查询MB89P579A供应商

FUJITSU SEMICONDUCTOR

CM25-10140-1E

CONTROLLER MANUAL

# F²MC-8L

## 8-BIT MICROCONTROLLER

# MB89570 Series

# HARDWARE MANUAL

FUJITSU

# $F^2$MC-8L

## 8-BIT MICROCONTROLLER

# MB89570 Series

# HARDWARE MANUAL

FUJITSU LIMITED

# PREFACE

## ■ Purpose and Intended Reader of This Manual

The MB89570 series is a product developed as one of the 8-bit microcontroller general-purpose versions of $F^2MC$-8L family is battery controlled applications using SM bus. This series can be used widely for devices from consumer products to industrial equipment.

This manual describes the functions and operations of the MB89570 series for engineers who develop products using microcontrollers of the MB89570 series. For details on various instruction sets used, see "$F^2MC$-8L Programming Manual".

## ■ Trademark

$F^2MC$ is the abbreviation of FUJITSU Flexible Microcontroller.

Other system and product names in this manual are trademarks of respective companies or organizations.

The symbols ™ and ® are sometimes omitted in this manual.

## ■ The $I^2C$ Licence

Purchase of FUJITSU $I^2C$ components conveys a license under the Philips $I^2C$ Patent Rights to use these components in an $I^2C$ system, provided that the system conforms to the $I^2C$ Standard Specification as defined by Philips.

## ■ Organization of This Manual

This manual is organized into the following 19 chapters.

**CHAPTER1 "OVERVIEW"**

This chapter describes the features and basic specifications of the MB89570 series.

**CHAPTER2 "HANDLING DEVICE"**

This chapter describes the precautions to be taken when using the MB89570 series.

**CHAPTER3 "CPU"**

This chapter describes the functions and operations of the CPU.

**CHAPTER4 "I/O PORT"**

This chapter describes the functions and operations of the I/O port.

**CHAPTER5 "TIMEBASE TIMER"**

This chapter describes the functions and operations of the timebase timer.

**CHAPTER6 "WATCHDOG TIMER"**

This chapter describes the functions and operations of the watchdog timer.

**CHAPTER7 "WATCH PRESCALER"**

This chapter describes the functions and operations of the watch prescaler.

**CHAPTER8 "8/16-BIT TIMER/COUNTER"**

This chapter describes the functions and operations of the 8/16-bit timer/counter.

**CHAPTER9 "16-BIT TIMER/COUNTER"**

This chapter describes the functions and operations of the 16-bit timer/counter.

**CHAPTER10 "EXTERNAL INTERRUPTS (EDGES)"**

This chapter describes the functions and operations of the external interrupt circuit (edge).

**CHAPTER11 "A/D CONVERTER"**

This chapter describes the functions and operations of the A/D converter.

**CHAPTER12 "D/A CONVERTER"**

This chapter describes the functions and operations of the D/A converter.

**CHAPTER13 "COMPARATOR"**

This chapter describes the functions and operations of the comparator.

**CHAPTER14 "UART/SIO"**

This chapter describes the functions and operations of the UART/SIO.

**CHAPTER15 "$I^2C$"**

This chapter describes the functions and operations of the $I^2C$.

**CHAPTER16 "MULTI-ADDRESS $I^2C$"**

This chapter describes the functions and operations of the multi-address $I^2C$.

**CHAPTER17 "BRIDGE CIRCUIT"**

This chapter describes the functions and operations of a bridge circuit.

**CHAPTER18 "LCD CONTROLLER/DRIVER"**

This chapter describes the functions and operations of the LCD controller/driver.

**CHAPTER19 "WILD REGISTER FUNCTION"**

This chapter describes the functions and operations of the wild register function.

**"APPENDIX"**

his appendix includes I/O maps, instruction lists, and other information.

1. The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.
2. The information and circuit diagrams in this document are presented as examples of semiconductor device applications, and are not intended to be incorporated in devices for actual use. Also, FUJITSU is unable to assume responsibility for infringement of any patent rights or other rights of third parties arising from the use of this information or circuit diagrams.
3. The contents of this document may not be reproduced or copied without the permission of FUJITSU, Ltd.
4. FUJITSU semiconductor devices are intended for use in standard applications (computers, office automation and other office equipment, industrial, communications, and measurement equipment, personal or household devices, etc.).
   ***CAUTION:***
   *Customers considering the use of our products in special applications where failure or abnormal operation may directly affect human lives or cause physical injury or property damage, or where extremely high levels of reliability are demanded (such as aerospace systems, atomic energy controls, sea floor repeaters, vehicle operating controls, medical devices for life support, etc.) are requested to consult with FUJITSU sales representatives before such use. The company will not be responsible for damages arising from such use without prior approval.*
5. 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.
6. 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.

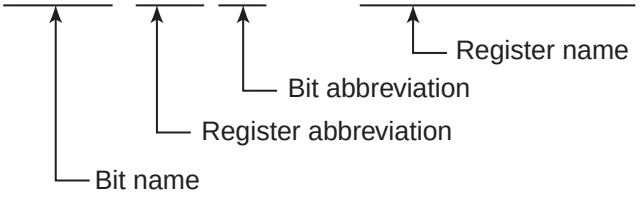
© 2001 FUJITSU LIMITED Printed in Japan

# READING THIS MANUAL

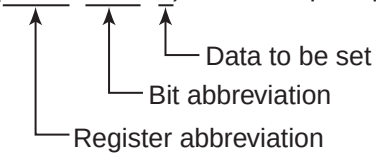
## ■ Notations of the Register Name and Pin Name

### ❍ Notations of the register name and bit name

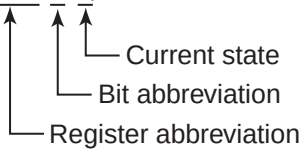
By writing "1" into the "sleep bit" (STBC: SLP) of the standby control register,




Prohibit (TBTC: TBIE=0) the interrupt request output of the timebase timer.




Accept an interrupt if the interrupt is permitted (CCR: I=1).




### ❍ Notations of a double-purpose pin

**P10/AN5 pin**

Some pins can be used by switching their functions using, for example, settings by a program. Each double-purpose pin is represented by separating the name of each function using "/".

## ■ Documents and Development Tools Required for Development

Items necessary for the development of this product are as follows.

To obtain the necessary documents and development tools, contact a company sales representative.

### ❍ Manuals required for development

**[Check field]**

☐ $F^2$MC-8L MB89570 series data sheet (provides a table of electrical characteristics and various examples of this product)

☐ $F^2$MC-8L Programming Manual (manual including instructions for the $F^2$MC-8L family)

* FR/$F^2$MC Family Softune C Compiler Manual (required only if C language is used for development) (manual describing how to develop and activate programs in the C language)

* FR/$F^2$MC Family Softune Assembler Manual for V3 (manual describing program development using the assembler language)

* FR/$F^2$MC Family Softune Linkage Kit Manual for V3 (manual describing functions and operations of the assembler, linker, and library manager

Manuals with the * mark are attached to each product.

Other manuals, such as those for development, are attached to respective products.

Software required for development

**[Check field]**

☐ Softune V3 Workbench

☐ Softune V3 for personal ICE (required only if the evaluation is performed for the personal-ICE)

☐ Softune V3 for compact ICE (required only if the evaluation is performed for the compact-ICE)

The type of software product is dependent on the OS to be used.

For details, see the $F^2$MC Development Tool Catalog or Product Guide.

❍ **What is needed for evaluation on the one-time PROM microcomputer (if the programming operation is performed at your side)**

**[Check field]**

☐ MB89P579

☐ EPROM programmer
Minato Electronics:
MODEL-1890A (Version 2.5 or later)
OU-910 [MOS UNIT] (Version 4.32r or later)
ML01-891 (3V conversion socket)

☐ Package conversion adapter
ROM2-100LQF-32DP-8LA (for LQFP)
ROM2-100TQF2-32DP-8LA (for TQFP)

❍ **Development tools**

**[Check field]**

☐ MB89PV570 (piggyback/evaluation device)

☐ Development tool

| Main unit | Pod | Probe |
|---|---|---|
| MB2141A + MB2144-508 | | MB2144-203 |

To use a the other development environment, contact respective makers.

❍ **References**

- "$F^2$MC Development Tool Catalog"
- "Microcomputer Product Guide"

# CONTENTS


**CHAPTER 1 OVERVIEW ..... 1**
1.1 MB89570 Series Features ..... 2
1.2 Product Lineup in the MB89570 Series ..... 5
1.3 Differences of Various Product and Precautions for Selecting the Products ..... 7
1.4 Block Diagram of the MB89570 Series ..... 8
1.5 Pin Assignment ..... 9
1.6 Package Dimensions ..... 10
1.7 Pin Description ..... 13
1.8 I/O Circuit Type ..... 18

**CHAPTER 2 HANDLING DEVICE ..... 23**
2.1 Notes on Handling Devices ..... 24

**CHAPTER 3 CPU ..... 25**
3.1 Memory Space ..... 26
3.1.1 Special Areas ..... 28
3.1.2 Storing 16-bit Data in Memory ..... 30
3.2 Dedicated Registers ..... 32
3.2.1 Condition Code Register (CCR) ..... 34
3.2.2 Register Bank Pointer (RP) ..... 37
3.3 General-purpose Registers ..... 38
3.4 Interrupts ..... 40
3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3, ILR4) ..... 42
3.4.2 Interrupt Processing ..... 44
3.4.3 Multiple Interrupts ..... 46
3.4.4 Interrupt Processing Time ..... 47
3.4.5 Stack Operation during Interrupt Processing ..... 48
3.4.6 Stack Area for Interrupt Processing ..... 49
3.5 Resets ..... 50
3.5.1 Reset Flag Register (RSFR) ..... 52
3.6 External Reset Pin ..... 54
3.6.1 Reset Operation ..... 55
3.6.2 Pin States during Reset ..... 57
3.7 Clock ..... 58
3.7.1 Clock Generator ..... 60
3.7.2 Clock Controller ..... 62
3.7.3 System Clock Control Register (SYCC) ..... 64
3.7.4 Clock Modes ..... 67
3.7.5 Oscillation Stabilization Wait Time ..... 70
3.8 Standby Mode (Low Power Consumption) ..... 72
3.8.1 Operating State in Standby Mode ..... 73
3.8.2 Sleep Mode ..... 74
3.8.3 Stop Mode ..... 75
3.8.4 Watch Mode ..... 77

3.8.5 Standby Control Register (STBC) ..... 78
3.8.6 State Transition Diagram 1 (Dual Clock) ..... 80
3.8.7 Notes on Using Standby Mode ..... 83
3.9 Memory Access Mode ..... 85

**CHAPTER 4 I/O PORT ..... 87**
4.1 Overview of the I/O Port ..... 88
4.2 Port 0 ..... 91
4.2.1 Registers of Port 0 (PDR0, DDR0) ..... 93
4.2.2 Operation of Port 0 ..... 94
4.3 Port 1 ..... 96
4.3.1 Registers of the Port 1 (PDR1, DDR1) ..... 98
4.3.2 Operation of the Port 1 ..... 100
4.4 Port 2 ..... 102
4.4.1 Registers of Port 2 (PDR2, DDR2) ..... 104
4.4.2 Operation of Port 2 ..... 106
4.5 Port 3 ..... 108
4.5.1 Register of Port 3 (PDR3) ..... 111
4.5.2 Operation of Port 3 ..... 112
4.6 Port 4 ..... 113
4.6.1 Register of Port 4 (PDR4) ..... 115
4.6.2 Operation of port 4 ..... 116
4.7 Port 5 ..... 117
4.7.1 Registers of the Port 5 (PDR5, DDR5) ..... 119
4.7.2 Operation of Port 5 ..... 121
4.8 Port 6 ..... 123
4.8.1 Register the Port 6 (PDR6) ..... 125
4.8.2 Operation of port 6 ..... 127
4.9 Port 7 ..... 128
4.9.1 Registers of Port 7 (PDR7, DDR7) ..... 130
4.9.2 Operation of Port 7 ..... 132
4.10 Port 8 ..... 134
4.10.1 Registers of Port 8 (PDR8, DDR8) ..... 137
4.10.2 Operation of Port 8 ..... 139
4.11 Port 9 ..... 141
4.11.1 Registers of Port 9 (PDR9, DDR9) ..... 144
4.11.2 Operation of Port 9 ..... 146
4.12 Port A ..... 148
4.12.1 Register of Port A (PDRA) ..... 150
4.12.2 Operation of the Port A ..... 151
4.13 Port B ..... 152
4.13.1 Register of Port B (PDRB) ..... 155
4.13.2 Operation of Port B ..... 156
4.14 Program Example of the I/O Ports ..... 157

**CHAPTER 5 TIMEBASE TIMER ..... 159**
5.1 Overview of the Timebase Timer ..... 160
5.2 Configuration of the Timebase Timer ..... 162

5.3 Timebase Timer Control Register (TBTC) ..... 164
5.4 Timebase Timer Interrupt ..... 166
5.5 Operation of the Timebase Timer ..... 167
5.6 Notes on Using the Timebase Timer ..... 169
5.7 Program Example of the Timebase Timer ..... 171

**CHAPTER 6 WATCHDOG TIMER ..... 173**
6.1 Overview of the Watchdog Timer ..... 174
6.2 Configuration of the Watchdog Timer ..... 175
6.3 Watchdog Timer Control Register (WDTC) ..... 177
6.4 Operation of the Watchdog Timer ..... 179
6.5 Notes on Using the Watchdog Timer ..... 181
6.6 Program Example of the Watchdog Timer ..... 182

**CHAPTER 7 WATCH PRESCALER ..... 185**
7.1 Overview of the Watch Prescaler ..... 186
7.2 Configuration of the Watch Prescaler ..... 188
7.3 Watch Prescaler Control Register (WPCR) ..... 190
7.4 Watch Prescaler Interrupt ..... 192
7.5 Operation of the Watch Prescaler ..... 193
7.6 Notes on Using the Watch Prescaler ..... 195
7.7 Program Example of the Watch Prescaler ..... 196

**CHAPTER 8 8/16-BIT TIMER/COUNTER ..... 199**
8.1 Overview of the 8/16-bit Timer/Counter ..... 200
8.2 Configuration of the 8/16-bit Timer/Counter ..... 203
8.3 Pins of the 8/16-bit Timer/Counter ..... 205
8.4 Registers of the 8/16-bit Timer/Counter ..... 207
8.4.1 Timer 1 Control Register (T1CR) ..... 208
8.4.2 Timer 2 Control Register (T2CR) ..... 211
8.4.3 Timer 1 Data Register (T1DR) ..... 214
8.4.4 Timer 2 Data Register (T2DR) ..... 216
8.5 8/16-bit Timer/Counter Interrupts ..... 218
8.6 Operation of the Interval Timer Function ..... 220
8.7 Operation of the Counter Function ..... 222
8.8 Operation of the Square Wave Output Initial Setting Function ..... 225
8.9 Operation of Stopping and Restarting the 8/16-bit Timer/Counter ..... 227
8.10 Status of the 8/16-bit Timer/Counter in Each Mode ..... 228
8.11 Notes on Using the 8/16-bit Timer/Counter ..... 229

**CHAPTER 9 16-BIT TIMER/COUNTER ..... 233**
9.1 Overview of the 16-bit Timer/Counter ..... 234
9.2 Configuration of the 16-bit Timer/Counter ..... 236
9.3 Pin of the 16-bit Timer/Counter ..... 238
9.4 Registers of the 16-bit Timer/Counter ..... 239
9.4.1 Timer Control Register (TMCR) ..... 240
9.4.2 16-bit Timer Count Register (TCR) ..... 242
9.5 16-bit Timer/Counter Interrupts ..... 243

9.6 Operation of the Interval Timer Function ..... 244
9.7 Operation of the Counter Function ..... 246
9.8 Status of the 16-bit Timer/Counter in Each Mode ..... 248
9.9 Notes on Using the 16-bit Timer/Counter ..... 249
9.10 Programe Example of the 16-bit Timer/Counter ..... 250

**CHAPTER 10 EXTERNAL INTERRUPTS (EDGES) ..... 255**
10.1 Overview of the External Interrupt Circuit ..... 256
10.2 Configuration of the External Interrupt Circuit ..... 257
10.3 Pins of the External Interrupt Circuit ..... 258
10.4 Registers of the External Interrupt Circuit ..... 260
10.4.1 External Interrupt Control Register (EIC1 to EIC2) ..... 261
10.5 External Interrupt Circuit Interrupts ..... 263
10.6 Operation of the External Interrupt Circuit ..... 264

**CHAPTER 11 A/D CONVERTER ..... 265**
11.1 Overview of the A/D Converter ..... 266
11.2 Configuration of the A/D Converter ..... 267
11.3 Pins of the A/D Converter ..... 270
11.4 Registers of the A/D Converter ..... 273
11.4.1 A/D Control Register 1 (ADC1) ..... 274
11.4.2 A/D Control Register 2 (ADC2) ..... 276
11.4.3 A/D Data Registers (ADDH, ADDL) ..... 278
11.4.4 A/D Enable Registers 1 to 2 (ADEN 1 to 2) ..... 279
11.5 A/D Converter Interrupt ..... 281
11.6 Operation of the A/D Converter ..... 282
11.7 Notes on Using the A/D Converter ..... 284
11.8 Program Example of the A/D Converter ..... 286

**CHAPTER 12 D/A CONVERTER ..... 289**
12.1 Overview of the D/A Converter ..... 290
12.2 Configuration of the D/A Converter ..... 291
12.3 Pins of the D/A Converter ..... 292
12.4 Registers of the D/A Converter ..... 293
12.4.1 D/A Control Register ..... 294
12.4.2 D/A Data Registers 1 and 2 (DADR1, 2) ..... 295
12.5 Operation of the D/A Converter ..... 296

**CHAPTER 13 COMPARATOR ..... 297**
13.1 Overview of the Comparator ..... 298
13.2 Configuration of the Comparator ..... 299
13.3 Pins of the Comparator ..... 303
13.4 Registers of the Comparator ..... 306
13.4.1 Comparator Control Register 1 (COCR1) ..... 308
13.4.2 Comparator Control Register 2 (COCR2) ..... 310
13.4.3 Comparator Status Register 1 (COSR1) ..... 312
13.4.4 Comparator Interrupt Control Register 1 (CICR1) ..... 315
13.4.5 Comparator Status Register 2 (COSR2) ..... 317

13.4.6 Comparator Interrupt Control Register (CICR2) ..... 319
13.4.7 Comparator Status Register 3 (COSR3) ..... 321
13.4.8 Comparator Status Register 4 (COSR4) ..... 323
13.4.9 Comparator Input Allow Register (CIER) ..... 325
13.5 Comparator Interrupts ..... 327
13.6 Operation of the Parallel Discharge Control ..... 329
13.7 Operation of the Sequential Discharge Control ..... 330
13.8 Sample Application ..... 331

**CHAPTER 14 UART/SIO ..... 333**
14.1 Overview of the UART/SIO ..... 334
14.2 Configuration of the UART/SIO ..... 335
14.3 Pins of the UART/SIO ..... 337
14.4 Registers of the UART/SIO ..... 339
14.4.1 Serial Mode Control Register 1 (SMC1) ..... 340
14.4.2 Serial Mode Control Register 2 (SMC2) ..... 342
14.4.3 Baud Rate Generator Reload Register (SRC) ..... 344
14.4.4 Serial Status and Data Register (SSD) ..... 345
14.4.5 Serial Input Data Register (SIDR) ..... 347
14.4.6 Serial Output Data Register (SODR) ..... 348
14.5 UART/SIO Interrupt ..... 349
14.6 Operation of the UART/SIO ..... 350
14.7 Operation of the Operation Mode 0 ..... 351
14.8 Operation of the Operation Mode 1 ..... 356

**CHAPTER 15 $I^2C$ ..... 363**
15.1 Overview of the $I^2C$ ..... 364
15.2 Configuration of the $I^2C$ ..... 366
15.3 Pins of the $I^2C$ ..... 370
15.4 Registers of the $I^2C$ ..... 372
15.4.1 $I^2C$ Bus Status Register (IBSR) ..... 374
15.4.2 $I^2C$ Bus Control Register (IBCR) ..... 376
15.4.3 $I^2C$ Clock Control Register (ICCR) ..... 379
15.4.4 $I^2C$ Address Register (IADR) ..... 381
15.4.5 $I^2C$ Data Register (IDAR) ..... 382
15.4.6 $I^2C$ Timeout Control Register (ITCR) ..... 383
15.4.7 $I^2C$ Timeout Status Register (ITSR) ..... 385
15.4.8 $I^2C$ Timeout Data Register (ITOD) ..... 387
15.4.9 $I^2C$ Timeout Clock Register (ITOC) ..... 388
15.4.10 $I^2C$ Master Timeout Register (IMTO) ..... 389
15.4.11 $I^2C$ Slave Timeout Register (ISTO) ..... 390
15.5 $I^2C$ Interrupts ..... 391
15.6 Operation of the $I^2C$ ..... 392
15.7 Notes on Using the $I^2C$ ..... 395
15.8 Operation of the Timeout Detection Function ..... 396

**CHAPTER 16 MULTI-ADDRESS $I^2C$ ..... 401**
16.1 Overview of the Multi-address $I^2C$ ..... 402

16.2 Configuration of the Multi-address I2C ..... 404
16.3 Pins of the Multi-address I2C ..... 408
16.4 Registers of the Multi-address I2C ..... 410
16.4.1 Multi-address I2C Bus Status Register (MBSR) ..... 412
16.4.2 Multi-address I2C Bus Control Register (MBCR) ..... 414
16.4.3 Multi-address I2C Clock Control Register (MCCR) ..... 417
16.4.4 Multi-address I2C Address Registers (MADR1 to 6) ..... 419
16.4.5 Multi-address I2C Data Register (MDAR) ..... 420
16.4.6 Multi-address I2C Timeout Control Register (MTCR) ..... 421
16.4.7 Multi-address I2C Timeout Status Register (MTSR) ..... 423
16.4.8 Multi-address I2C Timeout Data Register (MTOD) ..... 425
16.4.9 Multi-address I2C Timeout Clock Register (MTOC) ..... 426
16.4.10 Multi-address I2C Master Timeout Register (MMTO) ..... 427
16.4.11 Multi-address I2C Slave Timeout Register (MSTO) ..... 428
16.4.12 Multi-address I2C ALERT Register (MALR) ..... 429
16.5 Multi-address I2C Interrupts ..... 430
16.6 Operation of the Multi-address I2C ..... 432
16.7 Notes on Using the Multi-address I2C ..... 435
16.8 Operation of the Timeout Detection Function ..... 436

**CHAPTER 17 BRIDGE CIRCUIT ..... 441**
17.1 Overview of the Bridge Circuit ..... 442
17.2 Configuration of the Bridge Circuit ..... 443
17.3 Pins of the Bridge Circuit ..... 444
17.4 Registers of the Bridge Circuit ..... 447
17.4.1 Bridge Circuit Selection Register 1 ( BRSR1) ..... 448
17.4.2 Bridge Circuit Selection Register 2 (BRSR2) ..... 450
17.4.3 Bridge Circuit Selection Register 3 (BRSR3) ..... 452

**CHAPTER 18 LCD CONTROLLER DRIVER ..... 455**
18.1 Overview of the LCD Controller Driver ..... 456
18.2 Configuration of the LCD Controller Driver ..... 457
18.2.1 Internal Dividing Resistors of the LCD Controller Driver ..... 459
18.2.2 External Dividing Resistor of LCD Controller Driver ..... 461
18.3 Pins of the LCD Controller Driver ..... 463
18.4 Registers of the LCD Controller Driver ..... 465
18.4.1 LCDC Control Register 1 (LCR1) ..... 466
18.4.2 LCDC Control Register 2 (LCR2) ..... 469
18.4.3 LCDC Control Register 3 (LCR3) ..... 471
18.4.4 LCDC Control Register 4 (LCR4) ..... 473
18.5 LCD Display RAM in the LCD Controller Driver ..... 475
18.6 Operation of the LCD Controller Driver ..... 477
18.6.1 Output Waveforms during LCD Controller Driver Operation (1/2 Duty) ..... 479
18.6.2 Output Waveforms During LCD Controller Driver Operation (1/3 Duty) ..... 482
18.6.3 Output Waveforms During LCD Controller Driver Operation (1/4 Duty) ..... 485

**CHAPTER 19 WILD REGISTER FUNCTION ..... 489**
19.1 Overview of the Wild Register Function ..... 490

19.2 Configuration of the Wild Register Function ..... 491
19.3 Registers of the Wild Register Function ..... 493
19.3.1 Data Setting Registers (WRDR0 to 5) ..... 494
19.3.2 Upper Address Setting Registers (WRARH0 to 5) ..... 496
19.3.3 Lower Address Setting Registers (WRARL 0 to 5) ..... 498
19.3.4 Address Comparison Enable Register (WREN) ..... 500
19.4 Operation of the Wild Register Function ..... 502

**APPENDIX ..... 503**
APPENDIX A I/O Maps ..... 504
APPENDIX B Overview of Instructions ..... 510
B.1 Overview of F$^2$MC-8L instructions ..... 511
B.2 Addressing ..... 514
B.3 Special Instructions ..... 518
B.4 Bit Manipulation Instructions (SETB, CLRB) ..... 521
B.5 F$^2$MC-8L Instructions ..... 522
B.6 Instruction Map ..... 528
APPENDIX C Mask Options ..... 529
APPENDIX D One-time PROM and EPROM Microcontroller Write Specification ..... 530
APPENDIX E Pin Statuses of the MB89570 Series ..... 531

**INDEX .....533**

# CHAPTER 1 OVERVIEW

**This chapter describes the features and basic specifications of the MB89570 series.**


1.1 "MB89570 Series Features"

1.2 "Product Lineup in the MB89570 Series"

1.3 "Differences of Various Product and Precautions for Selecting the Products"

1.4 "Block Diagram of the MB89570 Series"

1.5 "Pin Assignment"

1.6 "Package Dimensions"

1.7 "Pin Description"

1.8 "I/O Circuit Type"

# 1.1 MB89570 Series Features

**The MB89570 series is a general-purpose single chip microcontroller that contains, in addition to a compact instruction set, such rich peripheral functions as the dual clock control, 5-stage operation speed control, SM bus compliant $I^2C$ bus interface, comparator for battery control, 10-bit A/D converter, LCD controller/driver, and external interrupts.**

## ■ MB89570 Series Features

❍ **Package**

- LQFP package (0.5 mm pitch)
- TQFP package (0.4 mm pitch)

❍ **High-speed operation at low voltage**

- Minimum instruction execution time: 0.4 μs (for oscillator frequency 10 MHz)

❍ **$F^2MC$-8L CPU core**

Instruction set appropriate to the controller

- Multiplication/division instruction
- 16-bit operation
- Branch instruction by the bit test
- Bit operation instructions, etc.

❍ **Dual clock control**

- Main clock: up to 10 MHz (4 clock operating speeds can be set. Oscillation stops in subclock mode)
- subclock: 32.768 kHz (Operating clock in subclock mode)

❍ **Dual timer operation**

- 21-bit timebase timer
- Watch prescaler (17 bits)

❍ **UART/serial interface**

- UART/SIO can be switched.

❍ **$I^2C$**

- SM bus compliant
- Timeout can be detected.

❍ **Multi-address $I^2C$**

- SM bus compliant
- Timeout can be detected.
- 6 addresses support
- ALERT function support

❍ **Bridge circuit**

- Three bus connection routes can be switched by the $I^2C$/multi-address $I^2C$ (UART)

❍ **External interrupt**

- External interrupt (4 x edge detection): Four inputs are independent and can be used for release from low-power mode (The rising edge, falling edge, and both edges can be selected for edge detection)

❍ **Comparator function**

- A selecting circuit for battery control is contained.
- A comparator capable of changing the hysteresis width is contained.

❍ **10-bit A/D converter**

- 12 channels of A/D converters with the 10-bit resolution are contained.

❍ **8-bit D/A converter**

- 8-bit D/A converter x 2 channels

❍ **16-bit timer counter**

- Usable as an event counter

❍ **8/16-bit timer/counter**

- Usable as 8-bit timer x 2 channels or 16-bit timer x 1 channel

❍ **LCD controller/driver**

- 14SEG x 4COM (up to 56 pixels)
  - Exclusively for segment output: 8
  - For both of general purpose and LCD segment: 6

❍ **Low-power consumption (standby mode)**

- Stop mode (Almost no power consumption for stopping oscillation)
- Sleep mode (about 1/3 of the normal power consumption to stop CPU)
- Watch mode (Power consumption to stop operations other than the watch prescaler is very low)
- Sub-mode

- **Up to 82 I/O ports**
  - General purpose I/O port (N channel open-drain): 28
  - General purpose I/O port (CMOS): 49
  - General purpose input port (CMOS): 1
  - General purpose output port (N channel open drain): 4

# 1.2 Product Lineup in the MB89570 Series

**12 products are available in the MB89570 series. Table 1.2-1 "MB89570 Series Product Lineup" lists the products available and Table 1.2-2 "MB89570 Series CPU and Peripheral Functions" lists the CPU and peripheral functions.**

## ■ Product Lineup in the MB89570 Series

**Table 1.2-1 MB89570 Series Product Lineup**

| | MB89PV570(*1) | MB89P579A | MB89577 |
|---|---|---|---|
| ROM size | - | 60KB | 32KB |
| RAM size | 3KB | 3KB | 3KB |
| Package | LQFP100 | LQFP100<br>TQFP100 | LQFP100<br>TQFP100 |
| Classifications | Evaluation product | One Time PROM product | MASK product |

*1: In MB89PV570, only the evaluation function (function in which development tools can be used) is available. The piggyback function (function in which $E^2$PROM can be mounted) cannot be used.

## ■ Selection of the Oscillation Stabilization Wait Time

In MB89577, it is possible to select the initial value of the oscillator stabilization wait time when the mask ROM product is ordered.

| Oscillation stabilization wait time selection | Remarks |
|---|---|
| $2^{14}/F_{CH}$ | 1.63 ms (If F=10Mz) |
| $2^{17}/F_{CH}$ | 13.1 ms (If F=10Mz) |
| $2^{18}/F_{CH}$ | 26.2 ms (If F=10Mz) |

**Table 1.2-2 MB89570 Series CPU and Peripheral Functions**

| Item | | Specifications |
|---|---|---|
| CPU function | | Number of basic instructions: 136<br>Instruction bit length: 8 bits<br>Instruction length: 1 to 3 bytes<br>Data bit length: 1, 8, and 16-bit<br>Minimum instruction execution time: 0.4 μs (at 10 MHz)<br>Interrupt processing time: 3.6 μs (at 10 MHz) |
| Peripheral function | Port | General purpose I/O port (N-ch open-drain): 28<br>General purpose I/O port (CMOS): 49<br>General purpose input only port (CMOS): 1<br>General purpose output only port (N-ch open-drain): 4<br>Total: 82 (maximum) |
| | Timebase timer | 21-bit<br>Interrupt cycle for 10 MHz main clock (0.82 ms, 3.3 ms, 26.2 ms, 419.4 ms) |
| | Watchdog timer | Reset generation cycle<br>For 10 MHz of the main clock (minimum 209.7 ms)<br>For 32. 768 kHz of the sub clock (minimum 500 ms) |
| | SM bus compliant $I^2C$ bus | Support of the $I^2C$ bus of PHILIPS and the SM bus proposed by Intel<br>$I^2C$ bus (SM bus compliant) x 1 channel<br>Multi-address $I^2C$ bus (SM bus compliant) x 1 channel<br>Master/slave sending/receiving. Slave general call address detection function. Bus error function. Arbitration function. Transfer direction detection function. Repeated generation and detection function of the start condition. Timeout detection function. ALERT function (only for the multi-address $I^2C$) |
| | UART/SIO | Data can be transferred in UART/SIO.<br>Variable data length (7/8 bits), baud rate generator contained, transfer rate (1200 to 78125 bps at 10 MHz), full duplex with double buffers, NRZ transfer format, error detection function, and data transferable both in clock synchronous (SIO) and clock asynchronous (UART) modes |
| | Comparator | A comparator that can change the hysteresis width is contained.<br>The battery voltage, mounting/dismounting, and instantaneous interruption are detected, and the parallel and serial charging/discharging are controlled.<br>The serial and parallel connection control is performed by software. |
| | A/D converter | 10-bit x 12 channels |
| | D/A converter | 8-bit x 2 channels |
| | 16-bit timer/counter | 16-bit timer operation<br>16-bit event counter operation |
| | 8/16-bit timer/counter | 8-bit timer x 2 channels (usable as 16-bit timer x 1 channel) |
| | LCD controller/driver | Up to 14SEG x 4COM (The LCD output/N-ch open-drain I/O port can be selected) |
| | External interrupt | 4 (Edges can be selected) |
| Standby mode | | Sub-mode/sleep mode/watch mode/stop mode |

# 1.3 Differences of Various Product and Precautions for Selecting the Products

**This section explains the differences among the models available in the MB89570 series and precautions when selecting various models.**

## ■ Differences of Various Product and Precautions for Selecting the Products

### ❍ Correspondence table between the package and the product type

| | MB89PV570 | MB89P579A | MB89577 |
|---|---|---|---|
| FPT-100P-M05<br>(LQF-100 0.5mm pitch) | No | Yes | Yes |
| FTP-100-M18<br>(TQFP-100 0.4 mm pitch) | No | Yes | Yes |
| MQP-100C-P02<br>(MQFP-100 0.5 mm pitch) | Yes | No(*1) | No(*1) |

*1: A pin pitch conversion socket (Sun Hayato) is available.
100SQF-100TQF-8L-FJ: for MQP-100C-P02 --> FPT-100P-M18 conversion

### ❍ Memory space

Before evaluating products using piggyback products, check the differences between the piggyback products and the products actually used.

### ❍ Consumption current

- When operating at a low speed, the power consumption of models with one-time PROM or EPROM is higher than that with mask ROM. However, the power consumption in sleep/stop mode are comparable for both.
- For details on each package, see Section 1.6 "Package Dimensions".
- For details on the power consumption, see the electric characteristics in "Data sheet".
- Operating voltage
- The operating voltage is dependent on the product type.
- For details, see "Data sheet".

### ❍ Mask option

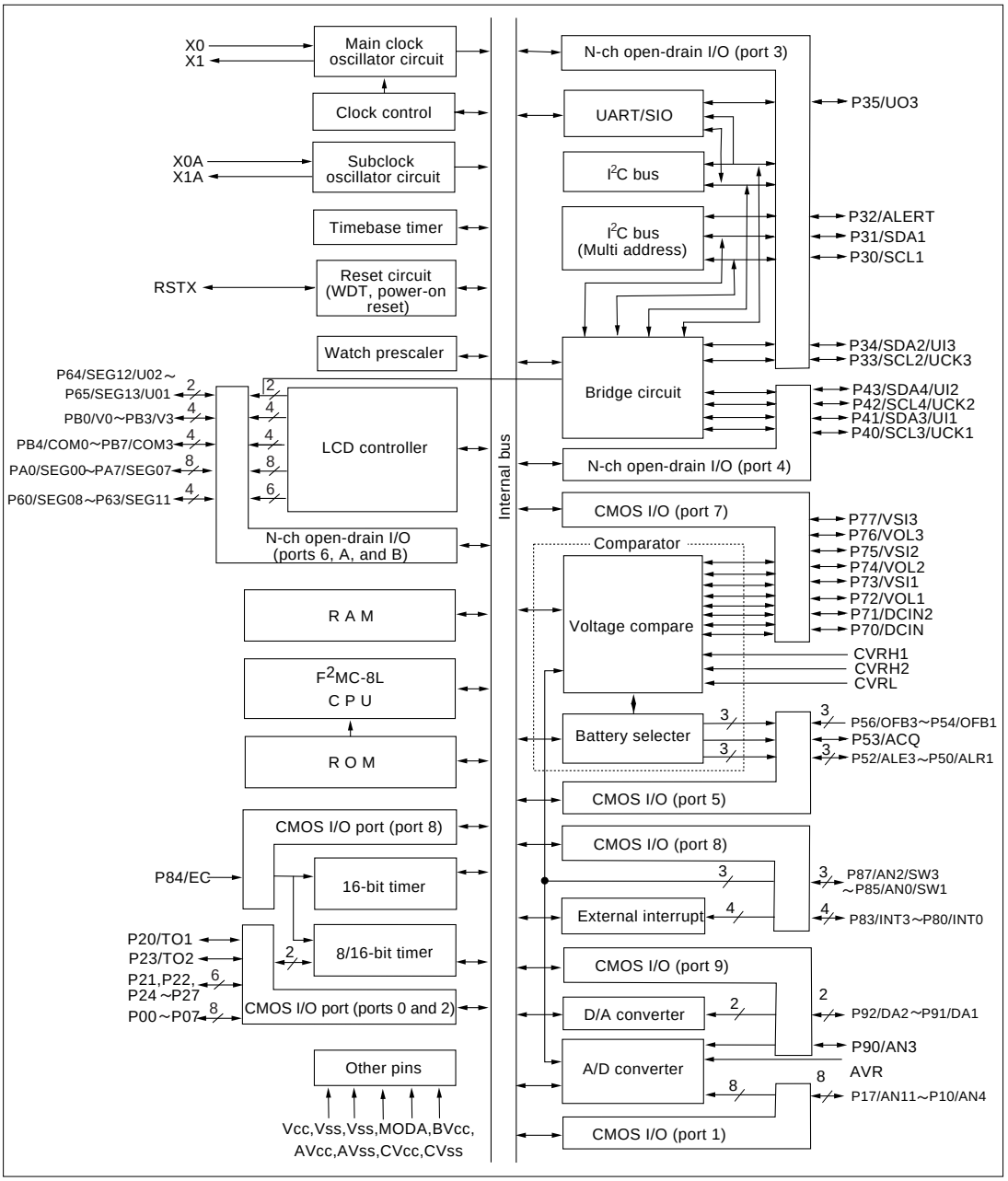
The options that can be selected and the methods of specifying are dependent on respective products. Before using the options, see Appendix C "Mask Options".

