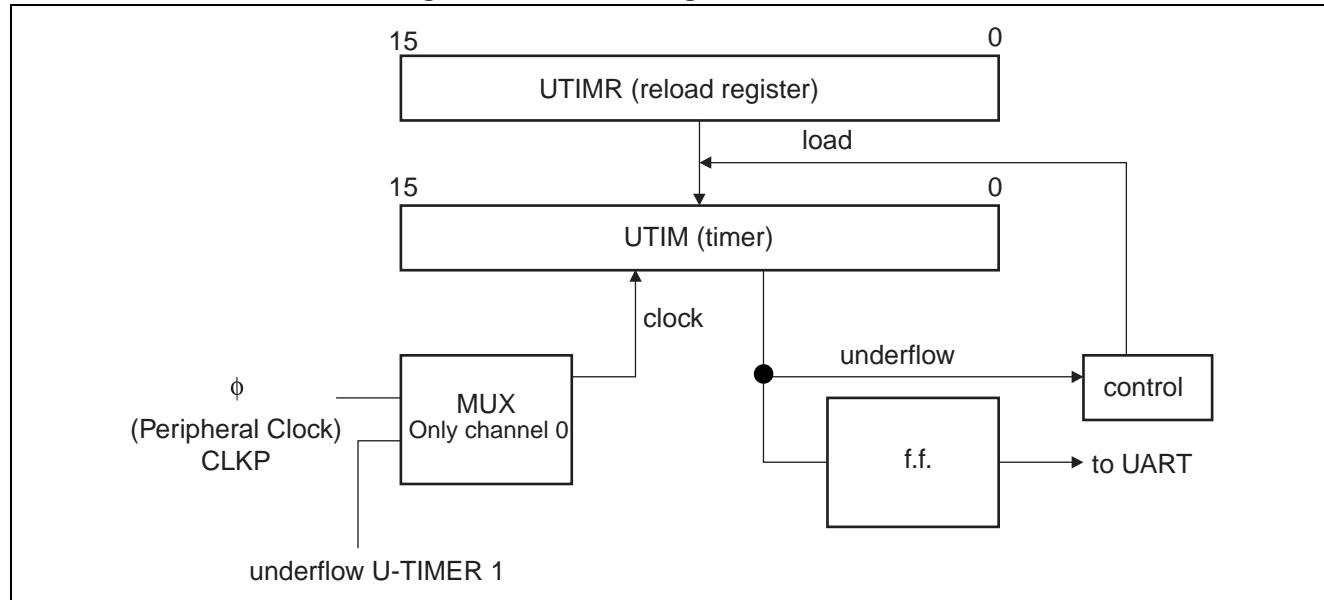
The U-TIMER, which generates an interrupt upon a counter underflow, can be used as an interval timer.

The MB91301 series has three built-in U-TIMER channels. When used as an interval timer, two sets of U-TIMERS can be cascaded to count a maximum interval of $2^{32} \times \phi$. Only the combinations of Channels 0 and 1 and Channels 0 and 2 can be connected in cascade fashion.

■ Block Diagram

Figure 8.1-1 shows the block diagram of U-TIMER.

Figure 8.1-1 Block Diagram of the U-TIMER



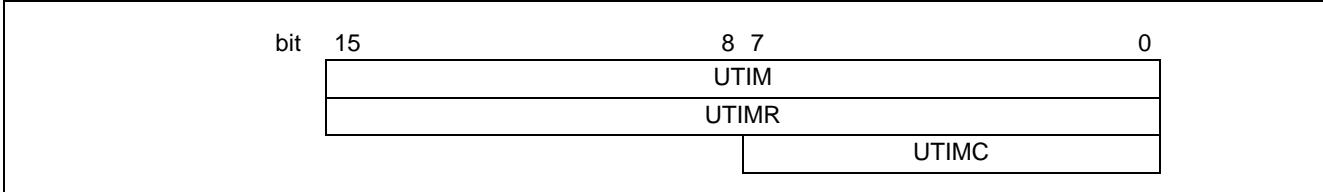
8.2 U-TIMER Registers

This section describes the configuration and functions of the registers used by the U-TIMER.

■ U-TIMER Registers

Figure 8.2-1 shows the registers used by the U-TIMER.

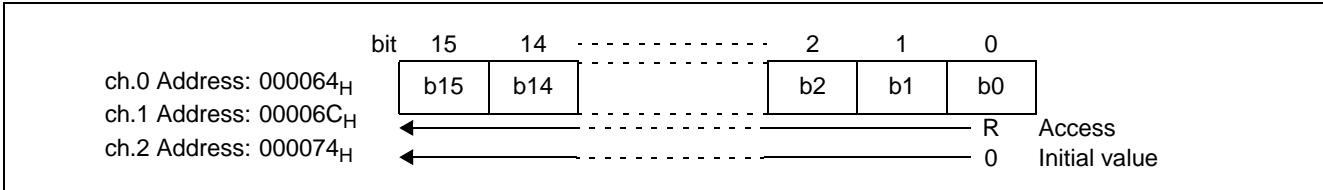
Figure 8.2-1 U-TIMER Registers



■ U-TIMER (UTIM:UTIM2 to UTIM0)

Figure 8.2-2 shows the bit configuration of the U-TIMER (UTIM:UTIM2 to UTIM0).

Figure 8.2-2 Bit Configuration of the U-TIMER (UTIM:UTIM2 to UTIM0)

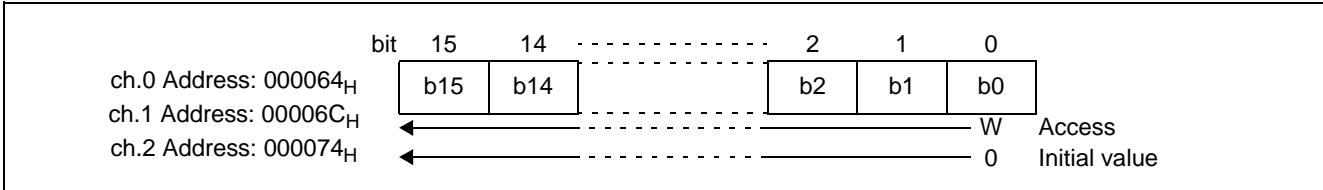


UTIM indicates the timer value. Use a 16-bit transfer instruction to access this register.

■ Reload Register (UTIMR:UTIMR2 to UTIMR0)

Figure 8.2-3 shows the bit configuration of the reload register (UTIMR:UTIMR2 to UTIMR0).

Figure 8.2-3 Bit Configuration of the Reload Register (UTIMR:UTIMR2 to UTIMR0)



UTIMR is a register that stores the value to be reloaded into UTIM if UTIM underflows.

Use a 16-bit transfer instruction to access this register.

Note:

When U-TIMER is used as a baud rate in UART mode 2 (CLK synchronous mode), setting "0" to UTIMR is prohibited.

CHAPTER 8 U-TIMER

■ U-TIMER Control Register (UTIMC:UTIMC2 to UTIMC0)

Figure 8.2-4 shows the bit configuration of the U-TIMER control register (UTIMC:UTIMC2 to UTIMC0).

Figure 8.2-4 Bit Configuration of the U-TIMER Control Register (UTIMC:UTIMC2 to UTIMC0)

bit	7	6	5	4	3	2	1	0	
ch.0 Address: 000067H	UCC1	-	-	UTIE	UNDR	CLKS	UTST	UTCR	
ch.1 Address: 00006FH	R/W	-	-	R/W	R/W	R/W	R/W	R/W	Access
ch.2 Address: 000077H	0	-	-	0	0	0	0	1	Initial value

UTIMC controls the operation of the U-TIMER.

Access with byte transfer instruction.

■ Bit Details of U-TIMER Control Register (UTIMC)

The following describes the functions of the U-TIMER control register (UTIMC) bits.

[bit7] UCC1 (U-timer Count Control 1): Control for counting method

This bit controls the U-TIMER counting method.

Table 8.2-1 Counting Method Control

UCC1	Operation
0	Normal operation $\alpha=2n+2$ [initial value]
1	+1 mode $\alpha=2n+3$

n is the setting value of UTIMR.

α is the cycle of the output clock for UART.

The U-TIMER can set a normal cycle, $2(n+1)$ as well as an odd-numbered division for the UART.

Set UCC1 to "1" to generate a cycle of $2n+3$.

Examples:

1. UTIMR=5, UCC1=0 --> Generation cycle = $2n+2= 12$ cycles
2. UTIMR=25, UCC1=1 --> Generation cycle = $2n+3= 53$ cycles
3. UTIMR=60, UCC1=0 --> Generation cycle = $2n+2=122$ cycles

Set UCC1, UCC0 to use the U-TIMER as the interval timer.

[bit6, bit5] (Reserved)

These bits are reserved.

[bit4] UTIE (U-TIMER Interrupt Enable): Interrupt enable by underflow

This bit is the interrupt enable bit for a U-TIMER underflow.

Table 8.2-2 Interrupt enable by underflow

UTIE	Operation
0	Interrupt disabled [initial value]
1	Interrupt enabled

[bit3] UNDR (UNDeR flow flag): Indicates generating underflow

This bit indicates that an underflow has occurred.

If the UNDR bit is set while the UTIE bit of bit4 is set to "1", an underflow interrupt occurs. The UNDR bit is cleared upon a reset or if "0" is written to it.

For a read by a read-modify-write instruction, "1" is always read.

Writing "1" to the UNDR has no effect.

[bit2] CLKS (clock select): Cascade specification

This bit is the cascade specification bit for ch.0 and ch.1 of the U-TIMER.

Table 8.2-3 Cascade specification

CLKS	Operation
0	Uses a peripheral clock (ϕ) as the clock source. [initial value]
1	Uses an underflow signal of Channel 0 as the U-TIMER source clock timing. *

*: f.f. shown in the block diagram

CLKS is valid only for ch.1 and ch.2. This bit must always be set to "0" for Channel 0.

ϕ (Peripheral clock = CLKP) has a different cycle depending on the gear setting.

[bit1] UTST (U-TIMER STart): Operation enable

This bit is the U-TIMER operation enable bit.

Table 8.2-4 Operation Enable

UTST	Operation
0	Stopped. Writing "0" during operation stops running of the U-TIMER. [initial value]
1	Operated. Writing "1" during operation does not stop the U-TIMER.

[bit0] UTCR (U-TIMER CleaR)

Writing "0" to UTCR clears the U-TIMER to 0000H (also clears the f.f. to "0").

The read value is always "1".

■ Precautions on the U-TIMER Control Register (UTIMC)

- In the stop state, assert the start bit UTST (started) to automatically reload data.
- In the stop state, assert both the clear bit UTCR and the start bit UTST at the same time to clear the counter to "0" and generate an underflow in the count-down immediately after the counter is cleared.
- During operation, the clear bit UTCR is asserted to clear the counter to "0". As a result, a short, whisker-like pulse may be output in the output waveform, possibly causing the UART or U-TIMER on the master side in cascade mode to malfunction. While the output clock is being used, do not clear it using the clear bit.
- In cascade mode, setting the slave-side UTIMR (reload register) to "0" or "1" causes the count to be performed incorrectly.
- In the timer stop state, assert both bit1 (U-TIMER start bit: UTST) and bit0 (U-TIMER clear bit: UTCR) of the U-TIMER control register at the same time to set bit3 (underflow flag: UNDR) of this register when the counter is loaded after it has been cleared. At this timing, the internal baud rate clock is set to "H" level.
- If the device attempts to set and clear the interrupt request flag at the same time, the flag is set and the clear operation becomes ineffective.
- If you select not to use ch.0 in cascade mode or use this module only as the timer function, always write "0" to bit2 (Reference clock selection bit: CLKS). Additionally, change the setting of the CLKS bit when this module has stopped.
- If the device attempts to write to and reload the data into the U-TIMER reload register at the same time, old data is loaded into the counter. New data is loaded into the counter only in the next reload timing.
- If the device attempts to clear and load U-TIMER at the same time, the timer clear operation takes precedence.

8.3 U-TIMER Operation

This section describes calculation of a baud rate for the U-TIMER and the timing in cascade mode.

■ Calculation of Baud Rate

The UART uses the underflow flip-flop (f.f. in the block diagram) of the corresponding U-TIMER (from U-TIMER0 to UART0 or from U-TIMER1 to UART1 or from U-TIMER2 to UART2) as the clock source for baud rates.

○ Asynchronous (start-stop synchronization) mode

The UART uses the U-TIMER output divided by 16.

[If UCC1=0]

$$\text{bps} = \frac{\phi}{(2n+2) \times 16}$$

n : UTIMR (reload value)

ϕ : Peripheral machine clock frequency

[If UCC1=1]

$$\text{bps} = \frac{\phi}{(2n+3) \times 16} \quad (\text{Varies depending on the gear})$$

Maximum bps 34 MHz 531250 bps, 68 MHz 1062500 bps

○ CLK synchronous mode

[If UCC1=0]

$$\text{bps} = \frac{\phi}{(2n+2)}$$

n: UTIMR (reload value)

ϕ : Peripheral machine clock frequency

[If UCC1=1]

$$\text{bps} = \frac{\phi}{(2n+3)} \quad (\text{Varies depending on the gear})$$

Maximum bps 34 MHz 8500000 bps, 68 MHz 17000000 bps

Note:

When U-TIMER is used as a baud rate in UART mode 2 (CLK synchronous mode), setting "0" to UTIMR is prohibited.

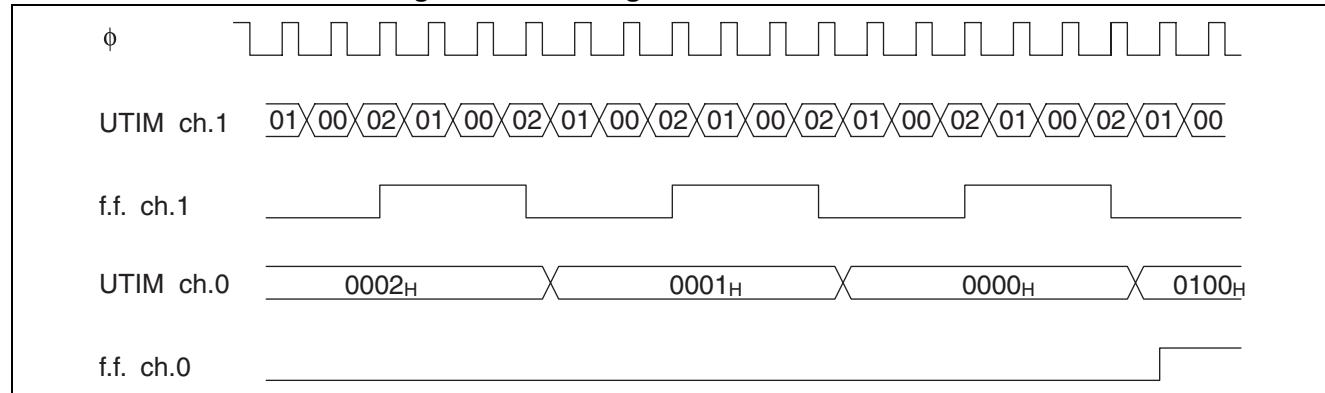
CHAPTER 8 U-TIMER

■ Cascade Mode

Channels 0 and 1 of the U-TIMER can be used in cascade mode.

Figure 8.3-1 shows a sample timing chart for when UTIMR ch.0 is set to "0100_H" and UTIMR ch.1 is set to "0002_H".

Figure 8.3-1 Timing Chart for Cascade Mode



CHAPTER 9 EXTERNAL INTERRUPT AND NMI CONTROLLER

This chapter describes the overview, the configuration and functions of registers, and operation of the external interrupt and NMI controller.

- 9.1 Overview of the External Interrupt and NMI Controller
- 9.2 External Interrupt and NMI Controller Registers
- 9.3 Operation of the External Interrupt and NMI Controller

9.1 Overview of the External Interrupt and NMI Controller

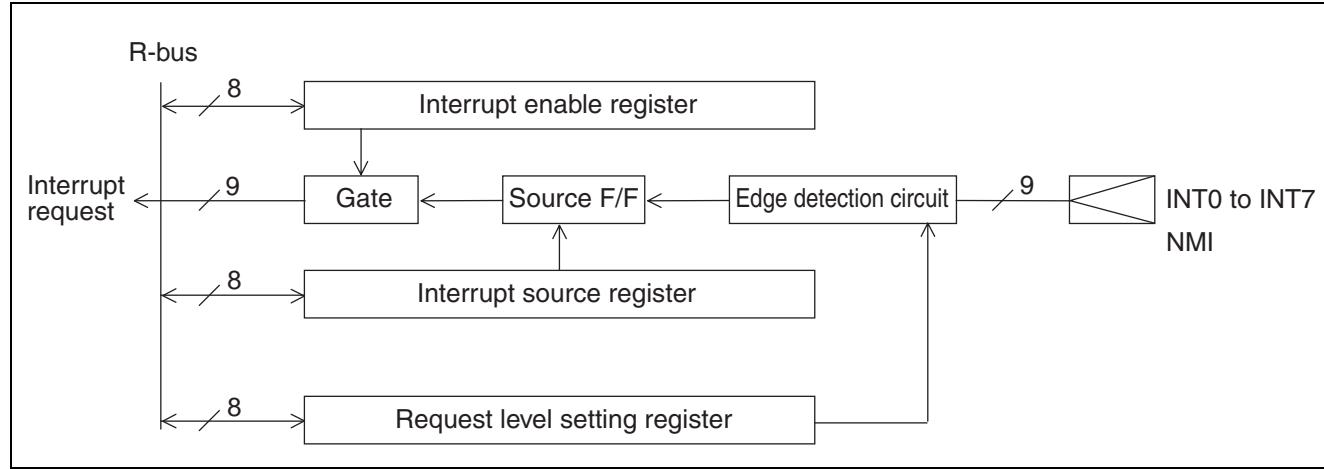
The external interrupt controller is a block that controls external interrupt requests input to NMI and INT0 to INT7.

"H" level, "L" level, rising edge, or falling edge can be selected as the level of a request to be detected (except for NMI).

■ Block Diagram of the External Interrupt and NMI Controller

Figure 9.1-1 shows a block diagram of the external interrupt and NMI controller.

Figure 9.1-1 Block Diagram of the External Interrupt and NMI Controller



9.2 External Interrupt and NMI Controller Registers

This section describes the configuration and functions of the registers used by the external interrupt and NMI controller.

■ External Interrupt and NMI Controller Registers

Figure 9.2-1 shows the registers used by the external interrupt and NMI controller.

Figure 9.2-1 External Interrupt and NMI Controller Registers

bit	7	6	5	4	3	2	1	0	
	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	External interrupt enable register (ENIR)
bit	15	14	13	12	11	10	9	8	
	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	External interrupt Request Register (EIRR)
bit	15	14	13	12	11	10	9	8	
	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	
bit	7	6	5	4	3	2	1	0	External interrupt request level setting register (ELVR)
	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	

9.2.1 Interrupt Enable Register (ENIR)

This section describes the bit configuration and function of the interrupt enable register (ENIR).

■ Interrupt Enable Register (ENIR: ENable Interrupt Request Register)

Figure 9.2-2 shows the bit configuration of the interrupt enable register (ENIR)

Figure 9.2-2 Bit Configuration of the Interrupt Enable Register (ENIR)

bit	7	6	5	4	3	2	1	0	Initial value
Address: 000041 _H	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	00000000 _B
	R/W								

The interrupt enable register (ENIR) performs mask control for external interrupt request output.

Output for an interrupt request is enabled based on the bit in this register to which "1" has been written (INT0 enable is controlled by EN0), after which the interrupt request is output to the interrupt controller. The pin corresponding to the bit to which "0" is written holds the interrupt source but does not generate a request to the interrupt controller.

Note:

No mask bit exists for NMI.

9.2.2 External Interrupt Request Register (EIRR)

This section describes the bit configuration and functions of the external interrupt Request Register EIRR.

■ External Interrupt Request Register (EIRR: External Interrupt Request Register)

Figure 9.2-3 shows the bit configuration of the external interrupt Request Register (EIRR).

Figure 9.2-3 Bit Configuration of the External Interrupt Request Register (EIRR)

bit	15	14	13	12	11	10	9	8	Initial value
Address: 000040 _H	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	00000000 _B
	R/W								

The EIRR register, when it is read, indicates that a corresponding external interrupt request exists. When it is written to, the contents of the flip-flop (NMI flag) that indicates this request are cleared. If "1" is read from the EIRR register, an external interrupt request exists at the pin corresponding to this bit.

Write "0" to this register to clear the request flip-flop of the corresponding bit.

Writing "1" to this register has no effect.

For a read by a read-modify-write instruction, "1" is read.

Note:

The NMI flag cannot be read or written to by a user.

For information about the NMI flag, see "NMI" in Section "9.3 Operation of the External Interrupt and NMI Controller".

When the INT0 to INT7 pins input the "H" level in the stop state, their respective ER0 to ER7 bits are set to "1".

9.2.3 External Interrupt Request Level Setting Register (ELVR)

This section describes the bit configuration and functions of the external interrupt request level setting register (ELVR).

■ External Interrupt Request Level Setting Register (ELVR: External Level Register)

Figure 9.2-4 shows the bit configuration of the external interrupt request level setting register (ELVR).

Figure 9.2-4 Bit Configuration of the External Interrupt Request Level Setting Register (ELVR)

bit	15	14	13	12	11	10	9	8	Initial value
Address: 000042H	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	00000000B
	R/W								
bit	7	6	5	4	3	2	1	0	Initial value
Address: 000043H	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	00000000B
	R/W								

The external interrupt request level setting register (ELVR) selects how a request is detected. Two bits are assigned to each of INT0 to INT7, which results in the settings shown in Table 9.2-1. Even though the bits of the EIRR are cleared while the request input is a level, the pertinent bits are set again as long as the input is an active level.

Table 9.2-1 Settings of the LBn and LAN Bits

LBn	LAn	Operation
0	0	"L" level indicates the existence of a request.
0	1	"H" level indicates the existence of a request.
1	0	A rising edge indicates the existence of a request.
1	1	A falling edge indicates the existence of a request.

Note:

A falling edge is always detected at NMI (except in the stop state).

In the stop state, the "L" level is detected.

INT should be set "H" level at stop.

9.3 Operation of the External Interrupt and NMI Controller

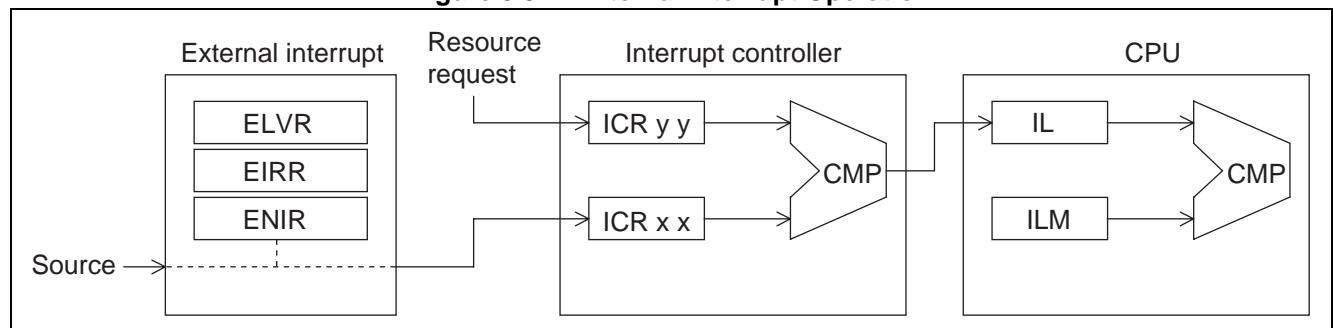
After a request level and an enable register are specified, if a request specified in the external interrupt request level setting register (ELVR) is input to the corresponding pin, this module generates an interrupt request signal to the interrupt controller.

■ Operation of an External Interrupt

For simultaneous interrupt requests, the interrupt controller determines the interrupt request with the highest priority and generates an interrupt for it.

Figure 9.3-1 shows external interrupt operation.

Figure 9.3-1 External Interrupt Operation



■ Return from Standby

To use an external interrupt to return from the stop state, use an "H" level request as the input request regardless of setting the external interrupt request level setting register (ELVR).

Note:

Cut off the pull-up for the INT0 to INT7 pin in the stop state.

■ Operating Procedure for an External Interrupt

Set up a register located inside the external interrupt block as follows:

- 1) Set the general-purpose I/O port served dual use as the pin for the external interrupt input to input port.
- 2) Disable the target bit in the enable register.
- 3) Set the target bit in the request level setting register.
- 4) Clear the target bit in the interrupt register.
- 5) Enable the target bit in the enable register.

Simultaneous writing of 16-bit data is supported for steps 4) and 5).

Before setting a register in this module, you must disable the enable register. In addition, before enabling the enable register, you must clear the interrupt Request Register. This procedure is required to prevent an interrupt source from occurring by mistake while a register is being set or an interrupt is enabled.

CHAPTER 9 EXTERNAL INTERRUPT AND NMI CONTROLLER

■ External Interrupt Request Level

If the request level is an edge request, a pulse width of at least three machine cycles (peripheral clock machine cycles) is required to detect an edge.

If the request input level is a level setting, the pulse width must be at least 3 machine cycles. Also, as long as the interrupt input pin retains the active level, the interrupt request is continuously made to the interrupt controller, even if the source register is cleared.

If the request input level is a level setting and request input arrives from outside and is then cancelled, the request to the interrupt controller remains active because a source holding circuit exists internally.

The interrupt Request Register must be cleared to cancel a request to the interrupt controller.

Figure 9.3-2 shows clearing of the source holding circuit when a level is set. Figure 9.3-3 shows an interrupt source and an interrupt request to the interrupt controller when interrupts are enabled.

Figure 9.3-2 Clearing the Source Holding Circuit when a Level is Set

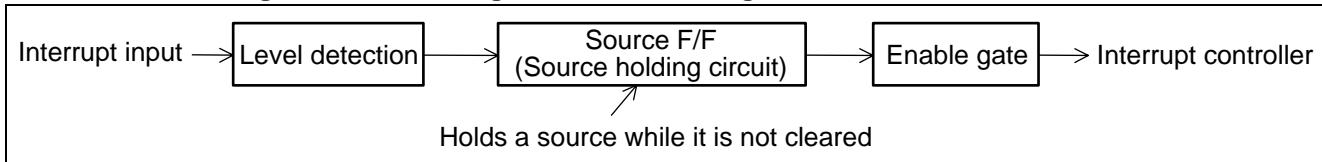
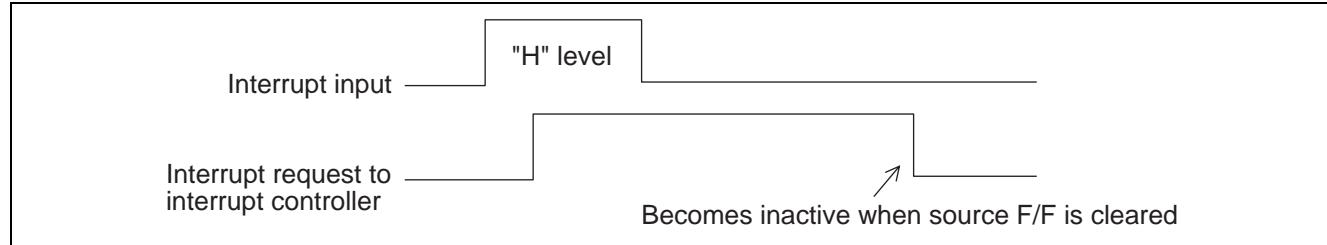


Figure 9.3-3 Interrupt Source and Interrupt Request to Interrupt Controller when Interrupts are Enabled



■ NMI

An NMI has the highest level among the user interrupts and usually cannot be masked. However, as an exception, if an NMI is activated before it is set in ILM, the CPU does not accept the NMI but only detects the NMI source. The NMI source is then held until ILM is set to the level that allows the NMI to be accepted. For this reason, before using an NMI, be sure to set ILM to 16 or more after a reset.

As the internal source flag of NMI cannot be accessed by the CPU, the NMI pin must be maintained at the level "H" after a reset.

An NMI is accepted under the following conditions:

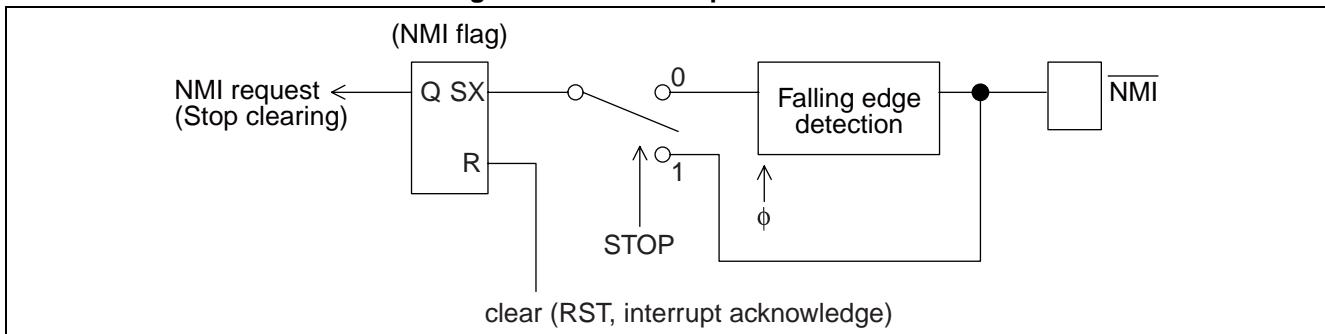
- Normal: falling edge
- STOP mode: "L" level

An NMI can be used to clear stop mode. Inputting the "L" level in the stop state clears the stop state and causes the oscillation stabilization wait time to start. To perform NMI processing after clearing the stop state, maintain the NMI pin at the "L" level and return it to the "H" level in the NMI processing routine.

The NMI request detector has an NMI flag that is set for an NMI request and is cleared only if an interrupt for the NMI itself is accepted or a reset occurs. Note that this bit is not readable or writable.

Figure 9.3-4 shows the NMI request detector.

Figure 9.3-4 NMI Request Detector

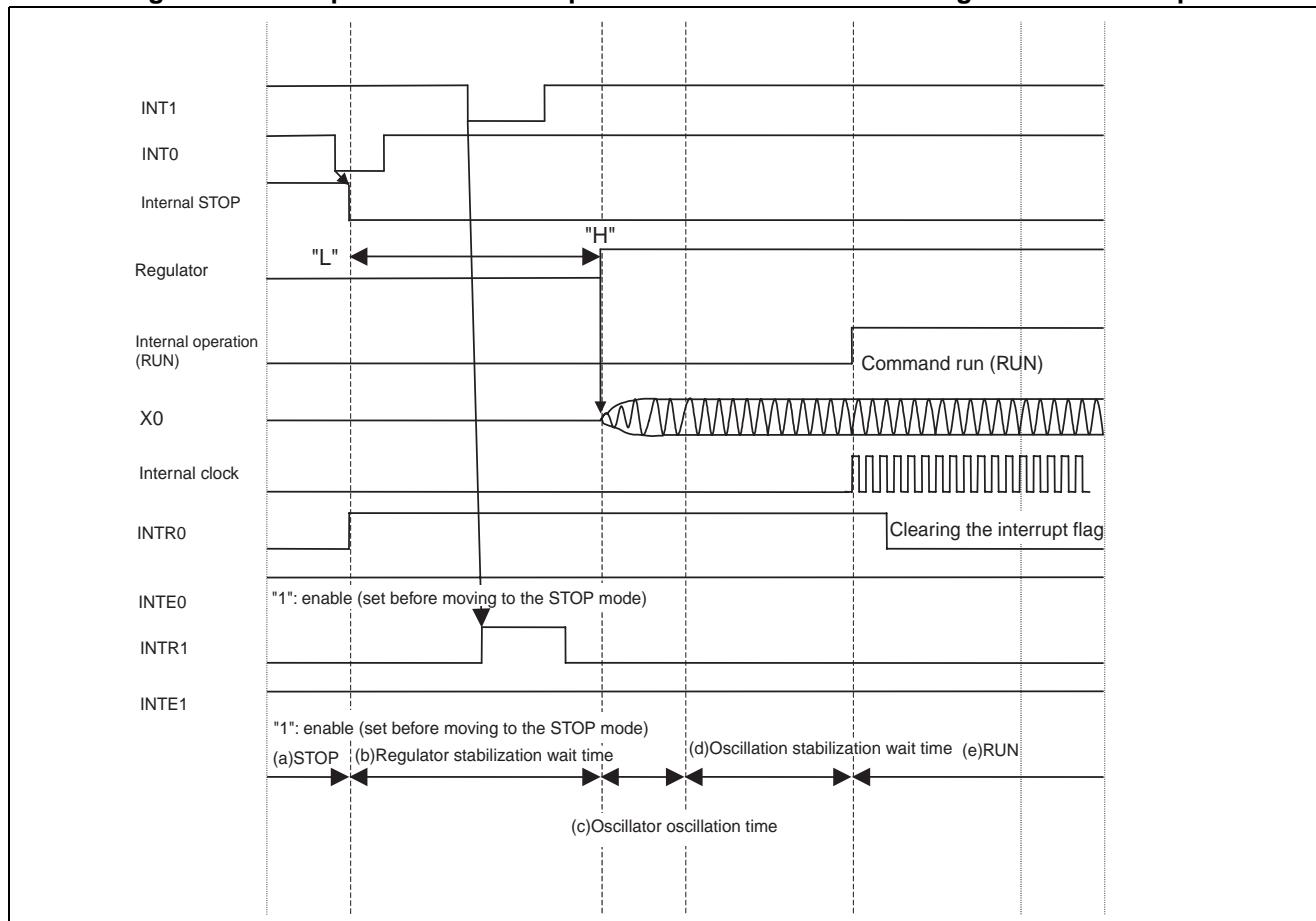


■ Notes on Returning from STOP State Using External Interrupt

In a STOP state, the first external interrupt signal to be input to the INT pin is input without being synchronized, enabling return from the STOP state. However, there is a period in which no other external interrupt signals can be recognized (period of $b + c + d$ shown in Figure 9.3-5) between when the STOP state is released and when the oscillation stabilization wait time passes. This is because an external input signal is synchronized with the internal clock after the STOP state is released; therefore, that interrupt source cannot be retained until the clock stabilizes.

When inputting an external interrupt signal after the STOP state is released, wait until the oscillation stabilization wait time elapses.

Figure 9.3-5 Sequence of Return Operation from STOP State Using External Interrupt



■ Return Operation from STOP State

The following operation is performed on the current circuit when an external interrupt is used to return from the STOP state.

- Procedure prior to STOP state transition

 - Setting a path for an external interrupt

It is necessary to set a path for inputting an external interrupt in order to release the STOP state before the device enters the STOP state. The PFR register (Port Function Register) is used to perform this procedure. In a normal state (state other than the STOP state), an interrupt input path is already allocated. Therefore, the above procedure should be ignored. However, the input path is controlled by the value of the PFR register in the STOP state.

Name of pin used to release STOP state	Register and bit to be set
INT7/SCK2/PG7	Set bit 7 in PDRG to "0".
INT6/SDT2/PG6	Set bit 6 in PDRG to "0".
INT5/SIN2/PG5	Set bit 5 in PDRG to "0".
INT4/ATG/PG4/FRCK	Set bit 4 in PDRG to "0".
INT3/PG3/ICU3	Set bit 3 in PDRG to "0".
INT2/PG2/ICU2	Set bit 2 in PDRG to "0".
INT1/PG1/ICU1	Set bit 1 in PDRG to "0".
INT0/PG0/ICU0	Set bit 0 in PDRG to "0".

- Inputting an external interrupt

When returning from a STOP state, an external interrupt signal is ready to transmit an input signal without synchronization. As soon as this interrupt signal becomes valid, the internal STOP signal immediately starts to fall. At the same time, the external interrupt circuit is switched to synchronize another level interrupt input.

- Regulator stabilization wait time

Once the internal STOP signal falls, the STOP regulator is switched to the RUN regulator. A stabilization wait time is secured for the internal output voltage because the operation becomes unstable when the internal operation starts before the output voltage of the RUN Regulator stabilizes. The clock is stopped during this period.

- Oscillator's oscillation time

The clock starts oscillating once the regulator stabilization wait time has passed. The oscillation time varies depending on the oscillator used.

- Oscillation stabilization wait time

The oscillation stabilization wait time is taken within the device after the time. The oscillation stabilization wait time is specified by the OS1 and OS0 bits in the standby control register. Once the oscillation stabilization wait time has passed, an internal clock is supplied, an instruction operation can be started by an external interrupt and any external interrupt source other than the source for returning from the STOP state can be accepted.

CHAPTER 10 DELAYED INTERRUPT MODULE

This chapter describes the functions and operation of the delayed interrupt module.

- 10.1 Overview of the Delayed Interrupt Module
- 10.2 Delayed Interrupt Module Registers
- 10.3 Operation of the Delayed Interrupt Module

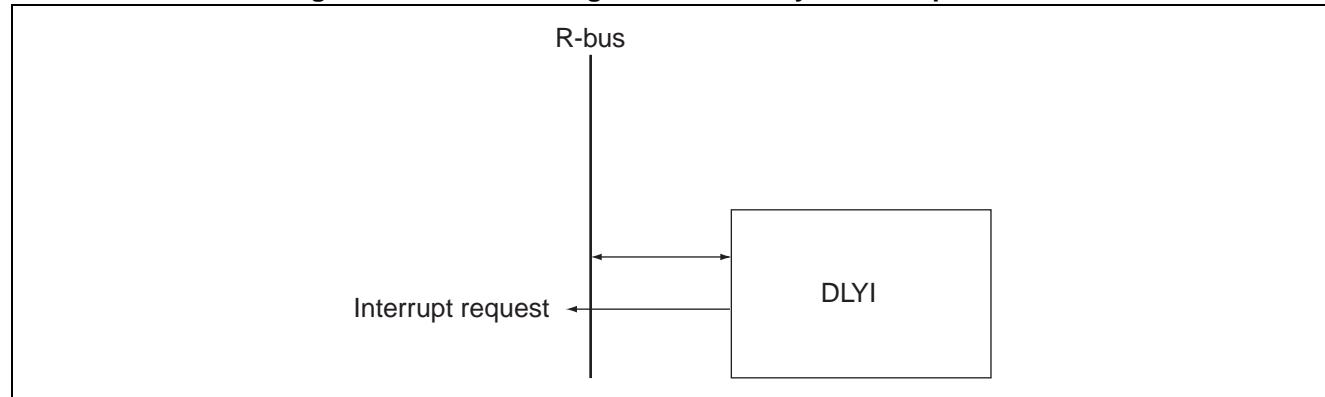
10.1 Overview of the Delayed Interrupt Module

The delayed interrupt module generates an interrupt for switching tasks. Use this module to allow a software program to generate an interrupt request for the CPU or to clear an interrupt request.

■ Block Diagram of the Delayed Interrupt Module

Figure 10.1-1 shows a block diagram of the delayed interrupt module.

Figure 10.1-1 Block Diagram of the Delayed Interrupt Module



10.2 Delayed Interrupt Module Registers

This section describes the configuration and functions of the registers used by the delayed interrupt module.

■ Delayed Interrupt Module Registers

The delayed interrupt module includes the delayed interrupt control register (DICR).

Figure 10.2-1 shows the configuration of the delayed interrupt control register (DICR).

Figure 10.2-1 Configuration of the Delayed Interrupt Control Register (DICR)

bit	7	6	5	4	3	2	1	0	Delayed interrupt control register (DICR)
	-	-	-	-	-	-	-	DLYI	

■ Delayed Interrupt Control Register (DICR)

The delayed interrupt control register (DICR: Delayed Interrupt Control Register) controls delayed interrupts.

Figure 10.2-2 shows the bit configuration of the delayed interrupt control register (DICR).

Figure 10.2-2 Bit Configuration of the Delayed Interrupt Control Register (DICR)

bit	7	6	5	4	3	2	1	0	Initial value -----0B
Address: 000044H	-	-	-	-	-	-	-	DLYI	R/W

The following describes the bit functions of the delayed interrupt control register (DICR) bits.

[bit0] DLYI

This bit controls the generation and clearing of the pertinent interrupt source.

Table 10.2-1 shows the function for the generation and clearing of the pertinent interrupt source.

Table 10.2-1 Functions for Generation and Clearing the Pertinent Interrupt Source

DLYI	Description
0	A delayed interrupt source is cleared or no request exists. [initial value]
1	A delayed interrupt source is generated.

10.3 Operation of the Delayed Interrupt Module

A delayed interrupt refers to an interrupt generated for switching tasks. Use this function to allow a software program to generate an interrupt request for the CPU or to clear an interrupt request.

■ Interrupt Number

A delayed interrupt is assigned to the interrupt source corresponding to the largest interrupt number.

On the MB91301 series, a delayed interrupt is assigned to interrupt number 63 (3F_H).

■ DLYI Bit of DICR

Write "1" to this bit to generate a delayed interrupt source. Write "0" to it to clear a delayed interrupt source.

This bit is the same as the interrupt source flag for a normal interrupt. Therefore, clear this bit and switch tasks in the interrupt routine.

CHAPTER 11 INTERRUPT CONTROLLER

This chapter describes the overview of the interrupt controller, the configuration and functions of registers, and interrupt controller operation. It also presents an example of using the hold request cancellation request function.

- 11.1 Overview of the Interrupt Controller
- 11.2 Interrupt Controller Registers
- 11.3 Interrupt Controller Operation
- 11.4 Example of Using the Hold Request Cancellation Request Function (HRCR)

11.1 Overview of the Interrupt Controller

The interrupt controller controls interrupt acceptance and arbitration processing.

■ Hardware Configuration of the Interrupt Controller

The interrupt controller consists of the following components:

- Interrupt control registers (ICR) register
- Interrupt priority decision circuit
- Interrupt level and interrupt number (vector) generator
- HOLD request cancellation request generator

■ Major Functions of the Interrupt Controller

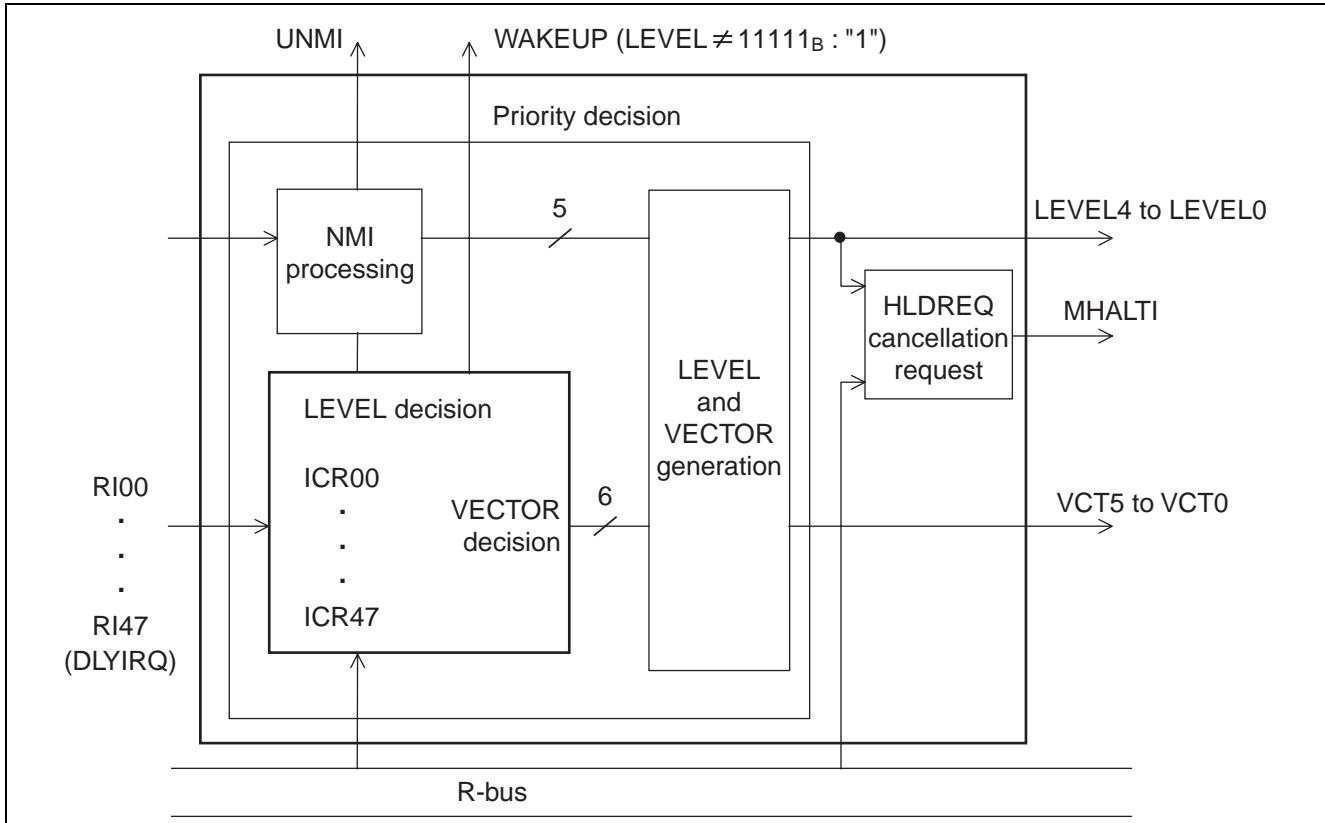
The interrupt controller has the following major functions:

- Detecting NMI requests and interrupt requests
- Deciding priority (using a level or number)
- Passing to the CPU an interrupt level based on the decision result to provide information about the interrupt source
- Passing to the CPU an interrupt number based on the decision result to provide information about the interrupt source
- Instruction for return from stop mode due to the occurrence of an interrupt with an NMI/interrupt level other than 11111_B (to CPU)
- Generating a HOLD request cancellation request for the bus master

■ Block Diagram

Figure 11.1-1 shows a block diagram of the interrupt controller.

Figure 11.1-1 Block Diagram of the Interrupt Controller



11.2 Interrupt Controller Registers

This section describes the configuration and functions of the registers used by the interrupt controller.

■ Interrupt Controller Registers

Figure 11.2-1 shows the registers of the interrupt controller.

Figure 11.2-1 Interrupt Controller Registers

bit	7	6	5	4	3	2	1	0	
Address: 000440 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR00
Address: 000441 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR01
Address: 000442 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR02
Address: 000443 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR03
Address: 000444 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR04
Address: 000445 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR05
Address: 000446 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR06
Address: 000447 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR07
Address: 000448 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR08
Address: 000449 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR09
Address: 00044A _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR10
Address: 00044B _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR11
Address: 00044C _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR12
Address: 00044D _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR13
Address: 00044E _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR14
Address: 00044F _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR15
Address: 000450 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR16
Address: 000451 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR17
Address: 000452 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR18
Address: 000453 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR19
Address: 000454 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR20
Address: 000455 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR21
Address: 000456 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR22
Address: 000457 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR23
Address: 000458 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR24
Address: 000459 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR25
Address: 00045A _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR26
Address: 00045B _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR27
Address: 00045C _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR28
Address: 00045D _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR29
Address: 00045E _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR30
Address: 00045F _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR31

(Continued)

(Continued)

	bit	7	6	5	4	3	2	1	0	
Address:	000460 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR32
Address:	000461 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR33
Address:	000462 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR34
Address:	000463 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR35
Address:	000464 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR36
Address:	000465 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR37
Address:	000466 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR38
Address:	000467 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR39
Address:	000468 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR40
Address:	000469 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR41
Address:	00046A _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR42
Address:	00046B _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR43
Address:	00046C _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR44
Address:	00046D _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR45
Address:	00046E _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR46
Address:	00046F _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	ICR47
Address:	000045 _H	MHALTI	-	-	LVL4	LVL3	LVL2	LVL1	LVL0	HRCL

11.2.1 Interrupt Control Register (ICR)

An interrupt control register is provided for each of the interrupt input and sets the interrupt level of the corresponding interrupt request.

■ Bit Configuration of Interrupt Control Register (ICR)

Figure 11.2-2 shows the bit configuration of the interrupt control register (ICR: Interrupt Control Register).

Figure 11.2-2 Bit Configuration of the Interrupt Control Register (ICR)

bit	7	6	5	4	3	2	1	0	Initial value
Address 000440 _H	-	-	-	ICR4	ICR3	ICR2	ICR1	ICR0	---11111 _B
to 00046F _H				R	R/W	R/W	R/W	R/W	

■ Detailed Bit of Interrupt Control Register (ICR)

The following describes the bit functions of the interrupt control register (ICR).

[bit4 to bit0] ICR4 to ICR0 interrupt level setting

These bits, which are the interrupt level setting bits, specify the interrupt level of the corresponding interrupt request.

If an interrupt request has an interrupt level specified in this register that exceeds the level mask value specified in the interrupt level mask register (ILM) of the CPU, it is masked by the CPU.

These bits are initialized to 11111_B by a reset.

Table 11.2-1 shows the correspondence between possible interrupt level setting bits and interrupt levels.

Table 11.2-1 Correspondence between Possible Interrupt Level Setting Bits and Interrupt Levels

ICR4	ICR3	ICR2	ICR1	ICR0	Interrupt level
0	0	0	0	0	0
0	1	1	1	0	14
0	1	1	1	1	15
1	0	0	0	0	16
1	0	0	0	1	17
1	0	0	1	0	18
1	0	0	1	1	19
1	0	1	0	0	20
1	0	1	0	1	21
1	0	1	1	0	22
1	0	1	1	1	23
1	1	0	0	0	24
1	1	0	0	1	25
1	1	0	1	0	26
1	1	0	1	1	27
1	1	1	0	0	28
1	1	1	0	1	29
1	1	1	1	0	30
1	1	1	1	1	31
(High)					
(Low)					
Interrupt disabled					

Note: The ICR4 bit is always "1". "0" cannot be written to it.

11.2.2 Hold Request Cancellation Request Level Setting Register (HRCL)

The hold request cancellation request level setting register (HRCL) is a level setting register used to generate a hold request cancellation request.

■ Bit Configuration of Hold Request Cancellation Request Level Setting Register (HRCL)

Figure 11.2-3 shows the bit configuration of the hold request cancellation request level setting register (HRCL).

Figure 11.2-3 Bit Configuration of the Hold Request Cancellation Request Level Setting Register (HRCL)

bit	7	6	5	4	3	2	1	0	Initial value
Address: 000045H	MHALTI	-	-	LVL4	LVL3	LVL2	LVL1	LVL0	0-11111B

■ Detailed Bit of Hold Request Cancellation Request Level Setting Register (HRCL)

The following describes the bit functions of the hold request cancellation request level setting register (HRCL).

[bit7] MHALTI: DMA transfer disable by NMI request

This bit is the DMA transfer disable bit controlled by an NMI request. An NMI request sets this bit to "1". Write "0" to this bit to clear it. At the end of an NMI routine, clear this bit the same way it would be cleared in a normal interrupt routine.

[bit4 to bit0] LVL4 to LVL0: Interrupt level setting

These bits set the interrupt level used to issue a hold request cancellation request to the bus master.

If an interrupt request with a higher level than the level specified in the HRCL register occurs, issue a hold request cancellation request to the bus master.

The LVL4 bit is always "1". "0" cannot be written to it.

11.3 Interrupt Controller Operation

This section describes the following items regarding operation of the interrupt controller:

- Priority decision
 - NMI
 - Hold request cancellation request
 - Return from standby mode (stop/sleep)
-

■ Priority Decision

The interrupt controller selects the interrupt source with the highest priority from among those that exist simultaneously and outputs the interrupt level and the interrupt number of this source to the CPU.

The following shows the priority decision criteria for interrupt sources:

- NMI
- Source that meets the following conditions:
 - Source with a value other than 31 as the interrupt level (31 means interrupts disabled)
 - Source with the smallest value for the interrupt level
 - Source with the smallest interrupt number that satisfies the both conditions above

If no interrupt source is selected according to the above decision criteria, 31 (11111_B) is output as the interrupt level. The interrupt number at this time is undefined.

Table 11.3-1 shows the relationship between interrupt sources, interrupt numbers and interrupt levels.

Table 11.3-1 Relationship between Interrupt Sources, Interrupt Numbers, and Interrupt Levels (1 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
Reset	0	00	–	3FC _H	000FFFFC _H	–
Mode vector	1	01	–	3F8 _H	000FFFF8 _H	–
Reserved for system	2	02	–	3F4 _H	000FFFF4 _H	–
Reserved for system	3	03	–	3F0 _H	000FFFF0 _H	–
Reserved for system	4	04	–	3EC _H	000FFFEC _H	–
Reserved for system	5	05	–	3E8 _H	000FFF8E _H	–
Reserved for system	6	06	–	3E4 _H	000FFF4E _H	–
No-coprocessor trap	7	07	–	3E0 _H	000FFF0E _H	–
Coprocessor error trap	8	08	–	3DC _H	000FFFDC _H	–
INTE instruction	9	09	–	3D8 _H	000FFF8D8 _H	–
Instruction break exception	10	0A	–	3D4 _H	000FFF4D4 _H	–

CHAPTER 11 INTERRUPT CONTROLLER

Table 11.3-1 Relationship between Interrupt Sources, Interrupt Numbers, and Interrupt Levels (2 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
Operand break trap	11	0B	–	3D0 _H	000FFFFD0 _H	–
Step trace trap	12	0C	–	3CC _H	000FFFCC _H	–
NMI request (tool)	13	0D	–	3C8 _H	000FFFC8 _H	–
Undefined instruction exception	14	0E	–	3C4 _H	000FFFC4 _H	–
NMI request	15	0F	Fixed to 15(F _H)	3C0 _H	000FFFC0 _H	–
External Interrupt 0	16	10	ICR00	3BC _H	000FFFBC _H	6
External Interrupt 1	17	11	ICR01	3B8 _H	000FFFBB8 _H	7
External Interrupt 2	18	12	ICR02	3B4 _H	000FFFBB4 _H	11
External Interrupt 3	19	13	ICR03	3B0 _H	000FFFBB0 _H	12
External Interrupt 4	20	14	ICR04	3AC _H	000FFFAC _H	–
External Interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H	–
External Interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H	–
External Interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H	–
Reload Timer 0	24	18	ICR08	39C _H	000FFF9C _H	8
Reload Timer 1	25	19	ICR09	398 _H	000FFF98 _H	9
Reload Timer 2	26	1A	ICR10	394 _H	000FFF94 _H	10
UART0 (reception completed)	27	1B	ICR11	390 _H	000FFF90 _H	0
UART1 (reception completed)	28	1C	ICR12	38C _H	000FFF8C _H	1
UART2 (reception completed)	29	1D	ICR13	388 _H	000FFF88 _H	2
UART0 (transmission completed)	30	1E	ICR14	384 _H	000FFF84 _H	3
UART1 (transmission completed)	31	1F	ICR15	380 _H	000FFF80 _H	4
UART2 (transmission completed)	32	20	ICR16	37C _H	000FFF7C _H	5
DMAC0 (end, error)	33	21	ICR17	378 _H	000FFF78 _H	–
DMAC1 (end, error)	34	22	ICR18	374 _H	000FFF74 _H	–
DMAC2 (end, error)	35	23	ICR19	370 _H	000FFF70 _H	–
DMAC3 (end, error)	36	24	ICR20	36C _H	000FFF6C _H	–
DMAC4 (end, error)	37	25	ICR21	368 _H	000FFF68 _H	–
A/D	38	26	ICR22	364 _H	000FFF64 _H	15
PPG0	39	27	ICR23	360 _H	000FFF60 _H	13
PPG1	40	28	ICR24	35C _H	000FFF5C _H	14

Table 11.3-1 Relationship between Interrupt Sources, Interrupt Numbers, and Interrupt Levels (3 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
PPG2	41	29	ICR25	358 _H	000FFF58 _H	–
PPG3	42	2A	ICR26	354 _H	000FFF54 _H	–
Reserved for system	43	2B	ICR27	350 _H	000FFF50 _H	–
U-TIMER0	44	2C	ICR28	34C _H	000FFF4C _H	–
U-TIMER1	45	2D	ICR29	348 _H	000FFF48 _H	–
U-TIMER2	46	2E	ICR30	344 _H	000FFF44 _H	–
Time-base timer overflow	47	2F	ICR31	340 _H	000FFF40 _H	–
I ² C I/F0	48	30	ICR32	33C _H	000FFF3C _H	–
I ² C I/F1	49	31	ICR33	338 _H	000FFF38 _H	–
Reserved for system	50	32	ICR34	334 _H	000FFF34 _H	–
Reserved for system	51	33	ICR35	330 _H	000FFF30 _H	–
16-bit free run timer	52	34	ICR36	32C _H	000FFF2C _H	–
ICU0 (fetch)	53	35	ICR37	328 _H	000FFF28 _H	–
ICU1 (fetch)	54	36	ICR38	324 _H	000FFF24 _H	–
ICU2 (fetch)	55	37	ICR39	320 _H	000FFF20 _H	–
ICU3 (fetch)	56	38	ICR40	31C _H	000FFF1C _H	–
Reserved for system	57	39	ICR41	318 _H	000FFF18 _H	–
Reserved for system	58	3A	ICR42	314 _H	000FFF14 _H	–
Reserved for system	59	3B	ICR43	310 _H	000FFF10 _H	–
Reserved for system	60	3C	ICR44	30C _H	000FFF0C _H	–
Reserved for system	61	3D	ICR45	308 _H	000FFF08 _H	–
Reserved for system	62	3E	ICR46	304 _H	000FFF04 _H	–
Delayed interrupt source bit	63	3F	ICR47	300 _H	000FFF00 _H	–
Reserved for system (used in REALOS)	64	40	–	2FC _H	000FFEFC _H	–
Reserved for system (used in REALOS)	65	41	–	2F8 _H	000FFEF8 _H	–
Reserved for system	66	42	–	2F4 _H	000FFEF4 _H	–
Reserved for system	67	43	–	2F0 _H	000FFEF0 _H	–
Reserved for system	68	44	–	2EC _H	000FFEEC _H	–
Reserved for system	69	45	–	2E8 _H	000FFEE8 _H	–
Reserved for system	70	46	–	2E4 _H	000FFEE4 _H	–

Table 11.3-1 Relationship between Interrupt Sources, Interrupt Numbers, and Interrupt Levels (4 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	Default address of TBR	RN
	Decimal	Hexadecimal				
Reserved for system	71	47	–	2E0 _H	000FFEE0 _H	–
Reserved for system	72	48	–	2DC _H	000FFEDC _H	–
Reserved for system	73	49	–	2D8 _H	000FFED8 _H	–
Reserved for system	74	4A	–	2D4 _H	000FFED4 _H	–
Reserved for system	75	4B	–	2D0 _H	000FFED0 _H	–
Reserved for system	76	4C	–	2CC _H	000FFECC _H	–
Reserved for system	77	4D	–	2C8 _H	000FFEC8 _H	–
Reserved for system	78	4E	–	2C4 _H	000FFEC4 _H	–
Reserved for system	79	4F	–	2C0 _H	000FFEC0 _H	–
Used in INT instruction	80 to 255	50 to FF	–	2BC _H to 000 _H	000FFEBC _H to 000FFC00 _H	–

■ NMI

An NMI (Non Maskable Interrupt) has the highest priority among the interrupt sources handled by the interrupt controller. Thus, an NMI is always selected if it occurs at the same time as other interrupt sources.

- If an NMI occurs, the following information is reported to the CPU:
 - Interrupt level:15 (01111_B)
 - Interrupt number:15 (0001111_B)
- Detecting an NMI

The external interrupt and NMI module sets and detects an NMI. This module only generates an interrupt level, interrupt number, and MHALTI in response to an NMI request.

- Preventing a DMA transfer occurring due to an NMI

If an NMI request occurs, the MHALTI bit of the HRCL register is set to "1" to prevent DMA transfer. To clear the state preventing DMA transfer, clear the MHALTI bit to "0" at the end of the NMI routine.

■ Hold Request Cancellation Request (HRCR: Hold Request Cancel Request)

For an interrupt with a higher priority to be processed during CPU hold, the device that has generated the hold request must cancel the request. Set the interrupt level in the HRCL register to be used as the criterion of generating a cancellation request.

○ Generation criteria

If an interrupt source with a higher interrupt level than the level specified in the HRCL register occurs, a hold request cancellation request is generated.

- If the interrupt level of the HRCL register is greater than the interrupt level after a priority decision, a cancellation request occurs.
- If the interrupt level of the HRCL register is equal to or less than the interrupt level after a priority decision, no cancellation request occurs.

Because the cancellation request remains valid, no DMA transfer occurs unless the interrupt source that has caused the cancellation request is cleared. Be sure to clear the corresponding interrupt source.

If an NMI is used, the cancellation request is valid because the MHALTI bit of the HRCL register is set to "1".

○ Possible levels

Values that can be set in the HRCL register range from " 10000_B " to " 11111_B ", which is the same range as for the ICR.

If this register is set to " 11111_B ", a cancellation request is issued for all the interrupt levels. If this register is set to " 10000_B ", a cancellation request is issued only for an NMI.

Table 11.3-2 shows the settings of interrupt levels at which a hold request cancellation request occurs.

Table 11.3-2 Settings of Interrupt Levels at which Hold Request Cancellation Request Occurs

HRCL register	Interrupt levels at which a cancellation request occurs
16	NMI only
17	NMI, Interrupt level 16
18	NMI, Interrupt levels 16 and 17
...	...
31	NMI, Interrupt levels 16 to 30 [initial value]

After a reset, since DMA transfer is not allowed at any interrupt level, no DMA transfer is performed if an interrupt has occurred. Be sure to set the HRCL register to the necessary value.

■ Return from Standby Mode (Sleep/Stop)

This module implements a function that causes a return from stop mode if an interrupt request occurs. If at least one interrupt request that includes NMI occurs (with an interrupt level other than "11111_B") from the peripheral, a return request from stop mode is generated for the clock controller.

Since the priority decision unit restarts operation when a clock is supplied after returning from stop, the CPU executes instructions until the result of the priority decision unit is obtained.

The same operation occurs after a return from the sleep state.

Registers in the interrupt controller can be accessed even in the sleep state.

Notes:

- The device returns from stop mode if an NMI request is issued. However, set an NMI so that valid input can be detected in the stop state.
 - Provide an interrupt level of "11111_B" in the corresponding peripheral control register for an interrupt source that you do not want to cause return from stop or sleep.
-

11.4 Example of Using the Hold Request Cancellation Request Function (HRCR)

To allow the CPU to perform high-priority processing during DMA transfer, cancel a hold request for DMA and clear the hold state. In this example, an interrupt is used to cancel a hold request to the DMA, allowing the CPU to perform priority operations.

■ Control Registers

- **HRCL (hold request cancellation request level setting register):**

If an interrupt with a higher interrupt level than the level in the HRCL register occurs, a hold request cancellation request is generated for DMA. This register sets the level to be used as the criterion for this purpose.

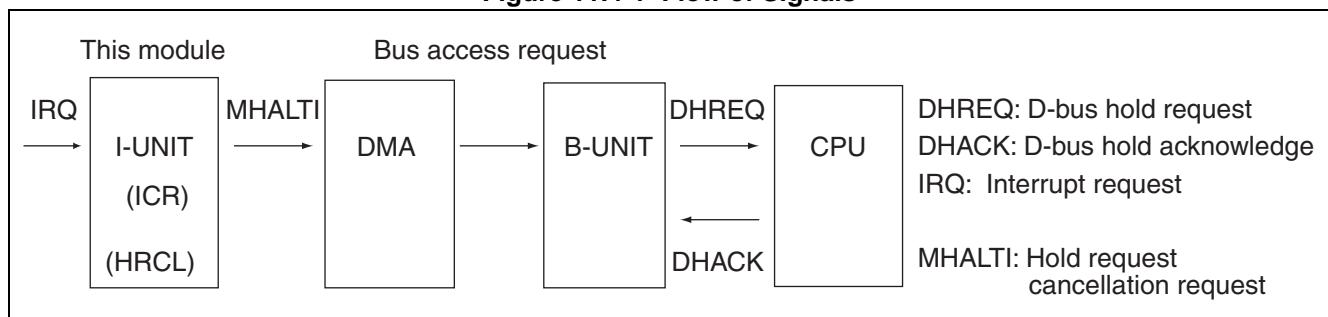
- **ICR:**

This register sets a level higher than the level in the HRCL register for the ICR corresponding to the interrupt source that will be used.

■ Hardware Configuration

The flow of signals is as follows.

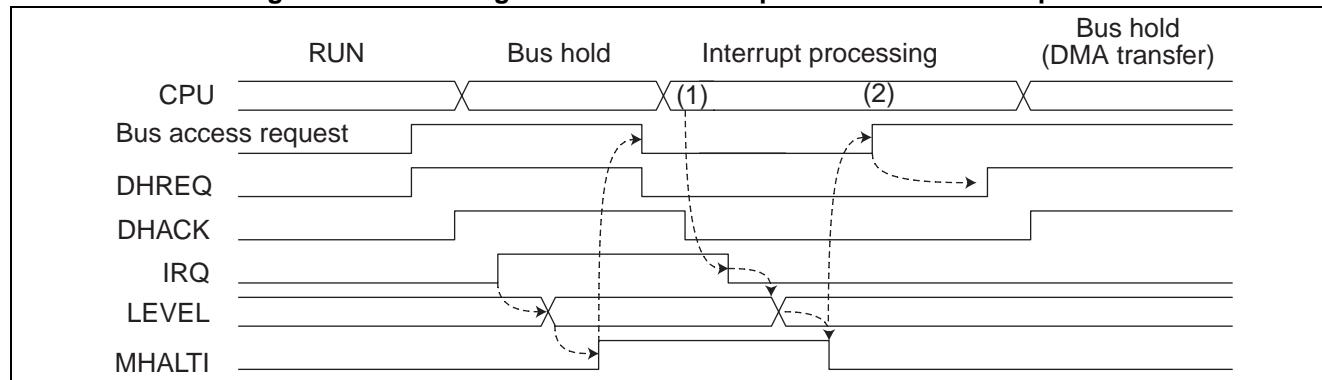
Figure 11.4-1 Flow of Signals



■ Hold Request Cancellation Request Sequence

Figure 11.4-2 shows the timing chart of a hold request cancellation request.

Figure 11.4-2 Timing Chart of a Hold Request Cancellation Request



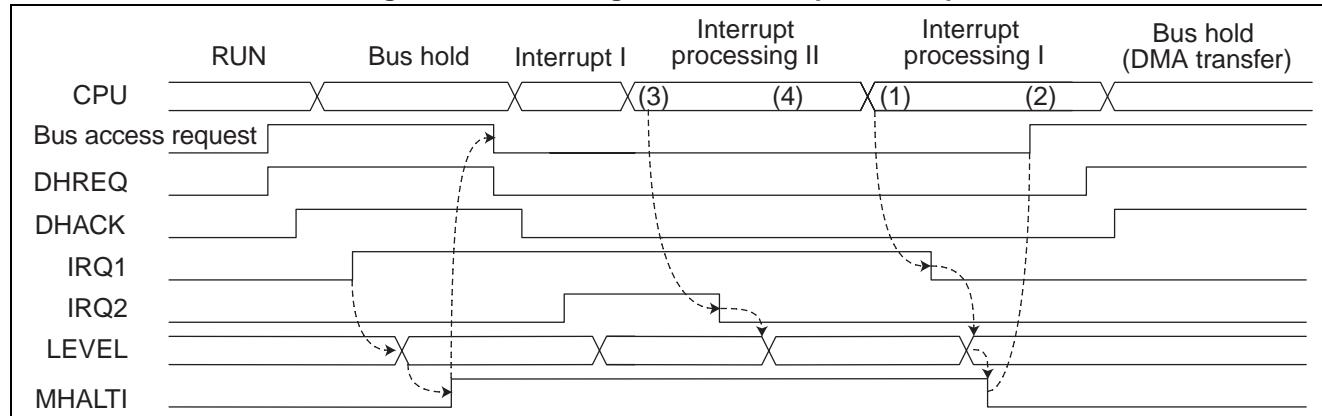
Example of Interrupt Routine [HRCL<ICR (LEVEL)]

- (1) Interrupt source clear
- to
- (2) RETI

If an interrupt request occurs, the interrupt level changes. If the interrupt level is higher than the level in the HRCL register, MHALTI is started for DMA. This causes DMA to cancel an access request and the CPU to return from the hold state to perform the interrupt processing.

Figure 11.4-3 shows the timing chart for multiple interrupts.

Figure 11.4-3 Timing Chart for Multiple Interrupts



Example of Interrupt Routine "Interrupt Level HRCL < ICR interrupt I) < ICR (interrupt II)"

- (1), (3) Interrupt source clear
- to
- (2), (4) RETI

In the above example, while interrupt routine I is being executed, an interrupt with a higher priority occurs. While the interrupt with a higher level than the level in the HRCL register occurs, DHREQ is low.

Note:

Be especially careful about the relationship between interrupt levels specified in the HRCL register and ICR.

CHAPTER 12 A/D CONVERTER

This chapter describes the overview A/D converter, the configuration and functions of registers, and A/D converter operation.

- 12.1 Overview of the A/D Converter
- 12.2 A/D Converter Registers
- 12.3 A/D Converter Operation
- 12.4 Precautions on the Using A/D Converter

12.1 Overview of the A/D Converter

The A/D converter is a module that converts an analog input voltage to a digital value in the successive approximation conversion method.

■ Features

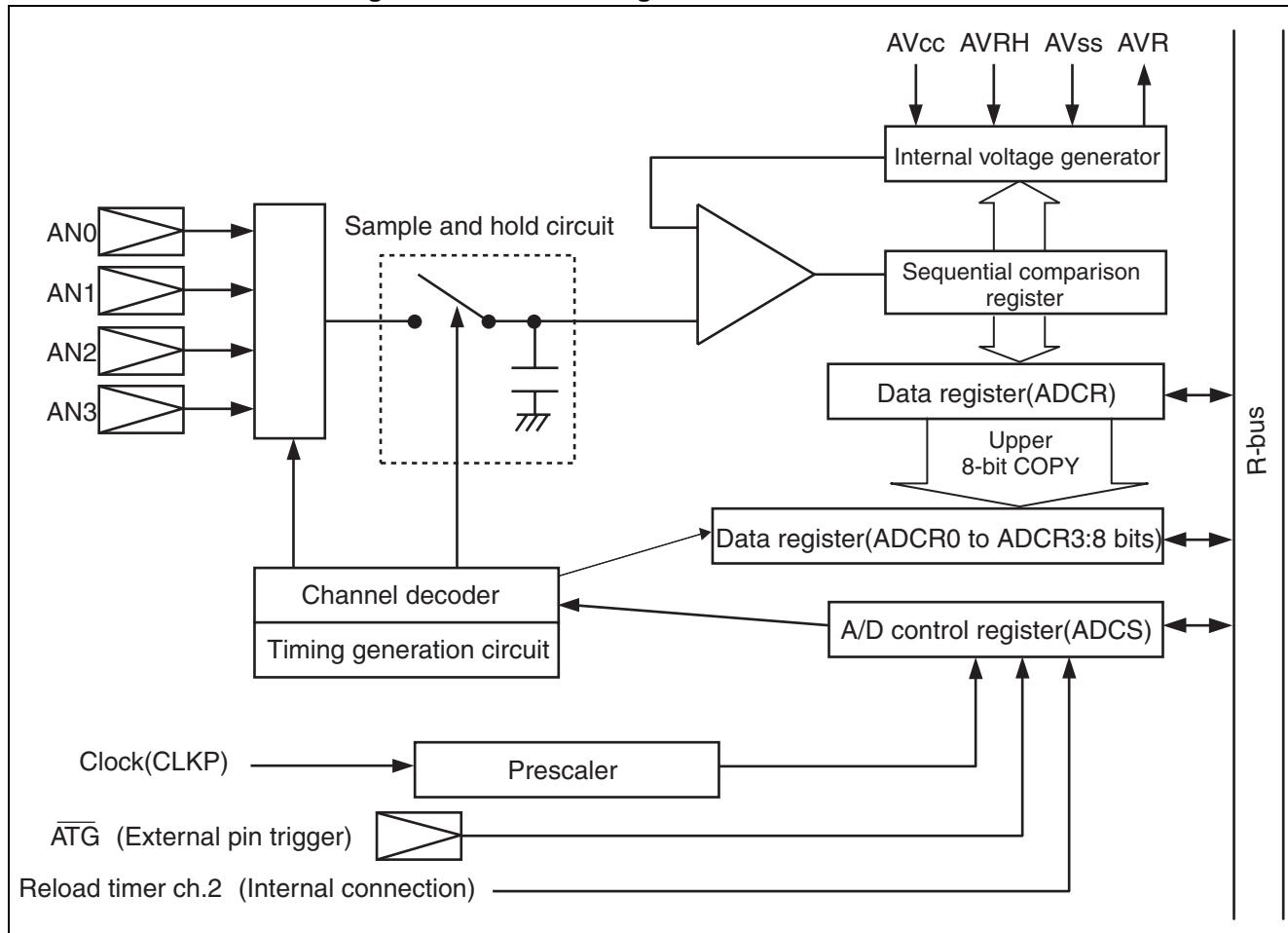
The A/D converter, which converts an analog voltage input to an analog input pin (input voltage) to a digital value, has the following features:

- Peripheral clock (CLKP): 140 clock cycles
- Minimum conversion time: $4.1 \mu\text{s}$ per channel (for a $34 \text{ MHz} = \text{CLKP machine clock}$)
- Built-in sample and hold circuit
- Resolution: 10 bits
- A program can select one of four analog input channels:
 - Single-shot conversion mode: Converts one channel.
 - Scan conversion mode: Continuously converts multiple channels. Up to four channels can be programmed.
- Selectable the following 3 operating modes;
 - Single conversion mode: Converts a specified channel.
 - Continuous conversion mode: Repetitiously converts a specified channel.
 - Stop conversion mode: Converts one channel, pauses, and stands by until the next activation occurs (conversion start can be synchronized).
- DMA transfer can be started due to an interrupt.
- To start conversion, select software, an external trigger (falling edge), or the reload timer (rising edge).

■ Block Diagram

Figure 12.1-1 shows a block diagram of the A/D converter.

Figure 12.1-1 Block Diagram of the A/D Converter



12.2 A/D Converter Registers

This section describes the configuration and functions of the registers used by the A/D converter.

■ A/D Converter Registers

Figure 12.2-1 shows the registers of the A/D converter.

Figure 12.2-1 A/D Converter Registers

Address: 00007A _H	bit	15	14	13	12	11	10	9	8	
		BUSY	INT	INTE	CRF	STS1	STS0	STRT	-	
Address: 000078 _H	bit	7	6	5	4	3	2	1	0	
		MD1	MD0	ANS2	ANS1	ANS0	ANE2	ANE1	ANE0	Control status register (ADCS)
Address: 00007C _H 00007D _H 00007E _H 00007F _H	bit	15	14	13	12	11	10	9	8	
		-	-	-	-	-	-	D9	D8	Data register (ADCR)
	bit	7	6	5	4	3	2	1	0	
		D7	D6	D5	D4	D3	D2	D1	D0	Conversion result register (ADCR0 to ADCR3)

12.2.1 Control Status Register (ADCS)

The control status register (ADCS) controls the A/D converter and displays its status.

■ Bit Configuration of Control Status Register (ADCS)

Figure 12.2-2 shows the bit configuration of the control status register (ADCS).

Figure 12.2-2 Bit Configuration of the Control Status Register (ADCS)

Address: 00007AH	bit	15	14	13	12	11	10	9	8	Initial value
		BUSY	INT	INTE	CRF	STS1	STS0	STRT	-	00000000B
		R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
	bit	7	6	5	4	3	2	1	0	Initial value
		MD1	MD0	ANS2	ANS1	ANS0	ANE2	ANE1	ANE0	00000000B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Note:

Do not rewrite the control status register (ADCS) while the A/D conversion is in progress, except for STRT bit and BUSY bit.

■ Detailed Bit of Control Status Register (ADCS)

The following describes the bit functions of the control status register (ADCS).