# 1.4 Block Diagram of the MB89570 Series

**This section shows an overall block diagram of the MB89570 series.**

## ■ Block Diagram of the MB89570 Series

**Figure 1.4-1 MB89570 Series Overall Block Diagram**

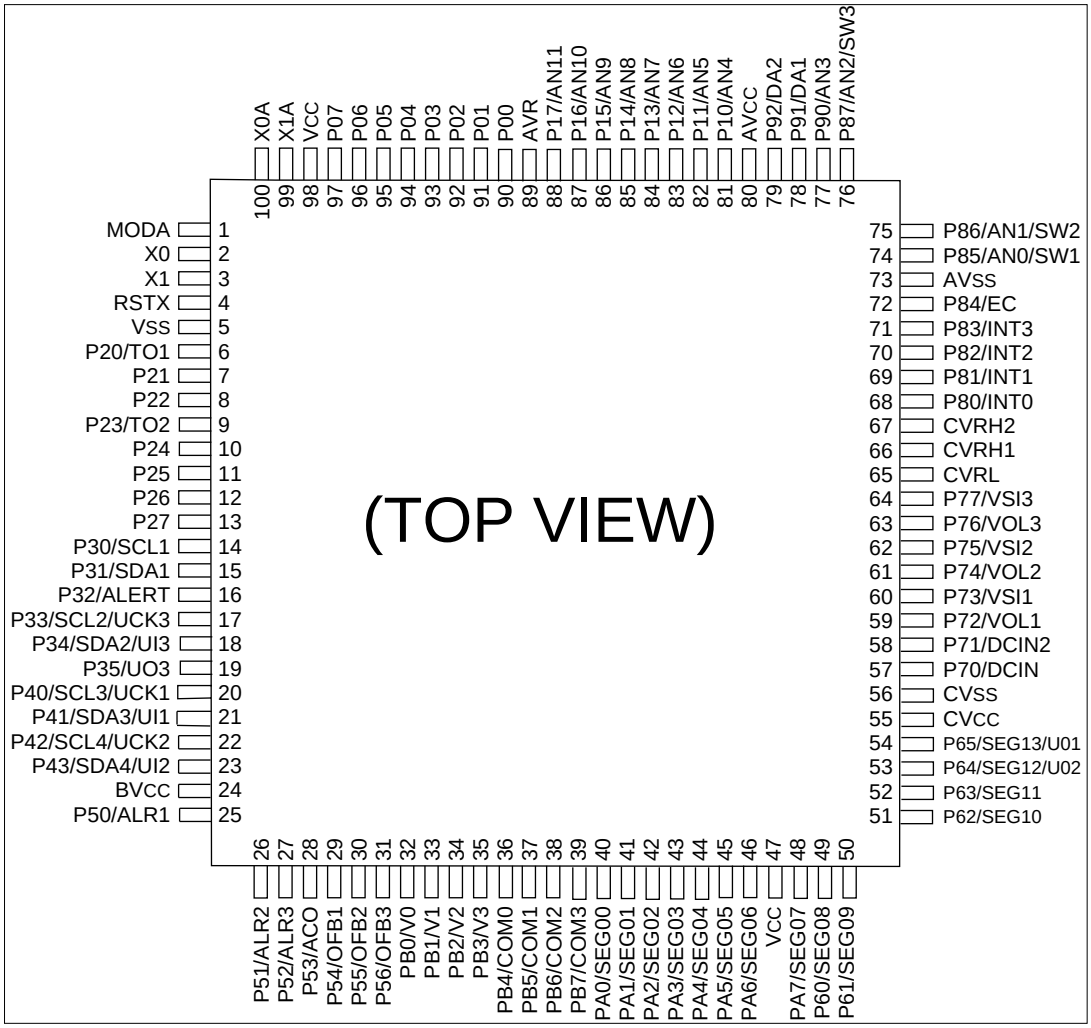


# 1.5 Pin Assignment

**Figure 1.5-1 "MB89570 Series Pin Assignment" shows a pin assignment of the MB89570 series.**

■ **Pin Assignment of the MB89570 Series (FPT-100P-M05, FPT-100P-M18, MQP-100C-P02)**

**Figure 1.5-1 MB89570 Series Pin Assignment**

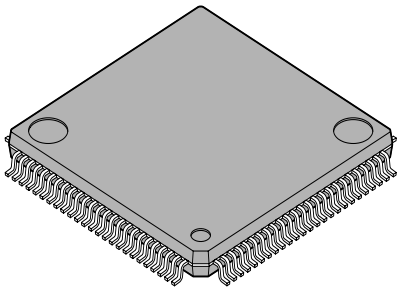


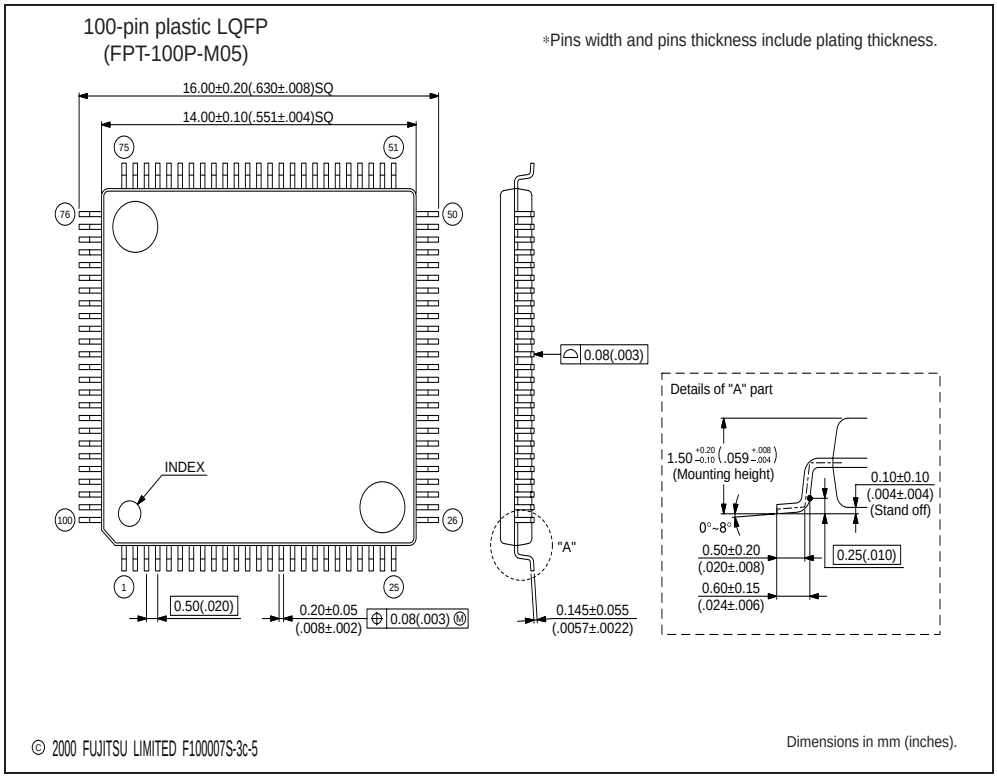
# 1.6 Package Dimensions

**Three types of packages are available in the MB89570 series.**

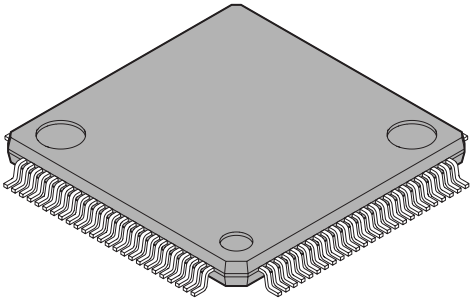
## ■ Package Dimensions of FPT-100P-M05

EIAJ code : P-LFQFP100-14 × 14-0.50

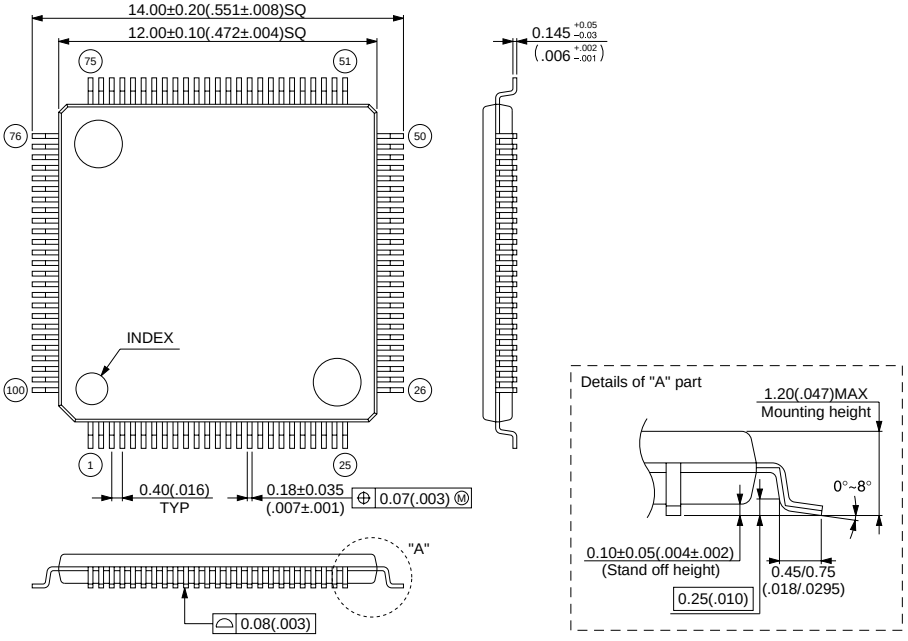
| 100-pin plastic LQFP | | |
|---|---|---|
|  | Lead pitch | 0.50 mm |
| | Package width × package length | 14.0 × 14.0 mm |
| | Lead shape | Gullwing |
| | Sealing method | Plastic mold |
| | Mounting height | 1.70 mm MAX |
| | Weight | 0.65g |
| (FPT-100P-M05) | | |



## ■ Package Dimensions of FPT-100P-M18

| | | |
|---|---|---|
| 100-pin plastic TQFP<br>  <br>(FPT-100P-M18) | Lead pitch | 0.40 mm |
| | Package width × package length | 12 × 12 mm |
| | Lead shape | Gullwing |
| | Sealing method | Plastic mold |
| | Length of flat portion of pins | 1.20 mm MAX |
| | | |
| | | |

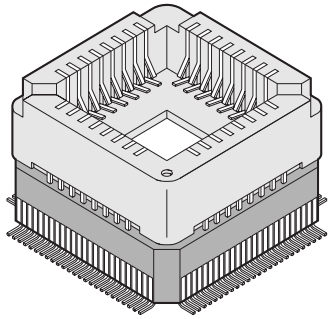
100-pin plastic TQFP
(FPT-100P-M18)



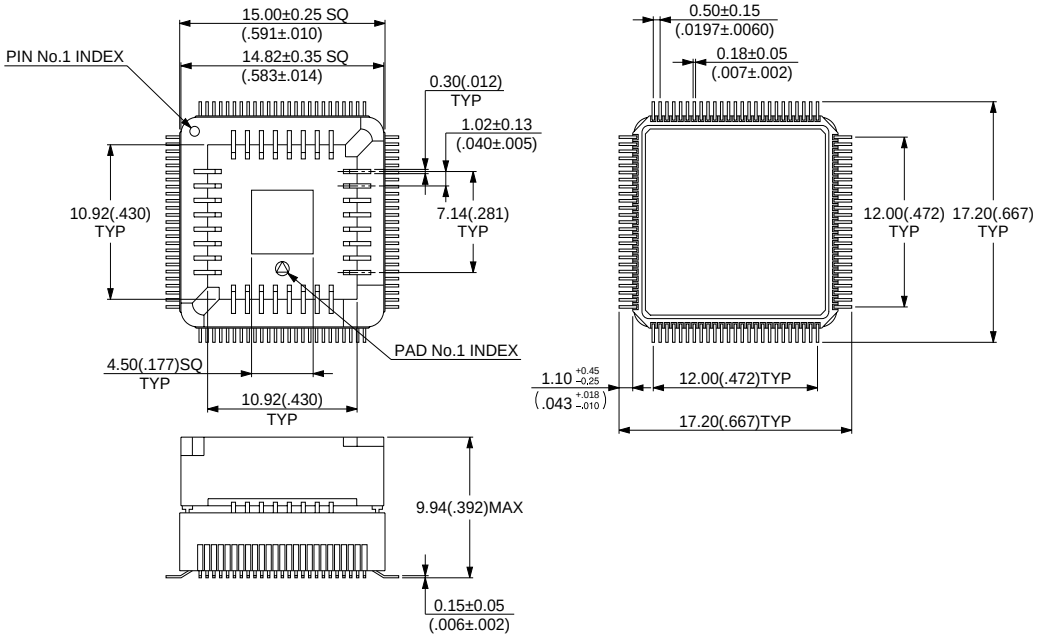

© 1997 FUJITSU LIMITED F100029S-1C-1

Dimensions in mm (inches).

## ■ Package Dimensions of MQP-100C-P02

| 100-pin ceramic MQFP | | |
|---|---|---|
|  | Lead pitch | 0.50 mm |
| | Lead shape | Straight |
| | Motherboard material | Ceramic |
| | Mounted package material | Plastic |
| | | |
| | | |
| (MQP-100C-P02) | | |

100-pin ceramic MQFP
(MQP-100C-P02)




© 1994 FUJITSU LIMITED M100002SC-2-2

Dimensions in mm (inches).

# 1.7 Pin Description

**Table 1.7-1 "Pin Description" provide pin function explanations. Alphabets in the I/O circuit field of Table 1.7-1 "Pin Description" correspond to those in the classification field of Table 1.8-1 "I/O Circuit Type".**

## ■ Pin Description

**Table 1.7-1 Pin Description**

| Pin No. | Pin name | I/O Circuit type | Function |
|---|---|---|---|
| 1 | MODA | A | Pin to specify the operation mode |
| 2 | X0 | C | Pin for crystal oscillator (max 10 MHz) |
| 3 | X1 | | |
| 4 | RSTX | D | Reset I/O pin |
| 5 | Vss | – | Power pin (GND) |
| 6 | P20/T01 | B | General-purpose I/O port. This pin is used also for 8/16-bit timer output. |
| 7 | P21 | | General-purpose I/O port |
| 8 | P22 | | |
| 9 | P23/T02 | | General-purpose I/O port. This pin is used also for 8/16-bit timer output. |
| 10 | P24 | | General-purpose I/O port |
| 11 | P25 | | |
| 12 | P26 | | |
| 13 | P27 | | |
| 14 | P30/SCL1 | F | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SCL1 I/O of the multi-address $I^2C$. |
| 15 | P31/SDA1 | | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SDA1 I/O of the multi-address $I^2C$. |
| 16 | P32/ALERT | H | General-purpose N-ch open-drain I/O port.<br>This pin is also used for ALERT I/O of the multi-address $I^2C$. |

**Table 1.7-1 Pin Description (Continued)**

| Pin No. | Pin name | I/O Circuit type | Function |
|---|---|---|---|
| 17 | P33/SCL2/UCK3 | G | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SCL2 I/O of $I^2C$ and UCK3 I/O of UART. |
| 18 | P34/SDA2/UI3 | | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SDA2 I/O of $I^2C$ and UI3 input of UART. |
| 19 | P35/U03 | H | General-purpose N-ch open-drain I/O port.<br>This pin is also used for UO3 output of UART. |
| 20 | P40/SCL3/UCK1 | G | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SCL/UCK1 I/O of a bridge circuit. |
| 21 | P41/SDA3/UI1 | | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SDA3/UI1 I/O of a bridge circuit. |
| 22 | P42/SCL4/UCK2 | | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SCL4/UCK2 I/O of a bridge circuit. |
| 23 | P43/SDA4/UI2 | | General-purpose N-ch open-drain I/O port.<br>This pin is also used for SDA4/UI2 I/O of a bridge circuit. |
| 24 | BVcc | – | Power pin of a bridge circuit |
| 25 | P50/ALR1 | B | General-purpose I/O port.<br>This pin is also used for alarm signal output when battery 1 runs down. |
| 26 | P51/ALR2 | | General-purpose I/O port.<br>This pin is also used for alarm signal output when battery 2 runs down. |
| 27 | P52/ALR3 | | General-purpose I/O port.<br>This pin is also used for alarm signal output when battery 3 runs down. |
| 28 | P53/AC0 | B | General-purpose I/O port.<br>This pin is also used for AC power set signal output. |
| 29 | P54/OFB1 | B | General-purpose I/O port.<br>This pin is also used for battery 1 discharge control signal output of the comparator. |
| 30 | P55/OFB2 | | General-purpose I/O port.<br>This pin is also used for battery 2 discharge control signal output of the comparator. |
| 31 | P56/OFB3 | | General-purpose I/O port.<br>This pin is also used for battery 3 discharge control signal output of the comparator. |

**Table 1.7-1 Pin Description (Continued)**

| Pin No. | Pin name | I/O Circuit type | Function |
|---|---|---|---|
| 32 | PB0/V0 | I | N-ch open-drain I/O pin<br>This pin is also used by the LCD driving power pin. |
| 33 | PB1/V1 | | |
| 34 | PB2/V2 | | |
| 35 | PB3/V3 | | |
| 36 | PB4/COM0 | J | N-ch open-drain I/O pin<br>This pin is also used by the LCD controller common output dedicated pin. |
| 37 | PB5/COM1 | | |
| 38 | PB6/COM2 | | |
| 39 | PB7/COM3 | | |
| 40 | PA0/SEG00 | J | N-ch open-drain I/O pin<br>This pin is also used by the LCD controller segment output dedicated pin. |
| 41 | PA1/SEG01 | | |
| 42 | PA2/SEG02 | | |
| 43 | PA3/SEG03 | | |
| 44 | PA4/SEG04 | | |
| 45 | PA5/SEG05 | | |
| 46 | PA6/SEG06 | | |
| 47 | Vcc | – | Power supply pin |
| 48 | PA7/SEG07 | J | N-ch open-drain I/O pin<br>This pin is also used by the LCD controller segment output dedicated pin. |
| 49 | PA60/SEG08 | J | N-ch open-drain I/O pin<br>This pin is also used by the LCD controller segment output pin. |
| 50 | PA61/SEG09 | | |
| 51 | PA62/SEG10 | | |
| 52 | PA63/SEG11 | | |
| 53 | PA64/SEG12/ UO2 | | N-ch open-drain I/O pin<br>This pin is also used by the LCD controller segment output pin/UART U0 pin. |
| 54 | PA65/SEG13/ UO1 | | |
| 55 | CVcc | – | Comparator power supply pin |
| 56 | CVss | – | Power supply pin (GND) |
| 57 | P70/DCIN | N | General-purpose I/O port.<br>This pin is also used for AC power monitoring input of the comparator. |
| 58 | P71/DCIN2 | | |
| 59 | P72/VOL1 | | General-purpose I/O port.<br>This pin is also used for battery 1 power instantaneous interruption monitoring input of the comparator. |

**Table 1.7-1 Pin Description (Continued)**

| Pin No. | Pin name | I/O Circuit type | Function |
|---|---|---|---|
| 60 | P73/VS11 | N | General-purpose I/O port<br>This pin is also used for battery 1 indicator monitoring input of the comparator. |
| 61 | P74/VOL2 | | General-purpose I/O port<br>This pin is also used for battery 2 power instantaneous interruption monitoring input of the comparator. |
| 62 | P75/VSI2 | | General-purpose I/O port<br>This pin is also used for battery 2 indicator monitoring input of the comparator. |
| 63 | P76/VOL3 | | General-purpose I/O port<br>This pin is also used for battery 3 power instantaneous interruption monitoring input of the comparator. |
| 64 | P77/VSI3 | | General-purpose I/O port<br>This pin is also used for battery 3 indicator monitoring input of the comparator. |
| 65 | CVRL | – | Standard power input pin of the comparator |
| 66 | CVRH1 | | |
| 67 | CVRH2 | | |
| 68 | P80/INT0 | K | General-purpose I/O port<br>These pins are also used for external interrupts.<br>When an external interrupt occurs, it is hysteresis input. |
| 69 | P81/INT1 | | |
| 70 | P82/INT2 | | |
| 71 | P83/INT3 | | |
| 72 | P84/EC | O | General-purpose input port<br>This pin is also used by EC of the 16-bit timer and 8/16-bit timer. |
| 73 | AVss | – | Power (GND) pin |
| 74 | P85/AN0/SW1 | L | General-purpose I/O port<br>This pin is also used for analog input/comparator input. |
| 75 | P86/AN1/SW2 | | |
| 76 | P87/AN2/SW3 | | |
| 77 | P90/AN3 | E | General-purpose I/O port<br>This pin is also used for analog input. |
| 78 | P91/DA1 | M | General-purpose I/O port<br>This pin is also used for D/A converter output. |
| 79 | P92/DA2 | | |
| 80 | AVcc | – | Power supply pin of the A/D and D/A converters |

**Table 1.7-1 Pin Description (Continued)**

| Pin No. | Pin name | I/O Circuit type | Function |
|---|---|---|---|
| 81 | P10/AN4 | E | General-purpose I/O port<br>This pin is also used for analog input. |
| 82 | P11/AN5 | | |
| 83 | P12/AN6 | | |
| 84 | P13AN7 | | |
| 85 | P14/AN8 | | |
| 86 | P15/AN9 | | |
| 87 | P16/AN10 | | |
| 88 | P17/AN11 | | |
| 89 | AVR | – | Reference input pin of the A/D converter |
| 90 | P00 | B | General-purpose I/O port |
| 91 | P01 | | |
| 92 | P02 | | |
| 93 | P03 | | |
| 94 | P04 | | |
| 95 | P05 | | |
| 96 | P06 | | |
| 97 | P07 | | |
| 98 | Vcc | – | Power supply pin |
| 99 | X1A | C | Crystal oscillator pin (sub clock) |
| 100 | X0A | | |

# 1.8 I/O Circuit Type

**Table 1.8-1 "I/O Circuit Type" list the I/O circuit forms. Alphabets in the classification field of Table 1.8-1 "I/O Circuit Type" correspond to those in the I/O circuit form field of Table 1.7-1 "Pin Description".**

## ■ I/O Circuit Type

**Table 1.8-1 I/O Circuit Type**

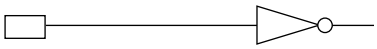
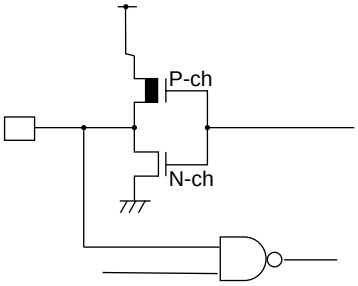
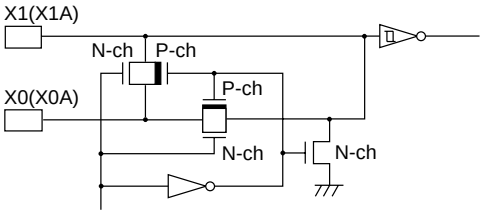
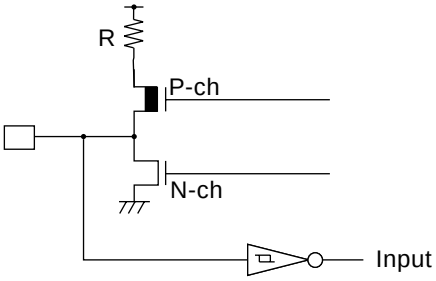
| Classification | Circuit | Remarks |
|---|---|---|
| A |  | • CMOS input |
| B |  | • CMOS I/O |
| C |  | • Oscillation feedback resistor<br>Main clock : approximately 1 MΩ<br>Sub clock : approximately 4.5 MΩ |
| D |  | • Reset input/output pin<br>• Output pull-up resistor (P-ch): approximately 50 kΩ<br>• Hysteresis input |

**Table 1.8-1 I/O Circuit Type (Continued)**

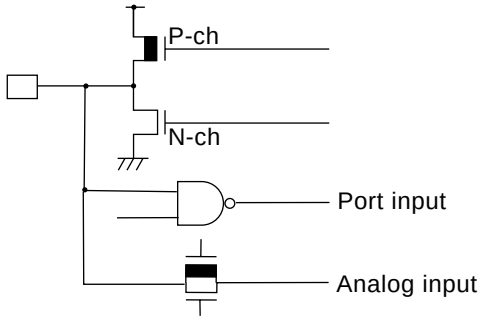
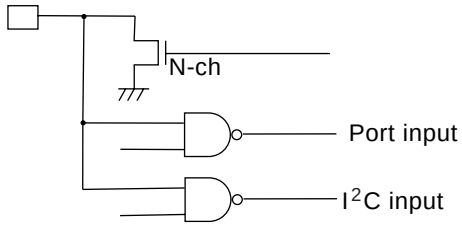
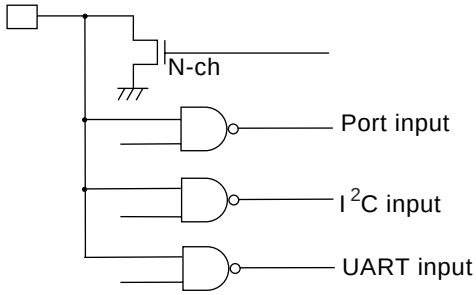
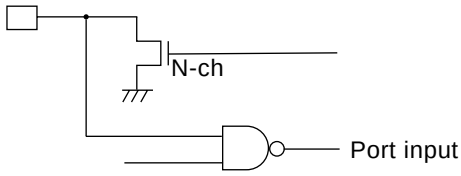
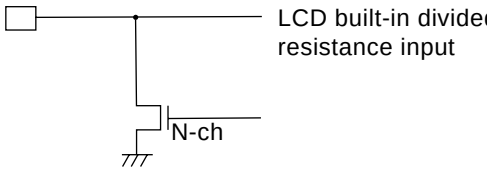
| Classification | Circuit | Remarks |
|---|---|---|
| E |  | • CMOS I/O<br>• Analog input |
| F |  | • N-ch open-drain output<br>• CMOS I/O<br>• I2C input |
| G |  | • N-ch open-drain output<br>• CMOS I/O<br>• I2C input<br>• UART input |
| H |  | • N-ch open-drain I/O |
| I |  | • N-ch open-drain I/O<br>• LCD built-in divided resistance input |

**Table 1.8-1 I/O Circuit Type (Continued)**

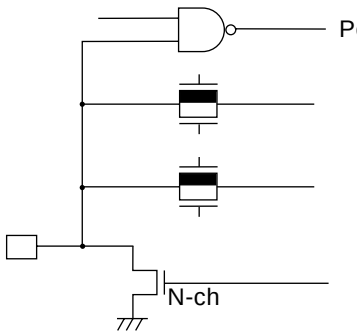
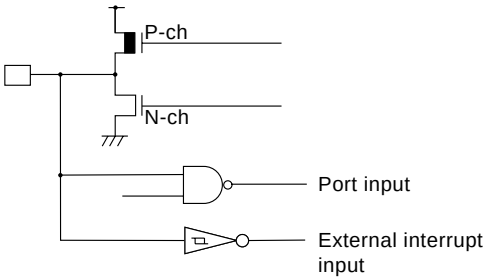
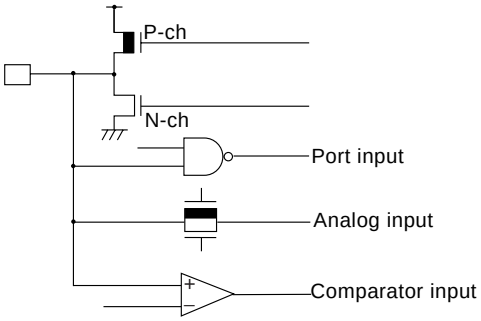
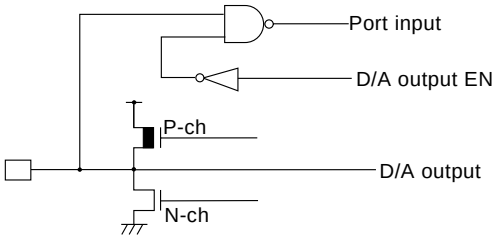
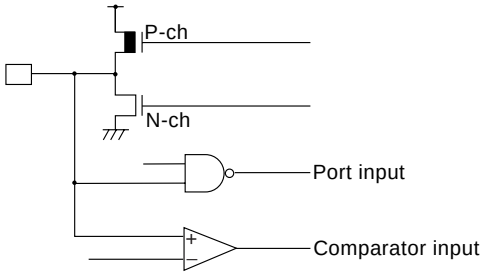
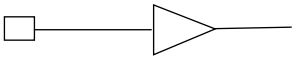
| Classification | Circuit | Remarks |
|---|---|---|
| J |  | • CMOS input<br>• LCD output<br>• N-ch open-drain I/O |
| K |  | • CMOS input<br>• Hysteresis input (during the input of an external interrupt) |
| L |  | • CMOS I/O<br>• Analog input<br>• Comparator input |
| M |  | • CMOS I/O<br>• D/A converter output |

**Table 1.8-1 I/O Circuit Type (Continued)**

| Classification | Circuit | Remarks |
|---|---|---|
| N |  | • CMOS I/O<br>• Comparator input |
| O |  | • CMOS input |

# CHAPTER 2 HANDLING DEVICE

**This chapter describes the precautions to be taken when using the MB89570 series.**

2.1 "Notes on Handling Devices"

# 2.1 Notes on Handling Devices

**This section describes the precautions to be taken when handling the power supply voltage and pins of the device.**

## ■ Notes on Handling Devices

### ❍ Preventing Latchup

Latchup may occur on CMOS ICs if voltage higher than $V_{CC}$ or lower than $V_{SS}$ is applied to input and output pins other than medium- to high-voltage pins, or if voltage higher than ratings is applied between $V_{CC}$ and $V_{SS}$.

When latchup occurs, power supply current increases rapidly and might thermally damage elements. When using, take great care not to exceed the absolute maximum ratings.

Also, take care to prevent the analog power supply ($AV_{CC}$, $BV_{CC}$, $CV_{CC}$, AVR, CVRH1, CVRH2, and CVRL) and analog input from exceeding the digital power supply ($V_{CC)}$ when the analog system power supply is turned on and off.

### ❍ Treatment of Unused Input Pins

Leaving unused input pins open could cause malfunctions. They should be connected to a pull-up or pull-down resistor.

### ❍ Treatment of Power Supply Pins on Microcontroller with A/D and D/A Converters

Connect to be $AV_{CC}$ = $BV_{CC}$= $CV_{CC}$ = $V_{CC}$ and $AV_{SS}$ = AVR = $CV_{SS}$ = CVRL = CVRH1 = CVRH2 = $V_{SS}$ even if the A/D and D/A converters are not in use.

### ❍ Treatment of Unused Input Pins

Be sure to leave (internally connected) N.C. pins open.

### ❍ Power Supply Voltage Fluctuations

Although $V_{CC}$ power supply voltage is assured to operate within the rated range, a rapid fluctuation of voltage could cause malfunctions, even if it occurs within the rated range. Stabilizing voltage supplied to the IC is therefore important. As stabilization guidelines, it is recommended to control power so that $V_{CC}$ ripple fluctuations (P-P value) will be less than 10% of the standard $V_{CC}$ value at the commercial frequency (50 to 60 Hz) and the transient fluctuation rate will be less than 0.1 V/ms at the time of a momentary fluctuation such as when power is switched.

### ❍ Precautions when Using an External Clock

Even when an external clock is used, oscillation stabilization time is required for power-on reset and wake-up from stop mode.

# CHAPTER 3 CPU

**This chapter describes the functions and operations of the CPU.**


3.1 "Memory Space"

3.2 "Dedicated Registers"

3.3 "General-purpose Registers"

3.4 "Interrupts"

3.5 "Resets"

3.6 "External Reset Pin"

3.7 "Clock"

3.8 "Standby Mode (Low Power Consumption)"

3.9 "Memory Access Mode"

# 3.1 Memory Space

**The memory space of the MB89570 series is 64 Kbytes and is made up of the I/O area, RAM area, ROM area, and external area.**
**Some areas in the memory space, such as the general-purpose registers and vector table, are used for specific applications.**

## ■ Configuration of the Memory Space

### ❍ I/O area (address: $0000_H$ - $007F_H$)

- This area is allocated to the control registers and data registers of the built-in peripheral devices.
- Since the I/O area is allocated to a part of the memory space, it can be accessed like normal memory. The area can be accessed faster using direct addressing.

### ❍ RAM area

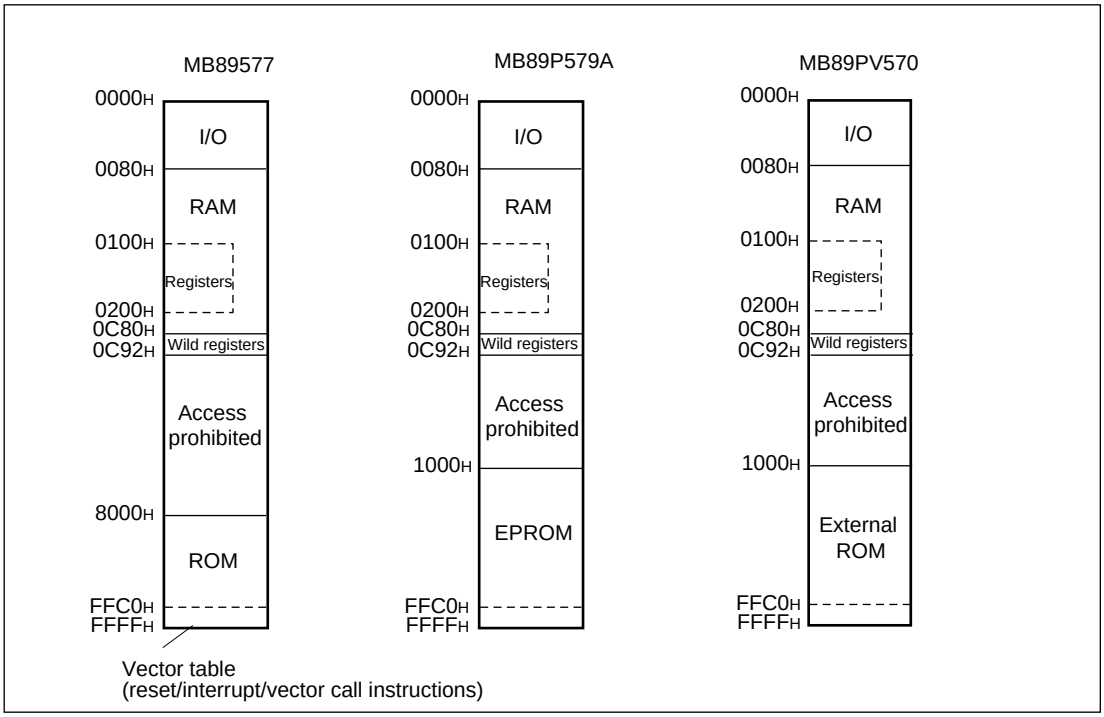
- Static RAM is contained as a built-in data area.
- The internal RAM size is dependent on the part number.
- $80_H$ to $FF_H$ can be accessed faster using direct addressing (Depending on the part number, the available area may be limited).
- $100_H$ to $1FF_H$ can be used as a general-purpose register area.
- If a reset occurs during a write operation to RAM, data at the address to which data is being written cannot be guaranteed.

### ❍ ROM area

- ROM is contained as an internal program area.
- The internal ROM size is dependent on the part number.
- $FFC0_H$ to $FFFF_H$ are used as, for example, a vector table.

## ■ Memory Map

**Figure 3.1-1 Memory Map**



## 3.1.1 Special Areas

**In addition to the I/O area, the general-purpose register area and vector table area are available as areas for specific applications.**

■ **General-purpose Register Area (Address: $0100_H$ - $01FF_H$)**

- This area is used for 8-bit arithmetic operations and transfer. Supplementary registers are provided.
- Since this area is allocated to a part of the RAM area, it can also be used as normal RAM.
- When this area is used as a general-purpose register, it can be accessed faster using shorter instructions by general-purpose register addressing.

For details, see Section 3.2.2 "Register bank pointer (RP)" and Section 3.3 "General-purpose Register".

■ **Vector Table Area (Address: $FFC0_H$ - $FFFF_H$)**

- This area is used as vector tables of the vector call instructions, interrupts, and reset.
- This area is allocated to the highest ranges of the ROM area, and the start address of the corresponding processing routine is set to the address of each vector table.

Table 3.1-1 "Vector Table" lists the addresses of the vector tables referenced corresponding to the vector call instructions, interrupts, and reset.

For details, see Section 3.4 "Interrupts", Section 3.5 "Reset", and "(6) CALLV #vct" of Appendix B.3 "Special Instructions".

**Table 3.1-1 Vector Table**

| Vector call instruction | Vector table address | |
|---|---|---|
| | High | Low |
| CALLV #0 | $FFC0_H$ | $FFC1_H$ |
| CALLV #1 | $FFC2_H$ | $FFC3_H$ |
| CALLV #2 | $FFC4_H$ | $FFC5_H$ |
| CALLV #3 | $FFC6_H$ | $FFC7_H$ |
| CALLV #4 | $FFC8_H$ | $FFC9_H$ |
| CALLV #5 | $FFCA_H$ | $FFCB_H$ |
| CALLV #6 | $FFCC_H$ | $FFCD_H$ |
| CALLV #7 | $FFCE_H$ | $FFCF_H$ |

**Table 3.1-1 Vector Table (Continued)**

| Interrupt name | Vector table address | |
|---|---|---|
| | High | Low |
| IRQF | $FFDC_H$ | $FFDD_H$ |
| IRQE | $FFDE_H$ | $FFDF_H$ |
| IRQD | $FFE0_H$ | $FFE1_H$ |
| IRQC | $FFE2_H$ | $FFE3_H$ |
| IRQB | $FFE4_H$ | $FFE5_H$ |
| IRQA | $FFE6_H$ | $FFE7_H$ |
| IRQ9 | $FFE8_H$ | $FFE9_H$ |
| IRQ8 | $FFEA_H$ | $FFEB_H$ |
| IRQ7 | $FFEC_H$ | $FFED_H$ |
| IRQ6 | $FFEE_H$ | $FFEF_H$ |
| IRQ5 | $FFF0_H$ | $FFF1_H$ |
| IRQ4 | $FFF2_H$ | $FFF3_H$ |
| IRQ3 | $FFF4_H$ | $FFF5_H$ |
| IRQ2 | $FFF6_H$ | $FFF7_H$ |
| IRQ1 | $FFF8_H$ | $FFF9_H$ |
| IRQ0 | $FFFA_H$ | $FFFB_H$ |
| Mode data | — * | $FFFD_H$ |
| Reset vector | $FFFE_H$ | $FFFF_H$ |

*: $FFFC_H$ is not available (Set $FF_H$)

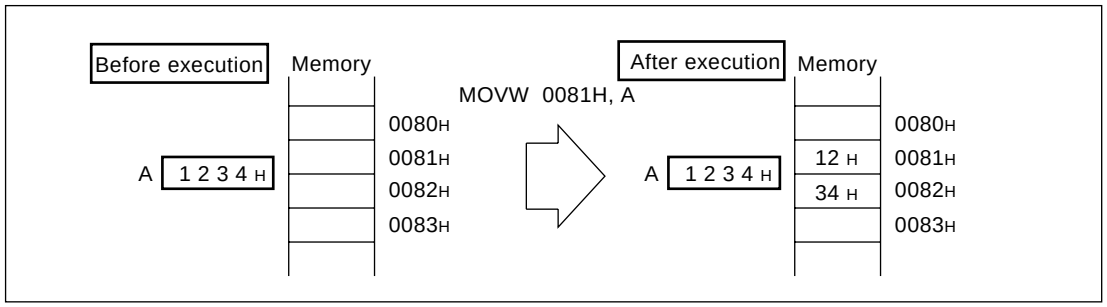
## 3.1.2 Storing 16-bit Data in Memory

**Higher data of 16-bit data and stacks are stored in the areas of smaller address values on memory.**

### ■ Storage of 16-bit Data on RAM

When writing 16-bit data into memory, the higher byte of the data is stored at the lower address. The lower byte of the data is stored at the next address. When reading memory, the same procedure is executed.Figure 3.1-2 "Storing 16-bit Data in Memory" shows the storing 16-bit data in memory.

**Figure 3.1-2 Storing 16-bit Data in Memory**



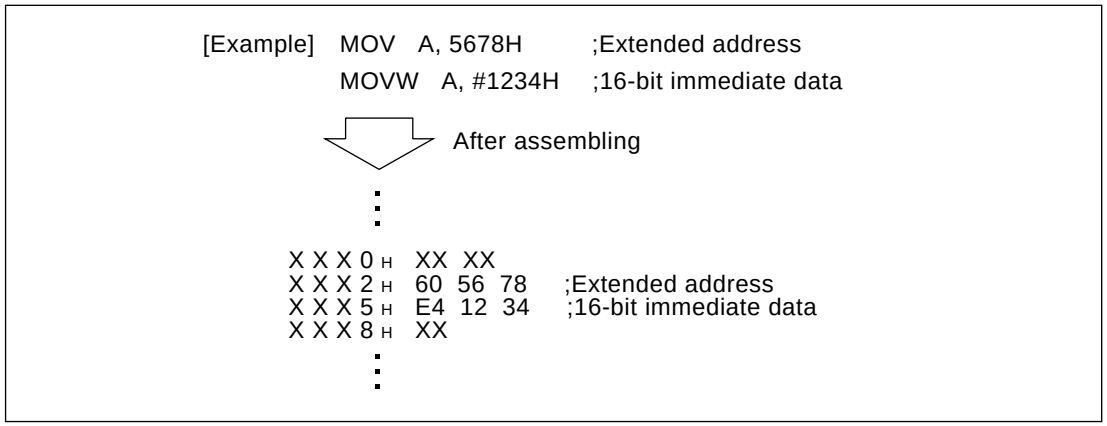

### ■ Storage of a 16-bit Operand

Also when 16 bits are specified in an operand of an instruction, the higher byte is stored at the nearby operation code (instruction) and the lower byte is stored at the next address.

This is the same if the operand points to a memory address or is 16-bit immediate data.

Figure 3.1-3 "16-bit Data in Instructions" shows the storing 16-bit data in instructions.

**Figure 3.1-3 16-bit Data in Instructions**



### ■ Storage of 16-bit Data on the Stack

Data of the 16-bit length register saved on the stack due, for example, to an interrupt, is also stored in the same manner, with the higher byte at the smaller address.

# 3.2 Dedicated Registers

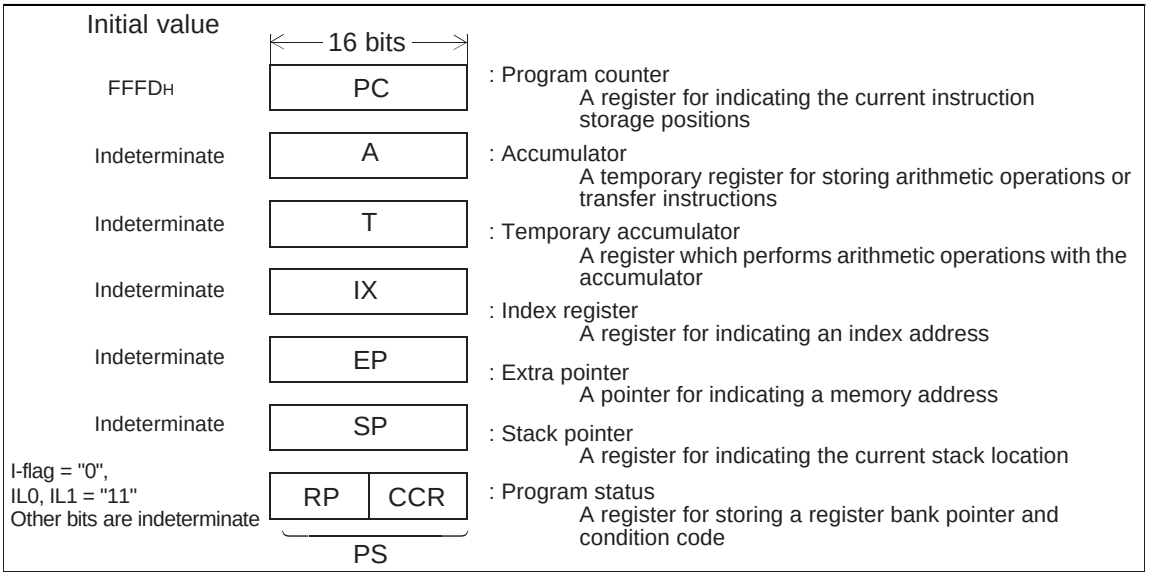
**The dedicated registers in the CPU consist of the program counter (PC), two arithmetic operation registers (A and T), three address pointers (IX, EP, and SP), and the program status (PS). All registers are 16 bits.**

## ■ Dedicated Register Configuration

The dedicated registers in the CPU consist of seven 16-bit registers. Some of these registers are also able to be used as 8-bit registers, using the lower 8 bits only.

Figure 3.2-1 "Dedicated Register Configuration" shows the structure of the dedicated registers.

**Figure 3.2-1 Dedicated Register Configuration**




## ■ Dedicated Register Functions

### ❍ Program counter (PC)

The program counter is a 16-bit counter that indicates the memory address of the instruction currently being executed by the CPU. Instruction execution, interrupts, resets, and similar update the contents of the program counter. The initial value during a reset is the read address of the mode data ($FFFD_H$).

### ❍ Accumulator (A)

The accumulator is a 16-bit arithmetic operation register. The accumulator is used to perform arithmetic operations and data transfers with data in memory or in other registers such as the temporary accumulator (T). The content of the accumulator can be treated as either word (16-bit) or byte (8-bit) data. Only the lower 8 bits (AL) of the accumulator are used for byte arithmetic operations or transfers. In this case, the upper 8 bits (AH) remain unchanged. The content of the accumulator after a reset is indeterminate.

❍ **Temporary accumulator (T)**

The temporary accumulator is an auxiliary 16-bit arithmetic operation register used to perform arithmetic operations with the data in the accumulator (A). The content of the temporary accumulator is treated as word data (16-bit) for word-length arithmetic operations with the accumulator and as byte data (8-bit) for byte-length arithmetic operations. For byte-length arithmetic operations, only the lower 8 bits of the temporary accumulator (TL) are used and the upper 8 bits (TH) are not used.

Executing a transfer instruction to transfer data to the accumulator (A) automatically transfer the previous content of the accumulator to the temporary accumulator. In this case also, a byte transfer leaves the upper 8 bits of the temporary accumulator (TH) unchanged. The content of the temporary accumulator after a reset is indeterminate.

❍ **Index register (IX)**

The index register is a 16-bit register used to hold the index address. The index register is used in conjunction with a single byte offset value (-128 to +127). Adding the sign-extended offset value to the index address generates the memory address for data access. The content of the index register after a reset is indeterminate.

❍ **Extra pointer (EP)**

The extra pointer is a 16-bit register used to hold a memory address for data access. The content of the extra pointer after a reset is indeterminate.

❍ **Stack pointer (SP)**

The stack pointer is a 16-bit register used to hold the address referenced during operations such as interrupts, subroutine calls, and the stack save and restore instructions. The value of the stack pointer during program execution is the address of the most recently saved data on the stack. The content of the stack pointer after a reset is indeterminate.

❍ **Program status (PS)**

The program status is a 16-bit control register. The upper 8 bits contain the register bank pointer (RP) which points to the address of the current general-purpose register bank.

The lower 8 bits contain the condition code register (CCR) which contains flags indicating the current CPU status. The two 8-bit registers which form the program status cannot be accessed independently (the program status can only be accessed by the MOVW A,PS and MOVW PS,A instructions).

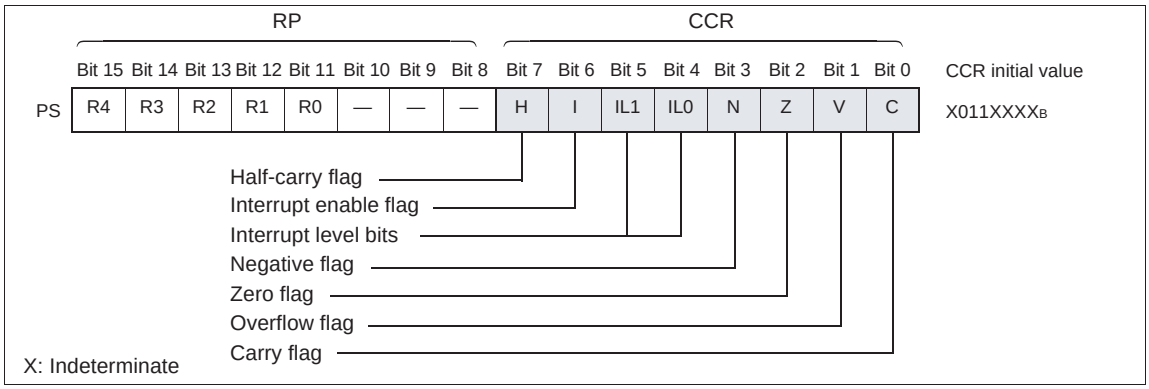
Refer to the "$F^2$MC-8L Programming Manual" for details on using the dedicated registers

## 3.2.1 Condition Code Register (CCR)

**The condition code register (CCR) located in the lower 8 bits of the program status (PS) consists of the C, V, Z, N, and H bits indicating the results of arithmetic operations and the contents of transfer data, and the I, IL1, and IL0 bits for control whether or not the CPU accepts interrupt requests.**

### ■ Structure of Condition Code Register (CCR)

**Figure 3.2-2 Structure of Condition Code Register**




### ■ Arithmetic Operation Result Bits

❍ **Half-carry flag (H)**

Set when a carry from bit 3 to bit 4 or a borrow from bit 4 to bit 3 occurs as a result of an arithmetic operation. Cleared otherwise. As this flag is for the decimal adjustment instructions, do not use this flag in cases other than addition or subtraction.

❍ **Negative flag (N)**

Set if the most significant bit (MSB) is set to 1 as a result of an arithmetic operation. Cleared when the bit is set to 0.

❍ **Zero flag (Z)**

Set when an arithmetic operation results in 0. Cleared otherwise.

❍ **Overflow flag (V)**

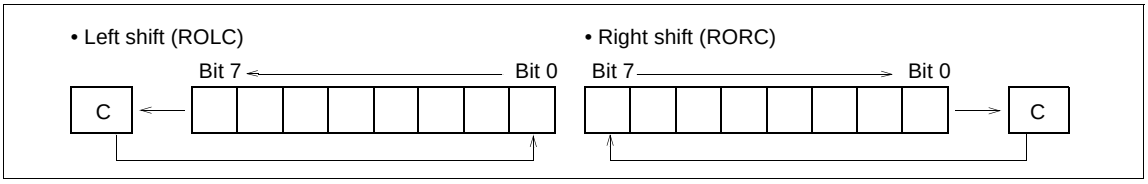
Set if the complement on 2 overflows as a result of an arithmetic operation. Reset if the overflow does not occur.

❍ **Carry flag (C)**

Set when a carry from bit 7 or borrow to bit 7 occurs as a result of an arithmetic operation. Cleared otherwise. Set to the shift-out value in case of a shift instruction.

Figure 3.2-3 "Change of Carry Flag by Shift Instruction" shows the change of the carry flag by a shift instruction.

**Figure 3.2-3 Change of Carry Flag by Shift Instruction**




**Note:**

The condition code register is part of the program status (PS) and cannot be accessed independently.

**Reference:**

In practice, the flag bits are rarely fetched and used directly. Instead, the bits are used indirectly by instructions such as branch instructions (such as BNZ) or the decimal adjustment instructions (DAA, DAS). The content of the flags after a reset is indeterminate.

## ■ Interrupt Acceptance Control Bit

### ❍ Interrupt enable flag (I)

Interrupt is enabled when this flag is set to "1" and the CPU accepts interrupt. Interrupt is prohibited when this flag is set to "0" and the CPU does not accept interrupt.

The initial value after a reset is "0".

Normal practice is to set the flag to "1" by the SETI instruction and clear to "0" by the CLRI instruction.

### ❍ Interrupt level bits (IL1, IL0)

These bits indicate the level of the interrupt currently being accepted by the CPU. The value is compared with the interrupt level setting registers (ILR1 to ILR3) which have a setting for each peripheral function interrupt request (IRQ0 to IRQB).

Given that the interrupt enable flag is enabled (I = "1"), the CPU only performs interrupt processing for interrupt requests with an interrupt level value that is less than the value of these bits. Table 3.2-1 "Interrupt Level" lists the interrupt level priorities. The initial value after a reset is "11".

**Table 3.2-1 Interrupt Level**

| IL1 | IL0 | Interrupt level | High-low |
|---|---|---|---|
| 0 | 0 | 1 | High ↕ Low (no interrupt) |
| 0 | 1 | | |
| 1 | 0 | 2 | |
| 1 | 1 | 3 | |

**Reference:**

The interrupt level bits (IL1, IL0) are normally "11" when the CPU is not processing an interrupt (during main program execution).

see Section 3.4 "Interrupts" for details on interrupts.

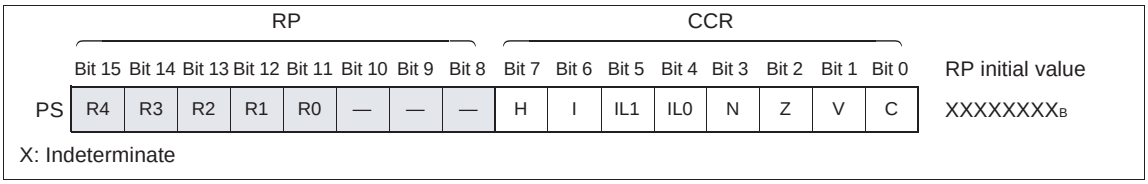
## 3.2.2 Register Bank Pointer (RP)

**The register bank pointer (RP) located in the upper 8 bits of the program status (PS) indicates the address of the general-purpose register bank currently in use. The RP is converted to form the actual address in general-purpose register addressing.**

### ■ Structure of Register Bank Pointer (RP)

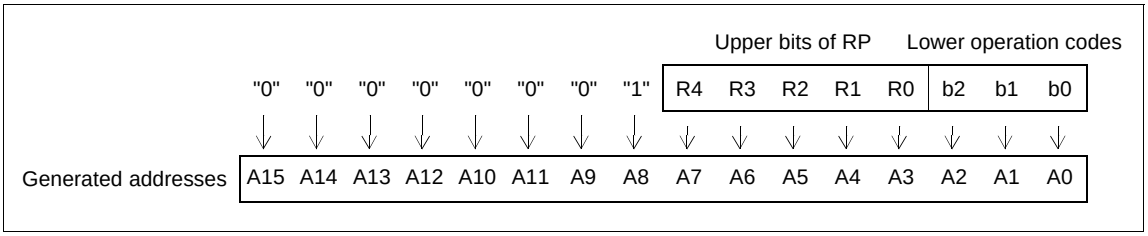
Figure 3.2-4 "Structure of Register Bank Pointer" shows the structure of the register bank pointer.

**Figure 3.2-4 Structure of Register Bank Pointer**




The register bank pointer indicates the address of the register bank currently in use. Figure 3.2-5 "Rule for Conversion of Actual Addresses of General-purpose Register Area" shows the relationship between the pointer contents and the actual address is based on the conversion rule.

**Figure 3.2-5 Rule for Conversion of Actual Addresses of General-purpose Register Area**




The register bank pointer points to the memory block (register bank) in the RAM area that is used for general-purpose registers. A total of 32 register banks are available. A register bank is specified by setting a value between 0 and 31 in the upper 5 bits of the register bank pointer. Each register bank contains 8-bit general-purpose registers. Registers are specified by the lower 3 bits of the operation codes.

Using the register bank pointer, the addresses $0100_H$ to $01FF_H$ can be used as the general-purpose register area. However, the available area is limited on some products if internal RAM only is used. The initial value after a reset is indeterminate.

**Note:**

The register bank pointer is part of the program status (PS) and cannot be accessed independently.

# 3.3 General-purpose Registers

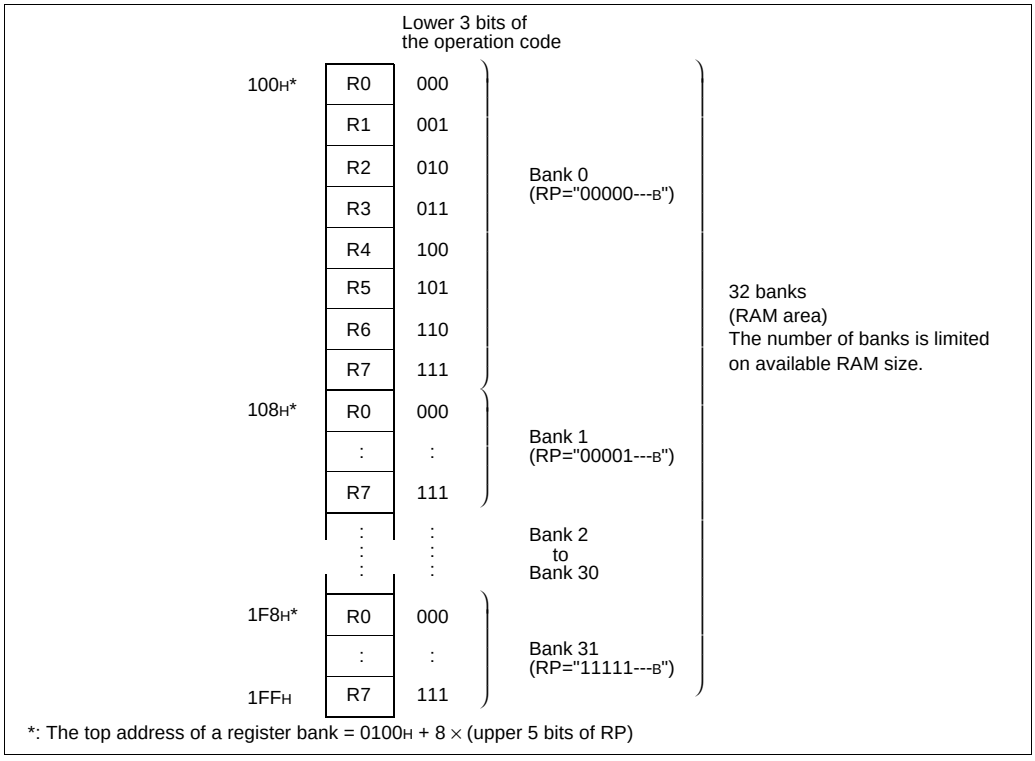
**The general-purpose registers are a memory block made up of banks, with 8 x 8-bit registers per bank.**
**The register bank pointer (RP) is used to specify the register bank.**
**The function permits the use of up to 32 banks, but the number of banks that can actually be used depends on how much RAM the device has.**
**Register banks are valid for interrupt processing, vector call processing, and subroutine calls.**

## ■ Structure of General-purpose Registers

- The general-purpose registers are 8 bits and located in the register banks of the general-purpose register area (in RAM).
- One bank contains eight registers (R0 to R7) and up to a total of 32 banks. However, the number of banks available for general-purpose registers is limited on some products if internal RAM only is used.
- The register bank currently in use is specified by the register bank pointer (RP). The lower three bits of the operation code specify general-purpose register 0 (R0) to general-purpose register 7 (R7).

Figure 3.3-1 "Register Bank Structure" shows the register bank structure.

**Figure 3.3-1 Register Bank Structure**



see Section 3.1.1 "Special Areas" for the general-purpose register area available for each product.

## ■ Features of General-purpose Registers

General-purpose registers have the following features:

- RAM can be accessed at high-speed using short instructions (general-purpose register addressing).
- Registers are grouped in blocks in the form of register banks. This simplifies the process of saving register contents and dividing registers by function.

Dedicated register banks can be permanently assigned for each interrupt processing or vector call (CALLV #0 to #7) processing routine by general-purpose register. For example, register bank 4 interrupt 2.

For example, a particular interrupt processing routine only uses a particular register bank which cannot be written to unintentionally by other routines. The interrupt processing routine only needs to specify its dedicated register bank at the start of the routine to effectively save the general-purpose registers in use prior to the interrupt. Therefore, saving the general-purpose registers to the stack or other memory location is not necessary. This allows high-speed interrupt handling while maintaining simplicity.

Also, as an alternative to saving general-purpose registers in subroutine calls, register banks can be used to create reentrant programs (programs that do not use fixed addresses and can be entered more than once) usually made by the index register (IX).

**Note:**

If an interrupt processing routine changes the register bank pointer (RP), ensure that the program does not also change the interrupt level bits in the condition code register (CCR: IL1, IL0) when specifying the register bank.

# 3.4 Interrupts

**The MB89570 series has 16 interrupt request input corresponding to peripheral functions. An interrupt level can be set independently.**
**If an interrupt request output is enabled in the peripheral function, an interrupt request from a peripheral function is compared with the interrupt level in the interrupt controller. The CPU performs interrupt operation according to how the interrupt is accepted. The CPU wakes up from standby modes, and returns to the interrupt or normal operation.**

## ■ Interrupt Requests from Peripheral Functions

Table 3.4-1 "Interrupt Request and Interrupt Vector" lists the interrupt requests corresponding to the peripheral functions. On acceptance of an interrupt, execution branches to the interrupt processing routine. The contents of interrupt the vector table address corresponding to the interrupt request specifies the branch destination address for the interrupt processing routine.

An interrupt processing level can be for each interrupt request in the interrupt level setting registers (ILR1, ILR2, ILR3, ILR4). Three levels are available.

If an interrupt request with the same or lower level occurs during execution of an interrupt processing routine, the letter interrupt is not normally processed until the current interrupt processing routine completes. If interrupt request set the same level occur simultaneously, the highest priority is IRQ0.

**Table 3.4-1 Interrupt Requests and Interrupt Vectors**

| Interrupt request | Vector table address | | Bit name of the interrupt level setting register | Priority if interrupt requests with the same level occur simultaneously |
|---|---|---|---|---|
| | Higher | Lower | | |
| IRQ0 (external interrupt1) | $FFFA_H$ | $FFFB_H$ | L01, L00 | High ↓ Low |
| IRQ1 (external interrupt2) | $FFF8_H$ | $FFF9_H$ | L11, L10 | |
| IRQ2 (Unused) | $FFF6_H$ | $FFF7_H$ | L21, L20 | |
| IRQ3 (A/D converter) | $FFF4_H$ | $FFF5_H$ | L31, L30 | |
| IRQ4 (Converter1) | $FFF2_H$ | $FFF3_H$ | L41, L40 | |
| IRQ5 (Converter2) | $FFF0_H$ | $FFF1_H$ | L51, L50 | |
| IRQ6 (UART/SIO) | $FFEE_H$ | $FFEF_H$ | L61, L60 | |
| IRQ7 (Timebase timer) | $FFEC_H$ | $FFED_H$ | L71, L70 | |
| IRQ8 (Watch prescaler) | $FFEA_H$ | $FFEB_H$ | L81, L80 | |
| IRQ9 ($I^2C$) | $FFE8_H$ | $FFE9_H$ | L91, L90 | |
| IRQA ($I^2C$ timeout) | $FFE6_H$ | $FFE7_H$ | LA1, LA0 | |
| IRQB (Multi-address $I^2C$) | $FFE4_H$ | $FFE5_H$ | LB1, LB0 | |
| IRQC (Multi-address $I^2C$ timeout) | $FFE2_H$ | $FFE3_H$ | LC1, LC0 | |
| IRQD (16-bit timer/counter) | $FFE0_H$ | $FFE1_H$ | LD1, LD0 | |
| IRQE (8/16-bit timer) | $FFDE_H$ | $FFDF_H$ | LE1, LE0 | |
| IRQF (Unused) | $FFDC_H$ | $FFDD_H$ | LF1, LF0 | |

## 3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3, ILR4)

**The interrupt level setting registers (ILR1, ILR2, ILR3, ILR4) together contain 16 blocks of 2-bit data, with each data corresponding to an interrupt request from a peripheral function. The interrupt level for each interrupt is set in that interruptis corresponding 2-bit data (interrupt level setting bits).**

### ■ Structure of Interrupt Level Setting Registers (ILR1, ILR2, ILR3, ILR4)

**Figure 3.4-1 Structure of the Interrupt Level Setting Register**

| Register | Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|---|
| ILR1 | 0 0 7 BH | L31 | L30 | L21 | L20 | L11 | L10 | L01 | L00 | $11111111_B$ |
| | | W | W | W | W | W | W | W | W | |
| ILR2 | 0 0 7 CH | L71 | L70 | L61 | L60 | L51 | L50 | L41 | L40 | $11111111_B$ |
| | | W | W | W | W | W | W | W | W | |
| ILR3 | 0 0 7 DH | LB1 | LB0 | LA1 | LA0 | L91 | L90 | L81 | L80 | $11111111_B$ |
| | | W | W | W | W | W | W | W | W | |
| ILR4 | 0 0 7 EH | LF1 | LF0 | LE1 | LE0 | LD1 | LD0 | LC1 | LC0 | $11111111_B$ |
| | | W | W | W | W | W | W | W | W | |

W: write only

Two bits of the interrupt level setting registers are allocated to each interrupt request. The value of the interrupt level setting bits in these registers sets the interrupt priority (interrupt levels 1 to 3).

The interrupt level setting bits are compared with the interrupt level bits in the condition code register (CCR: IL1, IL0).

The CPU does not accept interrupt requests set to interrupt level 3. Table 3.4-2 "Interrupt Level Setting Bit and Interrupt Level" shows the relationship between the interrupt level setting bits and the interrupt levels.

**Table 3.4-2 Interrupt Level Setting Bits and the Interrupt Level**

| L01 to LF1 | L00 to LF0 | Request Interrupt level | High-low |
|---|---|---|---|
| 0 | 0 | 1 | High ↕ Low (no interrupt) |
| 0 | 1 | | |
| 1 | 0 | 2 | |
| 1 | 1 | 3 | |

**Reference:**

The interrupt level bits in the condition code register (CCR: IL1, IL0) are normally "11" during main program execution.

**Note:**

As the IRL1, ILR2, and ILR3 registers are write-only, the bit manipulation instructions cannot be used.

# 3.4.2 Interrupt Processing

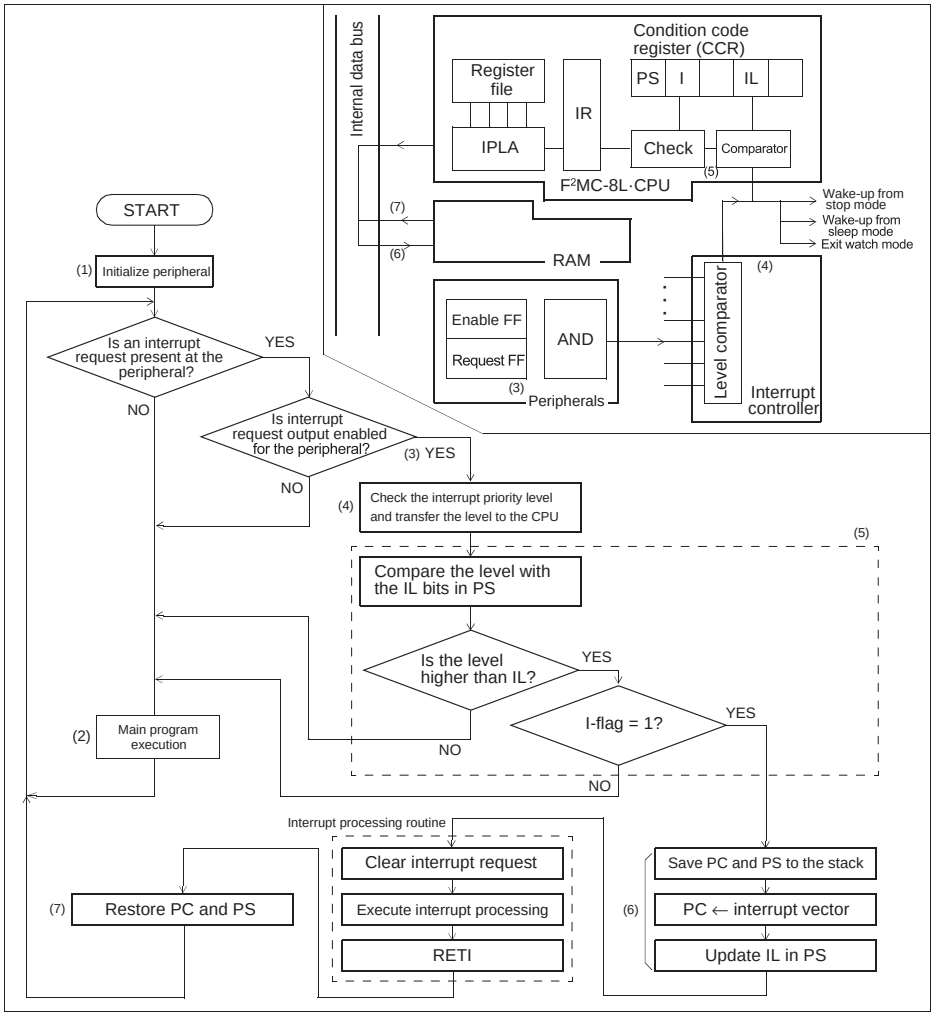
**The interrupt controller transmits the interrupt level to the CPU when an interrupt request is generated by a peripheral function. If the CPU is able to receive the interrupt, the CPU temporarily halts the currently executing program and executes the interrupt processing routine.**

## ■ Interrupt Processing

The procedure for interrupt operation is performed in the following order: interrupt source generated at peripheral function, set the interrupt request flag bit (request FF), discriminate the interrupt request enable bit (enable FF), the interrupt level (ILR1, ILR2, ILR3, ILR4 and CCR: IL1, IL0), simultaneously generated interrupt requests with the same level, then check the interrupt enable flag (CCR: I).

Figure 3.4-2 "Interrupt Processing" shows the interrupt processing.

**Figure 3.4-2 Interrupt Processing**



(1) After a reset, all interrupt requests are disabled.
Initialize the peripheral functions that are to generate interrupts in the peripheral function initialization program, set the interrupt levels in the appropriate interrupt level setting registers (ILR1, ILR2, ILR3, ILR4), and start peripheral function.

The interrupt level can be set to 1, 2 or 3. Level 1 is the highest priority, followed by level 2. Setting level 3 disables the interrupt for that peripheral function.

(2) Execute the main program (for multiple interrupts, execute the interrupt processing routine).

(3) The interrupt request flag bit (request FF) for a peripheral function is set to "1" when the peripheral function generates an interrupt source. If the interrupt request enable bit for the peripheral function is set to "enable" (enable FF = "1"), the peripheral function outputs the interrupt request to the interrupt controller.

(4) The interrupt controller continuously monitors for interrupt requests from the peripheral functions and passes the interrupt level of the current interrupt request with the highest interrupt level to the CPU. The interrupt controller also evaluates the priority order if requests with the same level are present simultaneously.

(5) If the interrupt level received by the CPU has a higher priority (a lower level value) than the level set in the interrupt level bits in the condition code register (CCR: IL1, IL0), the CPU checks the interrupt enable flag (CCR: I) and receives the interrupt if interrupts are enabled (CCR: I = "1").

(6) The CPU saves the contents of the program counter (PC) and program status (PS) on the stack, reads the top address of the interrupt processing routine from the interrupt vector table for the interrupt, updates the interrupt level bits in the condition code register (CCR: IL1, IL0) with the received interrupt level, and starts execution of the interrupt processing routine.

(7) Finally, on execution of the RETI instruction, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution from the instruction following the last instruction executed before the interrupt.

**Note:**

As the interrupt request flag bit of a peripheral function is not cleared automatically when an interrupt request is received, the bit must be cleared by the program (normally, by writing "0" to the interrupt request flag bit) at interrupt processing routine.

An interrupt wakes up the CPU from standby mode (low-power consumption). see Section 3.8 "Standby Modes (Low-power Consumption)" for details.

**Reference:**

If the interrupt request flag bit is cleared at the top of the interrupt processing routine, the peripheral function that has generated the interrupt becomes able to generate another interrupt during execution of the interrupt processing routine (resetting the interrupt request flag bit). However, the interrupts are not normally accepted until the current processing routine completes.

# 3.4.3 Multiple Interrupts

**Multiple interrupts can be performed by setting different interrupt levels to the interrupt level setting register for two or more interrupt requests from peripheral functions.**

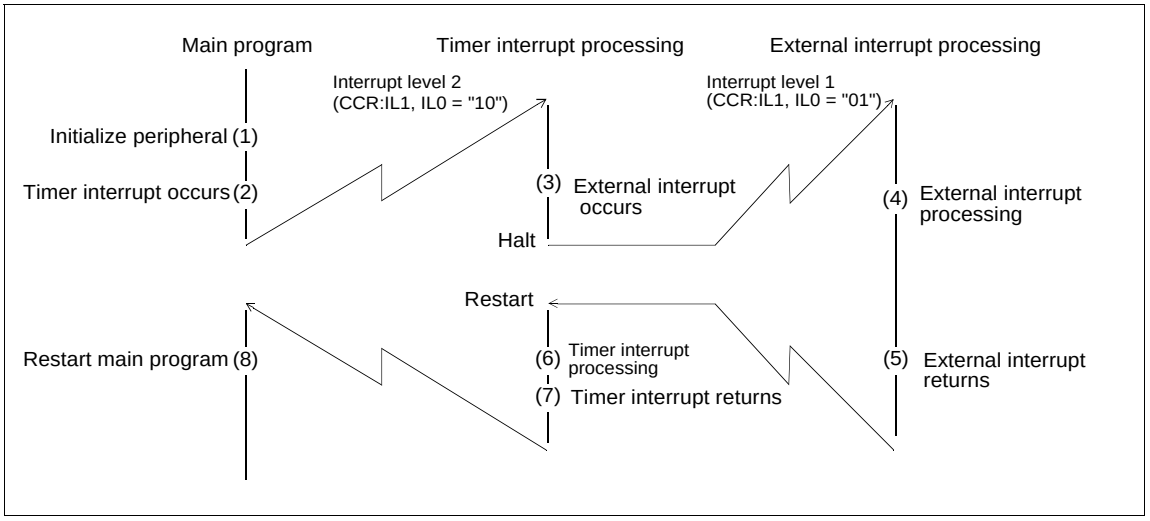
## ■ Multiple Interrupts

If the interrupt request having the higher interrupt levels occurs during the interrupt processing routines, the CPU halts the current interrupt process and switches to accept the interrupt with the higher priority. Interrupt levels can be set in the range 1 to 3. However, the CPU does not accept interrupt requests set to interrupt level 3.

### ❍ Example of multiple interrupts

As an example of multiple interrupt processing, assume that an external interrupt has a higher priority than the timer interrupt. The timer interrupt is set to level 2 and the external interrupt is set to level 1. Figure 3.4-3 "Example of Multiple Interrupts" shows the processing when the external interrupt occurs during execution of timer interrupt processing.

**Figure 3.4-3 Example of Multiple Interrupts**




- During execution of timer interrupt processing, the interrupt level bits in the condition code register (CCR:IL1, IL0) are automatically set to the same value as the interrupt level setting register (ILR1, ILR2, ILR3, ILR4) corresponding to the timer interrupt (level 2 in this example). If the interrupt request set to higher interrupt level (level 1 in this example) occurs at this time, the interrupt processing has priority.
- To temporarily disable multiple interrupts during the timer interrupt, the interrupt enable flag in the condition code register is set to "interrupts disabled" (CCR: I = "0") or the interrupt level bits (IL1, IL0) set to "00".
- On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution of the interrupted program.

  Restoring the program status (PS) returns the condition code register (CCR) to the value prior to the interrupt.

# 3.4.4 Interrupt Processing Time

**The total time from the generation of an interrupt request until control passes to the interrupt processing routine is the sum of the time required to complete execution of the current instruction and the interrupt handling time (the time required to prepare for interrupt processing). The maximum time for this process is 30 instruction cycles.**

## ■ Interrupt Processing Time

When an interrupt request occurs, the time until the interrupt is accepted and the interrupt processing routine is executed includes the interrupt request sampling time and the interrupt handling time.

### ❍ Interrupt request sampling time

Whether or not an interrupt request has occurred is determined by sampling and testing for interrupt requests during the final cycle of each instruction. Therefore, the CPU is unable to identify interrupt requests during execution of an instruction. The longest delay occurs when an interrupt request is generated immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles).

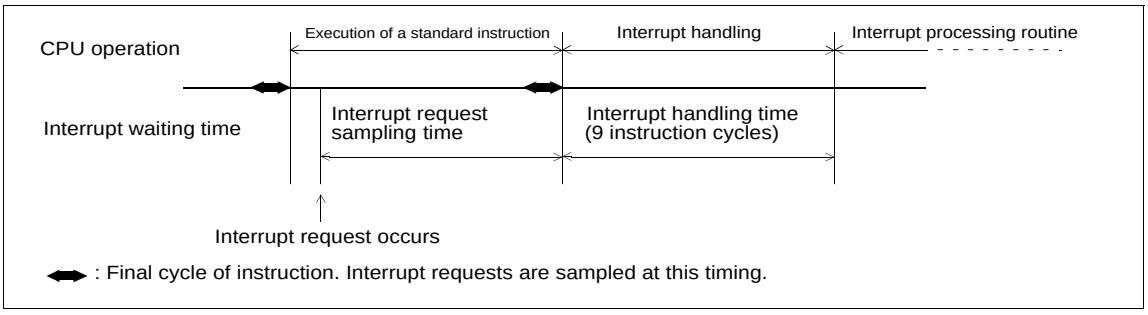
### ❍ Interrupt handling time

Nine instruction cycles are required to perform the following preparation for interrupt processing after the CPU accepts an interrupt request:

- Save the program counter (PC) and program status (PS).
- Set the top address of the interrupt processing routine (the interrupt vector) in the PC.
- Update the interrupt level bits (PS:CCR: IL1, IL0) in the program status (PS).

Figure 3.4-4 "Interrupt Processing Time" shows the interrupt processing time.

**Figure 3.4-4 Interrupt Processing Time**




The total interrupt processing time of 21 + 9 = 30 instruction cycles is required if an interrupt request occurs immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles). If, on the other hand, the program does not use the DIVU or MULU instructions, the maximum interrupt processing time is 6 + 9 = 15 instruction cycles.

The time of one instruction cycle changes with the clock mode and the main clock frequency as selected by the "speed-shift" (gear) function. see Section 3.7 "Clock" for details.

# 3.4.5 Stack Operation during Interrupt Processing

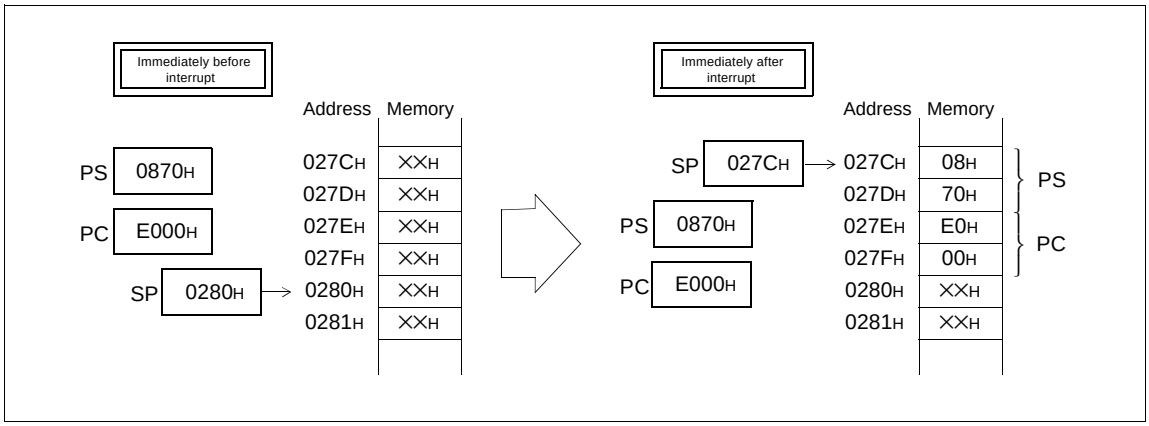
**This section describes the saving of the register contents to the stack and restore operation during interrupt processing.**

## ■ Stack Operation at Start of Interrupt Processing

The CPU automatically saves the current contents of the program counter (PC) and program status (PS) to the stack when an interrupt is accepted.

Figure 3.4-5 "Stack Operation at Start of Interrupt Processing" shows the stack operation at the start of interrupt processing.

**Figure 3.4-5 Stack Operation at Start of Interrupt Processing**




## ■ Stack Operation at Interrupt Return

On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU performs the opposite processing to interrupt initiation, restoring first the program status (PS) and then the program counter (PC) from the stack. This returns the PS and PC to their states immediately prior to the start of the interrupt.

**Note:**

The CPU does not automatically save the accumulator (A) or temporary accumulator (T) contents to the stack. Use the PUSHW and POPW instructions to save and restore A and T contents to and from the stack.