[bit15] BUSY (BUSY flag and stop)

This bit has the following different functions during reading and writing:

- Reading:

This bit indicates whether the A/D converter is operating. It is set when A/D conversion starts and cleared when it ends.

- Writing:

Write "0" to this bit during A/D operation to forcibly terminate operation. Use this bit for forcible termination in continuous mode and stop mode.

"1" cannot be written to the bit that indicates whether the A/D converter is operating. For a read by a RMW instruction, "1" is read. In single-shot mode, this bit is cleared when A/D conversion ends. In continuous mode and stop mode, this bit is not cleared until "0" is written to terminate operation.

This bit is initialized to "0" by a reset.

Do not forcibly terminate operation and start software at the same time (BUSY=0, STRT=1).

[bit14] INT (INTerrupt): Data display

It is set when conversion data is written to the ADCR. (It is when a conversion ends single conversion mode or when conversion of all channels ends in scan conversion mode.)

If this bit is set while INTE (bit13) is set to "1", an interrupt request occurs. If start of DMA transfer has been selected, DMA is started. Writing "1" to this bit is meaningless.

This bit is cleared if "0" is written to it or a clear signal from DMAC is received.

Note:

Clear this bit by writing "0" to it while the A/D is stopped. This bit is initialized to "0" by a reset. For a read by a read-modify-write instruction, "1" is read.

[bit13] INTE (INTerrupt Enable): Specifying interrupt by conversion termination

This bit specifies enabling or disabling of interrupts when conversion is completed.

Table 12.2-1 shows specifying interrupt by the conversion termination.

Table 12.2-1 Function of Specifying Interrupt by Conversion Termination

INTE	Function
0	Interrupt disabled (initial value)
1	Interrupt enabled

To start DMA transfer occurring due to an interrupt, set this bit. This bit is initialized to "0" by a reset.

[bit12] CRF (Convert Run Flag): A/D converting status

This bit shows A/D converting status.

This bit is read only.

Table 12.2-2 shows the function of A/D converting.

Table 12.2-2 Function of A/D Converting

CRF	Function
0	Stop (initial value)
1	Now converting

Note:

Do not change analog input value during A/D converting.

[bit11, bit10] STS1, STS0 (STart Source select)

These bits are initialized to " 00_B " by a reset. Set these bits to select the source of starting A/D conversion. Table 12.2-3 shows the possible settings.

Table 12.2-3 Settings of A/D Conversion Start Causes

STS1	STS0	Functions
0	0	Started due to software
0	1	Started due to an external pin trigger or software
1	0	Started due to a reload timer or software
1	1	Started due to an external pin trigger, reload timer, or software

In a mode with multiple start sources, the first detected source starts the A/D conversion.

Notes:

Since start sources change at the same time that rewriting occurs, be careful when this bit is rewritten during the A/D conversion operation.

- An external pin trigger is detected at a falling edge. If this bit is rewritten to select starting due to an external trigger while the external trigger input level is set to "L", the A/D converter may be started.
 - While a timer is selected, a rising edge of the reload timer channel 2 is selected. If this bit is rewritten to select starting due to a timer while the reload timer output level is set to "H", the A/D converter may be started.
-

[bit9] STRT (STaRT)

Write "1" to this bit to start the A/D converter. If the system is restarted, write to this bit again. In stop mode, restart is disabled because of the nature of the function.

This bit is initialized to "0" by a reset.

Do not forcibly terminate operation and start a software program at the same time (BUSY=0, STRT=1).

For a read by a read-modify-write instruction, "0" is read.

Wait for at least 10ms until external capacitor is charged at power on, after reset, or after returning from stop mode.

[bit8] Test bit

This bit is used for testing. For a write, write "0".

[bit7, bit6] MD1,MD0 (A/D converter MoDe set)

These bits select the operating mode.

Table 12.2-4 shows the settings for the operating modes.

Table 12.2-4 Operating Mode Settings

MD1	MD0	Function
0	0	Restart enabled both in single-shot mode and during operation
0	1	Restart disabled both in single-shot mode and during operation
1	0	Restart disabled both in continuous mode and during operation
1	1	Restart disabled both in stop mode and during operation

- Single-shot mode: Performs A/D conversion from the channels specified by ANS2 to ANS0 to the channels specified by ANE2 to ANE0. Stops when one conversion session is completed.
- Continuous mode: Repeatedly performs A/D conversion from the channels specified by ANS2 to ANS0 to the channels specified by ANE2 to ANE0. However, when the interrupt is enabled (INTE = 1), a conversion for all setting channel of ANS2 to ANS0 is completed and, temporarily stops until the interrupt is cleared. Clearing the interrupt restarts the conversion.
- Stop mode: Performs A/D conversion from each of the channels specified by ANS2 to ANS0 to each of the channels specified by ANE2 to ANE0 and then temporarily stops. The conversion is restarted when a start source occurs.

Note:

When the A/D conversion mode selection bits (MD1 and MD0) are set to "00_B", restart during A/D conversion is enabled. In this modes, software start (STS1 and STS0 = "00_B") can only be set. Restart in the following procedures.

- (1) Clear the INTE bit to "0".
- (2) Write the STRT bit to "1" and the INTE bit to "0" at the same time.

When the A/D conversion mode selection bits (MD1 and MD0) are set to "01_B", restart during A/D conversion is disabled. When restarting and ending of A/D conversion occurred simultaneously, restart operation is not executed, A/D conversion is terminated, and "300_H" is stored in the data register (ADCR1/ADCR0). Therefore, perform restart operation not to generate restarting and ending of A/D conversion simultaneously.

Notes:

- If A/D conversion is started in continuous or stop mode, the conversion operation continues until it is stopped due to the BUSY bit.
- Write "0" to the BUSY bit to stop A/D conversion.
- The restart disabled status in each of the single, continuous, and stop modes applies to all the start operation caused by a timer, external trigger, and software.

[bit5, bit4, bit3] ANS2, ANS1, ANS0 (ANalog Start channel set):Setting of A/D conversion start channel

These bits set the channel where A/D conversion will start.

When the A/D converter is started, A/D conversion starts at the channel selected in these bits.

Table 12.2-5 shows the settings for the A/D conversion start channels.

Table 12.2-5 Settings for A/D Conversion Start Channels

ANS2	ANS1	ANS0	Start channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	X	X	Setting disabled

- During reading: A conversion channel is read from these bits during A/D conversion.
The initial value of register is read in the stop state.
- During reset: These bits are initialized to "000_B" by a reset.

Note:

Write "0" when the ANS2 bit is written.

[bit2, bit1, bit0] ANE2, ANE1, ANE0 (ANalog End channel set) Setting of A/D conversion end channel

These bits set an A/D conversion end channel.

Table 12.2-6 shows the settings for the A/D conversion end channels.

Table 12.2-6 Settings for A/D Conversion End Channels

ANE2	ANE1	ANE0	End channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	x	x	Setting disabled

- Set the same channel as specified by ANS2 to ANS0 to perform one-channel conversion (single-shot conversion).
- If continuous or stop mode is set, A/D conversion returns to the start channel specified by ANS2 to ANS0 when the conversion for the channel specified by these bits is completed.
- Define channels ANS and ANE so that the ANS is equal to or less than the ANE.
- These bits are initialized to "000_B" by a reset.

Notes:

- Write "0" when the ANE2 bit is written.
- After setting the start channel to the A/D conversion start channel selection bit (ANS2, ANS1, ANS0), please set neither the A/D conversion mode selection bit (MD1, MD0) nor the A/D conversion end channel selection bit (ANE2, ANE1, ANE0) by the read-modify-write type instruction.

The last conversion channel is read from the ANS2, ANS1, ANS0 bits until the A/D conversion operating starts. Therefore, when the MD1, MD0 bits and ANE2, ANE1, ANE0 bits are set by the read-modify-write type instruction after setting the start channel to the ANS2, ANS1, ANS0 bits, the value of ANE2, ANE1, ANE0 bits may be re-written.

■ **Setting Example**

If the channel settings are ANS=1-channel and ANE=3-channel in single-shot mode:

The operation is performed for input channel (convert channels) to the A/D converter in the order of 1-channel, 2-channel, and 3-channel.

12.2.2 Data Register (ADCR)

The data register (ADCR) stores the A/D conversion result. A digital value is stored as the current result of conversion.

■ Data Register (ADCR)

Figure 12.2-3 shows the bit configuration of the data register (ADCR).

Figure 12.2-3 Bit Configuration of the Data Register (ADCR)

Address: 000078 _H	bit 15	14	13	12	11	10	9	8	Initial value
	-	-	-	-	-	-	D9	D8	000000XX _B
	R	R	R	R	R	R	R	R	

bit	7	6	5	4	3	2	1	0	Initial value
	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX _B
	R	R	R	R	R	R	R	R	

The data register (ADCR), which is a current conversion storage register, stores a digital value that results from conversion.

The value in the data register (ADCR) is updated every time a conversion session is completed. Normally, this register stores the last conversion value.

This register is set to an undefined value by a reset. "0" is read from the high-order bits 15 to 10 bits during reading.

12.2.3 Conversion result register (ADCR0 to ADCR3)

The conversion result registers (ADCR0 to ADCR3) store the results of A/D conversion. The register stores the digital value resulting from conversion of the corresponding channel.

■ Conversion Result Register (ADCR0 to ADCR3)

Figure 12.2-4 shows the bit configurations of the conversion result registers (ADCR0 to ADCR3).

Figure 12.2-4 Bit Configuration of the Conversion Result Register (ADCR0 to ADCR3)

bit	7	6	5	4	3	2	1	0	Initial value
Address: 00007C _H	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX _B
00007D _H									
00007E _H	R	R	R	R	R	R	R	R	
00007F _H									

The conversion result registers (ADCR0 to ADCR3) are conversion storage registers that store the digital values resulting from conversion of their respective channels. The conversion result is stored in the upper eight bits in the conversion result register (ADCR).

The values in the conversion result registers (ADCR0 to ADCR3) are updated upon completion of each conversion session conversion of their respective channels. The register usually contains the final value converted.

12.3 A/D Converter Operation

The A/D converter operates using the successive approximation conversion method and has a 10-bit resolution.

Upon completion of each conversion, this A/D converter stores the upper eight bits in the 8-bit conversion result register (ADCR0 to ADCR3) for the corresponding channel. To read a conversion result, read the corresponding conversion result register (ADCR0 to ADCR3).

The A/D converter has three modes: single-shot conversion mode, continuous conversion mode, and stop conversion mode. This section describes the operation of these modes.

■ Single-shot Conversion Mode

This mode sequentially converts the analog input specified by the ANS and ANE bits and stops operation after performing conversion up to the end channel specified by the ANE bit.

The one-channel conversion operation occurs when the start and end channels are the same (ANS=ANE).

Example:

If ANS=000_B, ANE=011_B:

Beginning --> AN0 --> AN1 --> AN2 --> AN3 --> End

If ANS=010_B, ANE=010_B:

Beginning --> AN2 --> End

Note:

When the A/D conversion mode selection bits (MD1 and MD0) are set to "00_B", restart during A/D conversion is enabled. In this mode, software start (STS1 and STS0 = 00_B) can only be set. Restart in the following procedures.

- (1) Clear the INTE bit to "0".
- (2) Write the STRT bit to "1" and the INTE bit to "0" at the same time.

When the A/D conversion mode selection bits (MD1 and MD0) are set to "01_B", restart during A/D conversion is disabled. When restarting and ending of A/D conversion occurred simultaneously, restart operation is not executed, A/D conversion is terminated, and "300_H" is stored in the data register (ADCR1/ADCR0). Therefore, perform restart operation not to generate restarting and ending of A/D conversion simultaneously.

■ Continuous Conversion Mode

This mode sequentially converts the analog input defined by the ANS and ANE bits, returns to the analog input of ANS after performing the conversion up to the end channel defined by the ANE bit, and continues the A/D conversion operation.

The one-channel conversion operation is continued if the start and end channels are the same (ANS=ANE).

Example:

If ANS=000_B, ANE=011_B:

Beginning --> AN0 --> AN1 --> AN2 --> AN3 --> AN0 -->-->-->--> Repeated

If ANS=010_B, ANE=010_B:

Beginning --> AN2 --> AN2 --> AN2 -->-->-->--> Repeated

Continuous conversion mode continues to repeatedly perform conversion until "0" is written to the BUSY bit. Write "0" to the BUSY bit to forcibly terminate operation. Be careful when you forcibly terminate the operation because the conversion in progress is stopped before it is completed. If operation is forcibly terminated, the conversion register holds the previous data that has been converted.

When interrupts are enabled (INTE=1), the A/D converter is suspended until the interrupt is cleared after conversion of all the channels selected from among ANS2 to ANS0 is completed once. Clearing the interrupt resumes the conversion.

■ Stop Conversion Mode

This mode sequentially converts the analog input specified by the ANS and ANE bits and temporarily stops operation each time conversion has been performed for one channel. To clear the temporary stop, start A/D conversion again.

This mode returns to the analog input of ANS after performing conversion up to the end channel specified by the ANE bit and then continues the A/D conversion operation.

The one-channel conversion operation is performed if the start and end channels are the same (ANS=ANE).

Example:

ANS=000_B, ANE=011_B

Beginning -->AN0 --> Stop --> Start --> AN1 --> Stop --> Start --> AN2 --> Stop -->
Start --> AN3 --> Stop --> Start -->AN0 -->-->-->--> Repeated

If ANS=010_B, ANE=010_B:

Beginning --> AN2 --> Stop --> Start --> AN2 --> Stop --> Start --> AN2
-->-->-->--> Repeated

Only start sources specified by STS1 and STS0 are used at this time.

Use this mode to synchronize the beginning of conversion.

12.4 Precautions on the Using A/D Converter

This section contains precautions on using the A/D converter.

■ Precautions on Using the A/D Converter

To start the A/D converter using an external trigger or an internal timer, set the A/D start source bits (STS1 and STS0) of the ADCS register. At this time, the input value of an external trigger or an internal timer must be set to inactive. If it is set to active, a malfunction occurs.

If STS1 and STS0 are set, set ATG=1 input and reload timer (channel 1)=0 output.

A correct conversion result will not be obtained if the external impedance exceeds the specified value, since then the analog input value cannot be sampled within the specified sampling time.

○ Restart of the A/D conversion

When the A/D conversion mode selection bits (MD1 and MD0) are set to " 00_B ", restart during A/D conversion is enabled. In this mode, software start (STS1 and STS0 = 00_B) can only be set. Restart in the following procedures.

- (1) Clear the INTE bit to "0".
- (2) Write the STRT bit to "1" and the INTE bit to "0" at the same time.

When the A/D conversion mode selection bits (MD1 and MD0) are set to " 01_B ", restart during A/D conversion is disabled. When restarting and ending of A/D conversion occurred simultaneously, restart operation is not executed, A/D conversion is terminated, and " 300_H " is stored in the data register (ADCR1/ADCR0). Therefore, perform restart operation not to generate restarting and ending of A/D conversion simultaneously.

CHAPTER 13 UART

This chapter describes the overview of the UART, the configuration and functions of registers, and UART operation.

- 13.1 Overview of the UART
- 13.2 UART Registers
- 13.3 UART Operation
- 13.4 Example of Using the UART
- 13.5 Example of Setting Baud Rates and U-TIMER Reload Values

13.1 Overview of the UART

The UART is a serial I/O port used to perform asynchronous (start-stop synchronization) communication and CLK synchronous communication.

The MB91301 series has three UART channels.

■ Features of the UART

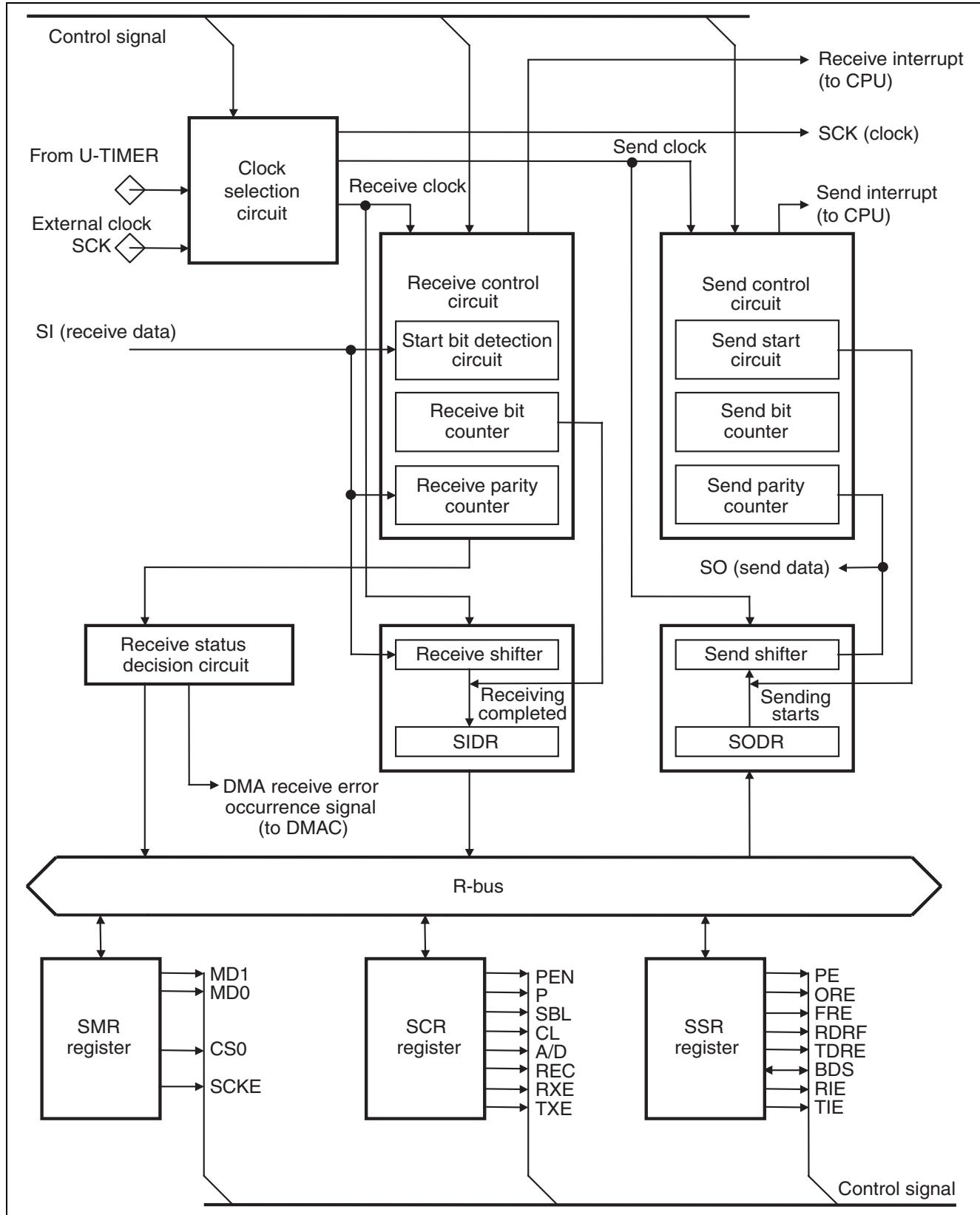
The UART has the following features:

- Full-duplex double buffer
- Either asynchronous (start-stop synchronization) or CLK synchronous communication can be selected.
- Multiprocessor mode is supported.
- Fully programmable baud rate: An arbitrary baud rate can be set using a built-in timer.
(See "CHAPTER 8 U-TIMER".)
- An external clock can be used to set a baud rate.
- Error detection functions (parity, framing, overrun)
- The transfer signal is an NRZ code.
- DMA transfer is started as the result of an interrupt.

■ Block Diagram

Figure 13.1-1 shows a block diagram of the UART.

Figure 13.1-1 Block Diagram of the UART



13.2 UART Registers

This section describes the configuration and functions of the registers used by the UART.

■ UART Registers

Figure 13.2-1 shows the configuration of the UART registers and Figure 13.2-2 lists the UART registers.

Figure 13.2-1 Configuration of UART Registers

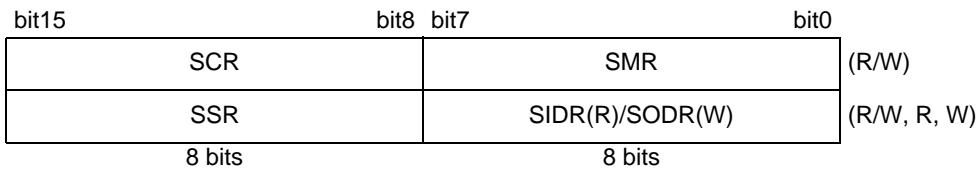


Figure 13.2-2 UART Registers

bit	7	6	5	4	3	2	1	0	
	D7	D6	D5	D4	D3	D2	D1	D0	Serial input register Serial output register (SIDR /SODR)
bit	7	6	5	4	3	2	1	0	
	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Serial status register (SSR)
bit	7	6	5	4	3	2	1	0	
	MD1	MD0	-	-	CS0	-	SCKE	-	Serial mode register (SMR)
bit	7	6	5	4	3	2	1	0	
	PEN	P	SBL	CL	A/D	REC	RXE	TXE	Serial control register (SCR)

13.2.1 Serial Mode Register (SMR)

The serial mode register (SMR) specifies the UART operating mode.

Set an operating mode while operation is stopped. Do not write to this register while operation is in progress.

■ Bit Configuration of Serial Mode Register (SMR)

Figure 13.2-3 shows the bit configuration of the serial mode register (SMR).

Figure 13.2-3 Bit Configuration of the Serial Mode Register (SMR)

Address:	bit 7	6	5	4	3	2	1	0	Initial value
000063H (ch.0)	MD1	MD0	-	-	CS0	-	SCKE	-	00--0-0-B
00006BH (ch.1)	R/W	R/W			W		R/W		
000073H (ch.2)									

■ Detailed Bit of Serial Mode Register (SMR)

The following describes each bit function of the serial mode register (SMR).

[bit7, bit6] MD1, MD0 (MoDe select): Setting of operating mode

These bits select a UART operating mode.

Table 13.2-1 shows the settings for the operating modes.

Table 13.2-1 Settings for Operating Modes

Mode	MD1	MD0	Operating mode
0	0	0	Asynchronous (start-stop synchronization) normal mode [initial value]
1	0	1	Asynchronous (start-stop synchronization) multiprocessor mode
2	1	0	CLK synchronous mode
-	1	1	Setting disabled

Note:

In Mode 1, which is CLK asynchronous mode (multiprocessor), more than one slave CPV can be connected to one host CPU. Since this resource cannot identify the data format of received data, however, only the master in multiprocessor mode is supported. Because the parity check function cannot be used, set PEN of the SCR register to "0".

[bit5, bit4] (Reserved)

These bits are unused. Always write "1" to these bits.

[bit3] CS0 (Clock Select): Selection of operating clock

This bit selects the UART operating clock.

Table 13.2-2 shows the selection of the operating clock.

Table 13.2-2 Function for Selection of Operation Clock

CS0	Function
0	Built-in timer (U-TIMER) [initial value]
1	External clock

[bit2] (Reserved)

This bit is unused. Always write "0" to this bit.

[bit1] SCKE (SCLK Enable): Setting of SCK pin

This bit specifies whether the SCK pin is used as a clock input pin or a clock output pin when communication is performed in CLK synchronous mode (Mode 2).

Set this bit to "0" in CLK asynchronous mode or external clock mode.

Table 13.2-3 shows the setting of the SCK pin.

Table 13.2-3 Function for Setting of the SCK Pin

SCKE	Function
0	SCK pin serves as clock input pin. [initial value]
1	SCK pin serves as clock output pin.

Note:

When using the SCK pin as a clock input pin, set the CS0 bit to "1" to select external clock mode.

[bit0] (Reserved)

This bit is unused.

13.2.2 Serial Control Register (SCR)

The serial control register (SCR) controls the transfer protocol that is used for serial communication.

This section describes the configuration and functions of the serial control register (SCR).

■ Bit Configuration of Serial Control Register (SCR)

Figure 13.2-4 shows the bit configuration of the serial control register (SCR).

Figure 13.2-4 Bit Configuration of the Serial Control Register (SCR)

bit	7	6	5	4	3	2	1	0	Initial value
Address: 000062 _H (ch.0)	PEN	P	SBL	CL	A/D	REC	RXE	TXE	00000100 _B
00006A _H (ch.1)	R/W	R/W	R/W	R/W	R/W	W	R/W	R/W	
000072 _H (ch.2)									

■ Detailed Bit of Serial Control Register (SCR)

The following describes each bit function of the serial control register (SCR).

[bit7] PEN (Parity Enable): Setting of parity

This bit specifies whether data communication is performed to add parity in serial communication.

Table 13.2-4 shows the setting of the parity.

Table 13.2-4 Function for Setting of Parity

PEN	Function
0	No parity [initial value]
1	Parity

Note:

Parity can be added only in normal mode (Mode 0) of asynchronous (start-stop synchronization) communication mode. No parity can be added in multiprocessor mode (Mode 1) or CLK synchronous communication mode (Mode 2).

[bit6] P (Parity): Specifying of even/odd parity

This bit specifies that even or odd parity be added to perform data communication.

Table 13.2-5 shows specifying of even/odd parity.

Table 13.2-5 Function for Specifying of Even/Odd Parity

P	Function
0	Even parity [initial value]
1	Odd parity

[bit5] SBL (Stop Bit Length): Specifying of stop bit length

This bit specifies the stop bit length, which marks the end of a frame in asynchronous (start-stop synchronization) communication.

Table 13.2-6 shows specifying of the stop bit length.

Table 13.2-6 Function for specifying of Stop Bit Length

SBL	Function
0	1 stop bit [initial value]
1	2 stop bits

[bit4] CL (Character Length): Specifying of data length of one frame

This bit specifies the data length of one frame that is sent or received.

Table 13.2-7 shows specifying of the data length of one frame.

Table 13.2-7 Function for Specifying of Data Length of One Frame

CL	Function
0	7-bit data [initial value]
1	8-bit data

Note:

7-bit data can be handled only in normal mode (Mode 0) of asynchronous (start-stop synchronization) communication mode. Use 8-bit data in multiprocessor mode (Mode 1) or CLK synchronous communication mode (Mode 2).

[bit3] A/D (Address/Data): Specifying of data format of frame

This bit specifies the data format of a frame that is sent or received in multiprocessor mode (Mode 1) of asynchronous (start-stop synchronization) communication mode.

Table 13.2-8 shows specifying of the data format of frame.

Table 13.2-8 Function for Specifying of Data Format of Frame

A/D	Function
0	Data frame [initial value]
1	Address frame

[bit2] REC (Receiver Error Clear): Clearing of error flag

Write "0" to this bit to clear the error flags (PE, ORE, and FRE) in the SSR register.

Writing "1" to this bit has no effect. "1" is always read from this bit.

[bit1] RXE (Receiver Enable): Controlling of receive operation

This bit controls the UART receive operation.

Table 13.2-9 shows controlling of the receive operation.

Table 13.2-9 Function for Controlling of Recive Operation

RXE	Function
0	Disables receive operation. [initial value]
1	Enables receive operation.

Note:

If a receive operation is disabled while it is in progress (while data is being input to the receive shift register), reception of the frame is completed. The receive operation is stopped when the received data is stored in the receive data buffer serial input data register (SIDR).

[bit0] TXE (Transmitter Enable): Controlling of send operation

This bit controls the UART send operation.

Table 13.2-10 shows controlling of the send operation.

Table 13.2-10 Function for Controlling of Send Operation

TXE	Function
0	Disables send operation. [initial value]
1	Enables send operation.

Note:

If a send operation is disabled while it is in progress (while data is being output from the transmission register), sending is stopped when no more send data is stored in the send data buffer serial output data register (SODR).

13.2.3 Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)

These registers are data buffer registers for receiving and sending.

■ Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)

Figure 13.2-5 shows the bit configurations of the serial input data register (SIDR) and the serial output data register (SODR).

Figure 13.2-5 Bit Configurations of the Serial Input Data Register (SIDR) and the Serial Output Data Register (SODR)

SIDR									
bit	7	6	5	4	3	2	1	0	Initial value
Address: 000061 _H (ch.0)	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX _B
000069 _H (ch.1)	R	R	R	R	R	R	R	R	
000071 _H (ch.2)									

SODR									
bit	7	6	5	4	3	2	1	0	Initial value
Address: 000061 _H (ch.0)	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX _B
000069 _H (ch.1)	W	W	W	W	W	W	W	W	
000071 _H (ch.2)									

If the data length is 7 bits, bit7 (D7) of SIDR and SODR is invalid data. Write to the SODR register only while the TDRE bit of the SSR register is set to "1".

Note:

Writing to the register with this address means writing to the SODR register. Reading from the register with this address means reading from the SIDR register.

13.2.4 Serial Status Register (SSR)

The serial status register (SSR) consists of flags that indicate the operation state of the UART.

This section describes the configuration and functions of the serial status register (SSR).

■ Bit Configuration of Serial Status Register (SSR)

Figure 13.2-6 shows the bit configuration of the serial status register (SSR)

Figure 13.2-6 Bit Configuration of the Serial Status Register (SSR)

bit	7	6	5	4	3	2	1	0	Initial value
Address: 000060 _H (ch.0)	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	00001000 _B
000068 _H (ch.1)	R	R	R	R	R	R/W	R/W	R/W	
000070 _H (ch.2)									

■ Detailed Bit of Serial Status Register (SSR)

The following describes each bit function of the serial status register (SSR).

[bit7] PE (Parity Error): Presence or absence of parity error

This bit, which is an interrupt request flag, is set when a parity error occurs during receiving.

Table 13.2-11 shows the presence or absence of the parity error.

Table 13.2-11 Presence or Absence of Parity Error

PE	Function
0	No parity error has occurred. [initial value]
1	A parity error has occurred.

To clear the flag when it has been set, write "0" to the REC bit (bit10) of the SCR register.

If the PE bit is set, the SIDR data becomes invalid.

[bit6] ORE (Over Run Error): Presence or absence of overrun error

This bit, which is an interrupt request flag, is set when an overrun error occurs during reception.

Table 13.2-12 shows the presence or absence of the overrun error.

Table 13.2-12 Presence or Absence of Overrun Error

ORE	Function
0	No overrun error has occurred. [initial value]
1	An overrun error has occurred.

To clear the flag when it has been set, write "0" to the REC bit of the SCR register.

If the ORE bit is set, the SIDR data becomes invalid.

[bit5] FRE (FRaming Error): Presence or absence of framing error

This bit, which is an interrupt request flag, is set when a framing error occurs during reception.

Table 13.2-13 shows the presence or absence of the framing error.

Table 13.2-13 Presence or Absence of Framing Error

FRE	Function
0	No framing error has occurred. [initial value]
1	A framing error has occurred.

To clear the flag when it has been set, write "0" to the REC bit of the SCR register.

If the FRE bit is set, the SIDR data becomes invalid.

Note:

Switch the internal and external baud rate clocks using bit3 of the serial mode register only while the UART is stopped, since the switching takes effect immediately after writing.

Bit3 of the serial mode register is write-only.

[bit4] RDRF (Receiver Data Register Full): Presence or absence of receive data

This bit, which is an interrupt request flag, indicates that the SIDR register has receive data.

Table 13.2-14 shows the presence or absence of receive data.

Table 13.2-14 Presence or Absence of Receive Data

RDRF	Function
0	No receive data exists. [initial value]
1	Receive data exists.

This bit is set when receive data is loaded into the SIDR register. It is automatically cleared when the data is read from the SIDR register.

[bit3] TDRE (Transmitter Data Register Empty): Writing of send data

This bit, which is an interrupt request flag, indicates whether send data can be written to SODR.

Table 13.2-15 shows the function for writing of the send data.

Table 13.2-15 Function for Writing of Send Data

TDRE	Function
0	Disables writing of send data.
1	Enables writing of send data. [initial value]

This bit is cleared when send data is written to the SODR register. It is set again when the written data is loaded into the send shifter and begins to be transferred, indicating that the next send data can be written.

[bit2] BDS (Bit Direction Select): Transfer direction selection

This bit is transfer direction selection bit.

Table 13.2-16 shows the transfer direction selection.

Table 13.2-16 Transfer Direction Selection

BDS	Function
0	Transfer starting from the least significant bit. (LSB) [initial value]
1	Transfer starting from the most significant bit. (MSB)

Note:

When the serial data register is read or written to data, the high-order and low-order sides are exchanged with each other. If you update this bit after writing data to the SDR register, therefore, the data is made invalid.

If "1" is written to the BDS bit and transmit data is written to the serial output data register (SODR) at the same time after halfword access to the serial status register (SSR), the BDS bit setting for transmit data is ignored.

To switch between the MSB/LSB transfer directions, set the BDS bit before writing data to the SODR.

[bit1] RIE (Receiver Interrupt Enable): Receive interrupt

This bit controls a reception interrupt.

Table 13.2-17 shows the function for controlling the receive interrupt.

Table 13.2-17 Function for Controlling Receive Interrupt

RIE	Function
0	Disables receive interrupts. [initial value]
1	Enables receive interrupts.

Note:

Receive interrupt sources include errors due to PE, ORE, and FRE as well as normal receive due to RDRF.

[bit0] TIE (Transmitter Interrupt Enable): Control send interrupt

This bit controls send interrupts.

Table 13.2-18 shows the function for controlling the send interrupt.

Table 13.2-18 Function for Controlling Send Interrupt

TIE	Function
0	Disables send interrupts. [initial value]
1	Enables send interrupts.

Note:

Send interrupt sources include send requests due to TDRE.

13.3 UART Operation

The UART has two operating modes: asynchronous (start-stop synchronization) mode and CLK synchronous mode.

Asynchronous (start-stop synchronization) mode consists of normal and multiprocessor modes.

This section describes the operation of these operating modes.

■ UART Operating Modes

The UART has the operating modes shown in Table 13.3-1. Set a value in the SMR and SCR registers to switch mode.

Table 13.3-1 UART Operating Modes

Mode	Parity	Data length	Operating mode	Stop bit length
0	Yes/No	7	Asynchronous (start-stop synchronization) normal mode	1 bit or 2 bits
	Yes/No	8		
1	No	8+1	Asynchronous (start-stop synchronization) multiprocessor mode	
2	No	8	CLK synchronous mode	No

Note:

The stop bit length in asynchronous (start-stop synchronization) mode can be specified only for a send operation. The stop bit length is always one bit for a receive operation. Since operation is possible only in the above modes, do not make any other setting.

■ Selecting a Clock for the UART

○ Internal timer

If you select the U-TIMER by setting CS0 to "0", the baud rate is determined according to the reload value set for the U-TIMER. At this time, you can calculate the baud rate as follows:

$$\text{Asynchronous (start-stop synchronization): } \phi/(16 \times \beta)$$

$$\text{CLK synchronous: } \phi/\beta$$

ϕ : Peripheral machine clock frequency (CLKP)

β : Cycle specified for the U-TIMER (2n+2 or 2n+3, or n is the reload value.)

In asynchronous (start-stop synchronization) mode, data can be transferred in the range from -1% to +1% of the specified baud rate.

○ External clock

If you select an external clock by setting CS0 to "1", the baud rate is as follows (the frequency of the external clock is assumed to be f):

$$\text{Asynchronous (start-stop synchronization): } f/16$$

$$\text{CLK synchronous: } f$$

However, the maximum value for f is 3.125 MHz.

13.3.1 Asynchronous (Start-stop Synchronization) Mode

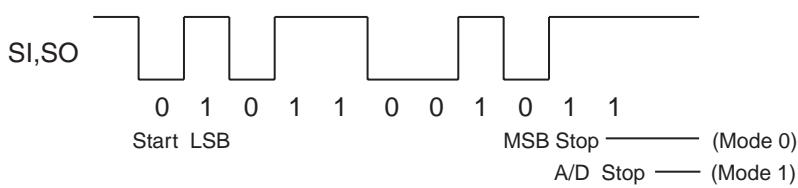
When the UART is used in operating mode 0 (normal mode) or operating mode 1 (multiprocessor mode), the asynchronous transfer method is used.

■ Transfer Data Format

UART handles only data in the NRZ (Non Return to Zero) format.

Figure 13.3-1 shows the data format.

Figure 13.3-1 Transfer Data Format (Modes 0 and 1)



Data that has been transferred is 01001101_B.

As shown in Figure 13.3-1, the transfer of data always starts with the start bit ("L" level data), transfers the data bit length specified in LSB first, and ends with a stop bit ("H" level data). If an external clock is selected, always input a clock.

The data length can be set to 7 bits or 8 bits in normal mode (Mode 0), but must be set to 8 bits in multiprocessor mode (Mode 1). In multiprocessor mode, no parity can be added; instead, the A/D bit is always added.

■ Receive Operation

If the RXE bit (bit1) of the SCR register is set to "1", a receive operation is always in progress.

If a start bit appears on the receive line, one-frame data is received according to the data format specified in the SCR register. If an error occurs after reception of one frame is completed, the error flag is set and then the RDRF flag (bit4 of the SSR register) is set. If, at this time, the RIE bit (bit1) of the same SSR register is set to "1", a receive interrupt is generated for the CPU. Check the flags of the SSR register and read the SIDR register if normal reception has occurred or perform the necessary processing if an error has occurred.

The RDRF flag is cleared when the SIDR register is read.

■ Send Operation

If the TDRE flag (bit3) of the SSR register is set to "1", send data is written to the SODR register. If, at this time, the TXE bit (bit0) of the SCR register is set to "1", transmission occurs.

The TDRE flag is set again when the data set in the SODR register is loaded into the send shift register and begins to be transferred, indicating that the next send data can be set. If, at this time, the TIE bit (bit0) of the same SSR register is set to "1", a send interrupt requesting that the send data be set in the SODR register is generated for the CPU.

The TDRE flag is cleared if data is set in the SODR register.

13.3.2 CLK Synchronous Mode

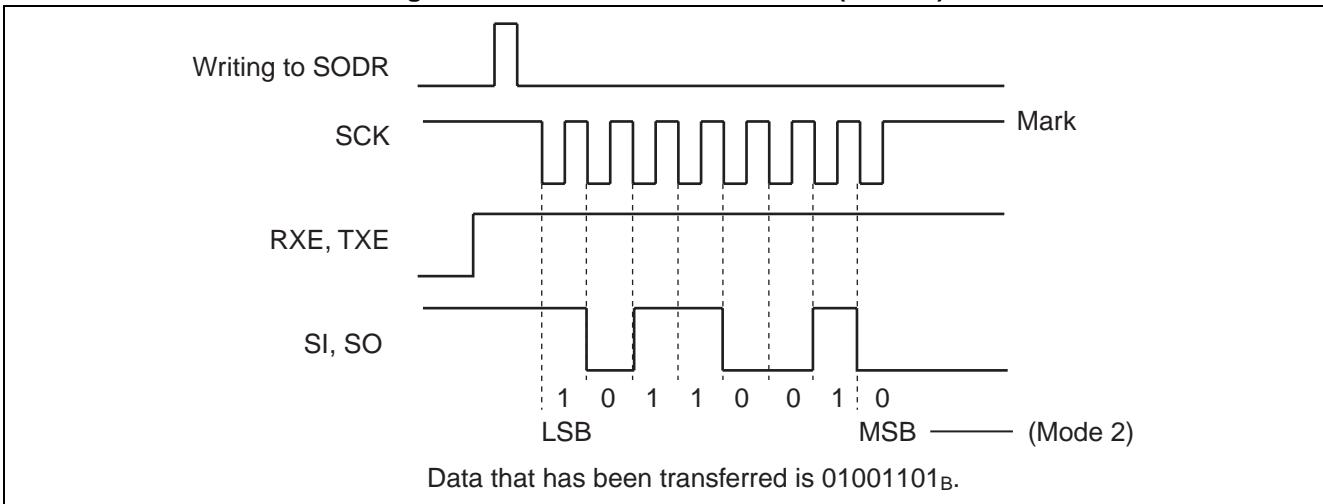
If the UART is used in operating mode 2, the clock synchronous transfer method is used.

■ Transfer Data Format

The UART handles only data in the NRZ (Non Return to Zero) format.

Figure 13.3-2 shows the relationship between send and receive clocks and data.

Figure 13.3-2 Transfer Data Format (Mode 2)



When the internal clock (U-TIMER) has been selected, a data receive synchronous clock is automatically generated as soon as data is sent. While an external clock has been selected, you must check that data exists in the send data buffer SODR register of the send side UART (TDRE flag is "0") and then supply an accurate clock for one byte. Before sending starts and after it ends, be sure to set the mark level.

The data length is 8 bits only, and no parity can be added. Only overrun errors are detected because there is no start or stop bit.

CHAPTER 13 UART

■ Initialization

The following shows the setting values of the control registers required to use CLK synchronous mode.

- SMR register
 - MD1, MD0: "10_B"
 - CS: Specifies the clock input.
 - SCKE: Set to "1" for an internal timer and to "0" for an external clock.
 - SOE: Set to "1" for send and to "0" for receive.
- SCR register
 - PEN: "0"
 - P,SBL,A/D: These bits are meaningless.
 - CL: "1"
 - REC: "0" (to initialize the register)
 - RXE, TXE: At least one of the bits must be set to "1".
- SSR register
 - RIE: Set to "1" to enable interrupts and to "0" to disable interrupts.
 - TIE: "0"

■ Start of Communication

Write to the SODR register to start communication.

If only reception is performed, dummy send data must be written to the SODR register.

■ End of Communication

Check for the end of communication by making sure that the RDRF flag of the SSR register has changed to "1". Use the ORE bit of the SSR register to check that communication has been performed correctly.

13.3.3 Occurrence of Interrupts and Timing for Setting Flags

The UART has five flags and two interrupt sources.

The five flags are PE, ORE, FRE, RDRF, and TDRE. PE means parity error, ORE means overrun error, and FRE means framing error. These flags are set when an error occurs during reception and are then cleared when "0" is written to REC of the SCR register. RDRF is set when receive data is loaded into the SIDR register and then cleared when data is read from the SIDR register. Mode 1 does not provide a parity detection function. Mode 2 does not provide a parity detection function and a framing error detection function. TDRE is set when the SODR register is empty, and writing to it is enabled and then cleared when data is written to the SODR register.

■ Occurrence of Interrupts and Timing for Setting Flags

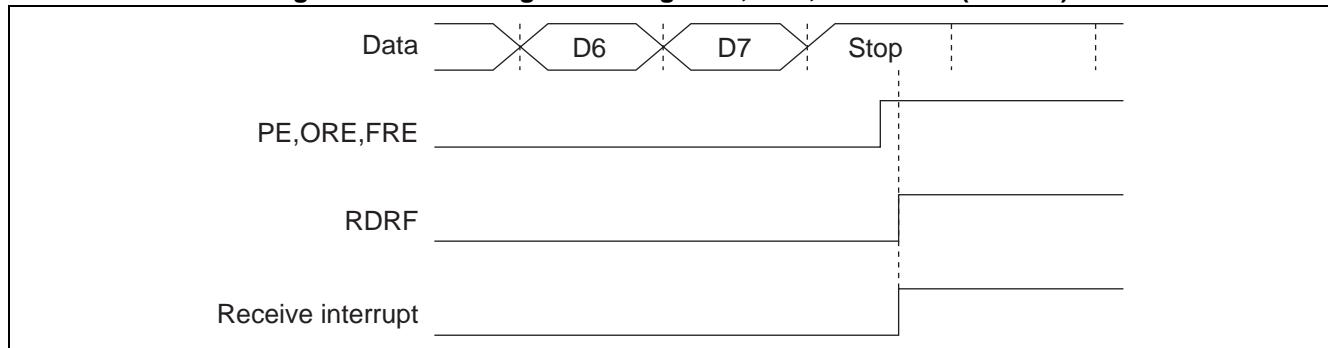
There are two interrupt sources, one for receiving and the other for sending. During receiving, an interrupt is requested due to PE, ORE, FRE, or RDRF. During sending, an interrupt is requested due to TDRE. The following shows the timing for setting the interrupt flags in each of these modes.

○ Receive operation in Mode 0

The PE, ORE, FRE, and RDRF flags are set when the last stop bit is detected after a receive transfer is completed, causing an interrupt request to be generated for the CPU. The SIDR data is invalid while PE, ORE, and FRE are active.

Figure 13.3-3 shows the timing for setting ORE, FRE, and RDRF in Mode 0.

Figure 13.3-3 Timing for Setting ORE, FRE, and RDRF (Mode 0)

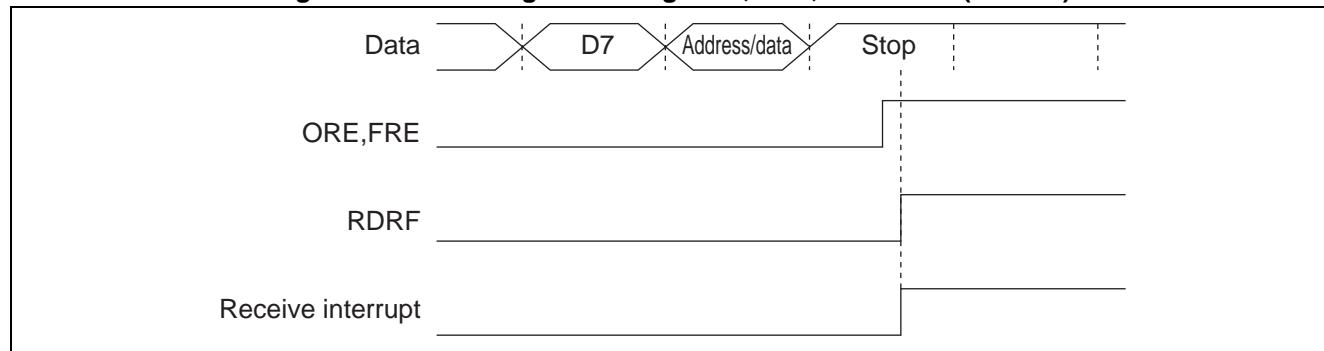


- **Receive operation in Mode 1**

The ORE, FRE, and RDRF flags are set when the last stop bit is detected after a receive transfer is completed, causing an interrupt request to be generated for the CPU. The data indicating an address or the data in last 9th bit is invalid because the length of data that can be received is 8 bits. The SIDR data is invalid while ORE and FRE are active.

Figure 13.3-4 shows the timing for setting ORE, FRE, and RDRF in Mode 1.

Figure 13.3-4 Timing for Setting ORE, FRE, and RDRF (Mode 1)

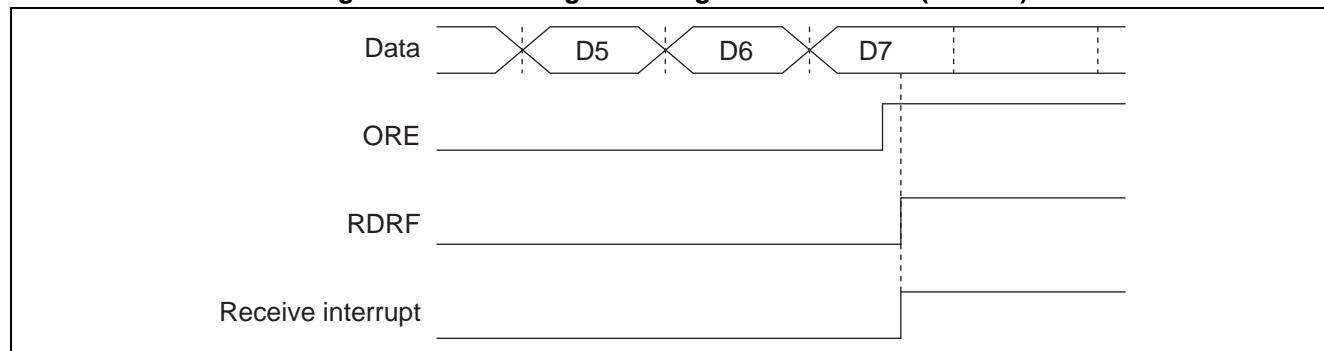


- **Reception operation in Mode 2**

The ORE and RDRF flags are set when the last data (D7) is set after the reception transfer is completed, generating an interrupt request to the CPU. The SIDR data is invalid while ORE is active.

Figure 13.3-5 shows the timing of setting ORE and RDRF in Mode 2.

Figure 13.3-5 Timing of Setting ORE and RDRF (Mode 2)



○ Send operation in modes 0, 1, and 2

TDRE is cleared when data is written to the SODR register. This bit is set when data is transferred to the internal shift register and the next data can be written, causing an interrupt request to be generated for the CPU. If "0" is written to TXE of the SCR register (as well as RXE in mode 2) during a send operation, TDRE of the SSR register is set to "1", disabling the UART send operation after the transmission shifter stops. The device sends data written to the SODR register before transmission stops after "0" is written to the TXE of the SCR register (as well as RXE in mode 2) during the send operation.

Figure 13.3-6 shows the timing for setting TDRE in modes 0 and 1. Figure 13.3-7 shows the timing for setting TDRE in mode 2.

Figure 13.3-6 Timing for Setting TDRE (Modes 0 and 1)

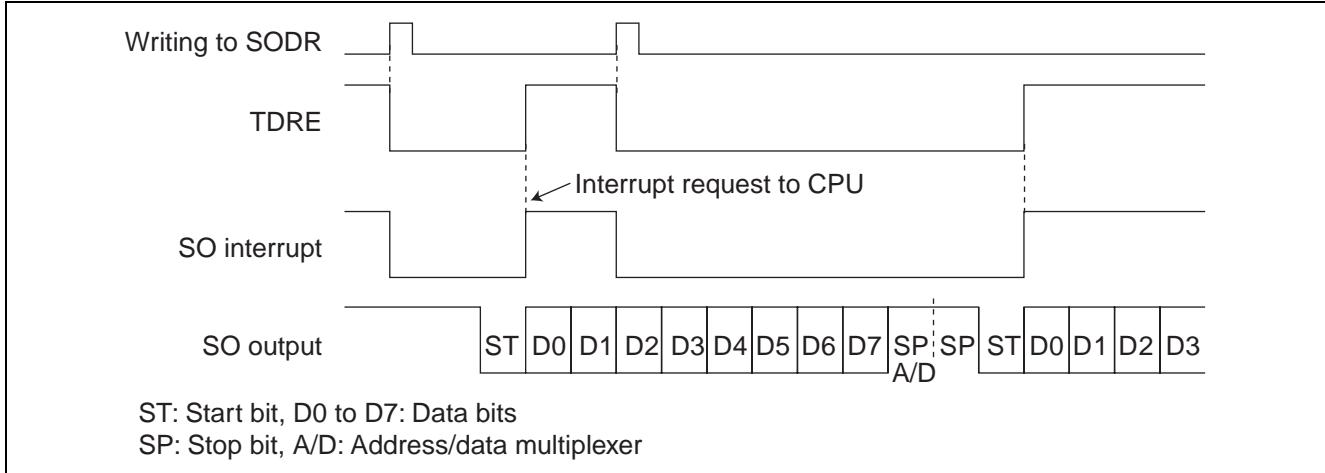
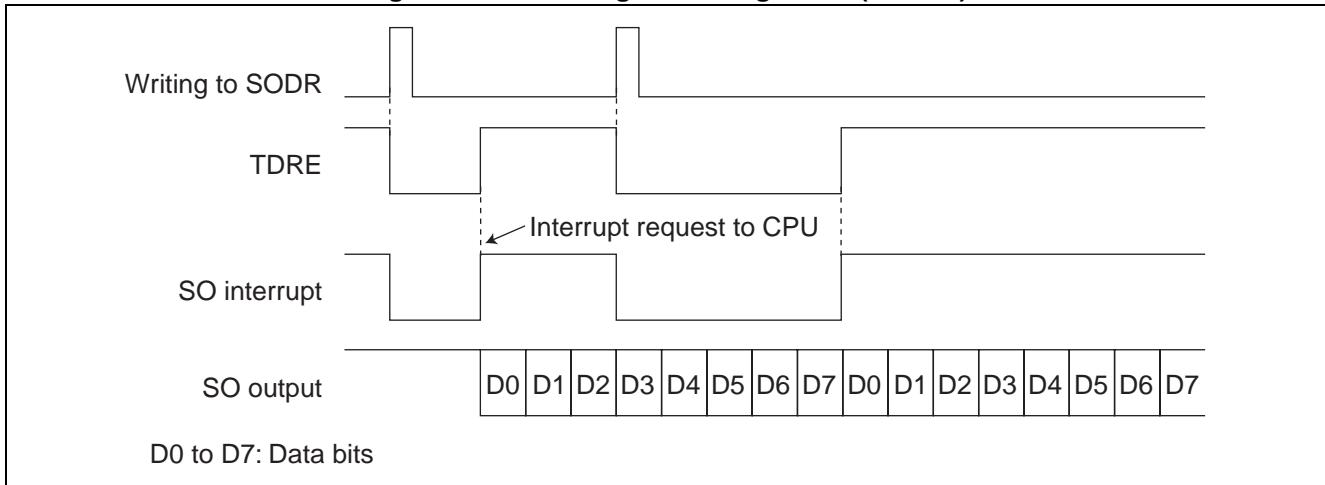


Figure 13.3-7 Timing for Setting TDRE (Mode 2)



■ Precautions on Usage

Writing to the serial output data register (SODR) starts communication.

Even for reception only, be sure to write false transmit data to the serial output data register (SODR).

Set a communication mode when operation has stopped. Data sent and received while a mode is set is not guaranteed.

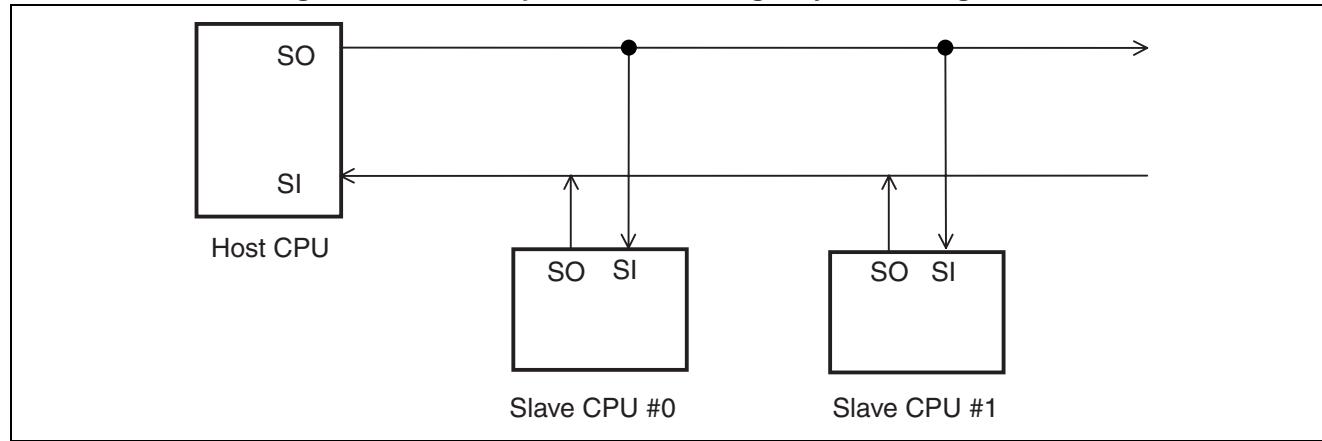
13.4 Example of Using the UART

This section provides an example of using the UART. Mode 1 is used if more than one slave CPU is connected to a single host CPU.

■ Example of Using the UART

Figure 13.4-1 shows an example of constructing a system using mode 1. This resource supports only a communication interface on the host.

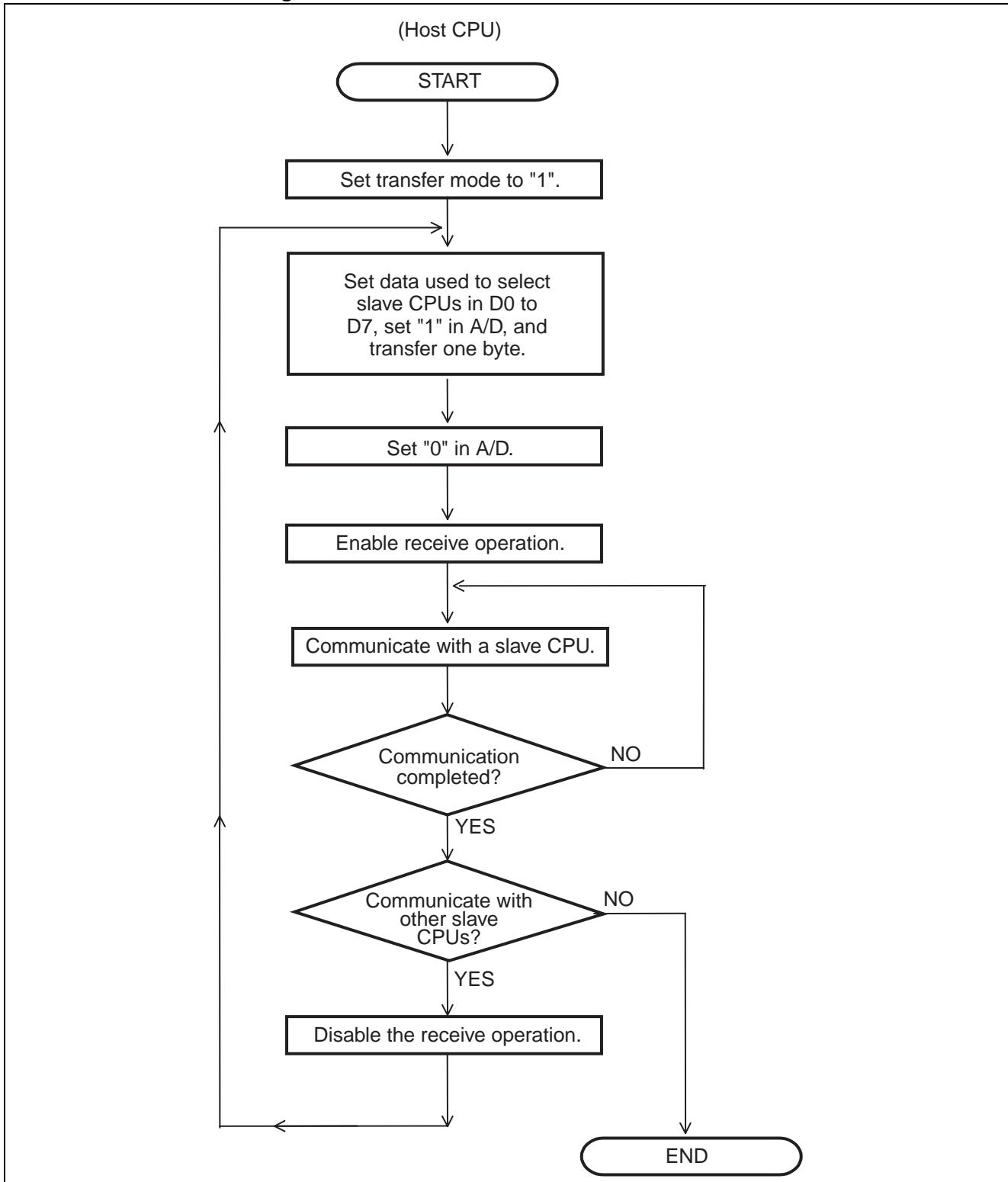
Figure 13.4-1 Example of Constructing a System Using Mode 1



Communication starts when the host CPU transfers address data. Address data is data used when A/D of the SCR register is set to "1". This data is used to select a destination slave CPU, enabling communication with the host CPU. Normal data is data used when A/D of the SCR register is set to "0". Figure 13.4-2 shows the flowchart.

In this mode, set the PEN bit of the SCR register to "0", since the parity check function cannot be used.

Figure 13.4-2 Communication Flowchart in Mode 1



13.5 Example of Setting Baud Rates and U-TIMER Reload Values

This section provides an example of setting baud rates and U-TIMER reload values.

■ Example of Setting Baud Rates and U-TIMER Reload Values

Table 13.5-1 shows setting values to be used in asynchronous (start-stop synchronization) mode. Table 13.5-2 shows setting values to be used in CLK synchronous mode.

A frequency in the tables represents a peripheral machine clock frequency. UCC1 is a value to be set in the UCC1 bit of the UTIMC register of the U-TIMER. A dash (-) in the tables means that the baud rate cannot be used because the error exceeds plus minus 1%.

Table 13.5-1 Setting Values in Asynchronous (Start-Stop Synchronization) Mode

Baud rate (bps)	ms	34MHz	20MHz	17MHz	10MHz
1200	833.33	884(UCC1=1)	520(UCC1=0)	441(UCC1=1)	259(UCC1=1)
2400	416.67	441(UCC1=1)	259(UCC1=1)	220(UCC1=0)	129(UCC1=0)
4800	208.33	220(UCC1=1)	129(UCC1=0)	109(UCC1=0)	64(UCC1=0)
9600	104.17	109(UCC1=1)	64(UCC1=0)	54(UCC1=1)	31(UCC1=1)
19200	52.08	54(UCC1=1)	31(UCC1=1)	26(UCC1=1)	-
38400	26.04	26(UCC1=1)	-	-	-
57600	17.36	17(UCC1=1)	-	-	-
10400	96.15	101(UCC1=0)	59(UCC1=0)	50(UCC1=0)	29(UCC1=0)
31250	32.00	33(UCC1=0)	19(UCC1=0)	16(UCC1=0)	9(UCC1=0)
62500	16.00	16(UCC1=0)	9(UCC1=0)	7(UCC1=1)	4(UCC1=0)

Table 13.5-2 Setting Values in CLK Synchronous Mode

Baud rate (bps)	ms	34MHz	20MHz	17MHz	10MHz
250K	4.00	67(UCC1=0)	39(UCC1=0)	33(UCC1=0)	19(UCC1=0)
500K	2.00	33(UCC1=0)	19(UCC1=0)	16(UCC1=1)	9(UCC1=0)
1M	1.00	16(UCC1=0)	9(UCC1=0)	7(UCC1=0) *	4(UCC1=0)

*: An error exceeding plus minus 1% occurs.

CHAPTER 14 DMA CONTROLLER (DMAC)

This chapter describes the overview of the DMA controller (DMAC), the configuration and functions of registers, and DMAC operation.

- 14.1 Overview of the DMA Controller (DMAC)
- 14.2 DMA Controller (DMAC) Registers
- 14.3 DMA Controller (DMAC) Operation
- 14.4 Operation Flowcharts
- 14.5 Data Bus
- 14.6 DMA External Interface

14.1 Overview of the DMA Controller (DMAC)

The DMA controller (DMAC) is a module that implements DMA (Direct Memory Access) transfer on FR family devices. When this module is used to control DMA transfer, various kinds of data can be transferred at high speed by bypassing the CPU, enhancing system performance.

■ Hardware Configuration

The DMA controller (DMAC) consists mainly of the following blocks:

- Five independent DMA channels
- 5-channel independent access control circuit
- 32-bit address registers (reload specifiable, two registers for each channel)
- 16-bit transfer count register (reload specifiable, one register for each channel)
- 4-bit block count register (one register for each channel)
- External transfer request input pins: DREQ0, DREQ1 (for ch.0, ch.1 only)
- External transfer request acceptance output pins: DACK0, DACK1 (for ch.0, ch.1 only)
- DMA end output pins: DEOP0, DEOP1 (for ch.0, ch.1 only)
- Fly-by transfer (memory → I/O and I/O → memory) (for ch.0, ch.1 only)
- 2-cycle transfer

■ Main Functions

The followings are the main functions related to data transfer by the DMA controller (DMAC):

Data can be transferred independently over multiple channels (5 channels)

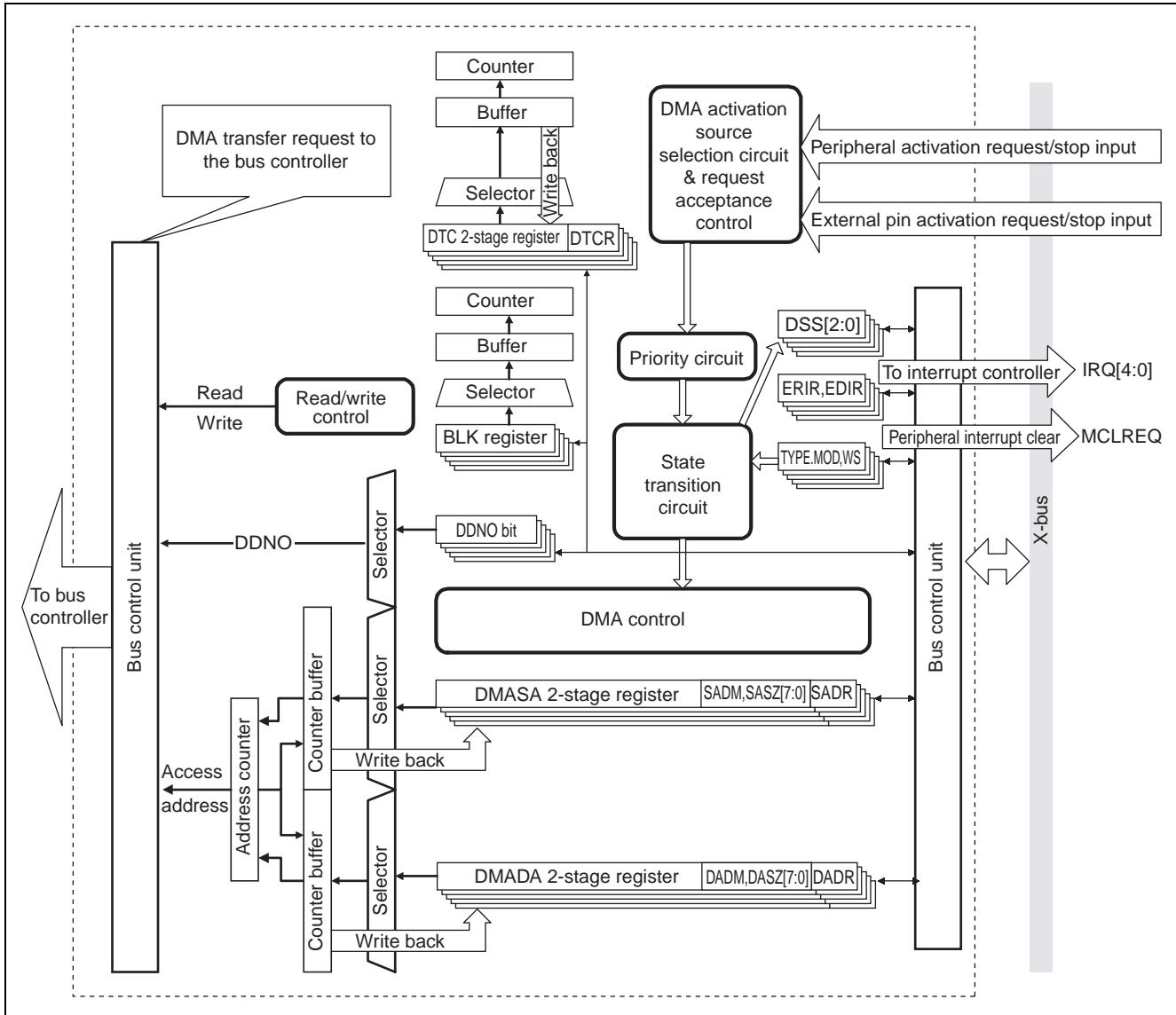
- Priority (ch.0>ch.1>ch.2>ch.3>ch.4)
- The order can be rotated between ch.0 and ch.1.
- DMAC start sources
 - External dedicated pin input (edge detection/level detection for ch.0, ch.1 only)
 - Built-in peripheral requests (shared interrupt requests, including external interrupts)
 - Software request (register write)
- Transfer mode
 - Demand transfer, burst transfer, step transfer, and block transfer
 - Addressing mode: 32-bit full addressing (increment/decrement/fixed)

The address increment/decrement range is from -255 to +255.
 - Data types: Byte, halfword, and word length
 - Single shot/reload selectable

■ Block Diagram

Figure 14.1-1 is a block diagram of the DMA controller (DMAC).

Figure 14.1-1 Block Diagram of the DMA Controller (DMAC)



14.2 DMA Controller (DMAC) Registers

This section describes the configuration and functions of the registers used by the DMA controller (DMAC).

■ DMA Controller (DMAC) Registers

Figure 14.2-1 shows the registers of the DMA controller (DMAC).

Figure 14.2-1 DMA Controller (DMAC) Registers

bit	31	24	23	16	15	8	7	0			
ch.0									Control/status register A		(DMACA0)
ch.0									Control/status register B		(DMACB0)
ch.1									Control/status register A		(DMACA1)
ch.1									Control/status register B		(DMACB1)
ch.2									Control/status register A		(DMACA2)
ch.2									Control/status register B		(DMACB2)
ch.3									Control/status register A		(DMACA3)
ch.3									Control/status register B		(DMACB3)
ch.4									Control/status register A		(DMACA4)
ch.4									Control/status register B		(DMACB4)
									DMAC all-channel control register		(DMACR)
ch.0									Transfer source address setting register		(DMASA0)
ch.0									Transfer destination address setting register		(DMADA0)
ch.1									Transfer source address setting register		(DMASA1)
ch.1									Transfer destination address setting register		(DMADA1)
ch.2									Transfer source address setting register		(DMASA2)
ch.2									Transfer destination address setting register		(DMADA2)
ch.3									Transfer source address setting register		(DMASA3)
ch.3									Transfer destination address setting register		(DMADA3)
ch.4									Transfer source address setting register		(DMASA4)
ch.4									Transfer destination address setting register		(DMADA4)

■ Notes on Setting Registers

When the DMA controller (DMAC) is set, some bits need to be set while DMA is stopped. If they are set while DMA is in progress (during transfer), correct operation cannot be guaranteed.

An asterisk following a bit when its function is described later indicates that the operation of the bit is affected if it is set during DMAC transfer. Rewrite this bit while DMAC transfer is stopped (start is disabled or temporarily stopped).

If the bit is set while DMA transfer start is disabled (when DMAE of DMACR=0, or DENB of DMACA=0), the setting takes effect when start is enabled.

If the bit is set while DMA transfer is temporarily stopped (DMAH[3:0] of DMACR not equal to "0000_B" or PAUS of DMACA=1), the setting takes effect when temporary stopping is canceled.

14.2.1 Control/Status Registers A (DMACA0 to DMACA4)

Control/status registers A (DMACA0 to DMACA4) control the operation of the DMAC channels. There is a separate register for each channel.

This section describes the configuration and functions of control/status registers A (DMACA0 to DMACA4).

■ Bit Configuration of Control/Status Registers A (DMACA0 to DMACA4)

Figure 14.2-2 shows the bit configuration of control/status registers A (DMACA0 to DMACA4).

Figure 14.2-2 Bit Configuration of Control/Status Registers A (DMACA0 to DMACA4)

Address	bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Initial value
000200 _H (ch.0)	DENB	PAUS	STRG		IS[4:0]		DDNO[3:0]		BLK[3:0]							00000000 0000XXXX _B	
000208 _H (ch.1)																	
000210 _H (ch.2)																	
000218 _H (ch.3)	bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000220 _H (ch.4)																XXXXXXXX XXXXXXXX _B	
																DTC[15:0]	

■ Detailed Bit of Control/Status Registers A (DMACA0 to DMACA4)

The following describes the functions of the bits of control/status registers A (DMACA0 to DMACA4).

[bit31] DENB (Dma ENaBle): DMA operation enable bit

This bit, which corresponds to a transfer channel, is used to enable and disable DMA transfer.

The activated channel starts DMA transfer when a transfer request is generated and accepted.

All transfer requests that are generated for a deactivated channel are disabled.

When the transfer on an activated channel reaches the specified count, this bit is set to "0" and transfer stops.

The transfer can be forced to stop by writing "0" to this bit. Be sure to stop a transfer forcibly ("0" write) only after temporarily stopping DMA using the PAUS bit (bit30 of DMACA). If the transfer is forced to stop without first temporarily stopping DMA, DMA stops but the transferred data cannot be guaranteed. Check whether DMA is stopped using the DSS[2:0] bits [bit18 to bit16 of DMACB].

Table 14.2-1 shows the function of the DMA operation enable bit.

Table 14.2-1 Function of DMA Operation Enable Bit

DENB	Function
0	Disables operation of DMA on the corresponding channel (initial value).
1	Enables operation of DMA on the corresponding channel.

- If a stop request is accepted during reset: Initialized to "0".
- This bit is readable and writable.
- If the operation of all channels is disabled by bit15 (DMAE bit) of the DMAC all-channel control register (DMACR), writing "1" to this bit is disabled and the stopped state is maintained. If the operation is disabled by the above bit while it is enabled by this bit, "0" is written to this bit and the transfer is stopped (forced stop).

[bit30] PAUS (PAUSe): Temporary stop instruction

This bit temporarily stops DMA transfer on the corresponding channel. If this bit is set, DMA transfer is not performed before this bit is cleared again (While DMA is stopped, the DSS bits are "1XX_B").

If this bit is set before starting, DMA transfer continues to be temporarily stopped.

New transfer requests that occur while this bit is set are accepted, but no transfer starts before this bit is cleared (See "Transfer Request Acceptance and Transfer" in Section "14.3.9 Operation from Starting to End/Stopping").

Table 14.2-2 shows the function of the temporary stop instruction.

Table 14.2-2 Function of Temporary Stop Instruction

PAUS	Function
0	Enables DMA operation of the corresponding channel (initial value)
1	Temporarily stops DMA on the corresponding channel.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit29] STRG (Software TRIGger): Transfer request

This bit generates a DMA transfer request for the corresponding channel. If "1" is written to this bit, a transfer request is generated when write operation to the register is completed and transfer on the corresponding channel is started.

However, if the corresponding channel is not activated, operations on this bit are disabled.

If starting by a write operation to the DMAE bit and a transfer request occurring due to this bit are simultaneous, the transfer request is enabled and transfer is started. If writing of "1" to the PAUS bit and a transfer request occurring due to this bit are simultaneous, the transfer request is enabled, but DMA transfer is not started before "0" is written to the PAUS bit.

Table 14.2-3 shows the function of the transfer request.

Table 14.2-3 Function of Transfer Request

STRG	Function
0	Disabled
1	DMA starting request

- When reset: Initialized to "0".
- The read value is always "0".
- Only a write value of "1" is valid. If "0" is written, operation is not affected.

[bit28 to bit24] IS4 to IS0 (Input Select): Transfer source selection

These bits select the source of a transfer request. Note that the software transfer request by the STRG bit function is always valid regardless of the setting of these bits. As listed in Table 14.2-4.

Table 14.2-4 Settings for Transfer Request Sources

IS	Function
00000 _B	Software starting disabled
00001 _B	Sending disabled
↓	↓
01101 _B	Sending disabled
01110 _B	External pin "H" level or ↑ edge
01111 _B	External pin "L" level or ↓ edge
10000 _B	UART0 (receiving complete)
10001 _B	UART1 (receiving complete)
10010 _B	UART2 (receiving complete)
10011 _B	UART0 (sending complete)
10100 _B	UART1 (sending complete)
10101 _B	UART2 (sending complete)
10110 _B	External interrupt 0
10111 _B	External interrupt 1
11000 _B	Reload timer 0
11001 _B	Reload timer 1
11010 _B	Reload timer 2
11011 _B	External interrupt 2
11100 _B	External interrupt 3
11101 _B	PPG0
11110 _B	PPG1
11111 _B	A/D

- When reset: Initialized to "00000_B".
- These bits are readable and writable.

Notes:

- If DMA start resulting from an interrupt from a peripheral function is set ($IS=1XXXX_B$), disable interrupts from the selected peripheral function with the ICR register.
- If demand transfer mode is selected, only $IS[4:0]=01110_B$, 01111_B can be set. Starting by other sources is disabled.
- External request input is valid only for ch.0 and ch.1. External request input cannot be selected for ch.2, ch.3 and ch.4. Whether level detection or edge detection is used is determined by the mode setting. Level detection is selected for demand transfer. For all other cases, edge detection is selected.

If transfer source of the external interrupt 0 to 3 is selected, the setting of CPU clock and peripheral clock using the base clock division setting register (DIVR0) cannot be used under the following condition.

Divide-by rate of CPU clock	Divide-by rate of peripheral clock
CLKB divide-by rate = 1	CLKP divide-by rate > 3
CLKB divide-by rate = 2	CLKP divide-by rate > 6
CLKB divide-by rate = 3	CLKP divide-by rate > 9
CLKB divide-by rate = 4	CLKP divide-by rate > 12
CLKB divide-by rate = 5	CLKP divide-by rate > 15

[bit23 to bit20] DDNO3 to DDNO0 (direct access number): Fly-by function for built-in peripherals

These bits specify the built-in peripheral of the transfer destination/source used by the corresponding channel.

Table 14.2-5 shows the settings of the direct access number.

Table 14.2-5 Settings of the Direct Access Number

DDNO	Function
0000 _B	Unused
0001 _B	Unused
0010 _B	Unused
0011 _B	Unused
0100 _B	Unused
0101 _B	Unused
0110 _B	Unused
0111 _B	Unused
1000 _B	Unused
1001 _B	Unused
1010 _B	Unused
1011 _B	Unused
1100 _B	Unused
1101 _B	Unused
1110 _B	Unused
1111 _B	Setting disabled

CHAPTER 14 DMA CONTROLLER (DMAC)

- When reset: Initialized to "0000_B".
- These bits are readable and writable.

Note:

This function is not supported by the MB91301 series. Any data written is ignored. Normally, write "0000_B".

[bit19 to bit16] BLK3 to BLK0 (BLocK size): Block size specification

These bits specify the block size for block transfer on the corresponding channel. The value specified by these bits becomes the number of words in one transfer unit (more exactly, the repetition count of the data width setting). If block transfer will not be performed, set "0001_B" (size 1). This register value is ignored during demand transfer. The size becomes 1.

Table 14.2-6 shows the function of the block size specification.

Table 14.2-6 Function of Block Size Specification

BLK	Function
XXXX _B	Block size specification of the corresponding channel

- When reset: Not initialized.
- These bits are readable and writable.
- If "0" is specified for all bits, the block size becomes 16 words.
- During reading, the block size is always read (reload value).

[bit15 to bit0] DTC (Dma Terminal Count register) : Transfer count register

The DTC register stores the transfer count. Each register has 16-bit length.

All registers have a dedicated reload register. When the register is used for a channel that is enabled to reload the transfer count register, the initial value is automatically written back to the register when the transfer is completed.

Table 14.2-7 shows the function of the transfer count register.

Table 14.2-7 Function of Transfer Count Register

DTC	Function
XXXX _H	Transfer count for the corresponding channel

When DMA transfer is started, data in this register is stored in the counter buffer of the DMA-dedicated transfer counter and is decremented by 1 (subtraction) after each transfer unit. When DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. Thus, the transfer count value during DMA operation cannot be read.

- When reset: Not initialized.
- These bits are readable and writable. Always access DTC using halfword length or word length.
- During reading, the count value is read. The reload value cannot be read.

14.2.2 Control/Status Registers B (DMACB0 to DMACB4)

Control/status registers B (DMACB0 to DMACB4) control the operation of each DMAC channel and exist independently for each channel.

This section describes the configuration of control/status registers B (DMACB0 to DMACB4) and their functions.

■ Bit Configuration of Control/Status Registers B (DMACB0 to DMACB4)

Figure 14.2-3 shows the bit configuration of control/status registers B (DMACB0 to DMACB4).

Figure 14.2-3 Bit Configuration of Control/Status Registers B (DMACB0 to DMACB4)

	bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Initial value
Address 000204H (ch.0)	TYPE[1:0]	MOD[1:0]	WS[1:0]	SADM	DADM	DTCR	SADR	DADR	ERIE	EDIE							00000000 00000000B
00020CH (ch.1)																	
000214H (ch.2)																	
00021CH (ch.3)	bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000224H (ch.4)	SASZ[7:0]								DASZ[7:0]								XXXXXXXX XXXXXXXX _B

■ Detailed Bit of Control/Status Registers B (DMACB0 to DMACB4)

The following describes the functions of the bits of control status registers B (DMACB0 to DMACB4).

[bit31, bit30] TYPE (TYPE): Transfer type setting

These bits are the transfer type setting bits and set the type of operation for the corresponding channel.

- 2-cycle transfer mode: In this mode, the transfer source address (DMASA) and transfer destination address (DMADA) are set and transfer is performed by repeating the read operation and write operation for the number of times specified by the transfer count. All areas can be specified as a transfer source or transfer destination (32-bit address).
- Fly-by transfer mode: In this mode, external <--> external transfer is performed in one cycle by setting a memory address as the transfer destination address (DMADA). Be sure to specify an external area for the memory address.

Table 14.2-8 shows the settings for the transfer types.

Table 14.2-8 Settings for the Transfer Types

TYPE	Function
00 _B	2-cycle transfer (initial value)
01 _B	Fly-by: Memory --> I/O transfer
10 _B	Fly-by: I/O --> memory transfer
11 _B	Setting disabled

- When reset: Initialized to "00_B".
- These bits are readable and writable.

[bit29, bit28] MOD (MODe) : Transfer mode setting

These bits are the transfer mode setting bits and set the operating mode of the corresponding channel.

Table 14.2-9 shows the settings for the transfer modes.

Table 14.2-9 Settings for Transfer Modes

MOD	Function
00 _B	Block/step transfer mode (initial value)
01 _B	Burst transfer mode
10 _B	Demand transfer mode
11 _B	Setting disabled

- When reset: Initialized to "00_B".
- These bits are readable and writable.

[bit27, bit26] WS (Word Size) : Transfer data width selection

These bits are the transfer data width selection bits and are used to select the transfer data width of the corresponding channel. Transfer operations are repeated in units of the data width specified in this register for as many times as the specified count.

Table 14.2-10 shows the selection of the transfer data width.

Table 14.2-10 Selection of the Transfer Data Width

WS	Function
00 _B	Byte-width transfer (initial value)
01 _B	Halfword-width transfer
10 _B	Word-width transfer
11 _B	Setting disabled

- When reset: Initialized to "00_B".
- These bits are readable and writable.

[bit25] SADM (Source-ADdr. count-Mode select) : Transfer source address count mode specification

This bit specifies the address processing of the transfer source address of the corresponding channel in each transfer operation.

An address increment is added or an address decrement is subtracted after each transfer operation according to the specified transfer source address count width (SASZ). When the transfer is completed, the next access address is written to the corresponding address register (DMASA).

As a result, the transfer source address register is not updated until DMA transfer is completed.

To make the address always the same, specify "0" or "1" for this bit and make the address count width (SASZ and DASZ) equal to "0".

Table 14.2-11 shows the function of the transfer source address count mode specification.

Table 14.2-11 Function of Transfer Source Address Count Mode Specification

SADM	Function
0	Increments the transfer source address. (initial value)
1	Decrements the transfer source address.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit24] DADM (Destination-ADdr. count-Mode select): Transfer destination address count mode specification

This bit specifies the address processing for the transfer destination address of the corresponding channel in each transfer operation.

An address increment is added or an address decrement is subtracted after each transfer operation according to the specified transfer destination address count width (DASZ). When the transfer is completed, the next access address is written to the corresponding address register (DMADA).

As a result, the transfer destination address register is not updated until the DMA transfer is completed.

To make the address always the same, specify "0" or "1" for this bit and make the address count width (SASZ, DASZ) equal to "0".

Table 14.2-12 shows the function of the transfer destination address count mode specification.

Table 14.2-12 Function of Transfer Destination Address Count Mode specification

DADM	Function
0	Increments the transfer source address. (initial value)
1	Decrements the transfer source address.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit23] DTCR (DTC-reg. Reload): Transfer count register reload specification

This bit controls reloading of the transfer count register for the corresponding channel.

If reload operation is enabled by this bit, the count register value is restored to its initial value after the transfer is completed then DMAC stops and then waiting starts for new transfer requests (an activation request by STRG or IS setting). If this bit is "1", the DENB bit is not cleared.

DENB=0 or DMAE=0 must be set to stop the transfer. In either case, the transfer is forcibly stopped.

If reloading of the counter is disabled, a single shot operation occurs. In single shot operation, operation stops after the transfer is completed even if reload is specified in the address register. The DENB bit is also cleared in this case.

CHAPTER 14 DMA CONTROLLER (DMAC)

Table 14.2-13 shows the function of the transfer count register reload specification.

Table 14.2-13 Function of Transfer Count Register Reload Specification

DTCR	Function
0	Disables transfer count register reloading (initial value)
1	Enables transfer count register reloading.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit22] SADR (Source-ADdr.-reg. Reload): Transfer source address register reload specification

This bit controls reloading of the transfer source address register for the corresponding channel.

If this bit enables the reload operation, the transfer source address register value is restored to its initial value after the transfer is completed.

If reloading of the counter is disabled, a single shot operation occurs. In single shot operation, operation stops after the transfer is completed even if reload is specified in the address register. The address register value also stops in this case while the initial value is being reloaded.

If this bit disables the reload operation, the address register value when the transfer is completed is the address to be accessed next to the final address. When address increment is specified, the next address is an incremented address.

Table 14.2-14 shows the function of the transfer source address register reload specification.

Table 14.2-14 Function of Transfer Source Address Register Reload Specification

SADR	Function
0	Disables transfer source address register reloading. (initial value)
1	Enables transfer source address register reloading.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit21] DADR (Dest.-ADdr.-reg. Reload):

Transfer destination address register reload specification

This bit controls reloading of the transfer destination address register for the corresponding channel.

If this bit enables reloading, the transfer destination address register value is restored to its initial value after the transfer is completed.

The details of other functions are the same as those described for bit22 (SADR).

Table 14.2-15 shows the function of the transfer destination address register reload specification.

Table 14.2-15 Function of Transfer Destination Address Register Reload Specification

DADR	Function
0	Disables transfer destination address register reloading. (initial value)
1	Enables transfer destination address register reloading.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit20] ERIE (ERror Interrupt Enable): Error interrupt output enable

This bit controls the occurrence of an interrupt for termination after an error occurs. The nature of the error that occurred is indicated by DSS2 to DSS0. Note that an interrupt occurs only for specific termination causes and not for all termination causes (Refer to bits DSS2 to DSS0, which are bit18 to bit16).

Table 14.2-16 shows the function of the error interrupt output enable.

Table 14.2-16 Function of Error Interrupt Output Enable

ERIE	Function
0	Disables error interrupt request output. (initial value)
1	Enables error interrupt request output.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit19] EDIE (EnD Interrupt Enable): End interrupt output enable

This bit controls the occurrence of an interrupt for normal termination.

Table 14.2-17 shows the function of the end interrupt output enable.

Table 14.2-17 Function of End Interrupt Output Enable

EDIE	Function
0	Disables end interrupt request output. (initial value)
1	Enables end interrupt request output.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit18 to bit16] DSS2 to DSS0 (Dma Stop Status): Transfer stop source indication

These bits indicate a code (end code) of 3 bits that indicates the source of stopping or termination of DMA transfer on the corresponding channel. For a list of end codes, see Table 14.2-18.

Table 14.2-18 End Codes

DSS	Function	Interrupt
000 _B	Initial value	None
X01 _B	Address error (underflow/overflow)	Error
X10 _B	Transfer stop request	Error
X11 _B	Normal end	End
1XX _B	DMA stopped temporarily (due, for example, to DMAH, PAUS bit, and an interrupt)	None

A transfer stop request is set only when it is requested by a peripheral device or the external pin DSTP function is used.

The Interrupt column indicates the type of interrupt requests that can occur.

- When reset: Initialized to "000_B".
- These bits can be cleared by writing "000_B" to them.
- These bits are readable and writable. Note that the only valid written value is "000_B".

[bit15 to bit8] SASZ (Source Addr count SiZe):Transfer source address count size specification

These bits specify the increment or decrement width for the transfer source address (DMASA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement width for each transfer unit. The address increment/decrement conforms to the instruction in the transfer source address count mode (SADM).

Table 14.2-19 shows the function of the transfer source address count size specification.

Table 14.2-19 Function of Transfer Source Address Count Size Specification

SASZ	Function
XX _H	Specify the increment/decrement width of the transfer source address. 0 to 255

- When reset: Not initialized
- These bits are readable and writable.

[bit7 to bit0] DASZ (Des Addr count SiZe):Transfer destination address count size specification

These bits specify the increment or decrement width for the transfer destination address (DMADA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement width for each transfer unit. The address increment/decrement conforms to the instruction in the transfer destination address count mode (DADM).

Table 14.2-20 shows the function of the transfer destination address count size specification.

Table 14.2-20 Function of Transfer Destination Address Count Specification

DASZ	Function
XX _H	Specify the increment/decrement width of the transfer destination address. 0 to 255

- When reset: Not initialized
- These bits are readable and writable.

14.2.3 Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)

The transfer source/transfer destination address setting registers (DMASA0 to DMASA4/DMADA0 to DMADA4) control the operation of the DMAC channels. There is a separate register for each channel.

This section describes the configuration and functions of the transfer source/transfer destination address setting registers (DMASA0 to DMASA4/DMADA0 to DMADA4).

■ Bit Configuration of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)

The transfer source/transfer destination address setting registers (DMASA0 to DMASA4/DMADA0 to DMADA4) are a group of registers that store the transfer source/transfer destination addresses. Each register is 32-bit length.

Figure 14.2-4 shows the bit configuration of the transfer source/transfer destination address setting registers (DMASA0 to DMASA4/DMADA0 to DMADA4).

Figure 14.2-4 Bit Configuration of the Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)

Address	Initial value															
	bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
001000 _H (ch.0)	XXXXXXXX XXXXXXXX _B															
001008 _H (ch.1)	XXXXXXXX XXXXXXXX _B															
001010 _H (ch.2)	XXXXXXXX XXXXXXXX _B															
001018 _H (ch.3)	bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
001020 _H (ch.4)	XXXXXXXX XXXXXXXX _B															
Address	Initial value															
	bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
001004 _H (ch.0)	XXXXXXXX XXXXXXXX _B															
00100C _H (ch.1)	XXXXXXXX XXXXXXXX _B															
001014 _H (ch.2)	XXXXXXXX XXXXXXXX _B															
00101C _H (ch.3)	bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
001024 _H (ch.4)	XXXXXXXX XXXXXXXX _B															

■ Detailed Bit of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)

The following describes the functions of the bits of each transfer source/transfer destination address setting registers (DMASA0 to DMASA4/DMADA0 to DMADA4).

[bit31 to bit0] DMASA (DMA Source Addr): Transfer source address setting

These bits set the transfer source address.

[bit31 to bit0] D (DMA Destination Addr): Transfer destination address setting

These bits set the transfer destination address.

If DMA transfer is activated, data in this register is stored in the counter buffer of the DMA-dedicated address counter and then the address is counted according to the settings for the transfer operation. When the DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. Thus, the address counter value during DMA operation cannot be read.

All registers have a dedicated reload register. When the register is used for a channel that is enabled for reloading of the transfer source/transfer destination address register, the initial value is automatically written back to the register when the transfer is completed. Other address registers are not affected.

- When reset: Not initialized.
- These bits are readable and writable. For this register, be sure to access these bits as 32-bit data.
- If these bits are read during transfer, the address before the transfer is read. If they are read after transfer, the next access address is read. Because the reload value cannot be read, it is not possible to read the transfer address in real time.

Note:

Do not set any of the DMAC's registers using this register. DMA transfer is not possible for the DMAC's registers themselves.

14.2.4 DMAC All-Channel Control Register (DMACR)

The DMAC all-channel control register (DMACR) controls the operation of all five DMAC channels. Be sure to access this register using byte length.

This section describes the configuration and functions of the DMAC all-channel control register (DMACR).

■ Bit Configuration of DMAC All-Channel Control Register (DMACR)

Figure 14.2-5 shows the bit configuration of the DMAC all-channel control register (DMACR).

Figure 14.2-5 Bit Configuration of the DMAC All-Channel Control Register (DMACR)

Address	bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Initial value
		DMAE	-	-	PM01	DMAH[3:0]				-	-	-	-	-	-	-	
000240H	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0XX00000 0000000B
bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	XXXXXXXX XXXXXXXX _B	

■ Detailed Bit of DMAC All-Channel Control Register (DMACR)

The following describes the bit functions of the DMAC all-channel control register (DMACR).

[bit31] DMAE (DMA Enable): DMA operation enable

This bit controls the operation of all DMA channels.

If DMA operation is disabled with this bit, transfer operations on all channels are disabled regardless of the start/stop settings for each channel and the operating status. Any channel carrying out transfer cancels the requests and stops transfer at a block boundary. All start operations on each channel in a disabled state are disabled.

If this bit enables DMA operation, start/stop operations are enabled for each channels. Simply enabling DMA operation with this bit does not activate each channel.

DMA operation can be forced to stop by writing "0" to this bit. However, be sure to force stopping ("0" write) only after temporarily stopping DMA using the DMAH[3:0] bits [bit27 to bit24 of DMACR]. If forced stopping is carried out without first temporarily stopping DMA, DMA stops, but the transfer data cannot be guaranteed. Check whether DMA is stopped using the DSS[2:0] bits [bit18 to bit16 of DMACB].

Table 14.2-21 shows the function of the DMA operation enable.

Table 14.2-21 Function of DMA Operation Enable

DMAE	Function
0	Disables DMA transfer on all channels. (initial value)
1	Enables DMA transfer on all channels.

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit28] PM01 (Priority mode ch.0,ch.1 robine): Channel priority rotation

This bit is set to alternate priority for each transfer between ch.0 and ch.1.

Table 14.2-22 shows the function of the channel Priority rotation.

Table 14.2-22 Function of Channel Priority Rotation

PM01	Function
0	Fixes the priority. (ch.0 > ch.1)(initial value)
1	Alternates the priority. (ch.1 > ch.0)

- When reset: Initialized to "0".
- This bit is readable and writable.

[bit27 to bit24] DMAH (DMA Halt): DMA temporary stop

These bits control temporary stopping of all DMA channels. If these bits are set, DMA transfer is not performed on any channel before these bits are cleared again.

When DMA transfer is activated after these bits are set, all channels remain temporarily stopped.

Transfer requests that occur on channels for which DMA transfer is enabled (DENB=1) while these bits are set are all enabled. The transfer can be started by clearing all these bits.

Table 14.2-23 shows the function of the DMA temporary stop.

Table 14.2-23 Function of DMA Temporary Stop

DMAH	Function
0000 _B	Enables the DMA operation on all channels. (initial value)
Other than 0000 _B	Temporarily stops DMA operation on all channels.

- When reset: Initialized to "0".
- These bits are readable and writable.

[bit30, bit29, and bit23 to bit0] (Reserved): Unused bits

These bits are unused.

- A read value is undefined.

14.2.5 Other Functions

The MB91301 series has the DACK, DEOP, and DREQ pins, which can be used for external transfer. These pins can also be used as general-purpose ports.

■ Function of the DACK, DEOP, and DREQ Pins

To use the DACK, DEOP, or DREQ pins for external transfer, a switch must be made from the port function to the DMA pin function.

To make the switch, set the PFR register.

14.3 DMA Controller (DMAC) Operation

A DMA controller (DMAC) is built into all FR family devices. The FR family DMAC is a multi-functional DMAC that controls data transfer at high speed without the use of CPU instructions.

This section describes the operation of the DMAC.

■ Principal Operations

- Functions can be set for each transfer channel independently.
- Once starting has been enabled, a channel starts transfer operation only after a specified transfer request has been detected.
- After a transfer request is detected, a DMA transfer request is output to the bus controller and the bus right is acquired by the bus controller before the transfer is started.
- The transfer is carried out as a sequence conforming to the mode settings made independently for the channel being used.

■ Transfer Mode

Each DMA channel performs transfer according to the transfer mode set by the MOD[1:0] bits of its DMACB register.

○ Block/step transfer

Only a single block transfer unit is transferred in response to one transfer request. DMA then stops requesting the bus controller for transfer until the next transfer request is received.

The block transfer unit is the specified block size (BLK[3:0] of DMACA).

○ Burst transfer

Transfer in response to one transfer request is carried out continuously for the number of times in the specified transfer count is reached.

The specified transfer count is the transfer count (BLK[3:0] of DMACA X DTC[15:0] of DMACA) X block size.

○ Demand transfer

Transfer is carried out continuously until the transfer request input (detected with a level at the DREQ pin) from an external device or a specified transfer count is reached.

The specified transfer count in a demand transfer is the specified transfer count (DTC[15:0] of DMACA). The block size is always "1" and the register value is ignored.

CHAPTER 14 DMA CONTROLLER (DMAC)

■ Transfer Type

○ 2-cycle transfer (normal transfer)

The DMA controller operates using a read operation and a write operation as its unit of operation.

Data is read from an address in the transfer source register and then written to another address in the transfer destination register.

○ Fly-by transfer (memory --> I/O)

The DMA controller operates using a read operation as its unit of operation.

If DMA transfer is performed when fly-by transfer is set, DMA issues a fly-by transfer (read) request to the bus controller and the bus controller lets the external interface carry out the fly-by transfer (read).

○ Fly-by transfer (I/O --> memory)

The DMA controller operates using a write operation as its unit of operation.

Otherwise, operation is the same as fly-by transfer (memory --> I/O) operation.

Access areas used for MB91301 series fly-by transfer must be external areas.

■ Transfer Address

The following types of addressing are available and can be set independently for each channel transfer source and transfer destination.

The method for specifying the address setting register (DMASA/DMADA) for a 2-cycle transfer and the method for a fly-by transfer are different.

○ Specifying the address for a 2-cycle transfer

The value read from a register (DMASA/DMADA) in which an address has been set in advance is used as the address for access. After receiving a transfer request, DMA stores the address from the register in the temporary storage buffer and then starts transfer.

After each transfer (access) operation, the next access address is generated (increment/decrement/fixed selectable) by the address counter and then written to the temporary storage buffer. Because the contents of the temporary storage buffer are written back to the register (DMASA/DMADA) after each block transfer unit is completed, the address register (DMASA/DMADA) value is updated after each block transfer unit is completed, making it impossible to determine the address in real time during transfer.

○ Specifying the address for a fly-by transfer

In a fly-by transfer, the value read from the transfer destination address register (DMADA) is used as the address for access. The transfer source address register (DMASA) is ignored. Be sure to specify an external area as the address to be set.

After receiving a transfer request, DMA stores the address from the register in the temporary storage buffer and then starts transfer.

After each transfer (access) operation, the next access address is generated (increment/decrement/fixed selectable) by the address counter and then written to the temporary storage buffer. Because the contents of this temporary storage buffer are written back to the register (DMADA) after each block transfer unit is completed, the address register (DMADA) value is updated after each block transfer unit is completed, making it impossible to determine the address in real time during transfer.

■ Transfer Count and Transfer End

○ Transfer count

The transfer count register is decremented (-1) after each block transfer unit is completed. When the transfer count register becomes "0", counting for the specified transfer ends, and the transfer stops with the end code displayed or is reactivated *.

Like the address register, the transfer count register value is updated only after each block transfer unit.

*: If transfer count register reloading is disabled, the transfer ends. If reloading is enabled, the register value is initialized and then waits for transfer (DTCR of DMACB)

○ Transfer end

Listed below are the sources for transfer end. When transfer ends, a source is indicated as the end code (DSS[2:0] of DMACB).

- End of the specified transfer count (DMACA:BLK[3:0] x DMACA:DTC[15:0]) => Normal end
- A transfer stop request from a peripheral circuit or the external pin (DSTP) occurred => Error
- An address error occurred => Error
- A reset occurred => Reset

The transfer stop source is indicated (DSS) and the transfer end interrupt or error interrupt for the end source is generated.

14.3.1 Setting a Transfer Request

The following three types of transfer requests are provided to activate DMA transfer:

- External transfer request pin
- Built-in peripheral request
- Software request

Software requests can always be used regardless of the settings of other requests.

■ External Transfer Request Pin

A transfer request is generated by input to the input pin prepared for a channel.

The MB91301 series supports ch.1, ch.2 (DREQ0, DREQ1).

If the input is valid at this point, the following sources are selected depending on the settings for the transfer type and the start source:

○ Edge detection

If the transfer type is block, step, or burst transfer, select edge detection:

- Falling edge detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01110_B.
- Rising edge detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01111_B.

○ Level detection

If the transfer type is demand transfer, select level detection:

- "H" level detection: Set with the transfer source selection register. When IS[4:0] of DMACA= 01110_B.
- "L" level detection: Set with the transfer source selection register. When IS[4:0] of DMACA= 01111_B.

■ Built-in Peripheral Request

A transfer request is generated by an interrupt from the built-in peripheral circuit.

For each channel, set the peripheral's interrupt by which a transfer request is generated (When IS[4:0] of DMACA=1XXXX_B.)

The built-in peripheral request cannot be used together with an external transfer request.

Note:

Because an interrupt request used in a transfer request seems like an interrupt request to the CPU, disable interrupts from the interrupt controller (ICR register).

■ Software Request

A transfer request is generated by writing to the trigger bit of a register (STRG of DMACA).

The software request is independent of the external transfer request pin and built-in peripheral request and can always be caused.

If a software request occurs together with a start (transfer enable) request, the transfer is started by immediate output of a DMA transfer request to the bus controller.

14.3.2 Transfer Sequence

The transfer type and the transfer mode that determine, for example, the operation sequence after DMA transfer has started can be set independently for each channel (Settings for TYPE[1:0] and MOD[1:0] of DMACB).

■ Selection of the Transfer Sequence

The following sequence can be selected with a register setting:

- Burst 2-cycle transfer
- Demand 2-cycle transfer
- Block/step 2-cycle transfer
- Burst fly-by transfer
- Demand fly-by transfer
- Block/step fly-by transfer

○ Burst 2-cycle transfer

In a burst 2-cycle transfer, as many transfers as specified by the transfer count are performed continuously for one transfer source. For a 2-cycle transfer, all 32-bit areas can be specified using a transfer source/transfer destination address.

A peripheral transfer request, software transfer request, or external pin (DREQ) edge input detection request can be selected as the transfer source.

Table 14.3-1 shows the specifiable transfer addresses (burst 2-cycle transfer).

Table 14.3-1 Specifiable Transfer Addresses (Burst 2-cycle Transfer)

Transfer source addressing	Direction	Transfer destination addressing
All 32-bit areas specifiable	→	All 32-bit areas specifiable

The following are some features of a burst transfer:

When one transfer request is received, transfer is performed continuously until the transfer count register reaches 0.

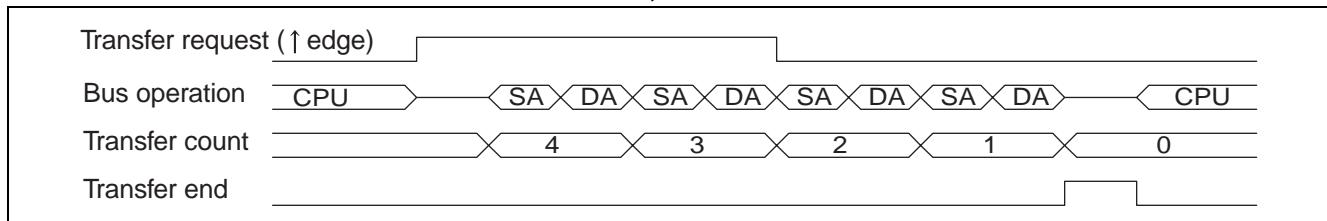
The transfer count is the transfer count x block size (BLK[3:0] of DMACA x DTC[15:0] of DMACA).

Another request occurring during transfer is ignored.

If the reload function of the transfer count register is enabled, the next request is accepted after transfer ends.

If a transfer request for another channel with a higher priority is received during transfer, the channel is switched at the boundary of the block transfer unit. Processing resumes only after the transfer request for the other channel is cleared.

Figure 14.3-1 Example of Burst Transfer for a Start on an External Pin Rising Edge, Number of Blocks =1, and Transfer Count = 4



○ Burst fly-by transfer

A burst fly-by transfer has the same features as a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory \rightarrow I/O) or write (I/O \rightarrow memory) only.

Table 14.3-2 shows the specifiable transfer addresses (burst fly-by transfer).

Table 14.3-2 Specifiable Transfer Addresses (Burst Fly-by Transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	None	External area

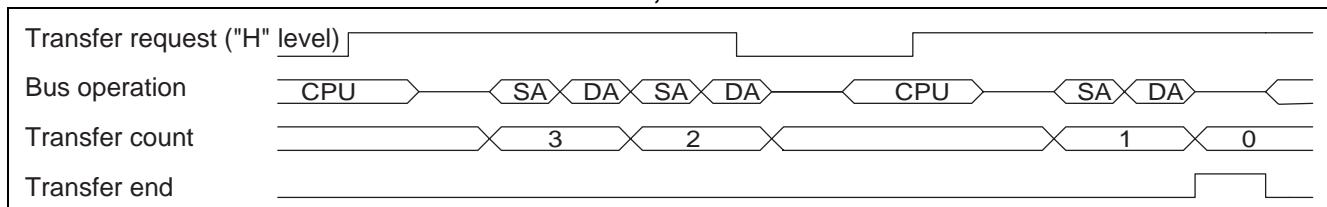
■ Demand Transfer 2-Cycle Transfer

A demand transfer sequence is generated only if "H" level or "L" level of an external pin is selected as a transfer request. Select the level with IS[4:0] of DMACA.

The following are some features of a continuous transfer:

- Each transfer operation of a transfer request is checked. While the external input level is within the range of the specified transfer request levels, transfer is performed continuously without the request being cleared. If the external input changes, the request is cleared and the transfer stops at the transfer boundary. This operation is repeated for the number of times specified by the transfer count.
- Otherwise, operations are the same as those of a burst transfer.

Figure 14.3-2 Example of Demand Transfer for a Start with the External Pin at "H" Level, Number of Blocks = 1, and Transfer Count = 3



CHAPTER 14 DMA CONTROLLER (DMAC)

Table 14.3-3 shows the specifiable transfer addresses (demand transfer 2-cycle transfer).

Table 14.3-3 Specifiable Transfer Addresses (Demand Transfer 2-cycle Transfer)

Transfer source address	Direction	Transfer destination addressing
External area	→	External area
External area	→	Built-in I/O
External area	→	Built-in RAM
Built-in I/O	→	External area
Built-in RAM	→	External area

Note:

For a demand transfer, be sure to set an external area address for the transfer source or transfer destination or both. Since DMA transfer is adjusted to the external bus timing in demand transfer mode, access to external areas is always needed.

○ Demand transfer fly-by transfer

A demand transfer fly-by transfer has the same features as a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory --> I/O) or write (I/O --> memory) only.

Table 14.3-4 shows the specifiable transfer addresses (demand transfer fly-by transfer).

Table 14.3-4 Specifiable Transfer Addresses (Demand Transfer Fly-by Transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	None	External area

○ Step/block transfer 2-cycle transfer

For a step/block transfer (Transfer for each transfer request is performed as many times as the specified block count), all 32-bit areas can be specified as the transfer source/transfer destination address.

Table 14.3-5 shows the specifiable transfer addresses (step/block transfer 2-cycle transfer).

Table 14.3-5 Specifiable Transfer Addresses (Step/Block Transfer 2-cycle Transfer)

Transfer source addressing	Direction	Transfer destination addressing
All 32-bit areas specifiable	→	All 32-bit areas specifiable

[Step transfer]

If "1" is set as the block size, a step transfer sequence is generated.

The followings are some features of a step transfer:

- If a transfer request is received, the transfer request is cleared after one transfer operation and then the transfer is stopped (The DMA transfer request to the bus controller is canceled).
- Another request occurring during transfer is ignored.

- If a transfer request for another channel with a higher priority is received during transfer, the channel is switched after the transfer is stopped and then restarted. Priority in a step transfer is valid only if transfer requests occur simultaneously.

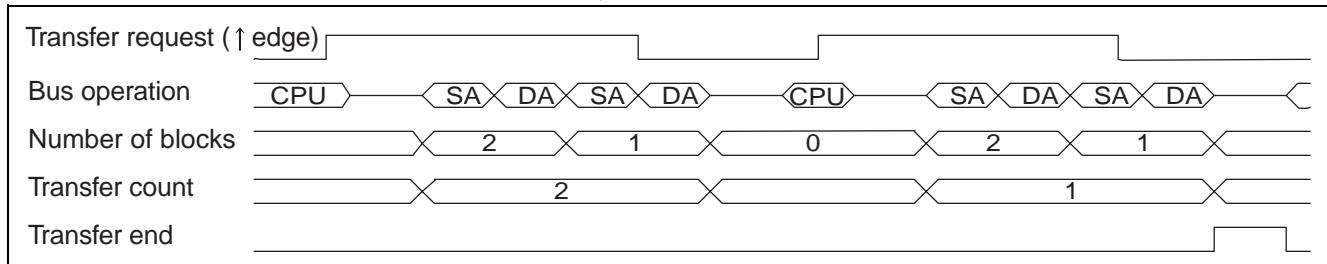
[Block transfer]

If any value other than "1" is specified as the block size, a block transfer sequence is generated.

The following are some features of a block transfer:

The block transfer has the same features as those of a step transfer except that one transfer unit consists of multiple transfer cycle counts (number of blocks).

Figure 14.3-3 Example of Block Transfer for a Start for an External Pin on a Rising Edge, Number of Blocks = 2, and Transfer Count = 2



○ Step/block transfer fly-by transfer

This transfer has the same features as those of a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory --> I/O) or write (I/O --> memory) only.

Table 14.3-6 shows the specifiable transfer addresses (step/block transfer fly-by transfer).

Table 14.3-6 Specifiable Transfer Addresses (Step/Block Transfer Fly-by Transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	None	External area

14.3.3 General Aspects of DMA Transfer

This section describes the block size for DMA transfers and the reload operation.

■ Block Size

- The unit and increment for transfer data is a set of (the number set in the block size specification register x data width) data.
- Since the amount of data transferred in one transfer cycle is determined by the value specified as the data width, one transfer unit consists of the number of transfer cycles for the specified block size.
- If a transfer request with a higher priority is received during transfer or if a temporary stop request for a transfer occurs, the transfer stops only at the transfer unit boundary, whether or not the transfer is a block transfer. This arrangement makes it possible to protect data for which division or temporary stopping is not desirable. However, if the block size is large, response time decreases.
- Transfer stops immediately only when a reset occurs, in which case the data being transferred cannot be guaranteed.

■ Reload Operation

In this module, the following three types of reloading can be set for each channel:

○ Transfer count register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer count register again and waiting for a start request starts.

Set this type of reloading when the entire transfer sequence is to be performed repeatedly.

If reload is not specified, the count register value remains "0" after the transfer is performed the specified number of times and no further transfer is performed.

○ Transfer source address register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer source address register again.

Set this type of reloading when transfer is to be repeated from a fixed area in the transfer source address area.

If reload is not specified, the transfer source address register value after the transfer is performed the specified number of times becomes the next address. Use this type when the address area is not fixed.

○ Transfer destination address register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer destination address register again.

Set this type of reloading when transfer is to be repeated to a fixed area in the transfer destination address area.

(The processing hereafter is the same as described in "Transfer source address register reloading" above.)

- If only reloading of the transfer source/transfer destination register is enabled, restart after transfer is performed the specified number of times is not implemented and only the values of each address register are set.

○ **Special examples of operating mode and the reload operation**

- If transfer is performed in continuous transfer mode by external pin input level detection and transfer count register reloading is used, transfer continues by reloading even though transfer ends during continuous input. Also in this case, an end code is set.
- If it is preferable that processing stops when data transfer ends and starts after input is detected again, do not specify reload.
- For a transfer in burst, block, or step transfer mode, transfer stops temporarily after reload when data transfer ends. Transfer does not start until new transfer request input is detected.

14.3.4 Addressing Mode

Specify the transfer destination/transfer source address independently for each transfer channel.

■ Address Register Specifications

The following two methods are provided to specify an address register. The method specified depends on the transfer sequence.

- In 2-cycle transfer mode, set the transfer source address in the transfer source address setting register (DMASA) and the transfer destination address in the transfer destination address setting register (DMADA).
- In fly-by transfer mode, specify the memory address in the transfer destination address setting register (DMADA). In this case, the value in the transfer source address setting register (DMASA) is ignored.

■ Features of the Address Register

This register has the maximum 32-bit length. With 32-bit length, all space in the memory map can be accessed.

■ Function of the Address Register

- The address register is read in each access operation and the read value is sent to the address bus.
- At the same time, the address for the next access is calculated by the address counter and the address register is updated using the calculated address.
- For address calculation, increment or decrement is selected independently for each channel, transfer destination, and transfer source. The address increment/decrement width is specified by the address count size register (SASZ/DASZ of DMACB).
- If reloading is not enabled, the address resulting from the address calculation of the last address remains in the address register when the transfer ends.
- If reloading is enabled, the initial value of the address is reloaded.

Notes:

- If an overflow or underflow occurs as a result of 32-bit length full address calculation, an address error is detected and transfer on the relevant channel is stopped. Refer to the description for the items related to the end code.
 - Do not set any of the DMAC's registers as the address register.
 - For demand transfer, be sure to set an address in an external area for the transfer source, transfer destination, or both.
 - Do not let the DMAC transfer data to any of the DMAC's registers.
-

14.3.5 Data Types

Select the data length (data width) transferred in one transfer operation from the followings:

- **Byte**
 - **Halfword**
 - **Word**
-

■ Data Length (Data Width)

Since the word boundary specification is also observed in DMA transfer, different low-order bits are ignored if an address with a different data length is specified for the transfer destination/transfer source address.

- Byte: The actual access address and the addressing match.
- Halfword: The actual access address has 2-byte length starting with "0" as the lowest-order bit.
- Word: The actual access address has a 4-byte length starting with "00" as the lowest-order 2 bits.

If the lowest-order bits in the transfer source address and transfer destination address are different, the addresses as set are output on the internal address bus. However, each transfer target on the bus is accessed after the addresses are corrected according to the above rules.

14.3.6 Transfer Count Control

Specify the transfer count within the range of the maximum 16-bit length (1 to 65536).

■ Transfer Count Control

Set the transfer count value in the transfer count register (DTC of DMACA).

The register value is stored in the temporary storage buffer when the transfer starts and is decremented by the transfer counter. When the counter value becomes "0", end of transfer for the specified count is detected, and the transfer on the channel is stopped or waiting for a restart request starts (when reload is specified).

The following are some features of the group of transfer count registers:

- Each register has 16-bit length.
- All registers have a dedicated reload register.
- If transfer is activated when the register value is "0", transfer is performed 65536 times.

■ Reload Operation

- The reload operation can be used only if reloading is enabled in a register that allows reloading.
- When transfer is activated, the initial value of the count register is saved in the reload register.
- If the transfer counter counts down to "0", end of transfer is reported and the initial value is read from the reload register and written to the count register.

14.3.7 CPU Control

When a DMA transfer request is accepted, DMA issues a transfer request to the bus controller.

The bus controller passes the right to use the internal bus to DMA at a break in bus operation and DMA transfer starts.

■ DMA Transfer and Interrupts

- During DMA transfer, interrupts are generally not accepted until the transfer ends.
- If a DMA transfer request occurs during interrupt processing, the transfer request is accepted and interrupt processing is stopped until the transfer is completed.
- If, as an exception, an NMI request or an interrupt request with a higher level than the hold suppress level set by the interrupt controller occurs, DMAC temporarily cancels the transfer request via the bus controller at a transfer unit boundary (one block) to temporarily stop the transfer until the interrupt request is cleared. In the meantime, the transfer request is retained internally. After the interrupt request is cleared, DMAC reissues a transfer request to the bus controller to acquire the right to use the bus and then restarts DMA transfer.

■ Suppressing DMA

When an interrupt source with a higher priority occurs during DMA transfer, the FR family device interrupts the DMA transfer and branches to the relevant interrupt routine. This feature is valid as long as there are any interrupt requests. When all interrupt sources are cleared, the suppression feature no longer works and the DMA transfer is restarted by the interrupt processing routine. Thus, if you want to suppress restart of DMA transfer after clearing interrupt sources in the interrupt source processing routine at a level that interrupts DMA transfer, use the DMA suppress function. The DMA suppress function can be activated by writing any value other than "0" to the DMAH[3:0] bits of the DMA all-channel control register and can be stopped by writing "0" to these bits.

This function is mainly used in the interrupt processing routines. Before the interrupt sources in an interrupt processing routine are cleared, the DMA suppress register is incremented by "1". If this is done, then no DMA transfer is performed. After interrupt processing, decrement the DMAH[3:0] bits by "1" before returning. If multiple interrupts have occurred, DMA transfer continues to be suppressed since the DMAH[3:0] bits are not "0" yet. If a single interrupt has occurred, the DMAH[3:0] bits become "0". DMA requests are then enabled immediately.

Notes:

- Since the register has only four bits, this function cannot be used for multiple interrupts exceeding 15 levels.
 - Be sure to assign the priority of the DMA tasks at a level that is at least 15 levels higher than other interrupt levels.
-

14.3.8 Hold Arbitration

When a device is operating in external bus extended mode, an external hold function can be used. The relationship between external hold requests and DMA transfer requests by this module when the hold function can be used is described below.

■ DMA Transfer Request during External Hold

DMA transfer is started when an external bus area is accessed, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

■ External Hold Request during DMA Transfer

The device is externally held. When an external bus area is accessed by DMA transfer, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

■ Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request

The device is externally held and internal DMA transfer is started. When an external bus area is accessed by DMA transfer, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

14.3.9 Operation from Starting to End/Stopping

Starting of DMA transfer is controlled independently for each channel, but before transfer starts, the operation of all channels needs to be enabled. This section describes operation from starting to end/stopping.

■ Operation Start

- **Enabling operation for all channels**

Before activating each DMAC channel, operation for all channels needs to be enabled in advance with the DMA operation enable bit (DMAE of DMACR). All start settings and transfer requests that occurred before operation is enabled are invalid.

- **Starting transfer**

The transfer operation can be started by the operation enable bit of the control register for each channel. If a transfer request to an activated channel is accepted, the DMA transfer operation is started in the specified mode.

- **Starting from a temporary stop**

If a temporary stop occurs before starting with channel-by-channel or all-channel control, the temporary stopped state is maintained even though the transfer operation is started. If transfer requests occur in the meantime, they are accepted and retained. When temporary stopping is released, transfer is started.

■ Transfer Request Acceptance and Transfer

Sampling for transfer requests set for each channel starts after starting.

If edge detection is selected for the external pin start source and a transfer request is detected, the request is retained within DMAC until the clear conditions are met (when the external pin start source is selected for block, step, or burst transfer).

If level detection or peripheral interrupt start is selected for the external pin start source, DMAC continues the transfer until all transfer requests are cleared. When they are cleared, DMAC stops the transfer after one transfer unit (demand transfer or peripheral interrupt start).

Since peripheral interrupts are handled as level detection, use interrupt clear by DMA to handle the interrupts.

Transfer requests are always accepted while other channel requests are being accepted and transfer performed. The channel that will be used for transfer is determined for each transfer unit after priority has been checked.

■ Clearing Peripheral Interrupts by DMA

This DMA has a function that clears peripheral interrupts. This function works when peripheral interrupt is selected as the DMA start source (when IS[4:0]=1XXXX_B).

Peripheral interrupts are cleared only for the set start sources. That is, only the peripheral functions set by IS[4:0] are cleared.

The timing for clearing an interrupt depends on the transfer mode (See Section "14.4 Operation Flowcharts").

- Block/step transfer: If block transfer is selected, a clear signal is generated after one block (step) transfer.
- Burst transfer: If burst transfer is selected, a clear signal is generated after transfer is performed the specified number of times.
- Demand transfer: Since only start requests from external pins are supported in demand transfer, no clear signal is generated.

■ Temporary Stopping

DMA transfer is stopped temporary in the following cases:

- **Setting of temporary stopping by writing to the control register (Set independently for each channel or all channels simultaneously)**

If temporary stopping is set using the temporary stop bit, transfer on the corresponding channel is stopped until release of temporary stopping is set again. You can check the DSS bits for temporary stopping.

When temporary stopping is released, the transfer is restarted.

- **NMI/hold suppress level interrupt processing**

If an NMI request or an interrupt request with a higher level than the hold suppress level occurs, all channels on which transfer is in progress are temporarily stopped at the boundary of the transfer unit and the bus right is opened to give priority to NMI/interrupt processing. Transfer request accepted during NMI/interrupt processing are retained, initiating a wait for completion of NMI processing.

Channels for which requests are retained restart transfer after NMI/interrupt processing is completed.

■ Operation End/Stopping

The end of DMA transfer is controlled independently for each channel. It is also possible to disable operation for all channels at once.

○ Transfer end

If reloading is disabled, transfer is stopped, "Normal end" is displayed as the end code, and all transfer requests are disabled after the transfer count register becomes "0" (Clear the DENB bit of DMACA).

If reloading is enabled, the initial value is reloaded, "Normal end" is displayed as the end code, and a wait for transfer requests starts after the transfer count register becomes "0" (Do not clear the DENB bit of DMACA).

○ Disabling all channels

If the operation of all channels is disabled with the DMA operation enable bit DMAE, all DMAC operations, including operations on active channels, are stopped. Then, even if the operation of all channels is enabled again, no transfer is performed unless a channel is restarted. In this case, no interrupt whatever occurs.

■ Stopping due to an Error

In addition to normal end after transfer for the number of times specified, stopping as the result of various types of errors and the forced stopping are provided.

○ Transfer stop requests from peripheral circuits

Depending on the peripheral circuit that outputs a transfer request, a transfer stop request is issued when an error is detected (Example: Error when data is received at or sent from a communication system peripheral).

The DMAC, when it receives such a transfer stop request, displays "Transfer stop request" as the end code and stops the transfer on the corresponding channel.

Table 14.3-7 shows the stopping due to an error.

Table 14.3-7 Stopping due to an Error

IS	Function	Transfer stop request
00000_B ↓ 01111_B	Disabled start by hardware ↓ External pin "L" level or ↓ edge	↑ None ↓
10000_B 10001_B 10010_B	UART0 ^{*1} UART1 ^{*1} UART2 ^{*1}	↑ Yes ↓
10011_B ↓ 11111_B	UART0 ^{*2} ↓ A/D	↑ None ↓

*1: A transfer stop request when a receive error is detected.

*2: A transmission is completed.

For details of the conditions under which a transfer stop request is generated, see the specifications for each peripheral circuit.

CHAPTER 14 DMA CONTROLLER (DMAC)

■ Occurrence of an Address Error

If inappropriate addressing, as shown below in parenthesis, occurs in an addressing mode, an address error is detected (if an overflow or underflow occurs in the address counter when a 32-bit address is specified).

If an address error is detected, "An address error occurred" is displayed as the end code and transfer on the corresponding channel is stopped.

14.3.10 DMAC Interrupt Control

Independent of peripheral interrupts that become transfer requests, interrupts can also be output for each DMAC channel.

■ DMAC Interrupt Control

The following interrupts can be output for each DMAC channel:

- Transfer end interrupt: Occurs only when operation ends normally.
- Error interrupt: Transfer stop request due to a peripheral circuit (error due to a peripheral)
- Error interrupt: Occurrence of address error (error due to software)

All of these interrupts are output according to the meaning of the end code.

An interrupt request can be cleared by writing "000_B" to DSS2 to DSS0 (end code) of DMACS. Be sure to clear the end code by writing "000_B" before restarting.

If reloading is enabled, the transfer is automatically restarted. At this point, however, the end code is not cleared and is retained until a new end code is written when the next transfer ends.

Since only one end source can be displayed in an end code, the result after considering the order of priority is displayed when multiple sources occur simultaneously. The interrupt that occurs at this point conforms to the displayed end code.

The following shows the priority for displaying end codes (in order of decreasing priority):

- Reset
- Clearing by writing "000_B"
- Peripheral stop request or external pin input (DSTP) stop request
- Normal end
- Stopping when address error detected
- Channel selection and control

■ DMA Transfer during Sleep

- The DMAC can also operate in sleep mode.
- If you anticipate operations during sleep mode, note the following:
 - Since the CPU is stopped, DMAC registers cannot be rewritten. Make settings before sleep mode is entered.
 - The sleep mode is released by an interrupt. Thus, if a peripheral interrupt is selected as the DMAC start source, interrupts must be disabled by the interrupt controller.
- If you do not want to release sleep mode with a DMAC end interrupt, disable these interrupts.

14.3.11 Channel Selection and Control

Up to five channels can be simultaneously set as transfer channels. In general, an independent function can be set for each channel.

■ Priority among Channels

Since DMA transfer is possible only on one channel at a time, priority must be set for the channels.

Two modes, fixed and rotation, are provided as the priority settings and can be selected for each channel group (described later).

○ Fixed mode

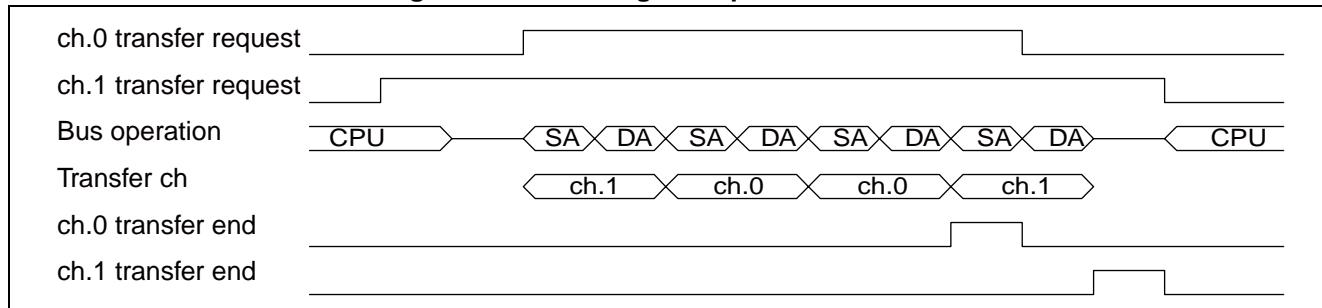
The order of priority is fixed by channel number, with priority ascending from channel 0 to channel 4:

(ch.0 > ch.1 > ch.2 > ch.3 > ch.4)

If a transfer request with a higher priority is received during a transfer, the transfer channel becomes the channel with the higher priority when the transfer for the transfer unit (number set in the block size specification register x data width) ends.

When higher priority transfer is completed, transfer is restarted on the previous channel.

Figure 14.3-4 Timing Example in Fixed Mode

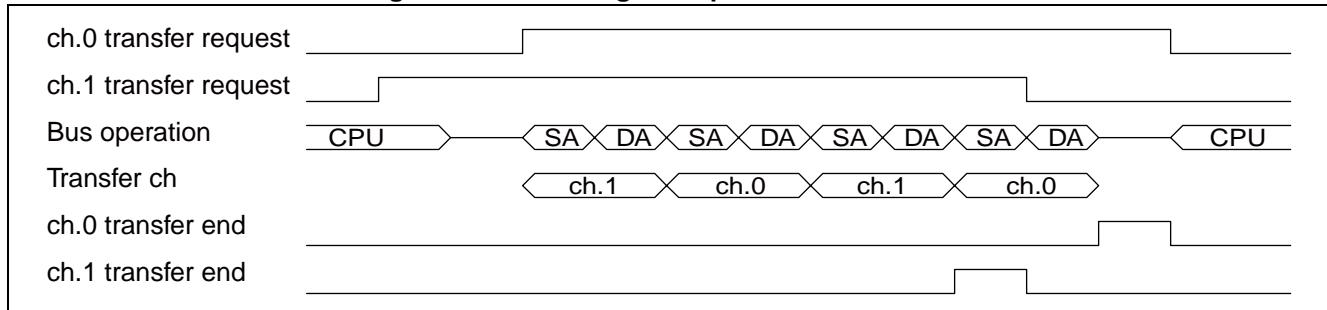


○ Rotation mode (between ch.0 and ch.1 only)

When operation is enabled, the initial states have the same order that they would have in fixed mode, but at the end of each transfer operation, the priority of the channels is reversed. Thus, if more than one transfer request is output at the same time, the channel is switched after each transfer unit.

This mode is effective when continuous or burst transfer is set.

Figure 14.3-5 Timing Example in Rotation Mode



■ Channel Group

The order of priority is set as shown in the following table.

Table 14.3-8 Unit for Selecting the Order of Priority

MODE	Priority	Remarks
Fixed	ch.0 > ch.1	—
Rotation	ch.0 > ch.1 ↑ ↓ ch.0 < ch.1	The initial state is the top row. If transfer occurs for the top row, the priority is reversed.

14.3.12 Supplement on External Pin and Internal Operation Timing

This section provides supplementary information about external pins and internal operation timing.

■ Minimum Effective Pulse Width of the DREQ Pin Input

Only channels 0 and 1 are applicable for the MB91301 series.

In all transfer modes for burst, step, block, and demand transfers, the minimum width required is five system clock cycles (5 cycles of external bus clock CLKT).

Note:

DACK output does not indicate acceptance of DREQ input. DREQ input is always accepted if DMA is enabled but transfer has not started. Therefore, it is not necessary to retain DREQ input until DACK output is asserted (except in demand transfer mode).

■ Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped

○ For 2-cycle transfer

For a demand transfer, be sure to set an address in an external area for the transfer source, the transfer destination, or both.

- If the transfer type is external → external:

Use the DREQ negation sense timing so that it is placed prior to the write strobe negation timing by a single cycle or more. There are following measures for achieving this:

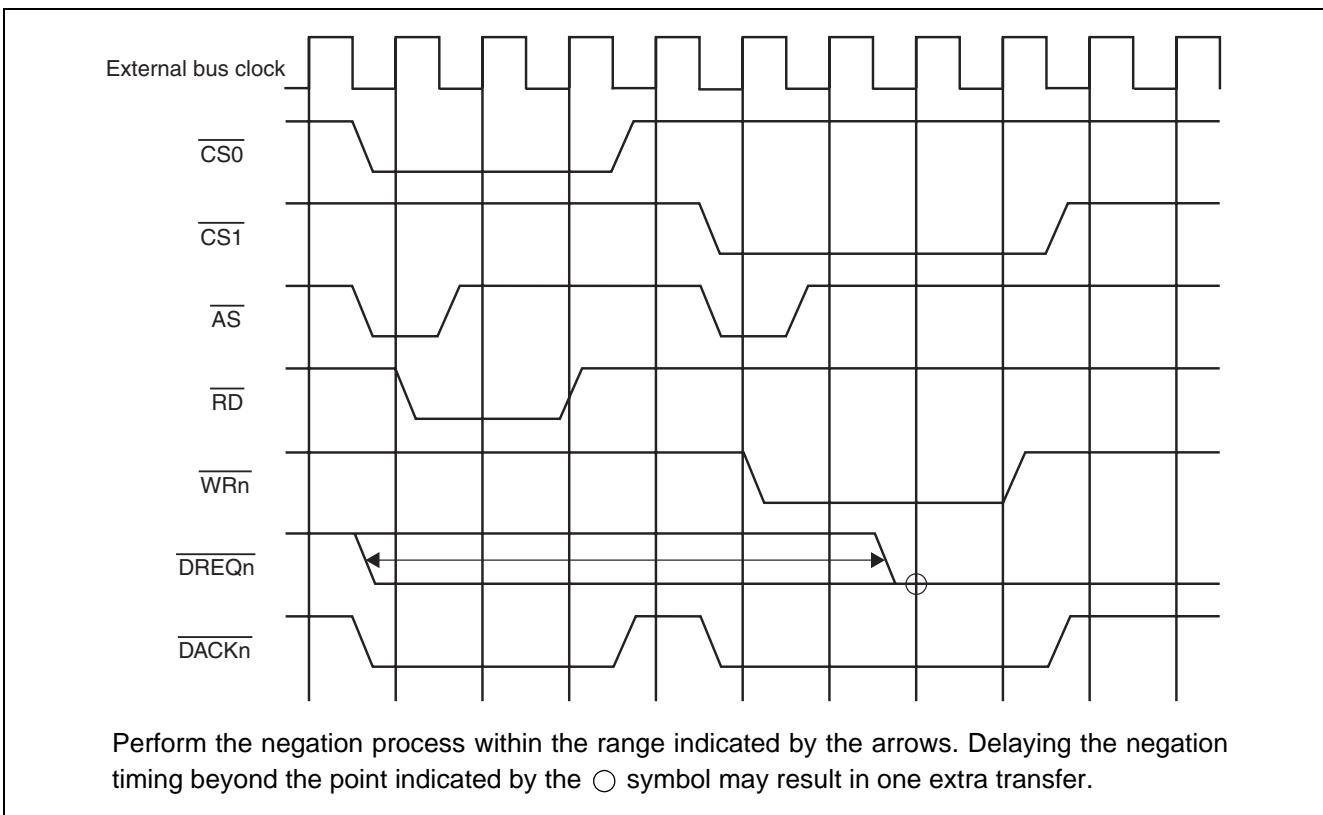
- Make the DREQ negation timing by a single cycle or more with the adjustment on the external I/O side or external glue logic side.
- Increase the wait value from the current value by a single cycle or more by using the auto wait capability in the external bus controller provided with the FR family.

If DREQ is negated after the period, the next transfer may be executed.

- If the transfer type is external → internal:

Negate before the last sense timing of the clock in the L section of the external \overline{RD} pin output when accessing the transfer source for the last DMA transfer (Section where DACK = "L" and $\overline{RD} = "L"$). If DREQ is negated later than this, a DMA request may be sensed, resulting in negation until the next transfer

Figure 14.3-6 Negate Timing Example of the DREQ Pin Input for 2-cycle External Transfer --> Internal Transfer



- If the transfer is internal \rightarrow external:

Use the DREQ negation sense timing so that it is placed prior to the write strobe negation timing by a single cycle or more. There are following measures for achieving this:

- Make the DREQ negation timing by a single cycle or more with the adjustment on the external I/O side or external glue logic side.
- Increase the wait value from the current value by a single cycle or more by using the auto wait capability in the external bus controller provided with the FR family.

If DREQ is negated after the period, the next transfer may be executed.

○ For fly-by transfer

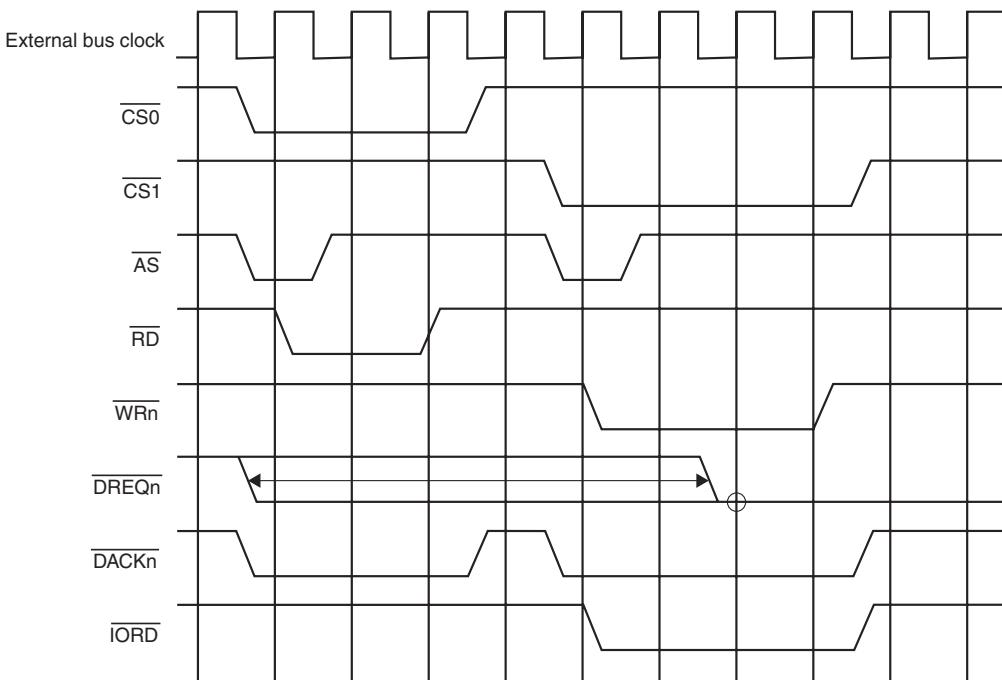
For a demand transfer, be sure to set an address in an external area for the transfer destination.

- For fly-by (timing to read pin) transfer:

After the \overline{IWR} pin output for the last DMA transfer goes to the "H" level, negate DREQ while the external \overline{RD} pin output is at the "L" level. (section where $DACK=L$ & $\overline{RD}=L$). If DREQ is negated later than this, the negation may continue until the next transfer.

- For fly-by (timing to \overline{IORD} pin) transfer:

After the external \overline{WR} pin output for the last DMA transfer goes to the "H" level, negate DREQ while $IORD$ is at the "L" level. (section where $DACK=L$ & $IORD=L$). If DREQ is negated later than this, the negation may continue until the next transfer.

Figure 14.3-7 Negate Timing Example of the DREQ Pin Input for Fly-by (Timing to \overline{IORD} Pin) Transfer

Perform the negation process within the range indicated by the arrows. Delaying the negation timing beyond the point indicated by the ○ symbol may result in one extra transfer.

■ Timing of the DREQ Pin Input for Continuing Transfer Over the Same Channel

○ For burst, step, block, and demand transfers

Operation in which transfer is continued over the same channel by the DREQ pin input cannot be guaranteed. If DREQ is reasserted at the fastest timing to clear requests retained internally after the transfer ends, at least one system clock cycle (one CLK output cycle) is provided to detect transfer requests for other channels. If, as a result, a transfer request for another channel with a higher priority is detected, transfer on that channel will be started.

Even if DREQ is reasserted earlier, it is ignored because the transfer has not been completed. If no transfer requests for other channels occur, transfer over the same channel is restarted by reasserting DREQ when the DACK pin output is asserted.

■ Timing of DACK Pin Output

The DACK output of this DMAC indicates that transfer with respect to an accepted transfer request is being performed.

The output of DACK is basically synchronized with the address output of external bus access timing. To use DACK output, it is necessary to enable the DACK output with a port.

■ Timing of the DEOP Pin Output

- The DEOP output of this DMA indicates that DMA transfer for the specified number of times of the accepted channel has been completed.
- DEOP output is output when access to an external area of the last transfer block starts. Thus, if any value other than "1" is set (block transfer mode) as the block size, DEOP is output when the last data of the last block is transferred. In this case, the acceptance of the next DREQ is already started even during transfer (before DEOP output) if the DACK pin output is asserted.
- The DEOP output is synchronized with \overline{RD} and \overline{WRn} of external bus access timing. However, if the transfer source/transfer destination is internal access, DEOP is not output. To use DEOP output, it is necessary to enable the DEOP output using the port register.

■ If an External Pin Transfer Request is Re-entered during Transfer

○ For burst, step, and block transfers

While the DACK signal is asserted within the DMAC, the next transfer request, if it is entered, is disabled. However, since operation of the external bus control unit and operation of the DMAC are not completely synchronous, the circuit must be initialized to create DREQ pin input using DACK and DEOP output to enable transfer requests by using DREQ input.

○ For a demand transfer

If reloading of the transfer count register is specified when transfer for as many transfers as specified has been completed, another transfer request is accepted.

■ If Another Transfer Request Occurs during Block Transfer

No request is detected before the transfer of the specified blocks is completed. At the block boundaries, transfer requests accepted at that time are evaluated and then transfer on the channel with the highest priority is performed.

■ Transfer between External I/O and External Memory

As targets of transfer by the DMAC, external I/O and external memory are not distinguished. Specify an external I/O as a fixed external address.

To perform fly-by transfer, set the address of external memory in the transfer destination address register. For external I/O, use DACK output and the signal decoded by the read signal \overline{RD} or write signal \overline{WRn} pin.

■ AC Characteristics of DMAC

DREQ pin input, DACK pin output, and DEOP pin output are provided as the external pins related to the DMAC. Output timing is synchronized with external bus access (refer to the AC standard for the DMAC).

14.4 Operation Flowcharts

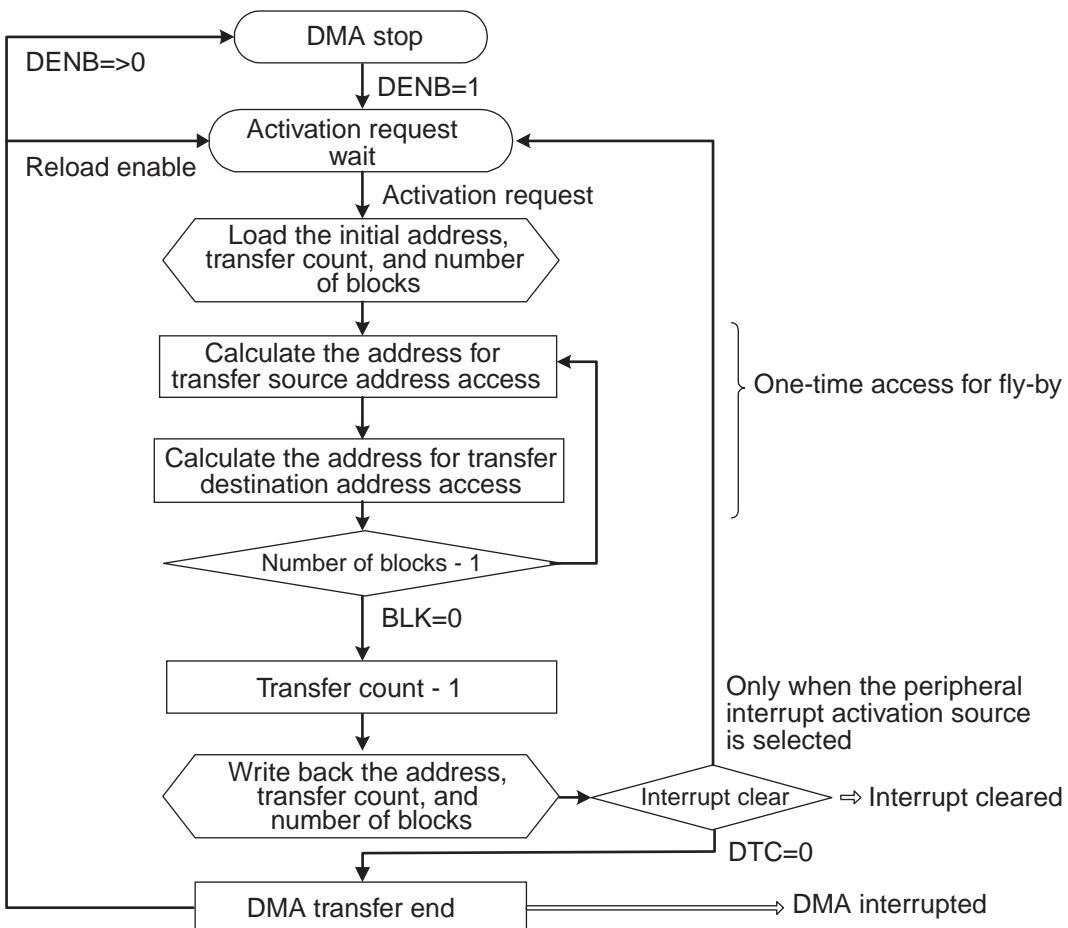
This section contains operation flowcharts for the following transfer modes:

- Block transfer
- Burst transfer
- Demand transfer

■ Block Transfer

Figure 14.4-1 shows the operation flowchart for block transfer.

Figure 14.4-1 Operation Flowchart for Block Transfer



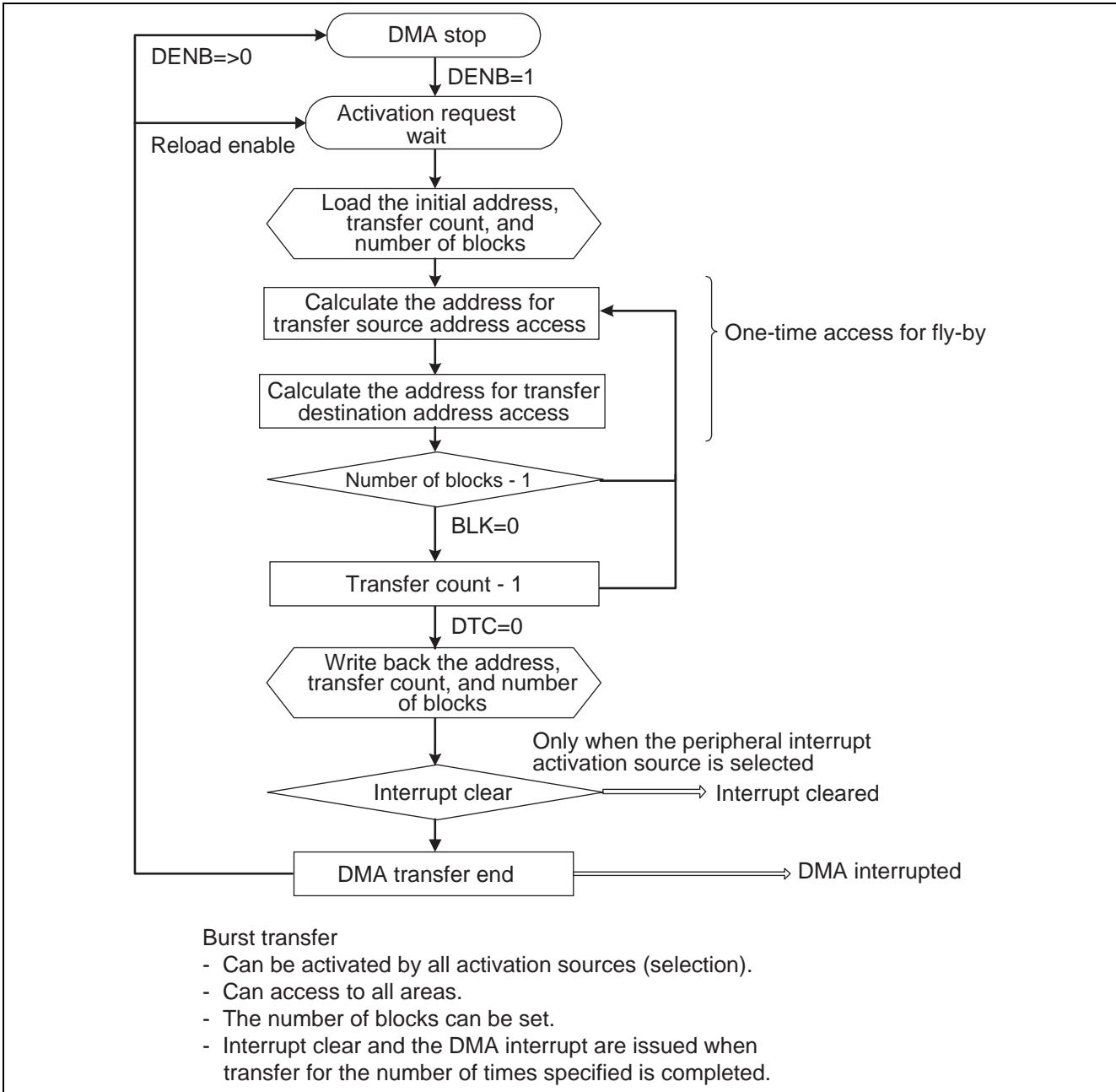
Block transfer

- Can be activated by all activation sources (selection).
- Can access to all areas.
- The number of blocks can be set.
- Interrupt clear is issued when the specified number of blocks is completed.
- The DMA interrupt is issued when transfer for the number of times specified is completed.

■ Burst Transfer

Figure 14.4-2 shows the operation flowchart for burst transfer.

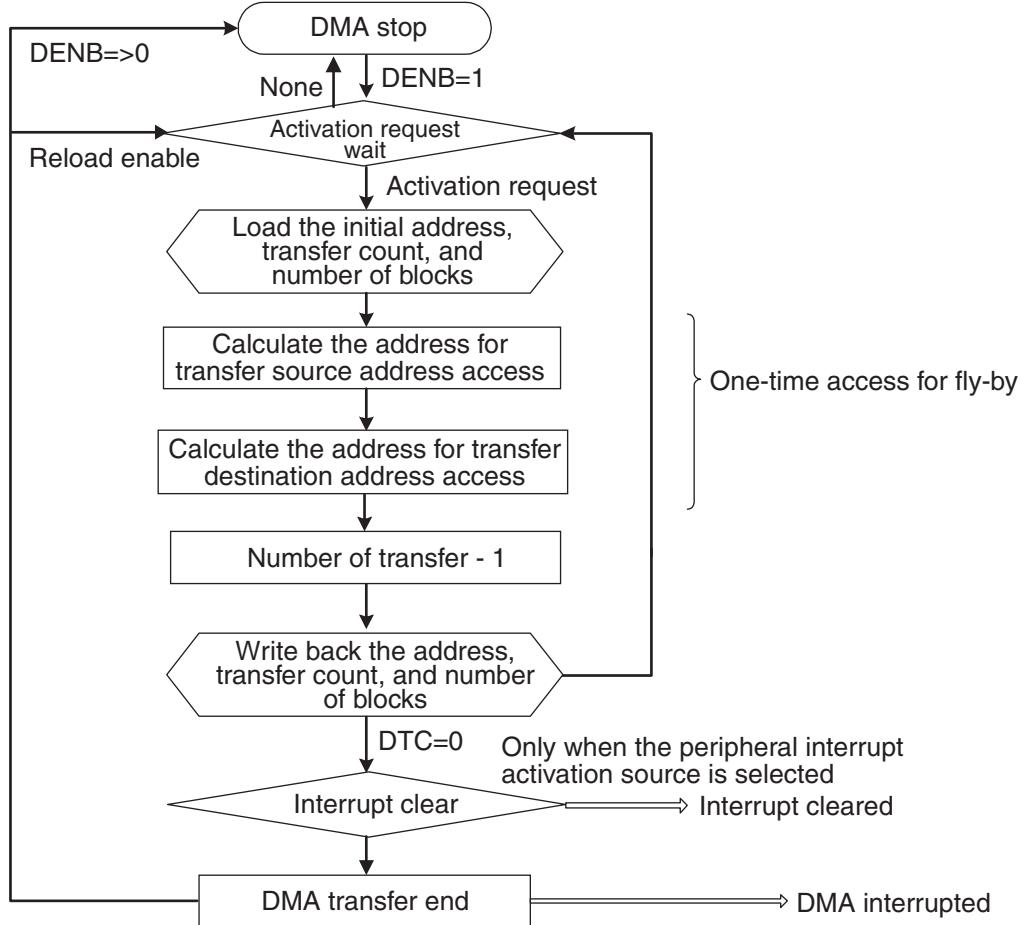
Figure 14.4-2 Operation Flowchart for Burst Transfer



■ Demand Transfer

Figure 14.4-3 shows the operation flowchart for demand transfer.

Figure 14.4-3 Operation Flowchart for Demand Transfer



Demand transfer

- Only requests (level detection) from the external pin (DREQ) are accepted. Activation by other sources is disabled.
- Access to an external area is required (since access to an external area becomes the next activation source).
- The number of blocks is always "1", regardless of the settings.
- Interrupt clear and the DMA interrupt are issued when transfer for the number of times specified is completed.