# 3.4.6 Stack Area for Interrupt Processing

**Interrupt processing execution uses the stack area in RAM. The contents of the stack pointer (SP) specifies the top address of the stack area.**

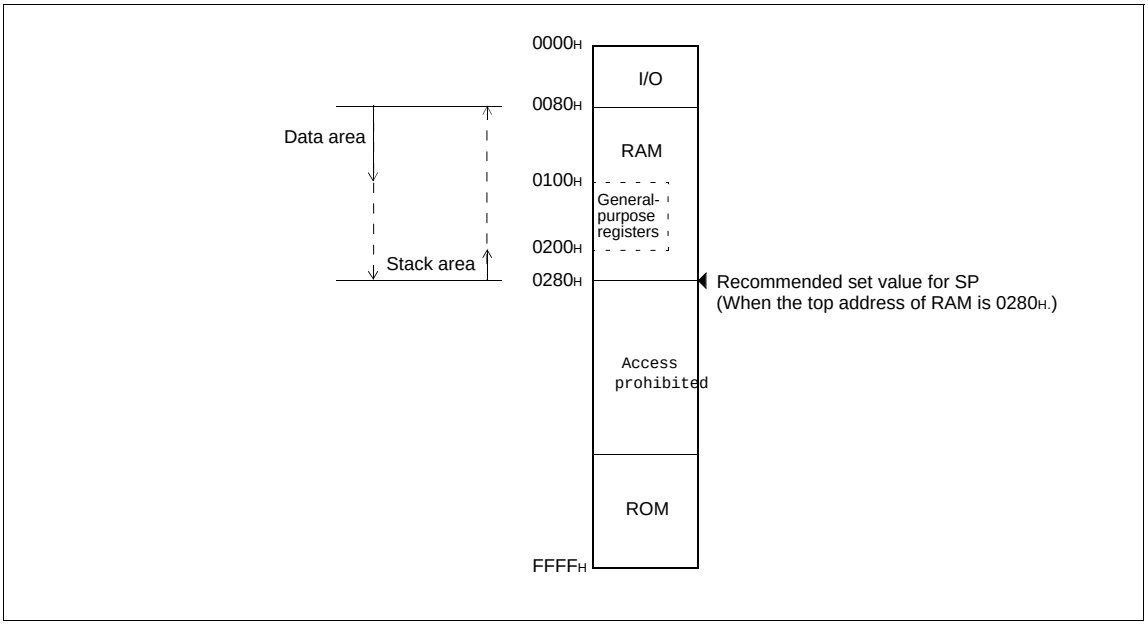
## ■ Stack Area for Interrupt Processing

The subroutine call instruction (CALL) and vector call instruction (CALLV) use the stack area to save and restore the program counter (PC). The stack area is also used by the PUSHW and POPW instructions to temporarily save and restore registers.

- The stack area is located in RAM along with the data area.
- Initializing the stack pointer (SP) to the top address of RAM and allocating data areas upwards from the bottom RAM address is recommended.

Figure 3.4-6 "Stack Area for Interrupt Processing" shows the example of stack area setting.

**Figure 3.4-6 Stack Area for Interrupt Processing**




**Reference:**

The stack area is used in the downward direction starting from a high address by functions such as interrupts, subroutine calls, and the PUSHW instruction. Instructions such as return instructions (RETI, RET) and the POPW instruction release stack area in the upward direction. Take care when the stack address is decreased by multiple interrupts or subroutine calls that the stack does not overlap the general-purpose register area or areas containing other data.

# 3.5 Resets

**The resets has the following four types of reset source:**

- **External reset**
- **Software reset**
- **Watchdog reset**
- **Power-on reset**

**At reset, main clock oscillation stabilization delay time may or may not occur by the operating mode and option settings.**

## ■ Reset Source

**Table 3.5-1 Reset Source**

| Reset source | Reset conditions |
|---|---|
| External reset | Set the external reset pin to the "L" level. |
| Software reset | Write "0" to the software reset bit in the standby control register (STBC: RST). |
| Watchdog reset | Watchdog timer overflow |
| Power-on reset | Power is turned on (only on products with a power-on reset). |

### ❍ External reset

Inputting an "L" level to the external reset pin (RSTX) generates an external reset. Returning the reset pin to the "H" level wakes up the CPU from the external reset.

When power is turned on to products with power-on reset or for external resets in stop mode, the reset operation is performed after the oscillation stabilization delay time has passed and the CPU wakes up from the external reset. External resets on products without power-on reset do not wait for the oscillation stabilization delay time.

The external reset pin can also function as a reset output pin (optional).

### ❍ Software reset

Writing "0" to the software reset bit in the standby control register (STBC: RST) generates a four-instruction cycle reset. The software reset does not wait for the oscillation stabilization delay time.

### ❍ Watchdog reset

The watchdog reset generates a four-instruction cycle reset if data is not written to the watchdog timer control register (WDTC) within a fixed time after the watchdog timer starts. The watchdog reset does not wait for the oscillation stabilization delay time.

### ❍ Power-on reset

A reset is generated by power-on. The reset operation is performed after the oscillation stabilization delay time has passed.

## ■ Main Clock Oscillation Stabilization Delay Time and the Reset Source

Whether there will be an oscillation stabilization delay time depends on the operating mode when reset occurs, and the power-on reset option selected.

Following reset, operation always starts out in the normal main clock operating mode, regardless of the kind of reset it was, or the operating mode (the clock mode and standby mode) prior to reset. Therefore, if reset occurs while the main clock oscillator is stopped or in a stabilization delay time, the system will be in a "main clock oscillation stabilization reset" state, and a clock stabilization period will be provided. If the device is set for no power-on reset, however, no main clock oscillation stabilization delay time is provided for power-on or external reset.

In software or watchdog reset, if the reset occurs while the device is in main clock mode, no stabilization time is provided. If it occurs in the subclock mode, however, a stabilization time is provided since the main clock oscillation is stopped.Table 3.5-2 "Reset Source and Oscillation Stabilization Delay Time" shows the relationships between the reset sources and the main clock oscillation stabilization delay time, and reset mode (mode fetch) operations.

**Table 3.5-2 Reset Source and Oscillation Stabilization Delay Time**

| Reset source | Operating state | Reset operation and main clock oscillation stabilization delay time |
|---|---|---|
| External reset*1 | At power on, during stop mode, or subclock mode | After the main clock oscillation stabilization delay time, if the external reset is waked up, reset is operated.*2 |
| Software and watchdog reset | Main clock mode | After 4-instruction-cycle reset occurs, reset is operated.*3 |
| | Subclock mode | Reset is operated after the main clock oscillation stabilization delay time.*2 |
| Power-on reset | | Device enters main clock oscillation stabilization delay time at power on. Reset is operated after delay time ends.*2 |

*1: No oscillation stabilization delay time is required for external reset while main clock mode is operating. Reset is operated after external reset is waked up.

*2: If the reset output option is selected, "L" is output at RSTX pin during the main clock oscillation stabilization delay time.

*3: If the reset output option is selected, "L" level is output at RSTX pin during 4-instruction-cycle.

## 3.5.1 Reset Flag Register (RSFR)

**The reset flag register (RSFR) can be used to identify the reset generation sources when a reset occurs.**

■ **Reset Flag Register (RSFR)**

**Figure 3.5-1 Reset Flag Register (RSFR)**

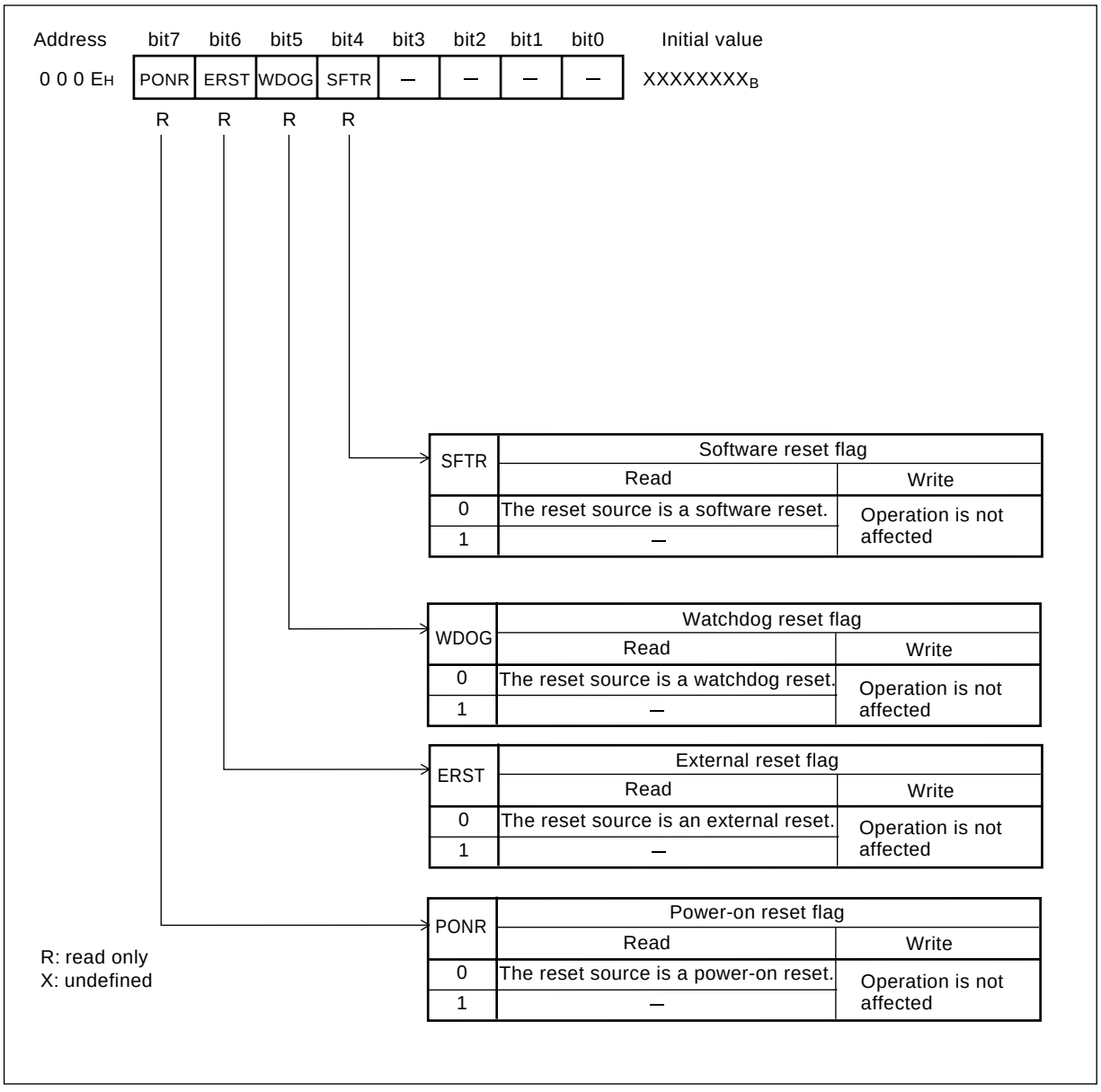


**Table 3.5-3 Explanation of the Functions of Each Bit of the Reset Flag Register (RSFR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | PONR:<br>Power-on reset flag | "1" is set when a power-on reset occurs.<br>"1" is set after power-on.<br>This register is cleared to "0" by reading it.<br>A write operation to this register is insignificant. |
| Bit 6 | ERST:<br>External reset flag | "1" is set when an external reset occurs.<br>"1" is set to the software reset flag while retaining each reset flag if each reset flag is set before the external reset flag is set.<br>This register is cleared to "0" by reading it.<br>A write operation to this register is insignificant. |
| Bit 5 | WDOG:<br>Watchdog reset flag | "1" is set when a watchdog reset occurs.<br>"1" is set to the watchdog reset flag while retaining each reset flag if each reset flag is set before the watchdog reset flag is set.<br>This register is cleared to "0" by reading it.<br>A write operation to this register is insignificant. |
| Bit 4 | SFTR:<br>Software reset flag | "1" is set when a software reset occurs.<br>"1" is set to the software reset flag while retaining each reset flag if each reset flag is set before the software reset flag is set.<br>This register is cleared to "0" by reading it.<br>A write operation to this register is insignificant. |
| Bit 3<br>Bit 2<br>Bit 1<br>Bit 0 | Unused bits | The read value is undefined.<br>Writing has no effect on operation. |

**Note:**

Each reset source flag is set when each reset source occurs. When each reset source flag register is read, each bit of the reset source flag register is cleared. Thus, determine the reset source by reading this register using the initialization routine after the reset.

# 3.6 External Reset Pin

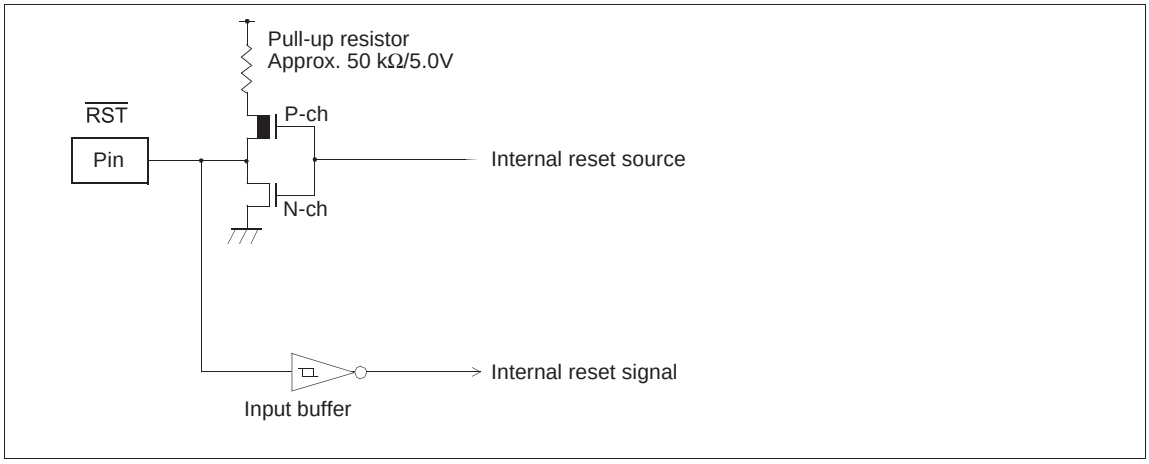
**Inputting an "L" level to the external reset pin generates a reset. If products are set to with the reset output (optional), the pin outputs an "L" level depending on internal reset sources.**

## ■ Block Diagram of External Reset Pin

The external reset pin (RSTX) on products with the reset output is a hysteresis input type and N-ch open-drain output type with a pull-up resistor.

The external reset pin on products without a reset output option is only for the reset input.Figure 3.6-1 "Block Diagram of External Reset Pin" shows the block diagram of the external reset pin.

**Figure 3.6-1 Block Diagram of External Reset Pin**




## ■ External Reset Pin Functions

Inputting an "L" level to the external reset pin (RSTX) generates an internal reset signal.

On products with the reset output, the pin outputs an "L" level depending on internal reset sources or during the oscillation stabilization delay time due to an external reset. Software reset, watchdog reset, and power-on reset are classed as internal reset sources.

**Note:**

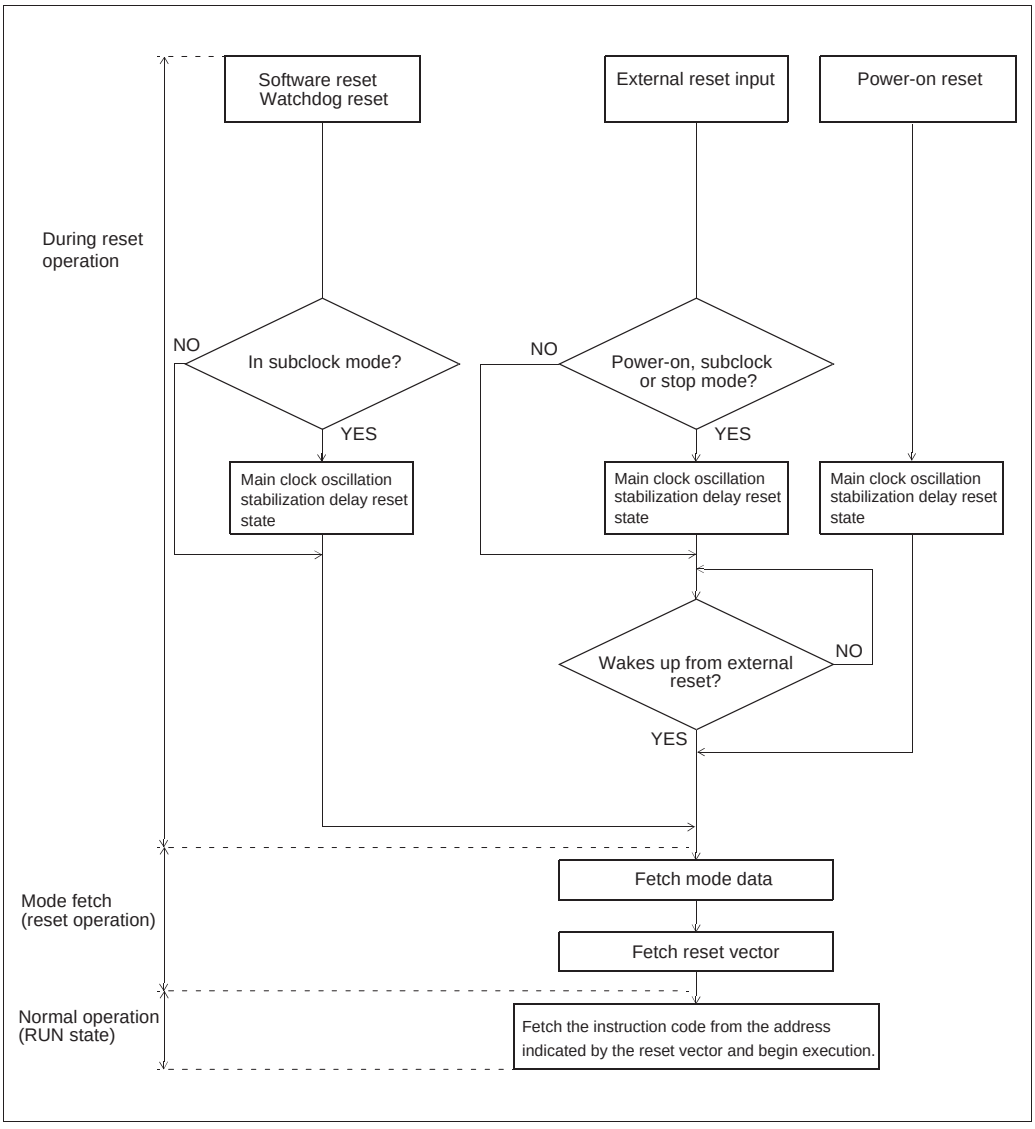
The external reset input accepts asynchronous with the internal clock. Therefore, initialization of the internal circuit requires a clock. Especially when an external clock is used, a clock is needed to be input at the reset.

## 3.6.1 Reset Operation

**When the CPU wakes up from a reset, the CPU selects the read address of the mode data and reset vector according to the mode pin settings, then performs a mode fetch. The mode fetch is performed after the oscillation stabilization delay time has passed when power is turned on to a product with power-on reset, or on wake-up from subclock or stop mode by a reset. If reset occurs during a write to RAM, the contents of the RAM address cannot be assured.**

### ■ Overview of Reset Operation

**Figure 3.6-2 Reset Operation Flow Diagram**



## ■ Mode Pins

The MB89570 series devices are single-chip mode devices. The mode pins (MOD1 and MOD0) must be tied to VSS. The mode pin settings determine whether the mode data and reset vector are read from internal ROM.

Do not change the mode pin settings, even after the reset has completed.

## ■ Mode Fetch

When the CPU wakes up from a reset, the CPU reads the mode data and reset vector from internal ROM.

### ❍ Mode data (address: $FFFD_H$)

Always set the mode to "$00_H$" (single-chip mode).

### ❍ Reset vector (address: $FFFE_H$ (upper), $FFFF_H$ (lower))

Contains the address where execution is to start after completion of the reset. The CPU starts executing instructions from the address contained in the reset vector.

## ■ Oscillation Stabilization Delay Reset State

On products with power-on reset, the reset operation for a power-on reset or external reset in subclock or stop (main/sub) mode starts after the main clock oscillation stabilization delay time selected by the stabilization delay time option. If the CPU has not woken up from the external reset input when the delay time completes, the reset operation does not start until the CPU wakes up from external reset.

As the oscillation stabilization delay time is also required when an external clock is used, a reset requires that the external clock is input.

The main clock oscillation stabilization delay time is timed by the timebase timer.

On products without power-on reset, the oscillation stabilization delay reset state is not used. Therefore, for such products, hold the external reset pin (RSTX) at the "L" level to disable the CPU operation until the source oscillation stabilizes.

## ■ Effect of Reset on RAM Contents

The contents of RAM are unchanged before and after a reset other than power-on reset. If an external reset is input close to a write timing, however, the contents of the write address cannot be assured. For this reason, all RAM locations being used should be initialized following reset.

## 3.6.2 Pin States during Reset

**Reset initialized the pin states.**

### ■ Pin States during Reset

When a reset source occurs, with a few exceptions, all I/O pins (peripheral pins) go to the high-impedance state and the mode data is read from internal ROM.

### ■ Pin States after Reading Mode Data

With a few exceptions, the I/O pins remain in the high-impedance state immediately after reading the mode data.

**Note:**

For devices connected to pins that change to high-impedance state when a reset source occurs take care that malfunction does not occur due to the change in the pin states.

# 3.7 Clock

**Dual clock oscillation circuits are contained in the clock generator. By connecting each external resonator, the high-speed main clock and low-speed subclock are generated independently (oscillator source). A clock generated externally can also be input.**
**The speed and supply of the dual clock is controlled by the clock controller according to the clock mode and standby mode.**

## ■ Clock Supply Map

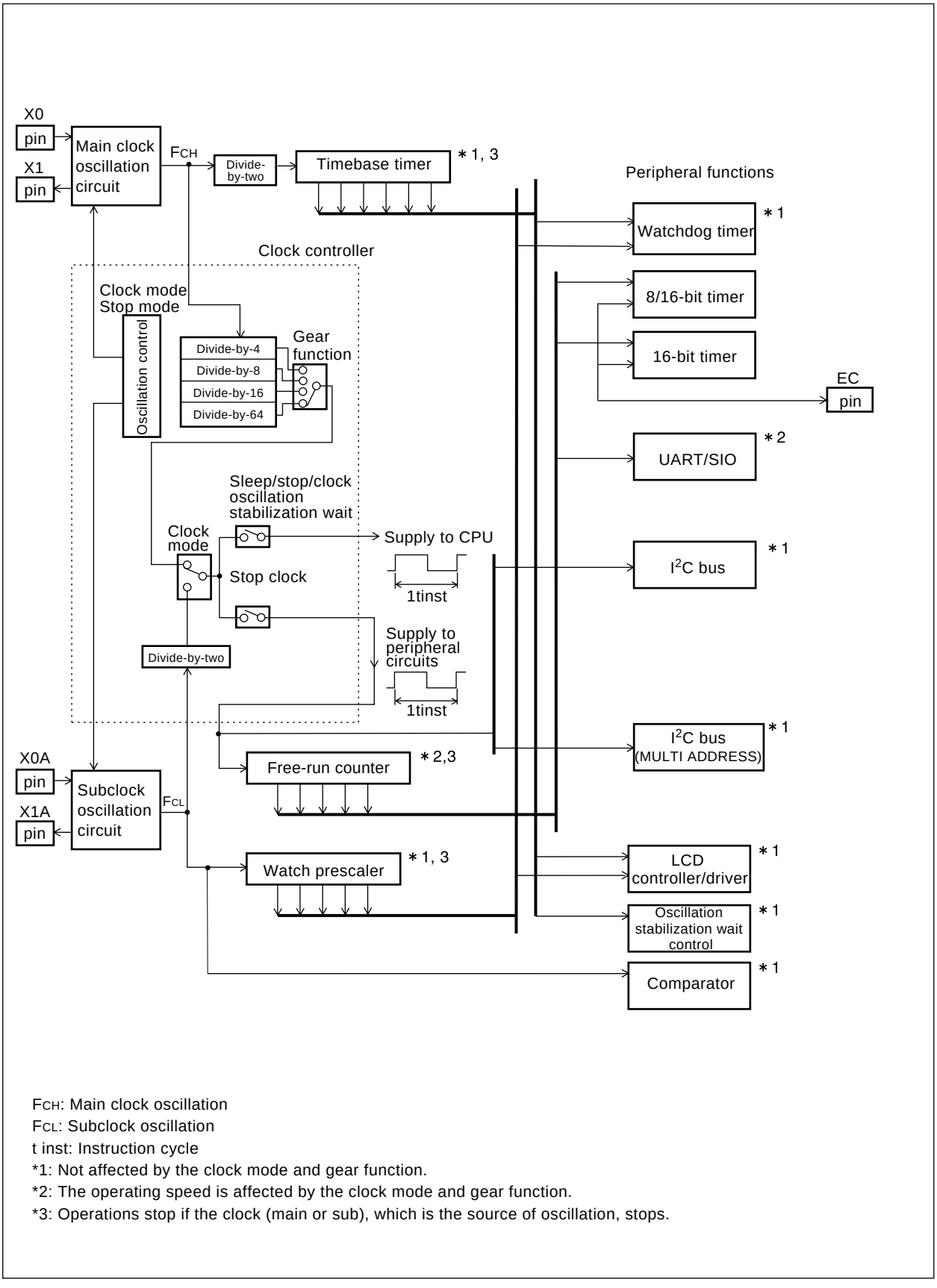
The clock oscillation and the supply to CPU and peripheral circuits (peripheral functions) are controlled by the clock controller. Thus, the operating clock of CPU and peripheral circuits are affected by switching between the main clock and the subclock (clock mode), speed switching of the main clock (gear function), and the standby mode (sleep/stop/clock).

The divide-by output of the free-run counter of the clock for peripheral circuits is supplied to each peripheral function.

The divide-by output of the timebase timer operating at divide-by-two oscillation of the main clock oscillation and peripheral functions to which divide-by output of the watch prescaler operating at the subclock is supplied are available and are not affected by the gear function.

The following Figure 3.7-1 "Clock Supply Map" shows a clock supply map.

**Figure 3.7-1 Clock Supply Map**




FCH: Main clock oscillation
FCL: Subclock oscillation
t inst: Instruction cycle
*1: Not affected by the clock mode and gear function.
*2: The operating speed is affected by the clock mode and gear function.
*3: Operations stop if the clock (main or sub), which is the source of oscillation, stops.

# 3.7.1 Clock Generator

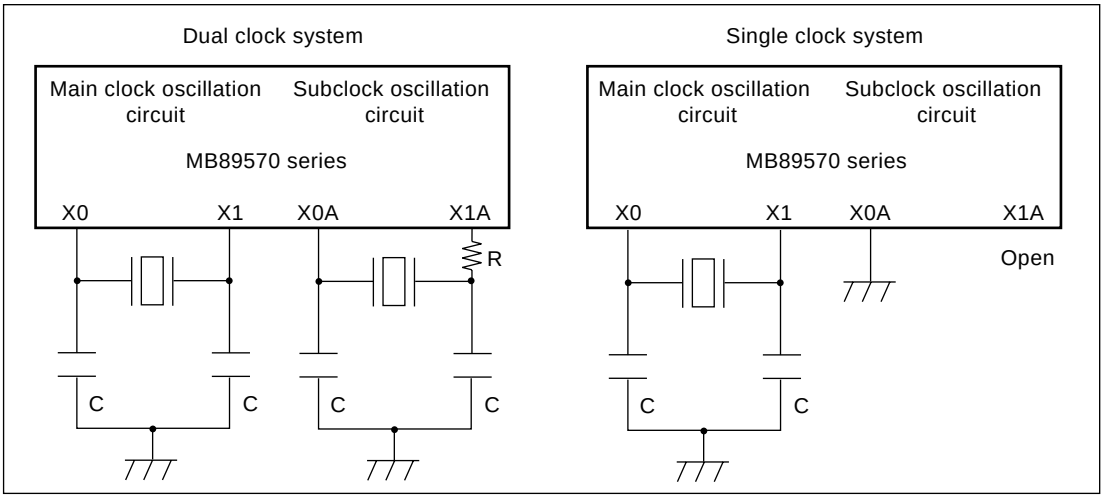
**The permission and stop of oscillation of the main clock and subclock are controlled by the clock mode and stop mode.**

## ■ Clock Generator

### ❍ Crystal resonator or ceramic resonator

Make connections as shown in Figure 3.7-2 "Connection Example of the Crystal and Ceramic Resonator".

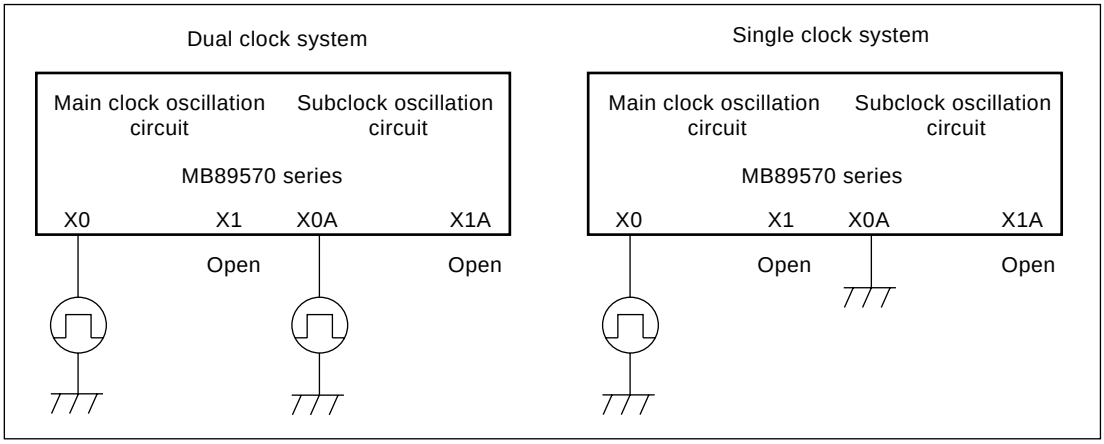
**Figure 3.7-2 Connection Example of the Crystal and Ceramic Resonator**




### ❍ External clock

Connect the external clock to the X0 pin as shown in Figure 3.7-3 "Connection Example of the External Clock" and open the X1 pin. If the subclock should be supplied externally, connect the external clock to the X0A pin and open the X1A pin. Note that the MB89PV570 outputs "L" from the X0A pin in sub-stop mode.

**Figure 3.7-3 Connection Example of the External Clock**




**Note:**

In MB89PV570, "L" is output from the X0A pin in sub-stop mode. If the subclock is supplied from an external clock, care must be taken to ensure that "H" is not applied to the X0A pin in sub-stop mode.

# 3.7.2 Clock Controller

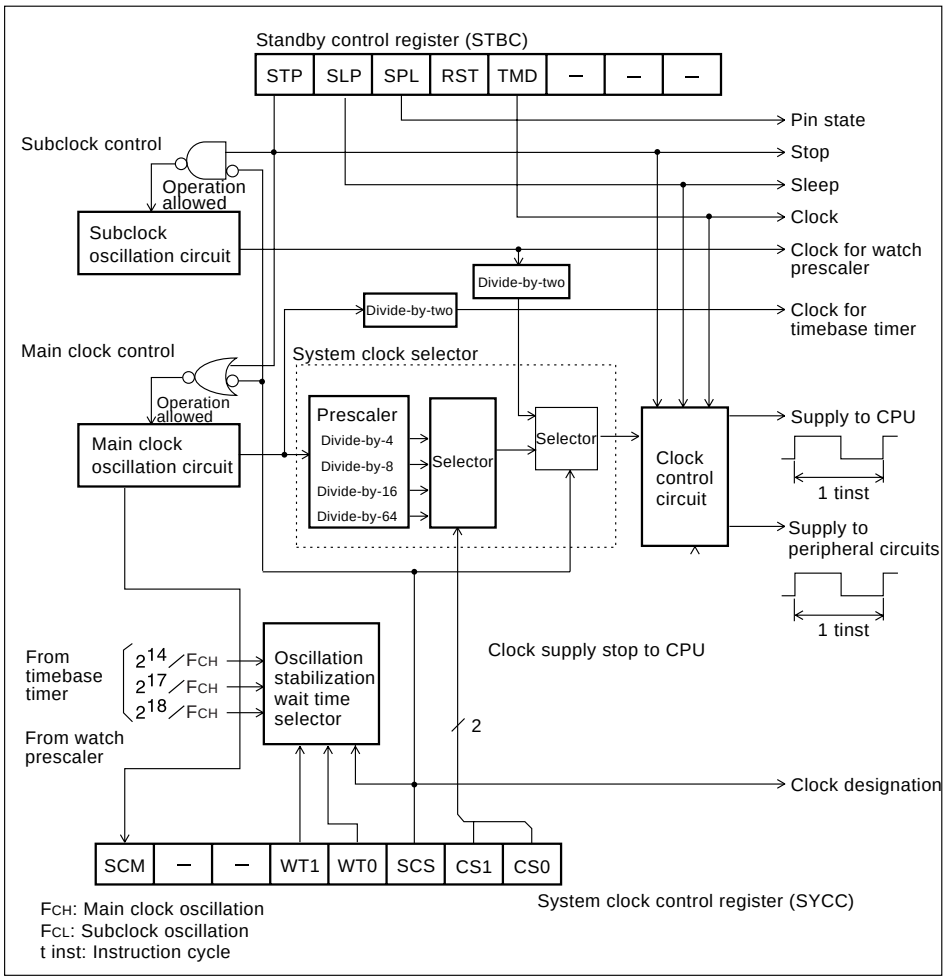
**The clock controller is made up of the following seven blocks:**

- **Main clock oscillation circuit**
- **Subclock oscillation circuit**
- **System clock selector**
- **Clock control circuit**
- **Oscillation stabilization wait time selector**
- **System clock control register (SYCC)**
- **Standby control register (STBC)**

## ■ Block Diagram of the Clock Controller

Figure 3.7-4 "Block Diagram of the Clock Controller" shows a block diagram of the clock controller.

**Figure 3.7-4 Block Diagram of the Clock Controller**



❍ **Main clock oscillation circuit**

Oscillation circuit of the main clock. This circuit stops oscillation in main stop mode and subclock mode.

❍ **Subclock oscillation circuit**

Oscillation circuit of the subclock. This circuit always oscillates in a mode other than sub-stop mode.

❍ **System clock selector**

One type is selected from the four clocks and the subclocks obtained by dividing the oscillation of the main clock to supply it to the clock control circuit.

❍ **Clock control circuit**

The operating clock supply to CPU and each peripheral circuit is controlled according to normal operation (RUN) and standby modes (sleep, stop, clock).

The clock controller stops the supply of clocks to CPU until the clock supply stop signal of the oscillation stabilization wait time selector is released.

❍ **Oscillation stabilization wait time selector**

One wait time is selected from the four kinds of oscillation stabilization wait time for the main clock created by the timebase timer and the oscillation stabilization wait time for the subclock created by the watch prescaler for the clock mode, standby mode, and reset and is output as the clock supply stop signal to CPU.

❍ **System clock control register (SYCC)**

The selection of the clock mode and main clock speed is performed, and the selection and state confirmation of the oscillation stabilization wait time of the main clock is performed.

❍ **Standby control register (STBC)**

The transition from normal operation (RUN) to standby mode, pin state settings in stop mode or watch mode, and software reset are performed.

## 3.7.3 System Clock Control Register (SYCC)

**The system clock control register (SYCC) is used to switch the main clock and the subclock, to select the speed of the main clock, and to select the oscillation stabilization wait time.**

### ■ Structure of the System Clock Control Register (SYCC)

**Figure 3.7-5 Structure of the System Clock Control Register (SYCC)**

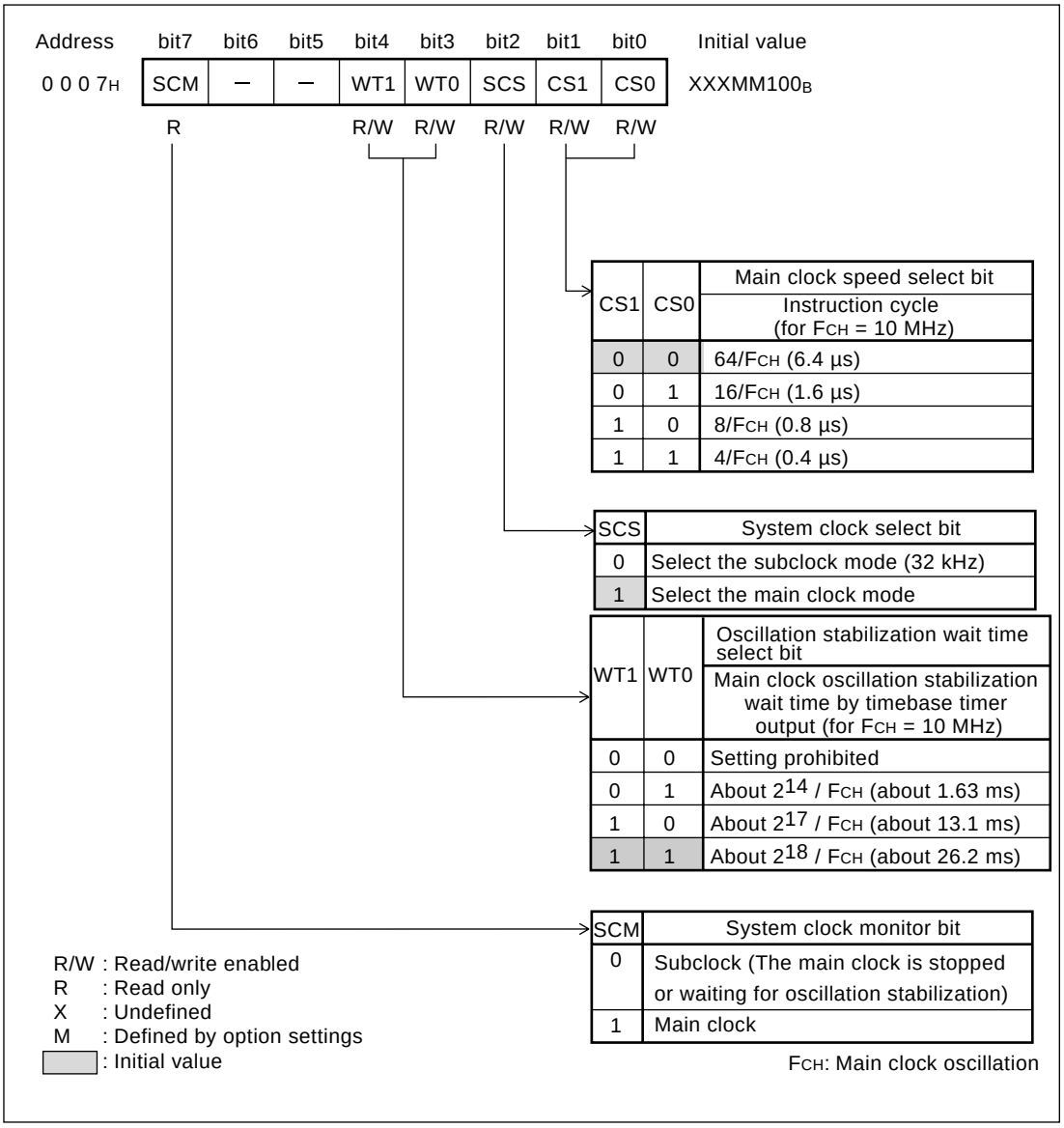


**Table 3.7-1 Explanation of the Functions of Each Bit of the System Clock Control Register (SYCC)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | SCM:<br>System clock monitor bit | • Bit to check the current clock mode (operating clock)<br>• If the bit is "0", the system is operating in subclock mode (The main clock is stopped or waiting for oscillation stabilization to make a transition to the main clock mode).<br>• If the bit is "1", the system is operating in main clock mode.<br>**[Reference:]**<br>This bit is read-only. Write operation to this bit has no significance and does not affect operations. |
| Bit 6<br>Bit 5 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 4<br>Bit 3 | WT1, WT0:<br>Oscillation stabilization wait time select bits | • Bits to select the oscillation stabilization wait time of the main clock<br>• The oscillation stabilization wait time selected by these bits is taken when making a transition from the subclock mode to the main clock mode, or returning to normal operation from the main stop mode by an external interrupt.<br>• The initial values of these bits are selected by option settings *. Thus, if an oscillation stabilization wait time is taken for a reset, the oscillation stabilization wait time selected by option settings is taken.<br>**Note:**<br>Do not rewrite these bits simultaneously when switching from the subclock to the main clock (SCS=1 --> 0). Before rewriting the bits, check that the oscillation stabilization is not waited upon using the SCM bit. |
| Bit 2 | SCS:<br>System clock select bit | • Bit to specify the clock mode<br>• A transition from the main clock mode to the subclock mode is caused by writing "0" into this bit.<br>• If "1" is written into this bit, the transition from the subclock mode to the main clock mode occurs after taking the oscillation stabilization wait time set by the WT1 and WT0 bits.<br>**Note:**<br>If the single clock system option is selected, this bit has no significance. Set always "1". |
| Bit 1<br>Bit 0 | CS1, CS0:<br>Main clock speed select bits | • Bit to select the clock speed in main clock mode.<br>• Four different speeds of the operating clock can be selected for CPU and each peripheral function (gear function). However, the operating clock of the timebase timer and watch prescaler is not affected by these bits. |

*: Options can be selected only for MB89577.

## ■ Instruction Cycle ($t_{inst}$)

The instruction cycle (minimum execution time) can be selected from the 1/4, 1/8, 1/16, or 1/64 of the main clock and the divide-by-two of the subclock (32.768 kHz) using the system clock select bit (SCS) and main clock speed select bits (CS1, CS0) of the SYCC register.

The instruction cycle at the maximum speed (SYCC: SCS=1, CS1, CS0=$11_B$) in main clock mode is $4/F_{CH}$ = about 0.4 μs if the main clock oscillation ($F_{CH}$) is 10 MHz.

The instruction cycle in subclock mode (SCS=0) is $2/F_{CL}$ = about 61.0 μs if the subclock oscillation ($F_{CL}$) is 32.768 kHz.

# 3.7.4 Clock Modes

**The main clock mode and subclock mode are available as the clock mode.**
**In main clock mode, the main clock is the main operating clock. The speed of the main clock can be switched by selecting from four kinds of clocks created by dividing its oscillation (gear function).**
**In subclock mode, the oscillation of the man clock is stopped and the subclock alone becomes the operating clock.**

## ■ Operating State of the Clock Mode

**Table 3.7-2 Operating State of the Clock Mode**

| Clock mode | Main clock speed SYCC register (CS1, CS0) | | Standby mode | Clock generation | | Operation clock in each section | | | | Release source of standby mode (other than resets) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Main | Sub | CPU | timebase timer | Each peripheral | watch prescaler | |
| Main clock mode | (1.1) | High speed | RUN | Oscillation | Oscillation | $F_{CH}/4$ | $F_{CH}/2$ | $F_{CH}/4$ | $F_{CL}$ | Various interrupt requests |
| | | | Sleep | | | Stop | | | | |
| | | | Stop | Stop | | | Stop | Stop | | External interrupt |
| | (1.0) | | RUN | Oscillation | Oscillation | $F_{CH}/8$ | $F_{CH}/2$ | $F_{CH}/8$ | $F_{CL}$ | Various interrupt requests |
| | | | Sleep | | | Stop | | | | |
| | | | Stop | Stop | | | Stop | Stop | | External interrupt |
| | (0.1) | | RUN | Oscillation | Oscillation | $F_{CH}/16$ | $F_{CH}/2$ | $F_{CH}/16$ | $F_{CL}$ | Various interrupt requests |
| | | | Sleep | | | Stop | | | | |
| | | | Stop | Stop | | | Stop | Stop | | External interrupt |
| | (0.0) | Low speed | RUN | Oscillation | Oscillation | $F_{CH}/64$ | $F_{CH}/2$ | $F_{CH}/64$ | $F_{CL}$ | Various interrupt requests |
| | | | Sleep | | | Stop | | | | |
| | | | Stop | Stop | | | Stop | Stop | | External interrupt |
| Subclock mode | - | | RUN | Stop | Oscillation | $F_{CL}$ | Stop(*1) | $F_{CL}$ | $F_{CL}$ | Various interrupt requests |
| | | | Sleep | | | Stop | | | | |
| | | | Stop | | Stop | | | Stop | Stop | External interrupt |
| | | | Watch mode | Stop | Oscillation | Stop | Stop(*1) | Stop | $F_{CL}$ | External interrupt, watch interrupt |

$F_{CH}$: Main clock oscillation
$F_{CL}$: Subclock oscillation
*1: Since the timebase timer is operated by the main clock, it stops operation in subclock mode.

In each clock mode, a transition can be made to the standby mode corresponding to each mode. For the standby mode, see Section 3.8 "Standby Mode (low power consumption)".

## ■ Gear Function (Function for Switching the Speed of the Main Clock)

Four different main clock speeds can be selected by writing "$00_B$" to "$11_B$" into the main clock speed select bits (SYCC: CS1, CS0) of the system clock control register.

CPU and each peripheral circuit operate at the switched main clock speed. However, the timebase timer and watch prescaler are not affected by the gear function.

By reducing the main clock speed, power consumption can be reduced.

## ■ Operation in Main Clock Mode

In normal operation in main clock mode (main RUN mode), both the main clock and subclock oscillate. The watch prescaler is operated by the subclock, whereas CPU, the timebase timer, and other peripheral circuits are operated by the main clock.

The speed of the main clock can be switched to a value other than that of the timebase timer during operation in main clock mode (gear function). A transition to the main sleep mode or main stop mode is enabled by specifying the standby mode.

Operation always starts in main RUN mode regardless of the type of reset that occurs (release by the reset in each operation mode).

### ❍ Transition from the main clock mode to the subclock mode

A transition from the main clock mode to the subclock mode is caused by writing "0" into the system clock select bit (SYCC: SCS) of the system clock control register.

The current operating clock can be checked by reading the system clock monitor bit (SYCC: SCM) of the system clock control register.

To make a transition to the subclock mode, for example, just after power-on, it is necessary to wait at least the subclock oscillation stabilization wait time created by the watch prescaler using software before making a transition.

## ■ Operation in Subclock Mode

In normal operation in subclock mode (sub RUN mode), the oscillation of the main clock is stopped and only the subclock is used for operation. By operating in a low speed clock, power consumption can be reduced.

All functions other than the timebase timer operate as in the main clock mode. A transition to the sub-sleep mode, sub-stop mode, or watch mode is enabled by specifying the standby mode during operation in subclock mode.

### ❍ Return from the subclock mode to the main clock mode

A return from the subclock mode to the main clock mode is caused by writing "1" into the system clock select bit (SYCC: SCS) of the system clock control register.

However, operation in main clock mode starts only after the oscillation stabilization wait time of the main clock passes. The oscillation stabilization wait time can be selected from three different wait times using the oscillation stabilization wait time select bits (SYCC: WT1, WT0) of the system clock control register.

**Note:**

Do not rewrite the oscillation stabilization wait time select bits (SYCC: WT1, WT0) simultaneously by switching from the subclock to the main clock. Also, do not rewrite the bits when the oscillation stabilization of the main clock is waited upon. In such cases, rewrite the bits after checking that the operating clock has been switched to the main clock (SYCC: SCM=1) by the system clock monitor bit.

To return to the main clock mode from the subclock mode using a reset, take the oscillation stabilization wait time of the main clock.

## 3.7.5 Oscillation Stabilization Wait Time

**If the main clock is operated in main RUN mode from a state in which the main clock is stopped, for example, when the power is turned on, or in main stop mode or subclock mode, it is necessary to take the oscillation stabilization wait time of the main clock. Likewise, the oscillation stabilization wait time of the subclock is needed in sub-stop mode because the oscillation of the subclock is stopped.**

### ■ Oscillation Stabilization Wait Time

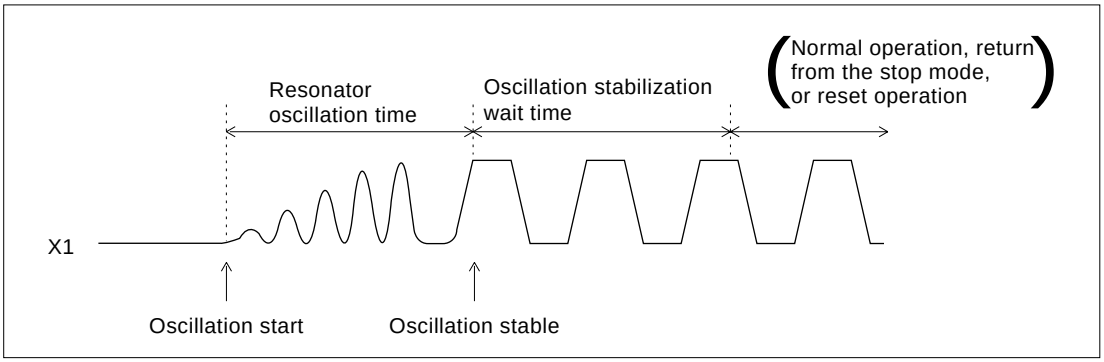
Ceramic and crystal resonators generally take several ms to several dozens of ms to oscillate steadily in natural frequency after starting oscillation.

Thus, CPU operation must be prohibited just after starting oscillation. The clock should be supplied to CPU only when the oscillation is sufficiently stable after the passage of the oscillation stabilization wait time.

Since the time needed to stabilize oscillation is dependent on the type (such as the crystal and ceramic) of resonator connected to the oscillator (clock generator), an oscillation stabilization wait time appropriate to the resonator to be used must be selected.

Figure 3.7-6 "Oscillator Operation after Oscillation Starts" shows an oscillator operation just after the oscillation starts.

**Figure 3.7-6 Oscillator Operation after Oscillation Starts**




### ■ Oscillation Stabilization Wait Time of the Main Clock

To start operation in main clock mode from a state in which the main clock is stopped, the oscillation stabilization wait time of the main clock must be taken.

The oscillation stabilization wait time of the main clock is a time interval counted from when the counter of the timebase timer is cleared until the overflow of the specified bit occurs.

#### ❍ Oscillation stabilization wait time during operation

One of the four kinds of oscillation stabilization wait time when returning to the main RUN mode from the main stop mode by an external reset or when making a transition from the subclock mode to the main clock mode can be selected using the oscillation stabilization wait time select bits (SYCC: WT1, WT0) of the system clock control register.

❍ **Oscillation stabilization wait time during reset**

The oscillation stabilization wait time during reset (initial value of WT1 and WT0) can be selected by option settings.

The oscillation stabilization wait time must be taken for (multiple) resets in subclock mode, a power-on reset, and the stop mode release by an external reset.

Table 3.7-3 "Operation Start Conditions and Oscillation Stabilization Wait Time of the Main Clock Mode" lists the relations between the operation start conditions and oscillation stabilization wait time of the main clock mode.

**Table 3.7-3 Operation Start Conditions and Oscillation Stabilization Wait Time of the Main Clock Mode**

| Start conditions for main clock operation | During power-on | During subclock mode | | Release from main stop mode | | Transition from the subclock mode to the main clock mode (SYCC: SCS (*1) =1) |
|---|---|---|---|---|---|---|
| | | External reset | Software reset and watchdog timer | External reset | External interrupt | |
| Oscillation stabilization wait time selection | Option settings (*3) | | | | SYCC:WT1, WT0 (*2) | |

*1: System clock select bit of the system clock control register
*2: Oscillation stabilization wait time select bit of the system clock control register
*3: Only in MB89577, options of the initial value of "SYCC: WT1, WT0" can be selected.

## ■ Oscillation Stabilization Wait Time of the Subclock

A certain oscillation stabilization wait time ($2^{15}/F_{CL}$, $F_{CL}$: subclock oscillation) of the subclock must be taken when returning to the sub RUN mode (subclock oscillation is started) from the sub-stop mode (state in which oscillation of the subclock is stopped) by an external reset.

The oscillation stabilization wait time of the subclock is a time interval from the start of operation in a state in which the watch prescaler is cleared until an overflow occurs.

Since the oscillation stabilization wait time of the subclock is needed during power-on, wait at least the subclock oscillation stabilization wait time using software to make a transition to the subclock mode after power-on.

# 3.8 Standby Mode (Low Power Consumption)

**The sleep mode, stop mode, and watch mode are available as the standby mode. A transition to the standby mode is caused by settings of the standby control register (STBC) both in main clock mode and subclock mode.**
**In main clock mode, transitions to the sleep mode and stop mode are possible. In subclock mode, transitions to the sleep mode, stop mode, and watch mode are possible. Power consumption can be reduced by stopping operations of CPU and peripheral functions using the standby mode. This section describes the relations between the standby mode and clock mode, and the operating states of each section in standby mode.**

## ■ Standby Mode

In clock mode, power consumption is reduced by reducing the operating clock of CPU and peripheral circuits such as switching of the main clock and subclock or switching of the main clock speed (gear function). In standby mode, however, power consumption is reduced by the clock supply stop (sleep mode) to CPU by the clock controller, clock supply stop (watch mode) to CPU and peripheral circuits, or stop of the oscillation itself (stop mode).

### ❍ Main sleep mode

The main sleep mode is a mode which stops operations of CPU and the watchdog timer. Peripheral functions excluding the watch prescaler operate on the main clock (Part of the functions can operate on the subclock).

### ❍ Sub-sleep mode

The sub-sleep mode is a mode which stops the main clock oscillation, CPU operations, and watchdog timer and timebase timer operations. Peripheral functions operate on the subclock.

### ❍ Main stop mode

The main stop mode is a mode which stops operations of CPU and peripheral functions. The main clock stops oscillation, but the subclock continues oscillation. In this mode, all functions are stopped excluding external interrupts, count operations of the watch prescaler, and some of the functions that run with the subclock.

### ❍ Sub-stop mode

The sub-stop mode is a mode which stops all functions other than external interrupts. The main clock and the subclock both stop oscillation.

### ❍ Watch mode

The clock mode is a mode a transition to which is possible only from the subclock mode. All functions other than the watch prescaler (watch interrupt), external interrupts, and part of the functions operating on the subclock stop.

# 3.8.1 Operating State in Standby Mode

**This section describes the operating states of CPU and peripheral functions in standby mode.**

## ■ Operating State in Standby Mode

**Table 3.8-1 The Operating States of CPU and Peripheral Functions in Standby Mode**

| Operation mode | | Main clock mode | | | | Subclock mode | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Function | | RUN | Sleep | Stop (SPL=0) | Stop (SPL=1) | RUN | Sleep | Stop (SPL=0) | Stop (SPL=1) | Clock |
| Main clock | | Operating | Operating | Stopped | Stopped | Stopped | Stopped | Stopped | Stopped | Stopped |
| Subclock | | Operating | Operating | Operating | Operating | Operating | Operating | Stopped | Stopped | Operating |
| CPU | Instruction | Operating | Stopped | Stopped | Stopped | Operating | Stopped | Stopped | Stopped | Stopped |
| | ROM | Operating | Retained | Retained | Retained | Operating | Retained | Retained | Retained | Retained |
| | RAM | | | | | | | | | |
| Peripheral functions | I/O port | Operating | Retained | Retained | Retained | Operating | Retained | Retained | Retained | Retained |
| | timebase timer | Operating | Operating | Stopped | Stopped | Stopped | Stopped | Stopped | Stopped | Stopped |
| | Watchdog timer | Operating | Stopped | Stopped | Stopped | Operating(*4) | Stopped | Stopped | Stopped | Stopped |
| | Multi-address $I^2C$ bus | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | $I^2C$ bus | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | UART/SIO | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | Comparator | Operating | Operating | Operating | Operating | Operating | Operating | Operating | Operating | Operating |
| | A/D converter | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | External interrupt | Operating | Operating | Operating | Operating | Operating | Operating | Operating | Operating | Operating |
| | watch prescaler | Operating | Operating | Operating(*1) | Operating(*1) | Operating | Operating | Stopped | Stopped | Operating |
| | LCD controller/driver | Operating | Operating | Operating(*2) | Operating(*2) | Operating(*2) | Operating(*2) | Stopped | Stopped | Operating(*2) |
| | D/A converter | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | 8/16-bit timer | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| | 16-bit timer | Operating | Operating | Stopped | Stopped | Operating | Operating | Stopped | Stopped | Stopped |
| Pin | | Operating | Retained | Retained | Hi-z | Operating | Retained | Retained | Hi-z | Retained |
| How to release | | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts | Reset/ various interrupts |

*1: The watch prescaler carries out the count operation, but no watch interrupt occurs.
*2: The subclock is selected as the operating clock. Operation permission is required in watch mode.

### ❍ Pin state in standby mode

Most I/O pins can, independent of the clock mode, retain the state just before transition to the stop or watch mode or can be put into high impedance using the pin state designate bit (STBC: SPL) of the standby control register.

# 3.8.2 Sleep Mode

**This section describes the operations in sleep mode.**

## ■ Operations in Sleep Mode

### ❍ Transition to the sleep mode

The sleep mode is a mode which stops the operating clock of CPU. CPU stops by retaining the contents of the registers and RAM just before transition to the sleep mode, but peripheral functions other than the watchdog timer continue their operations.

However, since the main clock oscillation stops in subclock mode, the timebase timer which uses the divide-by-two of the main clock oscillation as its count clock does not operate.

A transition to the sleep mode is caused by writing "1" into the sleep bit (STBC: SLP) of the standby control register. If an interrupt has occurred when "1" is written into the SLP bit, the write operation is ignored and execution of instructions continues without transition to the sleep mode (no transition to sleep mode after the interrupt).

### ❍ Sleep mode release

The sleep mode is released by a reset or an interrupt from the peripheral functions.

If a reset occurs in sub-sleep mode, the reset operation is performed after taking the oscillation stabilization wait time of the main clock.

Pin states are initialized by the reset operation.

If an interrupt request whose interrupt level is higher than "11" comes from a peripheral function or external interrupt circuit, the sleep mode is released regardless of the interrupt enable flag (CCR: I) or interrupt level bit (CCR: IL1, 0) of CPU.

After the release, normal interrupt operations are performed. If an interrupt can be accepted, interrupt processing is performed. If no interrupt can be accepted, processing starts with the instruction following the instruction executed just before the transition to sleep mode.

# 3.8.3 Stop Mode

**This section describes the operations in stop mode.**

## ■ Operations in Stop Mode

### ❍ Transition to the stop mode

The stop mode is a mode which stops the oscillation. Contents of the registers and RAM just before transition to the stop mode are retained and most functions are stopped.

In main clock mode, the main clock stops the oscillation, but the subclock continues the oscillation. Thus, though the count operation of the watch prescaler and part of the functions operating on the subclock continue their operations, other peripheral functions and CPU, excluding the external interrupt circuits, stop operations.

In subclock mode, both the main clock and subclock stop oscillation, and all functions other than the external interrupt circuits stop their functions. Thus, data can be retained with minimum power consumption.

A transition to the stop mode is caused by writing "1" into the stop bit (STBC: STP) of the standby control register. At this time, if the pin state designate bit (STBC: SPL) is "0", the states of external pins are retained. If the bit is "1", external pins are put into high impedance.

If an interrupt request has occurred when "1" is written into the STP bit, the write operation is ignored and execution of instructions continues without making a transition to the stop mode (No transition to the stop mode occurs even after interrupt processing is completed).

To make a transition to the stop mode in the main clock mode, prohibit (TBTC: TBIE=0) the interrupt request output of the timebase timer if necessary. Likewise, to make a transition to the stop mode in subclock mode, prohibit (WPCR: WIE=0) the watch interrupt request output of the watch prescaler.

### ❍ Stop mode release

The stop mode can be released by a reset or external interrupt.

If a reset occurs in stop mode, the reset operation is performed after taking the oscillation stabilization wait time of the main clock.

Pin states are initialized by the reset.

If an interrupt request whose interrupt level is higher than "11" comes from an external interrupt circuit in stop mode, the stop mode is released regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1, 0) of CPU. Since peripheral functions are stopped in stop mode, no interrupt requests other than external interrupts occur. Though the watch prescaler operates in main stop mode, no watch interrupts occur.

If the stop mode is released, a normal interrupt operation is performed following the passage of the oscillation stabilization wait time. If the interrupt is accepted, interrupt processing is performed. If the interrupt is not accepted, execution starts with the instruction following the instruction executed just before transition to the stop mode.

If the stop mode is released by an external interrupt, part of the peripheral functions restarts halfway through their operations. Thus, for example, the first interval time of the interval timer function is undefined. Each peripheral function should be initialized after returning from the stop mode.

**Note:**

The stop mode release by an interrupt can only be caused by an interrupt request of the external interrupt circuits.

# 3.8.4 Watch Mode

**This section describes the operations in watch mode.**

## ■ Operations in watch mode

### ❍ Transition to the watch mode

The watch mode is a mode which stops the operating clock of CPU and main peripheral circuits. A transition to the watch mode is only possible from the subclock mode (The main clock oscillation is stopped).

Contents of the registers and RAM just before transition to the watch mode are retained and most functions other than the watch prescaler (watch interrupt), external interrupt circuits, and part of the functions operating on the subclock are stopped. Thus, data can be retained with very low power consumption.

A transition to the watch mode is caused by writing "1" into the clock bit (STBC: TMD) of the standby control register when the subclock mode is set by the system clock select bit of the system clock control register.

If the pin state designate bit (STBC: SPL) of the standby control register during transition to the watch mode is "0", the external pin states are retained. If the bit is "1", external pins are put into high impedance.

If an interrupt request has occurred when "1" is written into the TMD bit, the write operation is ignored and execution of instructions continues without making a transition to the watch mode (No transition to the watch mode occurs even after interrupt processing is completed).

### ❍ Watch mode release

The watch mode can be released by a reset, watch interrupt, or external interrupt.

If a reset occurs in watch mode, the reset operation is performed after taking the oscillation stabilization wait time of the main clock.

Pin states are initialized by the reset.

If an interrupt request whose interrupt level is higher than "11" comes from the watch prescaler or an external interrupt circuit in watch mode, the watch mode is released regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1, 0) of CPU. Since most peripheral functions other than the watch prescaler are stopped in watch mode, no interrupt requests other than watch interrupts and external interrupts occur.

After releasing the watch mode, a normal interrupt operation is performed. If the interrupt is accepted, interrupt processing is performed. If the interrupt is not accepted, execution starts with the instruction following the instruction executed just before transition to the watch mode.

If the watch mode is released, part of the peripheral functions restarts halfway through their operations. Thus, for example, the first interval time of the interval timer function is undefined. Each peripheral function should be initialized after returning from the watch mode.

## 3.8.5 Standby Control Register (STBC)

**The standby control register (STBC) is used to make a transition to the sleep mode/ stop mode/watch mode, set the pin states in watch mode, and reset software.**

### ■ Standby Control Register (STBC)

**Figure 3.8-1 Standby Control Register (STBC)**

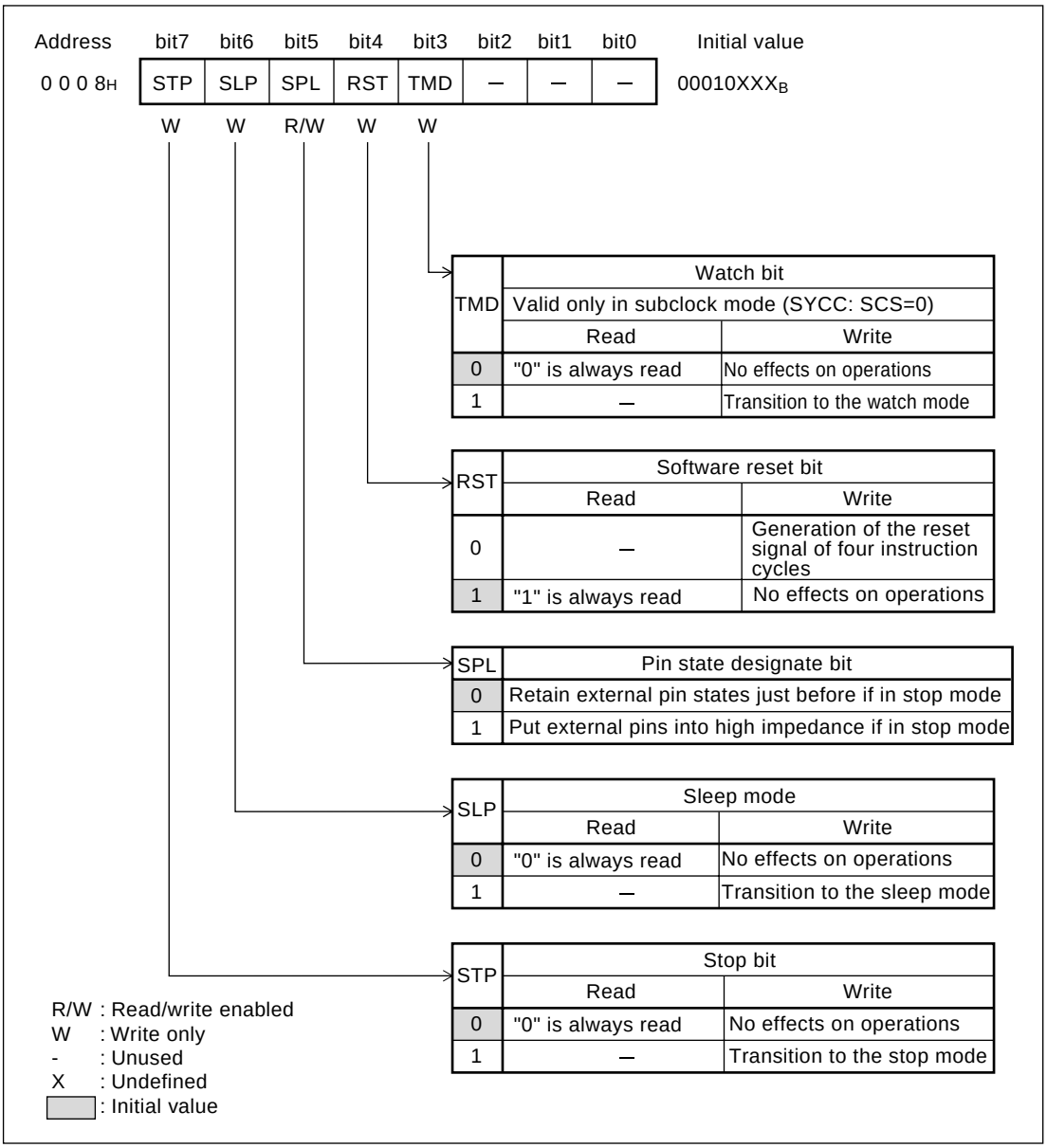


**Table 3.8-2 Explanation of the Functions of Each Bit of the Standby Control Register (STBC)**

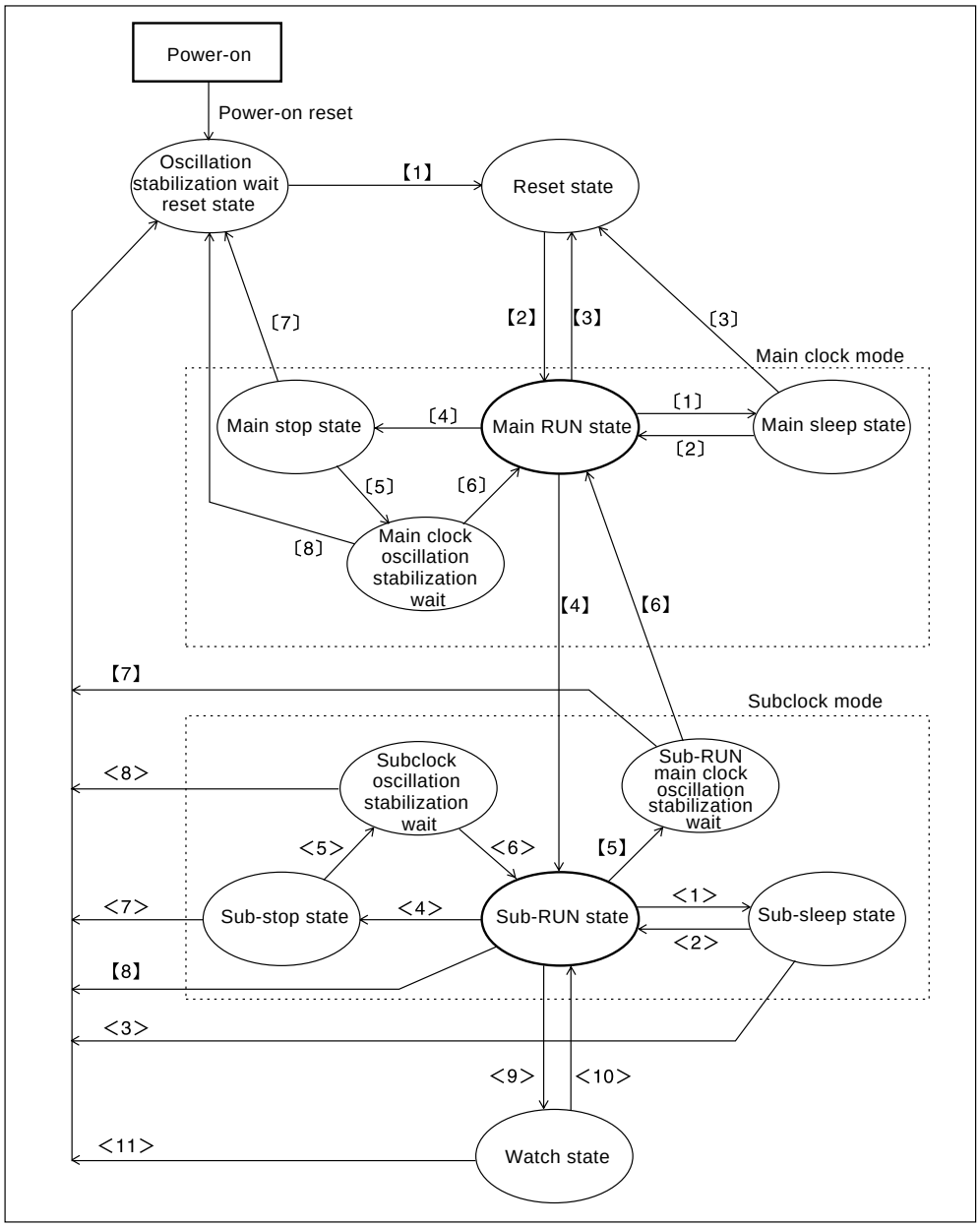
| Bit name | | Function |
|---|---|---|
| Bit 7 | STP:<br>Stop bit | • Bit to specify the transition to the stop mode<br>• A transition to the stop mode is caused by writing "1" into this bit.<br>• Operations are not affected if "0" is written into this bit.<br>• When this bit is read, "0" is always read. |
| Bit 6 | SLP:<br>Sleep bit | • Bit to specify the transition to the sleep mode<br>• A transition to the sleep mode is caused by writing "1" into this bit.<br>•<br>• Operations are not affected if "0" is written into this bit.<br>• When this bit is read, "0" is always read. |
| Bit 5 | SPL:<br>Pin state designate bit | • Bit to specify the external pin state in stop mode and watch mode<br>• If "0" is written into this bit, the external pin state (level) when making a transition to the stop or watch mode is retained.<br>• If "1" is written into this bit, the external pins are put into high impedance when making a transition to the stop or watch mode (The pins are raised to the "H" level if "pull-up resistor present" is selected in the pull-up setting register).<br>• This bit is set to "0" by a reset. |
| Bit 4 | RST:<br>Software reset bit | • Bit to specify the software reset<br>• An internal reset source in four instruction cycles caused by writing "0" into this bit.<br>• Operations are not affected if "1" is written into this bit.<br>• When this bit is read, "1" is always read.<br>**[Reference:]**<br>If the software reset is triggered in subclock mode, operation starts in main clock mode after taking the oscillation stabilization wait time. Thus, the reset signal is output during oscillation stabilization wait time. |
| Bit 3 | TMD:<br>Watch bit | • Bit to specify the transition to the watch mode<br>• Write operation to this bit is valid only in subclock mode (SYCC: SCS=0).<br>• A transition to the watch mode is caused by writing "1" into this bit.<br>• Operations are not affected if "0" is written into this bit.<br>• When this bit is read, "0" is always read. |
| Bit 2<br>Bit 1<br>Bit 0 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |

# 3.8.6 State Transition Diagram 1 (Dual Clock)

**This section shows a state transition diagram when the dual clock is used.**

## ■ State Transition Diagram 1 (Dual Clock)

**Figure 3.8-2 State Transition Diagram 1 (Dual Clock)**



❍ **Transition and release of the clock mode (non-standby mode)**

**Table 3.8-3 Transition and Release of the Clock Mode (Power-on Reset is Available, Dual Clock System)**

| State transition | Transition conditions |
|---|---|
| Transition to the normal state (main RUN) in main clock mode after power-on reset | [1] Main clock oscillation stabilization wait time end (timebase timer output)<br>[2] Release of reset input |
| Reset in the main RUN state | [3] External reset, software reset, or watchdog reset |
| Transition from the main RUN state to the sub-RUN state | [4] SYCC:SCS=0(*1) |
| Return from the sub-RUN state to the main RUN state | [5] SYCC:SCS=1<br>[6] Main clock oscillation stabilization wait time end (SYCC: can be checked by SCM)<br>[7] External reset, software reset, or watchdog reset |
| Reset in the sub-RUN state | [8] External reset, software reset, or watchdog reset |

SYCC: System clock control register

*1: A transition to the sub-RUN just after power-on occurs after the passage of the subclock oscillation stabilization wait time.

### ❍ Transition and release of the standby mode

**Table 3.8-4 Transition and Release of the Standby Mode (Power-on Reset is Available, Dual Clock System)**

| State transition | Transition conditions | |
|---|---|---|
| | Main clock mode | Subclock mode |
| Transition to the sleep mode | [1] STBC:SLP=1 | <1> STBC:SLP=1 |
| Release of the sleep mode | [2] Interrupt (various types)<br>[3] External reset | <2> Interrupt (various types)<br><3> External reset |
| Transition to the stop mode | [4] STBC:STP=1 | <4> STBC:STP=1 |
| Release of the stop mode | [5] External interrupt<br>[6] Main clock oscillation stabilization wait time end (timebase timer output)<br>[7] External reset<br>[8] External reset (Waiting for oscillation stabilization) | <5> External interrupt<br><6> Subclock oscillation stabilization wait time end (watch prescaler output)<br><7> External reset<br><8> External reset (Waiting for oscillation stabilization) |
| Transition to the watch mode | - | <9> STBC:TMD=1(*1) |
| Release of the watch mode | - | <10> External interrupt or watch interrupt<br><11> External reset |

STBC: Standby control register

*1: A transition to the watch mode is only possible from the sub-RUN state (SYCC: SCS=0).

**Note:**

Since CPU and the watchdog timer are stopped in standby mode, no software reset and watchdog reset occur. When a single clock system is used, a transition to the subclock mode is prohibited. If a transition to the subclock mode occurs, CPU stops and there is no other way to return other than resetting.

## 3.8.7 Notes on Using Standby Mode

**A transition to the standby mode does not occur even if the standby mode is set on the standby control register (STBC) when an interrupt request has arrived from a peripheral function. When returning to a normal operation state from the standby mode caused by an interrupt, operations after the return are dependent on whether or not the interrupt request is accepted.**

### ■ Transition to the Standby Mode and Interrupts

When an interrupt request whose interrupt priority is higher than "11" arrives at CPU from a peripheral function, a transition to the standby mode does not occur if "1" is written into the stop bit (STBC: STP), sleep bit (SLP) or clock bit (TMD) of the standby control register because such write operations are ignored (No transition to the standby mode occurs even after the interrupt processing is completed).

This is not related to whether the interrupt is accepted by CPU.

Even if CPU is processing an interrupt, a transition to the standby mode is possible if the interrupt request flag bit is cleared and there are no other interrupt requests.

### ■ Release of the Standby Mode by an Interrupt

The standby mode is released if an interrupt request whose interrupt priority is higher than "11" comes from the peripheral functions in sleep mode or stop mode. This is not related to whether the interrupt is accepted by CPU.

After releasing the standby mode, if the priority of the interrupt level setting register (ILR1- ILR4) corresponding to the interrupt request is higher than the interrupt level bits (CCR: IL1, 0) of the condition code register and if the interrupt enable flag allows the interrupts (CCR: I=1), branching to an interrupt processing routine occurs. If the interrupt is not accepted, execution of the instruction following the instruction starting the standby mode is restarted.

If no branching to an interrupt processing routine just after returning occurs, measures such as an interrupt prohibition are needed before setting the standby mode.

## ■ Precaution in Setting the Standby Mode

To set the standby mode using the standby control register (STBC), follow Table 3.8-5 "Low Power Consumption Settings by the Standby Control Register (STBC)". The priority order when "1" is written into these bits is the stop mode, watch mode, and sleep mode. However, it is preferable to set "1" to one bit at a time.

Do not make a transition to the stop mode, sleep mode, and watch mode just after switching from the subclock mode to the main clock mode (SYCC: SCS=0 --> 1). Make a transition to these modes after checking that the clock monitor bit (SYCC: SCM) of the system control register is "1".

However, the content written into the clock bit (TMD) is ignored during operation in main clock mode.

**Table 3.8-5 Low Power Consumption Settings by the Standby Control Register (STBC)**

| STBC register | | | Mode |
|---|---|---|---|
| STP (bit 7) | SLP (bit 6) | TMD (bit 3) | |
| 0 | 0 | 0 | Normal |
| 0 | 0 | 1 | Clock |
| 0 | 1 | 0 | Watch |
| 1 | 0 | 0 | Stop |

## ■ Oscillation Stabilization Wait Time

Since the oscillator for oscillation is stopped in stop mode for both the main clock mode and subclock mode, it is necessary to take the oscillation stabilization wait time after the oscillator in each mode starts operation.

As the oscillation stabilization wait time in main clock mode, take the oscillation stabilization wait time of the main clock created by the timebase timer (Select one from three different kinds of wait time). As the oscillation stabilization wait time in subclock mode, take the oscillation stabilization wait time of the subclock created by the watch prescaler.

In main clock mode, if the selected interval time of the timebase timer is shorter than the oscillation stabilization wait time, an interval timer interrupt request may occur during oscillation stabilization wait time. Before making a transition to the stop mode in main clock mode, prohibit (TBTC: TBIE=0) the interrupt request output of the timebase timer if necessary.

Likewise, a watch interrupt request may occur depending on the selected interval time of the watch prescaler. Before making a transition to the stop mode in subclock mode, prohibit (WPCR: WIE=0) the watch interrupt request output of the watch prescaler if necessary.

# 3.9 Memory Access Mode

**The operation mode for memory access of the MB89570 series is the single chip mode only.**

## ■ Single Chip Mode

The single chip mode uses only the internal RAM and ROM. Thus, CPU can access only the internal I/O area, RAM area, and ROM area (internal access).

## ■ Mode Pin (MODA)

Set always "Vss" to the mode pin (MODA).

The mode data and reset vector are read from the internal ROM during reset.

Do not change the settings of the mode pin after the reset operation (during operation) is completed.

Table 3.9-1 "Settings of the Mode Pin" lists the settings of the mode pin.

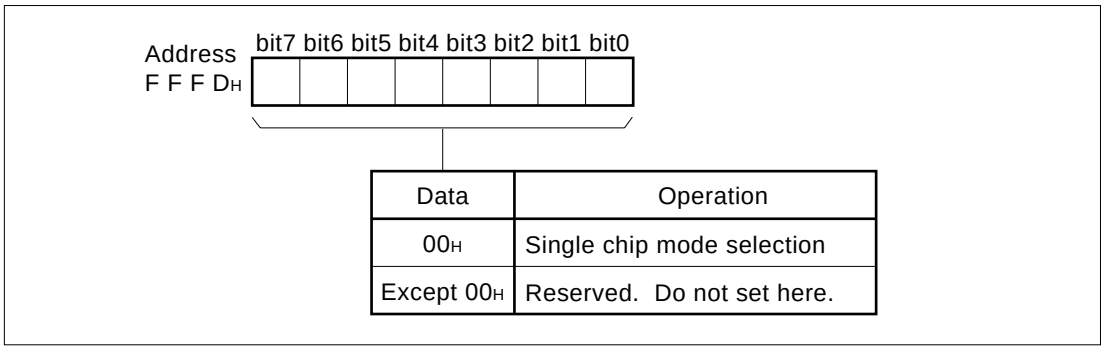
**Table 3.9-1 Settings of the Mode Pin**

| Pin state<br>MPOA | Contents |
|---|---|
| Vss | The mode data and reset vector are read from the internal ROM |
| Vcc | Setting prohibited |

## ■ Mode Data

Set always $00_H$ as the mode data in the internal ROM to select the single chip mode.