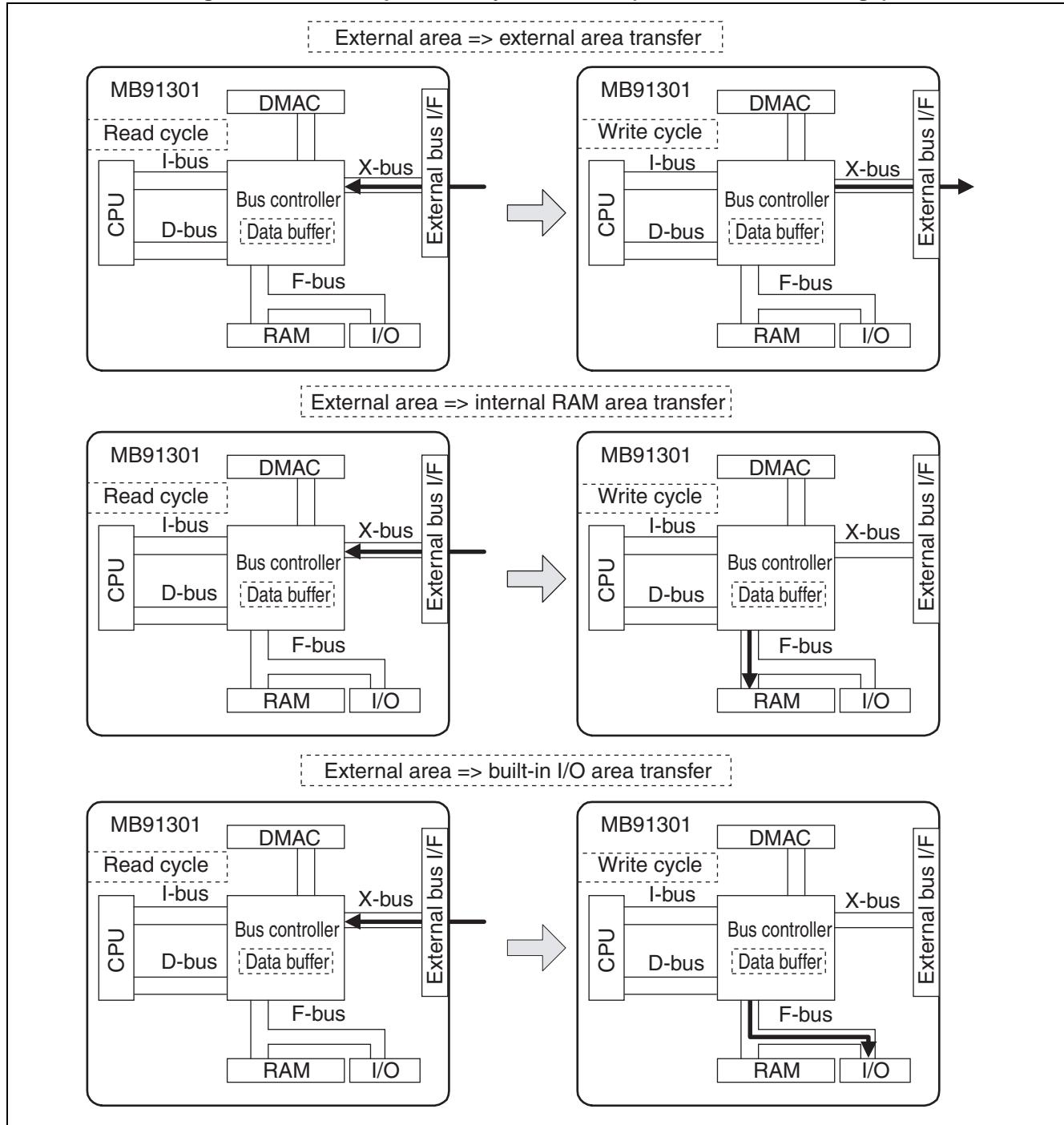
14.5 Data Bus

This section shows the flow of data during 2-cycle transfer and fly-by transfer.

■ Flow of Data during 2-Cycle Transfer

Figure 14.5-1 shows examples of six types of transfer during 2-cycle transfer.

Figure 14.5-1 Examples of 2-Cycle Transfer (Continued on Next Page)

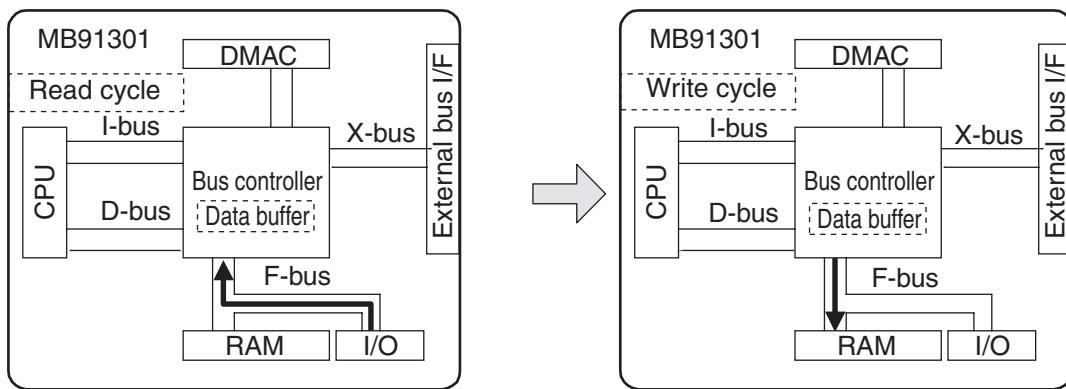


(Continued)

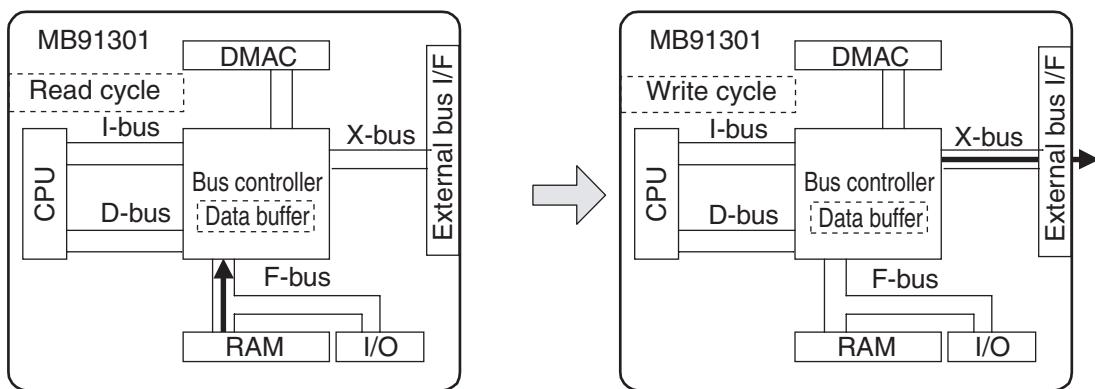
CHAPTER 14 DMA CONTROLLER (DMAC)

(Continued)

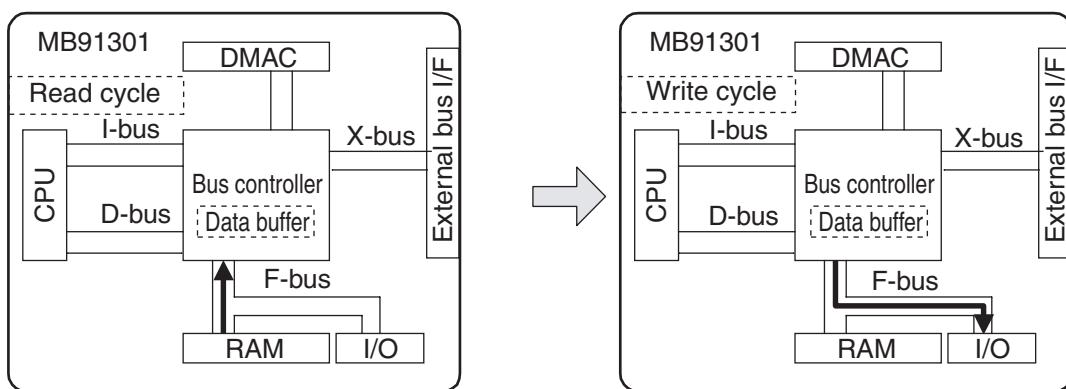
Built-in I/O area => internal RAM area transfer



Internal RAM area => external area transfer



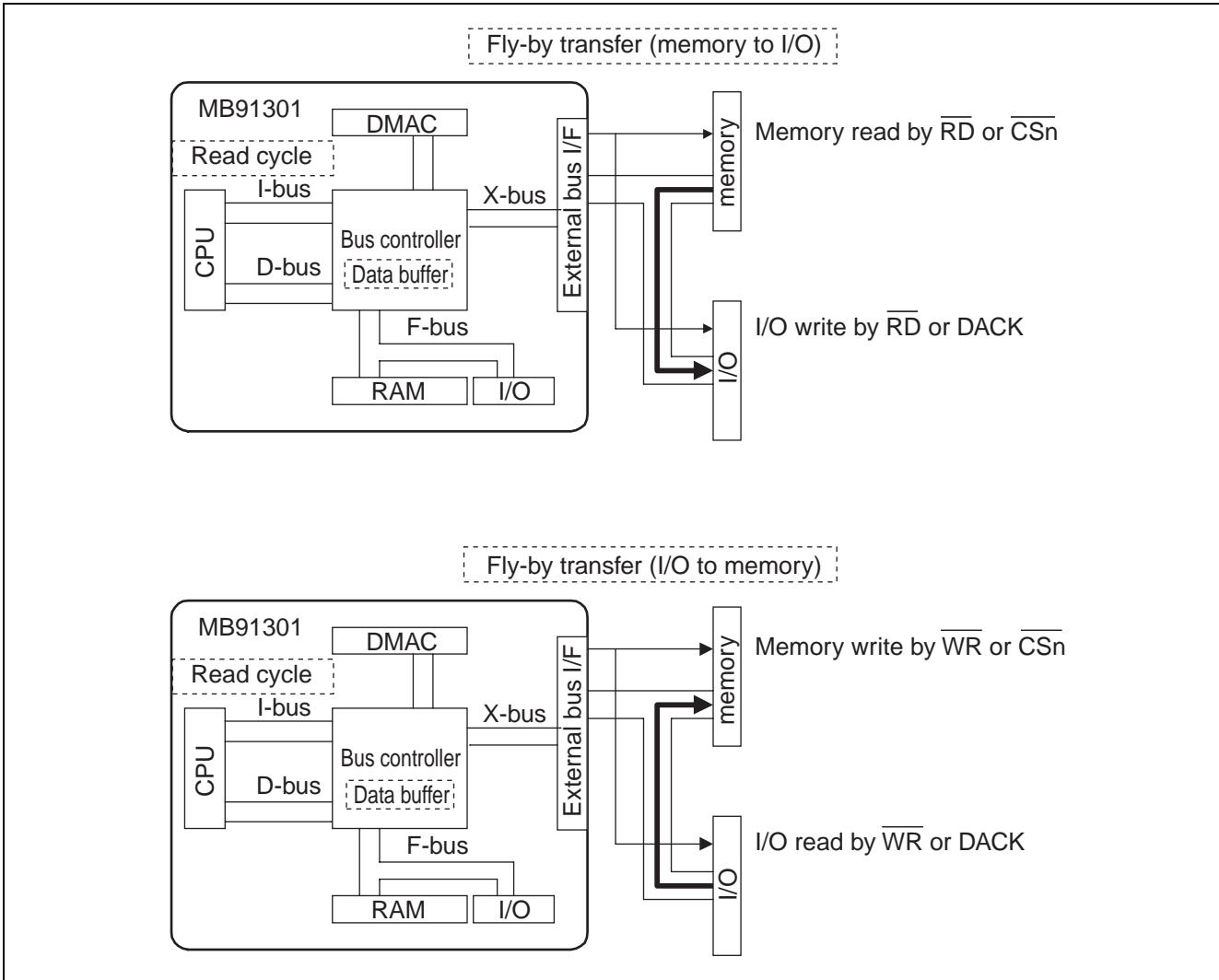
Internal RAM area => built-in I/O area transfer



■ Flow of Data during Fly-By Transfer

Figure 14.5-2 shows examples of two types of transfer during fly-by transfer.

Figure 14.5-2 Examples of Fly-By Transfer



14.6 DMA External Interface

This section provides operation timing charts for the DMA external interface.

■ DMA External Interface Pins

DMA channels 0, 1 have the following DMA-dedicated pins (DREQ, DACK, and DEOP):

- DREQ: DMA transfer request input pin for demand transfer. A transfer is requested with an input.
- DACK: This pin becomes active ("L" output) when DMA accesses an external area via the external interface.
- DEOP: This pin becomes active ("L" output) in synchronization with the last access to complete DMA transfer.
- IORD: This signal becomes active when the direction I/O -> memory is selected for fly-by transfer.
- IOWR: This signal becomes active when the direction memory -> I/O is selected for fly-by transfer.

Note:

Refer to "4.10 DMA Access Operation" for the operation example of DMA external interface.

14.6.1 Input Timing of the DREQx Pin

The DREQx pin is a DMA start request signal. If the pin is also used as a port, enable the DREQ input using the PFR register. This section shows the input timing of the DREQx pin.

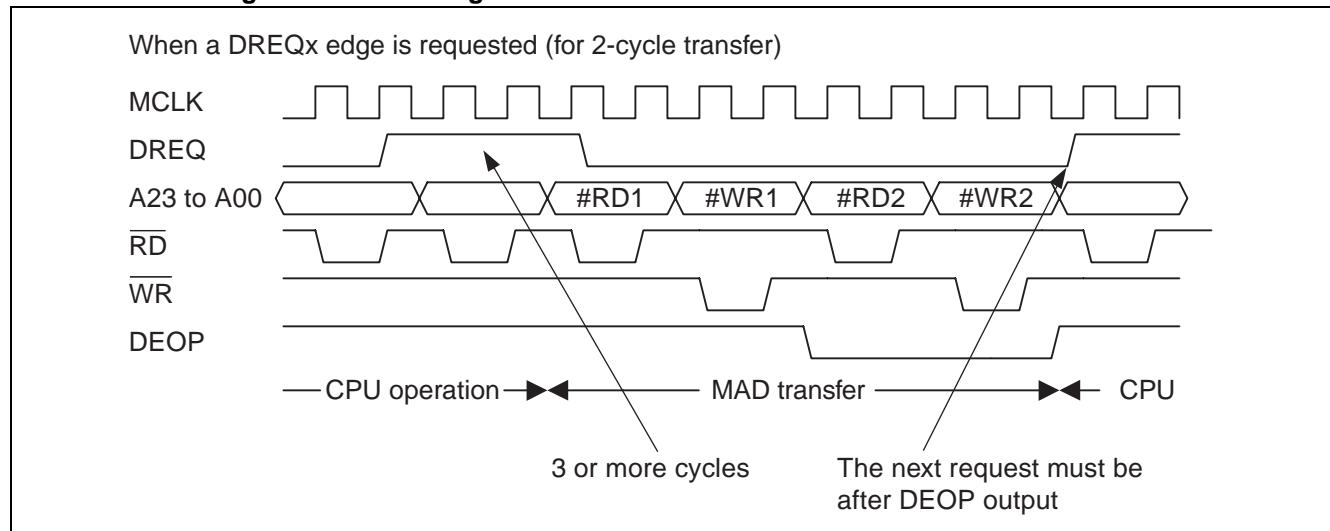
■ Timing of Transfer other than Demand Transfer

For transfer other than demand transfer, set the DMA start source to edge detection. Although there is no rule for rise/fall timing, use three or more clock cycles as the holding time for the DREQ signal. To make another transfer request, enter the request after the DMA transfer is completed (make a request after DEOP is output).

If a request is made before DEOP is output, it may be ignored.

Figure 14.6-1 shows the timing chart for transfer other than demand transfer.

Figure 14.6-1 Timing Chart for Transfer other than the Demand Transfer

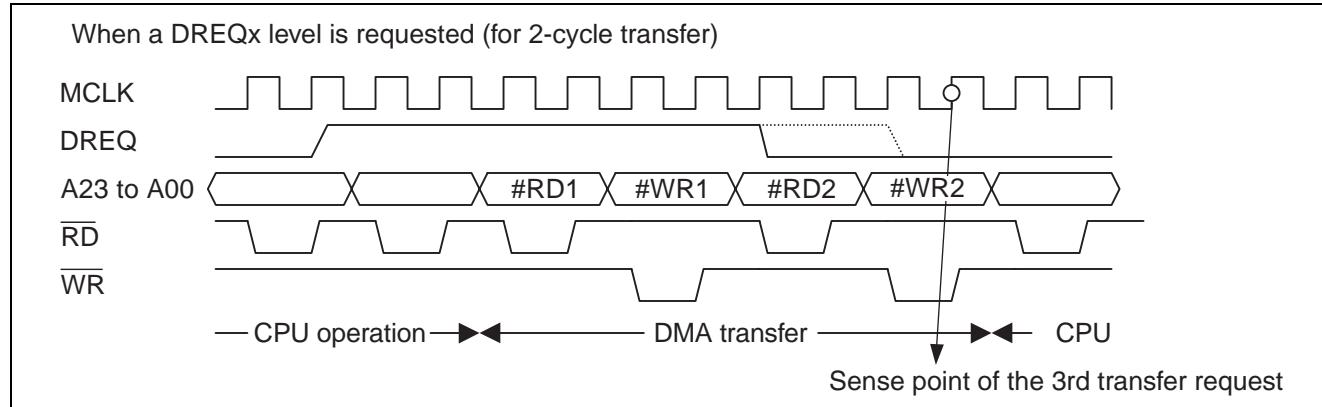


■ Timing of Demand Transfer

For demand transfer, set the DMA start source to level detection. Although there is no rule for starting, synchronize with RD/WRn of the DMA transfer when stopping a transfer. The sense timing is the rise of MCLK in the final external access.

Figure 14.6-2 shows the timing chart for demand transfer.

Figure 14.6-2 Timing Chart for Demand Transfer



Note:

In this case, because 2-cycle transfer is used and the transfer source and transfer destination are an external area, negate from the fall of #RD2 to before the final MCLK rise of #WR2 to stop the two DMA transfer operations.

14.6.2 FR30 Compatible Mode of DACK

FR30 compatible mode of DACK makes the DACK timing identical to the timing of DMA used in FR30 devices. This section provides the timing charts for the DACK pin in FR30 compatible mode for the following examples of transfer mode setting:

- 2-cycle transfer mode
- Fly-by transfer mode

■ Transfer Mode Settings

Set the transfer mode using the PFR register corresponding to the DACK pin.

When setting PFR, match the transfer mode (2-cycle transfer/fly-by transfer) of the corresponding DMA channel.

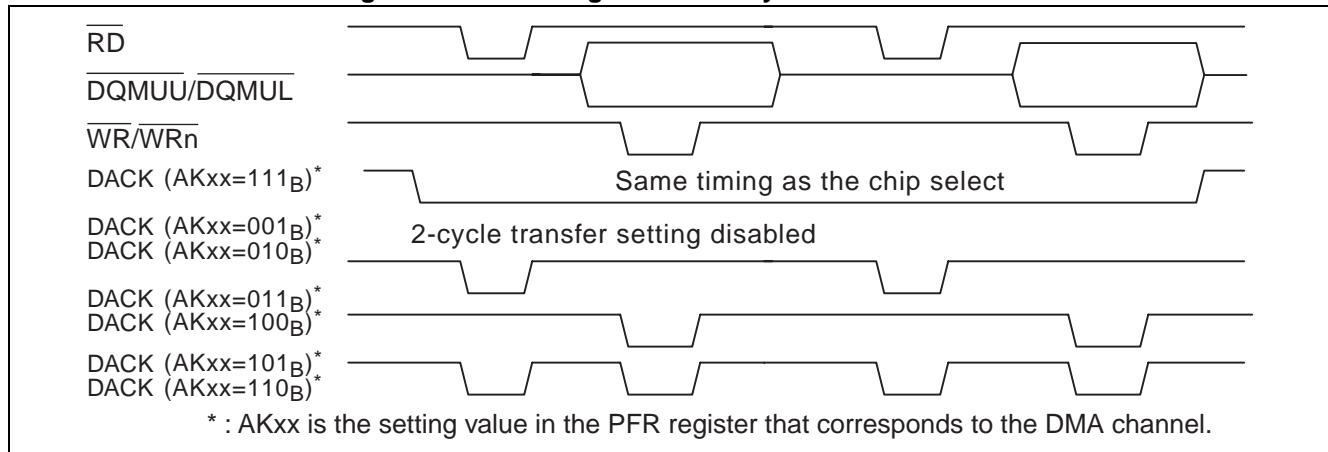
Note:

If 2-cycle transfer is set in FR30 compatible mode, the transfer is synchronized with \overline{RD} or \overline{WR} / WRn . To use WR , enable WR by setting "0X1XB" for TYPE3 to TYPE0 of the ACR register.

○ 2-cycle transfer mode

Figure 14.6-3 shows the timing chart in 2-cycle transfer mode.

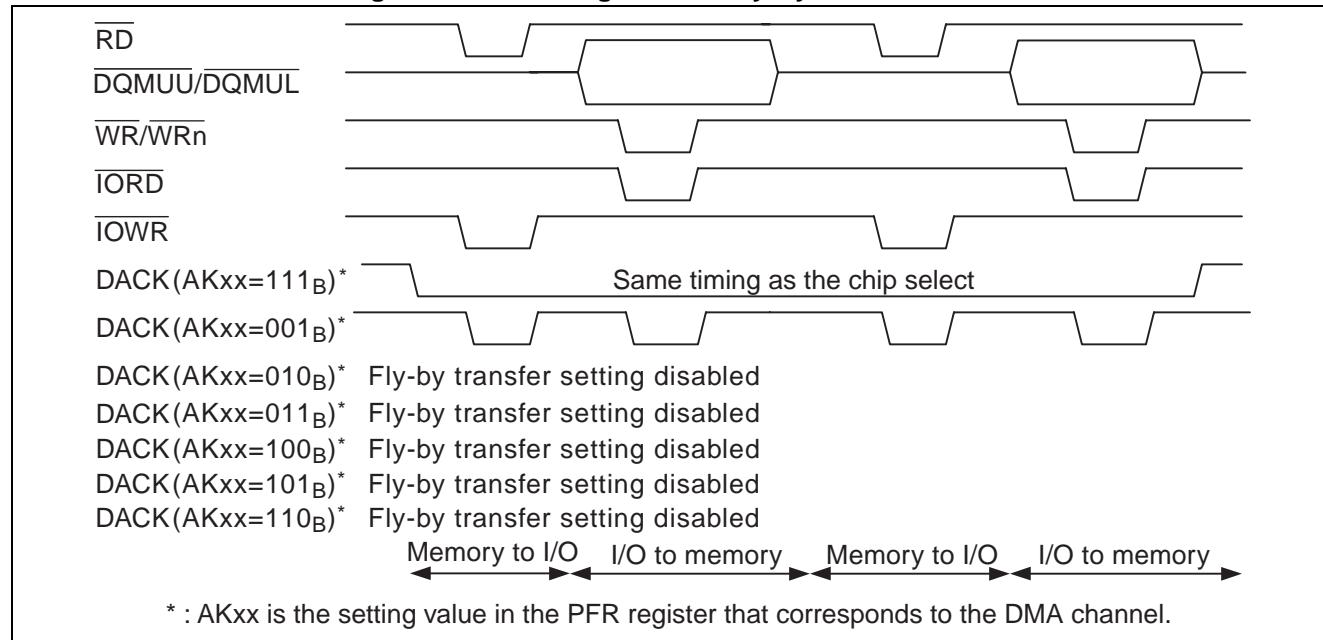
Figure 14.6-3 Timing Chart in 2-Cycle Transfer Mode



○ Fly-by transfer mode

Figure 14.6-4 shows the timing chart in fly-by transfer mode.

Figure 14.6-4 Timing Chart in Fly-By Transfer Mode



CHAPTER 15 BIT SEARCH MODULE

This chapter describes the overview of the bit search module, the configuration and functions of registers, and bit search module operation.

15.1 Overview of the Bit Search Module

15.2 Bit Search Module Registers

15.3 Bit Search Module Operation

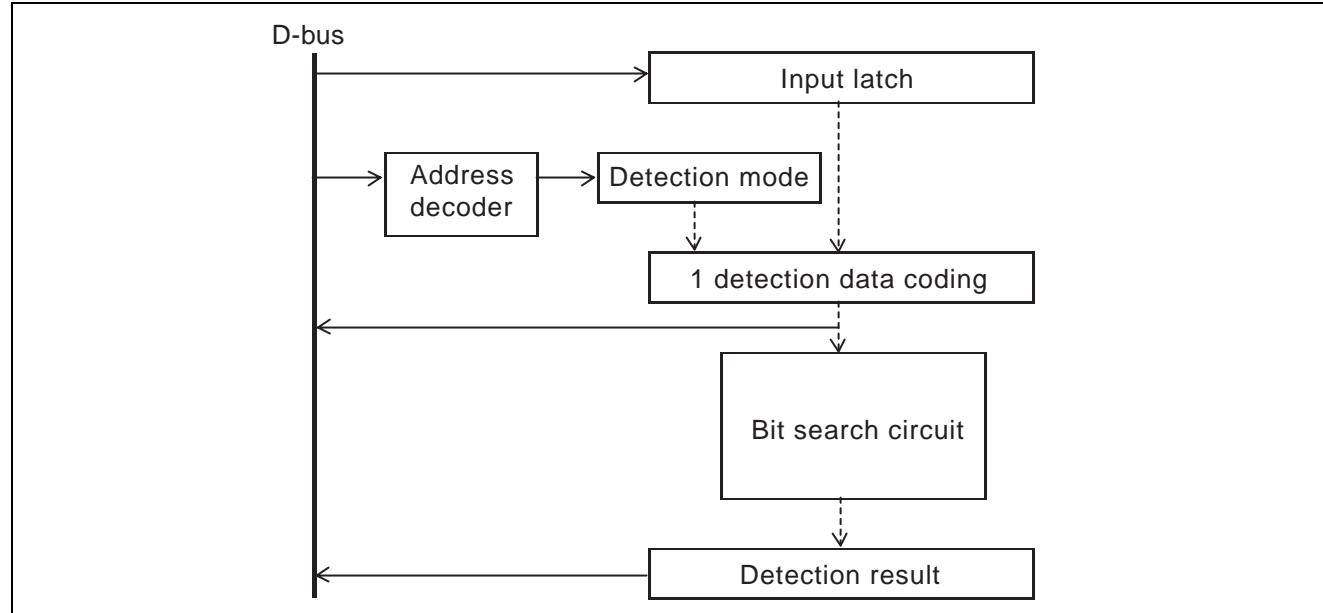
15.1 Overview of the Bit Search Module

The bit search module searches for "0", "1", or any points of change for data written to the input register and then returns the detected bit locations.

■ Block Diagram of the Bit Search Module

Figure 15.1-1 is a block diagram of the bit search module.

Figure 15.1-1 Block Diagram of the Bit Search Module

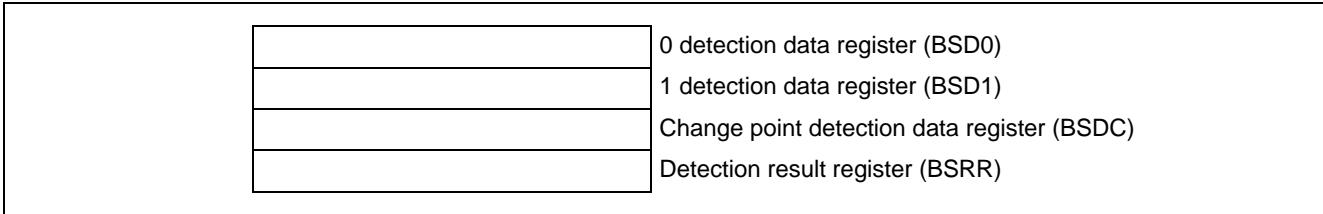


15.2 Bit Search Module Registers

This section describes the registers of the bit search module.

■ Bit Search Module Registers

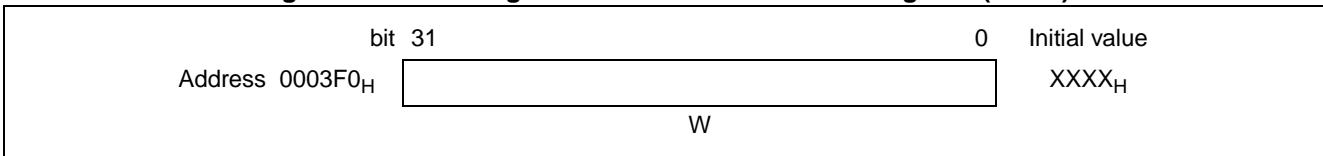
Figure 15.2-1 Bit Search Module Register



■ 0 Detection Data Register (BSD0)

The configuration of the 0 detection data register (BSD0) is shown in Figure 15.2-2.

Figure 15.2-2 Configuration of 0 Detection Data Register (BSD0)

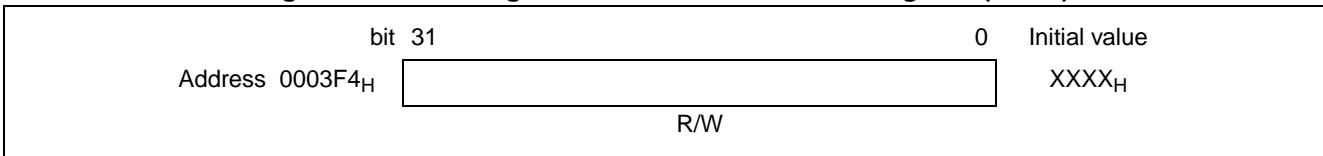


- 0 detection is performed for the written data.
- The initial value after a reset is undefined.
- The read value is undefined.
- Use a 32-bit length data transfer instruction for data transfer. Do not use 8-bit or 16-bit length data transfer instructions.

■ 1 Detection Data Register (BSD1)

The configuration of the 1 detection data register (BSD1) is shown in Figure 15.2-3.

Figure 15.2-3 Configuration of 1 Detection Data Register (BSD1)



Use a 32-bit length data transfer instruction for data transfer.

Do not use 8-bit or 16-bit length data transfer instructions.

○ Writing

1 detection is performed for the written data.

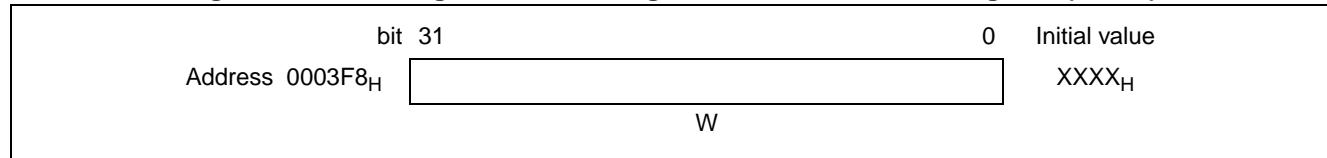
○ Reading

- Save data of the internal state of the bit search module is read. This register is used to save and restore the original state when the bit search module is used by, for example, an interrupt handler.
- Even though data is written to the 0 detection, change point detection, or data register, data can be saved and restored only by using the 1 detection data register.
- The initial value after a reset is undefined.

■ Change Point Detection Data Register (BSDC)

The configuration of the change point detection data register (BSDC) is shown in Figure 15.2-4.

Figure 15.2-4 Configuration of Change Point Detection Data Register (BSDC)



- Point of change are detected in the written value.
- The initial value after a reset is undefined.
- The read value is undefined.
- Use a 32-bit length data transfer instruction for data transfer. Do not use 8-bit or 16-bit length data transfer instructions.

■ Detection Result Register (BSRR)

The result of 0 detection, 1 detection, or change point detection is read. Which detection result is to be read is determined by the data register that has been written to last.

The configuration of the detection result register (BSRR) is shown in Figure 15.2-5.

Figure 15.2-5 Configuration of Detection Result Register (BSRR)



15.3 Bit Search Module Operation

The bit search module performs the following three operations:

- 0 detection
 - 1 detection
 - Change point detection
-

■ 0 Detection

The bit search module scans data written to the 0 detection data register from the MSB to LSB and returns the location where the first "0" is detected. The detection result can be obtained by reading the detection result register. The relationship between the detected location and the return value is given in Table 15.3-1.

If a "0" is not found (that is, the value is FFFFFFFFFF_H), 32 is returned as the search result.

[Execution example]

Write data	Read value (decimal)
$11111111111111111111000000000000_0000_B$ (FFFFF000_H)	--> 20
$11111000010010011110000010101010_B$ (F849E0AA_H)	--> 5
$10000000000000001010101010101010_B$ (8002AAAA_H)	--> 1
$11111111111111111111111111111111_B$ (FFFFFFFFFF_H)	--> 32

■ 1 Detection

The bit search module scans data written to the 1 detection data register from the MSB to LSB and returns the location where the first "1" is detected. The detection result can be obtained by reading the detection result register. The relationship between the detected location and the return value is given in Table 15.3-1.

If a "1" is not found (that is, the value is 00000000_H), 32 is returned as the search result.

[Execution example]

Write data	Read value (decimal)
$00100000000000000000000000000000_0000_B$ (20000000_H)	--> 2
$00000001001000110100010101100111_B$ (01234567_H)	--> 7
$000000000000000111111111111111_B$ (0003FFFF_H)	--> 14
$00000000000000000000000000000001_B$ (00000001_H)	--> 31
$00000000000000000000000000000000_B$ (00000000_H)	--> 32

CHAPTER 15 BIT SEARCH MODULE

■ Change Point Detection

The bit search module scans data written to the change point detection data register from bit30 to the LSB for comparison with the MSB value. The first location where a value that is different from that of the MSB is detected is returned. The detection result can be obtained by reading the detection result register.

The relationship between the detected location and the return value is given in Table 15.3-1. If a change point is not detected, 32 is returned. In change point detection, "0" is never returned as a result.

[Execution example]

Write data	Read value (decimal)
00100000000000000000000000000000 _B	(20000000 _H) --> 2
00000001001000110100010101100111 _B	(01234567 _H) --> 7
00000000000000011111111111111111 _B	(0003FFFF _H) --> 14
00000000000000000000000000000001 _B	(00000001 _H) --> 31
00000000000000000000000000000000 _B	(00000000 _H) --> 32
1111111111111111111111000000000000 _B	(FFFFF000 _H) --> 20
11111000010010011110000010101010 _B	(F849E0AA _H) --> 5
1000000000000001010101010101010 _B	(8002AAAA _H) --> 1
11111111111111111111111111111111 _B	(FFFFFFFF _H) --> 32

Table 15.3-1 Bit Locations and Return Values (Decimal)

Detected bit location	Return value						
31	0	23	8	15	16	7	24
30	1	22	9	14	17	6	25
29	2	21	10	13	18	5	26
28	3	20	11	12	19	4	27
27	4	19	12	11	20	3	28
26	5	18	13	10	21	2	29
25	6	17	14	9	22	1	30
24	7	16	15	8	23	0	31
						inexistent	32

■ Save/Restore Processing

If it is necessary to save and restore the internal state of the bit search module, such as when the bit search module is used in an interrupt handler, use the following procedure:

- 1) Read the 1 detection data register and save its contents (save).
- 2) Use the bit search module.
- 3) Write the data saved in 1) to the 1 detection data register (restore).

With the above operation, the value obtained when the detection result register is read the next time corresponds to the value written to the bit search module before "1".

If the data register written to last is the 0 detection or change point detection register, the value is restored correctly with the above procedure.

CHAPTER 16 I²C INTERFACE

This chapter describes the overview of the I²C interface, the configuration and functions of registers, and I²C interface operation.

- 16.1 Overview of the I²C Interface
- 16.2 I²C Interface Registers
- 16.3 Block Diagram of I²C Interface
- 16.4 Detailed on Registers of the I²C Interface
- 16.5 I²C Interface Operation
- 16.6 Operation Flowcharts

16.1 Overview of the I²C Interface

This section explains the overview of the I²C interface.

■ Overview

The I²C interface is a serial I/O port that supports INTER IC bus.

The I²C interface serves as a master or slave device on the I²C bus and has the following features:

- Master or slave sending and receiving
- Arbitration function
- Clock synchronization function
- Slave address and general call address detection function
- Transfer direction detection function
- Function that repeatedly generates and detects a START condition
- Bus error detection function
- 10-bit and 7-bit slave addresses
- Slave address reception acknowledge control in master mode
- Composite slave addresses supported
- Interrupt enabled for transmission or bus error
- Standard mode (maximum of 100 kbps) and high-speed mode (maximum of 400 kbps) available

16.2 I²C Interface Registers

This section describes the registers of the I²C interface.

■ I²C Interface Registers

○ Bus control register (IBCR0/IBCR1)

	bit 15	14	13	12	11	10	9	8
Address: 000094 _H /0000B4 _H	BER	BEIE	SCC	MSS	ACK	GCAA	INTE	INT
Initial value =>	R/W 0	R/W 0	W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0

○ Bus status register (IBSR0/IBSR1)

	bit 7	6	5	4	3	2	1	0
Address: 000095 _H /0000B5 _H	BB	RSC	AL	LRB	TRX	AAS	GCA	ADT
Initial value =>	R 0							

○ 10-bit slave address register (ITBA0/ITBA1)

	bit 15	14	13	12	11	10	9	8
Address: 000096 _H /0000B6 _H	-	-	-	-	-	-	TA9	TA8
Initial value =>	R 0	R 0	R 0	R 0	R 0	R 0	R/W 0	R/W 0
	bit 7	6	5	4	3	2	1	0
Address: 000097 _H /0000B7 _H	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0
Initial value =>	R/W 0							

○ 10-bit slave address mask register (ITMK0/ITMK1)

	bit 15	14	13	12	11	10	9	8
Address: 000098 _H /0000B8 _H	ENTB	RAL	-	-	-	-	TM9	TM8
Initial value =>	R/W 0	R 0	R 1	R 1	R 1	R 1	R/W 1	R/W 1
	bit 7	6	5	4	3	2	1	0
Address: 000099 _H /0000B9 _H	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0
Initial value =>	R/W 1							

CHAPTER 16 I²C INTERFACE

○ 7-bit slave address register (ISBA0/ISBA1)

bit	7	6	5	4	3	2	1	0	
Address:	00009B _H /0000BB _H	-	SA6	SA5	SA4	SA3	SA2	SA1	SA0
	R	R/W							

Initial value => 0 0 0 0 0 0 0 0 0

○ 7-bit slave address mask register (ISMK0/ISMK1)

bit	15	14	13	12	11	10	9	8	
Address:	00009A _H /0000BA _H	ENSB	SM6	SM5	SM4	SM3	SM2	SM1	SM0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Initial value => 0 1 1 1 1 1 1 1 1

○ Data register (IDAR0/IDAR1)

bit	7	6	5	4	3	2	1	0	
Address:	00009D _H /0000BD _H	D7	D6	D5	D4	D3	D2	D1	D0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Initial value => 0 0 0 0 0 0 0 0 0

○ Clock control register (ICCR0/ICCR1)

bit	15	14	13	12	11	10	9	8	
Address:	00009E _H /0000BE _H	TEST	-	EN	CS4	CS3	CS2	CS1	CS0
	W	R	R/W	R/W	R/W	R/W	R/W	R/W	

Initial value => 0 0 0 1 1 1 1 1 1

○ Clock disable register (IDBL0/IDBL1)

bit	7	6	5	4	3	2	1	0
Address:	00009F _H /0000BF _H	-	-	-	-	-	-	DBL
	R	R	R	R	R	R	R	R/W

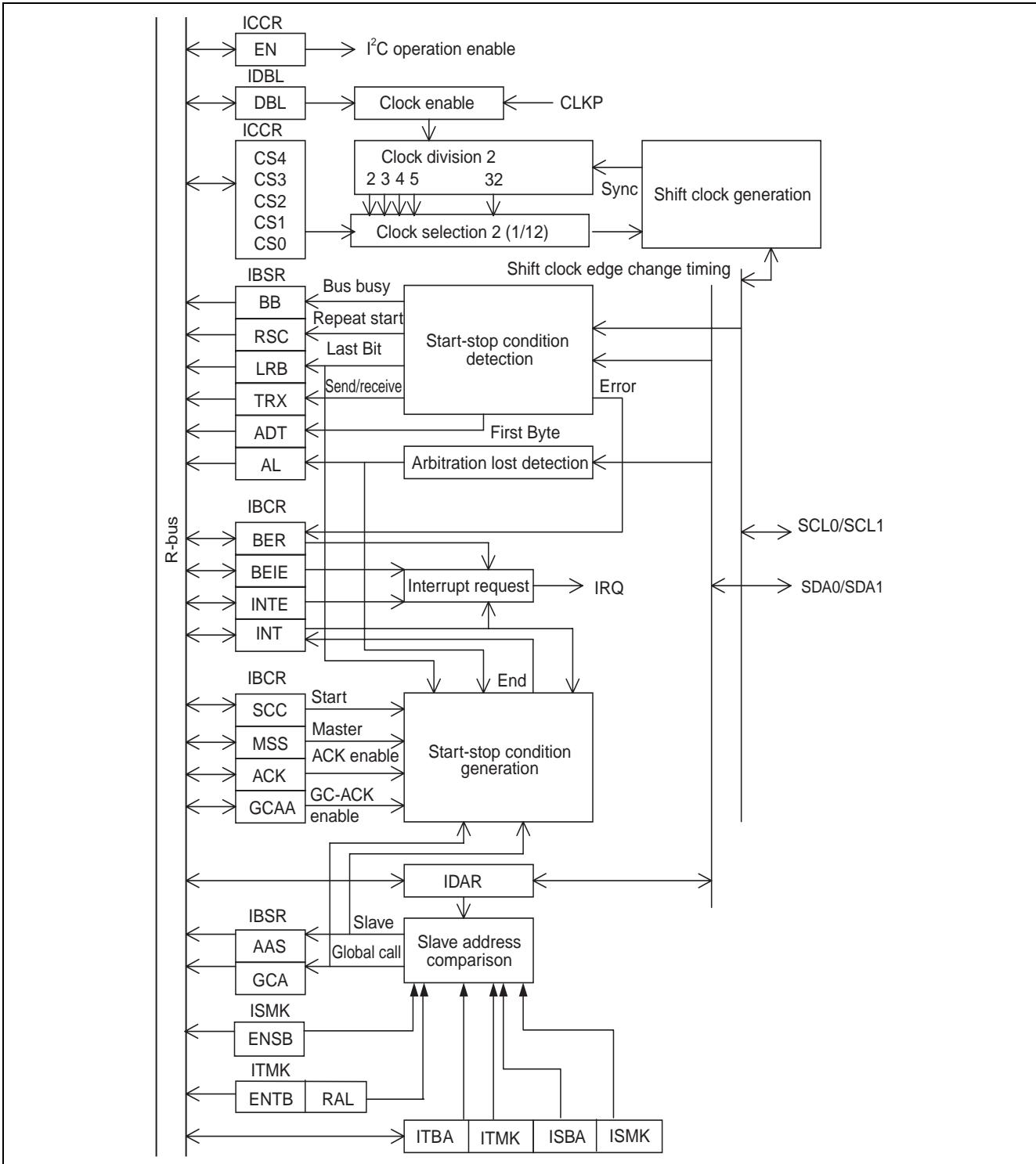
Initial value => 0 0 0 0 0 0 0 0 0

16.3 Block Diagram of I²C Interface

This section shows the block diagram of the I²C interface.

■ Block Diagram (for 1 channel)

Figure 16.3-1 Block Diagram of the I²C Interface



16.4 Detailed on Registers of the I²C Interface

This section describes the detailed register of the I²C interface.

■ Bus Status Register (IBSR0/IBSR1)

bit	7	6	5	4	3	2	1	0
Address: 000095 _H /0000B5 _H	BB	RSC	AL	LRB	TRX	AAS	GCA	ADT
Initial value=>	R	R	R	R	R	R	R	R
	0	0	0	0	0	0	0	0

This register is read only. All bits are cleared when the I²C stops operating (EN = 0 in ICCR).

[bit7] BB (Bus Busy)

This bit indicates the status of the I²C bus.

0	STOP condition detected
1	START condition detected (bus used)

[bit6] RSC (Repeated Start Condition)

This bit is the repeated START condition detection bit.

0	Repeated START condition not detected
1	Repeated START condition detected while bus is being used

This bit is cleared when the slave address transfer ends (ADT=0) or when the STOP condition is detected.

[bit5] AL (Arbitration Lost)

This bit is the arbitration lost detection bit.

0	Arbitration lost not detected
1	Arbitration lost detected during master transmission

Write "0" to the INT bit or "1" to the MSS bit of the IBCR register to clear this bit.

Arbitration lost is detected if:

- The transmission data does not match the data on the SDA line at the rising edge of SCL.
- A repeated START condition is generated in the first bit of the data by another master.
- The I²C interface cannot generate START or STOP condition because the SCL line is driven to "L" by another slave device.

[bit4] LRB (Last Received Bit)

This bit is an acknowledge storage bit that stores an acknowledge from the receiving device.

0	Slave acknowledge detected
1	Slave acknowledge not detected

This bit is rewritten if an acknowledge is detected (reception 9 bits). This bit is cleared if a START or STOP condition is detected.

[bit3] TRX (Transferring Data)

This bit indicates the transmission status during a data transfer.

0	Data transmission stopped
1	Data transmission in progress

This bit is set to "1" if:

- A START condition occurs in master mode.
- The transfer of the first byte ends when this bit is read in slave mode (transmission).
- When data is sent in master mode.

This bit is set to "0" if:

- The bus is idle (BB=0: IBCR).
- An arbitration lost occurs.
- "1" is written to the SCC bit in the master interrupt status (MSS=1, INT=1).
- The MSS bit is cleared in the master interrupt status (MSS=1, INT=1).
- No acknowledge occurred for the last transfer in slave mode.
- Data is received in slave mode.
- Data is received from a slave in master mode.

[bit2] AAS (Addressed As Slave)

This bit is the slave addressing detection bit.

0	No addressing in slave mode
1	Addressing in slave mode

This bit is cleared when a (repeated) START or STOP condition is detected.

This bit is set when a 7-bit or 10-bit slave address is detected.

[bit1] GCA (General Call Address)

This bit is the general call address (00_H) detection bit.

0	No general call address received in slave mode
1	General call address received in slave mode

This bit is cleared when a (repeated) START or STOP condition is detected.

[bit0] ADT (Address Data Transfer)

This bit is the slave address reception detection bit.

0	Received data is not a slave address (or the bus is open).
1	Received data is a slave address.

This bit is set to "1" if a START condition is detected. It is cleared after the second byte if the header section of a slave address is detected during 10-bit write access. Otherwise, it is cleared after the first byte.

"After the first or second byte" means the followings:

CHAPTER 16 I²C INTERFACE

- Writing "0" to the MSS bit during master interrupt: (MSS=1, INT=1: IBCR)
- Writing "1" to the SCC bit during master interrupt: (MSS=1, INT=1: IBCR)
- Clearing the INT bit
- Beginning of a transfer byte that is not used for the transfer target as master or slave

■ Bus Control Register (IBCR0/IBCR1)

	bit 15	14	13	12	11	10	9	8		
Address:	000094 _H	0000B4 _H	BER	BEIE	SCC	MSS	ACK	GCAA	INTE	INT
	R/W	R/W	W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value=>	0	0	0	0	0	0	0	0	0	

Bits other than BER and BEIE are cleared if the I²C interface is stopped (EN=0: ICCR).

[bit15] BER (Bus Error)

This bit is the bus error interrupt request flag bit.

For a read by a read modify instruction, "1" is always read.

During writing

0	Clears the bus error interrupt request flag.
1	Irrelevant

During reading

0	Bus error not detected
1	Error condition detected

If this bit is set, the EN bit of the CCR register is cleared, stopping the I²C interface and halting data transfer. All bits of the IBSR and IBCR registers other than BER and BEIE are cleared.

This bit must be cleared before the I²C interface is enabled (EN=1) again.

This bit is set to "1" if:

- An illegal START or STOP condition at a specific location is detected (while a slave address or data is being transferred).*
- The header section of a slave address is received during a 10-bit read access before a 10-bit write access with the first byte is performed. *
- A STOP condition is detected in master mode.

*: While the I²C interface is enabled during transfer, this detection flag is set after the first [STOP] condition is received to prevent an illegal bus error report.

[bit14] BEIE (Bus Error Interrupt Enable)

This bit is the bus error interrupt enable bit.

0	Bus error interrupt disabled
1	Bus error interrupt enabled

An interrupt occurs if this bit is set to "1" and the BER bit is set to "1".

[bit13] SCC (Start Condition Continue)

This bit is the repeated [START] condition generation bit.

During writing

0	Irrelevant
1	Generates a repeated START condition in master transfer.

The read value of this bit is always "0".

If "1" is written to this bit in master mode (MSS = 1 and INT = 1) a repeated START condition is generated, the INT bit is automatically cleared.

[bit12] MSS (Master Slave Select)

This bit is the master or slave selection bit.

0	Selects slave mode.
1	Selects master mode. Generates a START condition to enable the value of the IDAR register to be sent as a slave address.

This bit is cleared when arbitration lost occurs during master transmission, causing slave mode to start.

Write "0" to this bit during a master interrupt flag is set (MSS=1, INT=1) to automatically clear the INT bit. Then, generate a [STOP] condition to end the transfer.

Note:

The MSS bit is directly reset. To detect a STOP condition, check the BB bit of the IBSR register.

Write "1" while the bus is idle (MSS=0, BB=0) to generate a START condition. The value of the IDAR register are also sent.

While the bus is being used (BB=1, TRX=0, MSS=0), write "1" to the MSS bit to cause the I²C interface to start transmission after waiting for the bus to be opened. If, during this time, the I²C interface is specified as the address for a slave that is accompanied by a write access, the bus is opened after the transfer ends. If the interface is transmitting as a slave (AAS=1, TRX=1: IBCS) during this time, no data is sent even if the bus has been opened.

It is important to check whether the I²C interface is specified as a slave (AAS=1: IBSR), whether data transmission is normal at the next interrupt (MSS=1: IBCR), and whether an illegal termination has occurred (AL=1: IBSR).

Note:

When using in the following condition, transmission of general call address is prohibited since reception as a slave cannot be performed.

- In addition to this LSI, if another LSI that becomes master mode exists on the bus, this LSI sends general call address as a master, and arbitration lost occurs at the second byte or later.

[bit11] ACK (ACKnowledge)

This bit generates an acknowledge according to the setting of the data receive enable bit.

0	Acknowledge not generated when data is received
1	Acknowledge generated when data is received

This bit is disabled when slave address data is received in slave mode.

An acknowledge is returned if the I²C interface detects a 7-bit or 10-bit slave address when the corresponding enable bits (ENTB: ITMK, ENSB: ISMK) are set.

Write to this bit while an interrupt flag is set (INT=1), the bus is idle (BB=0: IBSR), or the I²C interface is stopped (EN=0: ICCR).

[bit10] GCAA (General Call Address Acknowledge)

This bit is an acknowledge enable bit used when a general call address is received.

0	Acknowledge not generated when general call address is received
1	Acknowledge generated when general call address is received

Write to this bit while an interrupt flag is set (INT=1), the bus is idle (BB=0: IBSR), or the I²C interface is stopped (EN=0: ICCR).

[bit9] INTE (INTerrupt Enable)

This bit is the interrupt enable bit.

0	Interrupts disabled
1	Interrupts enabled

When this bit is "1", the interrupt is generated if the INT bit is "1".

[bit8] INT (INTerrupt)

This bit is the transfer end interrupt request flag bit. For a read by a read modify instruction, "1" is always read.

During writing

0	Clears the transfer end interrupt request flag.
1	Irrelevant

During reading

0	<ul style="list-style-type: none"> • Transfer not ended • Not a transfer target • Bus is open.
1	<p>This bit is set to "1" if a one-byte transfer that includes the acknowledge bit is completed and the following conditions are met:</p> <ul style="list-style-type: none"> • Bus master • Slave address • A general call address was received. • Arbitration lost occurred. <p>If the interface is specified as a slave address, this bit is set at the end of slave address reception that includes an acknowledge.</p>

If this bit is set to "1", the SCL line is maintained at the "L" level. Write "0" to this bit to clear it and to open the SCL line to transfer the next byte. A repeated START or STOP condition is generated.

This bit is cleared when the SCC bit or the MSS bit is set to "1".

Note:

Contention of SCC, MSS, and INT bits

If data is simultaneously written to the SCC, MSS, and INT bits, contention occurs between the next-byte transfer, repeated START condition generation, and STOP condition generation. If this situation occurs, the priorities are as follows:

1. Next-byte transfer and STOP condition generation

When the INT bit is set to "0" and the MSS bit is set to "0", writing of the MSS bit has precedence and a STOP condition is generated.

2. Next-byte transfer and START condition generation

When the INT bit is set to "0" and the SCC bit is set to "1", writing of the SCC bit has precedence, a repeated START condition is generated, and the value of IDAR is sent.

3. Repeated START condition generation and STOP condition generation

When the SCC bit is set to "1" and the MSS bit is set to "0" at the same time, clearing of the MSS bit has precedence. A STOP condition is generated and the I²C interface enters slave mode.

Note:

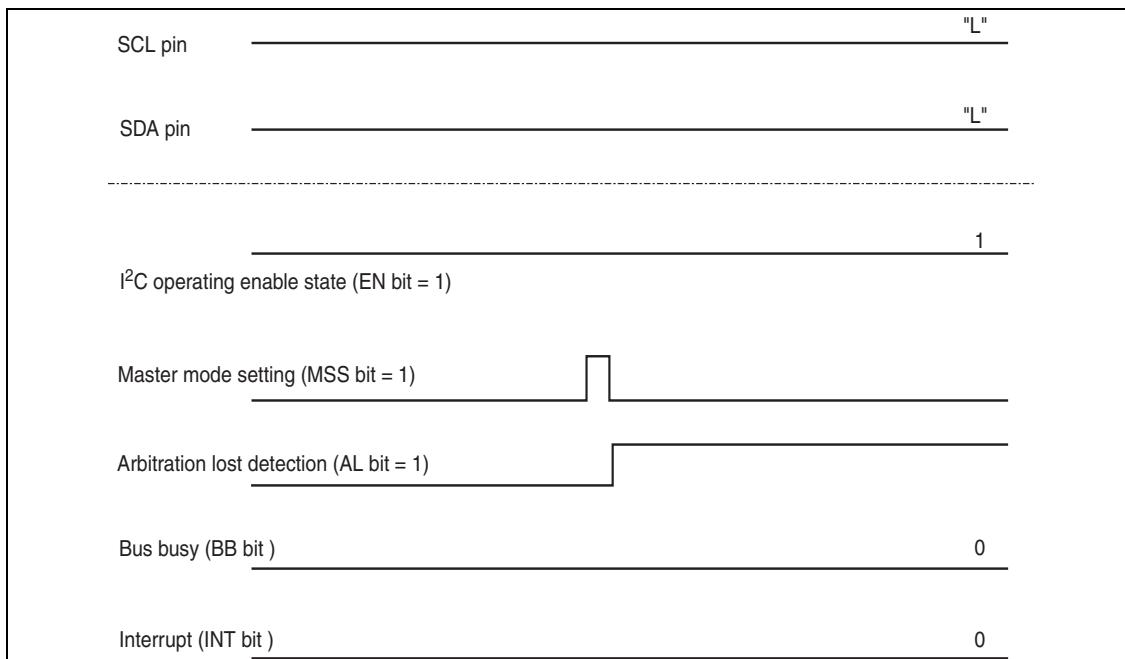
When an instruction which generates a start condition is executed (the MSS bit is set to "1") at the timing shown in Figure 16.4-2, arbitration lost detection (AL bit = 1) prevents an interrupt (INT bit = 1) from being generated.

• Condition 1 in which an interrupt (INT bit = 1) upon detection of "AL bit = 1" does not occur

When an instruction which generates a start condition is executed (setting the MSS bit in the IBCR register to "1") with no start condition detected (BB bit = 0) and with the SDA or SCL pin at the "L" level.

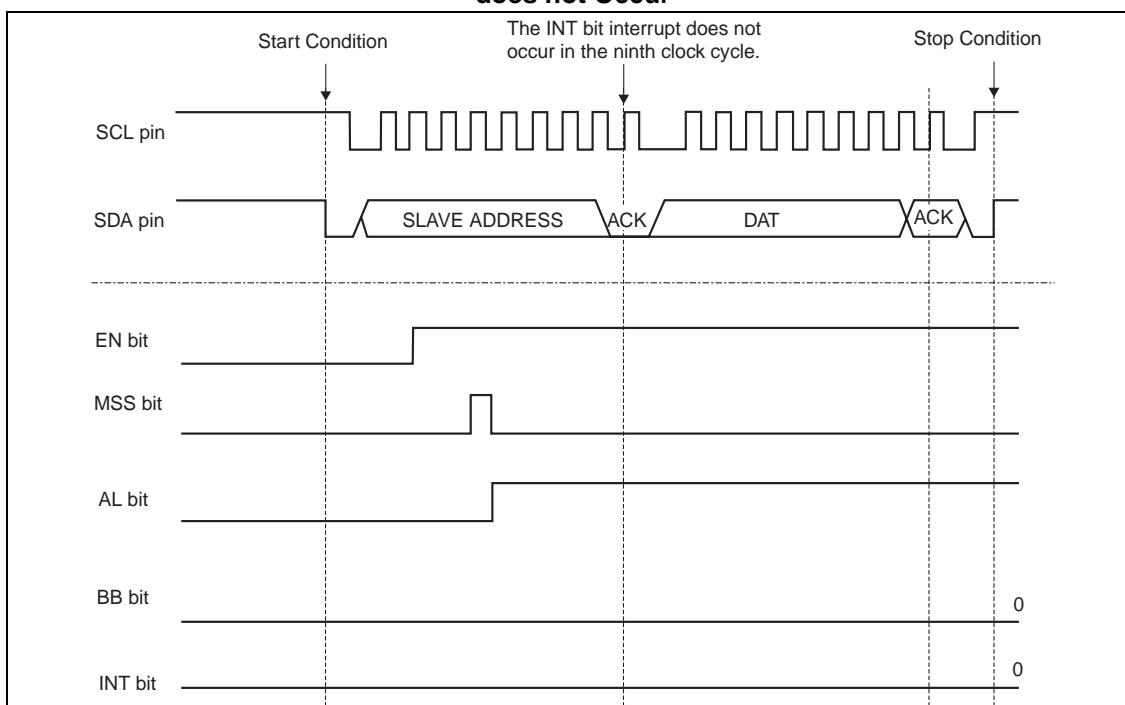
CHAPTER 16 I²C INTERFACE

Figure 16.4-1 Diagram of Timing at which an Interrupt upon Detection of "AL Bit = 1" does not Occur



- Condition 2 in which an interrupt (INT bit = 1) upon detection of "AL bit = 1" does not occur
When an instruction which generates a start condition by enabling I²C operation (EN bit = 1) is executed (setting the MMS bit in the IBCR register to "1") with the I²C bus occupied by another master.
This is because, as shown in Figure 16.4-2, when the other master on the I²C bus starts communication with I²C disabled (EN bit = 0), the I²C bus enters the occupied state with no start condition detected (BB bit = 0).

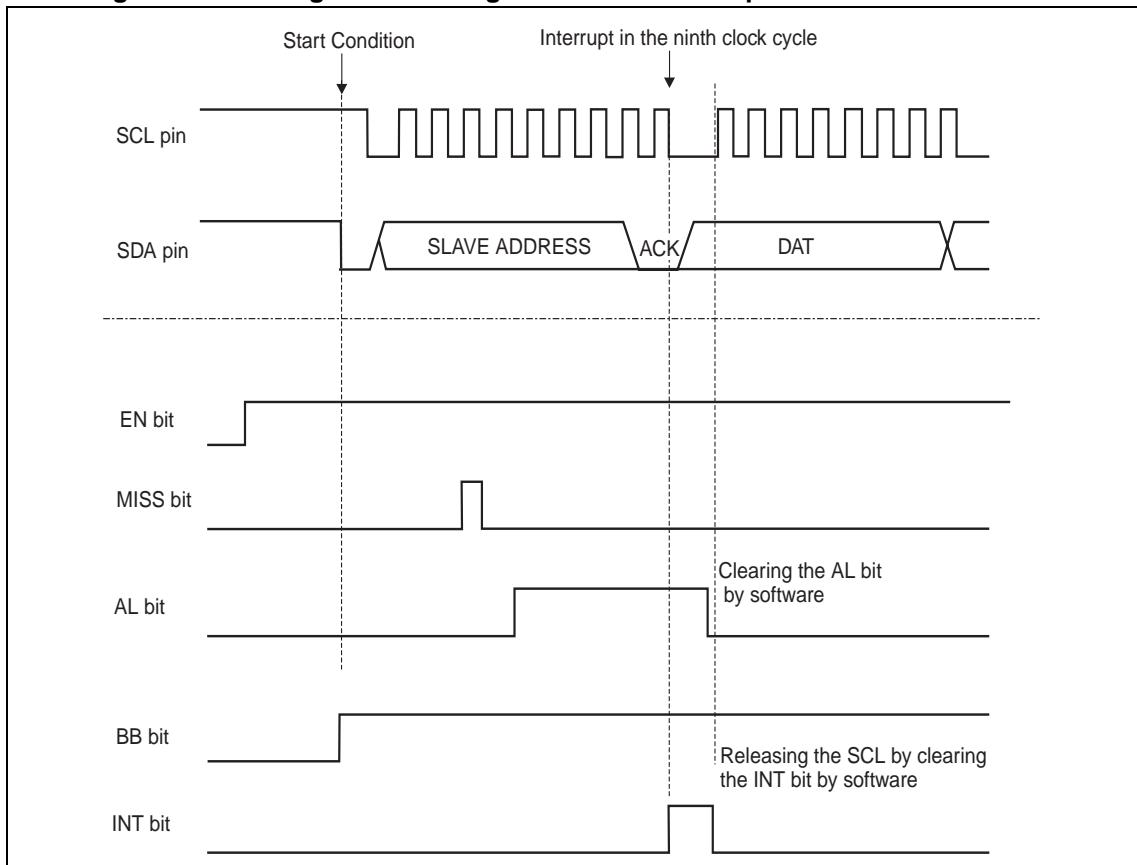
Figure 16.4-2 Diagram of Timing at which an Interrupt upon Detection of "AL Bit = 1" does not Occur



Example in which an interrupt (INT bit = 1) upon detection of "AL bit = 1" occurs

When an instruction which generates a start condition is executed (setting "1" to the MSS bit) with the bus busy detected (BB = 1) and the arbitration lost is performed, the INT bit interrupt is generated upon detection of AL bit =1.

Figure 16.4-3 Diagram of Timing at which an Interrupt in "AL Bit = 1" Occurs



If a symptom as described above can occur, follow the procedure below for software processing.

- 1) Execute the instruction that generates a start condition (set the MSS bit to "1").
- 2) Use, for example, the timer function to wait for the time * for three-bit data transmission at the I²C transfer frequency set in the ICCR register.

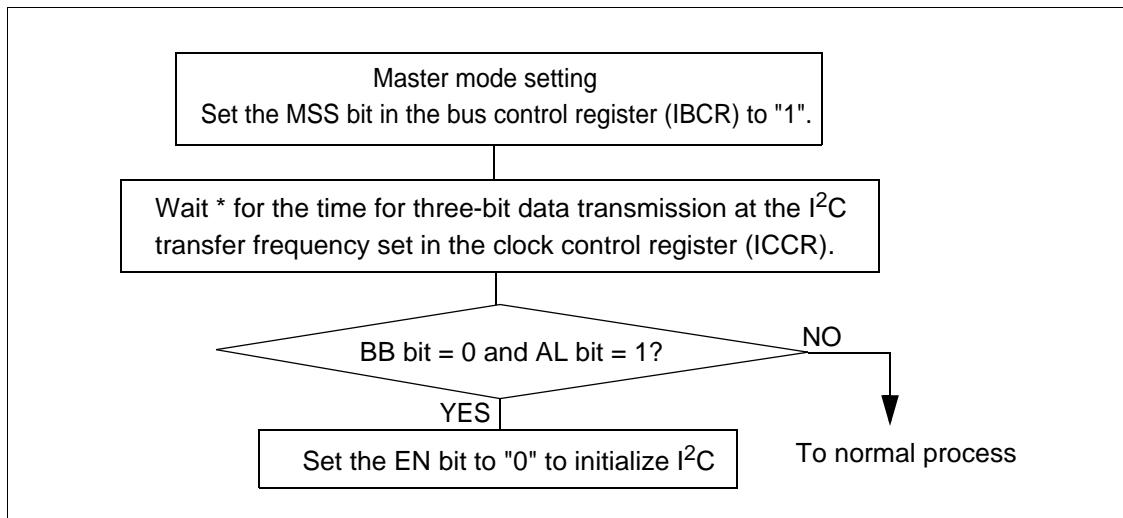
Example: Time for three-bit data transmission at an I²C transfer frequency of 100 kHz

$$\{1/(100 \times 10^3)\} \times 3 = 30 \mu\text{s}$$

- 3) Check the AL and BB bits in the IBSR register and, if the AL and BB bits are "1" and "0", respectively, set the EN bit in the ICCR register to "0" to initialize I²C. When the AL and BB bits are not so, perform normal processing.

CHAPTER 16 I²C INTERFACE

A sample flow is given below.



*: When "arbitration lost" is detected, the MSS bit is set to "1" and then the AL bit is set to "1" without fail after the time for three-bit data transmission at the I²C transfer frequency.

■ Clock Control Register (ICCR0/ICCR1)

bit	15	14	13	12	11	10	9	8
Address:	00009E _H	0000BE _H	TEST	-	EN	CS4	CS3	CS2
Initial value=>	0	0	0	1	1	1	1	1
	W	R	R/W	R/W	R/W	R/W	R/W	R/W

[bit15] Test bit

This bit is used for testing.

Be sure to write "0" to it.

[bit14] (Reserved)

This bit is unused.

Be sure to write "0" to it.

[bit13] EN (Enable)

This bit is the enable bit for the I²C interface.

0	Disabled
1	Enabled

If this bit is set to "0", all bits of the IBSR and IBCR registers (except the BER and BEIE bits) are cleared. This bit is cleared when a bus error occurs (BER = 1: IBCR).

Note:

If operation is disabled, the I²C interface immediately stops sending and receiving.

[bit12 to bit8] CS4 to CS0 (Clock Period Select 4 to 0)

These bits set the frequency of the serial clock.

These bits can be written only when operation is disabled (EN=0) or the EN bit is cleared.

The frequency of the shift clock, fsck, which is calculated as shown below.

$$fsck = \frac{\phi}{n \times 12+15} \quad N > 0 \quad \phi : \text{Machine clock (=CLKP)}$$

Register setting

n	CS4	CS3	CS2	CS1	CS0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
...
31	1	1	1	1	1

Setting disabled for CS4 to CS0=00000_B

Clock frequency CLKP [MHz]	100 kbps		400 kbps	
	n	fsck	n	fsck
34	27	100.3	6	390.8
25	20	98	4	396.8
17	13	99.4	3	333.3
12.5	9	101.6	2	320.5
8.5	6	97.7	1	314.8
6.25	4	99.2	1	231.5

CHAPTER 16 I²C INTERFACE

■ 10-bit Slave Address Register (ITBA0/ITBA1)

bit	15	14	13	12	11	10	9	8		
Address:	000096 _H /0000B6 _H	-	-	-	-	-	TA9	TA8	ITBAH	
	R	R	R	R	R	R	R/W	R/W		
Initial value =>	0	0	0	0	0	0	0	0		
bit	7	6	5	4	3	2	1	0		
Address:	000097 _H /0000A7 _H	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0	ITBAL
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Initial value =>	0	0	0	0	0	0	0	0		

Rewrite this register while operation is disabled (EN=0: ICCR).

[bit15 to bit10] (Reserved)

The values read from these bits are "0".

[bit9 to bit0] 10-bit slave address bits (A9 to A0)

If a 10-bit address is enabled (ENTB=1: ITMK), slave address data is received in the slave mode and then compared with the ITBA. An acknowledge is sent to the master after the address header of a 10-bit write access is received. If the received data of 1st and 2nd bytes and the TA9 bit to TA0 bit value are compared and produce a match, an acknowledge signal is sent to the master device and the AAS bit is set.

In addition, the I²C interface receives the address header of 10-bit read access after a repeated START condition is generated.

All bits of the slave address can be masked for the ITMK register setting.

The received slave address is overwritten to the ITBA register. This register is valid when the ASS bit of the IBSR register is set to "1".

■ 10-bit Slave Address Mask Register (ITMK0/ITMK1)

bit	15	14	13	12	11	10	9	8		
Address:	000098 _H /0000B8 _H	ENTB	RAL	-	-	-	-	TM9	TM8	
	R/W	R	R	R	R	R	R/W	R/W		
Initial value =>	0	0	1	1	1	1	1	1		
bit	7	6	5	4	3	2	1	0		
Address:	000099 _H /0000B9 _H	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Initial value =>	1	1	1	1	1	1	1	1		

[bit15] ENTB (10-bit slave address enable bit)

This bit is the 10-bit slave address enable bit.

0	10-bit slave address disabled
1	10-bit slave address enabled

Write access is enabled while the I²C interface is stopped (EN=0: ICCR).

[bit14] RAL (Slave address length bit)

This bit indicates the slave address length.

0	7-bit slave address
1	10-bit slave address

If the 10-bit and 7-bit slave address enable bits are both enabled (ENTB=1 and ENSB=1), this bit can be used to determine whether the transfer length of a 10-bit or 7-bit slave address becomes valid. This bit is valid if the AAS bit (IBSR) is set to "1".

This bit is cleared when the interface is disabled (EN = 0: ICCR).

This bit is read only.

[bit13 to bit10] (Reserved)

These bits are unused. The values read from these bits are always "1".

[bit9 to bit0] 10-bit slave address mask bits

These bits mask the bits of the 10-bit slave address register (ITBA). Write access is enabled while the I²C interface is disabled (EN=0: ICCR).

0	These bits not used for comparison of slave addresses
1	These bits used for comparison of slave addresses

Set these bits to enable transmission of an acknowledge to a compound 10-bit slave address. When using this register for comparison of 10-bit slave address, set these bits to "1". The received slave address is overwritten to ITBA. When ASS = 1(IBSR), the specified slave address can be determined by reading the ITBA register.

Each of TM9 to TM0 bits of ITMK corresponds to one bit of the ITBA address. If the value of each of the TM9 to TM0 bits is "1", the ITBA address becomes valid; if it is "0", the ITBA address becomes invalid.

Example: ITBA address is 0010010111_B and ITMK address is 111111100_B:

The slave address is in the space from 0010010100_B to 0010010111_B.

■ 7-bit Slave Address Register (ISBA0/ISBA1)

bit	7	6	5	4	3	2	1	0
Address: 00009B _H /0000BB _H	-	SA6	SA5	SA4	SA3	SA2	SA1	SA0
Initial value=>	R	R/W						

Rewrite this register while operation is stopped (EN=0: ICCR).

[bit7] (Reserved)

This bit is unused.

The value read from this bit is "0".

[bit6 to bit0] Slave address bits (SA6 to SA0)

If a 7-bit slave address is enabled (ENS_B = 1: ISMK) when slave address data is received in slave mode, these bits of ISBA and the received slave address data are compared. If a slave address match is detected, an acknowledge is sent to the master and the AAS bit is set.

CHAPTER 16 I²C INTERFACE

The I²C interface returns an acknowledge in response to reception of the address header of a 7-bit read access after a repeated START condition is generated.

All bits of a slave address are masked using of the ISMK. The received slave address data is overwritten to the ISBA register. This register is enabled only when AAS (IBSR register) is set to "1".

The I²C interface does not compare ISBA and the received slave address when a 10-bit slave address is specified or a general call is received.

■ 7-bit Slave Address Mask Register (ISMK0/ISMK1)

bit	15	14	13	12	11	10	9	8
Address: 00009A _H /0000BA _H	ENSB	SM6	SM5	SM4	SM3	SM2	SM1	SM0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value=>	0	1	1	1	1	1	1	1

Rewrite this register while operation is stopped (EN=0: ICCR).

[bit15] ENSB (7-bit slave address enable bit)

This bit is the 7-bit slave address enable bit.

0	7-bit slave address disabled
1	7-bit slave address enabled

[bit14 to bit8] 7-bit slave address mask bits

0	These bits not used for comparison of slave addresses
1	These bits used for comparison of slave addresses

Set these bits to enable transmission of an acknowledge to a compound 7-bit slave address.

When using this register for comparison of a 7-bit slave address, set these bits to "1". The received slave address is overwritten to ISBA. When ASS = 1 (IBSR), the specified slave address can be determined by reading the ISBA register.

After the I²C interface is enabled, the slave address (ISBA) is rewritten by reception operation. When ISMK is rewritten, ISMK must be set again to provide the expected operation.

Each of the SM6 to SM0 bits of ISMK corresponds to one bit of the ISBA address. If the value of each of the SM6 to SM0 bit is "1", the ISBA address becomes valid; if it is "0", the ISBA address becomes invalid.

Example: If ISBA address is 0010111_B and ISMK address is 1111100_B:
The slave address is in the space from 0010100_B to 0010111_B.

■ Data Register (IDAR0/IDAR1)

bit	7	6	5	4	3	2	1	0
Address:	00009DH/0000BDH	D7	D6	D5	D4	D3	D2	D0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value=> 0 0 0 0 0 0 0 0 0

[bit7 to bit0] Data bits (D7 to D0)

Bits D7 to D0 are a data register used for serial transfer. Data is transferred from the MSB.

Since the writing side of this register has double buffers, write data is loaded into the register for serial transfer while the bus is being used (BB=1). When the INT bit (IBCR) is cleared or the bus is idle (BB = 0: IBSR), the transfer data is loaded into the internal transfer register. Since, during reading, data is directly read from the register for serial transfer, receive data is valid only while the INT bit (IBCR) is set.

■ Clock Disable Register (IDBL0/IDBL1)

bit	7	6	5	4	3	2	1	0
Address:	00009FH/0000BFH	-	-	-	-	-	-	DBL
	R	R	R	R	R	R	R	R/W

Initial value=> 0 0 0 0 0 0 0 0 0

[bit0] Clock disable bit (DBL)

This bit specifies whether to supply or stop supply of the operating clock for the I²C interface.

This bit can be used in low-power consumption mode.

0	Supplies the clock for I ² C.
1	Stops supply of the clock for I ² C. The I ² C line is opened.

This bit is initialized to "0" by a reset. When "1" is written to this bit, the read value except this register (IBDL) is undefined and writing to other than this bit (this register) is invalid.

Note:

When this bit is set to "1", I²C immediately stops even if send and receive operation is in progress.

16.5 I²C Interface Operation

This section explains the operation of the I²C interface.

■ Operational Explanation

The I²C bus consists of two bidirectional bus lines used for transfer: one serial data line (SDA) and one serial clock line (SCL). The I²C interface has two corresponding open-drain I/O pins (SDA and SCL), enabling wired logic.

○ START Condition

Write "1" to the MSS bit while the bus is open (BB=0, MSS=0) to place the I²C interface in master mode and to generate a START condition.

The interface sends the value of the IDAR register as a slave address.

To generate a repeated START condition, write "1" to the SCC bit while the interrupt flag is set in bus master mode (MSS = 1 and INT = 1: IBCR).

Write "1" to the MSS bit while the bus is being used (BB=1 and TRX=0: IBSR, MSS=0 and INT=0: IBCR) to cause the interface to start transmission after waiting for the bus to be open. If, during this time, the interface in slave mode is receiving a write access, it starts transmission after the transfer is completed. Then the interface is open the bus.

If the interface is sending data, it does not start transmission even though the bus has been released.

To use this feature, it is important to check the followings:

- Whether the interface is specified as a slave (MSS=0: IBCR, AAS=1: IBSR).
- Whether data byte transmission is normal (AL=1: IBSR) when the next interrupt is received.

○ STOP Condition

Write "0" to the MSS bit in master mode (MSS=1, INT=1: IBCR) to generate a STOP condition and to place the interface in slave mode. Writing "0" to the MSS bit in any other state is irrelevant.

After the MSS bit is cleared, the interface tries to generate a STOP condition. However, a STOP condition will not be generated if the SCL line is driven to L. An interrupt is generated after the next byte is transferred.

Notes:

- It takes time from writing "0" to the MSS bit until the STOP condition is generated.
 - Disabling the I²C interface (DBL=1: IDBL or EN=0: ICCR) before the "STOP" condition occurs stops the operation immediately and generates an invalid clock on the SCL line.
 - Before disabling the I²C interface (DBL=1: IDBL or EN=0: ICCR), check that the "STOP" condition has occurred (BB=0: IBSR).
-

○ Slave Address Detection

In slave mode, BB=1 is set after a START condition is generated. The receive data from the master device is stored in the IDAR register.

When a 7-bit slave address is enabled (ISMK ENSB=1)

After 8-bit data is received, the IDAR and ISBA register values are compared. However, the bits masked in the ISMK register are not compared.

If the comparison result is a match, the AAS bit is set to "1", an acknowledge is sent to the master, and the value of bit0 of the receive data (bit0 of the IDAR register after reception) is inverted and stored in the TRX bit.

When a 10-bit slave address is enabled (ITMK ENTB=1)

If the header section of a 10-bit address {11110,TA1,TA0,write} is detected, an acknowledge is sent to the master and the value of bit0 of the receive data is inverted and stored in the TRX bit. No interrupt occurs at this time.

Then, the next transfer data and the low-order data of the ITBA register are compared. They are compared with the bits masked in the ISMK register. If the result is a match, the AAS bit is set to "1" and an acknowledge is sent to the master. An interrupt occurs at this time.

If the address specification as a slave has been made and a repeated START condition is detected, the AAS bit is set to "1" and an interrupt occurs after the header section of a 10-bit address {11110,TA0,TA1,read} is received.

The interface has two independent registers: a 10-bit slave address register (ITBA) and a 7-bit slave address register (ISBA). If both registers are enabled (ENSB=1: ISMK, ENTB=1: ITMK), an acknowledge can be sent to both 10-bit and 7-bit addresses.

The receive slave address length in slave mode (AAS = 1) can be determined using the RAL bit of ITMK register. In master mode, disabling both registers (ENSB = 0: ISMK, ENTB = 0: ITMK) can prevent a slave address from being generated for the I²C interface.

All slave addresses can be masked by setting the ITMK and ISMK registers.

○ Slave Address Mask

The slave address mask registers (ITMK/ISMK) can mask each of the bits of slave address registers. A bit set to "1" in the mask register is compared while a bit set to "0" is not compared.

A receive slave address can be recognized by reading the ITBA register (when a 10-bit address is received) or the ISBA register (when a 7-bit address is received) in the slave mode (ASS=1: IBSR).

If the bit mask is cleared, the interface can be used as the bus monitor because it is always accessed as a slave.

Note:

This feature does not become a real bus monitor because it returns an acknowledge when a slave address is received even though no other slave device is available.

CHAPTER 16 I²C INTERFACE

○ Master Addressing

In master mode, BB=1 and TRX=1 are set after a START condition is generated and the IDAR register value is sent starting with the MSB. After address data is sent and an acknowledge is received from a slave device, bit0 of the send data (bit0 of the IDAR register after transmission) is inverted and stored in the TRX bit. This operation is also performed for a repeated START condition.

Two bytes are sent for a 10-bit slave address during write access. The first byte data consists of the header section of that indicates a 10-bit sequence {11110,A9,A8,0} and the second byte data consists of the low-order 8-bit of the slave address {A7,A6,A5,A4,A3,A2,A1,A0}.

The series of transmissions described above bytes places the 10-bit slave device in the read access state and generates a repeated START condition as well as the header section {11110,A9,A8,1} that will be used for read access.

7-bit slave access	Write	START condition: A6,A5,A4,A3,A2,A1,A0,0
	Read	START condition: A6,A5,A4,A3,A2,A1,A0,1
10-bit slave access	Write	START condition: 11110,A9,A8,0,A7,A6,A5, A4,A3,A2,A1,A0
	Read	START condition: 11110,A9,A8,0,A7,A6, A5,A4,A3,A2, A1,A0 Repeated START condition: 11110,A9,A8,1

○ Arbitration

Arbitration occurs if, during sending in master mode, data is also sent by other master devices. Arbitration lost is recognized if the local device's transmission data is "1" and the level on the SDA line is set to "L". Then, AL=1 is set.

The AL bit is also set if an unnecessary START condition is detected in the first bit of the data and neither a START nor a STOP condition can be generated for the same reason.

If arbitration lost is detected, the MSS and TRX bits are set to "0" and the device immediately enters slave receive mode and returns an acknowledge when it receives the device's own slave address.

○ Acknowledge

The receiving device sends an acknowledge to the sending device.

The ACK bit (IBCR) sets whether an acknowledge should be sent when data is received.

If, during data transmission in slave mode (read access from other masters), an acknowledge is not returned from the master, the TRX bit is cleared to "0" and the device enters receive mode. This allows the master to generate a STOP condition when the slave opens the SCL line.

In master mode, read the LRB bit (IBSR) to check for an acknowledge.

○ Bus Error

A bus error is recognized and the I²C interface is stopped if:

- A violation of the basic convention on the I²C bus during data transfer (including the ACK bit) is detected.
- A stop condition in master mode is detected.
- A violation of the basic convention on the I²C bus while the bus is idle is detected.

○ **Communication Error that Causes No Errors**

If, during transmission in master mode, an illegal clock is generated on the SCL line due to noise or for some other reason, the transmission bit counter of the I²C interface may run quickly, causing the slave to hang while L has appeared on the SDA line in the ACK cycle.

An error (AL=1, BER=1) does not occur for such an illegal clock.

If this situation occurs, perform the following error processing:

1. A communication error can be assumed if LRB=1 when MSS=1, TRX=1, INT=1.
2. Set EN to "0" and then set EN to "1" to cause SCL to generate one clock on a pseudo basis. This action causes the slave to release the bus. The period from the time EN is set to "0" until EN is set to "1" must be long enough for the slave to recognize it as a clock (about as long as the H period of a transmission clock).
3. Since the IBSR and IBCR are cleared when EN is set to "0", perform retransmission processing from the START condition. No STOP condition is generated at this point if BSS is set to "0". Insert an interval equal to or longer than $n \times 7 \times t_{CPP}$ between the point at which EN is set to "1" and the point at which MSS is set to "1" (START condition).

Example:

High-speed mode: $6 \times 7 \times 30.3 = \text{about } 1.273 \mu\text{s}$

Standard mode: $27 \times 7 \times 30.3 = \text{about } 5.727 \mu\text{s}$

Note:

Since BER, if set, is not cleared even if EN is set to "0", first clear it and then resend it.

○ Other Items

1. Addressing after arbitration lost occurs

After arbitration lost occurs, check whether or not the local device is addressed using software.

When arbitration lost occurs, the device becomes a slave in terms of hardware. However, after one-byte transfer is completed, both the CLK and DATA lines are set to "L" level. Thus, if the device is not addressed, immediately open the CLK and DATA lines. If the device is addressed, open the CLK and DATA lines after preparing for slave transmission or reception (All of these things must be processed using software).

2. Interrupt condition when one-byte transfer is completed

An interrupt source is generated on I²C bus when one-byte transfer is completed or when an interrupt condition is met.

Since multiple interrupt conditions must be checked by one interrupt, each of the flags must be checked by the interrupt routine. The following lists the interrupt conditions used when one-byte transfer is completed:

- The device is a bus master.
- The device is an addressed slave.
- A general call address is received.
- Arbitration lost occurs.

3. Arbitration lost and interrupt source

When arbitration lost is detected, an interrupt source is generated, not immediately but after one-byte transfer is completed.

When arbitration lost is detected, the device becomes a slave in terms of hardware. However, in slave mode, a total of nine clocks must be output before an interrupt source can be generated. Thus, since an interrupt source is not immediately generated, no processing can be performed after arbitration lost occurs.

16.6 Operation Flowcharts

This section provides the operation flowcharts for the I²C interface.

■ Example of Slave Address and Data Transfer

Figure 16.6-1 Example of Slave Address and Data Transfer

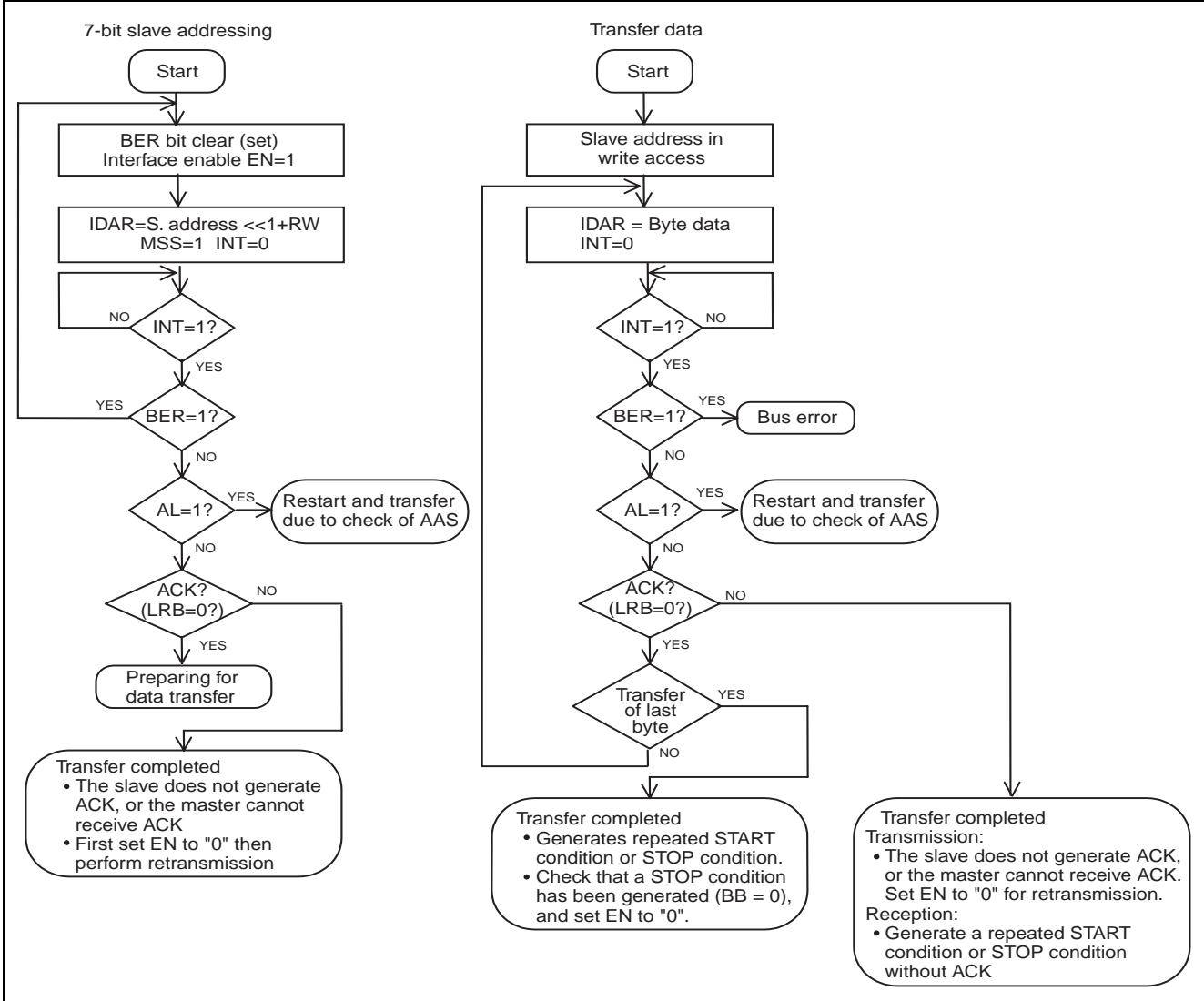


Figure 16.6-2 Example of Receive Data

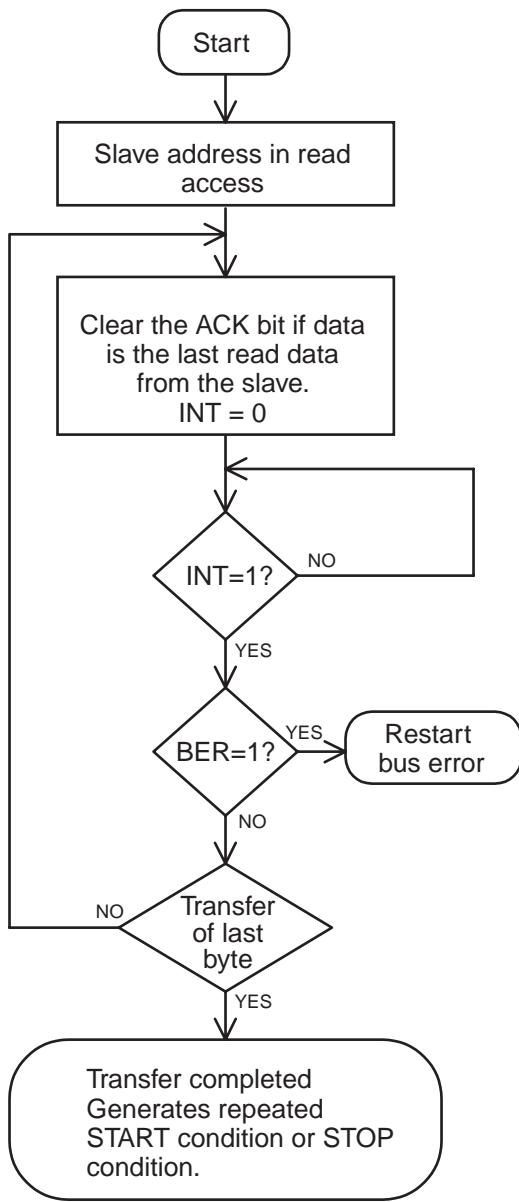
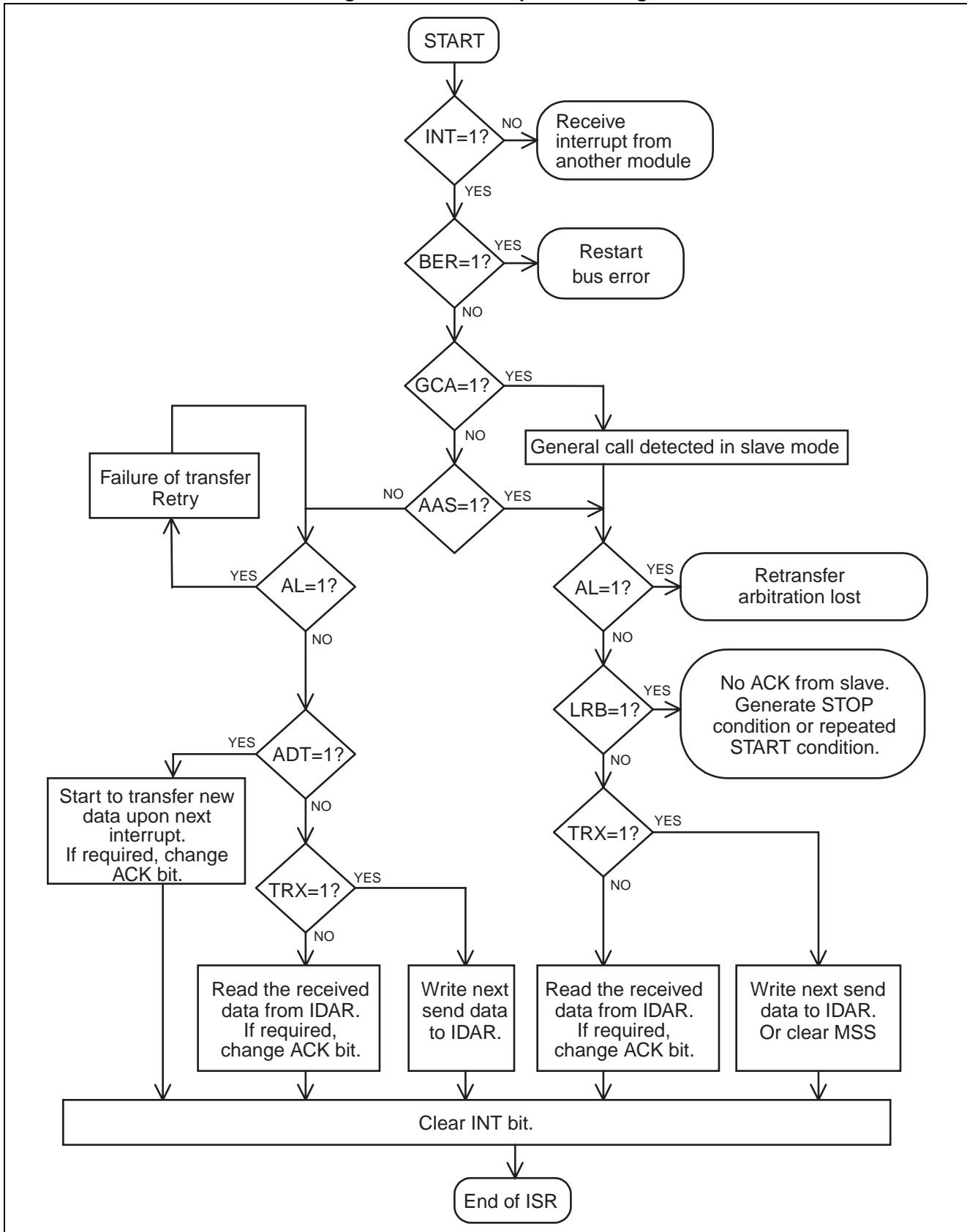


Figure 16.6-3 Interrupt Processing



CHAPTER 16 I²C INTERFACE

CHAPTER 17 16-BIT FREE RUN TIMER

This chapter gives an overview of the 16-bit free run timer, the configuration and functions of its registers, and its operation.

- 17.1 Overview of 16-bit Free Run Timer
- 17.2 Registers of the 16-bit Free Run Timer
- 17.3 Block Diagram of the 16-bit Free Run Timer
- 17.4 Details on Registers of the 16-bit Free Run Timer
- 17.5 Operation of the 16-bit Free Run Timer
- 17.6 Precautions on Using the 16-bit Free Run Timer

17.1 Overview of 16-bit Free Run Timer

This section explains the overview of the 16-bit free run timer

■ Overview

The 16-bit free run timer consists of a 16-bit up counter and control status register. The count values of this timer are used as a base timer for output compare and input capture operations.

- The user can select a count operating clock of four clocks.
- An interrupt can be generated as result of a counter overflow.
- The counter value can be initialized by the mode setting when a match with the compare register 0 of output compare occurs.

17.2 Registers of the 16-bit Free Run Timer

This section explains the registers of the 16-bit free run timer.

■ Registers of 16-bit Free Run Timer

Figure 17.2-1 Registers of Multifunctional Timer

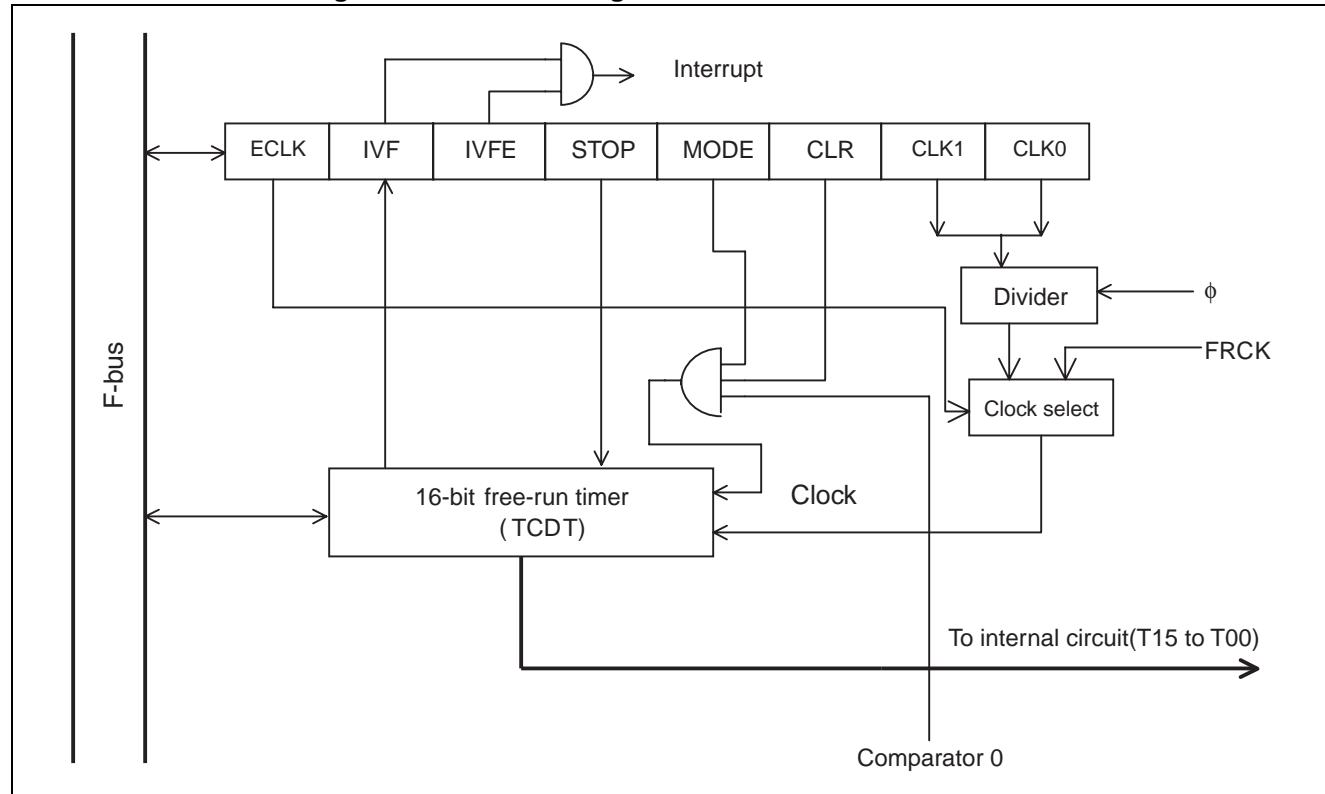
bit15	14	13	12	11	10	9	8	
T15	T14	T13	T12	T11	T10	T09	T08	Timer data register (high-order bits) (TCDT)
bit 7	6	5	4	3	2	1	0	
T07	T06	T05	T04	T03	T02	T01	T00	Timer data register (low-order bits) (TCDT)
bit 7	6	5	4	3	2	1	0	
ECLK	IVF	IVFE	STOP	MODE	CLR	CLK1	CLK0	Timer control status register (TCCS)

17.3 Block Diagram of the 16-bit Free Run Timer

This section shows the block diagram of the 16-bit free run timer.

■ Block Diagram of the 16-bit Free Run Timer

Figure 17.3-1 Block Diagram of the 16-bit Free Run Timer



17.4 Details on Registers of the 16-bit Free Run Timer

This section details the registers of the 16-bit free run timer.

■ Timer Data Register (TCDT)

TCDT	bit 15	14	13	12	11	10	9	8	
Address: 0000D4H	T15	T14	T13	T12	T11	T10	T09	T08	Initial value
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000H
	bit 7	6	5	4	3	2	1	0	
	T07	T06	T05	T04	T03	T02	T01	T00	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

This register allows reading the count value of the 16-bit free run timer. The counter value is cleared to "0000H" at reset. To set a timer value, write it to this register in the stop status (STOP = 1). Access this register in units of words. The 16-bit free run timer is initialized by following one of the methods below.

- Initialization by reset
- Initialization by clearing the control status register (CLR)
- Initialization by matching a compare clear register (ch.0 compare register) value with a timer counter value (A mode must be set.)

CHAPTER 17 16-BIT FREE RUN TIMER

■ Timer Control Status Register (TCCS)

TCCS Address: 0000D7H	bit 7	6	5	4	3	2	1	0	
	ECLK	IVF	IVFE	STOP	MODE	CLR	CLK1	CLK0	
	R/W (0)	Initial value							

[bit7]: ECLK

This bit selects the internal or external count clock source of the 16-bit free run timer.

The clock is changed immediately after writing to this bit. Change this bit while the output compare or input capture is stopped.

ECLK	Clock selection
0	Internal clock source is selected (initial value).
1	The clock is input from external terminal (FRCK).

Note:

If an internal clock is selected, the count clock must be set by bit1 and bit0 (CLK1 and CLK0). This count clock becomes a base clock. To input a clock signal from the FRCK, set the corresponding DDR bit to "0".

The minimum pulse width required for the external clock is 2T (T: peripheral clock machine cycle).

When the output compare unit is used with the external clock specified, a compare match and interrupt occur in the next clock cycle. To output a compare match and generate an interrupt, therefore, input at least one clock cycle after the compare match.

[bit6]: IVF

This bit is an interrupt request flag of the 16-bit free run timer. This bit is set to "1" either when the 16-bit free run timer causes an overflow or when mode setting causes a compare match with compare register 0. If an interrupt request permission bit (bit5: IVFE) is set, an interrupt occurs. This bit is cleared by setting it to "0". Writing "1" has no effect. The reading result of read-modify-write instructions is always "1".

IVF	Interrupt request flag
0	No interrupt request (initial value)
1	Interrupt request

[bit5]: IVFE

This bit is an interrupt permission bit for the 16-bit free run timer. When this bit is "1" and the interrupt flag (bit6: IVF) is set to "1", an interrupt occurs.

IVFE	Interrupt enable
0	Prohibits an interrupt (initial value)
1	Allows an interrupt.

[bit4]: STOP

This bit is used to stop the count of the 16-bit free run timer. When the bit is set to "1", the count of the timer is stopped. When the bit is set to "0", the count of the timer is started.

STOP	Count operation
0	Allows counting (operation) (initial value)
1	Prohibits counting (stop)

Note:

When the 16-bit free run timer is stopped, output compare operation is stopped as well.

[bit3]: MODE

This bit is used to set the initialization condition of the 16-bit free run timer. When this bit is set to "0", the counter value can be initialized by reset or with the clear bit (bit2: CLR). When the bit is set to "1", the counter value can be initialized by reset, with the clear bit (bit2: CLR), or by a match with the value of compare register 0 during an output compare operation.

MODE	Initialization of reset
0	Initialization by reset or the clear bit (initial value)
1	Initialization by reset, clear bit, or compare register 0

Note:

The counter value is initialized when the count value is changed.

[bit2]: CLR

This bit is used to initialize the value of the operating 16-bit free run timer to "0000_H". When this bit is set to "1", the counter is initialized to "0000_H". Setting this bit to "0" has no effect. The read value is always "0".

CLR	Meaning of flag
0	This value has no effect. (initial value)
1	The counter value is initialized to "0000 _H ".

Note:

The counter value is initialized when the count value is changed. When initializing the counter value while the timer is stopped, set the data register to "0000_H".

CHAPTER 17 16-BIT FREE RUN TIMER

[bit1, bit0]: CLK1, CLK0

These bits are used to select a count clock for the 16-bit free run timer. Immediately after these bits are set to a new value, the clock is switched. Therefore, change these bits while the output compare and input capture are stopped.

CLK1	CLK0	Count clock	$\phi=25\text{MHz}$	$\phi=12.5\text{MHz}$	$\phi=6.25\text{MHz}$	$\phi=3.125\text{MHz}$
0	0	$\phi/4$	160 ns	320 ns	640 ns	1.28 μs
0	1	$\phi/16$	640 ns	1.28 μs	2.56 μs	5.12 μs
1	0	$\phi/32$	1.28 μs	2.56 μs	5.12 μs	10.24 μs
1	1	$\phi/64$	2.56 μs	5.12 μs	10.24 μs	20.48 μs

ϕ : Machine clock

17.5 Operation of the 16-bit Free Run Timer

This section explains the operation of the 16-bit free run timer.

■ Operational Explanation

The 16-bit free run timer starts counting from the counter value " 0000_H " after releasing reset. The counter value becomes the reference time of the 16-bit output compare and the 16-bit input capture.

○ Explanation of 16-bit Free Run Timer Operation

The counter value is cleared under the following conditions.

- When an overflow occurs
- When the value matches that of the compare clear register (compare register of output compare ch.0) (A mode must be set.)
- When the CLR bit in the TCCS register is set to "1" during operation
- When " 0000_H " is written to TCDT while the timer is stopped
- When a reset occurs

An interrupt occurs when an overflow occurs and the counter is cleared because the counter value matches that of the compare clear register 0. (For a compare match interrupt, a mode must be set.)

Figure 17.5-1 Clearing the Counter when an Overflow Occurs

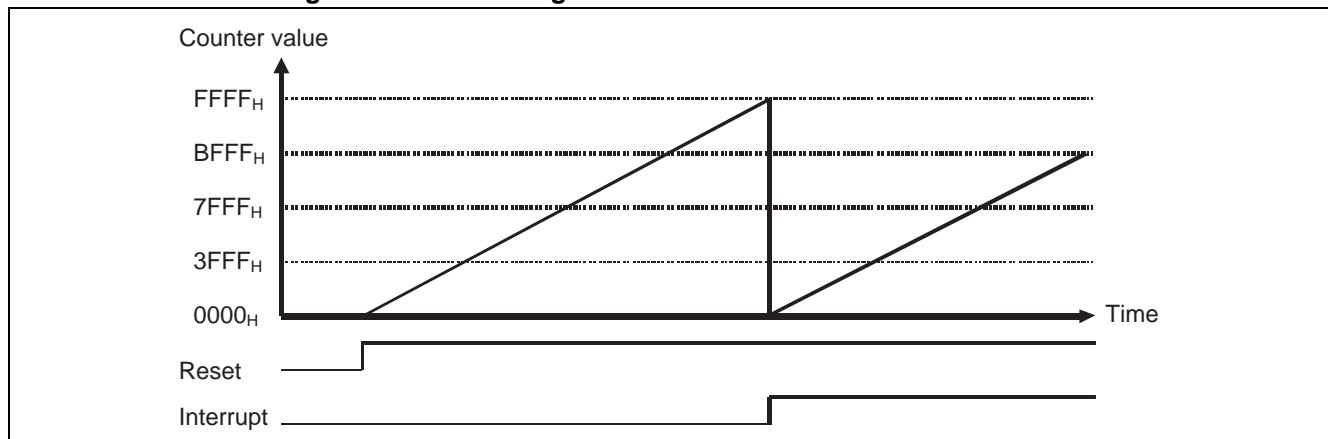
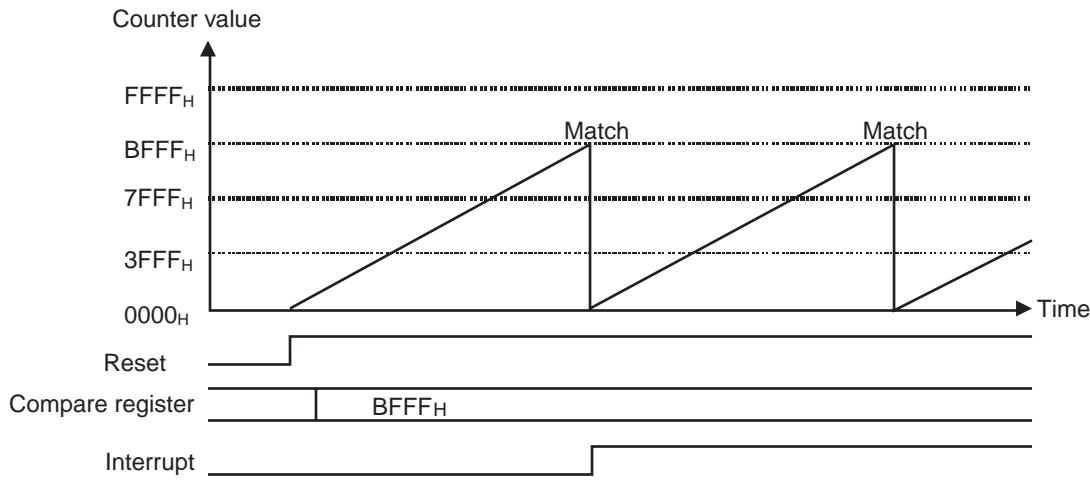
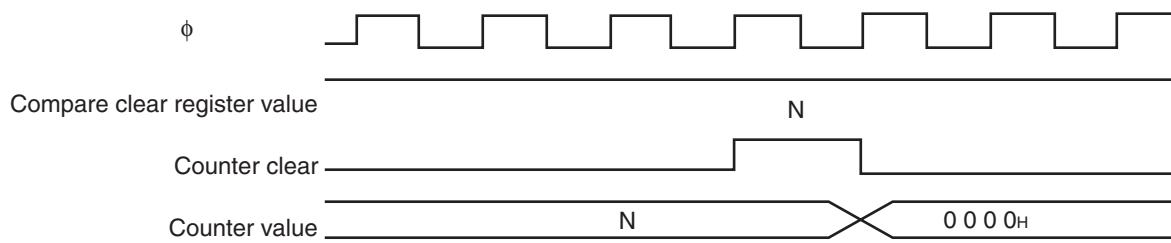


Figure 17.5-2 Clearing the Counter when the Counter Value Matches that of the Compare Clear Register

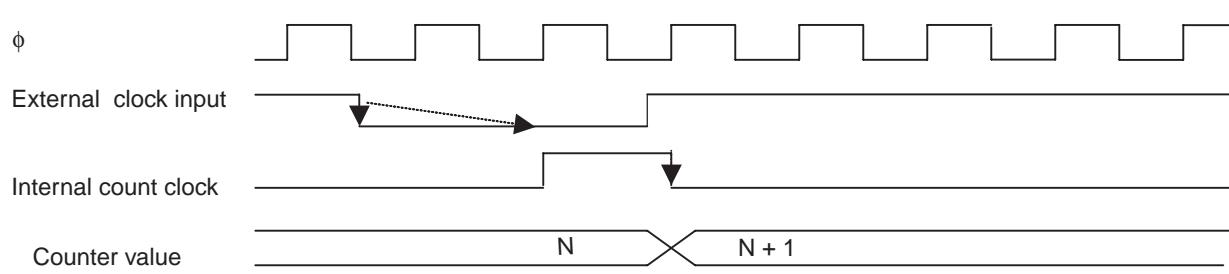
○ Timing to Clear the 16-bit Free Run Timer

The counter is cleared by reset, software, or when the counter value matches that of the compare clear register. When clearing the counter by way of reset or software, it is cleared immediately when a clear command is issued. However, when the counter is cleared because of a match with the value in the compare clear register 0, clearing is performed in synchronization with the count timing.

Figure 17.5-3 Clear Timing of the Free Run Timer

○ Count Timing of the 16-bit Free Run Timer

The 16-bit free run timer is incremented by the input clock (internal or external clock). When an external clock is selected, the falling edge (indicated by a down arrow) of the external clock is synchronized with the system clock, then the 16-bit free run timer counts up at the falling edge of the internal count clock.

Figure 17.5-4 Count Timing of the 16-bit Free Run Timer

17.6 Precautions on Using the 16-bit Free Run Timer

This section gives notes on the 16-bit free run timer.

■ Notes

1. If the interrupt request flag set timing and clear timing are simultaneously generated, the flag setting operation overrides the flag clearing operation.
2. When bit2 (counter initialize bit: CLR) in the control status register is set to "1", it holds the value until the internal counter clear timing and clears itself at that timing. If the clear timing and writing "1" are performed simultaneously, the writing takes precedence and the counter initialization bit keeps on holding "1" until the next clear timing.
3. The counter clear operation is valid only while the internal counter is operating (with the internal prescaler also operating). To clear the counter being stopped, set the timer count data register to "0000_H".

CHAPTER 17 16-BIT FREE RUN TIMER

CHAPTER 18 INPUT CAPTURE

This chapter explains the overview of the input capture, the configuration and functions of its registers, and its operation.

- 18.1 Overview of Input Capture
- 18.2 Input Capture Registers
- 18.3 Block Diagram of Input Capture
- 18.4 Details on Registers of Input Capture
- 18.5 Operation of Input Capture

18.1 Overview of Input Capture

This section explains the overview of the input capture.

■ Overview of Input Capture

The input capture module detects either or both of the rising and falling edges of an externally input signal, and stores the 16-bit free run timer value set at that time in a register. The input capture module can also generate an interrupt when an edge of the input signal is detected.

An input capture consists of an input capture data register, and a control register. An external input pin is assigned to each input capture.

- The user can select the effective edges (rising edge, falling edge, and both edges) of external input signals.
- Interrupts can be generated by detecting the effective edge of an external input signal.

Four channels for the input capture are contained.

18.2 Input Capture Registers

This section shows the input capture registers.

■ Registers of the Input Capture

Figure 18.2-1 Registers of the Input Capture

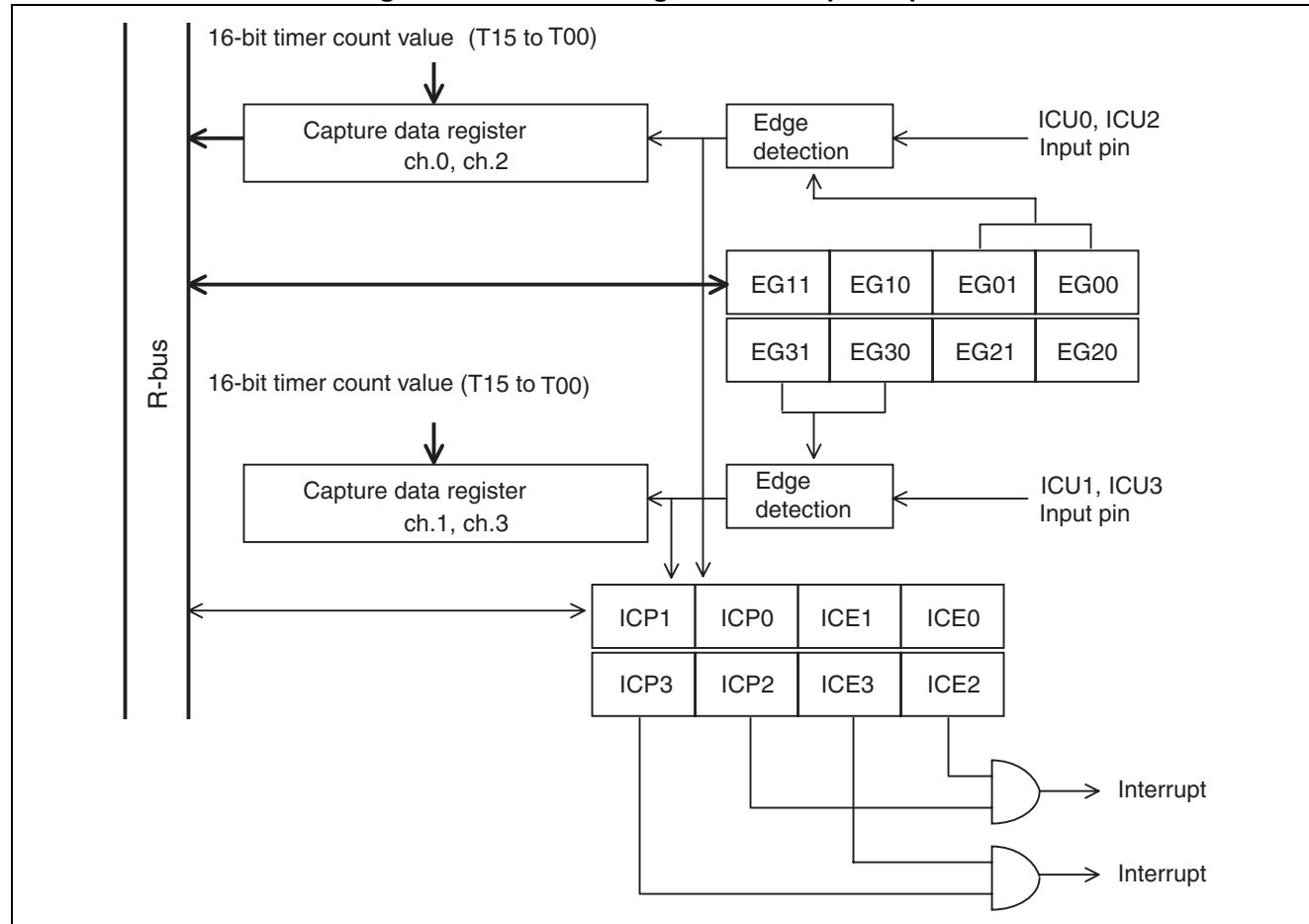
bit15	14	13	12	11	10	9	8	
CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08	Input capture data register (high-order bits) (IPCP)
bit7	6	5	4	3	2	1	0	
CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00	Input capture data register (low-order bits) (IPCP)
bit7	6	5	4	3	2	1	0	
ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20	Input capture control register (ICS23)
bit7	6	5	4	3	2	1	0	
ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	Input capture control register (ICS01)

18.3 Block Diagram of Input Capture

This section shows a block diagram of the input capture.

■ Block Diagram of the Input Capture

Figure 18.3-1 Block Diagram of the Input Capture



18.4 Details on Registers of Input Capture

This section describes the details on the registers of the input capture. The input capture has the following two data registers:

- **Input capture data registers (IPCP0 to IPCP3)**
- **Input capture control registers (ICS01, ICS23)**

■ Input Capture Data Registers (IPCP0 to IPCP3)

IPCP0 to IPCP3	bit15	14	13	12	11	10	9	8	
Address: 0000DA _H	CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08	Initial value
0000D8 _H	R	R	R	R	R	R	R	R	xxxxxxxx _B
0000DE _H									
0000DC _H									
	bit7	6	5	4	3	2	1	0	
0000DB _H	CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00	Initial value
0000D9 _H	R	R	R	R	R	R	R	R	xxxxxxxx _B
0000DF _H									
0000DD _H									

The input capture data registers (IPCP0 to IPCP3) are used to store the value of the 16-bits free run timer when an effective edge of the corresponding external pin input waveform is detected. This register is undefined at a reset. This register must be accessed in 16-bit or 32-bit data. The user cannot write any value to this register.

■ Input Capture Control Registers (ICS01, ICS23)

ICS23	bit7	6	5	4	3	2	1	0	
Address: 0000E1 _H	ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20	Initial value
	R/W	00000000 _B							
ICS01	bit7	6	5	4	3	2	1	0	
Address: 0000E3 _H	ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	Initial value
	R/W	00000000 _B							

[bit7, bit6]: ICP3, ICP2, ICP1, and ICP0

These bits are used as input-capture interrupt flags. When a effective edge of an external input pin is detected, these bits are set to "1". When the interrupt permission bits (ICE3, ICE2, ICE1, and ICE0) are also set, an interrupt is generated as soon as the effective edge is detected. To clear these bits, set them to "0". Setting these bits to "1" has no effect. Read operations with read-modify-write instructions always read "1" for these bits.

ICPn	Input capture interrupt flag
0	No effective edge is detected. (initial value)
1	An effective edge is detected.

"n" in ICPn indicates an input capture channel number.

[bit5, bit4]: ICE3, ICE2, ICE1, and ICE0

These bits are used as input-capture interrupt permission bits. When these bits are set to "1" and the interrupt flags (ICP3, ICP2, ICP1, and ICP0) are also set to "1", an input-capture interrupt occurs.

ICE _n	Input capture interrupt specification
0	Prohibits interrupts. (initial value)
1	Allows an interrupt.

"n" in ICE_n indicates the input capture channel number.

[bit3 to bit0]: EG31, EG30, EG21, EG20, EG11, EG10, EG01, and EG00

These bits are used to select the effective edge polarity of the external input. They are also used to enable input capture operations.

EG _n 1	EG _n 0	Edge detection polarity
0	0	No edge is detected. (stop status) (initial value)
0	1	A rising edge is detected. ↑
1	0	A falling edge is detected. ↓
1	1	Both edges are detected. ↑ & ↓

EG_n1 and EG_n0: n corresponds to the channel number of the input capture.

18.5 Operation of Input Capture

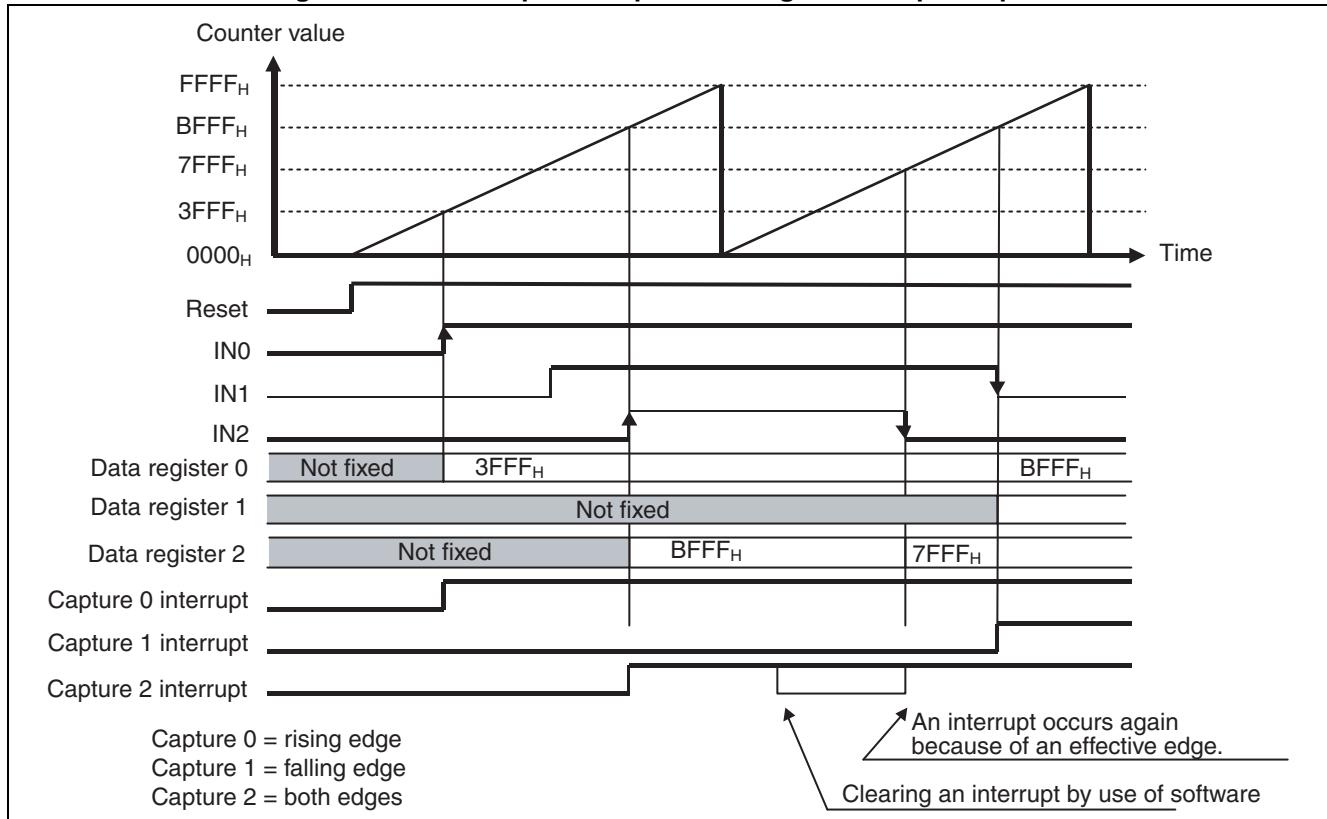
This section explains the operation of the input capture.

■ Operational Explanation

When the set effective edge is detected, the 16-bit input capture can capture the 16-bit free run timer value to the capture register to generate an interrupt.

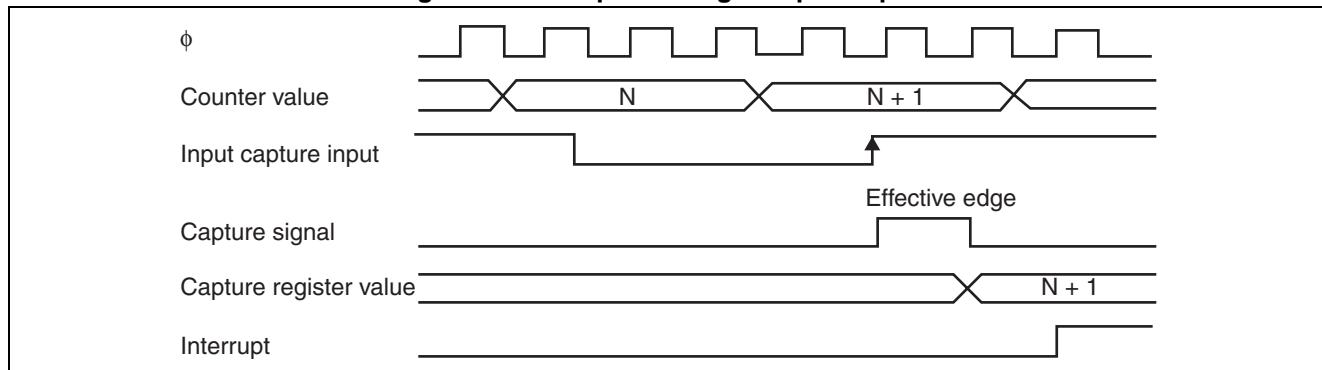
○ Operation of 16-bit Input Capture

Figure 18.5-1 Example of Capture Timing for the Input Capture



○ Input Timing of 16-bit Input Capture

Figure 18.5-2 Input Timing of Input Capture



CHAPTER 19 PROGRAM LOADER MODE (SUPPORTED ONLY BY THE MB91302A (IPL INTEGRATED MODEL))

This chapter outlines the program loader mode and describes the settings for the program loader and operations in that mode.

- 19.1 Overview of the Program Loader Mode
- 19.2 Setting the Program Loader
- 19.3 Operations in the Program Loader Mode
- 19.4 Example of Using the Program Loader Mode to Write to Flash Memory

19.1 Overview of the Program Loader Mode

This section gives an overview of the program loader mode.

■ Overview of the Program Loader Mode

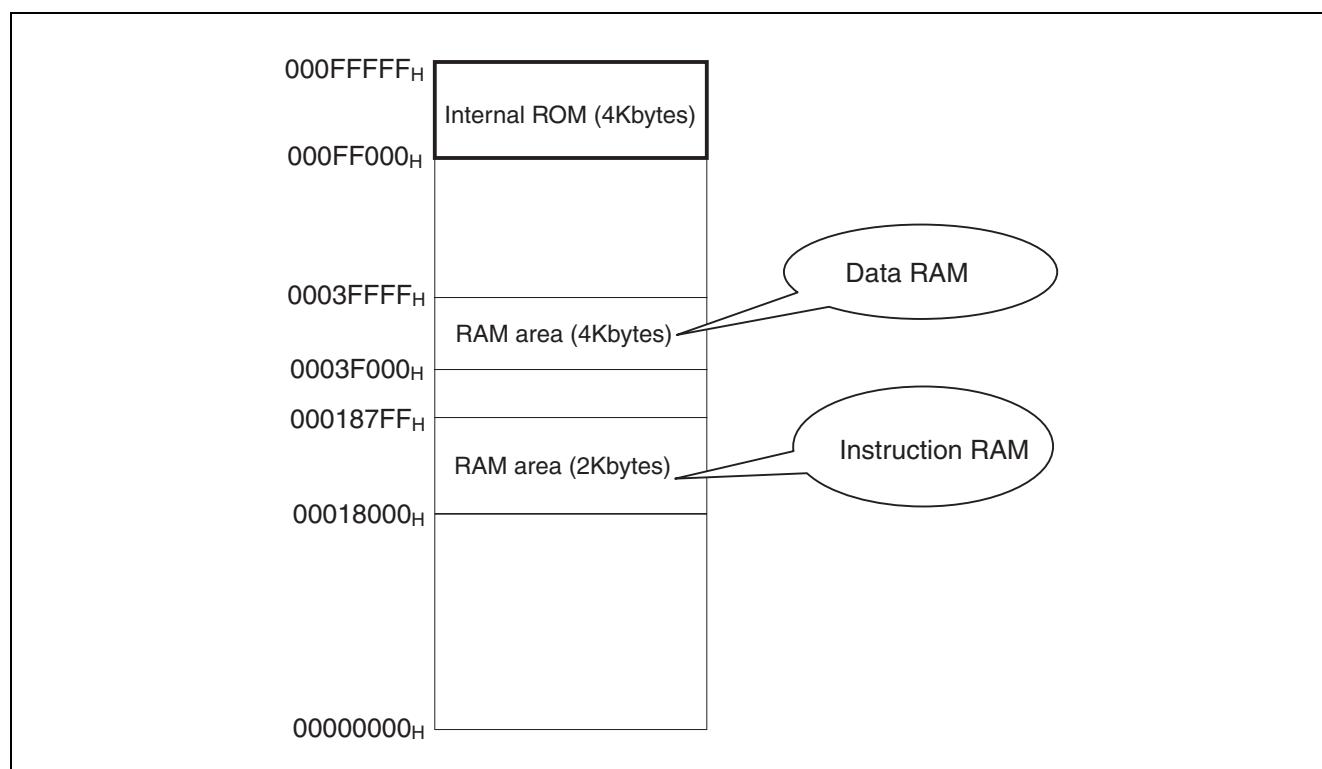
In the program loader mode, the program loader stored in the internal ROM uses UART ch.0 to perform serial communication with an external device, load a program from the external device to the internal RAM (2 Kbytes), and to start the loaded program.

For serial communication, asynchronous or synchronous communication can be selected depending on the state of the SIN0 pin of UART ch.0 upon initialization by $\overline{\text{INIT}}$. For asynchronous communication, however, the oscillation frequency is 17.0 MHz (quadrupled by the PLL to 68.0 MHz as the CPU's operating clock frequency). For asynchronous communication, be sure to use the device at an oscillation frequency of 17.0 MHz as it operates at a baud rate of 9600 bps.)

Note that the program loader mode is supported only by the MB91302A (IPL integrated model).

■ Memory Map

The loader program stored in the internal ROM (4K bytes) is executed in the internal-ROM/external-bus mode, resulting in a memory map as shown below. Programs can be located in the following instruction RAM area (instruction cache set for RAM mode). To access, for example, an external area, use the downloaded program to make the required register settings.



19.2 Setting the Program Loader

This section describes how to set the program loader.

■ Setting the Program Loader

The program loader stored in internal ROM is started when the MD2, MD1, MD0, and SIN0 pins are set as in Table 19.2-1 during initialization by $\overline{\text{INIT}}$.

For UART ch.0 used for serial communication with external devices, asynchronous or synchronous communication is determined depending on the state of the SIN0 pin upon initialization by $\overline{\text{INIT}}$. For the settings of SIN0, Figure 19.2-1 and Figure 19.2-2 show its reset timings.

Table 19.2-1 Settings for the MD2, MD1, MD0, and SIN0 Pins upon Initialization by $\overline{\text{INIT}}$

Specifications	Pin Name			
	MD2	MD1	MD0	SIN0
Asynchronous Communication	0	0	0	1
Synchronous Communication	0	0	0	0

Figure 19.2-1 Reset Timing (Asynchronous Communication)

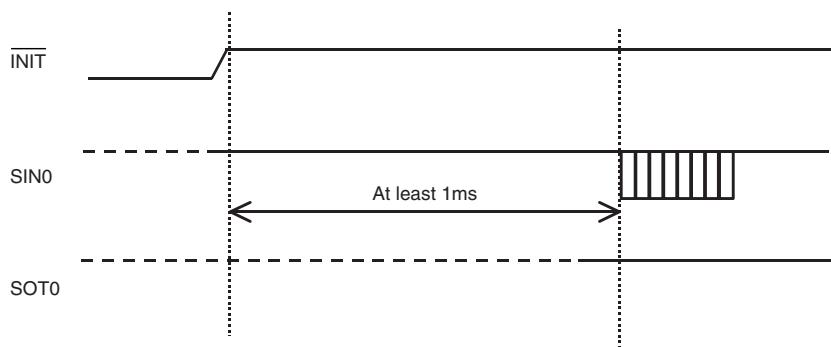
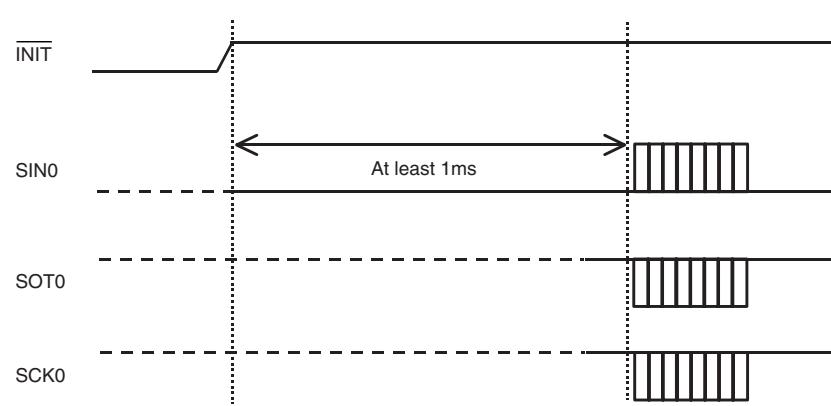


Figure 19.2-2 Reset Timing (Synchronous Communication)



19.3 Operations in the Program Loader Mode

This section describes the operations for asynchronous communication and synchronous communication in the program loader mode.

■ Operations in the Program Loader Mode

○ Asynchronous communication at an oscillation frequency of 17.0 MHz

Serial communication is performed in the UART ch.0 asynchronous mode (mode 0).

The baud rate is 9600 bps at a machine clock frequency of 68.0 MHz (obtained by quadrupling an oscillation frequency of 17.0 MHz using the PLL).

The settings for serial communication are a data length of 8 bits, a stop bit length of 1 bit, no parity, and LSB-first.

○ Synchronous Communication

Serial communication is performed in the UART ch.0 synchronous mode (mode 2). The baud rate can be selected freely with the clock input (SCK0) (The frequency of clock input SCK0 is used as the baud rate as it is).

The maximum frequency of the clock input is up to 1/8 of the frequency of the peripheral operating clock signal without exceeding 3.125 MHz.

The peripheral operating clock frequency is given by the following equation.

Peripheral operating clock frequency = machine clock frequency (obtained by quadrupling the oscillation frequency using the PLL)/2

The settings for the serial communication are a data length of 8 bits, no parity, and LSB-first.

In either mode

- Command data (00_H)
- Four bytes of download destination RAM address (00018000_H to $000187FF_H$)
- Downloaded four bytes (up to $000007FF_H$)

Three items of download information data are given to the FR side byte by byte, starting at the high-order byte, and their checksum data (the lower eight bits taken from the sum of all the data items) and entered the downloaded routine to the RAM. Then, the data to be downloaded to internal RAM is given to the FR side byte by byte starting at the high-order byte in the same way and checksum data is also given. Upon completion of transfer, a jump to RAM takes place and the downloaded program is executed.

■ Commands

Listed below are the commands issued to the FR and the response signals from the FR.

			FR
Command	Download	(Reception)	-> 00 _H (PC, etc.)
	Reset	(Reception)	-> 18 _H (PC, etc.)
	RAM Jump	(Reception)	-> C0 _H (PC, etc.)
Command Response	Abnormal Command	(Reception Command & F0 _H) 04 _H	-> (Reception) (PC,etc.)
	Abnormal checksum	{Reception Command (00 _H) & F0 _H } 02 _H	-> (Reception) (PC,etc.)
	RESET Command Reception	11 _H	-> (Reception) (PC,etc.)
	DOWN LOAD Command Reception	01 _H	-> (Reception) (PC,etc.)

■ Operation Example

- Transferring $0000005B_H$ - byte data to RAM address 00018000_H

	Processing order	PC,etc.		FR
Command Data	1	00_H	->	(Reception)
Download destination address	2	00_H	->	
	3	01_H	->	
	4	80_H	->	
	5	00_H	->	
Number of download bytes (91 bytes)	6	00_H	->	(Reception)
	7	00_H	->	
	8	00_H	->	
	9	$5B_H$	->	
Checksum Data	10	DC_H	->	(Reception)
Acknowledge data transmission from the FR	11	(Reception)	<-	01_H
Data Transmit	12	DATA	->	(Reception)
Checksum Data	13	*	->	(Reception)

*: The lower eight bits are fetched from all transmit data items added together.

- Program counter causing a jump to a RAM address after data transfer

	Processing order	PC,etc.		FR
Command Data	1	$C0_H$	->	(Reception)
Dummy Data	2	00_H	->	
	3	00_H	->	
	4	00_H	->	
	5	00_H	->	
Dummy Data	6	00_H	->	(Reception)
	7	00_H	->	
	8	00_H	->	
	9	00_H	->	
Checksum Data	10	$C0_H$	->	(Reception)
Jump to a RAM	11	-	-	*

*: The jump destination contained in the program counter is the download destination address specified when data is transferred to RAM. Command data ($C0_H$), dummy data (8 bytes), and checksum data are required before a jump can take place.

- Issuing a reset command, for example, from a personal computer

	Processing order	PC,etc.		FR
Command Data	1	18_H	->	(Reception)
Reset	3	-	-	*

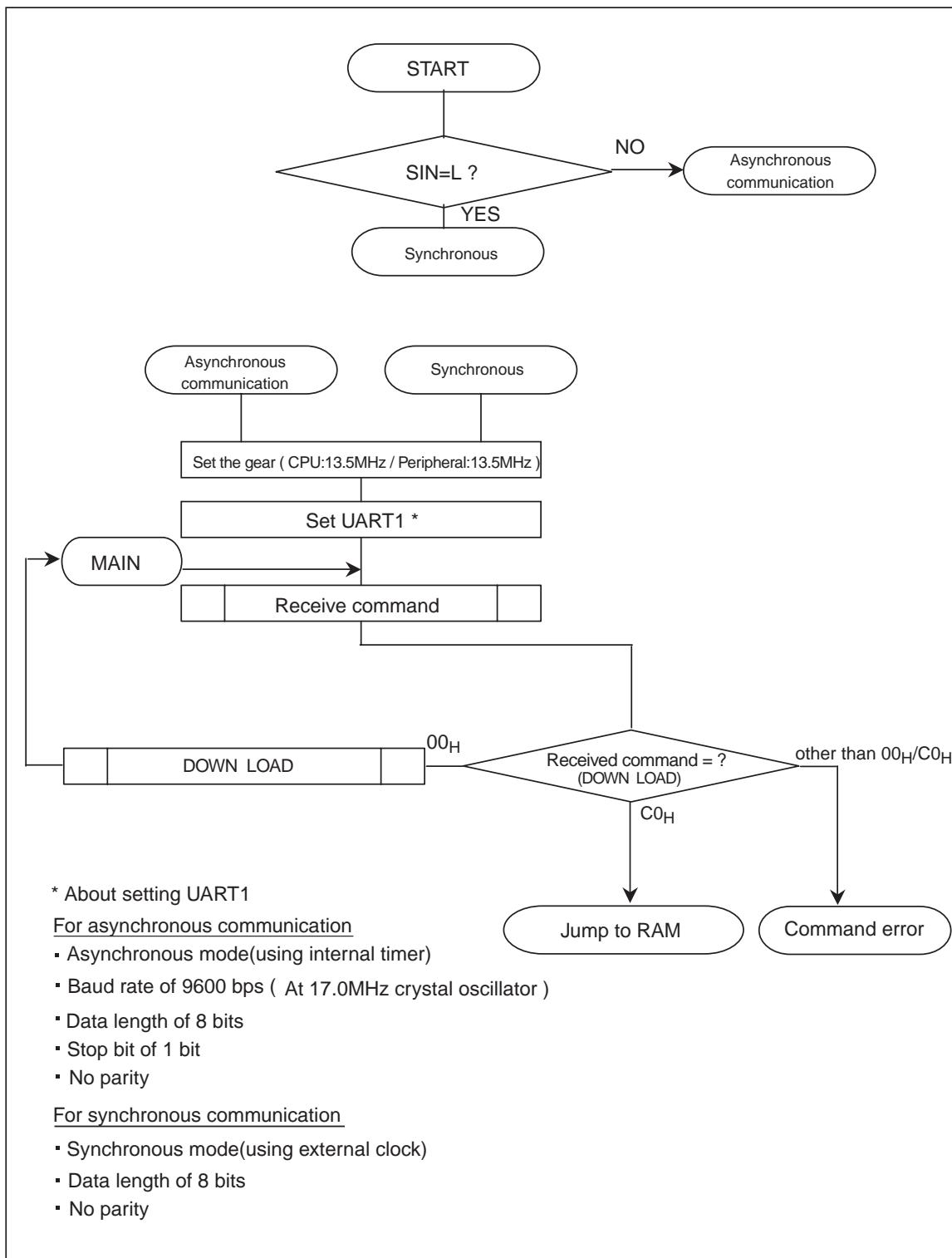
*: Issuing command data 18_H , for example, from a personal computer, causes immediate transition to the reset sequence.

For detailed operations, see the flowcharts for dedicated ROM embedded programs from the next page on.

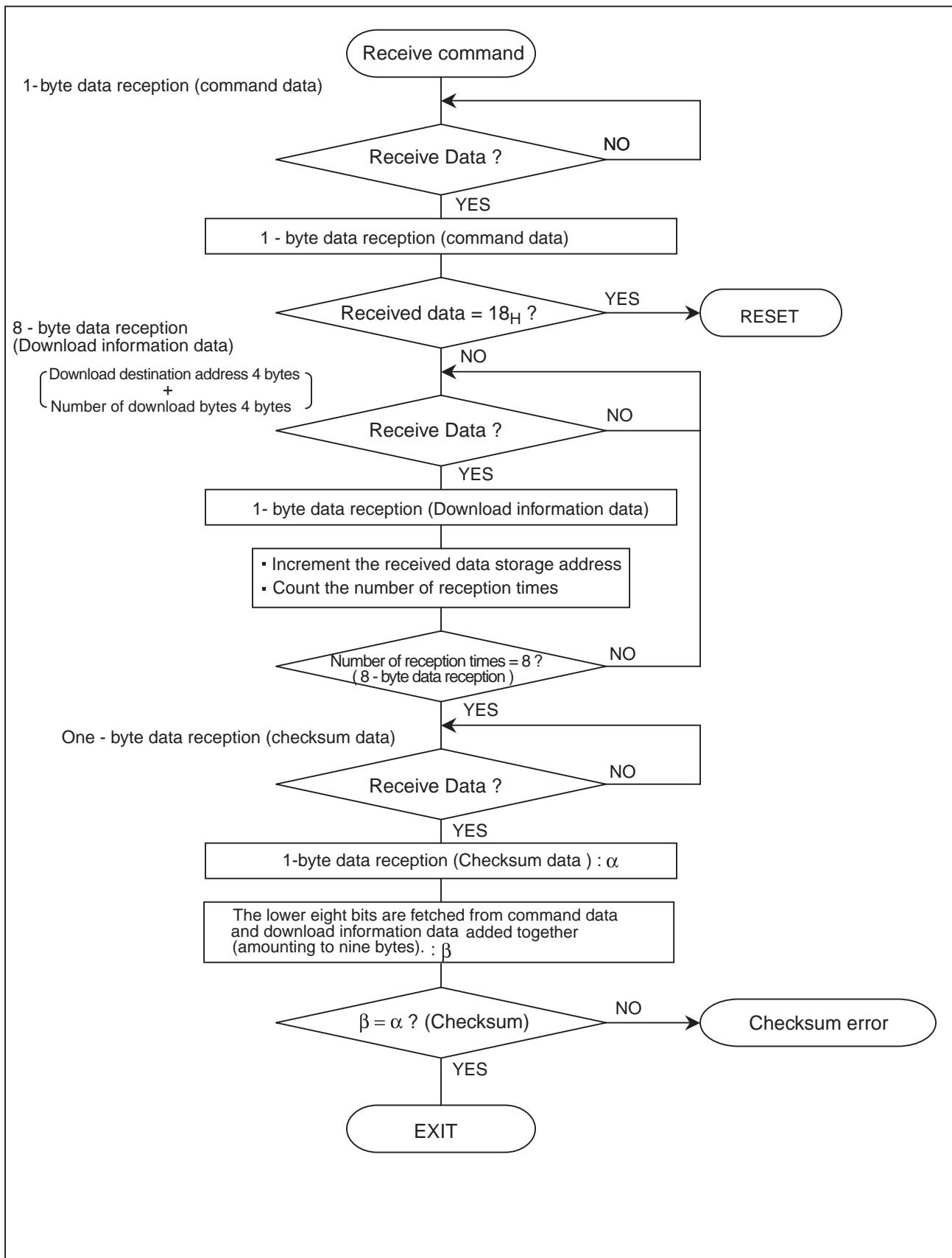
For detailed operations of the UART and the states of all of its pins, see "CHAPTER 13 UART" or the "At initialization (INIT)" column of the "Pin State Table" in the Appendix C.