**Figure 3.9-1 Structure of the Mode Data**



## ■ Selection of the Memory Access Mode

Only the single chip mode can be selected.

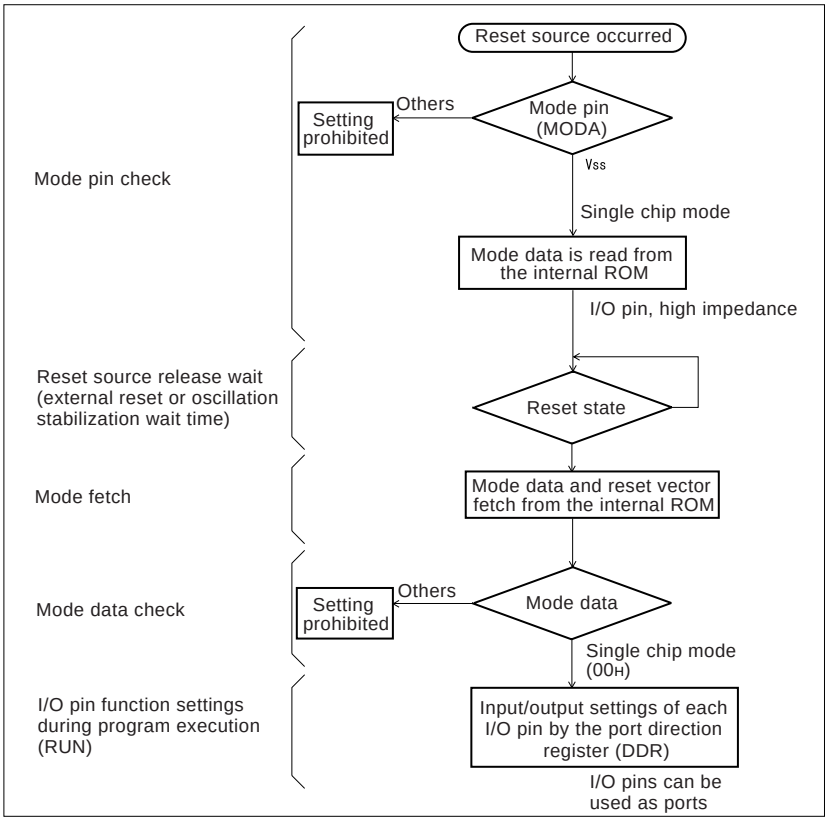
Table 3.9-2 "Mode pins and mode data" lists the mode pins and mode data.

**Table 3.9-2 Mode pins and mode data**

| Memory access mode | Mode pin (MODA) | Mode data |
|---|---|---|
| Single chip mode | Vss | $00_H$ |
| Other modes | Setting prohibited | Setting prohibited |

Figure 3.9-2 "Memory Access Selection Operation" shows the Operation of Memory Access Selection.

**Figure 3.9-2 Memory Access Selection Operation**



# CHAPTER 4 I/O PORT

**This chapter describes the functions and operations of the I/O port.**


4.1 "Overview of the I/O Port"

4.2 "Port 0"

4.3 "Port 1"

4.4 "Port 2"

4.5 "Port 3"

4.6 "Port 4"

4.7 "Port 5"

4.8 "Port 6"

4.9 "Port 7"

4.10 "Port 8"

4.11 "Port 9"

4.12 "Port A"

4.13 "Port B"

4.14 "Program Example of the I/O Ports"

# 4.1 Overview of the I/O Port

**The I/O port can be used as 12 (82 pins) general-purpose I/O ports (parallel I/O ports). Each port also serves for resources (I/O pins for various peripheral functions).**

## ■ Functions of the I/O Port

The I/O port provides the function to output data from CPU to the I/O pins and fetch the signal input into the I/O pin to send it to CPU by using the port data register (PDR). For some ports, it is possible to set the direction of I/O of the I/O pin in bit units using the port direction register (DDR).

The following lists the functions of each port and the resources that each port provides.

- Port 0: General-purpose I/O port
- Port 1: Serves as the general-purpose I/O port/resource (A/D converter pin)
- Port 2: Serves as the general-purpose I/O port/resource (timer output pin)
- Port 3: Serves as the general-purpose I/O port/resource ($I^2C$/multi-address $I^2C$, UAET/SIO pin)
- Port 4: Serves as the general-purpose I/O port/resource (bridge circuit pin)
- Port 5: Serves as the general-purpose I/O port/resource (comparator pin)
- Port 6: Serves as the general-purpose I/O port/resource (LCD controller/driver segment output pin)
- Port 7: Serves as the general-purpose I/O port/resource (comparator pin)
- Port 8: Serves as the general-purpose I/O port/resource (external interrupt, A/D converter, comparator pin)
- Port 9: Serves as the general-purpose I/O port/resource (A/D and D/A converter pin)
- Port A: Serves as the general-purpose I/O port/resource (LCD controller/driver segment output pin)
- Port B: Serves as the general-purpose I/O port/resource (LCD controller/driver segment output pin, LCD built-in split resistance input pin)

Table 4.1-1 "List of Functions of Each Port" lists the functions of each port and Table 4.1-2 "List of Functions of Each Port" lists the registers of each port.

**Table 4.1-1 List of Functions of Each Port**

| Port name | Pin name | Input type | Output type | Function | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Port 0 | P00 to P07 | CMOS | CMOS push-pull | General-purpose I/O port | P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |
| Port 1 | P10/AN4 to P17AN11 | | | General-purpose I/O port | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| | | | | Resources | AN11 | AN10 | AN9 | AN8 | AN7 | AN6 | AN5 | AN4 |
| Port 2 | P20 to P27 | | | General-purpose I/O port | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| | | | | Resources | – | – | – | – | T02 | – | – | T01 |
| Port 3 | P30/SCL1 to P35/U03 | CMOS(*1) | N-ch open-drain | General-purpose I/O port | – | – | P35 | P34 | P33 | P32 | P31 | P30 |
| | | | | Resources | – | – | U03 | SDA2/ UI3 | SCL2/ UCK3 | ALERT | SDA1 | SCL1 |
| Port 4 | P40/SCL3/ UCK1 to P43/SDA4/ UI2 | CMOS(*2) | N-ch open-drain | General-purpose I/O port | – | – | – | – | P43 | P42 | P41 | P40 |
| | | | | Resources | – | – | – | – | SDA4/ UI2 | SCL4/ UCK2 | SDA3\UI 1 | SCL3/ UCK1 |
| Port 5 | P50/ALR1 to P56/OFB3 | CMOS | CMOS push-pull | General-purpose I/O port | – | P56 | P55 | P54 | P53 | P52 | P51 | P50 |
| | | | | Resources | – | OFB3 | OFB2 | OFB1 | AC0 | ALR3 | ALR2 | ALR1 |
| Port 6 | P60/SEG08 to P65/ SEG13/U01 | CMOS(*3) | N-ch(*3) open-drain | General-purpose I/O port | | | P65 | P64 | P63 | P62 | P61 | P60 |
| | | | | Resources | | | SEG13/ U01 | SEG12/ U02 | SEG11 | SEG10 | SEG9 | SEG8 |
| Port 7 | P70/DCIN to P77/VSI3 | CMOS | CMOS push-pull | General-purpose I/O port | P77 | P76 | P75 | P74 | P73 | P72 | P71 | P70 |
| | | | | Resources | VSI3 | VOL3 | VSI2 | VOL2 | VSI1 | VOL1 | DCIN2 | DCIN |
| Port 8 | P80/INT0 to P87/AN2/ SW3 | CMOS(*4) | CMOS(*5) push-pull | General-purpose I/O port | P87 | P86 | P85 | P84 | P83 | P82 | P81 | P80 |
| | | | | Resources | AN2/ SW3 | AN1/ SW2 | AN0/ SW2 | EC | INT3 | INT2 | INT1 | INT0 |
| Port 9 | P90/AN3 to P92/DA2 | CMOS | CMOS push-pull | General-purpose I/O port | – | – | – | – | – | P92 | P91 | P90 |
| | | | | Resources | – | – | – | – | – | DA2 | DA1 | AN3 |
| Port A | PA0/SEG00 to PA7/ SEG07 | CMOS | N-ch open-drain | General-purpose I/O port | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| | | | | Resources | SEG07 | SEG06 | SEG05 | SEG04 | SEG03 | SEG02 | SEG01 | SEG00 |

**Table 4.1-1 List of Functions of Each Port (Continued)**

| Port name | Pin name | Input type | Output type | Function | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Port B | PB0/V0 to PB7/COM3 | CMOS(*6) | N-ch open-drain | General-purpose I/O port | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| | | | | Resources | COM3 | COM2 | COM1 | COM0 | V3 | V2 | V1 | V0 |

*1: Resources (UART) of P33 to P34 are hysteresis input.
*2: Resources become the input into the bridge circuit. Resources (UART) are hysteresis input.
*3: Resources of P64 to P65 (UART) are the output from the bridge circuit.
*4: Resources of P80 to P83 (for external interrupts) are hysteresis input.
*5: No output for P84
*6: PB0 to PB3 is output only.

**Table 4.1-2 List of Registers of Each Port**

| Register name | Read/write | Address | Initial value |
|---|---|---|---|
| Port 0 data register (PDR0) | R/W | $0000_H$ | $XXXXXXXX_B$ |
| Port 0 direction register (DDR0) | W | $0001_H$ | $00000000_B$ |
| Port 1 data register (PDR1) | R/W | $0002_H$ | $XXXXXXXX_B$ |
| Port 1 direction register (DDR1) | W | $0003_H$ | $00000000_B$ |
| Port 2 data register (PDR2) | R/W | $0004_H$ | $XXXXXXXX_B$ |
| Port 2 direction register (DDR2) | R/W | $0006_H$ | $00000000_B$ |
| Port 3 data register (PDR3) | R/W | $0020_H$ | $XX111111_B$ |
| Port 4 data register (PDR4) | R/W | $0021_H$ | $XXXX1111_B$ |
| Port 5 data register (PDR5) | R/W | $0022_H$ | $XXXXXXXX_B$ |
| Port 5 direction register (DDR5) | R/W | $0023_H$ | $X0000000_B$ |
| Port 6 data register (PDR6) | R/W | $0024_H$ | $XX111111_B$ |
| Port 7 data register (PDR7) | R/W | $0025_H$ | $XXXXXXXX_B$ |
| Port 7 direction register (DDR7) | R/W | $0026_H$ | $00000000_B$ |
| Port 8 data register (PDR8) | R/W | $0027_H$ | $XXXXXXXX_B$ |
| Port 8 direction register (DDR8) | R/W | $0028_H$ | $000X0000_B$ |
| Port 9 data register (PDR9) | R/W | $0029_H$ | $XXXXXXXX_B$ |
| Port 9 direction register (DDR9) | R/W | $002A_H$ | $XXXXX000_B$ |
| Port A data register (PDRA) | R/W | $0015_H$ | $11111111_B$ |
| Port B data register (PDRB) | R/W | $0017_H$ | $11111111_B$ |

R/W: Read/write enabled
R: Read only
W: Write only
X: Undefined

# 4.2 Port 0

**The port 0 is a general-purpose I/O port.**
**This section describes with a particular emphasis on the functions of the general-purpose I/O port.**
**The following shows the configuration of port 0, its pins, a pin block diagram, and related registers.**

## ■ Configuration of Port 0

The port 0 is made up of the following three elements:

❍ **Port 0**

- **General-purpose I/O pin (P00 to P07)**
- Port 0 data register (PDR0)
- Port 0 direction register (DDR0)

## ■ Pins of the Port 0

The port 0 has eight CMOS I/O pins.

Table 4.2-1 "Pins of Port 0" lists the pins of port 0.

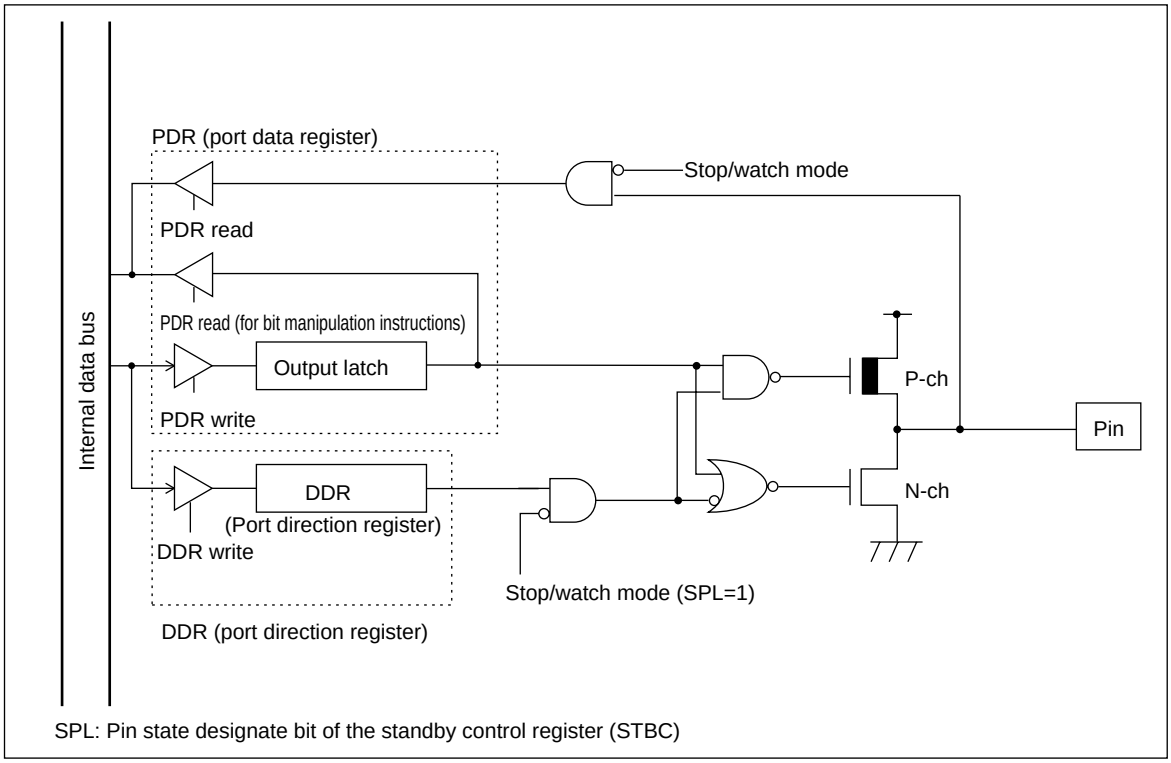
**Table 4.2-1 Pins of Port 0**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 0 | P00 | P00 General-purpose I/O | - | CMOS | CMOS | B |
| | P01 | P01 General-purpose I/O | - | | | |
| | P02 | P02 General-purpose I/O | - | | | |
| | P03 | P03 General-purpose I/O | - | | | |
| | P04 | P04 General-purpose I/O | - | | | |
| | P05 | P05 General-purpose I/O | - | | | |
| | P06 | P06 General-purpose I/O | - | | | |
| | P07 | P07 General-purpose I/O | - | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of the Port 0

**Figure 4.2-1 Block diagram of pins of port 0**




**Note:**

Do not use the pins to be used as analog input pins as a general-purpose port.

## ■ Registers PDR0 and DDR0 of Port 0

Two registers PDR0 and DDR0 are available as the registers related to port 0.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 0.

lists the correspondences between the registers and pins of port 0.

**Table 4.2-2 Correspondence between the Registers and Pins of Port 0**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 0 | PDR0, DDR0 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |

# 4.2.1 Registers of Port 0 (PDR0, DDR0)

**This section describes the registers related to port 0.**

## ■ Functions of the Registers of port 0

### ❍ Port 0 data register (PDR0)

The PDR0 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 0 direction register (DDR0)

The DDR0 register sets the I/O direction of pins for each bit.

If "1" is set to a bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port

Table 4.2-3 "Register Functions of Port 0" lists the register functions of port 0.

**Table 4.2-3 Register Functions of Port 0**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 0 data register (PDR0) | 0 | Pin state is "L" | "0" is set to the output latch. If port 0 operates as an output port, the "L" level is output to the pins. | R/W | $0000_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 0 operates as an output port, the "H" level is output to the pins. | | | |
| Port 0 direction register (DDR0) | 0 | Read not allowed | Output transistor operation is prohibited and a pin is made an input pin. | W | $0001_H$ | $00000000_B$ |
| | 1 | Read not allowed | Output transistor operation is allowed and a pin is made an output pin | | | |

R/W: Read/write enabled
W: Write only
X: Undefined

# 4.2.2 Operation of Port 0

**This section describes the operations of port 0.**

## ■ Operation of Port 0

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR0 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 0 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR0 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR0 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR0 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 0 operates as an input port.
- If data is written into the PDR0 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR0 register.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR0 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR0 register is not initialized by a reset. Thus, if port 0 is used as an output port, it is necessary to set output data to the PDR0 register and then set output to the corresponding DDR0 register.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, prohibition of the port input is forced regardless of the value of the DDR0 register and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.2-4 "Pin States of Port 0" lists the pin states of port 0.

**Table 4.2-4 Pin States of Port 0**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P00 to P07 | General-purpose I/O port | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.3 Port 1

**The port 1 is a general-purpose I/O port.**
**This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 1, its pins, a pin block diagram, and related registers.**

## ■ Configuration of Port 1

The port 1 is made up of the following three elements:

- General-purpose I/O pin/analog input pin (P10/AN5 to P17/AN12)
- Port 1 data register (PDR1)
- Port 1 direction register (DDR1)

## ■ Pins of Port 1

The port 1 has eight CMOS I/O pins.

If these pins are used as analog input pins by the A/D converter, do not use them as a general-purpose I/O port.

Table 4.3-1 "Pins of Port 1" lists the pins of port 1.

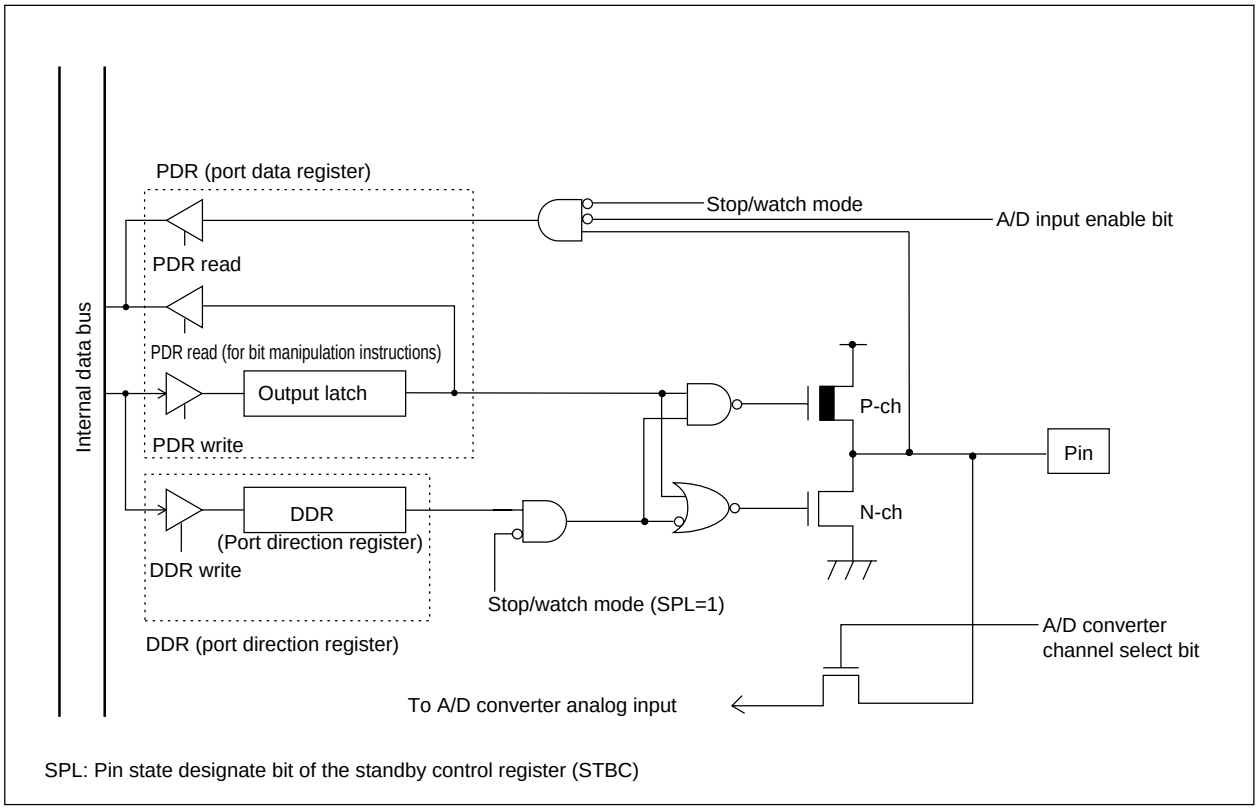
**Table 4.3-1 Pins of Port 1**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 1 | P10/AN4 | P10 General-purpose I/O | Analog input | Analog /CMOS | CMOS | E |
| | P11/AN5 | P11 General-purpose I/O | Analog input | | | |
| | P12/AN6 | P12 General-purpose I/O | Analog input | | | |
| | P13/AN7 | P13 General-purpose I/O | Analog input | | | |
| | P14/AN8 | P14 General-purpose I/O | Analog input | | | |
| | P15/AN9 | P15 General-purpose I/O | Analog input | | | |
| | P16/AN10 | P16 General-purpose I/O | Analog input | | | |
| | P17/AN11 | P17 General-purpose I/O | Analog input | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 1

**Figure 4.3-1 Block Diagram of the Pins of Port 1**




## ■ Registers of Port 1

Two registers PDR1 and DDR1 are available as registers related to port 1.

There is a 1:1 correspondence between the bits configuring each register and pins of port 1.

Table 4.3-2 "Correspondence between the Registers and Pins of Port 1" lists the correspondences between the registers and pins of port 1.

**Table 4.3-2 Correspondence between the Registers and Pins of Port 1**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 1 | PDR1, DDR1 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |

# 4.3.1 Registers of the Port 1 (PDR1, DDR1)

**This section describes the registers related to port 1.**

## ■ Functions of the Registers of Port 1

### ❍ Port 1 data register (PDR1)

The PDR1 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins are set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 1 direction register (DDR1)

The DDR1 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port.

Table 4.3-3 "Register Functions of Port 1" lists the register functions of port 1.

**Table 4.3-3 Register Functions of Port 1**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 1 data register (PDR1) | 0 | Pin state is "L" | "0" is set to the output latch. If port 1 operates as an output port, the "L" level is output to the pins. | R/W | $0002_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 1 operates as an output port, the "H" level is output to the pins. | | | |
| Port 1 direction register (DDR1) | 0 | Read not allowed | Output transistor operation is prohibited and a pin is made an input pin (analog input enabled). | W | $0003_H$ | $00000000_B$ |
| | 1 | Read not allowed | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
W: Write only
X: Undefined

## ■ Register Related to Port 1

### ❍ A/D port input enable register (ADEN2) $002E_H$

To use port 1 for analog input, set "1" to the bit corresponding to the ADEN2 register to help prevent the DC pass when an intermediate level is entered.

**Note:**

To use the port 1 for port input, "0" must be set to the input enable bit of the ADEN2 register.

# 4.3.2 Operation of the Port 1

**This section describes the operations of port 1.**

## ■ Operation of Port 1

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR1 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 1 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR1 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR1 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR1 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 1 operates as an input port.
- If data is written into the PDR1 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR1 register.

### ❍ Operation for analog input

- To use port 1 for analog input, write "0" into the bit of the DDR1 register corresponding to the analog input pin or "1" into the corresponding bit of the ADEN2 register.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR1 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR1 register is not initialized by a reset. Thus, if port 1 is used as an output port, it is necessary to set output data to the PDR1 register and then set output to the corresponding DDR1 register.
- The ADEN2 register is initialized "1" by a reset. Thus, if port 1 is used for port input, "0" must be set to the corresponding bit of the ADEN2 register.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, prohibition of the port input is effected regardless of the value of the DDR1 register and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.3-4 "Pin States of Port 1" lists the pin states of port 1.

**Table 4.3-4 Pin States of Port 1**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P10/AN4 to P17/AN11 | General-purpose I/O port/analog input | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.4 Port 2

**The port 2 is a general-purpose I/O port.**
**This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 2, its pins, a pin block diagram, and related registers.**

## ■ Configuration of Port 2

The port 2 is made up of the following three elements:

- General-purpose I/O pin (P20/T01 to P27)
- Port 2 data register (PDR2)
- Port 2 direction register (DDR2)

## ■ Pins of the Port 2

Port 2 has eight CMOS I/O pins.

Of these pins, if resources are used by the pins also serving for resources, do not use as a general-purpose I/O port.

Table 4.4-1 "Pins of Port 2" lists the pins of port 2.

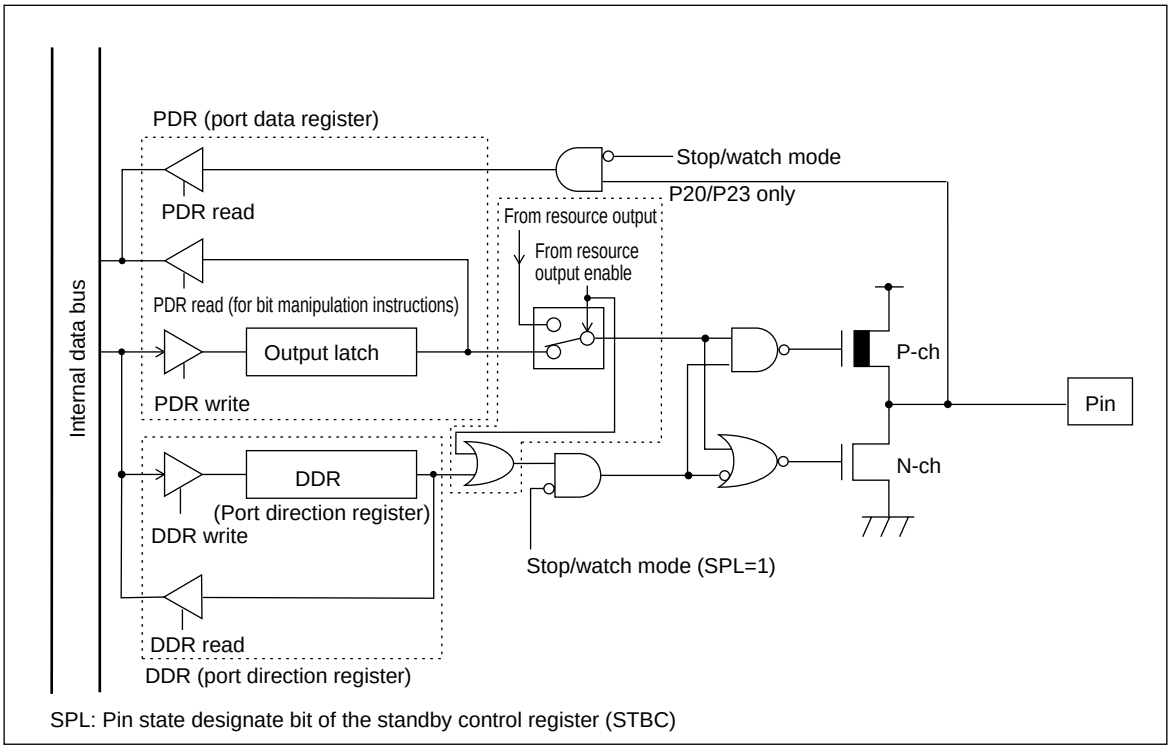
**Table 4.4-1 Pins of Port 2**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 2 | P20/T01 | P20 General-purpose I/O | 8/16-bit timer/counter, timer 1 output | CMOS | CMOS | B |
| | P21 | P21 General-purpose I/O | – | | | |
| | P22 | P22 General-purpose I/O | – | | | |
| | P23/T02 | P23 General-purpose I/O | 8/16-bit timer/counter, timer 2 output | | | |
| | P24 | P24 General-purpose I/O | – | | | |
| | P25 | P25 General-purpose I/O | – | | | |
| | P26 | P26 General-purpose I/O | – | | | |
| | P27 | P27 General-purpose I/O | – | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 2

**Figure 4.4-1 Block Diagram of Pins of Port 2**




## ■ Registers of the Port 2

Two registers PDR2 and DDR2 are available as the registers related to port 2.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 2.

Table 4.4-2 "Correspondence between the Registers and Pins of Port 2" lists the correspondence between the registers and pins of port 2.

**Table 4.4-2 Correspondence between the Registers and Pins of Port 2**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 2 | PDR2, DDR2 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |

# 4.4.1 Registers of Port 2 (PDR2, DDR2)

**This section describes the registers related to port 2.**

## ■ Functions of the Registers of Port 2

### ❍ Port 2 data register (PDR2)

The PDR2 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 2 direction register (DDR2)

The DDR2 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port

### ❍ Settings for resource output

To use a resource, set the operation enable bit of the resource.

Since the resource output takes precedence, settings of the PDR2 and DDR2 registers corresponding to the resource output pins have no significance regardless of the output value and output permission of the resource.

Table 4.4-3 "Register Functions of Port 2" lists the register functions of port 2.

**Table 4.4-3 Register Functions of Port 2**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 2 data register (PDR2) | 0 | Pin state is "L" | "0" is set to the output latch. If port 2 operates as an output port, the "L" level is output to the pins. | R/W | $0004_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 2 operates as an output port, the "H" level is output to the pins. | | | |
| Port 2 direction register (DDR2) | 0 | Input port state | Output transistor operation is prohibited and a pin is made an input pin. | R/W | $0006_H$ | $00000000_B$ |
| | 1 | Output port state | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
X: Undefined

# 4.4.2 Operation of Port 2

**This section describes the operations of port 2.**

## ■ Operation of Port 2

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR2 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 2 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR2 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR2 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR2 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 2 operates as an input port.
- If data is written into the PDR2 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR2 register.

### ❍ Operation during resource output

- If the operation enable bit of a resource is set, the corresponding pin is made ready for resource output.
- Because the pin values can be read through the PDR2 register even when a resource is allowed, output values of the resource can be read.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR2 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR2 register is not initialized by a reset. Thus, if port 2 is used as an output port, it is necessary to set output data to the PDR2 register and then set output to the corresponding DDR2 register.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, prohibition of the port input is forced regardless of the value of the DDR2 register and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.4-4 "Pin States of Port 2" lists the pin states of port 2.

**Table 4.4-4 Pin States of Port 2**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P20 to P27 | General-purpose I/O port | Hi-Z(*1) | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance
*1: Note that P20 and P23 will not be in high impedance even if (STBC=SPL) is set to "1".

# 4.5 Port 3

**The port 3 is a general-purpose I/O port. Each pin can be used by switching between the resource and port in bit units. This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 3, its pins, pin block diagrams, and the related register.**

## ■ Configuration of Port 3

Port 3 is made up of the following two elements:

- General-purpose I/O pin/resource I/O pin (P30/SCL1 to P35/U03)
- Port 3 data register (PDR3)

## ■ Pins of the Port 3

Port 3 has six CMOS input/N-ch open-drain output I/O pins. Table 4.5-1 "Pins of Port 3" lists the pins of port 3.

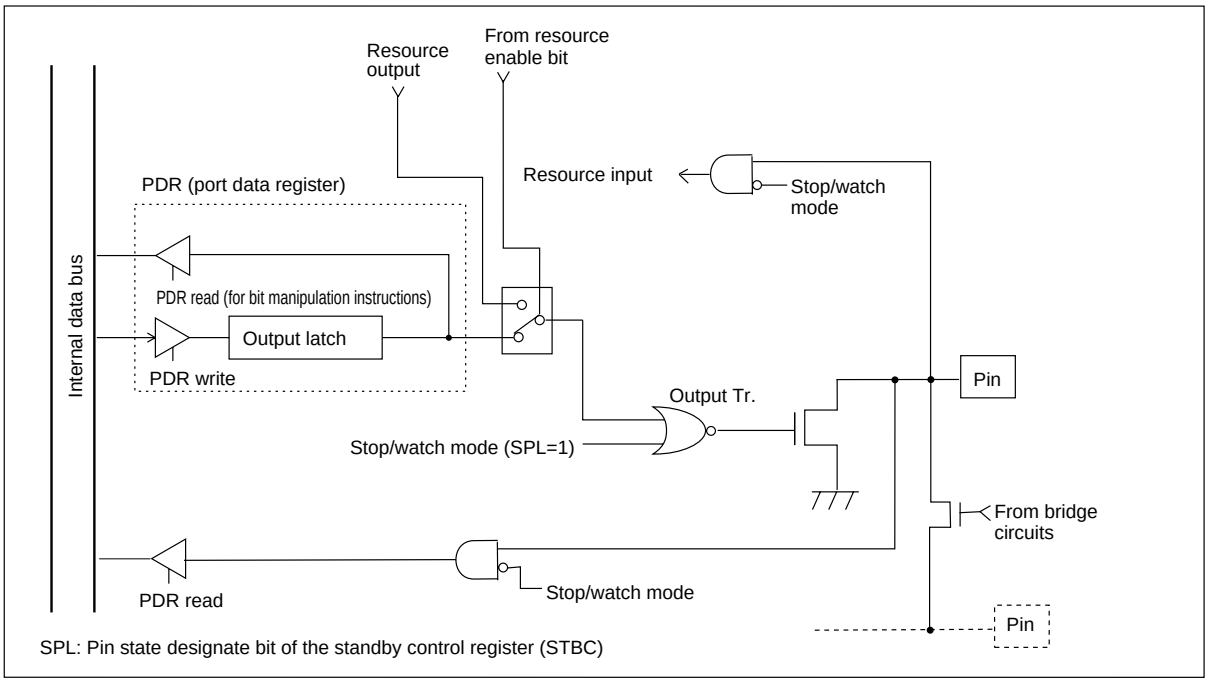
**Table 4.5-1 Pins of Port 3**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 3 | P30 | General-purpose I/O | SCL1 multi-address $I^2C$ | CMOS | N-ch open-drain | F |
| | P31 | General-purpose I/O | SDA1 multi-address $I^2C$ | | | |
| | P32 | General-purpose I/O | ALERT multi-address $I^2C$ | | | H |
| | P33 | General-purpose I/O | SCL2/UCK3 $I^2C$/UART | | | G |
| | P34 | General-purpose I/O | SDA2/UI3 $I^2C$/UART | | | |
| | P35 | General-purpose I/O | U03 UART | | | H |

For the circuit type, see Section 1.7 "Pin description".

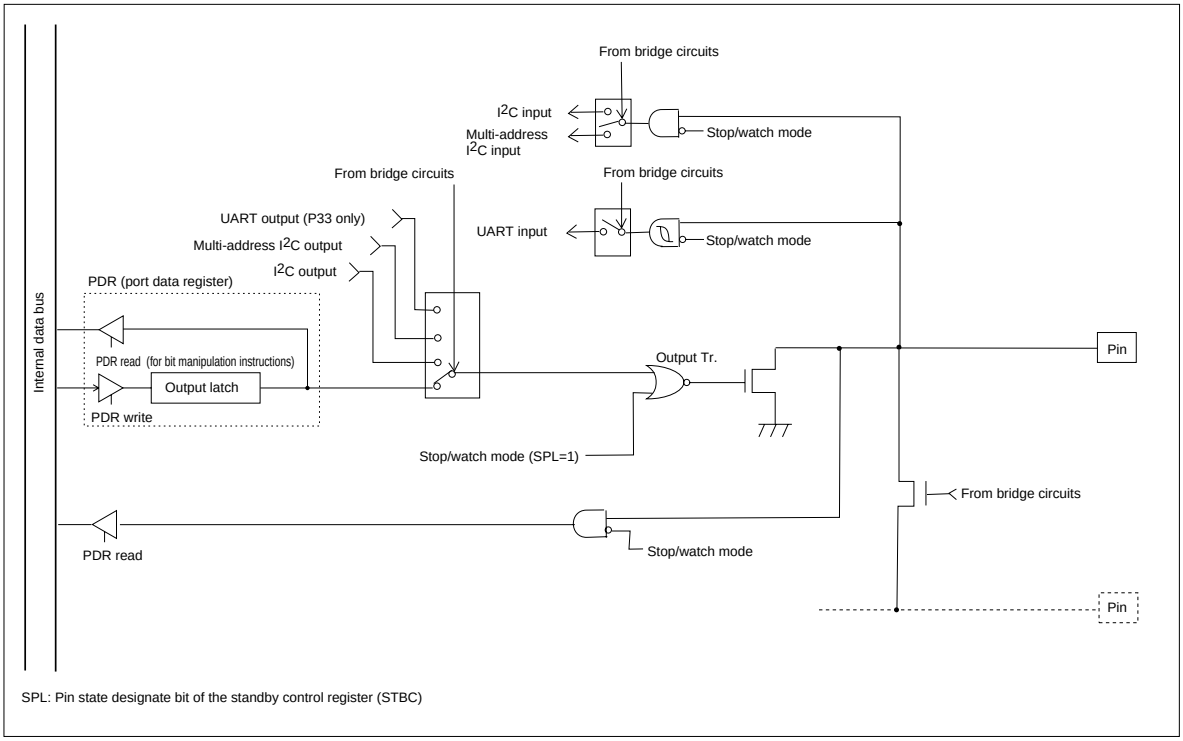
## ■ Block Diagram of Port 3

**Figure 4.5-1 Block Diagram of Pins of Port 3 (P30, P31)**



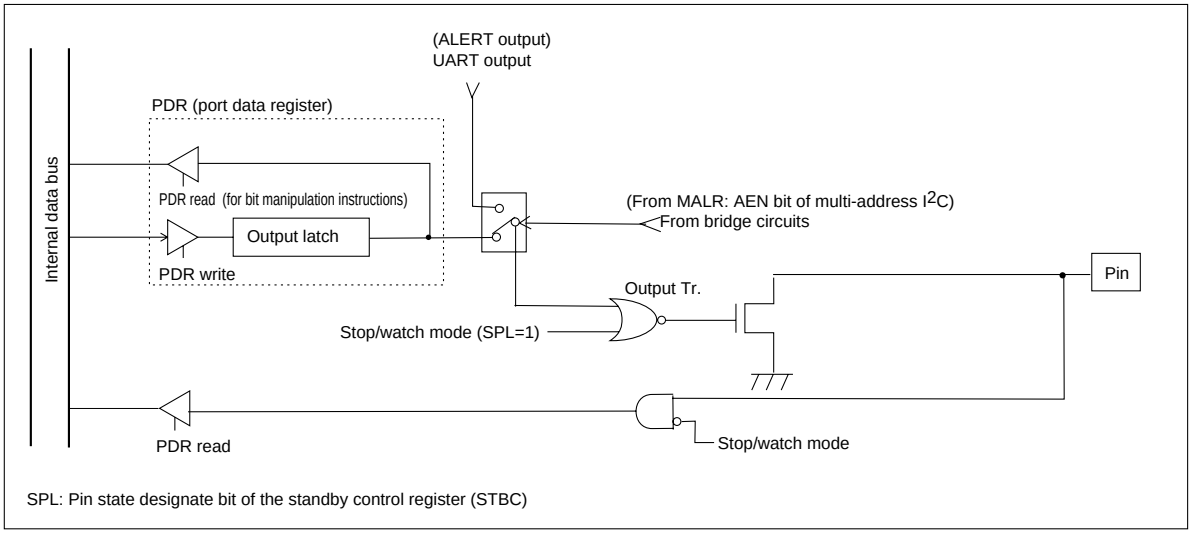

SPL: Pin state designate bit of the standby control register (STBC)

**Figure 4.5-2 Block Diagram of Pins of Port 3 (P33, P34)**




SPL: Pin state designate bit of the standby control register (STBC)

**Figure 4.5-3 Block Diagram of Pins of Port 3 (P32, P35)**




## ■ Register of Port 3

One register PDR3 is available as the register related to port 3.

There is a 1:1 correspondence between the bits configuring the PDR3 register and the pins of port 3.

Table 4.5-2 "Correspondence between the Registers and Pins of Port 3" lists the correspondence between the register and pins of port 3.