■ Flowcharts

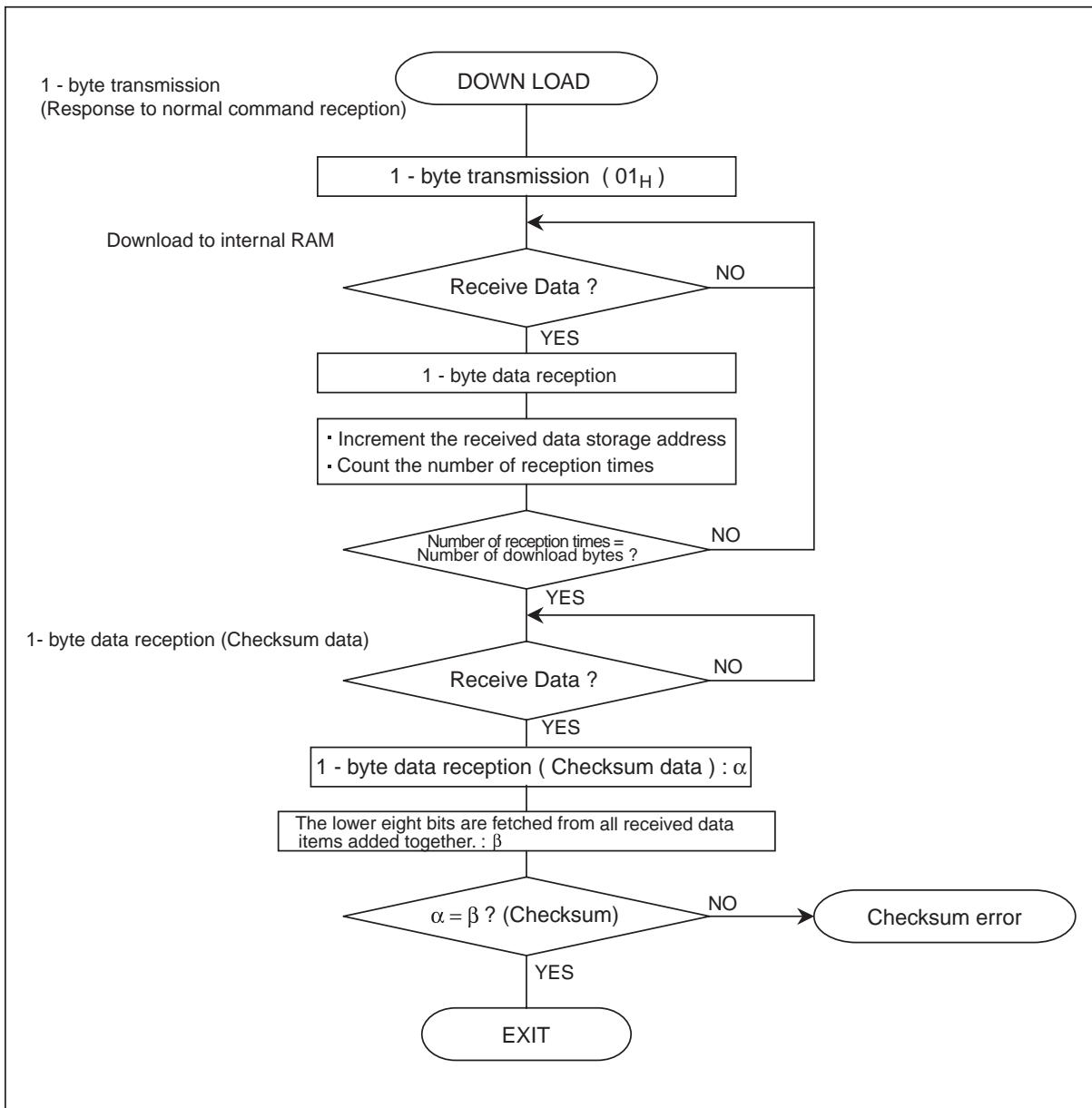
○ Main program flowchart



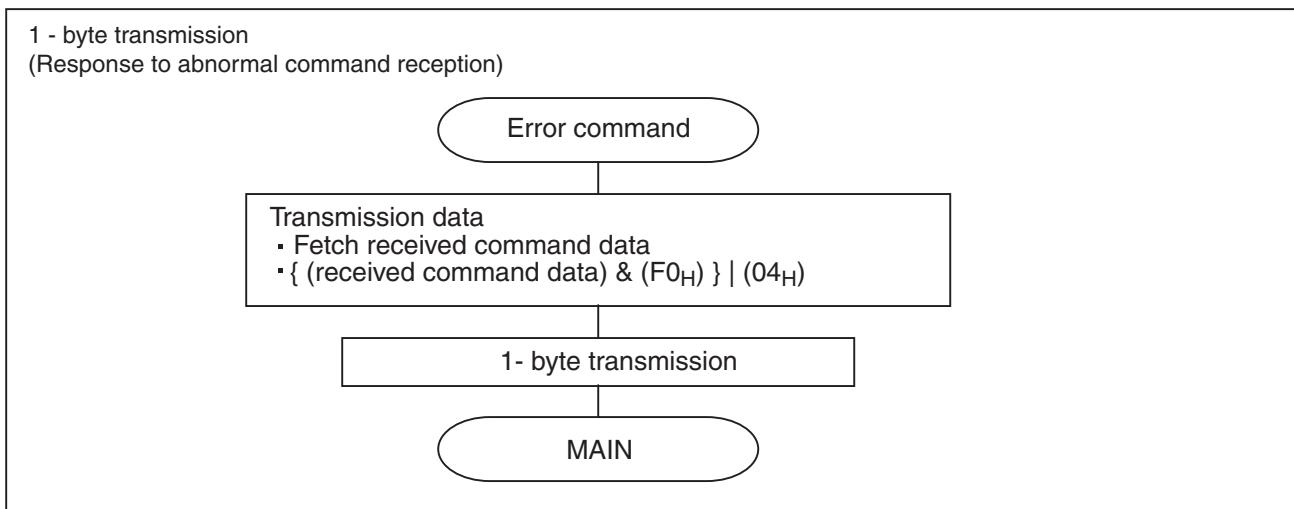
○ Subroutine "Command reception" in Asynchronous mode



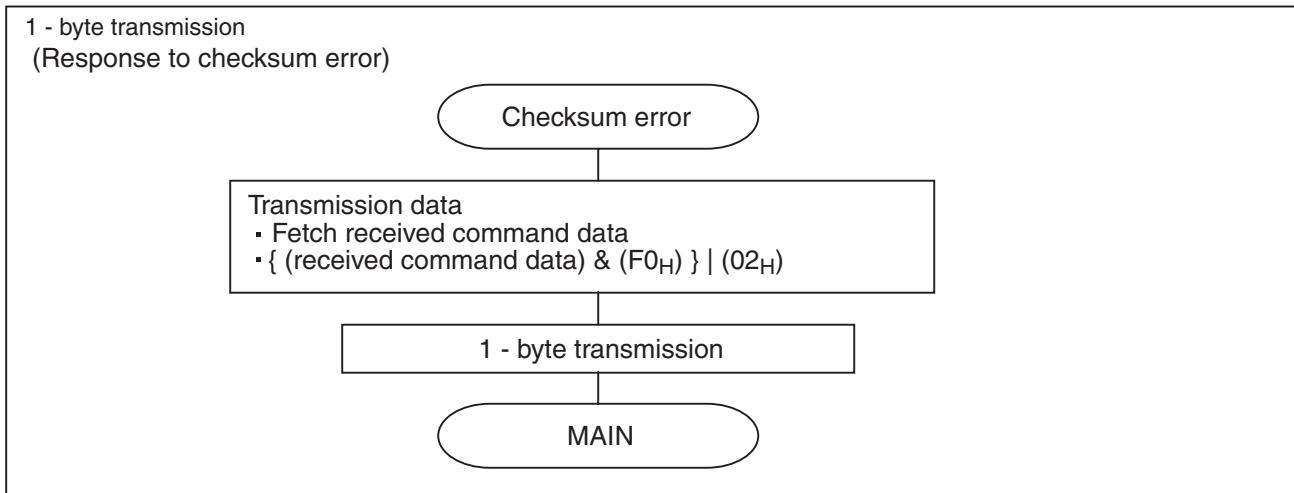
○ Subroutine "DOWN LOAD" in Asynchronous mode



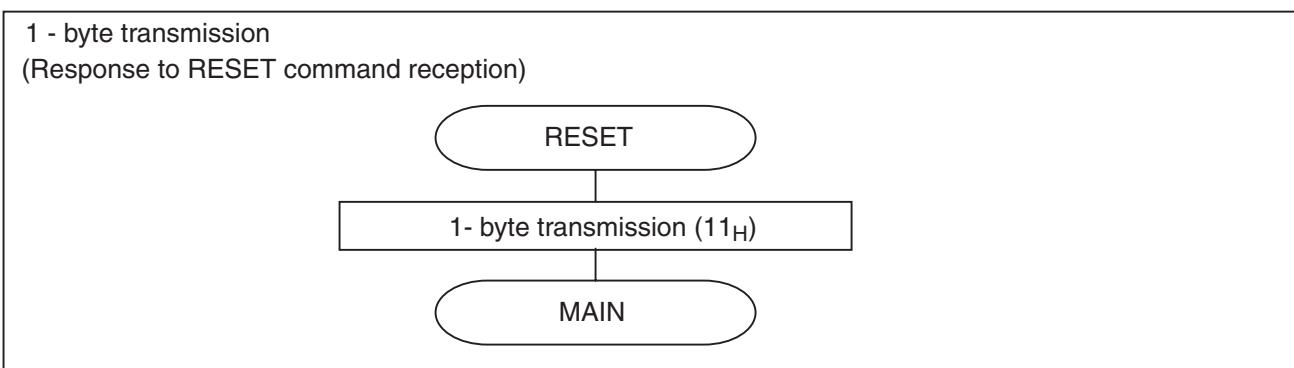
○ Subroutine "Command error" in Asynchronous mode



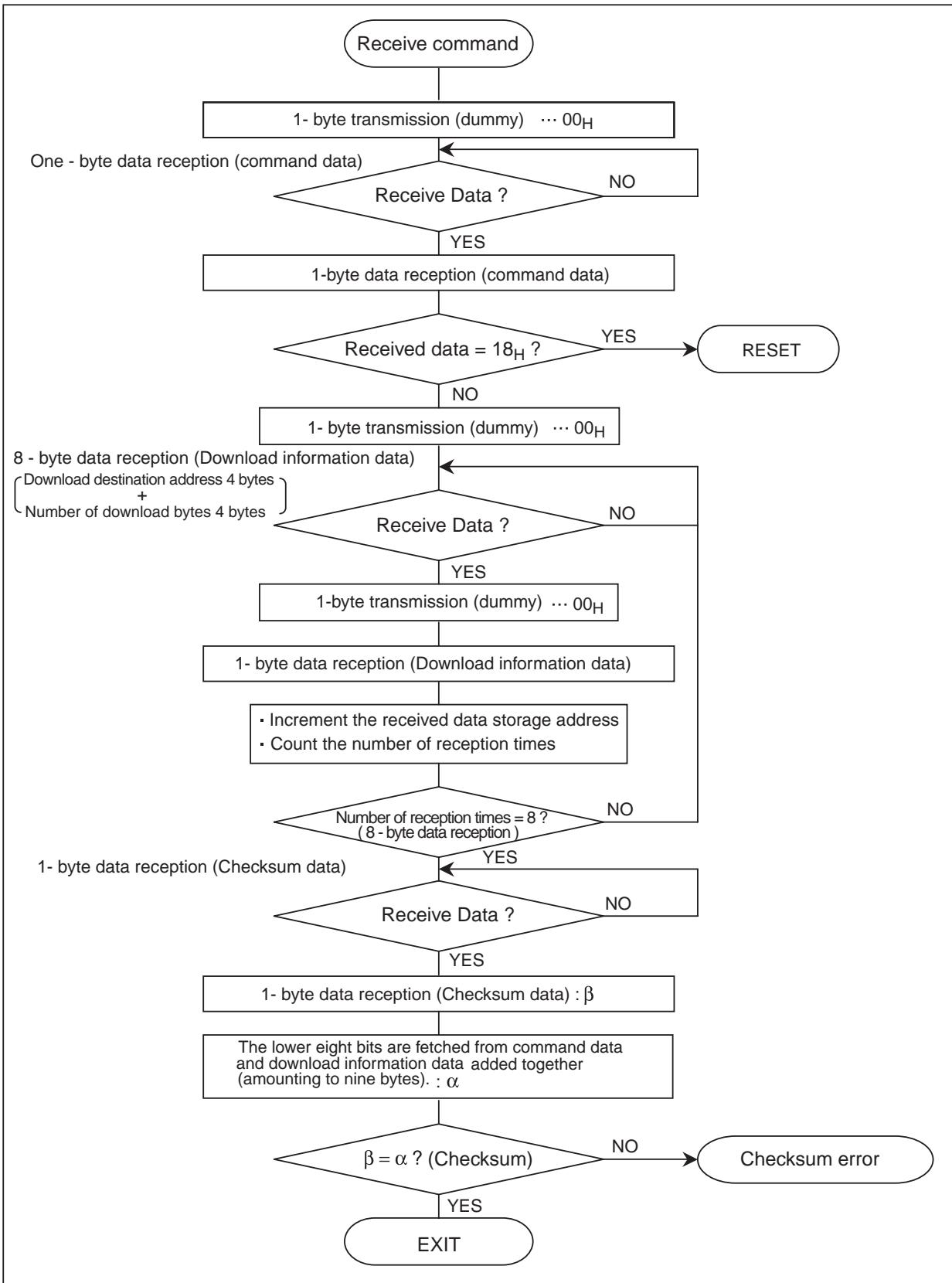
○ Subroutine "Checksum error" in Asynchronous mode



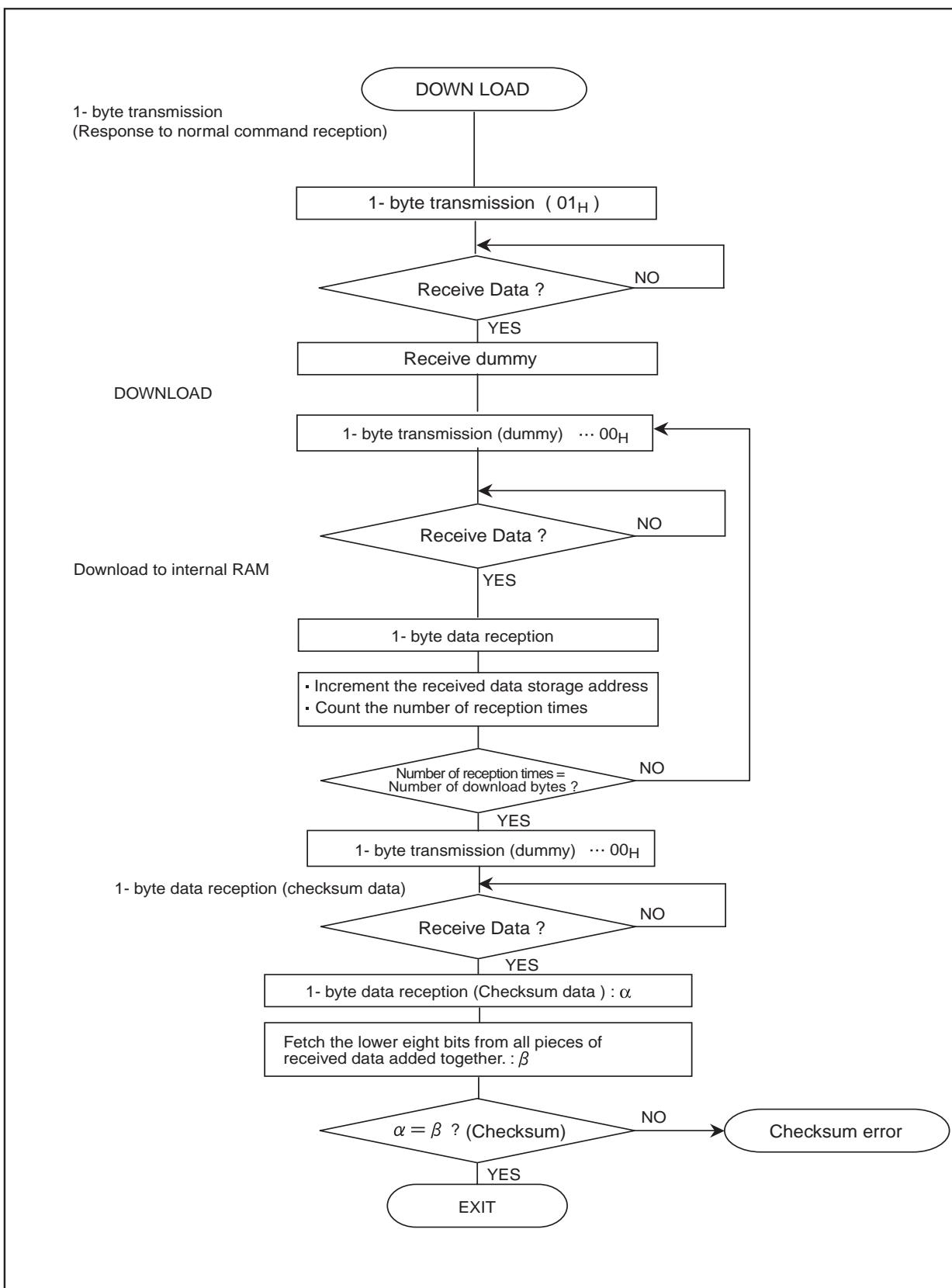
○ Subroutine "RESET" in Asynchronous mode



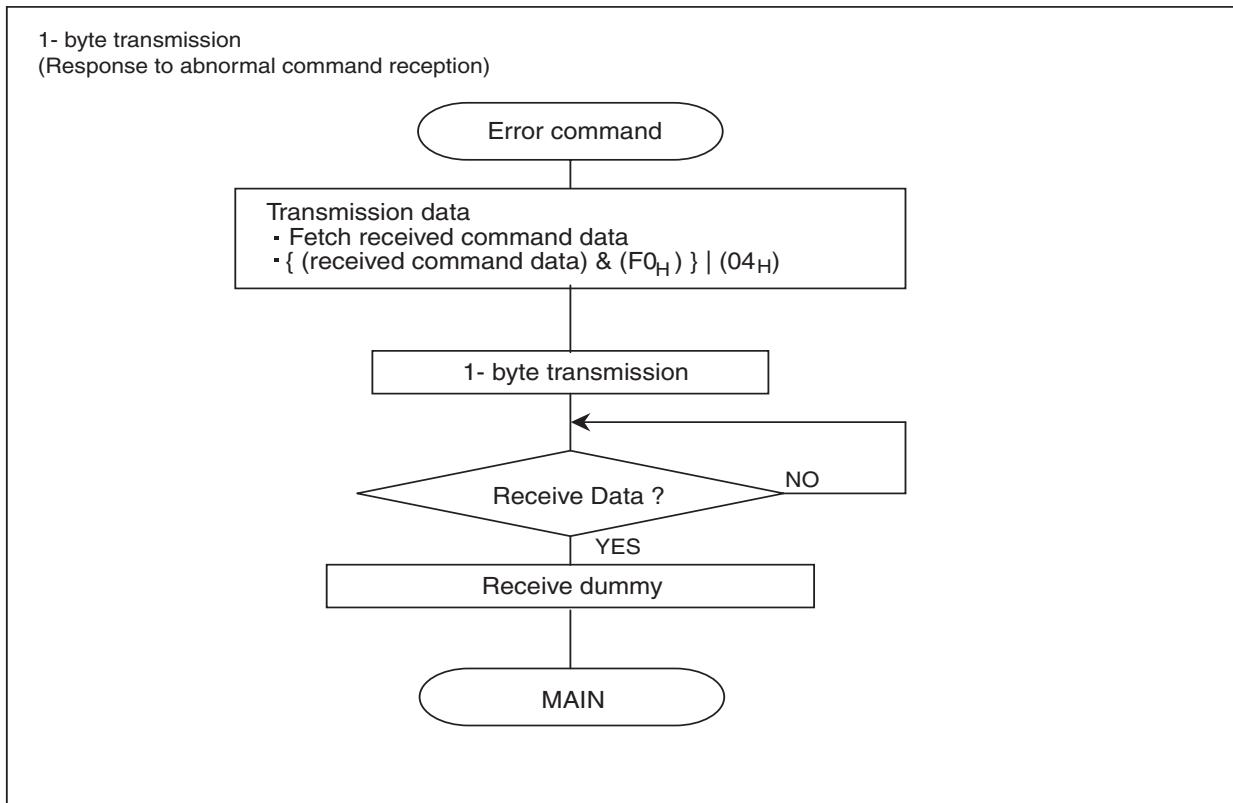
○ Subroutine "Command Reception" in Synchronous mode



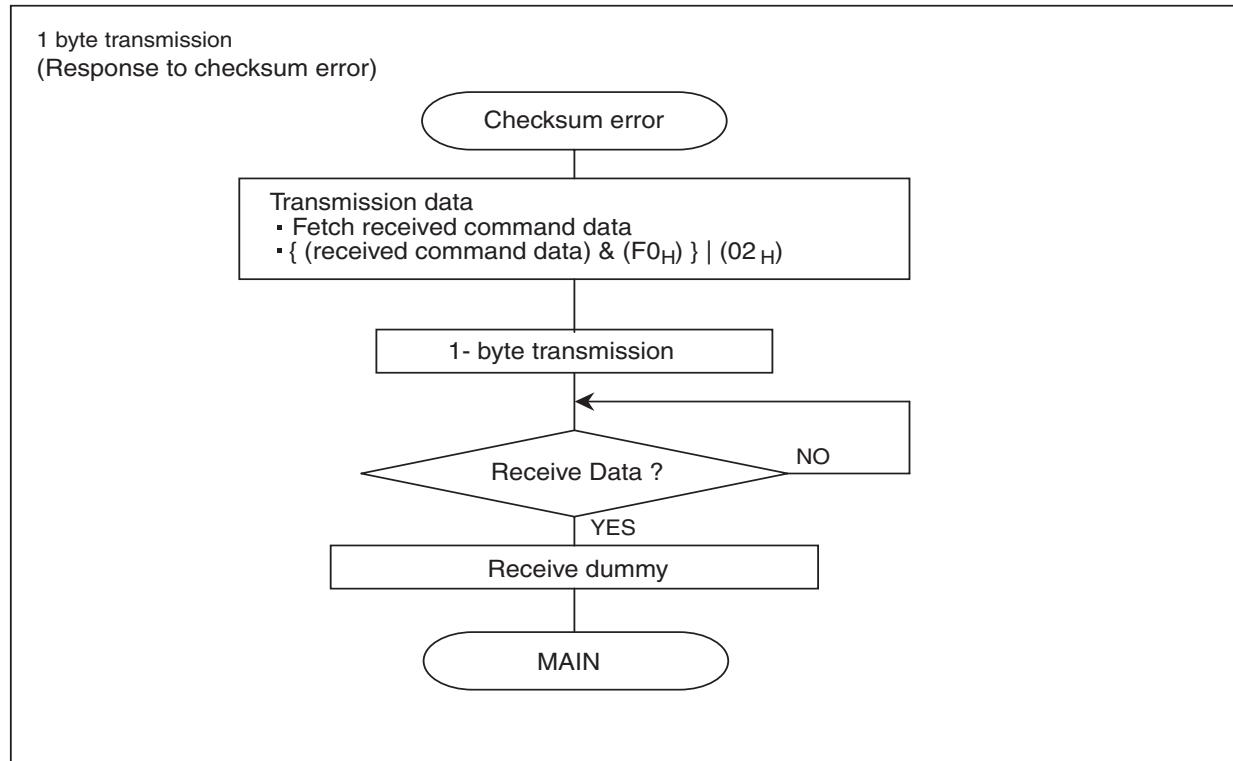
○ Subroutine "DOWN LOAD" in Synchronous mode



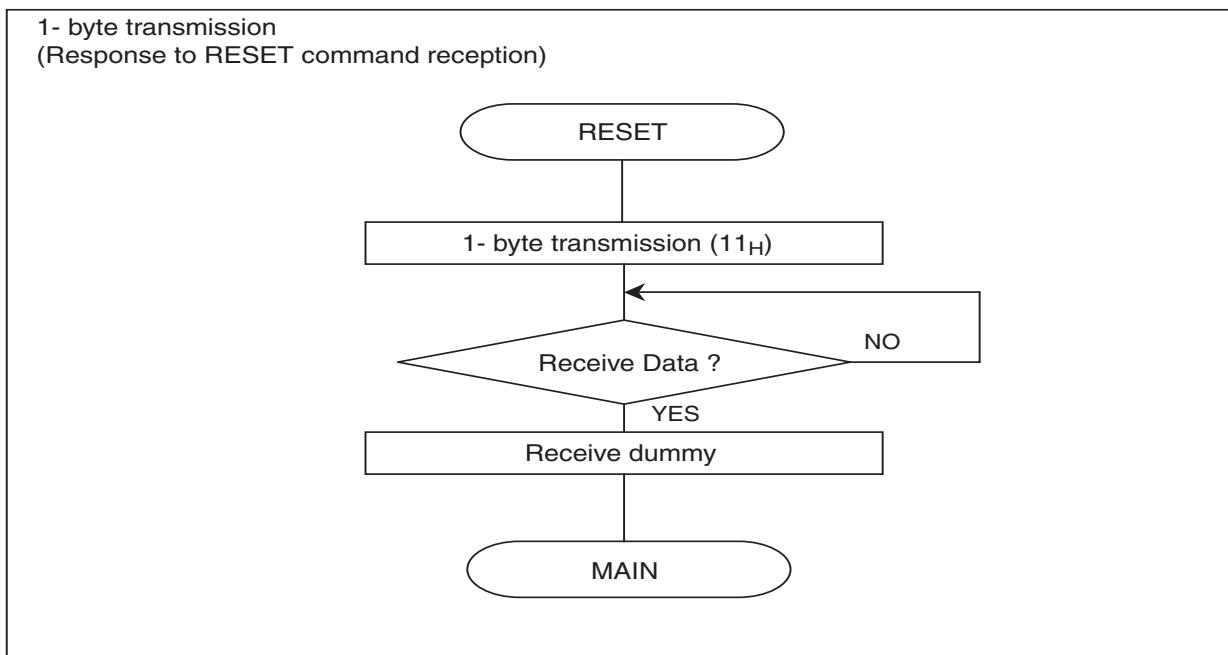
○ Subroutine "Command error" in Synchronous mode



○ Subroutine "Checksum error" in Synchronous mode



○ Subroutine "RESET" in Synchronous mode



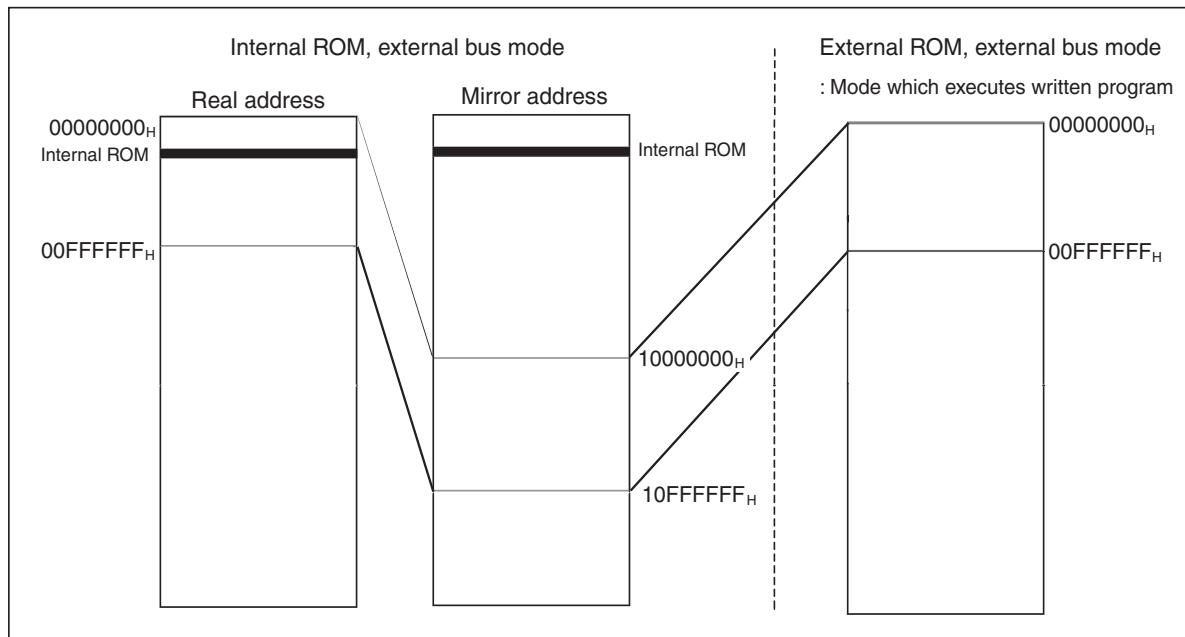
19.4 Example of Using the Program Loader Mode to Write to Flash Memory

This section describes an example of connection when flash memory connected to CS0 is located and written by using the program loader mode.

■ Allocation of Flash Memory

Flash memory connected to CS0 must be placed at arbitrary area between 00000000_H and $0FFFFFFF_H$. Program expanded to RAM by serial programmer writes to mirror address not real address. Since written program is executed only in external ROM, external bus mode, flash memory can be placed without considering internal ROM area.

Figure 19.4-1 Allocation of Flash Memory

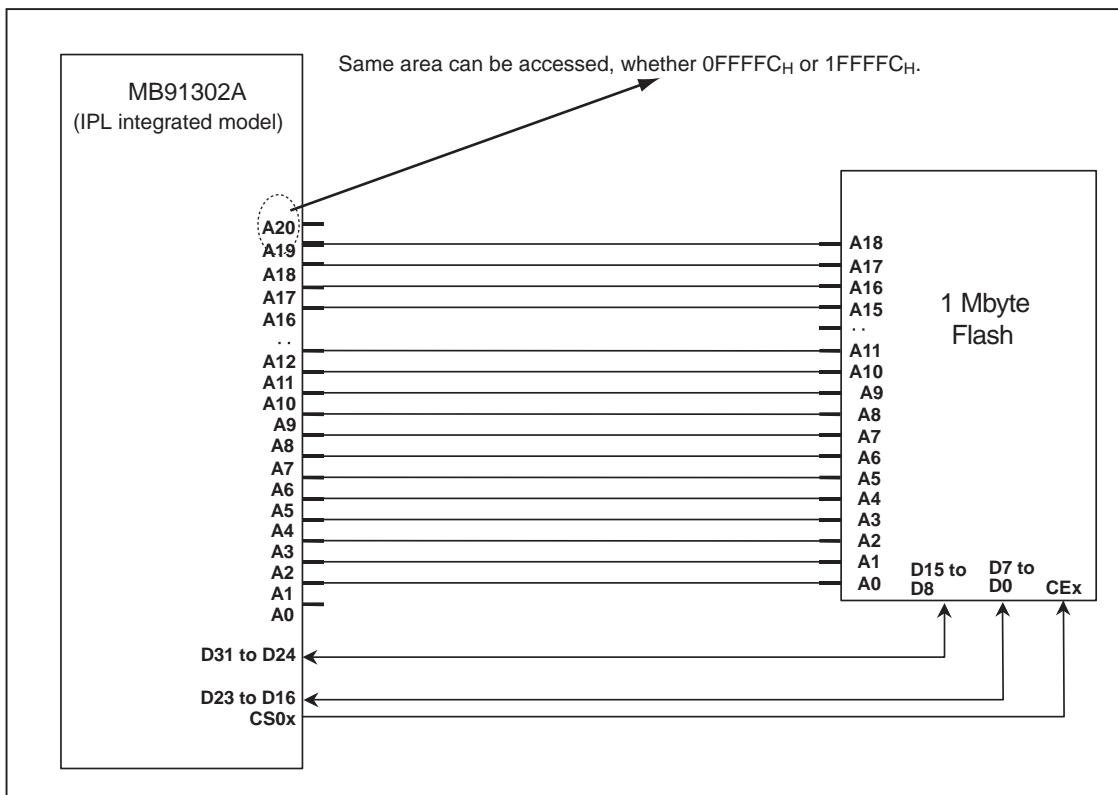


■ Examples of Connection for 1M Byte Flash

Flash memory must be located in an area not overlapping any internal area such as an internal resource or RAM before the entire flash memory can be accessed. This example assumes that one megabyte of flash memory connected to the CS0 area be accessed as "addresses 100000_H to $1FFFFFF_H$ ". Note that the FR family has the reset vector and mode vector fixed at addresses $0FFFFC_H$ and $0FFFF8_H$, respectively. That area must therefore be covered so that the program written to flash memory can be executed normally. When flash memory is 1 Mbyte, any address signal higher in order than A20 is not connected to the flash memory. It can therefore be solved by setting the CS0 address range to 000000_H to $1FFFFFF_H$ and accessing addresses 000000_H to $0FFFFFF_H$ and addresses 100000_H to $1FFFFFF_H$ as a mirror area.

Figure 19.4-2 shows memory access with offset addresses added (1 Mbyte).

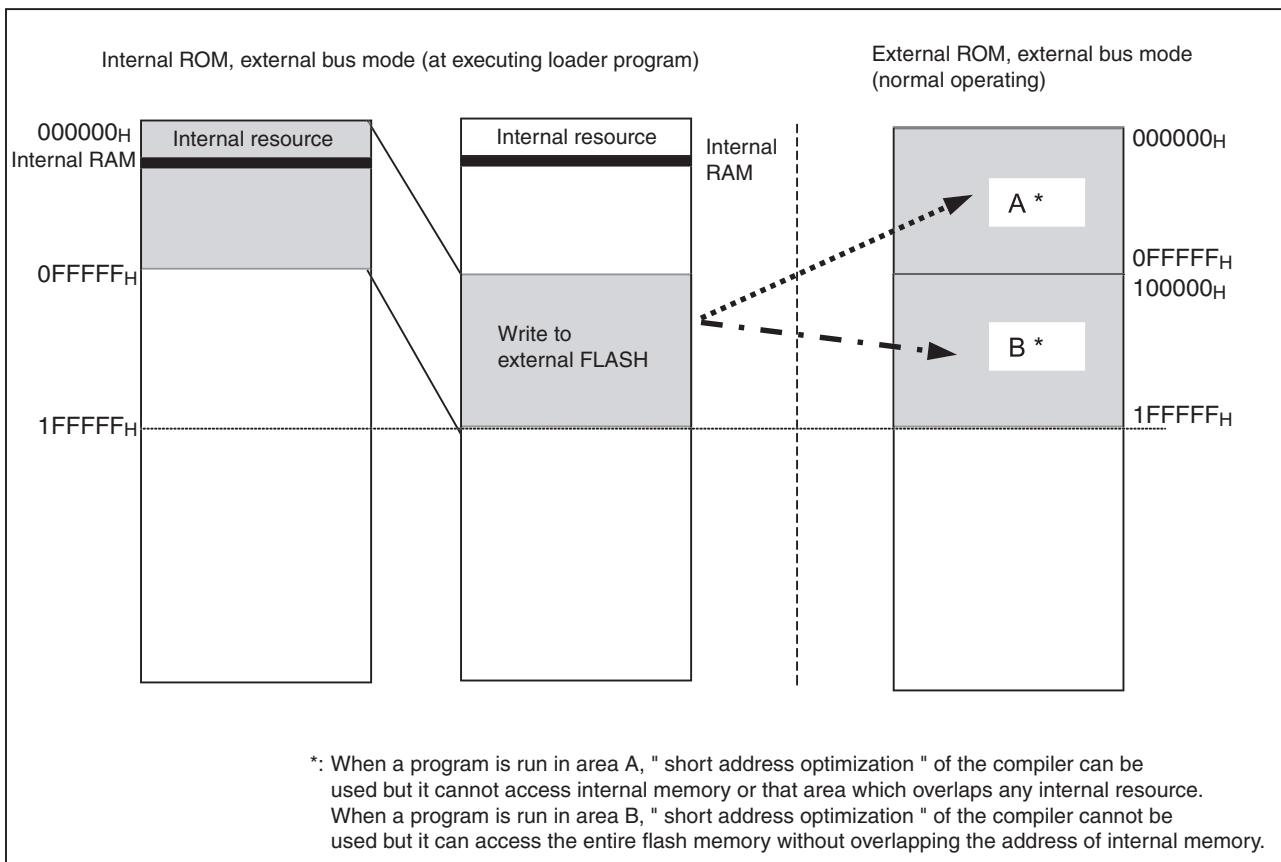
Figure 19.4-2 A Memory Access with Offset Addresses Added (1 Mbyte)



Note that, when a program written to flash memory is executed, the "external-vector activated, external-ROM/external-bus mode" is established. Therefore, the internal ROM area that the program loader stored in does not need any care.

Figure 19.4-3 shows a memory map for each mode.

Figure 19.4-3 Memory Map for Each Mode



CHAPTER 19 PROGRAM LOADER MODE (SUPPORTED ONLY BY THE MB91302A (IPL INTEGRATED MODEL))

CHAPTER 20 REAL-TIME OS EMBEDDED MB91302A-010 USER'S GUIDE

This chapter describes the features of the MB91302A-010 and its development methods.

- 20.1 Overview
- 20.2 Memory Map
- 20.3 Specifications for REALOS/FR Embedded in MB91302A-010
- 20.4 Section Allocation
- 20.5 Startup Routine
- 20.6 Initial Settings for SOFTUNE Workbench and REALOS/FR
- 20.7 Mode Pins, Mode Vectors, and Reset Vectors
- 20.8 Chip Evaluation System

20.1 Overview

This section describes the overview of MB91302A-010 and introduces the manuals.

■ Overview

The MB91302A-010 is a microcontroller fabricated by embedding μ ITRON 3.0^{*1} compliant SOFTUNE REALOS/FR^{*2} (here after called REALOS/FR) in the internal ROM of the MB91302A in the FR family of Fujitsu proprietary 32-bit RISC microcontrollers.

This chapter describes the features of the MB91302A-010 and its development methods. To develop programs for the MB91302A-010, use SOFTUNE Workbench and REALOS/FR bundled with the development kit MB91302A-RDK01. You should therefore refer to the manuals for related development tools in addition to this chapter.

■ Embedded REALOS/FR Version

SOFTUNE REALOS/FR Rev600001 (SOFTUNE REALOS/FR Kernel V30L08)

■ Related Manuals

FR FAMILY CONFORMING to μ ITRON3.0 SPECIFICATIONS SOFTUNE REALOS/FR USER'S GUIDE

FR FAMILY CONFORMING to μ ITRON3.0 SPECIFICATIONS SOFTUNE REALOS/FR KERNEL MANUAL

FR/F²MC FAMILY CONFORMING to μ ITRON SPECIFICATIONS SOFTUNE REALOS/FR/907/896 CONFIGURATOR MANUAL

FR-V/FR/F²MC FAMILY CONFORMING to μ ITRON SPECIFICATIONS SOFTUNE REALOS ANALYZER MANUAL

FR FAMILY ASSEMBLER MANUAL

SOFTUNE LINKAGE KIT MANUAL for V6

SOFTUNE Workbench OPERATION MANUAL for V6

SOFTUNE Workbench USER'S MANUAL

SOFTUNE Workbench COMMAND REFERENCE MANUAL for V6

*1: TRON is an abbreviation of "The Real-time Operating System Nucleus".

ITRON is an abbreviation of "Industrial TRON".

μ TRON is an abbreviation of "Micro Industrial TRON".

*2: SOFTUNE is a trademark of FUJITSU LIMITED.

REALOS is a trademark of FUJITSU LIMITED.

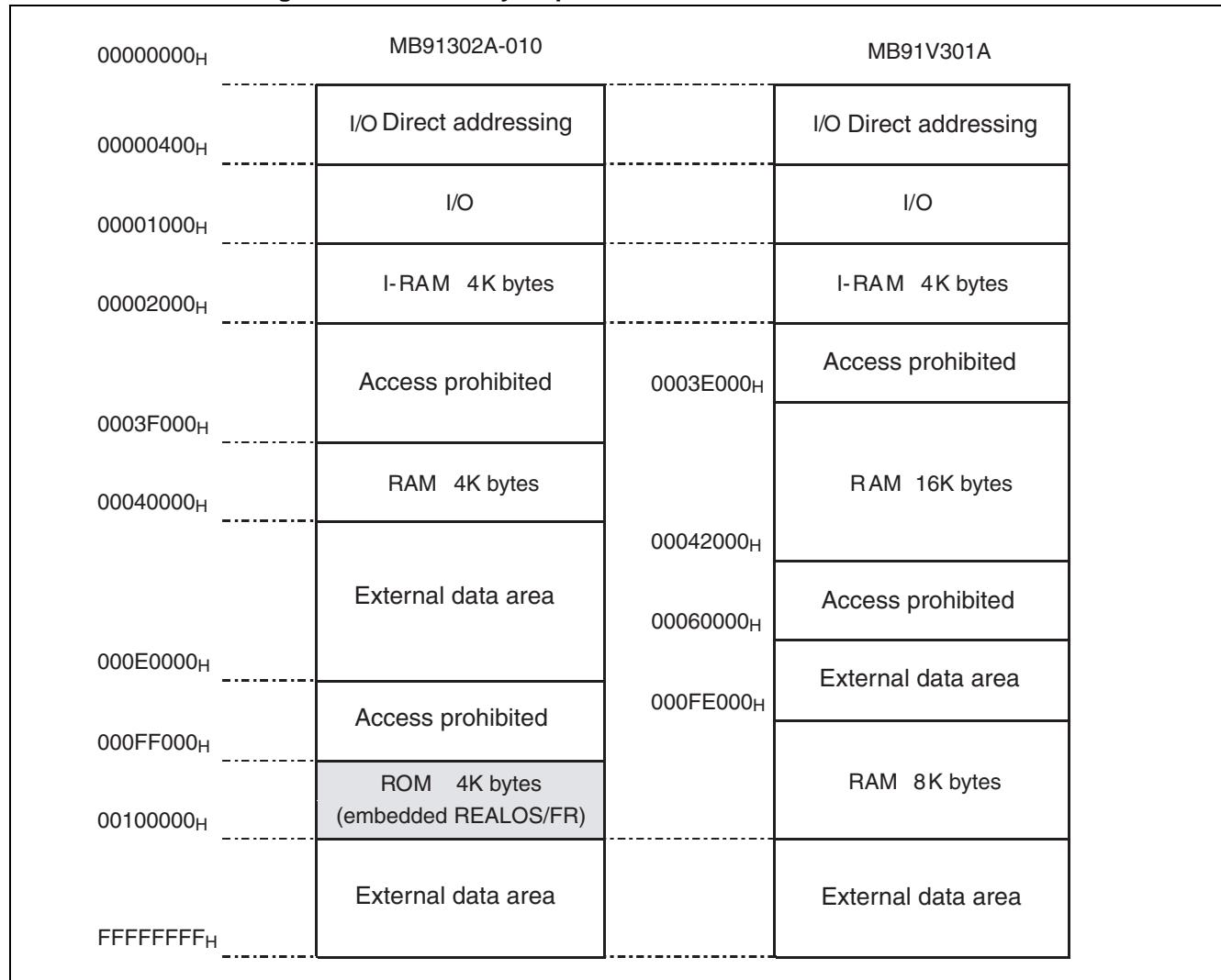
20.2 Memory Map

This section provides memory maps of the MB91302A-010 and its evaluation chip MB91V301A.

■ Memory Map

The following are memory maps for the MB91302A-010 and MB91V301A. The MB91302A-010 contains REALOS/FR conforming to µITRON 3.0 in the internal 4K byte ROM area located at addresses FF000_H to FFFF_H.

Figure 20.2-1 Memory Map of MB91302A-010 and MB91V301A



20.3 Specifications for REALOS/FR Embedded in MB91302A-010

The MB91302A-010 contains μ ITRON 3.0 compliant REALOS/FR in the internal 4 Kbytes ROM.

This section describes the system calls and objects supported by REALOS/FR embedded in internal 4 Kbytes ROM.

■ Outline of Embedded REALOS/FR

The size of the system stacks used by REALOS/FR is 64K bytes. Up to 32 cyclic handlers are supported. Up to 64 user tasks can be registered, for which priority levels from 1 to 32 can be assigned. Note that the alarm handler is not supported.

Table 20.3-1 Outline of Embedded REALOS/FR

System stack size	64K bytes
Alarm handler	not supported
Number of cyclic handlers	0 to 32
User task priority level	1 to 32
Number of user tasks	1 to 64
Number of semaphores	0 to 32
Number of event flags	0 to 32
Number of mailboxes	0 to 32

■ Contained System Calls

The MB91302A-010's internal ROM contains the following system calls. During actual development, register all of the following system calls using REALOS/FR's Configurator and be careful not to any other system call.

The evaluation chip MB91V301A on the target board is used for debugging.

For how to register system calls using REALOS/FR Configurator, refer to the SOFTUNE REALOS/FR Configurator Manual.

Table 20.3-2 System Calls

Function	System Call		
Task control	sta_tsk	ext_tsk	chg_pri
Synchronization with task	tslp_tsk	wup_tsk	
Synchronization / Communication	sig_sem	wai_sem	preq_sem
	set_flg	clr_flg	wai_flg
	snd_msg	rcv_msg	prcv_msg
Time control	def_cyc ret_tmr		
Interrupt control	ret_int		

■ Objects

The MB91302A-010's internal ROM contains the following objects.

The MB91302A-010 supports event flags, semaphores, and mailboxes.

Table 20.3-3 Objects

Objects Name	Number of Definitions
Event flag	0 to 32
Semaphore	0 to 32
Mailbox	0 to 32

○ Event flag

The MB91302A-010 supports up to 32 event flags.

The event flag definition of SOFTUNE REALOS/FR's Configurator is used to define event flags during program development. Even though the number of event flags to be actually used is less than 32, be sure to define 32 event flags including vacant definitions.

○ Semaphore

The MB91302A-010 supports up to 32 semaphores.

The semaphore definition of SOFTUNE REALOS/FR's Configurator is used to define semaphores during program development. Even though the number of semaphores to be actually used is less than 32, be sure to define 32 semaphores including vacant definitions.

○ Mailbox

The MB91302A-010 supports up to 32 mailboxes.

The mailbox definition of SOFTUNE REALOS/FR's Configurator is used to define semaphores during program development. Even though the number of mailboxes to be actually used is less than 32, be sure to define 32 semaphores including vacant definitions.

■ User Tasks

The MB91302A-010 supports up to 64 user tasks.

The task definition of SOFTUNE REALOS/FR's Configurator is used to define user tasks during program development.

Table 20.3-4 User Tasks

Number of user tasks	1 to 64
----------------------	---------

Even though the number of user tasks to be actually used is less than 64, be sure to use the task definition of SOFTUNE REALOS/FR's Configurator to define 64 tasks including empty tasks. The initial state of empty user tasks which are not actually used must be DORMANT. Even for an empty user task not to be used, write a vacant source code and compile it along with the other tasks.

In actual programs, be careful not to issue a system call to these unused tasks.

The following is an example of empty source code concerning unused task.

```
void boo (void) {}
```

20.4 Section Allocation

This section describes section allocation.

■ Section Allocation

The MB91302A-010 has the location address of the following section fixed. When developing an actual program, be sure to locate these sections at the following addresses. Sections are located through the project setting linker tab of SOFTUNE Workbench. sstack, knldata1, knldata2, DBGDAT2, mplmem, mplctl, and mpfmem are located in the RAM area; inidata, startcode, and R_eit are located in the ROM area.

The MB91302A-010 does not support memory-pool related system calls but requires that memory-pool related sections of mplmem, mplctl, and mpfmem be located.

Table 20.4-1 Sections at Fixed Location Addresses

Section Name	Location Address	Function / Size	Remarks
oscode	000FF000 _H	Real-time OS / 0xFE4	Internal ROM
sstack	10000000 _H	System stack / 0x10000	Allocated to external RAM
knldata1	10010000 _H	Real-time OS data / 0x1A68	
knldata2	10011F00 _H	Stack of idle tasks / 0x60	
DBGDAT2	10011FB0 _H	Debugging data / 0x4	
mplmen	10011FD0 _H	Data related to memory pools / 0x0	
mplctl	10011FE0 _H	Data related to memory pools / 0x0	
mpfmen	10011FF0 _H	Data related to memory pools / 0x0	
inidata	400FE000 _H	Real-time OS data / 0xDCC	Allocated to external ROM
startcode	400FE400 _H	Startup Routine / 0x544	
R_eit	400FFC00 _H	Vector entry/0x400	

20.5 Startup Routine

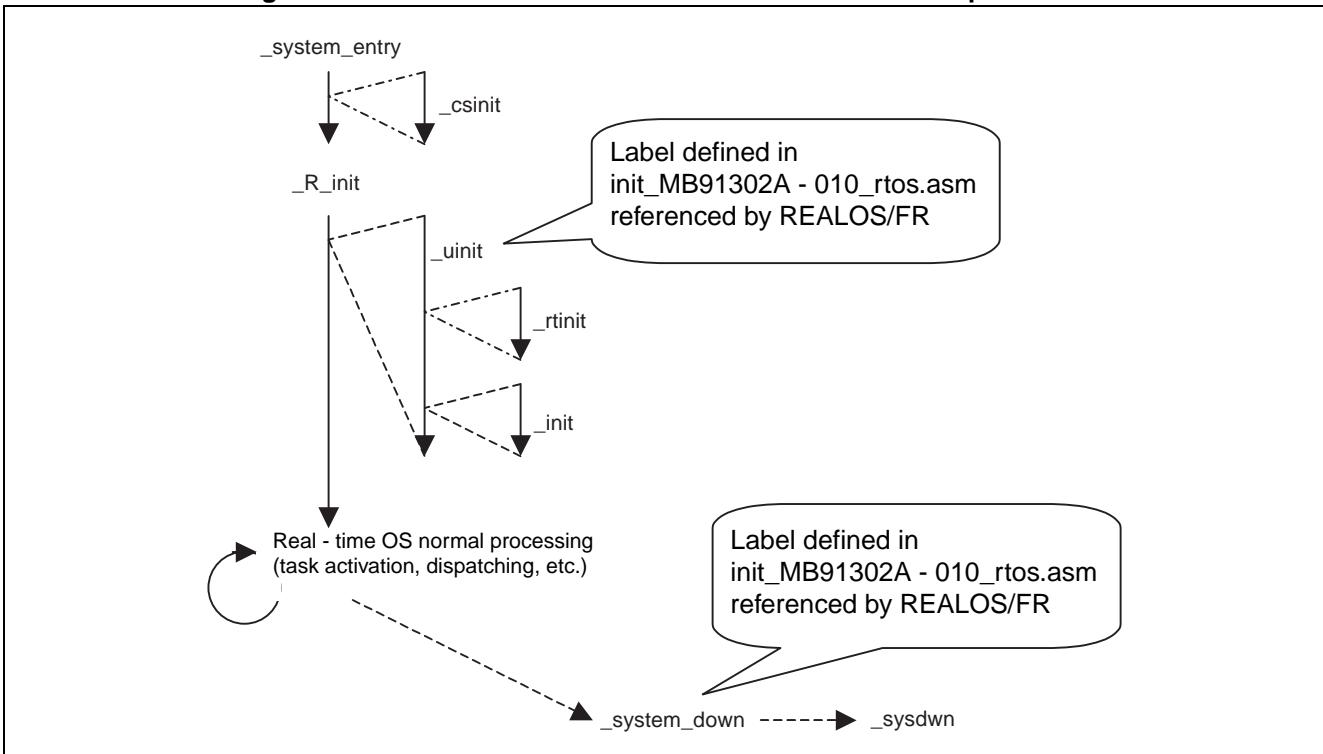
This section describes the startup routine.

■ Startup Routine

For the MB91302A-010, the startup routine `init_MB91302A-010_rtos.asm` is always located in the startcode section. REALOS/FR stored in internal ROM directly references the `_uinit` and `_system_down` labels defined in `init_MB91302A-010_rtos.asm`. Therefore, be sure to use `init_MB91302A-010_rtos.asm` provided by Fujitsu as the startup routine and do not modify the content. Updating the contents shifts the addresses of these labels referenced by REALOS/FR, preventing normal operation.

The following explains the process flow of the subroutine coded in `init_MB91302A-010_rtos.asm`. Of these, `_csinit` (bus setting/clock setting, etc.), `_rtinit` (reload timer setting), `_init` (other user initialization setting), and `_sysdwn` (routine used the system goes down) are subroutines called by the call instruction. These are created by the user to meet the system.

Figure 20.5-1 Flow for the Subroutine Coded in the Startup Routine



20.6 Initial Settings for SOFTUNE Workbench and REALOS/FR

SOFTUNE Workbench and REALOS/FR are used to develop programs.
This section describes initial settings for tools using practical examples.

■ Program Example

The following sample program is discussed here to explain tool initialization.

○ User Tasks

\$ Number of tasks	50
\$Stack size	0x1000 (per task)
\$ Initial situation	task ID1 to ID40 -> READY task ID41 to ID50 -> DORMANT
\$ Startup prioritized	task ID1 to ID30 -> 1 task ID31 to ID50 -> 2

○ Semaphore

\$ Number of semaphores	20
\$Semaphore count	semaphore ID1 to ID10-> maximum value 15; Initial value 0 semaphore ID11 to ID20-> maximum value 20; Initial value 0

○ Event flag

\$ Number of flags	32
\$ Initial pattern	semaphore ID1 to ID16-> 0x0000 semaphore ID17 to ID32-> 0xFFFF

○ Mailbox

Number of boxes	10
-----------------	----

○ Interrupt vector

\$ Use reload timer 0 for the system clock.	-> Timer handler name _timer
---	------------------------------

○ Memory

\$ Code ROM	0x40000000 - 0x403FFFFF (CS0 area x 16 bits)
\$ Work RAM	0x10000000 - 0x10FFFFFF

■ REALOS/FR Configurator Setup

When you create a new project for REALOS/FR using SOFTUNE Workbench, the project appears with a member of "project-name.rcf" (hereafter called rcf file) in the REALOS directory of the project window. When you double-click on this rcf file to start Configurator and perform the initial setting of REALOS/FR.

○ System definition tab

Define the number of entries of handlers to be used by the system. When defining each number of entries, be sure to use a fixed value shown below.

Table 20.6-1 Setup with the System Definition Tab

	Item	Set value	Remarks
1	Number of cyclic handlers (C)	D'32	Be sure to set "32" even though the number of actual cyclic handlers is less than "32".
2	Number of alarm handlers (L)	D'0	Be sure to set "0" as it is not supported.
3	Exception handler entry name (E)	Blank	Be sure to set blank as it is not supported.
4	Task priority level (P)	D'32	Be sure to set "32" even when the task priority level is smaller than "32".
5	Setting the include file	Blank	Blank

○ Memory definition tab

Set memory to be handled by the system.

Table 20.6-2 Setup with the Memory Definition Tab

	Item	Set value	Remarks
1	System stack size (S)	H'10000	Be sure to set 64K bytes.
2	Kernel code address (C)	Blank	Blank as it is set by the linker of SOFTUNE Workbench.
3	Kernel code address (D)	Blank	Blank as it is set by the linker of SOFTUNE Workbench.

○ System call definition tab

Register the system calls to be used. Register all of the following system calls available to the MB91302A-010. Never register any other system call.

Table 20.6-3 Setup with the System Call Definition Tab

	Item	Set value	Remarks
1	Registered system calls (S)	sta_tsk ext_tsk chg_pri tslp_tsk wup_tsk sig_sem wai_sem preq_sem set_flg clr_flg wai_flg pol_flg snd_msg rcv_msg prcv_msg ret_int def_cyc ret_tmr	Be sure to register all of these system calls and never register any other system call.

○ Task definition tab

Register user tasks. Start registration in order from ID number 1. Even though the number of user tasks for the actual system is less than 64, be sure to register 64 tasks. At this time, set the startup priority levels, stack, and initial state of unused user tasks using D'32 (minimum priority level), H'60 (minimum stack value acceptable), and DORMANT (idle state), respectively, not to start these empty user tasks.

Table 20.6-4 Setup with the Advanced Task Definition Window

Item	Set value	Remarks
1 Name	Free	Set freely
2 Entry (T)	Free	The user task name can be set freely. The entry of a user task with boo(){---} in C source code is _boo.
3 Startup priority level (P)	Free	Register unused user tasks with a priority level of 32.
4 Stack (S)	Free	H'60 is used to set unused user tasks.
5 Initial situation (A)	Free	Set unused user tasks first to DORMANT.
6 Start code (C)	Free	Set freely
7 ID number (I)	D'1 to D'64	Be sure to set 1 to 64 in ascending order.
8 Extensive information (O)	Free	Set freely
9 Time out (M)	Use	Set "Use"

Even for an unused user task, write a vacant source code in C source code and compile it along with other user tasks to be used for the system. The following is an example of empty source code concerning unused task.

```
void boo(void){}
```

This development example assumes the following with regard to user tasks.

\$ Number of tasks	50
\$Stack size	0x1000 (per task)
\$ Initial situation	task ID1 to ID40 -> READY task ID41 to ID50 -> DORMANT
\$ Startup prioritized	task ID1 to ID30 -> 1 task ID31 to ID50 -> 2

This development example requires the following settings with the task definition tab of Configurator.

task ID1 to ID30	Stack (S) -> H'1000 Initial situation (P) -> READY Startup prioritized (A) -> D'1
task ID31 to ID40	Stack (S) -> H'1000 Initial situation (P) -> READY Startup prioritized (A) -> D'2 ->
task ID41 to ID50	Stack (S) -> H'1000 Initial situation (P) -> DORMANT Startup prioritized (A) -> D'2 ->
task ID51 to ID64	Stack (S) -> H'60 (minimum stack) Initial situation (P) -> DORMANT (idle state) Startup prioritized (A) -> D'2 (minimum priority level)

} Define blank task

○ Semaphore Definition Tab

Register the semaphores to be used by the system. Register all of ID numbers 1 through 32 in ascending order.

Even though the number of semaphores to be actually used is less than 32, be sure to register 32 semaphores.

Table 20.6-5 Setup with the Semaphore Definition Tab

	Item	Set value	Remarks
1	Name (E)	Free	Set freely
2	Initial count (C)	Free	Set freely
3	Maximum count (M)	Free	Set freely
4	ID number (D)	D'1 to 32	Be sure to set D'1 through D'32 in ascending order.
5	Extensive information (O)	Free	Set freely

This development example assumes the following with regard to semaphores.

\$ Number of semaphores	20
\$ Semaphore count	Semaphore ID1 to ID10 -> Max. 15 / Initial value 0
	SemaphoreID11 to ID20 -> Max. 20 / Initial value 0

This development example requires the following settings with the semaphore definition tab of Configurator.

Semaphore ID1 to ID10	Initial count (C)	->	D'0
	Maximum count (M)	->	D'15
Semaphore ID11 to ID20	Initial count (C)	->	D'0
	Maximum count (M)	->	D'20
Semaphore ID21 to ID32	Initial count (C)	->	D'0
	Maximum count (M)	->	D'8

} Define blank task

○ Event Flag Definition Tab

Register the event flags to be used by the system. Register all of ID numbers 1 through 32 in ascending order.

Even though the number of event flags to be actually used is less than 32, be sure to register 32 event flags.

Table 20.6-6 Setup with the Event Flag Definition Tab

	Item	Set value	Remarks
1	Name (E)	Free	Set freely
2	Initial Pattern (P)	Free	Set freely
3	ID number (D)	D'1 to 32	Be sure to set D'1 through D'32 in ascending order.
4	Extensive information (O)	Free	Set freely

This development example assumes the following with regard to event flags.

\$ Number of event flags	32
\$ Initial Pattern	Flag ID1 to ID16 -> 0x0000
	Flag ID17 to ID32 -> 0xFFFF

For this development example, the event flag definition tab of Configurator appears as shown below.

Flag ID1 to ID16	Initial Pattern (P)	-> H'0000
Flag ID17 to ID32	Initial Pattern (P)	-> H'FFFF

○ Mailbox definition tab

Register the mailboxes to be used by the system. Register all of ID numbers 1 through 32 in ascending order.

Even though the number of mailboxes to be actually used is less than 32, be sure to register 32 mailboxes.

Table 20.6-7 Setup with the Mailbox Definition Tab

	Item	Set value	Remarks
1	Name (E)	Free	Set freely
2	ID number (D)	D'1 to 32	Be sure to set D'1 through D'32 in ascending order.
3	Extensive information (O)	Free	Set freely

○ Variable-length memory pool and fixed-length memory pool definition tabs

REALOS/FR embedded in internal ROM of the MB91302A-010 does not support these variable-length or fixed-length memory pools. Leave these definition tabs blank without making any setting.

○ Vector definition tab

Register interrupt handlers. To generate a system clock signal using one of internal reload timers 0 to 2, register the interrupt handler for the reload timer to any of interrupt numbers D'24 to D'26. D'1 registers the mode vector. For the MB91302A-010, either 06000000_H (external ROM area 32-bit mode with the MB91302A-010's internal ROM enabled), 05000000_H (external ROM area 16-bit mode with the MB91302A-010's internal ROM enabled) or 04000000_H (external ROM area 8-bit mode with the MB91302A-010's internal ROM enabled) is used for the mode vector value according to the hardware specifications of the target. For details on the mode vector, refer to the hardware manual for the MB91302A-010.

Table 20.6-8 Setup with Vector Definition Tab

Number (M)	Entry (E)	Remarks
D'0	_system_entry	Fix reset vector name to the left
D'1	06000000 _H or 05000000 _H or 04000000 _H	Register one of the three mode vectors to the left, that matches the hardware specifications of the target.
D'2 to D'23	Free	Set freely
D'24 to D'26	Timer handler name	To generate a system clock signal using one of internal reload timers 0 to 2, register the timer handler for the reload timer to any of interrupt numbers D'24 to D'26.
D'27 to D'255	Free	Set freely

This sample program uses reload timer 1 to generate a system clock signal. As the timer handler name is _timer, D'25 (interrupt number for reload timer 1) is set to _timer.

As CS0's hardware consists of x16-bit ROM, the mode vector is 05000000_H. Therefore, set D'1 (mode vector) to H'0500000.

○ Debug setting tab

The MB91302A-010 has no unique setting. (Default setting)

Note:

At the default setting, the REALOS analyzer log function is restricted.

For more information, refer to the manual for FR-V/FR/F²MC FAMILY CONFORMING TO μITRON SPECIFICATIONS SOFTUNE REALOS ANALYZER MANUAL.

○ Summary of Configurator setup

Table 20.6-9 lists those items under individual definition tabs of Configurator which must be set to a fixed value each.

Table 20.6-9 List of Fixed Values in Configurator

Item	Item Description	Fixed set value
System definition	Number of cyclic handlers	Number of entries fixed at 32
	Number of alarm handlers	D'0
	Exception handler entry name	Blank
	Task priority level	Fixed at D'32
	Setting the include file	Blank
Memory definition	System stack size	Fixed at 64K bytes
	Kernel code address	Blank
	Kernel data address	Blank
System call definition		Refer to Table 20.6-3
Task definition	Number of entries	Number of entries fixed at 64 (including empty tasks)
	Name	Set freely
	Entry	Set freely
	Startup priority level	Set freely
	Stack	Set freely
	Initial situation	Set freely
	Start code	Set freely
	ID number	Be sure to set 1 to 64 in ascending order.
	Extensive information	Set freely
	Time out	Time-out allotted, fixed
	Common stack	Set freely
Semaphore definition	Number of entries	Number of entries fixed at 32 (including empty semaphores)
	Name	Set freely
	Initial count	Set freely
	Maximum count	Set freely
	ID number	Be sure to set D'1 through D'32 in ascending order.
	Extensive information	Set freely
Event flag definition	Number of entries	Number of entries fixed at 32 (including empty event flags)
	Name	Set freely
	Initial Pattern	Set freely
	ID number	Be sure to set D'1 through D'32 in ascending order.
	Extensive information	Set freely
Mailbox definition	Number of entries	Number of entries fixed at 32 (including empty mailboxes)
	Name	Set freely
	ID number	Be sure to set D'1 through D'32 in ascending order.
	Extensive information	Set freely
Variable-length memory pool		Not supported
Fixed-length memory pool		Not supported
Vector definition	Reset vector name	Fix reset vector name to _system_entry
	Number	Set D'1 through D'255 in ascending order.
	Entry	Set freely
Debug setting		Fix default setting

■ Program Allocation

The following describes program location.

○ Section

At least the following sections exist by default, including REALOS/FR.

Table 20.6-10 Section

Section name	Size (byte)	Contents	Allocation top address	Memory
DATA	-	Data in C source code	Allocate freely	RAM
INIT	-	Initial data in C source code	Allocate freely	RAM
oscode	FE4H	Real time OS	Fixed at 000FF000H	Internal ROM
sstack	10000H	System stack	Fixed at 10000000H	RAM
knldata1	1A68H	Data of real time OS	Fixed at 10010000H	RAM
knldata2	60H	Stack of idle tasks	Fixed at 10011F00H	RAM
DBGDAT2	4H	Debug related sections	Fixed at 10011FB0H	RAM
mplmem	0H	Memory pool related sections	Fixed at 10011FD0H	RAM
mplctl	0H	Memory pool related sections	Fixed at 10011FE0H	RAM
mpfmem	0H	Memory pool related sections	Fixed at 10011FF0H	RAM
R_stk001 to 040	-	User stack	Allocate freely	RAM
inidata	DCCH	Initial data of real time OS	Fixed at 400FE000H	ROM
startcode	544H	Start up	Fixed at 400FF400H	ROM
CODE	-	C source code	Allocate freely	ROM
@INIT	-	Initial data of C source code	Allocate freely	ROM
CONST	-	Initial data of C source code	Allocate freely	ROM
uinitcode *	-	_csinit _init _rtinit _sysdwn	Allocate freely	ROM
R_eit	400H	Interrupt vector	Fixed at 400FFC00H	ROM

* : Section name uinitcode is the section name of _csinit _init _rtinit _sysdwn.

This can be freely named with the .section pseudo-instruction.

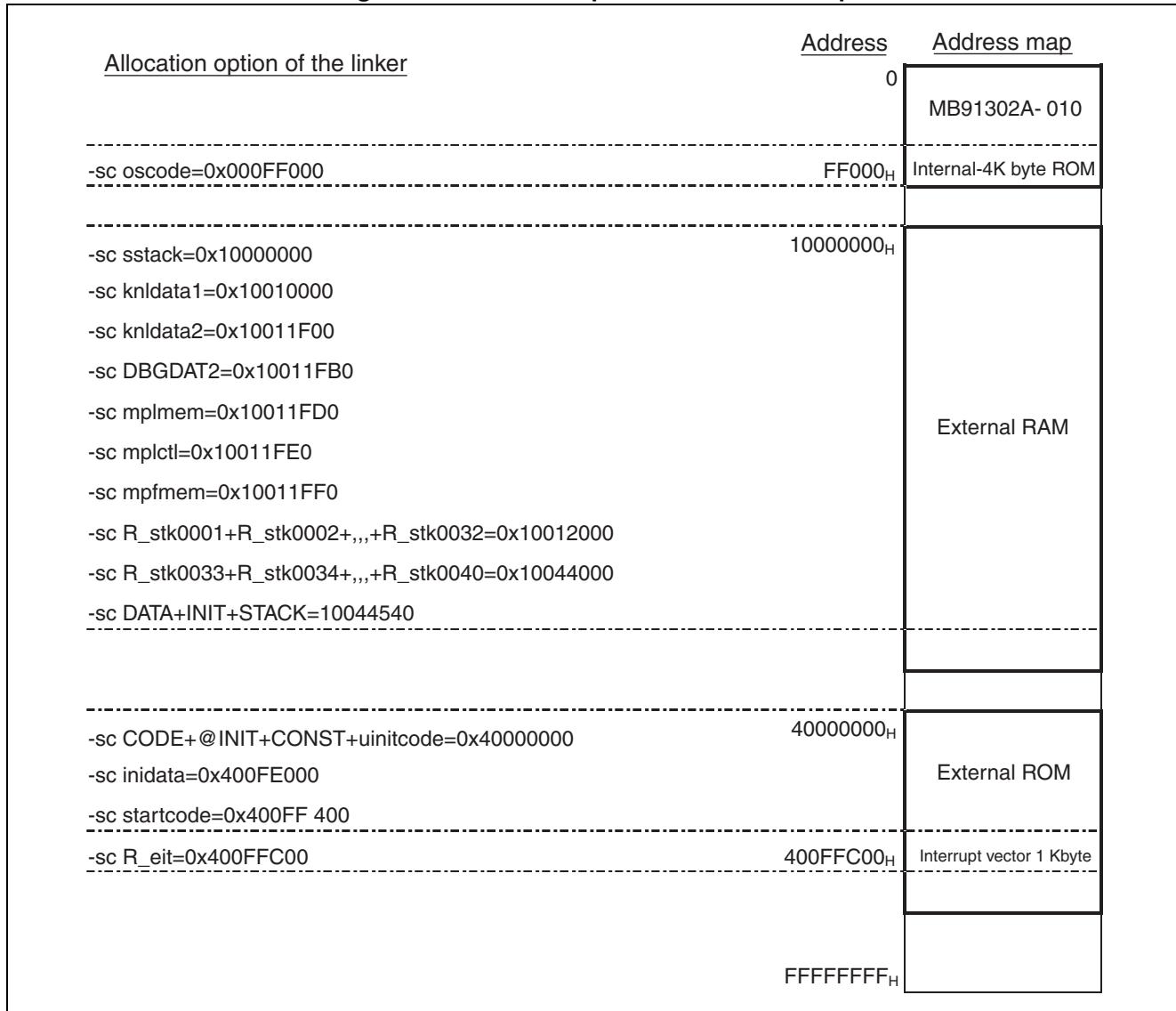
○ Linker Allocation Option

Given below are linker options and an address map for "Program Example" in Section "20.6 Initial Settings for SOFTUNE Workbench and REALOS/FR". For details on the options for the linker, refer to the manual for SOFTUNE Workbench.

Stacks (10000H bytes each) of 50 user tasks are located at R_stk0001 to R_stk0032 and 14 empty tasks are allocated at 60H-byte user stack each between R_stk0033 and R_stk0040.

This example assumes that "CODE + @INIT + CONST" can be stored between 40000000H and 400FDFFFH.

Figure 20.6-1 Linker Option and Address Map



20.7 Mode Pins, Mode Vectors, and Reset Vectors

This section describes the mode pins, mode vectors, and reset vectors of the MB91302A-010. For more information, see Sections "3.11.3 Reset Sequence" and "3.14 Operating Modes" as well.

■ Mode Pin

The MB91302A-010 can fetch the mode vector that determines the operation mode of the microcontroller from address 000FFFF8_H and the reset vector (startup routine's start address) from address 000FFFC_H after initialization by a reset.

Since addresses 000FFFF8_H and 000FFFC_H are located in the internal ROM containing REALOS/FR, however, the mode and reset vectors fetched after the release by a reset are taken from external ROM to the microcontroller. Therefore, set the mode pin on the MB91302A-010 to the external vector fetch mode.

Table 20.7-1 Setting of Mode Pins

Mode pin	Setting value
MD2 to MD0	"LLH" fixed (external vector fetch mode)

■ Mode Vector

The MB91302A-010 fetches the mode vector that determines the operation mode of the microcontroller from address 400FFFF8_H after being released from a reset.(The address actually output by the MB91302A-010 is 000FFFF8_H.)

For the MB91302A-010, set the internal-ROM/external-bus mode to enable the internal ROM containing REALOS/FR.(Mode data for internal ROM/external bus mode: 0B000001_H)

The mode vector fetched after a reset is canceled has the bit for setting the internal-ROM/external-bus mode and the bit for setting the bit width of the external CS0 area as well. Set the bit width of this CS0 area to the bit width of ROM mounted on the target board.

The mode vector can be set with handler number D'1 by using the vector definition tab of REALOS/FR's Configurator.

Table 20.7-2 Setting Value of Mode Vector

Target CS0's ROM bit width	Set value (hex)	Setting value for the vector definition tab
8 bits	04	04000000 _H
16 bits	05	05000000 _H
32 bits	06	06000000 _H

■ Reset Vectors

The MB91302A-010 fetches the reset vector from address 000FFFC_H after being released from a reset.

The reset vector is embedded in code automatically by the linker of SOFTUNE Workbench.

■ Mode Data and Reset Vector Location Addresses

The MB91302A-010 interrupt vector is sized 1 Kbyte and located in the R_eit section.

Use the linker of SOFTUNE Workbench to locate the R_eit section at addresses $400FFC00_H$ to $400FFFFF_H$. At this time, both of the mode vector and reset vector are located within R_eit by the linker.

Table 20.7-3 Mode Vector and Reset Vector Location Addresses

Vector name	Location addresses (hex)
Mode vector	$400FFFF8_H$
Reset Vector	$400FFFC_H$
Note R_eit	$400FFC00$ to $400FFFFF_H$

■ Mode Data and Reset Vector Access Addresses

The MB91302A-010 allows linear access to 32-bit address space but the address signals actually available are A23 to A00, where CS signals substitute for A31 to A24 to decode these addresses. The whole address space is used as CS0 until each CS area is allocated by the startup routine after initialization by a reset.

By setting addresses $400FFFF8_H$ and $400FFFC_H$ as CS0, therefore, the MB91302A-010 can fetch mode data and a reset vector located at addresses $400FFFF8_H$ and $400FFFC_H$ as data at addresses $000FFFF8_H$ and $000FFFC_H$ after initialization by a reset.

Table 20.7-4 Mode Data and Reset Vector Access Addresses

	After a reset is canceled	At CS area is set by register
Mode data	$000FFFF8_H$	$400FFFF8_H$
Reset vector	$000FFFC_H$	$400FFFC_H$

■ Fetching Mode Data and Reset Vectors after the Device is Released from a Reset

The MB91302A-010 fetches the mode data and reset vector from addresses $000FFFF8_H$ and $000FFFC_H$ after being released from the reset. Depending on the next operation of the MB91302A-010, these items of data located at addresses $400FFFF8_H$ and $400FFFC_H$ in the R_eit section are fetched when a reset is canceled.

Table 20.7-5 Sequence after a Release from a Reset

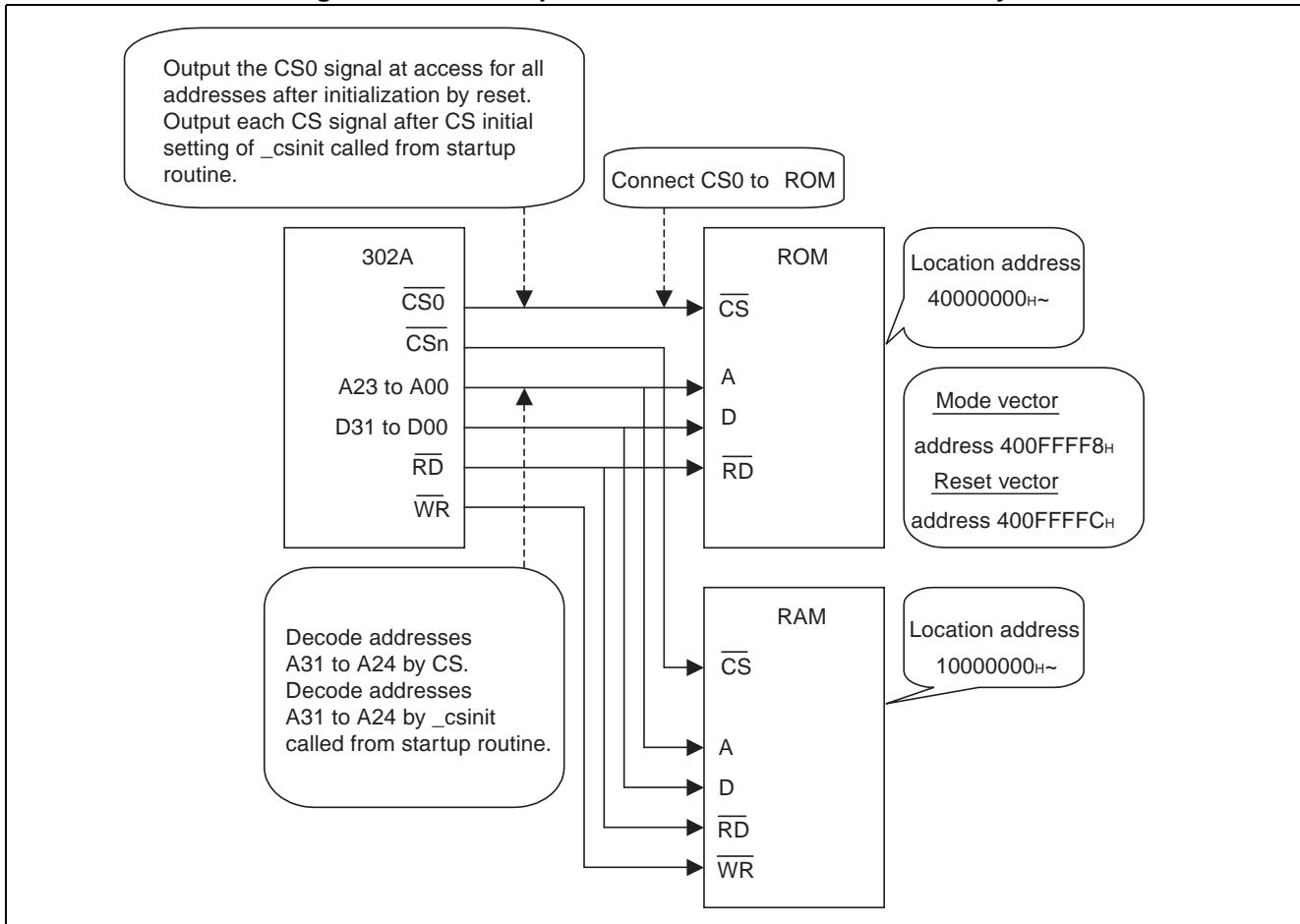
Operation	Remarks
1 Release from a Reset The entire 32-bit address space becomes CS0 upon initialization by a reset.	The CS0 signal is asserted upon any external access.
2 The MD2 to MD0 mode pins become "LLH" which is fixed on the target board after a reset is canceled. Fetch a mode vector from external address $000FFFF8_H$ at the timing 1). CS0 is asserted at this time.	For the MB91302A-010, set the MD2 to MD0 mode pins to "LLH". (External vector mode) For the mode vector, set 04_H , 05_H , or 06_H depending on the CS0 bus width.
3 The lower two bits of the mode vector is used to set the bit width of external CS0 area ROM. According to this, the reset vector is fetched from external address $000FFFC_H$. CS0 is asserted at this time.	Set CS0 to the external ROM that stores the R_eit area (addresses $400FFC00_H$ to $400FFFF_H$).
4 Load the reset vector fetched at 3) to the internal PC.	
5 Internal 4K byte ROM is enabled according to the mode vector fetched at 2) after completion of 4).	
6 The program starts running. Each CS area is according to the bus width/area set by _csinit called from the startup routine.	All address areas are accessed as CS0 until bus setting by _csinit. The startup routine must therefore be located always in CS0.

■ An Example of Connection of External Memory

Given below is an example of connection of the MB91302A-010 to external memory. External memory on the target must include ROM (addresses $400FE000_H$ to $400FFFF_H$) are mandatory as this area is referenced by the real-time OS for storing code and RAM (starting at address 10000000_H) for storing work data. Be sure to set ROM to CS0.

The MB91302A-010 "outputs the CS0 signal in response to access to any address after initialization by a reset" and "has the address upper bits (A31 to A24) decoded by CS signals". When released from the reset, therefore, the MB91302A-010 can fetch mode vector (at address $000FFFF8_H$) and reset vector (at address $000FFFC_H$) from actual addresses $400FFFF8_H$ and $400FFFC_H$ of external ROM.

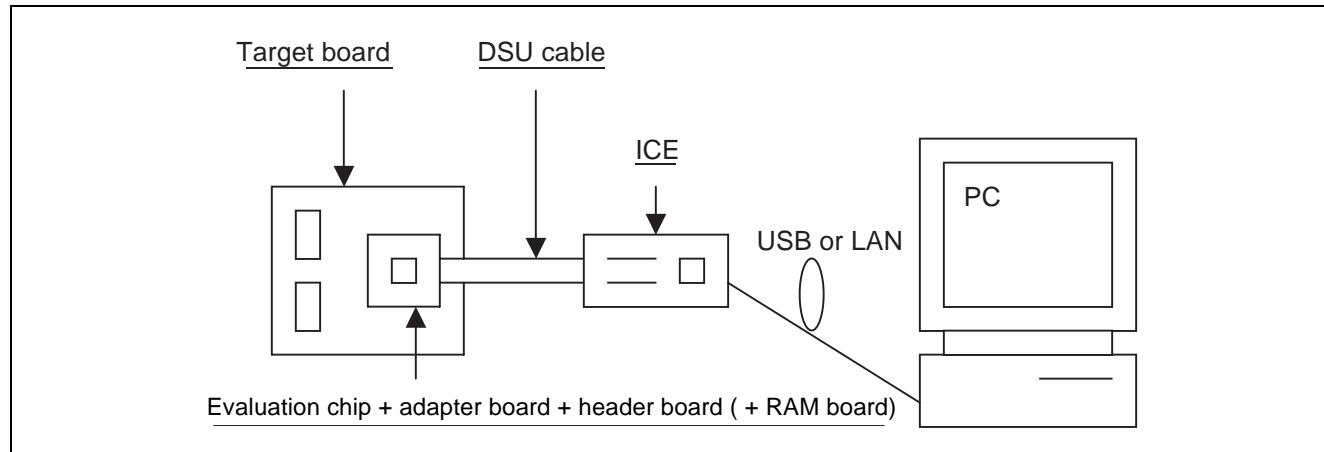
Figure 20.7-1 Example of Connection to External Memory



20.8 Chip Evaluation System

This section describes a sample configuration of the chip evaluation system.

■ Configuration Example :Target Board + Evaluation Chip + ICE



Name	Type	Remarks
ICE	MB2198-01	Connect with PC, USB, or LAN.
DSU cable	MB2198-10	Cable connecting the ICE with the adapter board
Evaluation chip	MB91V301A	Bundled with the development kit MB91V301A-RDK01.
Adapter board	MB2198-100	Used together with MB2198-101
Header board	MB2198-101	Used together with MB2198-100 Belongings: NQPACK144SE and HQPACK144SE
RAM board	MB2198-90	Not required if the target board has emulation memory.

In addition, the following development tools (software and the development kit) are required for development.

- FR family SOFTUNE professional pack (V6 supported) => Integrated development environment (Workbench, compiler, etc.)
- MB91V301A-RDK01 => Evaluation chip bundled with development software

APPENDIX

This appendix consists of the following parts: I/O map, interrupt vector, pin states in each CPU state, notes on using a little endian area, and instruction lists. The appendix contains detailed information that could not be included in the main text and reference material for programming.

APPENDIX A I/O MAP

APPENDIX B INTERRUPT VECTOR

APPENDIX C PIN STATE IN EACH CPU STATE

APPENDIX D NOTES ON USING A LITTLE ENDIAN AREA

APPENDIX E INSTRUCTION LISTS

APPENDIX A I/O MAP

Table A-1 shows the correspondence between the memory space area and the peripheral resource registers.

■ I/O Map

[Reading the table]

Address	Register				Block
	+0	+1	+2	+3	
000000H	PDR0[R/W] XXXXXXX	PDR1[R/W] XXXXXXX	PDR2[R/W] XXXXXXX	PDR3[R/W] XXXXXXX	T-unit Port Data Register

Note:

The initial values of bits in a register are indicated as follows:

- 1: Initial value "1"
 - 0: Initial value "0"
 - X: Initial value "X"
 - : A physical register does not exist at the location.
-

Table A-1 I/O Map (1 / 11)

Address	Register				Block
	+0	+1	+2	+3	
000000 _H	PDR0 [R/W] B XXXXXXXXXX	PDR1 [R/W] B XXXXXXXXXX	PDR2 [R/W] B XXXXXXXXXX	—	T-unit Port Data Register
000004 _H	—	—	PDR6 [R/W] B XXXXXXXXXX	—	
000008 _H	PDR8 [R/W] B XXXXXXXXXX	PDR9 [R/W] B -XXXXXXXX	PDRA [R/W] B XXXXXXXXXX	PDRB [R/W] B XXXXXXXXXX	
00000C _H	—				
000010 _H	PDRG [R/W] B XXXXXXXXXX	PDRH [R/W] B -----XXX	—	PDRJ [R/W] B XXXXXXXXXX	R-bus Port Data Register
000014 _H to 00003C _H	—				Reserved
000040 _H	EIRR [R/W] B,H,W 00000000	ENIR [R/W] B,H,W 00000000	ELVR [R/W] B,H,W 00000000 00000000		Ext int
000044 _H	DICR [R/W] B,H,W -----0	HRCL [R/W] B,H,W 0-11111	—		DLYI/I-unit
000048 _H	TMRLR0 [W] H,W XXXXXXXXX XXXXXXXXX		TMR0 [R] H,W XXXXXXXXX XXXXXXXXX		Reload Timer 0
00004C _H	—		TMCSR0 [R/W] B,H,W --XX0000 00000000		
000050 _H	TMRLR1 [W] H,W XXXXXXXXX XXXXXXXXX		TMR1 [R] H,W XXXXXXXXX XXXXXXXXX		Reload Timer 1
000054 _H	—		TMCSR1 [R/W] B,H,W --XX0000 00000000		
000058 _H	TMRLR2 [W] H,W XXXXXXXXX XXXXXXXXX		TMR2 [R] H,W XXXXXXXXX XXXXXXXXX		Reload Timer 2
00005C _H	—		TMCSR2 [R/W] B,H,W --XX0000 00000000		
000060 _H	SSR0 [R, R/W] B,H,W 00001000	SIDR0 [R] SODR0 [W] B, H,W XXXXXXXXX	SCR0 [R/W] B,H,W 00000100	SMR0 [R/W] B,H,W 00--0-0-	UART0
000064 _H	UTIMO [R] H,W (UTIMR0 [W] H,W) 00000000 00000000		DRCL0* ⁴ -----	UTIMCO [R/W] B 0--00001	U-TIMER 0

APPENDIX A I/O MAP

Table A-1 I/O Map (2 / 11)

Address	Register				Block
	+0	+1	+2	+3	
000068 _H	SSR1 [R/W] B,H,W 00001000	SIDR1 [R] SODR1 [W] B,H,W XXXXXXXX	SCR1 [R/W] B,H,W 00000100	SMR1 [R/W] B,H,W 00-0-0-	UART1
00006C _H	UTIM1 [R] H,W (UTIMR1 [W] H,W) 00000000 00000000	DRCL1* ⁴ -----	UTIMC1 [R/W] B 0--00001	U-TIMER 1	
000070 _H	SSR2 [R/W] B,H,W 00001000	SIDR2 [R] SODR2 [W] B,H,W XXXXXXXX	SCR2 [R/W] B,H,W 00000100	SMR2 [R/W] B,H,W 00-0-0-	UART2
000074 _H	UTIM2 [R] H,W (UTIMR2 [W] H,W) 00000000 00000000	DRCL2* ⁴ -----	UTIMC2 [R/W] B 0--00001	U-TIMER 2	
000078 _H	ADCR [R] B,H,W 000000XX XXXXXXXX	ADCS [R, R/W] B,H,W 00000000 00000000			A/D Converter sequential comparison
00007C _H	ADCR0 [R] B,H,W XXXXXXXXXX	ADCR1 [R] B,H,W XXXXXXXXXX	ADCR2 [R] B,H,W XXXXXXXXXX	ADCR3 [R] B,H,W XXXXXXXXXX	
000080 _H to 000090 _H	—	—	—	—	Reserved
000094 _H	IBCR0 [R/W] B,W,H 00000000	IBSR0 [R] B,W,H 00000000	ITBA0 [R/W] B,W,H 00000000 00000000	—	I ² C interface 0
000098 _H	ITMK0 [R/W] B,W,H 00111111 11111111	ISMK0 [R/W] B,W,H 01111111	ISBA0 [R/W] B,W,H 00000000	—	
00009C _H	—	IDAR0 [R/W] B,H,W 00000000	ICCR0 [R/W] B,W,H 00011111	IDBL0 [R/W] B,W,H 00000000	
0000A0 _H	—	—	—	—	Reserved
0000A4 _H	—	—	—	—	
0000A8 _H to 0000B0 _H	—	—	—	—	Reserved

Table A-1 I/O Map (3 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
0000B4 _H	IBCR1 [R/W] B,H,W 00000000	IBSR1 [R] B,H,W 00000000	ITBA1 [R/W] B,H,W 00000000 00000000		I ² C interface1	
0000B8 _H	ITMK1 [R/W] B,H,W 00111111 11111111		ISMK1 [R/W] B,H,W 00011111	ISBA1 [R,R/W] B,H,W 00000000		
0000BC _H	–	IDAR1 [R/W] B,H,W 00000000	ICCR1 [R/W] B,H,W 00011111	IDBL1 [R/W] B,H,W 00000000		
0000C0 _H	–	–	–	–		
0000C4 _H	–	–	–	–	Reserved	
0000C8 _H to 0000D0 _H	–	–	–	–		
0000D4 _H	TCDT [R/W] H,W 00000000 00000000		–	TCCS [R/W] B,H,W 00000000	16-bit free run timer	
0000D8 _H	IPCP1 [R] H,W XXXXXXXX XXXXXXXX		IPCP0 [R] H,W XXXXXXXX XXXXXXXX		16-bit ICU	
0000DC _H	IPCP3 [R] H,W XXXXXXXX XXXXXXXX		IPCP2 [R] H,W XXXXXXXX XXXXXXXX			
0000E0 _H	–	ICS23 [R/W] B,H,W 00000000	–	ICS01 [R/W] B,H,W 00000000		
0000E4 _H to 000114 _H	–				Reserved	
000118 _H	GCN10 [R/W] H 00110010 00010000		–	GCN20 [R/W] B 00000000	PPG timer	
00011C _H	–				Reserved	
000120 _H	PTMR0 [R] H 11111111 11111111		PCSR0 [W] H,W XXXXXXXX XXXXXXXX		PPG0	
000124 _H	PDUT0 [W] H,W XXXXXXXX XXXXXXXX		PCNH0 [R/W] B 00000000	PCNL0 [R/W] B 000000X0		
000128 _H	PTMR1 [R] H 11111111 11111111		PCSR1 [W] H,W XXXXXXXX XXXXXXXX		PPG1	
00012C _H	PDUT1 [W] H,W XXXXXXXX XXXXXXXX		PCNH1 [R/W] B 00000000	PCNL1 [R/W] B 000000X0		

APPENDIX A I/O MAP

Table A-1 I/O Map (4 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
000130 _H	PTMR2 [R] H 11111111 11111111		PCSR2 [W] H,W XXXXXXXX XXXXXXXX		PPG2	
000134 _H	PDU2 [W] H,W XXXXXXXX XXXXXXXX		PCNH2 [R/W] B 00000000	PCNL2 [R/W] B 000000X0		
000138 _H	PTMR3 [R] H 11111111 11111111		PCSR3 [W] H,W XXXXXXXX XXXXXXXX		PPG3	
00013C _H	PDU3 [W] H,W XXXXXXXX XXXXXXXX		PCNH3 [R/W] B 00000000	PCNL3 [R/W] B 000000X0		
000140 _H to 0001FC _H	-				Reserved	
000200 _H	DMACA0 [R/W] B,H,W ^{*1} 00000000 0000XXXX XXXXXXXX XXXXXXXX				DMAC	
000204 _H	DMACB0 [R/W] B,H,W 00000000 00000000 XXXXXXXX XXXXXXXX					
000208 _H	DMACA1 [R/W] B,H,W ^{*1} 00000000 0000XXXX XXXXXXXX XXXXXXXX					
00020C _H	DMACB1 [R/W] B,H,W 00000000 00000000 XXXXXXXX XXXXXXXX					
000210 _H	DMACA2 [R/W] B,H,W ^{*1} 00000000 0000XXXX XXXXXXXX XXXXXXXX					
000214 _H	DMACB2 [R/W] B,H,W 00000000 00000000 XXXXXXXX XXXXXXXX					
000218 _H	DMACA3 [R/W] B,H,W ^{*1} 00000000 0000XXXX XXXXXXXX XXXXXXXX					
00021C _H	DMACB3 [R/W] B,H,W 00000000 00000000 XXXXXXXX XXXXXXXX					
000220 _H	DMACA4 [R/W] B,H,W ^{*1} 00000000 0000XXXX XXXXXXXX XXXXXXXX					
000224 _H	DMACB4 [R/W] B,H,W 00000000 00000000 XXXXXXXX XXXXXXXX					
000228 _H to 00023C _H	-				Reserved	
000240 _H	DMACR [R/W] B 0XX00000 XXXXXXXX XXXXXXXX XXXXXXXX				DMAC	
000244 _H to 000300 _H	-				Reserved	

Table A-1 I/O Map (5 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
000304 _H	—			ISIZE [R/W] B,H,W -----10	I-Cache	
000308 _H to 0003E0 _H	—					
0003E4 _H	—			ICHCR [R/W] B,H,W 0-000000	I-Cache	
0003E8 _H to 0003EF _H	—					
0003F0 _H	BSD0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				Bit Search Module	
0003F4 _H	BSD1 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
0003F8 _H	BSDC [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
0003FC _H	BSRR [R] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000400 _H	DDR _G [R/W] B 00000000	DDR _H [R/W] B -----00	—	DDR _J [R/W] B 00000000	R-bus Data Direction Register	
000404 _H to 00040C _H	—				Reserved	
000410 _H	PFRG [R/W] B 00-----	PFRH [R/W] B -----0-	—	PFRJ [R/W] B --00-00-	R-bus Port Function Register	
000414 _H to 00041C _H	—				Reserved	
000420 _H	PCRG [R/W] B ^{*2} 00000000	PCR _H [R/W] B -----000	—	PCR _J [R/W] B ^{*2} 00000000	R-bus pull-up resistor control Register	
000424 _H to 00043C _H	—				Reserved	

APPENDIX A I/O MAP

Table A-1 I/O Map (6 / 11)

Address	Register				Block
	+0	+1	+2	+3	
000440 _H	ICR00 [R/W] B,H,W ---11111	ICR01 [R/W] B,H,W ---11111	ICR02 [R/W] B,H,W ---11111	ICR03 [R/W] B,H,W ---11111	Interrupt Controller
000444 _H	ICR04 [R/W] B,H,W ---11111	ICR05 [R/W] B,H,W ---11111	ICR06 [R/W] B,H,W ---11111	ICR07 [R/W] B,H,W ---11111	
000448 _H	ICR08 [R/W] B,H,W ---11111	ICR09 [R/W] B,H,W ---11111	ICR10 [R/W] B,H,W ---11111	ICR11 [R/W] B,H,W ---11111	
00044C _H	ICR12 [R/W] B,H,W ---11111	ICR13 [R/W] B,H,W ---11111	ICR14 [R/W] B,H,W ---11111	ICR15 [R/W] B,H,W ---11111	
000450 _H	ICR16 [R/W] B,H,W ---11111	ICR17 [R/W] B,H,W ---11111	ICR18 [R/W] B,H,W ---11111	ICR19 [R/W] B,H,W ---11111	
000454 _H	ICR20 [R/W] B,H,W ---11111	ICR21 [R/W] B,H,W ---11111	ICR22 [R/W] B,H,W ---11111	ICR23 [R/W] B,H,W ---11111	
000458 _H	ICR24 [R/W] B,H,W ---11111	ICR25 [R/W] B,H,W ---11111	ICR26 [R/W] B,H,W ---11111	ICR27 [R/W] B,H,W ---11111	
00045C _H	ICR28 [R/W] B,H,W ---11111	ICR29 [R/W] B,H,W ---11111	ICR30 [R/W] B,H,W ---11111	ICR31 [R/W] B,H,W ---11111	
000460 _H	ICR32 [R/W] B,H,W ---11111	ICR33 [R/W] B,H,W ---11111	ICR34 [R/W] B,H,W ---11111	ICR35 [R/W] B,H,W ---11111	
000464 _H	ICR36 [R/W] B,H,W ---11111	ICR37 [R/W] B,H,W ---11111	ICR38 [R/W] B,H,W ---11111	ICR39 [R/W] B,H,W ---11111	
000468 _H	ICR40 [R/W] B,H,W ---11111	ICR41 [R/W] B,H,W ---11111	ICR42 [R/W] B,H,W ---11111	ICR43 [R/W] B,H,W ---11111	
00046C _H	ICR44 [R/W] B,H,W ---11111	ICR45 [R/W] B,H,W ---11111	ICR46 [R/W] B,H,W ---11111	ICR47 [R/W] B,H,W ---11111	
000470 _H to 00047C _H	-				

Table A-1 I/O Map (7 / 11)

Address	Register				Block
	+0	+1	+2	+3	
000480 _H	RSRR [R/W] B,H,W 10000000 (INIT) -0-XX-00 (INIT) XXX--X00 (RST)	STCR [R/W] B,H,W 00110011 (INIT) 0011XX11 (INIT) 00X1XXXX (RST)	TBCR [R/W] B,H,W 00XXXX00 (INIT) 00XXXXXX (RST)	CTBR [W] B,H,W XXXXXXXX (INIT) XXXXXXXX (RST)	Clock Control unit
000484 _H	CLKR [R/W] B,H,W 00000000 (INIT) XXXXXXXXXX (RST)	WPR -----*4	DIVR0 [R/W] B,H,W 00000011 (INIT) XXXXXXXXXX (RST)	DIVR1 [R/W] B,H,W 00000000 (INIT) XXXXXXXXXX (RST)	
000488 _H to 0005FC _H	-				Reserved
000600 _H	DDR0 [R/W] B 00000000	DDR1 [R/W] B 00000000	DDR2 [R/W] B 00000000	-	T-unit Data Direction Register
000604 _H	-		DDR6 [R/W] B 00000000	-	
000608 _H	DDR8 [R/W] B 00000000	DDR9 [R/W] B -00000000	DDRA [R/W] B 00000000	DDRB [R/W] B 00000000	
00060C _H	-				
000610 _H	-				T-unit Port Function Register
000614 _H	-		PFR6 [R/W] B 11111111	PFR61 [R/W] B ----0000	
000618 _H	PFR8 [R/W] B 111--0--	PFR9 [R/W] B -0000111	PFRA1 [R/W] B 11111111	PFRB1 [R/W] B 00000000	
00061C _H	PFRB2 [R/W] B 000---00	-	PFRA2 [R/W] B --0----	-	
000620 _H	PCR0 [R/W] B 00000000	PCR1 [R/W] B 00000000	PCR2 [R/W] B 00000000	-	T-unit Pull-up Resistor Control Register
000624 _H	-		PCR6 [R/W] B 00000000	-	
000628 _H	PCR8 [R/W] B 00000000	PCR9 [R/W] B 00000000	PCRA [R/W] B 00000000	PCRB [R/W] B 00000000	
00062C _H	-				
000630 _H to 00063C _H	-				Reserved

APPENDIX A I/O MAP

Table A-1 I/O Map (8 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
000640 _H	ASR0 [R/W] H,W 00000000 00000000		ACRH [R/W] H,W 1111XX00	ACR0L [R/W] H,W 00000000	T-unit	
000644 _H	ASR1 [R/W] H,W XXXXXXXX XXXXXXXX		ACR1H [R/W] B,H,W XXXXXXXX	ACR1L [R/W] B,H,W XXXXXXXX		
000648 _H	ASR2 [R/W] H,W XXXXXXXX XXXXXXXX		ACR2H [R/W] B,H,W XXXXXXXX	ACR2L [R/W] B,H,W XXXXXXXX		
00064C _H	ASR3 [R/W] H,W XXXXXXXX XXXXXXXX		ACR3H [R/W] B,H,W XXXXXXXX	ACR3L [R/W] B,H,W XXXXXXXX		
000650 _H	ASR4 [R/W] H,W XXXXXXXX XXXXXXXX		ACR4H [R/W] B,H,W XXXXXXXX	ACR4L [R/W] B,H,W XXXXXXXX		
000654 _H	ASR5 [R/W] H,W XXXXXXXX XXXXXXXX		ACR5H [R/W] B,H,W XXXXXXXX	ACR5L [R/W] B,H,W XXXXXXXX		
000658 _H	ASR6 [R/W] H,W XXXXXXXX XXXXXXXX		ACR6H [R/W] B,H,W XXXXXXXX	ACR6L [R/W] B,H,W XXXXXXXX		
00065C _H	ASR7 [R/W] H,W XXXXXXXX XXXXXXXX		ACR7H [R/W] B,H,W XXXXXXXX	ACR7L [R/W] B,H,W XXXXXXXX		
000660 _H	AWR0H[R/W] B,H,W 01111111	AWR0L[R/W] B,H,W 11111111 (INIT) 11111011 (RST)	AWR1H[R/W] B,H,W XXXXXXXX	AWR1L[R/W] B,H,W XXXXXXXX		
000664 _H	AWR2H[R/W] B,H,W XXXXXXXX	AWR2L[R/W] B,H,W XXXXXXXX	AWR3H[R/W] B,H,W XXXXXXXX	AWR3L[R/W] B,H,W XXXXXXXX		
000668 _H	AWR4H[R/W] B,H,W XXXXXXXX	AWR4L[R/W] B,H,W XXXXXXXX	AWR5H[R/W] B,H,W XXXXXXXX	AWR5L[R/W] B,H,W XXXXXXXX		
00066C _H	AWR6H[R/W] B,H,W XXXXXXXX	AWR6L[R/W] B,H,W XXXXXXXX	AWR7H[R/W] B,H,W XXXXXXXX	AWR7L[R/W] B,H,W XXXXXXXX		
000670 _H	MCRA [R/W] B,H,W XXXXXXXX	MCRB [R/W] B,H,W XXXXXXXX	-			
000674 _H	-					

Table A-1 I/O Map (9 / 11)

Address	Register				Block				
	+0	+1	+2	+3					
000678 _H	IOWR0 [R/W] B,H,W XXXXXXXXXX	IOWR1 [R/W] B,H,W XXXXXXXXXX	—		T-unit				
00067C _H	—								
000680 _H	CSER [R/W] B,H,W 00000001	CHER [R/W] B,H,W 11111111	—	TCR [R/W] B,H,W 00000000 (INIT) 0000XXXX (RST)					
000684 _H	RCRH [R/W] B,H,W 00XXXXXX	RCRL [R/W] B,H,W XXXX0XXX	—						
00068C _H to 0007F8 _H	—								
0007FC _H	—	MODR [W] * ³ XXXXXXXXXX	—						
000800 _H to 000AFC _H	—								
000B00 _H	ESTS0 [R/W] B X0000000	ESTS1 [R/W] B XXXXXXXXXX	ESTS2 [R] B 1XXXXXXXX	—					
000B04 _H	ECTL0 [R/W] B 0X000000	ECTL1 [R/W] B 00000000	ECTL2 [W] B 000X0000	ECTL3 [R/W] B 00X00X11					
000B08 _H	ECNT0 [W] B XXXXXXXXXX	ECNT1 [W] B XXXXXXXXXX	EUSA [W] B XXX00000	EDTC [W] B 0000XXXX					
000B0C _H	EWPT [R] H 00000000 00000000		ECTL4[R] ([R/W]) B -0X00000	ECTL5 [R] ([R/W]) B ---000X	DSU (only evaluation chip)				
000B10 _H	EDTR0 [W] B XXXXXXXX XXXXXXXX		EDTR1 [W] H XXXXXXXX XXXXXXXX						
000B14 _H to 000B1C _H	—								
000B20 _H	EIA0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX								
000B24 _H	EIA1 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX								
000B28 _H	EIA2 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX								

APPENDIX A I/O MAP

Table A-1 I/O Map (10 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
000B2C _H	EIA3 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				DSU (only evaluation chip)	
000B30 _H	EIA4 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B34 _H	EIA5 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B38 _H	EIA6 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B3C _H	EIA7 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B40 _H	EDTA [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B44 _H	EDTM [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B48 _H	EOA0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B4C _H	EOA1 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B50 _H	EPCR [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B54 _H	EPSR [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B58 _H	EIAM0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B5C _H	EIAM1 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B60 _H	EOAM0/EODM0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B64 _H	EOAM1/EODM1 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B68 _H	EOD0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B6C _H	EOD1 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
000B70 _H to 000FFC _H	—				Reserved	

Table A-1 I/O Map (11 / 11)

Address	Register				Block	
	+0	+1	+2	+3		
001000 _H	DMASA0 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				DMAC	
001004 _H	DMADA0 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001008 _H	DMASA1 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
00100C _H	DMADA1 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001010 _H	DMASA2 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001014 _H	DMADA2 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001018 _H	DMASA3 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
00101C _H	DMADA3 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001020 _H	DMASA4 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001024 _H	DMADA4 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX					
001028 _H to 001FFC _H	—				Reserved	

*1: The lower 16-bit (DTC15 to DTC0) of DMACA0 to DMACA4 cannot access by byte.

*2: No register in 000420_H and 000423_H in MB91302A and MB91V301A.

*3: This register is set by the mode vector fetch. It cannot access during the normal operation.

*4: Reserved register. Access is disabled.

APPENDIX B INTERRUPT VECTOR

Table B-1 shows the interrupt vector table, which gives the interrupt source and interrupt vector/interrupt control register allocations for the MB91301 series.

■ Interrupt Vectors

Table B-1 Interrupt Vectors (1 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	TBR default address	RN
	Decimal	Hexadecimal				
Reset	0	00	–	3FC _H	000FFFFC _H	–
Mode vector	1	01	–	3F8 _H	000FFFF8 _H	–
Reserved for system	2	02	–	3F4 _H	000FFFF4 _H	–
Reserved for system	3	03	–	3F0 _H	000FFFF0 _H	–
Reserved for system	4	04	–	3EC _H	000FFFEC _H	–
Reserved for system	5	05	–	3E8 _H	000FFFE8 _H	–
Reserved for system	6	06	–	3E4 _H	000FFFE4 _H	–
No-coprocessor trap	7	07	–	3E0 _H	000FFFE0 _H	–
Coprocessor error trap	8	08	–	3DC _H	000FFFDC _H	–
INTE instruction	9	09	–	3D8 _H	000FFFD8 _H	–
Reserved for system	10	0A	–	3D4 _H	000FFFD4 _H	–
Reserved for system	11	0B	–	3D0 _H	000FFFD0 _H	–
Step trace trap	12	0C	–	3CC _H	000FFFCC _H	–
NMI request (tool)	13	0D	–	3C8 _H	000FFFC8 _H	–
Undefined instruction exception	14	0E	–	3C4 _H	000FFFC4 _H	–
NMI request	15	0F	15(F _H), fixed	3C0 _H	000FFFC0 _H	–
External Interrupt 0	16	10	ICR00	3BC _H	000FFFBC _H	6
External Interrupt 1	17	11	ICR01	3B8 _H	000FFF8 _H	7
External Interrupt 2	18	12	ICR02	3B4 _H	000FFF4 _H	11
External Interrupt 3	19	13	ICR03	3B0 _H	000FFF0 _H	12
External Interrupt 4	20	14	ICR04	3AC _H	000FFFAC _H	

Table B-1 Interrupt Vectors (2 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	TBR default address	RN
	Decimal	Hexadecimal				
External Interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H	
External Interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H	-
External Interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H	-
Reload Timer 0	24	18	ICR08	39C _H	000FFF9C _H	8
Reload Timer 1	25	19	ICR09	398 _H	000FFF98 _H	9
Reload Timer 2	26	1A	ICR10	394 _H	000FFF94 _H	10
UART0 (reception completed)	27	1B	ICR11	390 _H	000FFF90 _H	0
UART1 (reception completed)	28	1C	ICR12	38C _H	000FFF8C _H	1
UART2 (reception completed)	29	1D	ICR13	388 _H	000FFF88 _H	2
UART0 (transmission completed)	30	1E	ICR14	384 _H	000FFF84 _H	3
UART1 (transmission completed)	31	1F	ICR15	380 _H	000FFF80 _H	4
UART2 (transmission completed)	32	20	ICR16	37C _H	000FFF7C _H	5
DMAC0 (end, error)	33	21	ICR17	378 _H	000FFF78 _H	-
DMAC1 (end, error)	34	22	ICR18	374 _H	000FFF74 _H	-
DMAC2 (end, error)	35	23	ICR19	370 _H	000FFF70 _H	-
DMAC3 (end, error)	36	24	ICR20	36C _H	000FFF6C _H	-
DMAC4 (end, error)	37	25	ICR21	368 _H	000FFF68 _H	-
A/D	38	26	ICR22	364 _H	000FFF64 _H	15
PPG0	39	27	ICR23	360 _H	000FFF60 _H	13
PPG1	40	28	ICR24	35C _H	000FFF5C _H	14
PPG2	41	29	ICR25	358 _H	000FFF58 _H	-
PPG3	42	2A	ICR26	354 _H	000FFF54 _H	-
Reserved for system	43	2B	ICR27	350 _H	000FFF50 _H	-
U-TIMER0	44	2C	ICR28	34C _H	000FFF4C _H	-
U-TIMER1	45	2D	ICR29	348 _H	000FFF48 _H	-
U-TIMER2	46	2E	ICR30	344 _H	000FFF44 _H	-

APPENDIX B INTERRUPT VECTOR

Table B-1 Interrupt Vectors (3 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	TBR default address	RN
	Decimal	Hexadecimal				
Time-base timer overflow	47	2F	ICR31	340 _H	000FFF40 _H	–
I ² C I/F0	48	30	ICR32	33C _H	000FFF3C _H	–
I ² C I/F1	49	31	ICR33	338 _H	000FFF38 _H	–
Reserved for system	50	32	ICR34	334 _H	000FFF34 _H	–
Reserved for system	51	33	ICR35	330 _H	000FFF30 _H	–
16-bit free run timer	52	34	ICR36	32C _H	000FFF2C _H	–
ICU0 (fetch)	53	35	ICR37	328 _H	000FFF28 _H	–
ICU1 (fetch)	54	36	ICR38	324 _H	000FFF24 _H	–
ICU2 (fetch)	55	37	ICR39	320 _H	000FFF20 _H	–
ICU3 (fetch)	56	38	ICR40	31C _H	000FFF1C _H	–
Reserved for system	57	39	ICR41	318 _H	000FFF18 _H	–
Reserved for system	58	3A	ICR42	314 _H	000FFF14 _H	–
Reserved for system	59	3B	ICR43	310 _H	000FFF10 _H	–
Reserved for system	60	3C	ICR44	30C _H	000FFF0C _H	–
Reserved for system	61	3D	ICR45	308 _H	000FFF08 _H	–
Reserved for system	62	3E	ICR46	304 _H	000FFF04 _H	–
Delayed interrupt source bit	63	3F	ICR47	300 _H	000FFF00 _H	–
Reserved for system (used by REALOS)	64	40	–	2FC _H	000FFEFC _H	–
Reserved for system (used by REALOS)	65	41	–	2F8 _H	000FFEF8 _H	–
Reserved for system	66	42	–	2F4 _H	000FFEF4 _H	–
Reserved for system	67	43	–	2F0 _H	000FFEF0 _H	–
Reserved for system	68	44	–	2EC _H	000FFEEC _H	–
Reserved for system	69	45	–	2E8 _H	000FFEE8 _H	–
Reserved for system	70	46	–	2E4 _H	000FFEE4 _H	–
Reserved for system	71	47	–	2E0 _H	000FFEE0 _H	–
Reserved for system	72	48	–	2DC _H	000FFEDC _H	–
Reserved for system	73	49	–	2D8 _H	000FFED8 _H	–

Table B-1 Interrupt Vectors (4 / 4)

Interrupt source	Interrupt number		Interrupt level	Offset	TBR default address	RN
	Decimal	Hexadecimal				
Reserved for system	74	4A	–	2D4 _H	000FFED4 _H	–
Reserved for system	75	4B	–	2D0 _H	000FFED0 _H	–
Reserved for system	76	4C	–	2CC _H	000FFECC _H	–
Reserved for system	77	4D	–	2C8 _H	000FFEC8 _H	–
Reserved for system	78	4E	–	2C4 _H	000FFEC4 _H	–
Reserved for system	79	4F	–	2C0 _H	000FFEC0 _H	–
Used in INT instruction	80 to 255	50 to FF	–	2BC _H to 000 _H	000FFEB _H to 000FFC00 _H	–

- Notes:
- The ICR is a register set in the interrupt controller that sets the interrupt level for each interrupt request. The ICR is provided to support each interrupt request.
 - The TBR is a register that indicates the first address of the EIT vector table. The vector address can be obtained by adding the offset value defined for each TBR and EIT source to the address.

Reference:

The 1 Kbyte area from the address indicated by the TBR is the vector area for EIT.

The size of each vector is 4 bytes and the relation between the vector number and vector address can be represented as follows:

$$\begin{aligned} \text{vctadr} &= \text{TBR} + \text{vctofs} \\ &= \text{TBR} + (3FC_{H}-4 \times \text{vct}) \end{aligned}$$

vctadr: Vector address

vctofs: Vector offset

vct: Vector number

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-1 to Table C-4 list the pin states in each CPU state.

■ Meaning of Terms in the Pin State Table

Terms related to the pin state have the following meanings:

- Input ready
Means that the input function can be used.
- Input 0 fixed
Input level is internally fixed at 0 to prevent leakage due to the input open.
- Output Hi-Z
Means that the pin drive transistor is disabled and the pin is set to high impedance.
- Output held
Means that the state output just before a mode is entered is output as is.
That is, if any built-in peripheral with output is active, the output is determined by the built-in peripheral. For output as a port, the output is held.
- Preceding state held
Means that the state output just before a mode is entered is output as is. Also means input ready if the state is the input state.

■ Pin State Table

Table C-1 Pin States in External Bus 32-Bit Mode (1 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits			HIZ=0	HIZ=1		
			CS shared	CS not shared							
1 to 5	P13 to P17	D11 to D15	D11 to D15	P13 to P17							
8 to 15	P20 to P27	D16 to D23	D16 to D23	P20 to P27							
18 to 25	-	D24 to D31	D24 to D31	D24 to D31							
28	P80	RDY	P80	P80							
29	P81	<u>BGRNT</u>	P81	P81							
30	P82	BRQ	P82	P82							
31	P83	<u>RD</u>	<u>RD</u>	<u>RD</u>							
32	P84	DQMUU/ <u>WR0</u>	DQMUU/ <u>WR0</u>	DQMUU/ <u>WR0</u>		"H" output					
33	P85	DQMUL/ <u>WR1</u>	DQMUL/ <u>WR1</u>	P85							
34	P86	DQMLU/ <u>WR2</u>	DQMLU/ <u>WR2</u>	P86							
35	P87	DQMLL/ <u>WR3</u>	DQMLL/ <u>WR3</u>	P87							
36	P90	SYSCLK	SYSCLK	SYSCLK	Asserted : "L" output Negated : CLK output	P : Previous state held F : SYSCLK output	P : Previous state held F : "H" or "L" output	Output Hi-Z/ input 0 fixed	F : CLK output	F : CLK output	
37	P91	MCLKE	MCLKE	MCLKE	"H" output	F : "L" output	F : "L" output	F : Output Hi-Z	Output Hi-Z	"H" output	
38	P92	MCLK	MCLK	MCLK	Asserted : "L" output Negated : CLK output	P : Previous state held F : "H" output	P : Previous state held F : "H" output	F : Output Hi-Z	Output Hi-Z	F : CLK output	
39	P93	-	P93	P93	Output Hi-Z Input ready	Previous state held	Previous state held	Output Hi-Z	Port	Port	
40	P94	<u>SRAS/LBA/</u> <u>AS</u>	P94	P94	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	F : "H" output	
41	P95	<u>SCAS/BAA</u>	P95	P95	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	"H" output	
42	P96	<u>SWE/WR</u>	P96	P96	Output Hi-Z Input ready	P : Previous state held F : SWEX output	Previous state held	Output Hi-Z/ input 0 fixed	Output Hi-Z	Previous state held	

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-1 Pin States in External Bus 32-Bit Mode (2 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits			HIZ=0	HIZ=1		
			Bus width 8 bits	Bus width 8 bits						CS shared	CS not shared
45 to 52	P40 to P47	A00 to A07	A00 to A07	A00 to A07	FF output	P : Previous state held F : Address output	The same as stated left	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z	
55 to 62	P50 to P57	A08 to A15	A08 to A15	A08 to A15							
64 to 67	P60 to P63	A16 to A19	A16 to A19	A16 to A19							
68	P64	A20/SDA0*	A20	A20							
69	P65	A21/SCL0*	A21	A21							
70	P66	A22/SDA1*	A22	A22							
71	P67	A23/SCL1*	A23	A23							
76 to 79	-	AN3 to AN0	AN0 to AN3	AN0 to AN3	Input invalid	Previous state held	Input invalid	Input invalid	Previous state held	Previous state held	Previous state held
81	PG0	INT0/ICU0*	PG0	PG0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	P : Previous state held F : Input ready	P : Output Hi-Z F : Input ready	Normal operation	Normal operation	
82	PG1	INT1/ICU1*	PG1	PG1							
83	PG2	INT2/ICU2*	PG2	PG2							
84	PG3	INT3/ICU3*	PG3	PG3							
85	PG4	INT4/ATG/ FRCK*	PG4	PG4							
86	PG5	INT5/SIN2	PG5	PG5							
87	PG6	INT6/SOT2	PG6	PG6							
88	PG7	INT7/SCK2	PG7	PG7							
90	PJ0	SIN0	PJ0	PJ0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	Normal operation	
91	PJ1	SOT0	PJ1	PJ1							
92	PJ2	SCK0	PJ2	PJ2							
93	PJ3	SIN1	PJ3	PJ3							
94	PJ4	SOT1	PJ4	PJ4							
95	PJ5	SCK1	PJ5	PJ5							
96	PJ6	PPG0	PJ6	PJ6							
97	PJ7	TRG0	PJ7	PJ7							
98	PH0	TIN0	PH0	PH0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	Normal operation	
99	PH1	TIN1/PPG3	PH1	PH1							
100	PH2	TIN2/TRG3	PH2	PH2							
103	PB0	DREQ0	PB0	PB0		Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	
104	PB1	DACK0	PB1	PB1							
105	PB2	DEOP0	PB2	PB2							
106	PB3	DREQ1	PB3	PB3							
107	PB4	DACK1/ TRG1	PB4	PB4							
108	PB5	DEOP1/ PPG1	PB5	PB5							
109	PB6	IOWR	PB6	PB6							
110	PB7	IORD	PB7	PB7							

Table C-1 Pin States in External Bus 32-Bit Mode (3 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Initial value	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits		Sleep mode	HIZ=0		
			CS0	CS0	CS0		"H" output	"H" output	Output Hi-Z	F : SREN=0: "H" output SREN=1 : Output Hi-Z
122	PA0	CS0	CS0	CS0						
123	PA1	CS1	CS1	CS1						
124	PA2	CS2	CS2	CS2						
125	PA3	CS3	CS3	CS3						
126	PA4	CS4/TRG2	CS4	CS4						
127	PA5	CS5/PPG2	CS5	CS5						
128	PA6	CS6	CS6	CS6						
129	PA7	CS7	CS7	CS7						
132 to 139	P00 to P07	D00 to D07	D00 to D07	P00 to P07		Output Hi-Z Input ready	P : Previous state held F : Output held or Hi-Z	P : Previous state held F : Output held or Hi-Z	Output Hi-Z/ input 0 fixed	
142 to 144	P10 to P12	D08 to D10	D08 to D10	P10 to P12					Output Hi-Z	Output Hi-Z

P : General-purpose port selected, F : Specified function selected

* : The following port's function can be used on only MB91302A and MB91V301A; SDA0, SCL0, SDA1, SCL1 of 68 to 71 pin, ICU0 to ICU3, FRCK of 81 to 85 pin.

Note : The bus width is determined after a mode vector fetch.

The bus width at initialization time is 8 bits.

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-2 Pin States in External Bus 16-Bit Mode (1 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization ($\overline{\text{INIT}}$)		Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Initial value		$\text{HIZ}=0$	$\text{HIZ}=1$	CS shared	CS not shared
				Bus width 32 bits	Bus width 8 bits					
1 to 5	P13 to P17	D11 to D15	P13 to P17	P13 to P17						
8 to 15	P20 to P27	D16 to D23	D16 to D23	P20 to P27						
18 to 25	-	D24 to D31	D24 to D31	D24 to D31						
28	P80	RDY	P80	P80		Output Hi-Z Input ready	P : Previous state held F : Output held or Hi-Z	P : Previous state held F : Output held or Hi-Z	Output Hi-Z/ input 0 fixed	Output Hi-Z
29	P81	$\overline{\text{BGRNT}}$	P81	P81			P : Previous state held F : "H" output	Previous state held	"L" output	"L" output
30	P82	BRQ	P82	P82			P : Previous state held F : BRQ input invalid		BRQ input	BRQ input
31	P83	$\overline{\text{RD}}$	$\overline{\text{RD}}$	$\overline{\text{RD}}$		"H" output	P : Previous state held F : "H" output	Previous state held	Output Hi-Z	Output Hi-Z
32	P84	DQMUU/ $\overline{\text{WR0}}$	DQMUU/ $\overline{\text{WR0}}$	DQMUU/ $\overline{\text{WR0}}$						
33	P85	DQMUL/ $\overline{\text{WR1}}$	DQMUL/ $\overline{\text{WR1}}$	P85		F : "H" output	P : Previous state held F : "H" output	Previous state held	Output Hi-Z	Output Hi-Z
34	P86	DQMLU/ $\overline{\text{WR2}}$	P86	P86						
35	P87	DQMUL/ $\overline{\text{WR3}}$	P87	P87						
36	P90	SYSCLK	SYSCLK	SYSCLK	Asserted : "L" output Negated : CLK output	P : Previous state held F : SYSCLK output	P : Previous state held F : "H" or "L" output	Output Hi-Z/ input 0 fixed	F : CLK output	F : CLK output
37	P91	MCLKE	MCLKE	MCLKE	"H" output	F : "L" output	F : "L" output	F : Output Hi-Z	Output Hi-Z	"H" output
38	P92	MCLK	MCLK	MCLK	Asserted : "L" output Negated : CLK output	P : Previous state held F : "H" output	P : Previous state held F : "H" output	F : Output Hi-Z	Output Hi-Z	F : CLK output
39	P93	-	P93	P93	Output Hi-Z Input ready	Previous state held	Previous state held	Output Hi-Z	Output Hi-Z	Output Hi-Z
40	P94	$\overline{\text{SRAS/LBA/AS}}$	P94	P94	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	F : "H" output
41	P95	$\overline{\text{SCAS/BAA}}$	P95	P95	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	"H" output
42	P96	$\overline{\text{SWE/WR}}$	P96	P96	Output Hi-Z Input ready	P : Previous state held F : SWEX output	Previous state held	Output Hi-Z/ input 0 fixed	Output Hi-Z	Previous state held

Table C-2 Pin States in External Bus 16-Bit Mode (2 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization ($\overline{\text{INIT}}$)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits			HIZ=0	HIZ=1		
			Bus width 8 bits							CS shared	CS not shared
45 to 52	P40 to P47	A00 to A07	A00 to A07	A00 to A07	A00 to A07	FF output	P : Previous state held F : Address output	The same as stated left	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z
55 to 62	P50 to P57	A08 to A15	A08 to A15	A08 to A15	A08 to A15						
64 to 67	P60 to P63	A16 to A19	A16 to A19	A16 to A19	A16 to A19						
68	P64	A20/SDA0*	A20	A20	A20						
69	P65	A21/SCL0*	A21	A21	A21						
70	P66	A22/SDA1*	A22	A22	A22						
71	P67	A23/SCL1*	A23	A23	A23						
76 to 79	-	AN0 to AN3	AN0 to AN3	AN0 to AN3	Input invalid	Previous state held	Input invalid	Input invalid	Previous state held	Previous state held	Previous state held
81	PG0	INT0/ICU0*	PG0	PG0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	P : Previous state held F : Input ready	P : Output Hi-Z F : Input ready	Normal operation	Normal operation	
82	PG1	INT1/ICU1*	PG1	PG1							
83	PG2	INT2/ICU2*	PG2	PG2							
84	PG3	INT3/ICU3*	PG3	PG3							
85	PG4	INT4/ATG/ FRCK*	PG4	PG4							
86	PG5	INT5/SIN2	PG5	PG5							
87	PG6	INT6/SOT2	PG6	PG6							
88	PG7	INT7/SCK2	PG7	PG7							
90	PJ0	SIN0	PJ0	PJ0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	Normal operation	
91	PJ1	SOT0	PJ1	PJ1							
92	PJ2	SCK0	PJ2	PJ2							
93	PJ3	SIN1	PJ3	PJ3							
94	PJ4	SOT1	PJ4	PJ4							
95	PJ5	SCK1	PJ5	PJ5							
96	PJ6	PPG0	PJ6	PJ6							
97	PJ7	TRG0	PJ7	PJ7							
98	PH0	TIN0	PH0	PH0	Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	Normal operation	
99	PH1	TIN1/PPG3	PH1	PH1							
100	PH2	TIN2/TRG3	PH2	PH2							
103	PB0	DREQ0	PB0	PB0		Output Hi-Z Input ready	P : Previous state held F : Normal operation	Previous state held	Output Hi-Z/ input 0 fixed	Normal operation	Normal operation
104	PB1	DACK0	PB1	PB1							
105	PB2	DEOP0	PB2	PB2							
106	PB3	DREQ1	PB3	PB3							
107	PB4	DACK1/ TRG1	PB4	PB4							
108	PB5	DEOP1/PPG1	PB5	PB5							
109	PB6	$\overline{\text{IOWR}}$	PB6	PB6							
110	PB7	$\overline{\text{IORD}}$	PB7	PB7							

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-2 Pin States in External Bus 16-Bit Mode (3 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	HIZ=0			HIZ=1			
			Bus width 32 bits	Bus width 8 bits					CS shared	CS not shared	
122	PA0	$\overline{\text{CS}0}$	$\overline{\text{CS}0}$	$\overline{\text{CS}0}$	"H" output	"H" output	"H" output	Output Hi-Z	F : SREN=0 : "H" output SREN=1 : Output Hi-Z	F : SREN=0 : "H" output SREN=1 : Output Hi-Z	
123	PA1	$\overline{\text{CS}1}$	$\overline{\text{CS}1}$	$\overline{\text{CS}1}$							
124	PA2	$\overline{\text{CS}2}$	$\overline{\text{CS}2}$	$\overline{\text{CS}2}$							
125	PA3	$\overline{\text{CS}3}$	$\overline{\text{CS}3}$	$\overline{\text{CS}3}$							
126	PA4	$\overline{\text{CS}4}/\text{TRG}2$	$\overline{\text{CS}4}$	$\overline{\text{CS}4}$							
127	PA5	$\overline{\text{CS}5}/\text{PPG}2$	$\overline{\text{CS}5}$	$\overline{\text{CS}5}$							
128	PA6	$\overline{\text{CS}6}$	$\overline{\text{CS}6}$	$\overline{\text{CS}6}$							
129	PA7	$\overline{\text{CS}7}$	$\overline{\text{CS}7}$	$\overline{\text{CS}7}$							
132 to 139	P00 to P07	D00 to D07	P00 to P07	P00 to P07	Output Hi-Z Input ready	P : Previous state held F : Output held or Hi-Z	P : Previous state held F : Output held or Hi-Z	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z	
142 to 144	P10 to P12	D08 to D10	P10 to P12	P10 to P12							

P : General-purpose port selected, F : Specified function selected

* : The following port's function can be used on only MB91302A and MB91V301A; SDA0, SCL0, SDA1, SCL1 of 68 to 71 pin, ICU0 to ICU3, FRCK of 81 to 85 pin.

Note : The bus width is determined after a mode vector fetch.

The bus width at initialization time is 8 bits.

Table C-3 Pin States in External Bus 8-Bit Mode (1 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Initial value		HIZ=0	HIZ=1		
			Bus width 32 bits	Bus width 8 bits					CS shared	CS not shared
1 to 5	P13 to P17	D11 to D15	P13 to P17	P13 to P17	Output Hi-Z Input ready	P : Previous state held F : Output held or Hi-Z	P : Previous state held F : Output held or Hi-Z	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z
8 to 15	P20 to P27	D16 to D23	P20 to P27	P20 to P27						
18 to 25	-	D24 to D31	D24 to D31	D24 to D31						
28	P80	RDY	P80	P80	Output Hi-Z Input ready	P : Previous state held F : RDY input	Previous state held	Output Hi-Z/ input 0 fixed	P : Previous state held F : RDY input	P : Previous state held F : RDY input
29	P81	<u>BGRNT</u>	P81	P81						
30	P82	BRQ	P82	P82						
31	P83	<u>RD</u>	<u>RD</u>	<u>RD</u>	"H" output	P : Previous state held F : "H" output	Previous state held	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z
32	P84	DQMUU/ <u>WR0</u>	DQMUU/ <u>WR0</u>	DQMUU/ <u>WR0</u>						
33	P85	DQMUL/ <u>WR1</u>	P85	P85						
34	P86	DQMLU/ <u>WR2</u>	P86	P86	F : "H" output	P : Previous state held F : "H" output	Previous state held	Output Hi-Z	Output Hi-Z	Output Hi-Z
35	P87	DQMLL/ <u>WR3</u>	P87	P87						
36	P90	SYSCLK	SYSCLK	SYSCLK	Asserted: "L" output Negated: CLK output	P : Previous state held F : SYSCLK output	P : Previous state held F : "H" or "L" output	Output Hi-Z/ input 0 fixed	F : CLK output	F : CLK output
37	P91	MCLKE	MCLKE	MCLKE	"H" output	F : "L" output	F : "L" output	Output Hi-Z	Output Hi-Z	"H" output
38	P92	MCLK	MCLK	MCLK	Asserted : "L" output Negated : CLK output	P : Previous state held F : "H" output	P : Previous state held F : "H" output	Output Hi-Z	Output Hi-Z	F : CLK output
39	P93	-	P93	P93	Output Hi-Z Input ready	Previous state held	Previous state held	Output Hi-Z	Output Hi-Z	Output Hi-Z
40	P94	<u>SRAS/LBA/</u> <u>AS</u>	P94	P94	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	F : "H" output
41	P95	<u>SCAS/BAA</u>	P95	P95	Output Hi-Z Input ready	P : Previous state held F : "H" output	"H" output	Output Hi-Z	Output Hi-Z	"H" output
42	P96	<u>SWE/WR</u>	P96	P96	Output Hi-Z Input ready	P : Previous state held F : SWEX output	Previous state held	Output Hi-Z/ input 0 fixed	Output Hi-Z	Previous state held

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-3 Pin States in External Bus 8-Bit Mode (2 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization ($\overline{\text{INIT}}$)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits			HIZ=0	HIZ=1		
			Bus width 8 bits	Bus width 8 bits						CS shared	CS not shared
45 to 52	P40 to P47	A00 to A07	A00 to A07	A00 to A07							
55 to 62	P50 to P57	A08 to A15	A08 to A15	A08 to A15							
64 to 67	P60 to P63	A16 to A19	A16 to A19	A16 to A19							
68	P64	A20/SDA0*		A20	A20						
69	P65	A21/SCL0*		A21	A21						
70	P66	A22/SDA1*		A22	A22						
71	P67	A23/SCL1*		A23	A23						
76 to 79	-	AN0 to AN3	AN0 to AN3	AN0 to AN3		Input invalid	Previous state held	Input invalid	Input invalid	Previous state held	Previous state held
81	PG0	INT0/ICU0*	PG0	PG0							
82	PG1	INT1/ICU1*	PG1	PG1							
83	PG2	INT2/ICU2*	PG2	PG2							
84	PG3	INT3/ICU3*	PG3	PG3							
85	PG4	INT4/ATG/ FRCK*	PG4	PG4							
86	PG5	INT5/SIN2	PG5	PG5							
87	PG6	INT6/SOT2	PG6	PG6							
88	PG7	INT7/SCK2	PG7	PG7							
90	PJ0	SIN0	PJ0	PJ0							
91	PJ1	SOT0	PJ1	PJ1							
92	PJ2	SCK0	PJ2	PJ2							
93	PJ3	SIN1	PJ3	PJ3							
94	PJ4	SOT1	PJ4	PJ4							
95	PJ5	SCK1	PJ5	PJ5							
96	PJ6	PPG0	PJ6	PJ6							
97	PJ7	TRG0	PJ7	PJ7							
98	PH0	TIN0	PH0	PH0							
99	PH1	TIN1/PPG3	PH1	PH1							
100	PH2	TIN2/TRG3	PH2	PH2							
103	PB0	DREQ0	PB0	PB0							
104	PB1	DACK0	PB1	PB1							
105	PB2	DEOP0	PB2	PB2							
106	PB3	DREQ1	PB3	PB3							
107	PB4	DACK1/ TRG1	PB4	PB4							
108	PB5	DEOP1/PPG1	PB5	PB5							
109	PB6	$\overline{\text{IOWR}}$	PB6	PB6							
110	PB7	$\overline{\text{IORD}}$	PB7	PB7							

Table C-3 Pin States in External Bus 8-Bit Mode (3 / 3)

Pin no.	Port name	Specified function name	Function name	At initialization (INIT)		Initial value	Sleep mode	Stop mode		Bus open (BGRNT)	
				Function name	Bus width 32 bits			HIZ=0	HIZ=1		
			CS0	CS1	CS2			CS3	CS4	CS5	CS6
122	PA0	CS0	CS0	CS0	CS0	"H" output	"H" output	"H" output	Output Hi-Z	F : SREN=0: "H" output SREN=1 : Output Hi-Z	F : SREN=0: "H" output SREN=1 : Output Hi-Z
123	PA1	CS1	CS1	CS1	CS1						
124	PA2	CS2	CS2	CS2	CS2						
125	PA3	CS3	CS3	CS3	CS3						
126	PA4	CS4/TRG2	CS4	CS4	CS4						
127	PA5	CS5/PPG2	CS5	CS5	CS5						
128	PA6	CS6	CS6	CS6	CS6						
129	PA7	CS7	CS7	CS7	CS7						
132 to 139	P00 to P07	D00 to D07	P00 to P07	P00 to P07	Output Hi-Z Input ready	P : Previous state held F : Output held or Hi-Z	P : Previous state held F : Output held or Hi-Z	Output Hi-Z/ input 0 fixed	Output Hi-Z	Output Hi-Z	Output Hi-Z
142 to 144	P10 to P12	D08 to D10	P10 to P12	P10 to P12							

P : General-purpose port selected, F : Specified function selected

* : The following port's function can be used on only MB91302A and MB91V301A; SDA0, SCL0, SDA1, SCL1 of 68 to 71 pin, ICU0 to ICU3, FRCK of 81 to 85 pin.

Note : The bus width is determined after a mode vector fetch.

The bus width at initialization time is 8 bits.

APPENDIX C PIN STATE IN EACH CPU STATE

Table C-4 Pin States in Single Chip Mode (1 / 2)

Pin no.	Port name	Specified function name	At initialization ($\overline{\text{INIT}}$)		Sleep mode	Stop mode	
			Function name	Initial value		$\text{HIZ}=0$	$\text{HIZ}=1$
			Bus width 8 bits	Internal ROM mode vector (MD2-0=000)			
1 to 5	P13 to P17	-	P13 to P17				
8 to 15	P20 to P27	-	P20 to P27				
18 to 25	-	-	-				
28	P80	-	P80				
29	P81	-	P81				
30	P82	-	P82				
31	P83	-	P83				
32	P84	-	P84				
33	P85	-	P85				
34	P86	-	P86				
35	P87	-	P87				
36	P90	-	P90				
37	P91	-	P91				
38	P92	-	P92				
39	P93	-	P93				
40	P94	$\overline{\text{SRAS}}$	P94				
41	P95	$\overline{\text{SCAS/BAA}}$	P95				
42	P96	$\overline{\text{SWE/WR}}$	P96				
45 to 52	P40 to P47	-	P40 to P47				
55 to 62	P50 to P57	-	P50 to P57				
64 to 67	P60 to P63	-	P60 to P63				
68	P64	SDA0*	P64				
69	P65	SCL0*	P65				
70	P66	SDA1*	P66				
71	P67	SCL1*	P67				
76 to 79	-	AN0 to AN3	AN0 to AN3	Input invalid		Input invalid	Input invalid

Table C-4 Pin States in Single Chip Mode (2 / 2)

Pin no.	Port name	Specified function name	At initialization (INIT)		Sleep mode	Stop mode	
			Function name	Initial value		HIZ=0	HIZ=1
			Bus width 8 bits	Internal ROM mode vector (MD2-0=000)			
81	PG0	INT0/ICU0*	PG0				
82	PG1	INT1/ICU1*	PG1				
83	PG2	INT2/ICU2*	PG2				
84	PG3	INT3/ICU3*	PG3				
85	PG4	INT4/ATG/FRCK*	PG4				
86	PG5	INT5/SIN2	PG5				
87	PG6	INT6/SOT2	PG6				
88	PG7	INT7/SCK2	PG7				
90	PJ0	SIN0	PJ0				
91	PJ1	SOT0	PJ1				
92	PJ2	SCK0	PJ2				
93	PJ3	SIN1	PJ3				
94	PJ4	SOT1	PJ4				
95	PJ5	SCK1	PJ5				
96	PJ6	PPG0	PJ6				
97	PJ7	TRG0	PJ7				
98	PH0	TIN0	PH0				
99	PH1	TIN1/PPG3	PH1				
100	PH2	TIN2/TRG3	PH2				
103	PB0	-	PB0				
104	PB1	-	PB1				
105	PB2	-	PB2				
106	PB3	-	PB3				
107	PB4	TRG1	PB4				
108	PB5	PPG1	PB5				
109	PB6	-	PB6				
110	PB7	-	PB7				
122	PA0	-	PA0				
123	PA1	-	PA1				
124	PA2	-	PA2				
125	PA3	-	PA3				
126	PA4	TRG2	PA4				
127	PA5	PPG2	PA5				
128	PA6	-	PA6				
129	PA7	-	PA7				
132 to 139	P00 to P07	-	P00 to P07				
142 to 144	P10 to P12	-	P10 to P12				

P : General-purpose port selected, F : Specified function selected

* : The following port's function can be used on only MB91302A and MB91V301A; SDA0, SCL0, SDA1, SCL1 of 68 to 71 pin, ICU0 to ICU3, FRCK of 81 to 85 pin.

Note : The bus width is determined after a mode vector fetch.

The bus width at initialization time is 8 bits.

APPENDIX D NOTES ON USING A LITTLE ENDIAN AREA

This section provides notes on the use of a little endian area classified with the following items:

- D.1 C Compiler (fcc911)
- D.2 Assembler (fasm911)
- D.3 Linker (flnk911)
- D.4 Debugger (sim911, eml911, mon911)

D.1 C Compiler (fcc911)

Note that when programming is done in the C language, behavior cannot be guaranteed if the following operations are performed for a little endian area:

- Allocation of a variable with an initial value
 - Structure assignment
 - Operations other than string arrangement using a string manipulation function
 - Specification of the -K lib option when a string manipulation function is used
 - Use of the double type or long double type
 - Allocation of a stack to a little endian area
-

■ Allocation of a Variable with an Initial Value

Allocation of a variable with an initial value to a little endian area is not allowed.

No compiler has a function that generates the initial value of a little endian area. Although it is possible to allocate a variable to a little endian area, an initial value cannot be set.

Include processing at the beginning of a program that sets an initial value.

[Example] Setting an initial value for the variable little_data in a little endian area

```
extern int little_data;

void little_init(void){
    little_data = Initial value;
}

void main(void)
    little_init();
    ...
}
```

■ Structure Assignment

When a structure is assigned to another structure, the compiler selects the optimum transfer method (byte, halfword, or word). Thus, if structure assignment is performed between a structure variable allocated to an ordinary area and a structure variable allocated to a little endian area, a correct result cannot be obtained.

It is therefore necessary to assign each member in the structure.

APPENDIX D NOTES ON USING A LITTLE ENDIAN AREA

[Example] Assigning a structure to the structure variable little_st in a little endian area

```
struct tag { char c; int i; } normal_st;
extern struct tag little_st;

#define STRMOVE(DEST,SRC) DEST.c=SRC.c;DEST.i=SRC.i;

void main(void) {
    STRMOVE(little_st,normal_st);
}
```

Since the allocation of the members of a structure is different from compiler to compiler, the allocation of members by one compiler will be different from the allocation by another compiler. If the allocation method is different, it is not possible to obtain the correct result even the method described above is used.

If the allocation of members of a structure varies, do not allocate any structure variable to a little endian area.

■ Operations other than the String Arrangement Using a String Manipulation Function

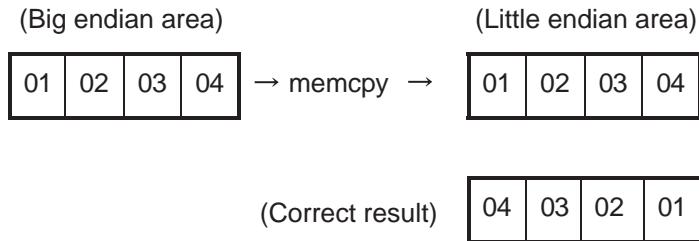
Since string manipulation functions provided as a standard library perform their processing in bytes, correct results cannot be obtained if processing using a string manipulation function is performed in an area with a type other than the char type, unsigned char type, or signed char type allocated within a little endian area.

Do not perform processing such as that described above.

[Example of incorrect coding] Transfer of word data using memcpy

```
int big = 0X01020304; /* Big endian area */
extern int little;      /* Little endian area */
memcpy(&little,&big,4); /* Transfer using memcpy */
```

The result of the above code is shown below, and, as the result of transferring word data, is an error.



■ Specification of the -K lib Option When Using a String Manipulation Function

If the -K lib option is specified, the compiler performs inline expansion for some of the string manipulation functions. At this point, processing may be changed to processing using halfwords or words as a way to select the optimum processing.

If processing is changed in this manner, processing on a little endian area will not be performed correctly.

Do not specify the -K lib option when performing processing for a little endian area that uses a string manipulation function.

Also, do not specify the -O4 option and -K speed option, each of which includes the -K lib option.

■ Use of the Double Type or Long Double Type

Access to double type or long double type data is performed by accessing one high-order word or one low-order word. Thus, when a double type or long double type variable allocated to a little endian area is accessed, correct results cannot be obtained.

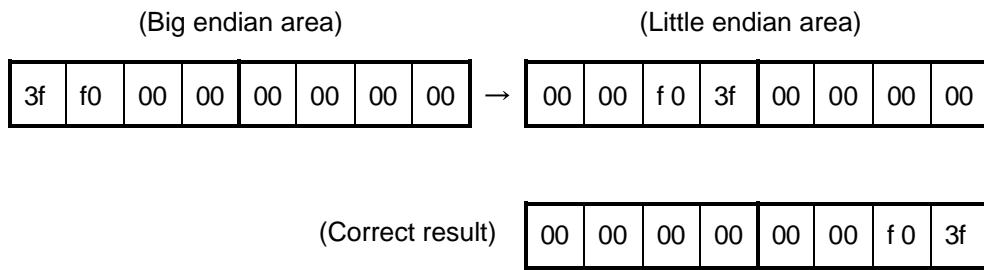
The assignment of variables of the same type allocated to a little endian area can be done, but as a result of optimization, the assignment of variables may be replaced by the assignment of constants.

Do not allocate double type and long double type variables to a little endian area.

[Example of incorrect coding] Transfer of data of the double type

```
double big = 1.0; /* Big endian area */  
extern int little; /* Little endian area */  
little = big;; /* Transfer of double type data */
```

The result of the above code is shown below, and as the result of transferring double type data, is an error.



■ Allocation of a Stack to a Little Endian Area

If some or all of the stacks are allocated to a little endian area, behavior cannot be guaranteed.

D.2 Assembler (fasm911)

The following items regarding little endian areas need to be noted when the FR family assembly language is used for programming:

- Section
 - Data access
-

■ Section

A little endian area is primarily intended to be used for data exchange with CPUs with little endian lines. Consequently, define a little endian area as a data section without an initial value.

If a code, stack, or data section with an initial value is specified in a little endian area, the MB91301 series access operation cannot be guaranteed.

[Example]

```
/* Section definition of a correct little endian area */
.SECTION Little_Area, DATA, ALIGN=4

Little_Word:
    .RES.W 1

Little_Half:
    .RES.H 1

Little_Byte:
    .RES.B 1
```

■ Data Access

When data in a little endian area is accessed, the data values can be coded without awareness of the endian method used. However, access to data in a little endian area must be performed using the same size as the data size.

[Example]

```
LDI      #0X01020304, r0
LDI      #Little_Word, r1

LDI      #0X0102, r2
LDI      #Little_Half, r3

LDI      #0X01, r4
LDI      #Little_Byte, r5

/* Access 32-bit data using the ST instruction (or the LD instruction). */
ST       r0, @r1

/* Access 16-bit data using the STH instruction (or the LDH instruction). */
STH     r2, @r3

/* Access 8-bit data using the STB instruction (or the LDB instruction). */
STB     r4, @r5
```

If data is accessed with the MB91301 series using a size that is different from the data size, the data values cannot be guaranteed. For example, if two consecutive 16-bit data items are accessed using a 32-bit access instruction, the data value cannot be guaranteed.

D.3 Linker (fInk911)

The following items related to section allocation for linking need to be noted when a program that uses a little endian area is used:

- **Restriction on section types**
 - **Lack of error detection**
-

■ **Restriction on Section Types**

Only data sections without an initial value can be allocated to a little endian area.

If a data section with an initial value, stack section, or code section is allocated to a little endian area, program operation cannot be guaranteed because arithmetic processing, such as an address resolution, is performed internally by the linker using the big endian method.

■ **Lack of Error Detection**

Since the linker does not recognize little endian areas, the linker does not issue an error message if allocation violating the above restriction is performed. Use the linker only after carefully studying the sections that will be allocated to little endian areas.

D.4 Debugger (sim911, eml911, mon911)

This section provides notes on using a simulator debugger or emulator debugger/monitor debugger.

■ Simulator Debugger

There is no memory space specification command that can indicate a little endian area.

As a result, memory management commands and instructions executed to manage memory are handled as if they were big endian.

■ Emulator Debugger/Monitor Debugger

Note that, if a little endian area is accessed using the following commands, the data values are not handled as normal values:

- **The set memory, show memory, enter, examine, and set watch commands**

When floating-point (single or double) data is processed, the specified value is neither set nor displayed.

- **The search memory command**

When halfword or word data is searched, the specified value is not used in the search.

- **Line assembly and disassembly (including the disassembly display in the source window)**

Normal instruction codes can neither be set nor displayed.

Do not allocate any instruction codes to a little endian area.

- **The call and show call commands**

If a stack area is placed in a little endian area, normal operation cannot be expected.

Do not allocate a stack area to a little endian area.

APPENDIX E INSTRUCTION LISTS

This section provides lists of the FR family instructions.

E.1 How to Read the Instruction Lists

E.2 FR Family Instruction Lists

E.1 How to Read the Instruction Lists

Before the lists are presented, the following items are explained to make the lists easier to understand:

- How to read the instruction lists
 - Addressing mode symbols
 - Instruction format
-

■ How to Read the Instruction Lists

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
ADD Rj, Rj	A	AG	1	CCCC	Ri + Rj --> Rj	-
*ADD #s5, Rj	C	A4	1	CCCC	Ri + s5 --> Ri	
,	,	,	,	,	,	
,	,	,	,	,	,	

1. 2. 3. 4. 5. 6. 7.

1. Instruction name.
 - An asterisk (*) indicates an extended instruction that is not contained in the CPU specifications and is obtained by extension of or addition to the assembler.
2. Symbols indicating addressing modes that can be specified for the operand.
 - For the meaning of symbols, see "Addressing Mode Symbols".
3. Instruction format.
4. Instruction code in hexadecimal notation.
5. Number of machine cycles.
 - a: Memory access cycle that may be extended by the Ready function.
 - b: Memory access cycle that may be extended by the Ready function. However, the cycle is interlocked if a direct instruction references a register intended for an LD operation, increasing the number of execution cycles by 1.
 - c: Interlocked if the direct instruction is an instruction that reads or writes to R15, SSP, or USP, or an instruction in instruction format A. The number of execution cycles increases by 1 and so it becomes 2.
 - d: Interlocked if the direct instruction references MDH/MDL. The number of execution cycles increases to 2.
However, if "ST Rs,@R15" instruction accesses to special registers (TBR, RP, USP, SSP, MDH, MDL) immediately after DIV1 instruction, the cycle always be interlocked and the number of execution cycles increases to 2.
 - The minimum for a, b, c, and d is 1 cycle.

APPENDIX E INSTRUCTION LISTS

6. Indicates a flag change.

Flag change	Flag meaning
C: Change -: No change 0: Clear 1: Set	N: Negative flag Z: Zero flag V: Overflow flag C: Carry flag

7. Instruction operation.

■ Addressing Mode Symbols

Table E.1-1 Explanation of Addressing Mode Symbols (1 / 2)

Symbol	Meaning
Ri	Register direct (R0 to R15, AC, FP, SP)
Rj	Register direct (R0 to R15, AC, FP, SP)
R13	Register direct (R13, AC)
Ps	Register direct (program status register)
Rs	Register direct (TBR, RP, SSP, USP, MDH, MDL)
CRi	Register direct (CR0 to CR15)
CRj	Register direct (CR0 to CR15)
#i8	Unsigned 8-bit immediate (-128 to 255) Note: -128 to -1 is handled as 128 to 255.
#i20	Unsigned 20-bit immediate (-80000_H to $FFFFF_H$) Note: $-7FFFF_H$ to -1 is handled as $7FFFF_H$ to $FFFFF_H$.
#i32	Unsigned 32-bit immediate (-80000000_H to $FFFFFFF_H$) Note: -80000000_H to -1 is handled as 80000000_H to $FFFFFFF_H$.
#s5	Signed 5-bit immediate (-16 to 15)
#s10	Signed 10-bit immediate (-512 to 508, multiples of 4 only)
#u4	Unsigned 4-bit immediate (0 to 15)
#u5	Unsigned 5-bit immediate (0 to 31)
#u8	Unsigned 8-bit immediate (0 to 255)
#u10	Unsigned 10-bit immediate (0 to 1020, multiples of 4 only)
@dir8	Unsigned 8-bit direct address (0_H to FF_H)
@dir9	Unsigned 9-bit direct address (0_H to $1FE_H$, multiple of 2 only)
@dir10	Unsigned 10-bit direct address (0_H to $3FC_H$, multiples of 4 only)
label9	Signed 9-bit branch address (-100_H to FC_H , multiples of 2 only)
label12	Signed 12-bit branch address (-800_H to $7FC_H$, multiples of 2 only)
label20	Signed 20-bit branch address (-80000_H to $7FFFH$)
label32	Signed 32-bit branch address (-80000000_H to $7FFFFFFF_H$)
@Ri	Register indirect (R0 to R15, AC, FP, SP)
@Rj	Register indirect (R0 to R15, AC, FP, SP)
@(R13,Rj)	Register relative indirect (Rj: R0 to R15, AC, FP, SP)
@(R14,disp10)	Register relative indirect (disp10: -200_H to $1FC_H$, multiples of 4 only)
@(R14,disp9)	Register relative indirect (disp9: -100_H to FE_H , multiples of 2 only)

APPENDIX E INSTRUCTION LISTS

Table E.1-1 Explanation of Addressing Mode Symbols (2 / 2)

Symbol	Meaning
@(R14,disp8)	Register relative indirect (disp8: -80 _H to 7F _H)
@(R15,udisp6)	Register relative indirect (udisp6: 0 to 60, multiples of 4 only)
@Ri+	Register indirect with post-increment (R0 to R15, AC, FP, SP)
@R13+	Register indirect with post-increment (R13, AC)
@SP+	Stack pop
@-SP	Stack push
(reglist)	Register list

■ Instruction Format

Table E.1-2 Instruction Format

Type	Instruction format						
A	<p style="text-align: center;">MSB LSB</p> <p style="text-align: center;">16 bits</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>Rj</td> <td>Ri</td> </tr> <tr> <td>8</td> <td>4</td> <td>4</td> </tr> </table>	OP	Rj	Ri	8	4	4
OP	Rj	Ri					
8	4	4					
B	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>i8/o8</td> <td>Ri</td> </tr> <tr> <td>4</td> <td>8</td> <td>4</td> </tr> </table>	OP	i8/o8	Ri	4	8	4
OP	i8/o8	Ri					
4	8	4					
C	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>u4/m4</td> <td>Ri</td> </tr> <tr> <td>8</td> <td>4</td> <td>4</td> </tr> </table>	OP	u4/m4	Ri	8	4	4
OP	u4/m4	Ri					
8	4	4					
*C'	<p style="text-align: center;">ADD, ADDN, CMP, LSL, LSR, ASR instructions only</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>s5/u5</td> <td>Ri</td> </tr> <tr> <td>7</td> <td>5</td> <td>4</td> </tr> </table>	OP	s5/u5	Ri	7	5	4
OP	s5/u5	Ri					
7	5	4					
D	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>u8/rel8/dir/ reglist</td> </tr> <tr> <td>8</td> <td>8</td> </tr> </table>	OP	u8/rel8/dir/ reglist	8	8		
OP	u8/rel8/dir/ reglist						
8	8						
E	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>SUB-OP</td> <td>Ri</td> </tr> <tr> <td>8</td> <td>4</td> <td>4</td> </tr> </table>	OP	SUB-OP	Ri	8	4	4
OP	SUB-OP	Ri					
8	4	4					
F	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>OP</td> <td>rel11</td> </tr> <tr> <td>5</td> <td>11</td> </tr> </table>	OP	rel11	5	11		
OP	rel11						
5	11						

E.2 FR Family Instruction Lists

The FR family instruction lists are presented in the order listed below.

■ FR Family Instruction Lists

- Table E.2-1 "Add-Subtract Instructions"
- Table E.2-2 "Compare Instructions"
- Table E.2-3 "Logic Instructions"
- Table E.2-4 "Bit Manipulation Instructions"
- Table E.2-5 "Multiply Instructions"
- Table E.2-6 "Shift Instructions"
- Table E.2-7 "Immediate Set/16-bit/32-bit Immediate Transfer Instructions"
- Table E.2-8 "Memory Load Instructions"
- Table E.2-9 "Memory Store Instructions"
- Table E.2-10 "Register-to-Register Transfer Instructions"
- Table E.2-11 "Normal Branch (No Delay) Instructions"
- Table E.2-12 "Delayed Branch Instructions"
- Table E.2-13 "Other Instructions"
- Table E.2-14 "20-bit Normal Branch Macro Instructions"
- Table E.2-15 "20-bit Delayed Branch Macro Instructions"
- Table E.2-16 "32-bit Normal Branch Macro Instructions"
- Table E.2-17 "32-bit Delayed Branch Macro Instructions"
- Table E.2-18 "Direct Addressing Instructions"
- Table E.2-19 "Resource Instructions"
- Table E.2-20 "Coprocessor Control Instructions"

■ Add-Subtract Instructions

Table E.2-1 Add-Subtract Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
ADD Rj, Ri	A	A6	1	CCCC	Ri + Rj --> Ri	
*ADD #s5, Ri	C'	A4	1	CCCC	Ri + s5 --> Ri	The assembler treats the highest-order bit as the sign.
ADD #u4, Ri	C	A4	1	CCCC	Ri + extu(i4) --> Ri	Zero extension
ADD2 #u4, Ri	C	A5	1	CCCC	Ri + extu(i4) --> Ri	Minus extension
ADDC Rj, Ri	A	A7	1	CCCC	Ri + Rj + c --> Ri	Addition with carry
ADDN Rj, Ri	A	A2	1	----	Ri + Rj --> Ri	
*ADDN #s5, Ri	C'	A0	1	----	Ri + s5 --> Ri	The assembler treats the highest-order bit as the sign.
ADDN #u4, Ri	C	A0	1	----	Ri + extu(i4) --> Ri	Zero extension
ADDN2 #u4, Ri	C	A1	1	----	Ri + extu(i4) --> Ri	Minus extension
SUB Rj, Ri	A	AC	1	CCCC	Ri - Rj --> Ri	
SUBC Rj, Ri	A	AD	1	CCCC	Ri - Rj - c --> Ri	Addition with carry
SUBN Rj, Ri	A	AE	1	----	Ri - Rj --> Ri	

■ Compare Instructions

Table E.2-2 Compare Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
CMP Rj, Ri	A	AA	1	CCCC	Ri - Rj	
*CMP #s5, Ri	C'	A8	1	CCCC	Ri - s5	The assembler treats the highest-order bit as the sign.
CMP #u4, Ri	C	A8	1	CCCC	Ri - extu(i4)	Zero extension
CMP2 #u4, Ri	C	A9	1	CCCC	Ri - extu(i4)	Minus extension

APPENDIX E INSTRUCTION LISTS

■ Logic Instructions

Table E.2-3 Logic Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMV	Remarks
AND Rj, Ri	A	82	1	CC--	Ri &= Rj	-	Word
AND Rj, @Ri*	A	84	1+2a	CC--	(Ri) &= Rj	○	Word
ANDH Rj, @Ri*	A	85	1+2a	CC--	(Ri) &= Rj	○	Halfword
ANDB Rj, @Ri*	A	86	1+2a	CC--	(Ri) &= Rj	○	Byte
OR Rj, Ri	A	92	1	CC--	Ri = Rj	-	Word
OR Rj, @Ri*	A	94	1+2a	CC--	(Ri) = Rj	○	Word
ORH Rj, @Ri*	A	95	1+2a	CC--	(Ri) = Rj	○	Halfword
ORB Rj, @Ri*	A	96	1+2a	CC--	(Ri) = Rj	○	Byte
EOR Rj, Ri	A	9A	1	CC--	Ri ^ = Rj	-	Word
EOR Rj, @Ri*	A	9C	1+2a	CC--	(Ri) ^ = Rj	○	Word
EORH Rj, @Ri*	A	9D	1+2a	CC--	(Ri) ^ = Rj	○	Halfword
EORB Rj, @Ri*	A	9E	1+2a	CC--	(Ri) ^ = Rj	○	Byte

*: To code these instructions in the assembler, set Rj to a general-purpose register other than R15.

■ Bit Manipulation Instructions

Table E.2-4 Bit Manipulation Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMV	Remarks
BANDL #u4, @Ri	C	80	1+2a	----	(Ri)&=(0xF0+u4)	○	Low-order 4 bits are manipulated.
BANDH #u4, @Ri	C	81	1+2a	----	(Ri)&=((u4<<4)+0x0F)	○	High-order 4 bits are manipulated.
BAND #u8, @Ri ^{*1}				----	(Ri)&=u8	-	
BORL #u4, @Ri	C	90	1+2a	----	(Ri) = u4	○	Low-order 4 bits are manipulated.
BORLH #u4, @Ri	C	91	1+2a	----	(Ri) = (u4<<4)	○	High-order 4 bits are manipulated.
BOR #u8, @Ri ^{*2}				----	(Ri) = u8	-	
BEORL #u4, @Ri	C	98	1+2a	----	(Ri) ^ = u4	○	Low-order 4 bits are manipulated.
BEORH #u4, @Ri	C	99	1+2a	----	(Ri) ^ = (u4<<4)	○	High-order 4 bits are manipulated.
BEOR #u8, @Ri ^{*3}				----	(Ri) ^ = u8	-	
BTSTL #u4, @Ri	C	88	2+a	0C--	(Ri) & u4	-	Low-order 4 bits are tested.
BTSTH #u4, @Ri	C	89	2+a	CC--	(Ri) & (u4<<4)	-	High-order 4 bits are tested.

*1: The assembler generates BANDL if the bit is set at $u8\&0F_H$, and BANDH if the bit is set at $u8\&F0_H$.
In some cases, both BANDL and BANDH may be generated.

*2: The assembler generates BORL if the bit is set at $u8\&0F_H$, and BORH if the bit is set at $u8\&F0_H$.
In some cases, both BORL and BORH are generated.

*3: The assembler generates BEORL if the bit is set at $u8\&0F_H$, and BEORH if the bit is set at $u8\&F0_H$.
In some cases, both BEORL and BEORH are generated.