**Table 4.5-2 Correspondence between the Registers and Pins of Port 3**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 3 | PDR3 | – | – | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | – | – | P35 | P34 | P33 | P32 | P31 | P30 |

# 4.5.1 Register of Port 3 (PDR3)

**This section describes the register related to port 3.**

## ■ Functions of the Register of Port 3

### ❍ Port 3 data register (PDR3)

The PDR3 register sets the states of pins.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

Table 4.5-3 "Register Functions of Port 3" lists the register functions of port 3.

**Table 4.5-3 Register Functions of Port 3**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 3 data register (PDR3) | 0 | Pin state is "L" | The "L" level is output to the pins ("0" is set to the output latch and the output transistor is turned "ON"). | R/W | $0020_H$ | $XX111111_B$ |
| | 1 | Pin state is "Hi-Z" | Pins are raised to Hi-Z ("1" is set to the output latch and the output transistor is turned "OFF"). | | | |

R/W: Read/write enabled
X: Undefined
Hi-Z: High impedance

## ■ Registers Related to Port 3

### ❍ Bridge circuit select register 2 (BRSR2) $005D_H$, Bridge circuit select register 3 (BRSR3) $0019_H$

If port 3 is used as a general-purpose port, select the port function by using the BRSR2/ BRSR3 registers.

## 4.5.2 Operation of Port 3

**This section describes the operations of port 3.**

### ■ Operation of Port 3

❍ **Operation as an output port**

- If data is written into the PDR3 register, data is retained on the output latch. If the value of the output latch is "0", the output transistor is turned "ON" and the "L" level is output to the pins. If the value of the output latch is "1", the output transistor is turned "OFF" and the pins are put into high impedance. When the output pins are pulled up, the port is pulled up if the value of the output latch is "1".

❍ **Operation as an input port**

- Pin values can be read by reading the PDR3 register.

❍ **Operation during a reset**

- If CPU is reset, the value of the PDR3 register is initialized to "1". Thus, all output transistors are turned "OFF" and the pins are put into in high impedance.

❍ **Operation during resource output**

- To use resources, set the output enable bit of each resource. Since the resource output takes precedence, settings of the PDR3 register corresponding to the resource output pins have no significance.

❍ **Operation in stop mode and watch mode**

If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the output transistor is forced "OFF" and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.5-4 "Pin States of Port 3" lists the pin states of port 3.

**Table 4.5-4 Pin States of Port 3**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P30 to P35 | General-purpose I/O port/<br>resource I/O | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.6 Port 4

**Port 4 is a general-purpose I/O port. Each pin can be used by switching between the resource and port in units of bits. This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 4, its pins, a pin block diagram, and the related register.**

## ■ Configuration of Port 4

Port 4 is made up of the following two elements:

- General-purpose I/O pin/resource I/O pin (P40/SCL3 to P43/SDA4)
- Port 4 data register (PDR4)

## ■ Pins of the Port 4

Port 4 has four CMOS input/N-ch open-drain output I/O pins.

Table 4.6-1 "Pins of Port 4" lists the pins of port 4.

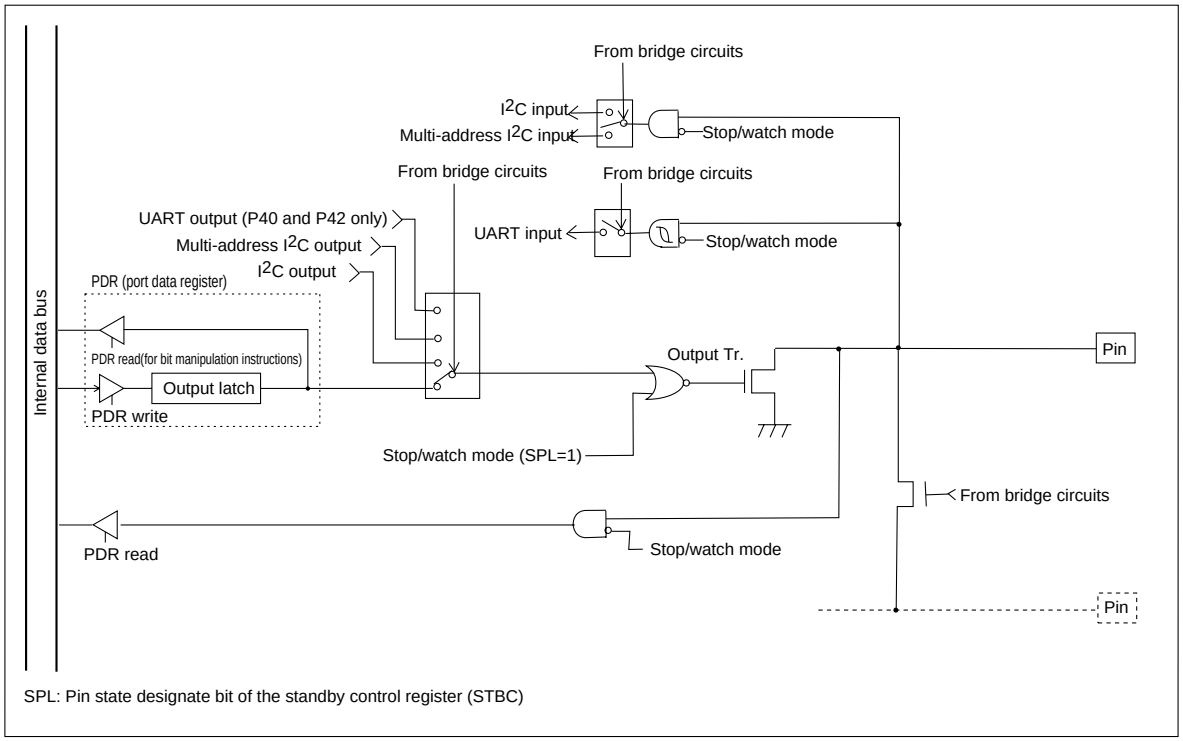
**Table 4.6-1 Pins of Port 4**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 4 | P40 | General-purpose I/O | SCL3/UCK1 bridge circuit | CMOS | N-ch open-drain | G |
| | P41 | General-purpose I/O | SDA3/UI1 bridge circuit | | | |
| | P42 | General-purpose I/O | SCL4/UCK2 bridge circuit | | | |
| | P43 | General-purpose I/O | SDA4/UI2 bridge circuit | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 4

**Figure 4.6-1 Block Diagram of Pins of Port 4**




SPL: Pin state designate bit of the standby control register (STBC)

## ■ Register of the Port 4

One register PDR4 is available as the register related to port 4.

There is a 1:1 correspondence between the bits configuring the PDR4 register and the pins of port 4.

Table 4.6-2 "Correspondence between the Register and Pins of Port 4" lists the correspondences between the register and pins of port 4.

**Table 4.6-2 Correspondence between the Register and Pins of Port 4**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 4 | PDR4 | – | – | – | – | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | – | – | – | – | P43 | P42 | P41 | P40 |

# 4.6.1 Register of Port 4 (PDR4)

**This section describes the register related to port 4.**

## ■ Functions of the Register of Port 4

### ❍ Port 4 data register (PDR4)

The PDR4 register sets the states of pins.

Since the values of the output latch instead of the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

Table 4.6-3 "Register Functions of Port 4" lists the register functions of port 4.

**Table 4.6-3 Register Functions of Port 4**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 4 data register (PDR4) | 0 | Pin state is "L" | The "L" level is output to the pins ("0" is set to the output latch and the output transistor is turned "ON"). | R/W | $0021_H$ | $XXXX1111_B$ |
| | 1 | Pin state is "Hi-Z" | Pins are raised to Hi-Z ("1" is set to the output latch and the output transistor is turned "OFF"). | | | |

R/W: Read/write enabled
X: Undefined
Hi-Z: High impedance

## ■ Registers Related to Port 4

### ❍ Bridge circuit select register 2 (BRSR2) $005D_H$, Bridge circuit select register 3 (BRSR3) $0019_H$

If port 4 is used as a general-purpose port, select the port function by using the BRSR2/ BRSR3 registers.

# 4.6.2 Operation of port 4

**This section describes the operations of port 4.**

## ■ Operation of Port 4

### ❍ Operation as an output port

- If data is written into the PDR4 register, the data is retained on the output latch. If the value of the output latch is "0", the output transistor is turned "ON" and the "L" level is output to the pins. If the value of the output latch is "1", the output transistor is turned "OFF" and the pins are put into high impedance. When the output pins are pulled up, the port is pulled up if the value of the output latch is "1".

### ❍ Operation as an input port

- Pin values can be read by reading the PDR4 register.

### ❍ Operation during a reset

- If CPU is reset, the value of the PDR4 register is initialized to "1". Thus, all output transistors are turned "OFF" and the pins are put into in high impedance.

### ❍ Operation during resource output

- To use resources, set the output enable bit of each resource. Since the resource output takes precedence, settings of the PDR4 register corresponding to the resource output pins have no significance.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the output transistor is forced "OFF" and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.6-4 "Pin States of Port 4" lists the pin states of port 4.

**Table 4.6-4 Pin States of Port 4**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P40 to P43 | General-purpose I/O port/ resource I/O | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.7 Port 5

**Port 5 is a general-purpose I/O port which also serves as a resource output. Each pin can be used by switching between the resource and the port in units of bits. This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 5, its pins, a pin block diagram, and the related register.**

## ■ Configuration of Port 5

Port 5 is made up of the following three elements:

- General-purpose I/O pin/resource I/O pin (P50/ALR1 to P56/OFB3)
- Port 5 data register (PDR5)
- Port 5 direction register (DDR5)

## ■ Pins of Port 5

Port 5 has seven CMOS I/O pins.

Of these pins, if resources are used by the pins which also serve as resources, do not use them as a general-purpose I/O port.

Table 4.7-1 "Pins of Port 5" lists the pins of port 5.

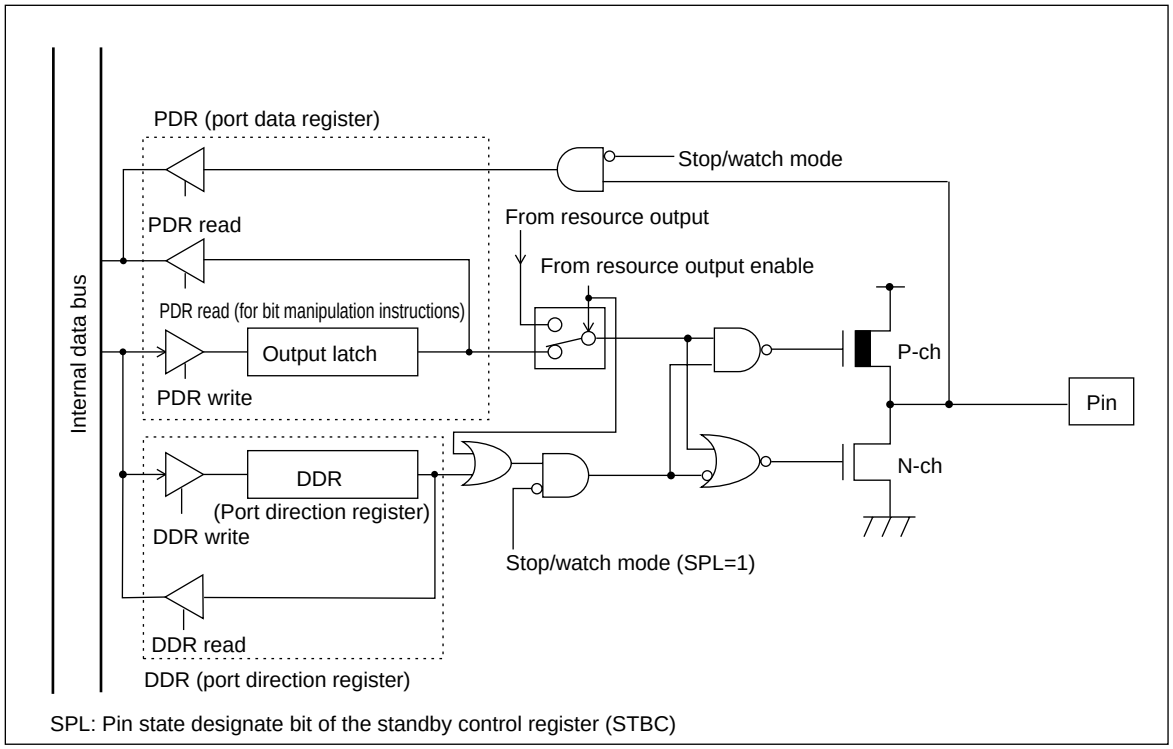
**Table 4.7-1 Pins of Port 5**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 5 | P50/ALR1 | P50 General-purpose I/O | ALR1 Comparator output | CMOS | CMOS | B |
| | P51/ALR2 | P51 General-purpose I/O | ALR2 Comparator output | | | |
| | P52/ALR3 | P52 General-purpose I/O | ALR3 Comparator output | | | |
| | P53/AC0 | P53 General-purpose I/O | AC0 Comparator output | | | |
| | P54/OFB1 | P54 General-purpose I/O | OFB1 Comparator output | | | |
| | P55/OFB2 | P55 General-purpose I/O | OFB2 Comparator output | | | |
| | P56/OFB3 | P56 General-purpose I/O | OFB3 Comparator output | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 5

**Figure 4.7-1 Block diagram of pins of port 5**




## ■ Registers of Port 5

Two registers PDR5 and DDR5 are available as the registers related to port 5.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 5.

Table 4.7-2 "Correspondence between the Register and Pins of Port 5" lists the correspondences between the registers and pins of port 5.

**Table 4.7-2 Correspondence between the Register and Pins of Port 5**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 5 | PDR5, DDR5 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | – | P56 | P55 | P54 | P53 | P52 | P51 | P50 |

# 4.7.1 Registers of the Port 5 (PDR5, DDR5)

**This section describes the registers related to port 5.**

## ■ Functions of the Registers of Port 5

### ❍ Port 5 data register (PDR5)

The PDR5 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins set as input ports, values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 5 direction register (DDR5)

The DDR5 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port

### ❍ Settings for resource output

To use a resource, set the operation enable bit of the resource.

Since the resource output takes precedence, settings of the PDR5 and DDR5 registers corresponding to the resource output pins have no significance regardless of the output value and output permission of the resource.

Table 4.7-3 "Register Functions of Port 5" lists the register functions of port 5.

**Table 4.7-3 Register Functions of Port 5**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 5 data register (PDR5) | 0 | Pin state is "L" | "0" is set to the output latch. If port 5 operates as an output port, the "L" level is output to the pins. | R/W | $0022_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 5 operates as an output port, the "H" level is output to the pins. | | | |
| Port 5 direction register (DDR5) | 0 | Input port state | Output transistor operation is prohibited and a pin is made an input pin. | R/W | $0023_H$ | $X0000000_B$ |
| | 1 | Output port state | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
X: Undefined

# 4.7.2 Operation of Port 5

**This section describes the operations of port 5.**

## ■ Operation of Port 5

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR5 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 5 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR5 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR5 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR5 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 5 operates as an input port.
- If data is written into the PDR5 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR5 register.

### ❍ Operation during resource output

- If the operation enable bit of a resource is set, the corresponding pin becomes ready for resource output. Since the resource output takes precedence, settings of the DDR5 register corresponding to the resource output pins have no significance.
- Because the pin values can be read through the PDR5 register even when resource output is allowed, output values of the resource can be read.

### ❍ Operation during a reset

- If CPU is reset, the bit values of the DDR5 register are initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The bits of the PDR5 register are not initialized by a reset. Thus, if port 5 is used as an output port, it is necessary to set output data to the PDR5 register and then set output to the corresponding DDR5 register.

### ❍ Operation in stop mode and watch mode

If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the pins are put into high impedance because the output transistor is forced "OFF" regardless of the value of the DDR5 register. The input is fixed to prevent leakage due to input opening.

Table 4.7-4 "Pin States of Port 5" lists the pin states of port 5.

**Table 4.7-4 Pin States of Port 5**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P50 to P56 | General-purpose I/O port/<br>resource I/O | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.8 Port 6

**Port 6 is a general-purpose I/O port which also serves for LCD controller/driver segment output. The I/O port and LCD controller/driver segment output can be selected by the register setting. This section describes with a particular emphasis on the functions of the general-purpose I/O port.**
**The following shows the configuration of port 6, its pins, a pin block diagram, and the related register.**

## ■ Configuration of Port 6

Port 6 is made up of the following two elements:

❍ **Port 6**

- General-purpose I/O pin/resource output (P60/SEG08 to P65/SEG13/U01)
- Port 6 data register (PDR6)

## ■ Pins of Port 6

Port 6 has six N-ch open-drain output I/O pins.

If the resource output pins are selected, do not use them as a general-purpose port.

Table 4.8-1 "Pins of Port 6" lists the pins of port 6.

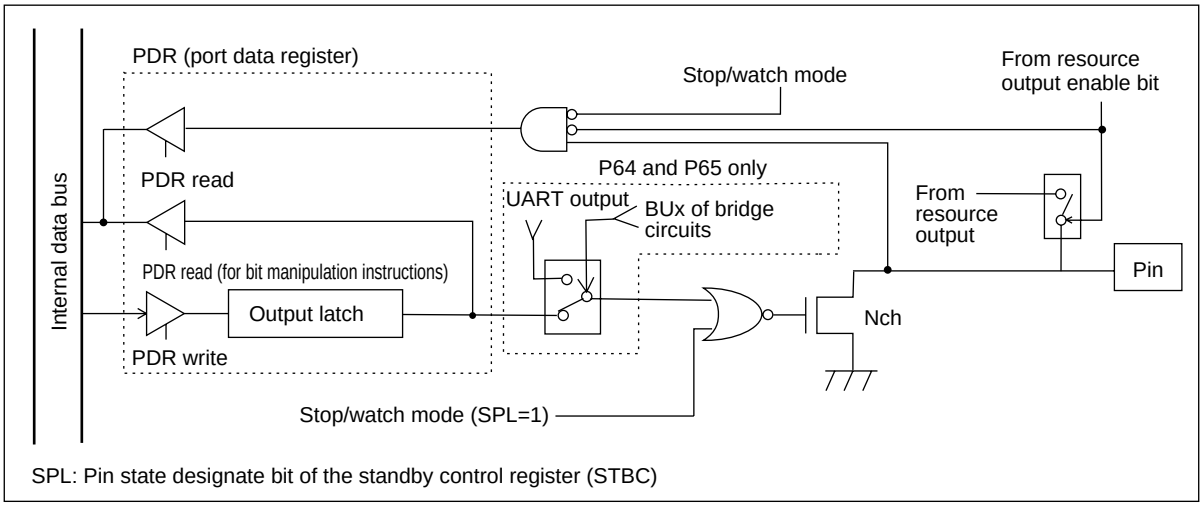
**Table 4.8-1 Pins of Port 6**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 6 | P60/SEG08 | P60 general-purpose I/O | SEG08 LCD controller/driver segment output | CMOS | N-ch open-drain | J |
| | to | to | to | | | |
| | P63/SEG11 | P63 general-purpose I/O | SEG11 LCD controller/driver segment output | | | |
| | P64/SEG12/U02 | P64 general-purpose I/O | SEG12 LCD controller/driver segment output/U02 output | | | |
| | P65/SEG13/U01 | P65 general-purpose I/O | SEG13 LCD controller/driver segment output/U01 output | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 6

**Figure 4.8-1 Block diagram of the pins of port 6**




**Note:**

Do not set PDR=0 for pins to be used as LCD controller/driver segment.

## ■ Register of Port 6

One register PDR6 is available as the register related to port 6.

There is a 1:1 correspondence between the bits configuring the register PDR6 and the pins of port 6.

Table 4.8-2 "Correspondence between the Register and Pins of Port 6" lists the correspondences between the register and pins of port 6.

**Table 4.8-2 Correspondence between the Register and Pins of Port 6**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 6 | PDR6 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | – | – | P65 | P64 | P63 | P62 | P61 | P60 |

# 4.8.1 Register the Port 6 (PDR6)

**This section describes the register related to port 6.**

## ■ Functions of the Register of Port 6

### ❍ Port 6 data register (PDR6)

The PDR6 register sets the states of pins.

Since the values of the output latch instead of the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Settings for resource output

To use port with the setting for resource output, set "1" to PDR of the pin to be used so that the resource output is not affected.

Table 4.8-3 "Register Functions of Port 6" lists the register functions of port 6.

**Table 4.8-3 Register Functions of Port 6**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 6 data register (PDR6) | 0 | Pin state is "L" | "0" is set to the output latch and the "L" level is output to the pins. | R/W | $0024_H$ | $XX111111_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch and the "Hi-Z" level is output to the pins. | | | |

R/W: Read/write enabled
X: Undefined
Hi-Z: High impedance

## ■ Registers Related to Port 6

### ❍ LCD controller/driver control register 2 (LCR2) $005F_H$

To use the port 6 for LCD controller/driver output, set "1" to the corresponding bit of the LCR2 register.

**Note:**

To use the port 6 as a port, it is necessary to set "0" to the selection bit of the LCR2 register.

### ❍ Bridge circuit select register 3 (BRSR3) $0019_H$

To use P64 and P65 for UART output, enable the serial data output (SMC2: TXOE=1) and then select the UART function through the BRSR3 register.

# 4.8.2 Operation of port 6

**This section describes the operations of port 5.**

## ■ Operation of Port 6

### ❍ Operation as an output port

- If data is written into the PDR6 register, the data is retained on the output latch and then output directly to the pins.
- When using port 6 as an output port, it cannot be used for LCD controller/driver segment output.

### ❍ Operation during LCD controller/driver segment output

- Set "1" to the bit of the PDR6 register corresponding to the LCD controller/driver segment output pin to put the output transistor into high impedance (Prohibit UART output for P64 and P65).

### ❍ Operation as an input port

- Pin values can be read by reading the PDR6 register (when the LCD controller/driver segment output is not selected).

### ❍ Operation during a reset

- If CPU is reset, the value of the PDR6 register is initialized to "1". Thus, all output transistors are turned "OFF" (input port) and the pins are put into high impedance.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.8-4 "Pin States of Port 6" lists the pin states of port 6.

**Table 4.8-4 Pin States of Port 6**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P60 to P65 | General-purpose I/O port/LCD controller/driver segment output/ UART output | Hi-Z(*1) | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance
*1: The previous state is retained during LCD controller/driver segment output.

# 4.9 Port 7

**Port 7 is a general-purpose I/O port which also serves as a resource input.**
**This section describes with a particular emphasis on the functions of the general-purpose I/O port.**
**The following shows the configuration of port 7, its pins, a pin block diagram, and the related register.**

## ■ Configuration of Port 7

Port 7 is made up of the following three elements:

❍ **Port 7**

- General-purpose I/O pin/resource output (P70/DCIN to P76/VSI3)
- Port 7 data register (PDR7)
- Port 7 direction register (DDR7)

## ■ Pins of Port 7

Port 7 has eight CMOS I/O pins.

If these pins are used as a comparator input pin, do not use them as a general-purpose I/O port.

Table 4.9-1 "Pins of Port 7" lists the pins of port 7.

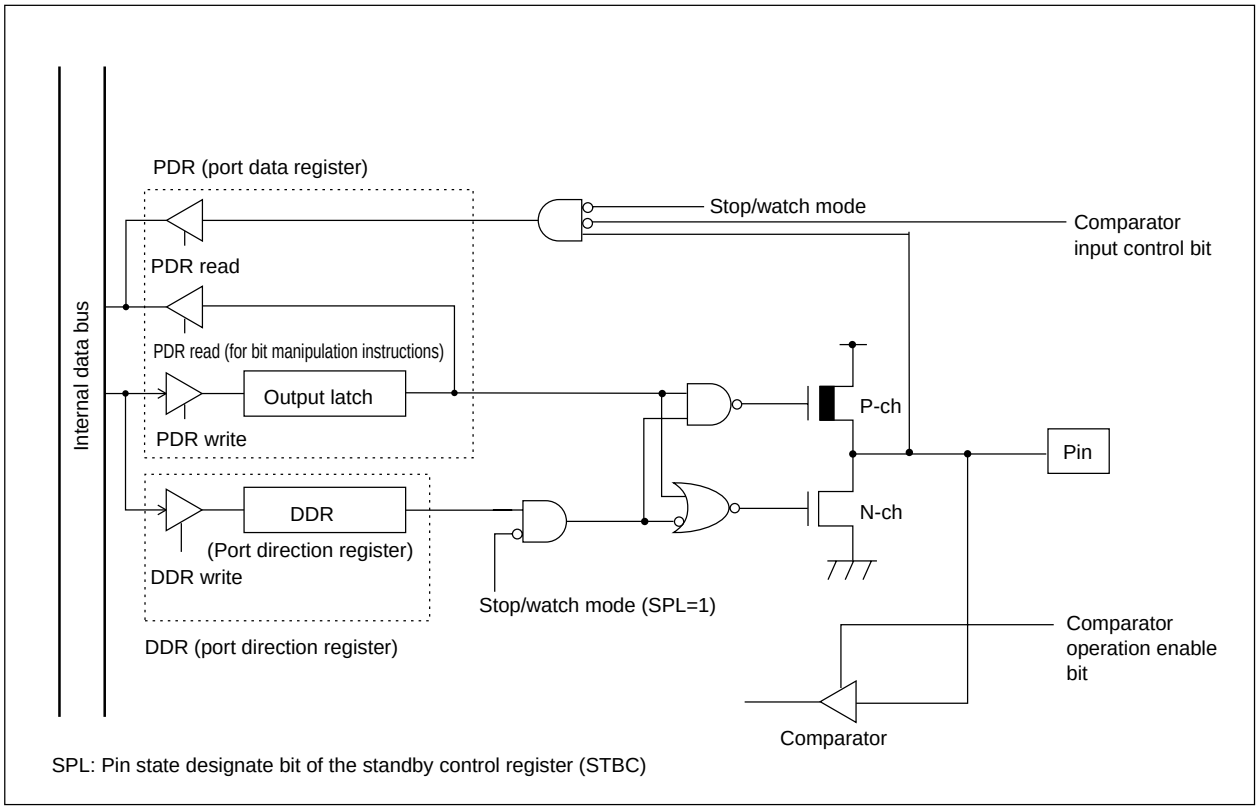
**Table 4.9-1 Pins of Port 7**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 7 | P70/ DCIN | P70 general-purpose I/O | Comparator input | Comparator/ CMOS | CMOS | N |
| | P71/ DCIN2 | P71 general-purpose I/O | Comparator input | | | |
| | P72/ VOL1 | P72 general-purpose I/O | Comparator input | | | |
| | P73/VSI1 | P73 general-purpose I/O | Comparator input | | | |
| | P74/ VOL2 | P74 general-purpose I/O | Comparator input | | | |
| | P75/VSI2 | P75 general-purpose I/O | Comparator input | | | |
| | P76/ VOL3 | P76 general-purpose I/O | Comparator input | | | |
| | P77/VSI3 | P77 general-purpose I/O | Comparator input | | | |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port 7

**Figure 4.9-1 Block diagram of pins of port 7**




**Note:**

Do not use the pins to be used as a comparator input pin as a general-purpose port.

## ■ Registers PDR7 and DDR7 of Port 7

Two registers PDR7 and DDR7 are available as the registers related to port 7.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 7.

Table 4.9-2 "Correspondence between the Registers and Pins of Port 7" lists the correspondence between the registers and pins of port 7.

**Table 4.9-2 Correspondence between the Registers and Pins of Port 7**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 7 | PDR7, DDR7 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | P77 | P76 | P75 | P74 | P73 | P72 | P71 | P70 |

# 4.9.1 Registers of Port 7 (PDR7, DDR7)

**This section describes the registers related to port 7.**

## ■ Functions of the Registers of Port 7

### ❍ Port 7 data register (PDR7)

The PDR7 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins are set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 7 direction register (DDR7)

The DDR7 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port.

### ❍ Settings for resource input

To use port 7 for resource input, set "0" to the bit of the DDR7 register corresponding to the resource input pin and set "1" to the bit corresponding to the CIER register.

Table 4.9-3 "Register Functions of Port 7" lists the register functions of port 7.

**Table 4.9-3 Register Functions of Port 7**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 7 data register (PDR7) | 0 | Pin state is "L" | The "L" level is output to the pins if port 7 operates as an output port. | R/W | $0025_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | The "H" level is output to the pins if port 7 operates as an output port. | | | |
| Port 7 direction register (DDR7) | 0 | Input port state | Output transistor operation is prohibited and a pin is made an input pin. | R/W | $0026_H$ | $00000000_B$ |
| | 1 | Output port state | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
X: Undefined

## ■ Register Related to Port 7

### ❍ Comparator input enable register (CIER) 0059H

To use port 7 for comparator input and to help prevent the DC pass when an intermediate level is entered, set "1" to the corresponding bit of the CIER3 register.

**Note:**

To use port 7 for port input, "0" must be set to the input enable bit of the COSR3 register.

# 4.9.2 Operation of Port 7

**This section describes the operations of port 7.**

## ■ Operation of Port 7

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR7 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 7 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR7 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR7 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR7 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 7 operates as an input port.
- If data is written into the PDR7 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR7 register.

### ❍ Operation during comparator input

- Set the operation enable bit of the comparator to use port 7 for comparator input.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR7 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR7 register is not initialized by a reset. Thus, to use port 7 as an output port, data must be output to the PDR7 register and the output must be set to the corresponding DDR7 register.
- The CIER register is initialized to "1" by a reset. Thus, to use port 7 for port input, "0" must be set to the corresponding bit of the CIER register.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, prohibition of the port input occurs regardless of the value of the DDR7 register and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.9-4 "Pin States of Port 7" lists the pin states of port 7.

**Table 4.9-4 Pin States of Port 7**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P70 to P77 | General-purpose I/O port/<br>comparator input | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.10 Port 8

**The port 8 is a general-purpose I/O port which also serves as a resource input. Each pin can be used by switching between the resource and the port in units of bits. This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port 8, its pins, pin block diagrams, and the related register.**

## ■ Configuration of Port 8

Port 8 is made up of the following three elements:

- General-purpose I/O pin/resource input pin (P80/INT0 to P87/AN2/SW3)
- Port 8 data register (PDR8)
- Port 8 direction register (DDR8)

## ■ Pins of Port 8

Port 8 has eight CMOS I/O pins.

Of these pins, if resources are used by the pins which also serve as resources, do not use them as a general-purpose I/O port.

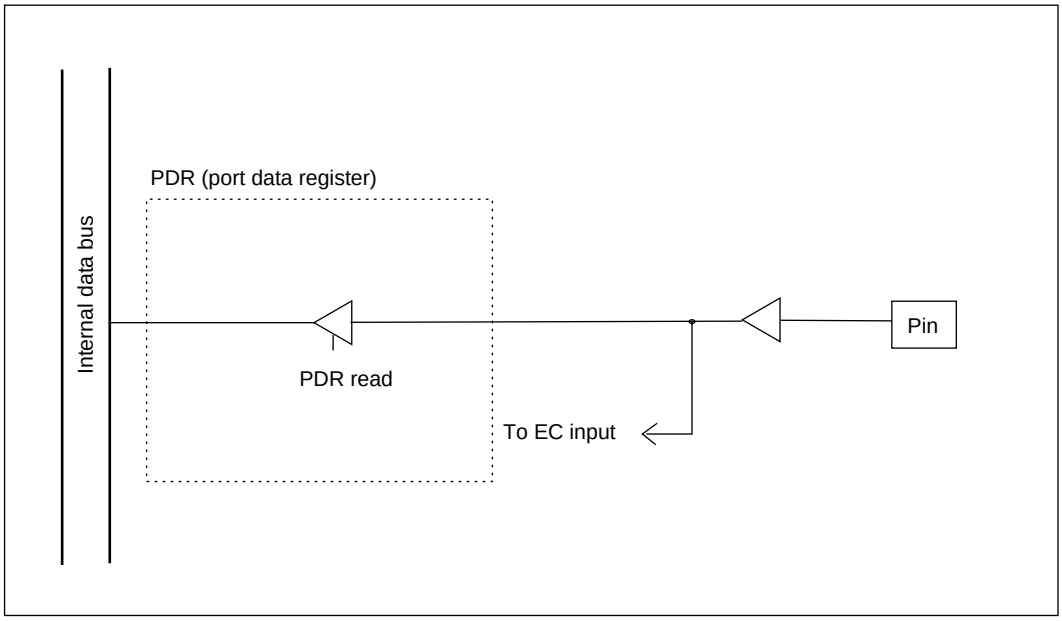
Table 4.10-1 "ins of Port 8" lists the pins of port 8.

**Table 4.10-1 Pins of Port 8**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 8 | P80/INT0 | P80 general-purpose I/O | INT0 external interrupt | CMOS(*1) | CMOS | K |
| | P81/INT1 | P81 general-purpose I/O | INT1 external interrupt | | | |
| | P82/INT2 | P82 general-purpose I/O | INT2 external interrupt | | | |
| | P83/INT3 | P83 general-purpose I/O | INT3 external interrupt | | | |
| | P84/ | P84 general-purpose input | – | CMOS | – | O |
| | P85/AN0/SW1 | P85 general-purpose I/O | AN0 analog input<br>SW1 comparator input | | CMOS | L |
| | P86/AN1/SW2 | P86 general-purpose I/O | AN1 analog input<br>SW2 comparator input | | | |
| | P87/AN2/SW3 | P87 general-purpose I/O | AN2 analog input<br>SW3 comparator input | | | |

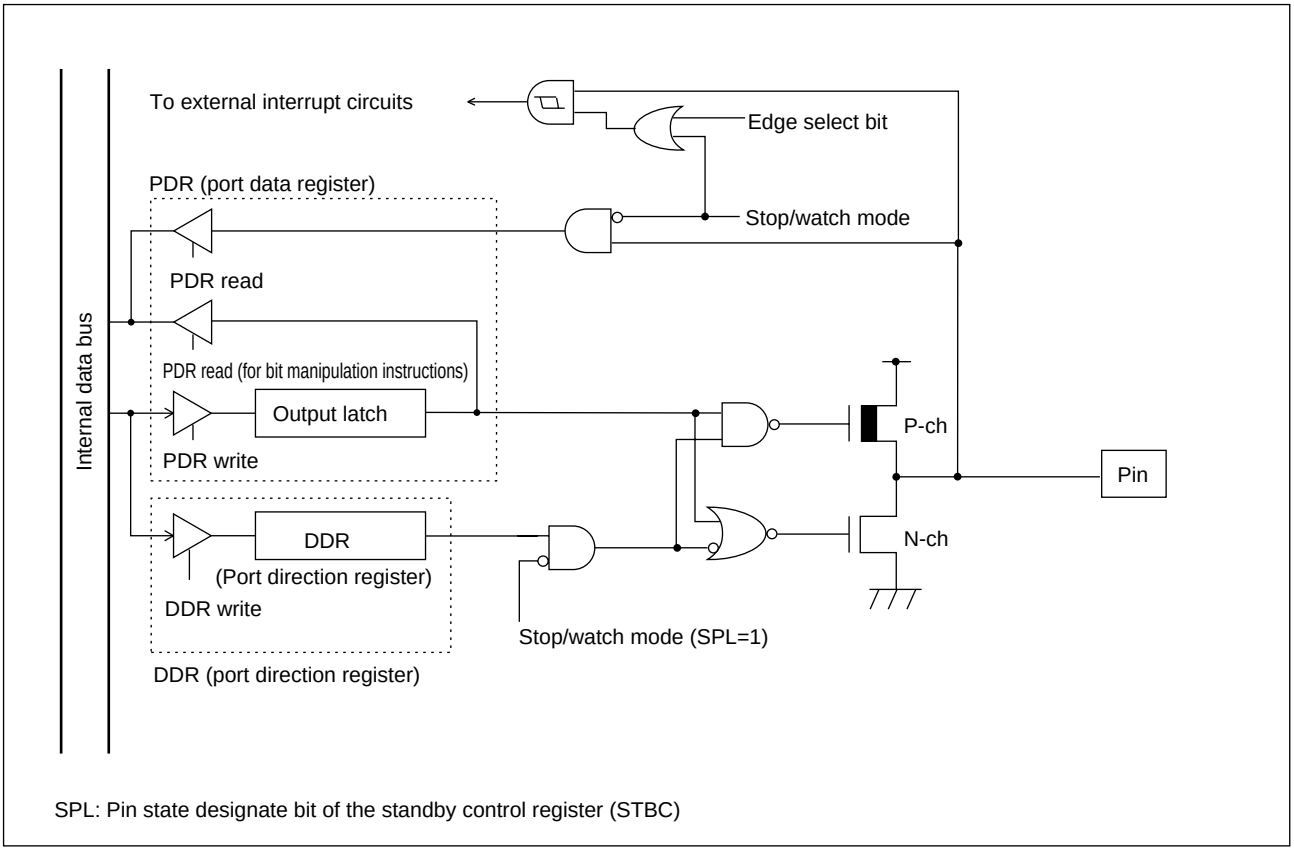
*1: The resource is hysteresis. For the circuit type, see Section 1.7 "Pin description".

**Figure 4.10-1 Block Diagram of Pins of Port 8 (P84)**



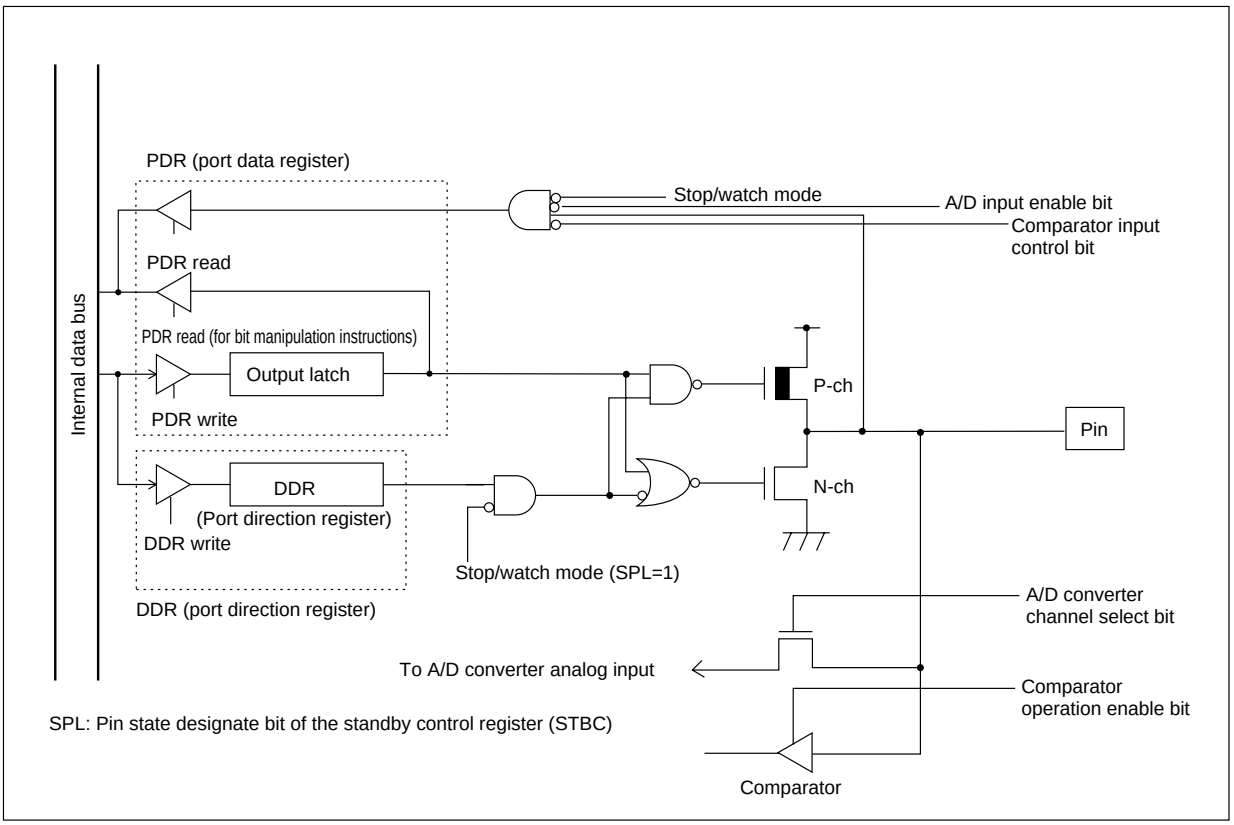

## ■ Block Diagram of Port 8

**Figure 4.10-2 Block Diagram of Pins of Port 8 (P80 to P83)**




SPL: Pin state designate bit of the standby control register (STBC)

**Figure 4.10-3 Block Diagram of Pins of Port 8 (P85 to P87)**




## ■ Registers of Port 8

Two registers PDR8 and DDR8 are available as the registers related to port 8.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 8.