APPENDIX E INSTRUCTION LISTS

■ Multiply Instructions

Table E.2-5 Multiply Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
MUL Rj,Ri	A	AF	5	CCC-	Ri × Rj --> MDH,MDL	32bits × 32bits=64bits
MULU Rj,Ri	A	AB	5	CCC-	Ri × Rj --> MDH,MDL	No sign
MULH Rj,Ri	A	BF	3	CC--	Ri × Rj --> MDL	16bits × 16bits=32bits
MULUH Rj,Ri	A	BB	3	CC--	Ri × Rj --> MDL	No sign
DIV0S Ri	E	97-4	1	----		Step operation
DIV0U Ri	E	97-5	1	----		32bits/32bits=32bits
DIV1 Ri	E	97-6	d	-C-C		
DIV2 Ri	E	97-7	1	-C-C		
DIV3	E	9F-6	1	----		
DIV4S	E	9F-7	1	----		
DIV Ri ^{*1}			36	-C-C	MDL / Ri --> MDL, MDL % Ri --> MDH	
DIVU Ri ^{*2}			33	-C-C	MDL / Ri --> MDL, MDL % Ri --> MDH	

*1: DIV0S, DIV1 x 32, DIV2, DIV3, or DIV4S is generated. The instruction code length becomes 72 bytes.

*2: DIV0U or DIV1 x 32 is generated. The instruction code length becomes 66 bytes.

■ Shift Instructions

Table E.2-6 Shift Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LSL Rj, Ri	A	B6	1	CC-C	Ri << Rj --> Ri	Logical shift
LSL #u5, Ri (u5:0 to 31)	C'	B4	1	CC-C	Ri << u5 --> Ri	
LSL #u4, Ri	C	B4	1	CC-C	Ri << u4 --> Ri	
LSL2 #u4, Ri	C	B5	1	CC-C	Ri <<(u4+16) --> Ri	
LSR Rj, Ri	A	B2	1	CC-C	Ri >> Rj --> Ri	Logical shift
LSR #u5, Ri (u5:0 to 31)	C'	B0	1	CC-C	Ri >> u5 --> Ri	
LSR #u4, Ri	C	B0	1	CC-C	Ri >> u4 --> Ri	
LSR2 #u4, Ri	C	B1	1	CC-C	Ri >>(u4+16) --> Ri	
ASR Rj, Ri	A	BA	1	CC-C	Ri >> Rj --> Ri	Arithmetic shift
ASR #u5, Ri (u5:0 to 31)	C'	B8	1	CC-C	Ri >> u5 --> Ri	
ASR #u4, Ri	C	B8	1	CC-C	Ri >> u4 --> Ri	
ASR2 #u4, Ri	C	B9	1	CC-C	Ri >>(u4+16) --> Ri	

■ Immediate Set/16-bit/32-bit Immediate Transfer Instructions

Table E.2-7 Immediate Set/16-bit/32-bit Immediate Transfer Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LDI:32 #i32, Ri	E	9F-8	3	----	i32 --> Ri	
LDI:20 #i20, Ri	C	9B	2	----	i20 --> Ri	High-order 12 bits are zero-extended.
LDI:8 #i8, Ri	B	C0	1	----	i8 --> Ri	High-order 24 bits are zero-extended.
LDI # {i8 i20 i32} ,Ri					{i8 i20 i32} --> Ri	

*: If the immediate data is represented as absolute values, the assembler selects automatically from i8, i20, and i32.

If immediate data contains a relative value or external reference symbol, i32 is selected.

APPENDIX E INSTRUCTION LISTS

■ Memory Load Instructions

Table E.2-8 Memory Load Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LD @Rj, Ri	A	04	b	----	(Rj) --> Ri	
LD @(R13,Rj), Ri	A	00	b	----	(R13+Rj) --> Ri	
LD @(R14,disp10), Ri	B	20	b	----	(R14+disp10) --> Ri	
LD @(R15,udisp6), Ri	C	03	b	----	(R15+udisp6) --> Ri	
LD @R15+, Ri	E	07-0	b	----	(R15) --> Ri, R15+=4	
LD @R15+, Rs	E	07-8	b	----	(R15) --> Rs, R15+=4	Rs: Special register*
LD @R15+, PS	E	07-9	1+a+b	CCCC	(R15) --> PS, R15+=4	
LDUH @Rj, Ri	A	05	b	----	(Rj) --> Ri	Zero extension
LDUH @(R13,Rj), Ri	A	01	b	----	(R13+Rj) --> Ri	Zero extension
LDUH @(R14,disp9), Ri	B	40	b	----	(R14+disp9) --> Ri	Zero extension
LDUB @Rj, Ri	A	06	b	----	(Rj) --> Ri	Zero extension
LDUB @(R13,Rj), Ri	A	02	b	----	(R13+Rj) --> Ri	Zero extension
LDUB @(R14,disp8), Ri	B	60	b	----	(R14+disp8) --> Ri	Zero extension

*: Special register Rs: TBR, RP, USP, SSP, MDH, and MDL

Note:

In the o8 and o4 fields of the hardware specifications, the assembler calculates values and sets them as shown below:

disp10/4 --> o8, disp9/2 --> o8, disp8 --> o8; disp10, disp9, and disp8 have a sign.
udisp6/4 --> o4; udisp6 has no sign.

■ Memory Store Instructions

Table E.2-9 Memory Store Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
ST Ri, @Rj	A	14	a	----	Ri --> (Rj)	Word
ST Ri, @(R13,Rj)	A	10	a	----	Ri --> (R13+Rj)	Word
ST Ri, @(R14,disp10)	B	30	a	----	Ri --> (R14+disp10)	Word
ST Ri, @(R15,udisp6)	C	13	a	----	Ri --> (R15+udisp6)	
ST Ri, @-R15	E	17-0	a	----	R15=-4, Ri --> (R15)	
ST Rs, @-R15	E	17-8	a	----	R15=-4, Rs --> (R15)	Rs: Special register*
ST PS, @-R15	E	17-9	a	----	R15=-4, PS --> (R15)	
STH Ri, @Rj	A	15	a	----	Ri --> (Rj)	Halfword
STH Ri, @(R13,Rj)	A	11	a	----	Ri --> (R13+Rj)	Halfword
STH Ri, @(R14,disp9)	B	50	a	----	Ri --> (R14+disp9)	Halfword
STB Ri, @Rj	A	16	a	----	Ri --> (Rj)	Byte
STB Ri, @(R13,Rj)	A	12	a	----	Ri --> (R13+Rj)	Byte
STB Ri, @(R14,disp8)	B	70	a	----	Ri --> (R14+disp8)	Byte

*: Special register Rs: TBR, RP, USP, SSP, MDH, and MDL

Note:

In the o8 and o4 fields of the hardware specifications, the assembler calculates values and sets them as shown below:

disp10/4 --> o8, disp9/2 --> o8, disp8 --> o8; disp10, disp9, and disp8 have a sign.
udisp6/4 --> o4; udisp6 has no sign.

■ Register-to-Register Transfer Instructions

Table E.2-10 Register-to-Register Transfer Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
MOV Rj, Ri	A	8B	1	----	Rj --> Ri	Transfer between general-purpose registers
MOV Rs, Ri	A	B7	1	----	Rs --> Ri	Rs: Special register *
MOV Ri, Rs	E	B3	1	----	Ri --> Rs	Rs: Special register *
MOV PS, Ri	E	17-1	1	----	PS --> Ri	
MOV Ri, PS	E	07-1	c	CCCC	Ri --> PS	

*: Special register Rs: TBR, RP, USP, SSP, MDH, and MDL

APPENDIX E INSTRUCTION LISTS

■ Normal Branch (No Delay) Instructions

Table E.2-11 Normal Branch (No Delay) Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
JMP @Ri	E	97-0	2	----	Ri --> PC	
CALL label12	F	D0	2	----	PC+2-->RP , PC+2+(label12-PC-2)-->PC	
CALL @Ri	E	97-1	2	----	PC+2-->RP , Ri-->PC	
RET	E	97-2	2	----	RP --> PC	Return
INT #u8	D	AC	3+3a	----	SSP-=4,PS --> (SSP), SSP-=4,PC+2 --> (SSP), 0--> I flag,0 --> S flag, (TBR+0x3FC-u8x4) --> PC	
INTE	E	9F-3	3+3a	----	SSP-=4,PS --> (SSP), SSP-=4,PC+2 --> (SSP), 0 --> S flag, (TBR+0x3D8) --> PC	For emulator
RETI	E	97-3	2+2A	CCCC	(R15) --> PC,R15-=4, (R15) --> PS,R15-=4	
BRA label9	D	E0	2	----	PC+2+(label9-PC-2) -->PC	
BNO label9	D	E1	1	----	No branch	
BEQ label9	D	E2	2/1	----	if(Z==1) then PC+2+(label9-PC-2) -->PC	
BNE label9	D	E3	2/1	----	↑ s/Z==0	
BC label9	D	E4	2/1	----	↑ s/C==1	
BNC label9	D	E5	2/1	----	↑ s/C==0	
BN label9	D	E6	2/1	----	↑ s/N==1	
BP label9	D	E7	2/1	----	↑ s/N==0	
BV label9	D	E8	2/1	----	↑ s/V==1	
BNV label9	D	E9	2/1	----	↑ s/V==0	
BLT label9	D	EA	2/1	----	↑ s/V xor N==1	
BGE label9	D	EB	2/1	----	↑ s/V xor N==0	
BLE label9	D	EC	2/1	----	↑ s/(V xor N) or Z==1	
BGT label9	D	ED	2/1	----	↑ s/(V xor N) or Z==0	
BLS label9	D	EE	2/1	----	↑ s/C or Z==1	
BHI label9	D	EF	2/1	----	↑ s/C or Z==0	

Notes:

- "2/1" under CYCLE indicates 2 when branching occurs and 1 when branching does not occur.
 - In the rel11 and rel8 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
 $(\text{label12-PC-2})/2 \rightarrow \text{rel11}$, $(\text{label9-PC-2})/2 \rightarrow \text{rel8}$; label12 and label9 have a sign.
 - To execute the RETI instruction, the S flag must be 0.
-

APPENDIX E INSTRUCTION LISTS

■ Delayed Branch Instructions

Table E.2-12 Delayed Branch Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
JMP:D @Ri	E	9F-0	1	----	Ri --> PC	
CALL:D label12	F	D8	1	----	PC+4 --> RP , PC+2+(label12-PC-2) --> PC	
CALL:D @Ri	E	9F-1	1	----	PC+4 --> RP ,Ri --> PC	
RET:D	E	9F-2	1	----	RP --> PC	Return
BRA:D label9	D	F0	1	----	PC+2+(label9-PC-2) --> PC	
BNO:D label9	D	F1	1	----	No branch	
BEQ:D label9	D	F2	1	----	if(Z==1) then PC+2+(label9-PC-2) --> PC	
BNE:D label9	D	F3	1	----	↑ s/Z==0	
BC:D label9	D	F4	1	----	↑ s/C==1	
BNC:D label9	D	F5	1	----	↑ s/C==0	
BN:D label9	D	F6	1	----	↑ s/N==1	
BP:D label9	D	F7	1	----	↑ s/N==0	
BV:D label9	D	F8	1	----	↑ s/V==1	
BNV:D label9	D	F9	1	----	↑ s/V==0	
BLT:D label9	D	FA	1	----	↑ s/V xor N==1	
BGE:D label9	D	FB	1	----	↑ s/V xor N==0	
BLE:D label9	D	FC	1	----	↑ s/(V xor N) or Z==1	
BGT:D label9	D	FD	1	----	↑ s/(V xor N) or Z==0	
BLS:D label9	D	FE	1	----	↑ s/C or Z==1	
BHI:D label9	D	FF	1	----	↑ s/C or Z==0	

Notes:

- In the rel11 and rel8 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
(label12-PC-2)/2 --> rel11, (label9-PC-2)/2 --> rel8; label12 and label9 have a sign.
 - A delayed branch always occurs after the next instruction (delay slot) is executed.
 - Instructions that can be placed in the delay slot are all 1-cycle, a-, b-, c-, and d-cycle instructions.
Multicycle instructions cannot be placed in the delay slot.
-

■ Other Instructions

Table E.2-13 Other Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMV	Remarks
NOP	E	9F-A	1	----	No change	-	
ANDCCR #u8	D	83	c	CCCC	CCR and u8 --> CCR	-	
ORCCR #u8	D	93	c	CCCC	CCR or u8 --> CCR	-	
STILM #u8	D	87	1	----	i8 --> ILM	-	ILM immediate set
ADDSP #s10 ^{*1}	D	A3	1	----	R15 += s10	-	ADD SP instruction
EXTSB Ri	E	97-8	1	----	Sign extension 8 --> 32bit	-	
EXTUB Ri	E	97-9	1	----	Zero extension 8 --> 32bit	-	
EXTSH Ri	E	97-A	1	----	Sign extension 16 --> 32bit	-	
EXTUH Ri	E	97-B	1	----	Zero extension 16 --> 32bit	-	
LDM0 (reglist)	D	8C		----	(R15) --> reglist, R15 increment	-	Load multi R0-R7
LDM1 (reglist)	D	8D		----	(R15) --> reglist, R15 increment	-	Load multi R8-R15
LDM (reglist) ^{*2}				----	(R15) --> reglist, R15 increment	-	Load multi R0-R15
STM0 (reglist)	D	8E		----	R15 decrement, reglist --> (R15)	-	Store multi R0-R7
STM1 (reglist)	D	8F		----	R15 decrement, reglist --> (R15)	-	Store multi R8-R15
STM (reglist) ^{*3}				----	R15 decrement, reglist --> (R15)	-	Store multi R0-R15
ENTER #u10 ^{*4}	D	0F	1+a	----	R14 --> (R15 - 4), R15 - 4 --> R14, R15 - u10 --> R15	-	Entry processing of a function
LEAVE	E	9F-9	b	----	R14 + 4 --> R15, (R15 - 4) --> R14	-	Exit processing of a function
XCHB @Rj, Ri ^{*5}	A	8A	2a	----	Ri --> TEMP (Rj) --> Ri TEMP --> (Rj)	○	For semaphore management Byte data

*1: For s10, the assembler calculates s10/4 and then changes to s8 to set a value. s10 has a sign.

*2: If any of R0 to R7 is specified in reglist, LDM0 is generated. If any of R8 to R15 is generated, LDM1 is generated. In some cases, both LDM0 and LDM1 are generated.

*3: If any of R0 to R7 is specified in reglist, STM0 is generated. If any of R8 to R15 is generated, STM1 is generated. In some cases, both STM0 and STM1 are generated.

*4: For u10, the assembler calculates u10/4 and then changes to u8 to set a value. u10 has a sign.

*5: To code this instruction in the assembler, set Ri to a general-purpose register other than R15.

Notes:

- The number of execution cycles of LDM0(reglist) and LDM1(reglist) can be calculated as $a*(n-1)+b+1$ cycles if the number of specified registers is n.
 - The number of execution cycles of STM0(reglist) and STM1(reglist) can be calculated as $a*n+1$ cycles if the number of specified registers is n.
-

■ 20-bit Normal Branch Macro Instructions

Table E.2-14 20-bit Normal Branch Macro Instructions

Mnemonic	Operation	Remarks
*CALL20 label20,Ri	Address of the next instruction --> RP, label20 --> PC	Ri: Temporary register (See Reference 1)
*BRA20 label20,Ri	label20 --> PC	Ri: Temporary register (See Reference 2)
*BEQ20 label20,Ri	if(Z==1) then label20 --> PC	Ri: Temporary register (See Reference 3)
*BNE20 label20,Ri	↑ s/Z==0	↑
*BC20 label20,Ri	↑ s/C==1	↑
*BNC20 label20,Ri	↑ s/C==0	↑
*BN20 label20,Ri	↑ s/N==1	↑
*BP20 label20,Ri	↑ s/N==0	↑
*BV20 label20,Ri	↑ s/V==1	↑
*BNV20 label20,Ri	↑ s/V==0	↑
*BLT20 label20,Ri	↑ s/V xor N==1	↑
*BGE20 label20,Ri	↑ s/V xor N==0	↑
*BLE20 label20,Ri	↑ s/(V xor N) or Z==1	↑
*BGT20 label20,Ri	↑ s/(V xor N) or Z==0	↑
*BLS20 label20,Ri	↑ s/C or Z==1	↑
*BHI20 label20,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL20

1) If label20-PC-2 is between -800_H and +7FE_H, create an instruction as shown below:

CALL label12

2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:20 #label20,Ri
CALL @Ri

[Reference 2] BRA20

1) If label20-PC-2 is between -100_H and +FE_H, create an instruction as shown below:

BRA label9

2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:20 #label20,Ri
JMP @Ri

[Reference 3] Bcc20

1) If label20-PC-2 is between -100_H and +FE_H, create an instruction as shown below:

Bcc label9

2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

Bxcc false xcc is the opposite condition of cc.
LDI:20 #label20,Ri
JMP @Ri

false:

APPENDIX E INSTRUCTION LISTS

■ 20-bit Delayed Branch Macro Instructions

Table E.2-15 20-bit Delayed Branch Macro Instructions

Mnemonic	Operation	Remarks
*CALL20:D label20,Ri	Address of the next instruction + 2 --> RP, label20 --> PC	Ri: Temporary register (See Reference 1)
*BRA20:D label20,Ri	label20 --> PC	Ri: Temporary register (See Reference 2)
*BEQ20:D label20,Ri	if(Z==1) then label20 --> PC	Ri: Temporary register (See Reference 3)
*BNE20:D label20,Ri	↑ s/Z==0	↑
*BC20:D label20,Ri	↑ s/C==1	↑
*BNC20:D label20,Ri	↑ s/C==0	↑
*BN20:D label20,Ri	↑ s/N==1	↑
*BP20:D label20,Ri	↑ s/N==0	↑
*BV20:D label20,Ri	↑ s/V==1	↑
*BNV20:D label20,Ri	↑ s/V==0	↑
*BLT20:D label20,Ri	↑ s/V xor N==1	↑
*BGE20:D label20,Ri	↑ s/V xor N==0	↑
*BLE20:D label20,Ri	↑ s/(V xor N) or Z==1	↑
*BGT20:D label20,Ri	↑ s/(V xor N) or Z==0	↑
*BLS20:D label20,Ri	↑ s/C or Z==1	↑
*BHI20:D label20,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL20:D

- 1) If label20-PC-2 is between -800_H and $+7FE_H$, create an instruction as shown below:
 CALL:D label12

- 2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

```
LDI:20 #label20,Ri  
CALL:D @Ri
```

[Reference 2] BRA20

- 1) If label20-PC-2 is between -100_H and $+FE_H$, create an instruction as shown below:
 BRA :D label9
- 2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

```
LDI:20 #label20,Ri  
JMP:D @Ri
```

[Reference 3] Bcc20:D

- 1) If label20-PC-2 is between -100_H and $+FE_H$, create an instruction as shown below:
 Bcc:D label9
 xcc is the opposite condition of cc.
- 2) If label20-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:
 Bxcc false
 LDI:20 #label20,Ri
 JMP:D @Ri

false:

■ 32-bit Normal Branch Macro Instructions

Table E.2-16 32-bit Normal Branch Macro Instructions

Mnemonic	Operation	Remarks
*CALL32 label32,Ri	Address of the next instruction --> RP, label32 --> PC	Ri: Temporary register (See Reference 1)
*BRA32 label32,Ri	label32 --> PC	Ri: Temporary register (See Reference 2)
*BEQ32 label32,Ri	if(Z==1) then label32 --> PC	Ri: Temporary register (See Reference 3)
*BNE32 label32,Ri	↑ s/Z==0	↑
*BC32 label32,Ri	↑ s/C==1	↑
*BNC32 label32,Ri	↑ s/C==0	↑
*BN32 label32,Ri	↑ s/N==1	↑
*BP32 label32,Ri	↑ s/N==0	↑
*BV32 label32,Ri	↑ s/V==1	↑
*BNV32 label32,Ri	↑ s/V==0	↑
*BLT32 label32,Ri	↑ s/V xor N==1	↑
*BGE32 label32,Ri	↑ s/V xor N==0	↑
*BLE32 label32,Ri	↑ s/(V xor N) or Z==1	↑
*BGT32 label32,Ri	↑ s/(V xor N) or Z==0	↑
*BLS32 label32,Ri	↑ s/C or Z==1	↑
*BHI32 label32,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL32

1) If label32-PC-2 is between -800_H and +7FE_H, create an instruction as shown below:

CALL label12

2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:32 #label32,Ri
CALL @Ri

[Reference 2] BRA32

1) If label32-PC-2 is between -100_H and +FE_H, create an instruction as shown below:

BRA label9

2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:32 #label32,Ri
JMP @Ri

[Reference 3] Bcc32

1) If label32-PC-2 is between -100_H and +FE_H, create an instruction as shown below:

Bcc label9

2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

Bxcc false xcc is the opposite condition of cc.

LDI:32 #label32,Ri

JMP @Ri

false:

APPENDIX E INSTRUCTION LISTS

■ 32-bit Delayed Branch Macro Instructions

Table E.2-17 32-bit Delayed Branch Macro Instructions

Mnemonic	Operation	Remarks
*CALL32:D label32,Ri	Address of the next instruction + 2 --> RP, label32 --> PC	Ri: Temporary register (See Reference 1)
*BRA32:D label32,Ri	label32 --> PC	Ri: Temporary register (See Reference 2)
*BEQ32:D label32,Ri	if(Z==1) then label32 --> PC	Ri: Temporary register (See Reference 3)
*BNE32:D label32,Ri	↑ s/Z==0	↑
*BC32:D label32,Ri	↑ s/C==1	↑
*BNC32:D label32,Ri	↑ s/C==0	↑
*BN32:D label32,Ri	↑ s/N==1	↑
*BP32:D label32,Ri	↑ s/N==0	↑
*BV32:D label32,Ri	↑ s/V==1	↑
*BNV32:D label32,Ri	↑ s/V==0	↑
*BLT32:D label32,Ri	↑ s/V xor N==1	↑
*BGE32:D label32,Ri	↑ s/V xor N==0	↑
*BLE32:D label32,Ri	↑ s/(V xor N) or Z==1	↑
*BGT32:D label32,Ri	↑ s/(V xor N) or Z==0	↑
*BLS32:D label32,Ri	↑ s/C or Z==1	↑
*BHI32:D label32,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL32:D

- 1) If label32-PC-2 is between -800_H and $+7FE_H$, create an instruction as shown below:
 CALL:D label12

- 2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:32 #label32,Ri
 CALL:D @Ri

[Reference 2] BRA32:D

- 1) If label32-PC-2 is between -100_H and $+FE_H$, create an instruction as shown below:
 BRA:D label9

- 2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

LDI:32 #label32,Ri
 JMP:D @Ri

[Reference 3] Bcc32:D

- 1) If label32-PC-2 is between -100_H and $+FE_H$, create an instruction as shown below:
 Bcc:D label9

- 2) If label32-PC-2 is outside the range in 1) or contains an external reference symbol, create an instruction as shown below:

Bxcc false xcc is the opposite condition of cc.
 LDI:32 #label32,Ri
 JMP:D @Ri

false:

■ Direct Addressing Instructions

Table E.2-18 Direct Addressing Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
DMOV @dir10, R13	D	08	b	----	(dir10) --> R13	Word
DMOV R13, @dir10	D	18	a	----	R13 --> (dir10)	Word
DMOV @dir10, @R13+	D	0C	2a	----	(dir10) --> (R13),R13+=4	Word
DMOV @R13+, @dir10	D	1C	2a	----	(R13) --> (dir10),R13+=4	Word
DMOV @dir10, @-R15	D	0B	2a	----	R15-=4,(R15) --> (dir10)	Word
DMOV @R15+, @dir10	D	1B	2a	----	(R15) --> (dir10),R15+=4	Word
DMOVH @dir9, R13	D	09	b	----	(dir9) --> R13	Halfword
DMOVH R13, @dir9	D	19	a	----	R13 --> (dir9)	Halfword
DMOVH @dir9, @R13+	D	0D	2a	----	(dir9) --> (R13),R13+=2	Halfword
DMOVH @R13+, @dir9	D	1D	2a	----	(R13) --> (dir9),R13+=2	Halfword
DMOVB @dir8, R13	D	0A	b	----	(dir8) --> R13	Byte
DMOVB R13, @dir8	D	1A	a	----	R13 --> (dir8)	Byte
DMOVB @dir8, @R13+	D	0E	2a	----	(dir8) --> (R13),R13++	Byte
DMOVB @R13+, @dir8	D	1E	2a	----	(R13) --> (dir8),R13++	Byte

Note:

In the dir8, dir9, and dir10 fields, the assembler calculates values and sets them as shown below:
 dir8 --> dir, dir9/2 --> dir, dir10/4 --> dir; dir8, dir9, and dir10 have no sign.

■ Resource Instructions

Table E.2-19 Resource Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LDRES @Ri+, #u4	C	BC	a	----	(Ri) --> u4 resource Ri+=4	u4: Channel number
STRES #u4, @Ri+	C	BD	a	----	u4 resource --> (Ri) Ri+=4	u4: Channel number

Note:

These instructions cannot be used for the MB91301 series as it has no resource having a channel number.

APPENDIX E INSTRUCTION LISTS

■ Coprocessor Control Instructions

Table E.2-20 Coprocessor Control Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
COPPOP #u4, #u8, CRj, CRI	E	9F-C	2+a	----	Operation instruction	
COPLD #u4, #u8, Rj, CRI	E	9F-D	1+2a	----	Rj --> CRI	
COPST #u4, #u8, CRj, Ri	E	9F-E	1+2a	----	CRj --> Ri	
COPSV #u4, #u8, CRj, Ri	E	9F-F	1+2a	----	CRj --> Ri	No error trap

Notes:

- {CRI | CRj}:= CR0 | CR1 | CR2 | CR3 | CR4 | CR5 | CR6 | CR7 | CR8 | CR9 | CR10 | CR11 | CR12 | CR13 | CR14 | CR15
u4:= Channel specified
u8:= Command specified
- Since the MB91301 series has no coprocessor, this instruction cannot be used.

INDEX

**The index follows on the next page.
This is listed in alphabetic order.**

Index

Numerics

0 Detection	0 Detection	447
0 Detection Data Register	0 Detection Data Register (BSD0)	445
1 Detection	1 Detection	447
1 Detection Data Register	1 Detection Data Register (BSD1)	445
10-bit Slave Address Mask Register	10-bit Slave Address Mask Register (ITMK0/ITMK1).....	464
10-bit Slave Address Register	10-bit Slave Address Register (ITBA0/ITBA1).....	464
16-bit Free Run Timer	Block Diagram of the 16-bit Free Run Timer	480
	Registers of 16-bit Free Run Timer	479
16-bit Input Capture	Operational Explanation.....	495
16-bit Reload Register	Bit Configuration of the 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0)	276
16-bit Reload Timer	16-bit Reload Timer Registers	271
	Overview of the 16-bit Reload Timer	270
	Precautions on Using the 16-bit Reload Timer	280
	When the 16-bit Reload Timer is Used for Activation.....	304
16-bit Timer Register	Bit Configuration of the 16-bit Timer Register (TMR:TMR2 to TMR0).....	275
1M Byte Flash	Examples of Connection for 1M Byte Flash.....	511
2-Cycle Transfer	2-Cycle Transfer (External ->I/O)	248
	2-Cycle Transfer (I/O ->External)	249
	2-Cycle Transfer (I/O ->SDRAM/FCRAM).....	250
	2-Cycle Transfer (Internal RAM ->External I/O, RAM)	247
	2-Cycle Transfer (SDRAM/FCRAM ->I/O).....	251
	Demand Transfer 2-Cycle Transfer	411
	Flow of Data during 2-Cycle Transfer	435
7-bit Slave Address Mask Register	7-bit Slave Address Mask Register (ISMK0/ISMK1).....	466
7-bit Slave Address Register	7-bit Slave Address Register (ISBA0/ISBA1)	465

A

A/D Converter	A/D Converter (Sequential Conversion Type).....	4
	A/D Converter Registers.....	348
	Notes on Using the Internal DC-DC Regulator and A/D Converter	39
	Precautions on Using the A/D Converter	358
AC Characteristics	AC Characteristics of DMAC.....	431
Access	Burst Access Operation	216
	Byte Access.....	199
	Data Access.....	72
	External Bus Access.....	187
	Halfword Access.....	197
	Program Access	72
	Word Access	196
Access Addresses	Mode Data and Reset Vector Access Addresses.....	531
ACR	Configuration of Area Configuration Registers 0 to 7 (ACR0 to ACR7)	152
ADCR	Conversion Result Register (ADCR0 to ADCR3)	355
	Data Register (ADCR)	354
ADCS	Bit Configuration of Control Status Register (ADCS)	349
	Detailed Bit of Control Status Register (ADCS)	349
Add	Add-Subtract Instructions	579
Address Error	Occurrence of an Address Error	424
Address Multiplexing Format	Address Multiplexing Format.....	231
Address Register	Address Register Specifications	416
	Features of the Address Register	416
	Function of the Address Register	416
Addressing	Direct Addressing Area	42
Addressing Mode	Addressing Mode Symbols	575
Architecture	Internal Architecture	46

Area Configuration Registers	
Configuration of Area Configuration Registers 0 to 7 (ACR0 to ACR7)	152
Area Select Registers	
Configuration of Area Select Registers 0 to 7 (ASR0 to ASR7)	150
Functions of Bits in the Area Select Registers (ASR0 to ASR7)	151
Area Wait Registers	
Configuration of the Area Wait Registers (AWR0 to AWR7)	158
ASR	
Configuration of Area Select Registers 0 to 7 (ASR0 to ASR7)	150
Functions of Bits in the Area Select Registers (ASR0 to ASR7)	151
Auto-precharge OFF Mode	
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)	167
Auto-precharge ON Mode	
Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode)	169
Auto-refresh Operation	
Auto-refresh Operation Timing	227
Auto-Wait Cycle Timing	
Auto-Wait Cycle Timing	208
AWR	
Configuration of the Area Wait Registers (AWR0 to AWR7)	158
B	
Base Clock Division Setting Register	
Base Clock Division Setting Register 0 (DIVR0).....	120
Base Clock Division Setting Register 1 (DIVR1).....	123
Basic	
Basic Timing (For Successive Accesses)	203
Basic Block Diagram	
Basic Block Diagram of the I/O Port.....	258
Baud Rate	
Calculation of Baud Rate	313
Baud Rates	
Example of Setting Baud Rates and U-TIMER Reload Values.....	382
Big Endian	
Differences between Little Endian and Big Endian	191
Bit Manipulation Instructions	
Bit Manipulation Instructions	581
Bit Ordering	
Bit Ordering	71
Bit Search Module	
Bit Search Module	4
Bit Search Module Registers.....	445
Block Diagram of the Bit Search Module	444
Block Diagram	
Basic Block Diagram of the I/O Port	258
Block Diagram	7, 109, 146, 270, 308, 331, 347, 361, 385
Block Diagram (for 1 channel).....	453
Block Diagram of One Channel of the PPG Timer	284
Block Diagram of the 16-bit Free Run Timer.....	480
Block Diagram of the Bit Search Module	444
Block Diagram of the Delayed Interrupt Module	326
Block Diagram of the Entire PPG Timer	283
Block Diagram of the External Interrupt and NMI Controller	316
Block Diagram of the Input Capture	492
Block Size	
Block Size	414
Block Transfer	
Block Transfer	432
If Another Transfer Request Occurs during Block Transfer	431
Branch	
Normal Branch (No Delay) Instructions	586
Branch Instruction	
Limitations on Branch Instruction with Delay Slot	77
Operation of Branch Instruction with Delay Slot	75
Operation of Branch Instruction without Delay Slot	78
Branch Instructions	
Branch Instructions with Delay Slot	74
Branch Instructions without Delay Slot.....	78
Delayed Branch Instructions	588
Branch Macro Instructions	
20-bit Delayed Branch Macro Instructions	592
20-bit Normal Branch Macro Instructions	591
32-bit Delayed Branch Macro Instructions	594
32-bit Normal Branch Macro Instructions	593
BSD0	
0 Detection Data Register (BSD0).....	445
BSD1	
1 Detection Data Register (BSD1).....	445
BSDC	
Change Point Detection Data Register (BSDC)	446
BSRR	
Detection Result Register (BSRR).....	446
Built-in Memory	
Built-in Memory.....	3

INDEX

Built-in Peripheral Request	
Built-in Peripheral Request	408
Burst Access	
Burst Access Operation	216
Burst Read/Write Operation	
Burst Read/Write Operation Timing	225
Burst Transfer	
Burst Transfer	433
Bus Control Register	
Bus Control Register (IBCR0/IBCR1)	456
Bus Interface	
Bus Interface	3
Ordinary Bus Interface	202
Bus Modes	
Bus Modes	140
Bus Right	
Releasing the Bus Right	252
Bus Status Register	
Bus Status Register (IBSR0/IBSR1)	454
Bus Width	
Data Bus Width	185, 193
Relationship between Data Bus Width and Control Signal	182
Byte Access	
Byte Access	199
Byte Ordering	
Byte Ordering	71
C	
Cache Enable Register	
Configuration of the Cache Enable Register (CHER)	174
Functions of Bits in the Cache Enable Register (CHER)	174
Cache Size Register	
Configuration of Cache Size Register (ISIZE)	54
Cascade Mode	
Cascade Mode	314
Change Point Detection	
Change Point Detection	448
Change Point Detection Data Register	
Change Point Detection Data Register (BSDC)	446
Channel Group	
Channel Group	427
CHER	
Configuration of the Cache Enable Register (CHER)	174
Functions of Bits in the Cache Enable Register (CHER)	174
Chip Select Area	
Example of Setting the Chip Select Area	181
Chip Select Enable Register	
Configuration of the Chip Select Enable Register (CSER)	172
Functions of Bits in the Chip Select Enable Register (CSER)	172
CLKB	
CPU Clock (CLKB)	106
CLKP	
Peripheral Clock (CLKP)	106
CLKR	
Clock Source Control Register (CLKR)	118
CLKT	
External Bus Clock (CLKT)	107
Clock	
Clock Division	108
Clock Generation Control	103
CPU Clock (CLKB)	106
External Bus Clock (CLKT)	107
External Clock	32
Internal Clock Operation	277
Peripheral Clock (CLKP)	106
Program Example of Smooth Startup and Stop of Clock	129
Selecting a Clock for the UART	373
Smooth Startup and Stop of Clock	128
Source Clock	103
Clock Control Register	
Clock Control Register (ICCR0/ICCR1)	462
Clock Disable Register	
Clock Disable Register (IDBL0/IDBL1)	467
Clock Generation	
Clock Generation Control	103
Clock Source Control Register	
Clock Source Control Register (CLKR)	118
Communication	
End of Communication	376
Start of Communication	376
Compare Instructions	
Compare Instructions	579
Configurator	
REALOS/FR Configurator Setup	523
Connection	
An Example of Connection of External Memory	532
Example of Connection with External Devices	190
Examples of Connection for 1M Byte Flash	511
Examples of Connection with External Devices	194
Continuous Conversion Mode	
Continuous Conversion Mode	357
Control Status Register	
Bit Configuration of Control Status Register (ADCS)	349

Bit Configuration of the Control Status Register (TMCSR)	272
Bit Functions of the Control Status Register (TMCSR)	272
Detailed Bit of Control Status Register (ADCS).....	349
Control Status Registers	
Bit Function of Control Status Registers (PCNH,PCNL)	286
Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0).....	286
Control/Status Register	
Bit Configuration of Control/Status Registers A (DMACA0 to DMACA4)	388
Bit Configuration of Control/Status Registers B (DMACB0 to DMACB4).....	393
Detailed Bit of Control/Status Registers A (DMACA0 to DMACA4)	388
Detailed Bit of Control/Status Registers B (DMACB0 to DMACB4).....	393
Conversion	
A/D Converter (Sequential Conversion Type).....	4
Conversion Result Register	
Conversion Result Register (ADCR0 to ADCR3).....	355
Coprocessor Control Instructions	
Coprocessor Control Instructions	596
Coprocessor Error Trap	
Coprocessor Error Trap	93
coprocessor Trap	
No-coprocessor Trap	93
CPU	
CPU Clock (CLKB)	106
FR CPU	2
CSER	
Configuration of the Chip Select Enable Register (CSER)	172
Functions of Bits in the Chip Select Enable Register (CSER)	172
CTBR	
Time Base Counter Clear Register (CTBR)	117
D	
DACK	
Function of the DACK, DEOP, and DREQ Pins	404
Timing of DACK Pin Output	430
Data Access	
Data Access.....	72, 569
Data Bus Width	
Data Bus Width	185, 193
Relationship between Data Bus Width and Control Signal	182
Data Direction Registers	
Configuration of the Data Direction Registers (DDR)	261
Data Format	
Data Format	184, 192
Data Length	
Data Length (Data Width)	417
Data Register	
Data Register (ADCR)	354
Data Register (IDAR0/IDAR1)	467
Data Width	
Data Length (Data Width)	417
DC-DC Regulator	
Notes on Using the Internal DC-DC Regulator and A/D Converter.....	39
DDR	
Configuration of the Data Direction Registers (DDR)	261
Debugger	
Emulator Debugger/Monitor Debugger	571
Simulator Debugger	571
Dedicated Registers	
List of Dedicated Registers	63
Delay Slot	
Branch Instructions with Delay Slot	74
Branch Instructions without Delay Slot	78
Limitations on Branch Instruction with Delay Slot	77
Operation of Branch Instruction with Delay Slot	75
Operation of Branch Instruction without Delay Slot	78
Precaution on Delay Slot	93
Delayed Branch Instructions	
Delayed Branch Instructions	588
Delayed Branch Macro Instructions	
20-bit Delayed Branch Macro Instructions	592
32-bit Delayed Branch Macro Instructions	594
Delayed Interrupt Control Register	
Delayed Interrupt Control Register (DICR)	327
Delayed Interrupt Module	
Block Diagram of the Delayed Interrupt Module	326
Delayed Interrupt Module Registers	327
Demand Transfer	
Demand Transfer	434
Demand Transfer 2-Cycle Transfer.....	411
Timing of Demand Transfer	440
Timing of Transfer other than Demand Transfer	439
Demand Transfer Request	
Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped	428

INDEX

DEOP	Function of the DACK, DEOP, and DREQ Pins	404
	Timing of the DEOP Pin Output	431
Detection	0 Detection	447
	0 Detection Data Register (BSD0)	445
	1 Detection	447
	1 Detection Data Register (BSD1)	445
	Change Point Detection	448
	Lack of Error Detection	570
Detection Result Register	Detection Result Register (BSRR)	446
Device	State of Device and Each Transition	132
Device Initialization	Reset (Device Initialization)	94
Device States	Device States	131
DICR	Delayed Interrupt Control Register (DICR)	327
	DLYI Bit of DICR	328
Dimensions	Dimensions	8
Direct Addressing	Direct Addressing Area	42
Direct Addressing Instructions	Direct Addressing Instructions	595
Division	Clock Division	108
DIVR	Base Clock Division Setting Register 0 (DIVR0)	120
	Base Clock Division Setting Register 1 (DIVR1)	123
DLYI	DLYI Bit of DICR	328
DMA	Clearing Peripheral Interrupts by DMA	422
	DMA Access Operation	236
	DMA Fly-By Transfer (I/O ->Memory)	237
	DMA Fly-By Transfer (I/O ->SDRAM/FCRAM)	241
	DMA Fly-By Transfer (Memory ->I/O)	239
	DMA Fly-By Transfer (SDRAM/FCRAM ->I/O)	243
	DMA Transfer and Interrupts	419
	DMA Transfer during Sleep	425
	DMA Transfer Request during External Hold	420
	External Hold Request during DMA Transfer	420
	Operation Timing for DMA Fly-By Transfer (I/O ->Memory)	214
	Operation Timing for DMA Fly-By Transfer (Memory ->I/O)	215
	Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request	420
	Suppressing DMA	419
DMA Access Operation	DMA Access Operation	236
DMA Controller	DMA Controller (DMAC) Registers	386
	DMAC (DMA Controller)	3
DMA External Interface	DMA External Interface Pins	438
DMA Transfer	DMA Transfer and Interrupts	419
	DMA Transfer during Sleep	425
	External Hold Request during DMA Transfer	420
DMA Transfer Request	DMA Transfer Request during External Hold	420
	Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request	420
DMAC	AC Characteristics of DMAC	431
	Configuration of the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
	DMA Controller (DMAC) Registers	386
	DMAC (DMA Controller)	3
	Functions of Bits in the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
DMAC All-Channel Control Register	Bit Configuration of DMAC All-Channel Control Register (DMACR)	402
	Detailed Bit of DMAC All-Channel Control Register (DMACR)	402
DMAC Interrupt Control	DMAC Interrupt Control	425
DMACA	Bit Configuration of Control/Status Registers A (DMACA0 to DMACA4)	388
	Detailed Bit of Control/Status Registers A (DMACA0 to DMACA4)	388
DMACB	Bit Configuration of Control/Status Registers B (DMACB0 to DMACB4)	393
	Detailed Bit of Control/Status Registers B (DMACB0 to DMACB4)	393
DMACR	Bit Configuration of DMAC All-Channel Control Register (DMACR)	402
	Detailed Bit of DMAC All-Channel Control Register (DMACR)	402
DMADA	Bit Configuration of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)	400
	Detailed Bit of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)	400

DMASA

Bit Configuration of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)	400
Detailed Bit of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4)	400

Double Type

Use of the Double Type or Long Double Type....	567
--	-----

DREQ

Function of the DACK, DEOP, and DREQ Pins ..	404
Minimum Effective Pulse Width of the DREQ Pin Input	428
Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped....	428
Timing of the DREQ Pin Input for Continuing Transfer Over the Same Channel	430

E**EIRR**

External Interrupt Request Register (EIRR: External Interrupt Request Register)	319
---	-----

EIT

EIT (Exception, Interrupt, and Trap).....	79
EIT Causes	79
EIT Interrupt Levels	80
EIT Operations	90
EIT Vector Table	84
Priority of EIT Causes to Be Accepted.....	88
Return from EIT	79

ELVR

External Interrupt Request Level Setting Register (ELVR: External Level Register).....	320
--	-----

Embedded REALOS/FR

Embedded REALOS/FR Version.....	516
Outline of Embedded REALOS/FR	518

Emulator Debugger

Emulator Debugger/Monitor Debugger	571
--	-----

ENable Interrupt Request Register

Interrupt Enable Register (ENIR: ENable Interrupt Request Register)	318
---	-----

ENIR

Interrupt Enable Register (ENIR: ENable Interrupt Request Register)	318
---	-----

Entire PPG Timer

Block Diagram of the Entire PPG Timer	283
---	-----

Error Detection

Lack of Error Detection	570
-------------------------------	-----

Error Trap

Coprocessor Error Trap	93
------------------------------	----

Evaluation Chip

Configuration Example :Target Board + Evaluation Chip + ICE	534
---	-----

Exception

EIT (Exception, Interrupt, and Trap)	79
Operation of Undefined Instruction Exception	92

External Bus Access

External Bus Access	187
---------------------------	-----

External Bus Clock

External Bus Clock (CLKT)	107
---------------------------------	-----

External Clock

External Clock	32
External Clock Input after Power-on.....	39

External Devices

Example of Connection with External Devices.....	190
--	-----

Examples of Connection with External Devices.....	194
---	-----

External Hold

DMA Transfer Request during External Hold.....	420
--	-----

External Hold Request

External Hold Request during DMA Transfer.....	420
Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request	420

External I/O

2-Cycle Transfer (Internal RAM ->External I/O, RAM)	247
---	-----

Transfer between External I/O and External Memory	431
---	-----

External Interface

DMA External Interface Pins	438
-----------------------------------	-----

External Interrupt

Operating Procedure for an External Interrupt	321
Operation of an External Interrupt	321

External Interrupt and NMI Controller

Block Diagram of the External Interrupt and NMI Controller	316
--	-----

External Interrupt and NMI Controller Registers	317
---	-----

External Interrupt Request Level

External Interrupt Request Level	322
--	-----

External Interrupt Request Level Setting Register

External Interrupt Request Level Setting Register (ELVR: External Level Register)	320
---	-----

External Interrupt Request Register

External Interrupt Request Register (EIRR: External Interrupt Request Register)	319
---	-----

External Level Register

External Interrupt Request Level Setting Register (ELVR: External Level Register)	320
---	-----

External Memory

An Example of Connection of External Memory	532
---	-----

Transfer between External I/O and External Memory	431
---	-----

External Transfer Request

External Transfer Request Pin	408
-------------------------------------	-----

INDEX

External Wait	
With External Wait.....	219
Without External Wait.....	218
External Wait Cycle Timing	
External Wait Cycle Timing.....	209
F	
FCRAM	
2-Cycle Transfer (I/O ->SDRAM/FCRAM).....	250
2-Cycle Transfer (SDRAM/FCRAM ->I/O).....	251
Connecting SDRAM/FCRAM to Many Areas	230
DMA Fly-By Transfer (I/O ->SDRAM/FCRAM)	241
DMA Fly-By Transfer (SDRAM/FCRAM ->I/O)	243
SDRAM/FCRAM Interface.....	224
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)	167
Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode)	169
Features	
Features.....	45, 79, 144, 346
Features of PPG Timer.....	282
Features of the UART.....	360
Other Features.....	5
Flag	
I Flag	81
Flags	
Occurrence of Interrupts and Timing for Setting Flags	377
Flash Memory	
Allocation of Flash Memory.....	511
Flowcharts	
Flowcharts.....	503
Fly-By Transfer	
DMA Fly-By Transfer (I/O ->Memory).....	237
DMA Fly-By Transfer (I/O ->SDRAM/FCRAM)	241
DMA Fly-By Transfer (Memory ->I/O).....	239
DMA Fly-By Transfer (SDRAM/FCRAM ->I/O)	243
Flow of Data during Fly-By Transfer	437
Operation Timing for DMA Fly-By Transfer (I/O ->Memory)	214
Operation Timing for DMA Fly-By Transfer (Memory ->I/O)	215
FR	
Embedded REALOS/FR Version.....	516
FR CPU.....	2
Outline of Embedded REALOS/FR.....	518
REALOS/FR Configurator Setup.....	523
FR Family	
FR Family Instruction Lists	578
16-bit Free Run Timer	
Operational Explanation	485
Free Run Timer	
Block Diagram of the 16-bit Free Run Timer	480
Free Run Timer	5
Operational Explanation	485
Registers of 16-bit Free Run Timer	479
G	
GCN	
Activating Multiple Channels with the GCN.....	303
Bit Configuration of General Control Register 10 (GCN10)	293
Bit Configuration of General Control Register 20 (GCN20)	296
Details of General Control Register 10 (GCN10)	293
General Control Register	
Bit Configuration of General Control Register 10 (GCN10)	293
Bit Configuration of General Control Register 20 (GCN20)	296
Details of General Control Register 10 (GCN10)	293
General-purpose Registers	
General-purpose Registers	70
H	
Halfword Access	
Halfword Access.....	197
Hardware Configuration	
Hardware Configuration	343, 384
Hardware Configuration of the Interrupt Controller.....	330
Hold Request Cancellation Request	
Hold Request Cancellation Request (HRCR: Hold Request Cancel Request)	341
Hold Request Cancellation Request Sequence	344
Hold Request Cancellation Request Level Setting Register	
Bit Configuration of Hold Request Cancellation Request Level Setting Register (HRCL)	336
Detailed Bit of Hold Request Cancellation Request Level Setting Register (HRCL)	336
HRCL	
Bit Configuration of Hold Request Cancellation Request Level Setting Register (HRCL)	336
Detailed Bit of Hold Request Cancellation Request Level Setting Register (HRCL)	336
HRCR	
Hold Request Cancellation Request (HRCR: Hold Request Cancel Request)	341

I	
I Flag	
I Flag	81
I/O Circuit Types	
I/O Circuit Types	26
I/O Map	
I/O Map	536
I/O Pins	
I/O Pins.....	147
I/O Port	
Basic Block Diagram of the I/O Port.....	258
I/O Port Modes	259
I/O Wait Registers	
Configuration of the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
Functions of Bits in the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
I ² C	
I ² C Bus Interface	5
I ² C Bus Interface	
I ² C Bus Interface	5
I ² C Interface	
I ² C Interface Registers.....	451
I ² C interface	
Operational Explanation	468
IBCR	
Bus Control Register (IBCR0/IBCR1)	456
IBSR	
Bus Status Register (IBSR0/IBSR1)	454
ICCR	
Clock Control Register (ICCR0/ICCR1)	462
ICE	
Configuration Example :Target Board + Evaluation Chip + ICE	534
ICHCR	
Instruction Cache Control Register (ICHCR)	54
ICR	
Bit Configuration of Interrupt Control Register (ICR)	334
Configuration of Interrupt Control Register (ICR)	82
Detailed Bit of Interrupt Control Register (ICR)	334
Mapping of Interrupt Control Register (ICR)	82
ICS	
Input Capture Control Registers (ICS01,ICS23).....	493
IDAR	
Data Register (IDAR0/IDAR1)	467
IDBL	
Clock Disable Register (IDBL0/IDBL1)	467
ILM	
Interrupt Level Mask (ILM) Register	81
Immediate Set	
Immediate Set/16-bit/32-bit Immediate Transfer Instructions	583
Immediate Transfer Instructions	
Immediate Set/16-bit/32-bit Immediate Transfer Instructions	583
INIT	
Setting Initialization Reset (INIT) Clear Sequence	98
Settings Initialization Reset (INIT)	95
Settings Initialization Reset (INIT) State	135
INIT	
INIT Pin Input (Settings Initialization Reset Pin)	96
Initial Value	
Allocation of a Variable with an Initial Value.....	565
Initialization	
Initialization.....	376
Reset (Device Initialization)	94
Initialization Reset	
INIT Pin Input (Settings Initialization Reset Pin)	96
Operation Initialization Reset (RST).....	95
Operation Initialization Reset (RST) Clear Sequence	98
Operation Initialization Reset (RST) State.....	134
Setting Initialization Reset (INIT) Clear Sequence	98
Settings Initialization Reset (INIT)	95
Settings Initialization Reset (INIT) State	135
Input Capture	
Block Diagram of the Input Capture	492
Overview of Input Capture	490
Registers of the Input Capture.....	491
Input Capture Control Registers	
Input Capture Control Registers (ICS01,ICS23).....	493
Input Capture Data Registers	
Input Capture Data Registers (IPCP0 to IPCP3)	493
Instruction	
How to Read the Instruction Lists	573
Instruction Format	577
Operation of INT Instruction	91
Operation of INTE Instruction	91
Operation of RETI Instruction	93
Operation of Undefined Instruction Exception.....	92
Instruction Cache	
Areas Cacheable by the Instruction Cache.....	59
Configuration of Instruction Cache.....	51
Instruction Cache.....	3
Instruction Cache Tags.....	52
Updating Entries in the Instruction Cache	59
Instruction Cache Control Register	
Instruction Cache Control Register (ICHCR).....	54

INDEX

Instruction Cache Status	
Instruction Cache Status in Each Operating Mode	58
Instruction Format	
Instruction Format	577
Instruction Lists	
FR Family Instruction Lists	578
How to Read the Instruction Lists	573
Instructions	
20-bit Delayed Branch Macro Instructions	592
20-bit Normal Branch Macro Instructions	591
32-bit Delayed Branch Macro Instructions	594
32-bit Normal Branch Macro Instructions	593
Add-Subtract Instructions.....	579
Bit Manipulation Instructions	581
Compare Instructions.....	579
Coprocessor Control Instructions	596
Delayed Branch Instructions.....	588
Direct Addressing Instructions.....	595
Immediate Set/16-bit/32-bit Immediate Transfer Instructions	583
Logic Instructions.....	580
Memory Load Instructions	584
Memory Store Instructions	585
Multiply Instructions	582
Normal Branch (No Delay) Instructions	586
Other Instructions	589
Overview of Instructions	48
Register-to-Register Transfer Instructions.....	585
Resource Instructions.....	595
Shift Instructions	583
INT Instruction	
Operation of INT Instruction	91
INTE Instruction	
Operation of INTE Instruction	91
Interface	
SDRAM/FCRAM Interface.....	224
Internal Architecture	
Internal Architecture.....	46
Internal Clock	
Internal Clock Operation.....	277
Internal RAM	
2-Cycle Transfer (Internal RAM ->External I/O,RAM)	247
Interrupt	
Block Diagram of the Delayed Interrupt Module.....	326
Block Diagram of the External Interrupt and NMI Controller	316
Delayed Interrupt Module Registers	327
DMAC Interrupt Control.....	425
EIT (Exception,Interrupt,and Trap)	79
EIT Interrupt Levels	80
External Interrupt and NMI Controller Registers	317
External Interrupt Request Level	322
Interrupt Number	328
Interrupt Sources and Timing Chart (PPG Output: Normal Polarity)	301
Interrupt Stack	83
Level Mask for Interrupt and NMI.....	81
Operating Procedure for an External Interrupt.....	321
Operation of an External Interrupt	321
Interrupt Control Register	
Bit Configuration of Interrupt Control Register (ICR)	334
Configuration of Interrupt Control Register (ICR)	82
Detailed Bit of Interrupt Control Register (ICR)	334
Mapping of Interrupt Control Register (ICR)	82
Interrupt Controller	
Hardware Configuration of the Interrupt Controller.....	330
Interrupt Controller	4
Interrupt Controller Registers	332
Major Functions of the Interrupt Controller	330
Interrupt Enable Register	
Interrupt Enable Register (ENIR: ENable Interrupt Request Register)	318
Interrupt Level Mask	
Interrupt Level Mask (ILM) Register	81
Interrupt Levels	
EIT Interrupt Levels	80
Interrupt Vectors	
Interrupt Vectors	548
Interrupts	
DMA Transfer and Interrupts	419
Occurrence of Interrupts and Timing for Setting Flags.....	377
Interval Timers	
Other Interval Timers	5
IOWR	
Configuration of the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
Functions of Bits in the I/O Wait Registers for DMAC (IOWR0,IOWR1)	170
IPCP	
Input Capture Data Registers (IPCP0 to IPCP3).....	493
ISBA	
7-bit Slave Address Register (ISBA0/ISBA1).....	465
ISIZE	
Configuration of Cache Size Register (ISIZE)	54
ISMK	
7-bit Slave Address Mask Register (ISMK0/ISMK1)	466
ITBA	
10-bit Slave Address Register (ITBA0/ITBA1)...	464

ITMK	
10-bit Slave Address Mask Register (ITEMK0/ITEMK1)	464
K	
-K lib Option	
Specification of the -K lib Option When Using a String Manipulation Function	567
L	
Latch Up	
Preventing a Latch Up	32
Level Mask	
Level Mask for Interrupt and NMI	81
Line-up	
Product Line-up	6
Little Endian	
Allocation of a Stack to a Little Endian Area	567
Differences between Little Endian and Big Endian	191
Restrictions on the Little Endian Area	191
Location Addresses	
Mode Data and Reset Vector Location Addresses	531
Logic Instructions	
Logic Instructions	580
Long Double Type	
Use of the Double Type or Long Double Type	567
Low-power Modes	
Low-power Modes	131, 136
M	
MB91301 Series	
Features of the MB91301 Series	2
MB91302A	
Pin Layout of the MB91302A	12
MB91V301A	
Pin Layout of the MB91V301A.....	11
Unique to the Evaluation Chip MB91V301A	36
MCRA	
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)	167
MCRB	
Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode)	169
MD	
Mode Pins (MD0 to MD2)	33
MDH	
Multiply and Divide Registers (MDH/MDL)	65
MDL	
Multiply and Divide Registers (MDH/MDL)	65
Memory	
Built-in Memory.....	3
Memory Connection	
Memory Connection Example.....	232
Memory Load Instructions	
Memory Load Instructions.....	584
Memory Map	
Memory Map	43, 73, 498, 517
Memory Setting Register	
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode).....	167
Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode).....	169
Memory Store Instructions	
Memory Store Instructions	585
Minimum Effective Pulse Width	
Minimum Effective Pulse Width of the DREQ Pin Input.....	428
Mode	
Addressing Mode Symbols	575
Cascade Mode.....	314
Continuous Conversion Mode.....	357
Fetching Mode Data and Reset Vectors after the Device is Released from a Reset.....	532
Instruction Cache Status in Each Operating Mode	58
Mode Data and Reset Vector Access Addresses	531
Mode Data and Reset Vector Location Addresses	531
Mode Pin	530
Mode Settings	141
Operations in the Program Loader Mode.....	500
Overview of the Program Loader Mode	498
Single-shot Conversion Mode	356
Sleep Mode	136
Stop Conversion Mode	357
Stop Mode	138
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode).....	167
Structure of the Memory Setting Register (MCRB for FCRAM Auto-precharge ON Mode).....	169
Transfer Mode Settings	441
Mode Pins	
Mode Pins (MD0 to MD2).....	33
Mode Vector	
Mode Vector	530
Modes	
Bus Modes	140
I/O Port Modes.....	259
Low-power Modes.....	131, 136
Operating Modes	140
UART Operating Modes	373

INDEX

Monitor Debugger	
Emulator Debugger/Monitor Debugger	571
Multiple Channels	
Activating Multiple Channels with the GCN	303
Multiply and Divide Registers	
Multiply and Divide Registers (MDH/MDL)	65
Multiply Instructions	
Multiply Instructions	582
N	
NC	
Treatment of NC and OPEN Pins.....	33
Negate Timing	
Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped	428
NMI	
Block Diagram of the External Interrupt and NMI Controller	316
External Interrupt and NMI Controller	
Registers	317
Level Mask for Interrupt and NMI	81
NMI.....	322, 340
Operation of User Interrupt/NMI	90
No Delay	
Normal Branch (No Delay) Instructions	586
No-coprocessor Trap	
No-coprocessor Trap	93
Normal Branch	
Normal Branch (No Delay) Instructions	586
Normal Branch Macro Instructions	
20-bit Normal Branch Macro Instructions	591
32-bit Normal Branch Macro Instructions	593
Normal Polarity	
Interrupt Sources and Timing Chart (PPG Output: Normal Polarity)	301
Normal Reset	
Normal Reset Operation.....	101
Notes	
Notes	487
O	
Objects	
Objects	519
One-shot Operation	
One-shot Operation	299
OPEN Pins	
Treatment of NC and OPEN Pins.....	33
Operating Modes	
Operating Modes.....	140
Operating States	
Operating States of the Counter	279
Operation	
Operation Timing of the \overline{WRn} + Byte Control Type	204
Operation End	
Operation End/Stopping	423
Operation Initialization Reset	
Operation Initialization Reset (RST)	95
Operation Initialization Reset (RST) Clear Sequence.....	98
Operation Initialization Reset (RST) State	134
Operation Timing	
Auto-refresh Operation Timing	227
Burst Read/Write Operation Timing	225
Operation Timing for the $\overline{CSn} \rightarrow \overline{RD}/\overline{WRn}$ Setup and $\overline{RD}/\overline{WRn} \rightarrow \overline{CSn}$ Hold Settings	213
Operation Timing for DMA Fly-By Transfer (I/O \rightarrow Memory).....	214
Operation Timing for DMA Fly-By Transfer (Memory \rightarrow I/O).....	215
Operation Timing for Synchronous Write Enable Output	210
Operation Timing for the \overline{CSn} Delay Setting	212
Operation Timing of Read \rightarrow Write	206
Single Read Operation Timing	226
Single Read/Write Operation Timing	225, 226
Ordering	
Bit Ordering	71
Byte Ordering.....	71
Ordinary Bus Interface	
Ordinary Bus Interface	202
Oscillation Circuit	
Quartz Oscillation Circuit	32
Oscillation Stabilization Wait	
Oscillation Stabilization Wait Reset (RST) Status.....	134
Oscillation Stabilization Wait RUN State	134
Sources of an Oscillation Stabilization Wait	99
Oscillation Stabilization Wait Time	
Selecting an Oscillation Stabilization Wait Time	100
Other Instructions	
Other Instructions	589
Overview	
Overview	450
P	
PC	
Program Counter (PC).....	63
PCNH	
Bit Function of Control Status Registers (PCNH,PCNL)	286
Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0)	286

PCNL	
Bit Function of Control Status Registers (PCNH,PCNL)	286
Control Status Registers (PCNH:PCNH3 to PCNH0, PCNL:PCNL3 to PCNL0).....	286
PCR	
Configuration of the Pull-up Resistor Control Registers (PCR)	262
PCSR	
Bit Configuration of PPG Cycle Set Register (PPCSCR:PPCSR3 to PPCSR0).....	290
PDR	
Configuration of the Port Data Registers (PDR)	260
PDUT	
Bit Configuration of PPG Duty Set Register (PPDUT:PPDUT3 to PPDUT0).....	291
Peripheral Clock	
Peripheral Clock (CLKP).....	106
Peripheral Interrupts	
Clearing Peripheral Interrupts by DMA	422
PFR	
Configuration of the Port Function Registers (PFR).....	263
Function of the Port Function Registers (PFR)	264
Pin Functions	
Description of Pin Functions	15
Pin Layout	
Pin Layout of the MB91302A	12
Pin Layout of the MB91V301A.....	11
Pin No. Table	
Pin No. Table	13
Pin State Table	
Meaning of Terms in the Pin State Table	552
Pin State Table.....	553
Pin/Timing Control Register	
Configuration of the Pin/Timing Control Register (TCR)	175
Functions of Bits in the Pin/Timing Control Register (TCR)	175
Pins	
Treatment of NC and OPEN Pins	33
PLL	
PLL Multiply-by Rate	104
PLL Operation Enable	104
Port Data Registers	
Configuration of the Port Data Registers (PDR)	260
Port Function Registers	
Configuration of the Port Function Registers (PFR).....	263
Function of the Port Function Registers (PFR)	264
Power Supply	
Power Supply Pins	32
Power-on	
External Clock Input after Power-on.....	39
Power-on Sequence	229
Processing after Power-on	39
PPG Cycle Set Register	
Bit Configuration of PPG Cycle Set Register (PPCSR:PPCSR3 to PPCSR0)	290
PPG Duty Set Register	
Bit Configuration of PPG Duty Set Register (PPDUT:PPDUT3 to PPDUT0)	291
PPG Operation	
PPG Operation	297
PPG Output	
Examples for Setting PPG Output to all-"L" or all-"H"	301
Interrupt Sources and Timing Chart (PPG Output: Normal Polarity).....	301
PPG Timer	
Block Diagram of One Channel of the PPG Timer	284
Block Diagram of the Entire PPG Timer	283
Features of PPG Timer.....	282
Register List of PPG Timer.....	285
PPG Timer Register	
Bit Configuration of PPG Timer Register (PTMR:PTMR3 to PTMR0)	292
Precaution	
Precaution on Delay Slot	93
Precautions	
Precautions on the U-TIMER Control Register (UTIMC)	312
Precautions on Usage	379
Precautions on Use	33
Precautions on Using the 16-bit Reload Timer	280
Precautions on Using the A/D Converter.....	358
Precautions when Using	305
Prefetch Operation	
Prefetch Operation	221
Principal Operations	
Principal Operations	405
Priority	
Priority Among Channels	426
Priority Decision	337
Priority of EIT Causes to Be Accepted.....	88
Priority of State Transition Requests.....	135
Product Line-up	
Product Line-up.....	6
Program Access	
Program Access.....	72
Program Allocation	
Program Allocation.....	528
Program Counter	
Program Counter (PC).....	63

INDEX

Program Example	
Program Example.....	522
Program Example of Smooth Startup and Stop of Clock	129
Program Loader	
Setting the Program Loader	499
Program Loader Mode	
Operations in the Program Loader Mode	500
Overview of the Program Loader Mode.....	498
Program Status	
Program Status (PS) Register.....	66
PS	
Program Status (PS) Register.....	66
PTMR	
Bit Configuration of PPG Timer Register (PTMR:PTMR3 to PTMR0)	292
Pull-up Resistor Control Registers	
Configuration of the Pull-up Resistor Control Registers (PCR)	262
Pulse Width	
Minimum Effective Pulse Width of the DREQ Pin Input	428
Q	
Quartz Oscillation	
Quartz Oscillation Circuit	32
R	
RAM	
2-Cycle Transfer (Internal RAM ->External I/O,RAM).....	247
RCR	
Bit Functions of the Refresh Control Register (RCR)	178
Structure of the Refresh Control Register (RCR)	177
REALOS	
Embedded REALOS/FR Version.....	516
Outline of Embedded REALOS/FR.....	518
REALOS/FR Configurator Setup.....	523
REALOS/FR Configurator	
REALOS/FR Configurator Setup.....	523
Receive Operation	
Receive Operation.....	374
Refresh	
Self Refresh	228
Refresh Control Register	
Bit Functions of the Refresh Control Register (RCR)	178
Structure of the Refresh Control Register (RCR)	177
Register	
Interrupt Level Mask (ILM) Register	81
Program Status (PS) Register	66
Register-to-Register Transfer Instructions	
Register-to-Register Transfer Instructions	585
Reload Operation	
Reload Operation	414, 418
Reload Register	
Bit Configuration of the 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0).....	276
Reload Register (UTIMR:UTIMR2 to UTIMR0)	309
Reload Timer	
16-bit Reload Timer Registers	271
Overview of the 16-bit Reload Timer.....	270
Precautions on Using the 16-bit Reload Timer.....	280
Reload Timer.....	4
When the 16-bit Reload Timer is Used for Activation	304
Reload Values	
Example of Setting Baud Rates and U-TIMER Reload Values	382
Reset	
Fetching Mode Data and Reset Vectors after the Device is Released from a Reset.....	532
Normal Reset Operation	101
Operation Initialization Reset (RST)	95
Operation Initialization Reset (RST) State	134
Oscillation Stabilization Wait Reset (RST) Status.....	134
Reset (Device Initialization).....	94
Setting Initialization Reset (INIT) Clear Sequence.....	98
Settings Initialization Reset (INIT)	95
Settings Initialization Reset (INIT) State	135
Software Reset (STCR: SRST Bit Writing)	96
Synchronous Reset Operation	101
Watchdog Reset.....	97
Reset Vector	
Mode Data and Reset Vector Access Addresses.....	531
Mode Data and Reset Vector Location Addresses.....	531
Reset Vectors	
Fetching Mode Data and Reset Vectors after the Device is Released from a Reset.....	532
Reset Vectors	530
Resource Instructions	
Resource Instructions	595
RETI Instruction	
Operation of RETI Instruction	93
Return Pointer	
Return Pointer (RP).....	64
RP	
Return Pointer (RP).....	64

RSRR	Detailed Bit of Serial Mode Register (SMR)	363
Reset Source Register/Watchdog Timer Control Register (RSRR)	110	
RST	Serial Output Data Register	
Operation Initialization Reset (RST)	95	
Operation Initialization Reset (RST) Clear Sequence.....	98	
Operation Initialization Reset (RST) State	134	
Oscillation Stabilization Wait Reset (RST) Status.....	134	
RUN State	Setting Initialization Reset	
Oscillation Stabilization Wait RUN State	134	
RUN State (Normal Operation)	133	
S	Setting Initialization Reset	
SCR	Settings Initialization Reset (INIT)	95
Bit Configuration of Serial Control Register (SCR)	365	
Dedicated Bit of Serial Control Register (SCR)	365	
SDRAM	Settings Initialization Reset (INIT) State	135
2-Cycle Transfer (I/O ->SDRAM/FCRAM)	250	
2-Cycle Transfer (SDRAM/FCRAM ->I/O)	251	
Connecting SDRAM/FCRAM to Many Areas	230	
DMA Fly-By Transfer (I/O ->SDRAM/FCRAM)	241	
DMA Fly-By Transfer (SDRAM/FCRAM ->I/O)	243	
SDRAM/FCRAM Interface	224	
Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM Auto-precharge OFF Mode)	167	
Section	Shift Instructions	
Restriction on Section Types	570	
Section	568	
Section Allocation	Shift Instructions	583
Section Allocation	520	
Self Refresh	SIDR	
Self Refresh	228	
Send Operation	Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)	368
Send Operation	374	
Sequential Conversion	Simulator Debugger	
A/D Converter (Sequential Conversion Type)	4	
Serial Control Register	Simulator Debugger	571
Bit Configuration of Serial Control Register (SCR)	365	
Dedicated Bit of Serial Control Register (SCR)	365	
Serial Input Data Register	Simultaneous Occurrence	
Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)	368	
Serial Mode Register	Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request	420
Bit Configuration of Serial Mode Register (SMR)	363	
Sleep	Single Read Operation	
	Single Read Operation Timing	226
Sleep Mode	Single Read/Write Operation	
	Single Read/Write Operation Timing	225, 226
Sleep State	Single-shot Conversion Mode	
	Single-shot Conversion Mode	356
Slave Address	Slave Address	
	Example of Slave Address and Data Transfer	473
Sleep	Sleep	
	DMA Transfer during Sleep	425
Sleep Mode	Sleep Mode	
	Sleep Mode	136
Sleep State	Sleep State	
	Sleep State	133
Smooth Startup	Smooth Startup	
	Program Example of Smooth Startup and Stop of Clock	129
	Smooth Startup and Stop of Clock	128
SMR	SMR	
	Bit Configuration of Serial Mode Register (SMR)	363
	Detailed Bit of Serial Mode Register (SMR)	363
SODR	SODR	
	Serial Input Data Register (SIDR)/Serial Output Data Register (SODR)	368
Software Request	Software Request	
	Software Request	409

INDEX

Software Reset	
Software Reset (STCR: SRST Bit Writing).....	96
Source Clock	
Source Clock	103
Specifications	
Overview of Specifications	51
SRST	
Software Reset (STCR: SRST Bit Writing).....	96
SSP	
System Stack Pointer (SSP).....	64, 83
SSR	
Bit Configuration of Serial Status Register (SSR).....	369
Detailed Bit of Serial Status Register (SSR).....	369
Stack	
Allocation of a Stack to a Little Endian Area	567
Interrupt Stack	83
Standby Control Register	
Standby Control Register (STCR)	112
Standby Mode	
Return from Standby Mode (Sleep/Stop)	342
Startup Routine	
Startup Routine	521
State Transition Requests	
Priority of State Transition Requests	135
STCR	
Software Reset (STCR: SRST Bit Writing).....	96
Standby Control Register (STCR)	112
Step Trace Trap	
Operation of Step Trace Trap	92
Stop Conversion Mode	
Stop Conversion Mode	357
Stop Mode	
Stop Mode	138
STOP state	
Return Operation from STOP State	324
Stop State	
Stop State	133
Stopping	
Operation End/Stopping.....	423
Stopping due to an Error	423
String Arrangement	
Operations other than the String Arrangement Using a String Manipulation Function	566
String Manipulation Function	
Operations other than the String Arrangement Using a String Manipulation Function	566
Specification of the -K lib Option When Using a String Manipulation Function.....	567
Structure Assignment	
Structure Assignment.....	565
Subtract Instructions	
Add-Subtract Instructions.....	579
Successive Accesses	
Basic Timing (For Successive Accesses).....	203
Synchronous Reset	
Synchronous Reset Operation	101
Synchronous Write Enable Output	
Operation Timing for Synchronous Write Enable Output	210
System Calls	
Contained System Calls	518
System Stack Pointer	
System Stack Pointer (SSP)	64, 83
T	
Table Base Register	
Table Base Register (TBR)	64, 84
Target Board	
Configuration Example :Target Board + Evaluation Chip + ICE	534
TBCR	
Time Base Counter Control Register (TBCR).....	115
TBR	
Table Base Register (TBR)	64, 84
TCCS	
Timer Control Status Register (TCCS)	482
TCDT	
Timer Data Register (TCDT)	481
TCR	
Configuration of the Pin/Timing Control Register (TCR)	175
Functions of Bits in the Pin/Timing Control Register (TCR)	175
Temporary Stopping	
Temporary Stopping.....	422
Time Base Counter	
Time Base Counter	125
Time Base Counter Clear Register	
Time Base Counter Clear Register (CTBR)	117
Time Base Counter Control Register	
Time Base Counter Control Register (TBCR).....	115
Time-base Timer	
Time-base Timer.....	126
Timer Control Status Register	
Timer Control Status Register (TCCS)	482
Timer Data Register	
Timer Data Register (TCDT)	481
Timer Register	
Bit Configuration of the 16-bit Timer Register (TMR:TMR2 to TMR0)	275
Timing Chart	
Interrupt Sources and Timing Chart (PPG Output: Normal Polarity)	301

TMCSR	
Bit Configuration of the Control Status Register (TMCSR).....	272
Bit Functions of the Control Status Register (TMCSR)	272
TMR	
Bit Configuration of the 16-bit Timer Register (TMR:TMR2 to TMR0).....	275
TMRLR	
Bit Configuration of the 16-bit Reload Register (TMRLR:TMRLR2 to TMRLR0).....	276
Trace	
Operation of Step Trace Trap	92
Transfer	
Block Transfer.....	432
Burst Transfer.....	433
Demand Transfer	434
Example of Slave Address and Data Transfer	473
Flow of Data during 2-Cycle Transfer.....	435
Flow of Data during Fly-By Transfer	437
If an External Pin Transfer Request is Re-entered	
During Transfer	431
If Another Transfer Request Occurs during Block Transfer	431
Timing of the DREQ Pin Input for Continuing Transfer Over the Same Channel	430
Timing of Transfer other than Demand Transfer	439
Transfer between External I/O and External Memory	431
Transfer Address	
Transfer Address.....	406
Transfer Count	
Transfer Count and Transfer End.....	407
Transfer Count Control	
Transfer Count Control.....	418
Transfer Data	
Transfer Data Format	374, 375
Transfer End	
Transfer Count and Transfer End.....	407
Transfer Instructions	
Immediate Set/16-bit/32-bit Immediate Transfer Instructions.....	583
Register-to-Register Transfer Instructions	585
Transfer Mode	
Transfer Mode	405
Transfer Mode Settings.....	441
Transfer Request	
Transfer Request Acceptance and Transfer	421
Transfer Sequence	
Selection of the Transfer Sequence	410
Transfer Source/Transfer Destination Address Setting Register	
Bit Configuration of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4).....	400
Detailed Bit of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to DMASA4/DMADA0 to DMADA4).....	400
Transfer Type	
Transfer Type.....	406
Transition	
State of Device and Each Transition	132
Transition Requests	
Priority of State Transition Requests.....	135
Trap	
Coprocessor Error Trap	93
EIT (Exception,Interrupt, and Trap)	79
No-coprocessor Trap.....	93
Operation of Step Trace Trap	92
U	
UART	
Example of Using the UART	380
Selecting a Clock for the UART.....	373
UART	4
UART Operating Modes	373
UART Registers	362
Undefined Instruction	
Operation of Undefined Instruction Exception.....	92
Underflow Operation	
Underflow Operation	278
Unused Input Pins	
Unused Input Pins.....	32
User Interrupt	
Operation of User Interrupt/NMI.....	90
User Stack Pointer	
User Stack Pointer (USP)	65
User Tasks	
User Tasks	519
USP	
User Stack Pointer (USP)	65
UTIM	
U-TIMER (UTIM:UTIM2 to UTIM0)	309
UTIMC	
Bit Details of U-TIMER Control Register (UTIMC)	310
Precautions on the U-TIMER Control Register (UTIMC)	312
U-TIMER Control Register (UTIMC:UTIMC2 to UTIMC0).....	310
U-TIMER	
Example of Setting Baud Rates and U-TIMER Reload Values	382

INDEX

Overview of the U-TIMER.....	308
U-TIMER (UTIM:UTIM2 to UTIM0).....	309
U-TIMER Registers.....	309
U-TIMER Control Register	
Bit Details of U-TIMER Control Register	
(UTIMC).....	310
Precautions on the U-TIMER Control Register	
(UTIMC).....	312
U-TIMER Control Register (UTIMC:UTIMC2 to	
UTIMC0)	310
UTIMR	
Reload Register	
(UTIMR:UTIMR2 to UTIMR0)	309
V	
Variable	
Allocation of a Variable with an Initial Value	565
Vector Table	
EIT Vector Table	84
W	
Wait Cycle Timing	
Auto-Wait Cycle Timing.....	208
External Wait Cycle Timing.....	209
Watchdog Reset	
Watchdog Reset	97
Word Access	
Word Access.....	196

CM71-10114-4E

FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL

FR60

32-BIT MICROCONTROLLER

MB91301 Series

HARDWARE MANUAL

March 2007 the fourth edition

Published **FUJITSU LIMITED** Electronic Devices

Edited Business Promotion Dept.