Table 4.10-2 "Correspondence between the Registers and Pins of Port 8" lists the correspondences between the registers and pins of port 8.

**Table 4.10-2 Correspondence between the Registers and Pins of Port 8**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 8 | PDR8, DDR8 | bit 7 | bit 6 | bit 5 | bit 4(*1) | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | P87 | P86 | P85 | P84 | P83 | P82 | P81 | P80 |

*1: bit4 of DDR8 does not correspond to P84.

# 4.10.1 Registers of Port 8 (PDR8, DDR8)

**This section describes the registers related to port 8.**

## ■ Functions of the Registers of Port 8

### ❍ Port 8 data register (PDR8)

The PDR8 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins are set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 8 direction register (DDR8)

The DDR8 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port

Table 4.10-3 "Register Functions of Port 8" lists the register functions of port 8.

**Table 4.10-3 Register Functions of Port 8**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 8 data register (PDR8) | 0 | Pin state is "L" | "0" is set to the output latch. If port 8 operates as an output port, the "L" level is output to the pins. | R/W | $0027_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 8 operates as an output port, the "H" level is output to the pins. | | | |
| Port 8 direction register (DDR8) | 0 | Input port state | Output transistor operation is prohibited and a pin is made an input pin. | R/W | $0028_H$ | $000X0000_B$ |
| | 1 | Output port state | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
X: Undefined

## ■ Registers Related to Port 8

### ❍ A/D port input enable register (ADEN1) $002D_H$

To use port 8 for analog input, set "1" to the corresponding bit of the ADEN1 register. This can help prevent the DC pass when an intermediate level is entered.

### ❍ Comparator input enable register (CIER) 0059H

To use port 8 for comparator input and to help prevent the DC pass when an intermediate level is entered, set "1" to the corresponding bit of the CIER register.

**Note:**

To use port 8 for port input, "0" must be set to the input enable bit of the ADEN1/CIER registers.

# 4.10.2 Operation of Port 8

**This section describes the operations of port 8.**

## ■ Operation of Port 8

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR8 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 8 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR8 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR8 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR8 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 8 operates as an input port.
- If data is written into the PDR8 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR8 register.

### ❍ Operation during resource I/O

- Set "0" to the DDR8 register corresponding to the resource input pin to use port 8 for resource input.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR8 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR8 register is not initialized by a reset. Thus, to use port 8 as an output port, output data must be set to the PDR8 register and the output must be set to the corresponding DDR8 register.
- The ADEN1/CIER registers are initialized to "1" by a reset. Thus, to use port 8 for port input, "0" must be set to the corresponding bits of the ADEN1/CIER registers.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the pins are put into high impedance because the output transistor is forced "OFF" regardless of the value of the DDR8 register. The input other than that of P84 is fixed to prevent leakage due to input opening.

Table 4.10-4 "Pin States of Port 8" lists the pin states of port 8.

**Table 4.10-4 Pin States of Port 8**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P80 to P87 | General-purpose I/O port/<br>resource I/O | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.11 Port 9

**Port 9 is a general-purpose I/O port which also serves for resource I/O.
This section describes the functions of the general-purpose I/O port.
The following shows the configuration of port 9, its pins, pin block diagrams, and the related register.**

## ■ Configuration of Port 9

Port 9 is made up of the following three elements:

❍ **Port 9**

- General-purpose I/O pin/analog input pin (P90/AN3, P91/DA1, P92/DA2)
- Port 9 data register (PDR9)
- Port 9 direction register (DDR9)

## ■ Pins of Port 9

Port 9 has three CMOS I/O pins.

Do not use these pins as a general-purpose port when resources are used.

Table 4.11-1 "Pins of Port 9" lists the pins of port 9.

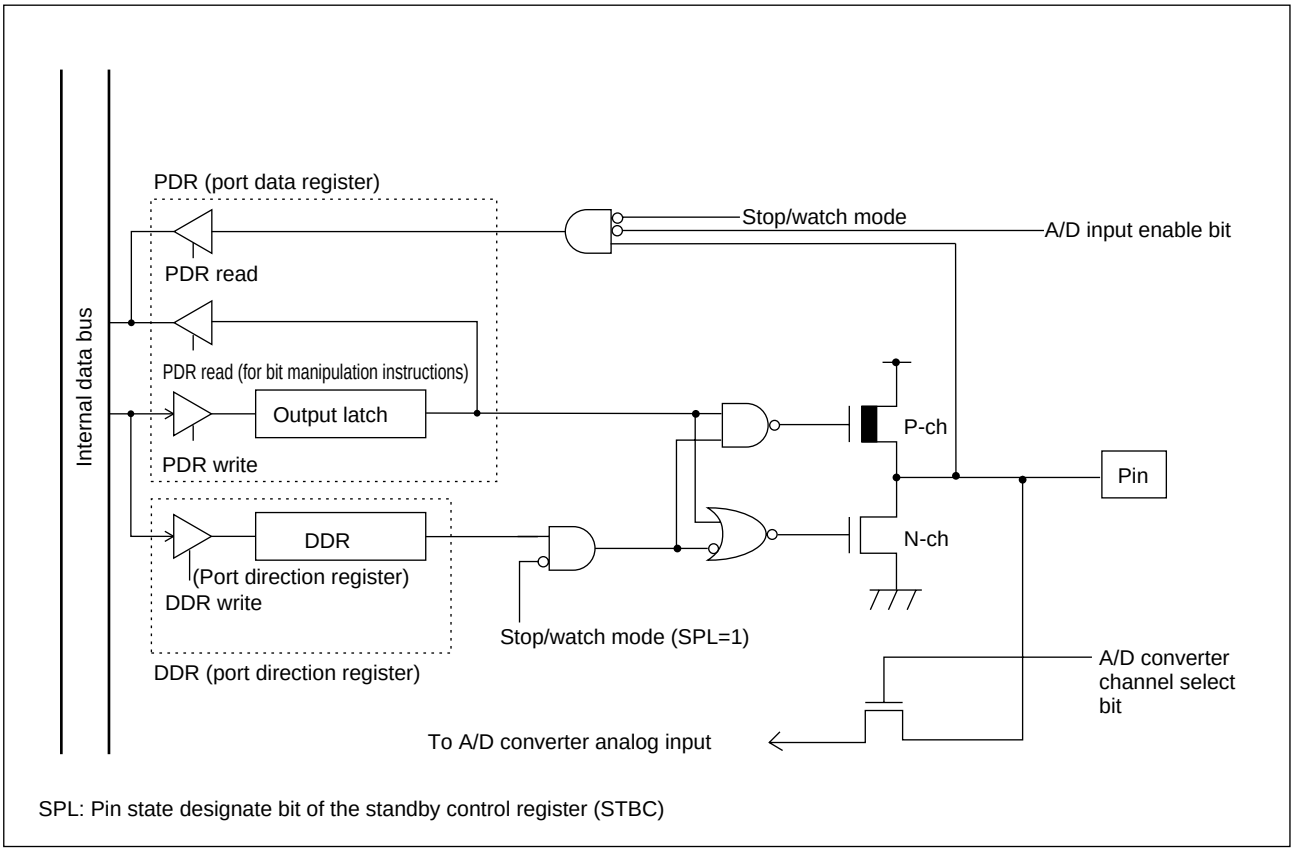
**Table 4.11-1 Pins of Port 9**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port 9 | P90/AN3 | P90 general-purpose I/O | Analog input | Analog/ CMOS | CMOS | E |
| | P91/DA1 | P91 general-purpose I/O | D/A converter output | CMOS | CMOS/ DA | M |
| | P92/DA2 | P92 general-purpose I/O | D/A converter output | | | |

For the circuit type, see Section 1.7 "Pin description".

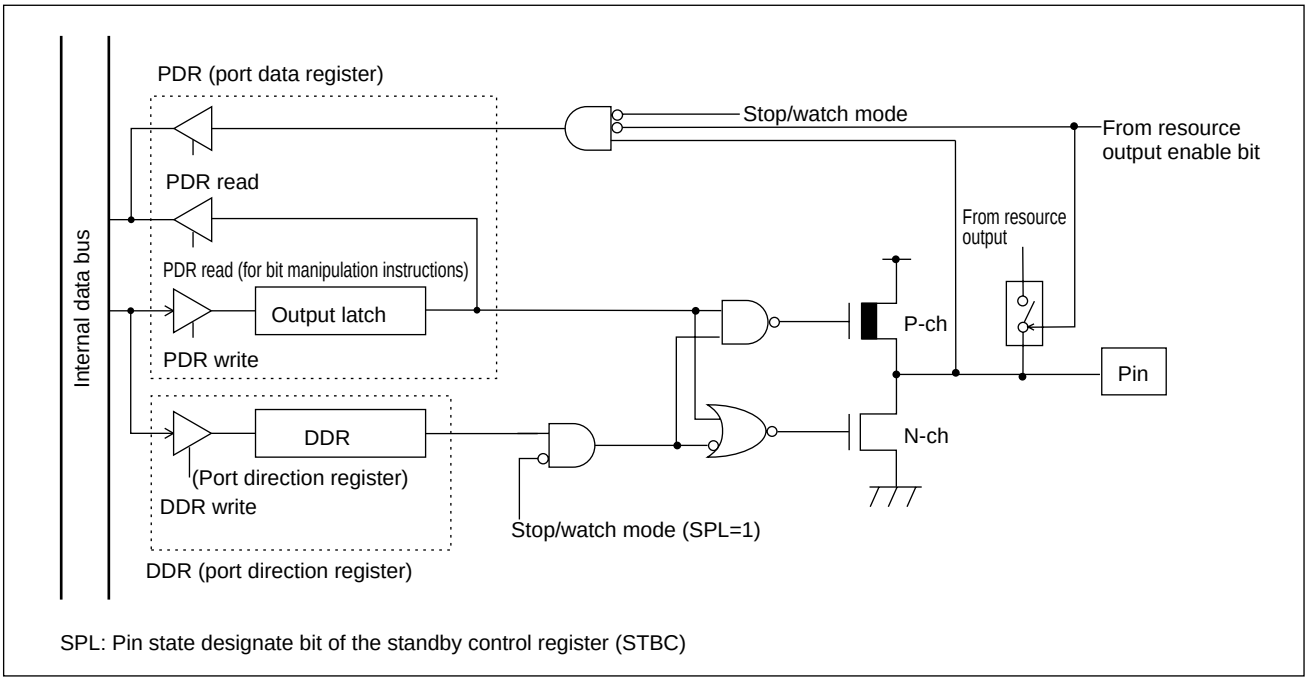
## ■ Block Diagram of Port 9

**Figure 4.11-1 Block Diagram of Pins of Port 9 (P90)**




SPL: Pin state designate bit of the standby control register (STBC)

**Figure 4.11-2 Block Diagram of Pins of Port 9 (P91, P92)**




**Note:**

Do not use the pins to be used as an analog input pin as a general-purpose port.

## ■ Registers PDR9 and DDR9 of Port 9

Two registers PDR9 and DDR9 are available as the registers related to port 9.

There is a 1:1 correspondence between the bits configuring each register and the pins of port 9.

Table 4.11-2 "Correspondence between the Registers and Pins of Port 9" lists the correspondence between the registers and pins of port 9.

**Table 4.11-2 Correspondence between the Registers and Pins of Port 9**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port 9 | PDR9, DDR9 | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | – | – | – | – | – | P92 | P91 | P90 |

# 4.11.1 Registers of Port 9 (PDR9, DDR9)

**This section describes the registers related to port 9.**

## ■ Functions of the Registers of Port 9

### ❍ Port 9 data register (PDR9)

The PDR9 register indicates the states of pins. Thus, if the pins are set as output ports, the same values ("0" or "1") can be read as those of the output latch. However, if the pins are set as input ports, the values of the output latch cannot be read.

Since the values of the output latch rather than the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Port 9 direction register (DDR9)

The DDR9 register sets the I/O direction of pins for each bit.

If "1" is set to the bit corresponding to a port, the port becomes an output port. If "0" is set to the bit corresponding to a port, the port becomes an input port

Table 4.11-3 "Register Functions of Port 9" lists the register functions of port 9.

**Table 4.11-3 Register Functions of Port 9**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port 9 data register (PDR9) | 0 | Pin state is "L" | "0" is set to the output latch. If port 9 operates as an output port, the "L" level is output to the pins. | R/W | $0029_H$ | $XXXXXXXX_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch. If port 9 operates as an output port, the "H" level is output to the pins. | | | |
| Port 9 direction register (DDR9) | 0 | Input port state | Output transistor operation is prohibited and a pin is made an input pin. | R/W | $002A_H$ | $XXXXX000_B$ |
| | 1 | Output port state | Output transistor operation is allowed and a pin is made an output pin. | | | |

R/W: Read/write enabled
X: Undefined

## ■ Register Related to Port 9

### ❍ A/D port input enable register (ADEN1) $002D_H$

To use port 9 for analog input, set "1" to the corresponding bit of the ADEN1 register. This can also serve to prevent the DC pass when an intermediate level is entered.

**Note:**

To use port 9 for port input, "0" must be set to the input enable bit of the ADEN1 registers.

# 4.11.2 Operation of Port 9

**This section describes the operations of port 9.**

## ■ Operation of Port 9

### ❍ Operation as an output port

- If "1" is set to the corresponding DDR9 register bit, the port becomes an output port.
- The operation of the output transistor is allowed when port 9 operates as an output port and data of the output latch is output to the pins.
- If data is written into the PDR9 register, the data is retained on the output latch and then output directly to the pins.
- Pin values can be read by reading the PDR9 register.

### ❍ Operation as an input port

- If "0" is set to the corresponding DDR9 register bit, the port becomes an input port.
- The output transistor is "OFF" and the pins are in high impedance when port 9 operates as an input port.
- If data is written into the PDR9 register, the data is retained on the output latch but is not output to the pins.
- Pin values can be read by reading the PDR9 register.

### ❍ Operation during resource I/O

- To use port 9 for analog input, set "1" to the corresponding bit of the DDR9 register and ADEN1 register corresponding to the analog input pin.
- Since the D/A converter output takes precedence if the D/A converter output is allowed, settings of the corresponding DDR9 and PDR9 have no significance.

### ❍ Operation during a reset

- If CPU is reset, the value of the DDR9 register is initialized to "0". Thus, the output transistor is turned "OFF" (input port) and the pins are put into high impedance.
- The PDR9 register is not initialized by a reset. Thus, to use port 9 as an output port, output data must be set to the PDR9 register and the output must be set to the corresponding DDR9 register.
- The ADEN1 register is initialized to "1" by a reset. Thus, to use port 9 for port input, "0" must be set to the corresponding bit of the ADEN1 register.

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, prohibition of the port input occurs regardless of the value of the DDR9 register and the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.11-4 "Pin States of Port 9" lists the pin states of port 9.

**Table 4.11-4 Pin States of Port 9**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| P90 | General-purpose I/O port/analog input | Hi-Z | Hi-Z |
| P91 to P92 | General-purpose I/O port/D/A converter output | Hi-Z | Hi-Z |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance

# 4.12 Port A

**The port A is a general-purpose I/O port which also serves for LCD controller/driver segment output. The I/O port and LCD controller/driver segment output can be selected by the register setting This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of the port A, its pins, a pin block diagram, and the related register.**

## ■ Configuration of Port A

The port A is made up of the following two elements:

❍ Port A

- General-purpose I/O pin/resource output (PA0/SEG00 to PA7/SEG07)
- Port A data register (PDRA)

## ■ Pins of Port A

The port A has each eight N-ch open-drain I/O pins.

Do not use these pins as a general-purpose port when they are selected as a LCD controller/driver segment output pin.

Table 4.12-1 "Pins of Port A" lists the pins of port A.

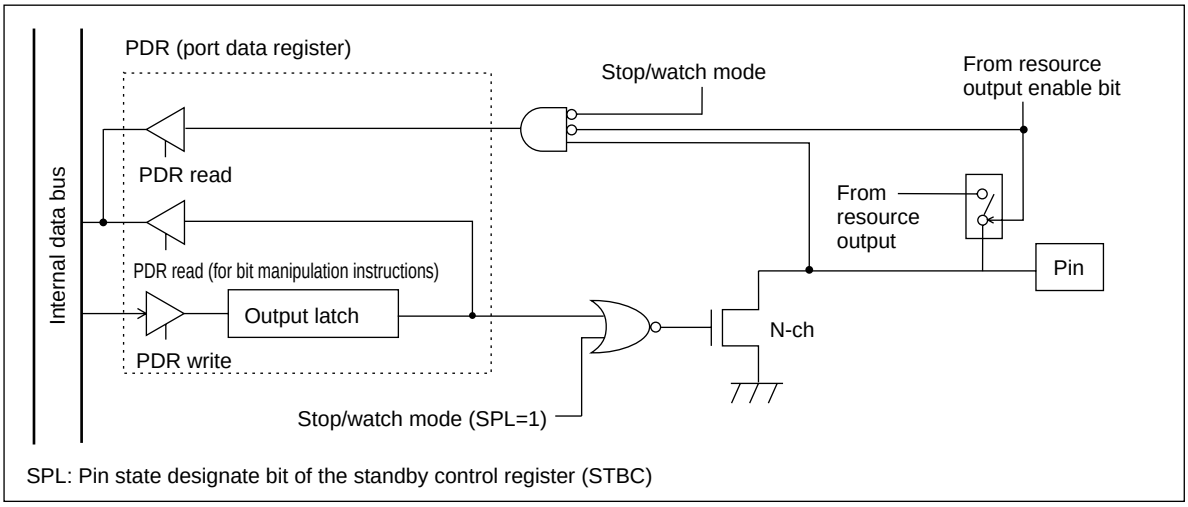
**Table 4.12-1 Pins of Port A**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port A | PA0/SEG00 to PA7/SEG07 | PA0 general-purpose I/O to PA7 general-purpose I/O | SEG00 LCDC segment output to SEG07 LCD segment output | CMOS | N-ch open-drain | J |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port A

**Figure 4.12-1 Block Diagram of Pins of Port A**




**Note:**

Do not set PDR=0 for the pins to be used as an LCD controller/driver segment.

## ■ Register of Port A

One register PDRA is available as the register related to port A.

There is a 1:1 correspondence between the bits configuring the register PDRA and the pins of port A.

Table 4.12-2 "Correspondence between the Register and Pins of Port A" lists the correspondence between the registers and pins of port A.

**Table 4.12-2 Correspondence between the Register and Pins of Port A**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port A | PDRA | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |

# 4.12.1 Register of Port A (PDRA)

**This section describes the register related to port A.**

## ■ Functions of the Register of Port A

### ❍ Port A data register (PDRA)

The PDRA register sets the states of pins.

Since the values of the output latch instead of the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Settings for LCD controller/driver segment output

To use the port with the setting for LCD controller/driver segment output, set "1" to PDR of the pin to be used so that the LCD controller/driver segment output is not affected.

Table 4.12-3 "Register Functions of Port A" lists the register functions of port A.

**Table 4.12-3 Register Functions of Port A**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port A data register (PDRA) | 0 | Pin state is "L" | "0" is set to the output latch and the "L" level is output to the pins. | R/W | $0015_H$ | $11111111_B$ |
| | 1 | Pin state is "H" | "1" is set to the output latch and the "Hi-Z" level is output to the pins. | | | |

R/W: Read/write enabled
Hi-Z: High impedance

## ■ Register Related to Port A

### ❍ LCD controller/driver control register 3 (LCR3) $0016_H$

To use port A for LCD controller/driver output, set "0" to the corresponding bit of the LCR3 register.

**Note:**

To use port A as a port, "1" must be set to the selection bit of the LCR3 register.

# 4.12.2 Operation of the Port A

**This section describes the operations of port A.**

## ■ Operation of Port A

### ❍ Operation as an output port

- If data is written into the PDRA register, the data is retained on the output latch and then output directly to the pins.
- When the port is used as an output port, it cannot be used for LCD controller/driver segment output.

### ❍ Operation for LCD controller/driver segment output

- Set "1" to the bit of the PDRA register corresponding to the LCD controller/driver segment output pin to put the output transistor into high impedance.

### ❍ Operation as an input port

- Pin values can be read by reading the PDRA register (when the LCD controller/driver segment output is not selected).

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.12-4 "Pin States of Port A" lists the pin states of port A.

**Table 4.12-4 Pin States of Port A**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| PA0 to PA7 | General-purpose I/O port/LCD controller/driver segment output/ UART output | Hi-Z(*1) | "L" output |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance
*1: High impedance during LCD controller/driver segment output

# 4.13 Port B

**The port B is a general-purpose I/O port which also serves for LCD controller/driver I/O. The I/O port and LCD controller/driver I/O can be selected by the register setting This section describes the functions of the general-purpose I/O port.**
**The following shows the configuration of port A, its pins, pin block diagrams, and the related register.**

## ■ Configuration of Port B

Port B is made up of the following two elements:

❍ **Port B**

- General-purpose I/O pin/resource I/O (PB0/V0 to PB3/V3, PB4/COM0 to PB7/COM03)
- Port B data register (PDRB)

## ■ Pins of the Port B

Port B has each eight N-ch open-drain I/O pins.

Do not use these pins as a general-purpose port when they are selected as a LCD controller/driver I/O pin.

Table 4.13-1 "Pins of Port B" lists the pins of port B.

**Table 4.13-1 Pins of Port B**

| Port name | Pin name | Function | Shared resource | I/O type | | Circuit type |
|---|---|---|---|---|---|---|
| | | | | Input | Output | |
| Port B | PB0/V0 to PB3/V3 | PB0 general-purpose output to PB3 general-purpose output | V0 LCD controller/driver input to V3 LCD controller/driver power input | - | N-ch open-drain | I |
| | PB4/COM0 to PB7/COM03 | PB4 general-purpose I/O to PB7 general-purpose I/O | COM0 LCD controller/driver common output to COM0 LCD controller/driver common output | CMOS | N-ch open-drain | J |

For the circuit type, see Section 1.7 "Pin description".

## ■ Block Diagram of Port B

**Figure 4.13-1 Block Diagram of Pins of Port B (PB0 to PB3)**

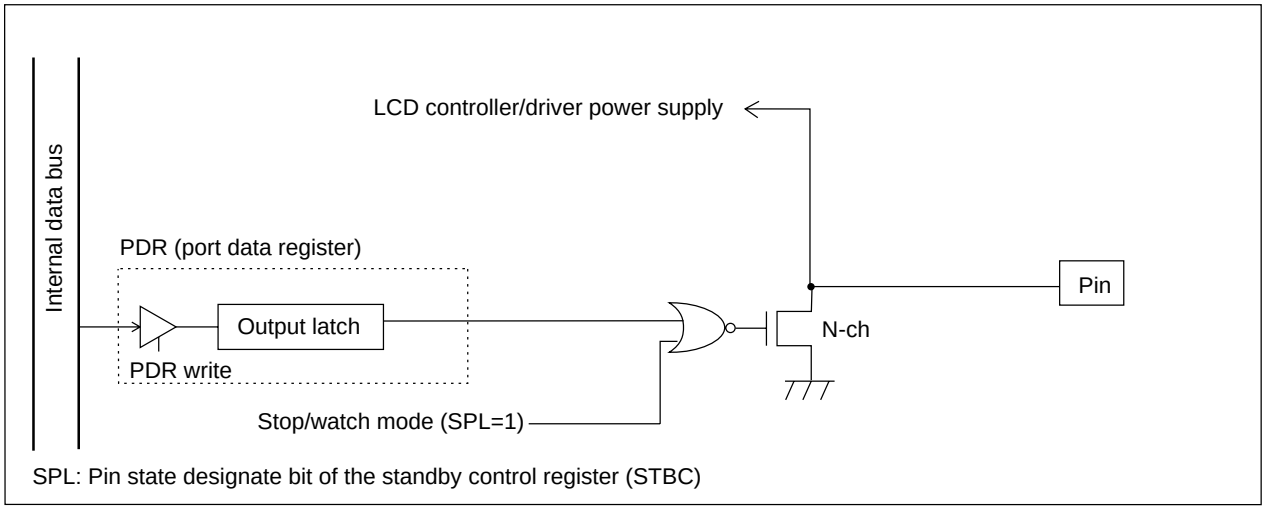
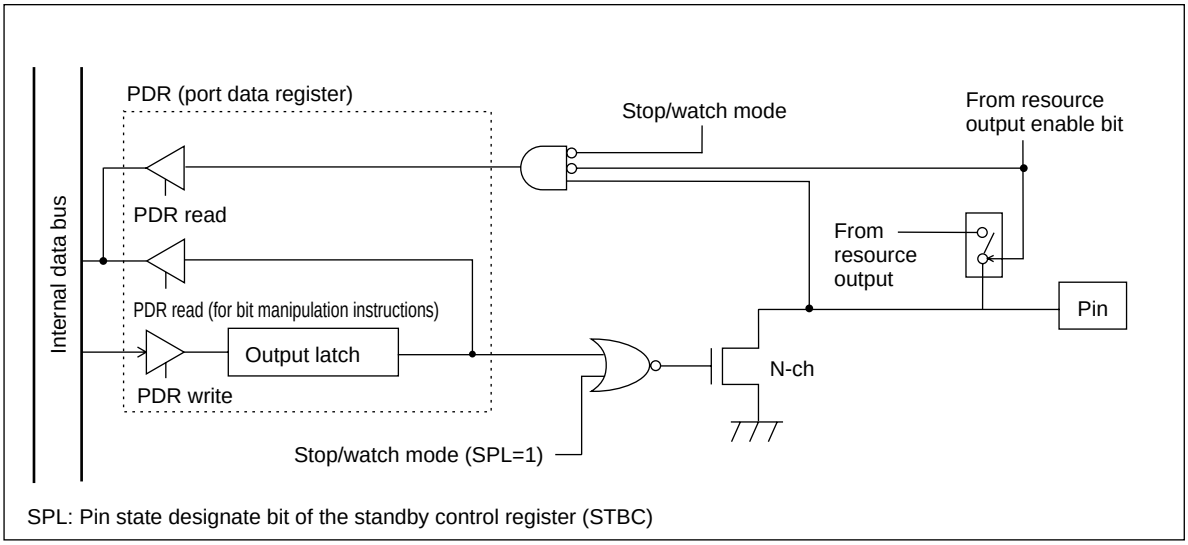



**Figure 4.13-2 Block Fiagram of Pins of Port B (PB4 to PB7)**




**Note:**

Do not set PDR=0 for the pins to be used as an LCD controller/driver segment.

## ■ Register of Port B

One register PDRB is available as the register related to port B.

There is a 1:1 correspondence between the bits configuring the register PDRB and the pins of port B.

Table 4.13-2 "Correspondence between the Register and Pins of Port B" lists the correspondences between the registers and pins of port B.

**Table 4.13-2 Correspondence between the Register and Pins of Port B**

| Port name | Bits of the related registers and corresponding pins | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Port B | PDRB | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| | Corresponding pin | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |

# 4.13.1 Register of Port B (PDRB)

**This section describes the register related to port B.**

## ■ Functions of the Register of Port B

### ❍ Port B data register (PDRB)

The PDRB register sets the states of pins.

Since the values of the output latch instead of the pins are read when a bit manipulation instruction (SETB, CLRB) is used, the values of the output latch whose bits are not manipulated do not change.

### ❍ Settings for LCD controller/driver I/O

To use the port with the setting for LCD controller/driver I/O, set "1" to PDR of the pin to be used so that the LCD controller/driver I/O is not affected.

Table 4.13-3 "Register Functions of Port B" lists the register functions of port B.

**Table 4.13-3 Register Functions of Port B**

| Register name | Data | Read | Write | Read/write | Address | Initial value |
|---|---|---|---|---|---|---|
| Port B data register (PDRB) | 0 | Pin state is "L" | "0" is set to the output latch and the "L" level is output to the pins. | R/W(*1) | $0017_H$ | $11111111_B$ |
| | 1 | Pin state is "H" (*1) | "1" is set to the output latch and the "Hi-Z" level is output to the pins. | | | |

R/W: Read/write enabled
Hi-Z: High impedance
*1: Only PB4 to PB7 are read enabled. Do not use instructions of the PWM set.
"0" can always be read from PB0 to PB3.

## ■ Register Related to Port B

### ❍ LCD controller/driver control register 4 (LCR4) $0018_H$

To use port B for LCD controller/driver I/O, set "0" to the corresponding bit of the LCR4 register.

**Note:**

To use port B as a port, it is necessary to set "0" to the selection bit of the LCR4 register.

# 4.13.2 Operation of Port B

**This section describes the operations of port B.**

## ■ Operation of Port B

### ❍ Operation as an output port

- If data is written into the PDRB register, the data is retained on the output latch and then output directly to the pins.
- When the port is used as an output port, it cannot be used for LCD controller/driver I/O.

### ❍ Operation for LCD controller/driver segment I/O

- Set "1" to the bit of the PDRB register corresponding to the LCD controller/driver segment I/O pin to put the output transistor into high impedance.

### ❍ Operation as an input port

- By reading the PDRB register, the values of only the PB4 to PB7 pins can be read (when the LCD controller/driver segment output is not selected).

### ❍ Operation in stop mode and watch mode

- If the pin state designate bit (STBC: SPL) of the standby control register is set to "1" when a transition to the stop mode or watch mode occurs, the pins are put into high impedance. The input is fixed to prevent leakage due to input opening.

Table 4.13-4 "Pin States of Port B" lists the pin states of port B.

**Table 4.13-4 Pin States of Port B**

| Pin name | Normal operation<br>Main sleep<br>Main stop (SPL=0)<br>Sub-sleep<br>Sub-stop (SPL=0)<br>Watch mode (SPL=0) | Main stop (SPL=1)<br>Sub-stop (SPL=1)<br>Watch mode (SPL=1) | During reset |
|---|---|---|---|
| PB0 to PB3 | General-purpose I/O port/LCD controller/driver input | Hi-Z(*1) | LCD controller/ driver input |
| PB4 to PB7 | General-purpose I/O port/LCD controller/driver I/O | Hi-Z(*1) | "L" output |

SPL: Pin state designate bit of the standby control register (STBC: SPL)
Hi-Z: High impedance
*1: High impedance cannot be specified during LCD controller/driver segment output

# 4.14 Program Example of the I/O Ports

**This section shows a program example using the I/O port.**

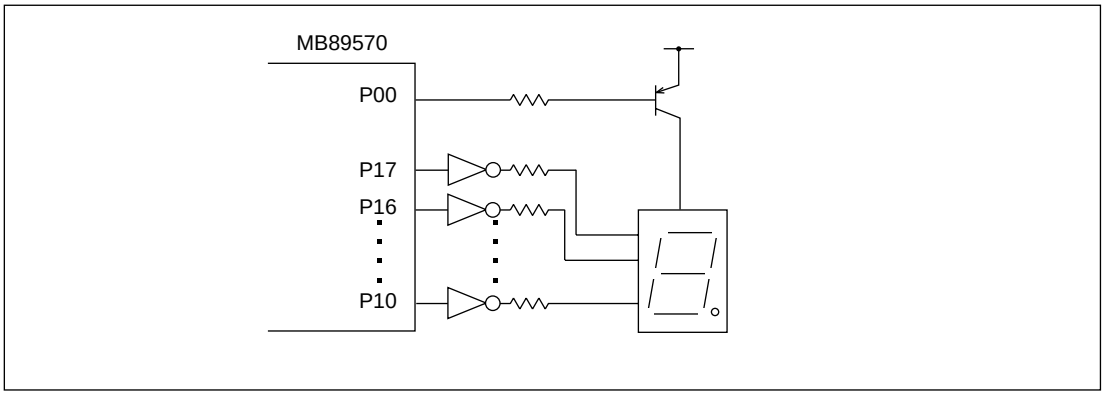
## ■ Program Example of the I/O Ports

### ❍ Processing specifications

- Turn on all LED of seven segments (eight segment if Dp is included) from the ports 0/1.
- The P00 pin corresponds to the anode common pin of LED, and the pins P10 to P17 pins correspond to each segment pin.

Figure 4.14-1 "Example of an 8-segment LED Connection" shows an example of an 8-segment LED connection.

**Figure 4.14-1 Example of an 8-segment LED Connection**




### ❍ Coding example

```
PDR0  .EQU  0000H    ; Address of the port 0 data register
DDR0  .EQU  0001H    ; Address of the port 0 direction register
PDR1  .EQU  0002H    ; Address of the port 1 data register
DDR1  .EQU  0003H    ; Address of the port 1 direction register

;----------Main program-----------------------------------------------
      .SECTION       CSEG, CODE, ALIGN = 1    ; [CODE SEGMENT]
         :
      CLRB  PDR0:0              ; Set P00 to the "L" level.
      MOV   PDR1, #11111111B    ; Set the port 1 all to the "H" level
      MOV   DDR0, #11111111B    ; Set P00 for output,
                                ; enabled by #XXXXXXX1B.
      MOV   DDR1, #11111111B    ; Set all bits of the port 1
                                  for output
         :
      ENDS
;----------------------------------------------------------------------
      .END
```

# CHAPTER 5 TIMEBASE TIMER

**This chapter describes the functions and operations of the timebase timer.**


5.1 "Overview of the Timebase Timer"

5.2 "Configuration of the Timebase Timer"

5.3 "Timebase Timer Control Register (TBTC)"

5.4 "Timebase Timer Interrupt"

5.5 "Operation of the Timebase Timer"

5.6 "Notes on Using the Timebase Timer"

5.7 "Program Example of the Timebase Timer"

# 5.1 Overview of the Timebase Timer

**The timebase timer is a 21-bit free-run counter that counts up in synchronization with the internal count clock (divide-by-two of the main clock oscillation) and provides the interval timer function in which four kinds of interval time can be selected. The timebase timer also supplies timer output for the oscillation stabilization wait time and an operating clock for the watchdog timer and others.**
**The timebase timer stops its operations in a mode in which the main clock oscillation stops.**

## ■ Interval Timer Function

The interval timer is a function used to generate an interrupt repeatedly at constant intervals.

- An interrupt occurs if the interval timer bit of the counter of the timebase timer overflows.
- The interval timer bit (interval time) can be selected from four kinds of interval time.

Table 5.1-1 "Interval Time of the Timebase Timer" lists the interval time of the timebase timer.

**Table 5.1-1 Interval Time of the Timebase Timer**

| Internal count clock cycle | Interval time |
|---|---|
| $2/F_{CH}$ (0.2µs) | $2^{13}/F_{CH}$ (Approx. 0.82 ms)<br>$2^{15}/F_{CH}$ (Approx. 3.3 ms)<br>$2^{18}/F_{CH}$ (Approx. 26.2 ms)<br>$2^{22}/F_{CH}$ (Approx. 419.4 ms) |

$F_{CH}$: Main clock oscillation
Values in ( ) shows the interval time when the main clock operates with 10 MHz oscillation.

## ■ Clock Supply Function

The clock supply function is a function used to supply timer output (four options) for the oscillation stabilization wait time of the main clock and the operating clock to part of the peripheral functions.

Table 5.1-2 "Clocks Supplied from the Timebase Timer" lists the cycles of clocks supplied to each peripheral function from the timebase timer.

**Table 5.1-2 Clocks Supplied from the Timebase Timer**

| Clock supply destination | Clock cycle | Remarks |
|---|---|---|
| Main clock oscillation stabilization wait time | $2^{14}/F_{CH}$ (Approx. 1.63 ms) | Selected by the oscillation stabilization wait time select bits (SYCC: WT1, WT0) of the system clock control register in the clock controller |
| | $2^{17}/F_{CH}$ (Approx. 113.1 ms) | |
| | $2^{18}/F_{CH}$ (Approx. 26.2 ms) | |
| Watchdog timer | $2^{21}/F_{CH}$ (Approx. 209.7 ms) | Count-up clock of the watchdog timer |
| LCD controller/driver | $2^{8}/F_{CH}$ (Approx. 25.6 µs) | Clock for frame cycle generation |

$F_{CH}$: Main clock oscillation
Values in ( ) shows the interval time when the main clock operates with 10 MHz oscillation

**Note:**

The oscillation cycle is unstable just after the oscillation start and the oscillation stabilization wait time serves as a guideline.

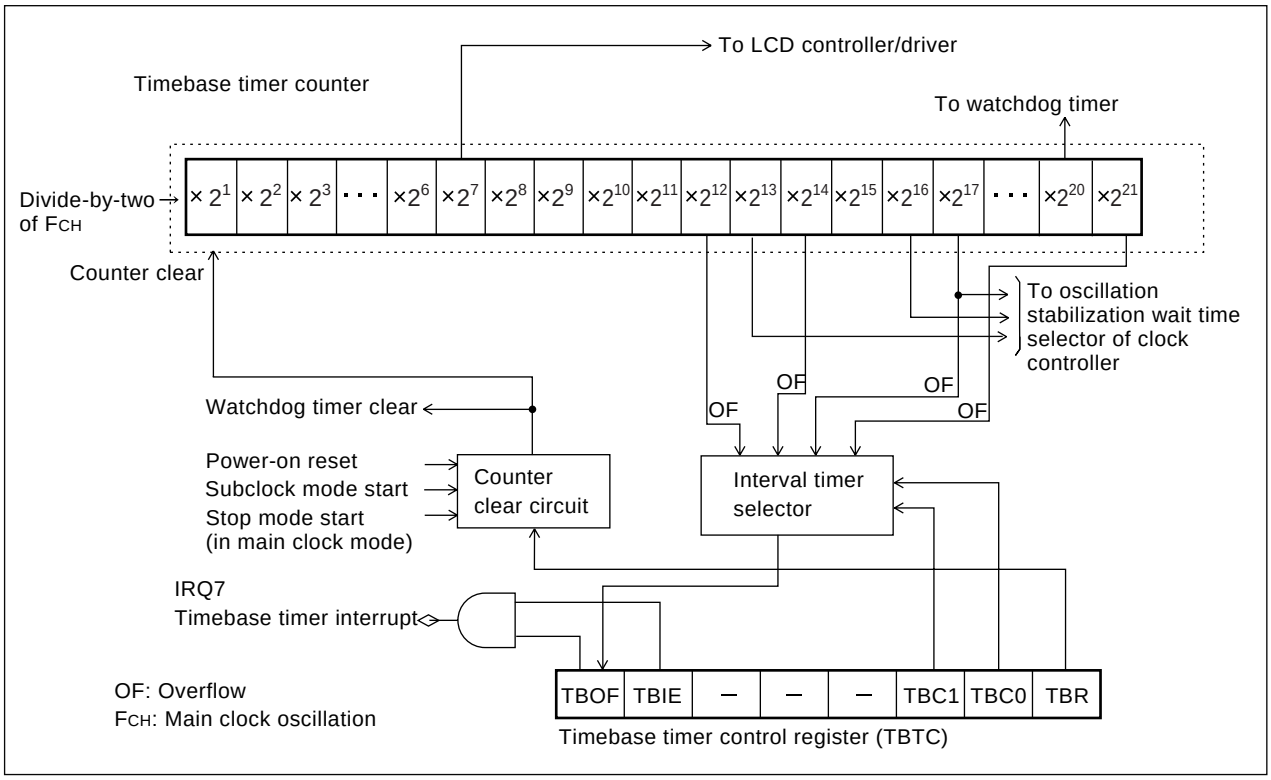
# 5.2 Configuration of the Timebase Timer

**The timebase timer is made up of the following four blocks:**

- **Timebase timer counter**
- **Counter clear circuit**
- **Interval timer selector**
- **Timebase timer control register (TBTC)**

## ■ Block Diagram of the Timebase Timer

**Figure 5.2-1 Block Diagram of the Timebase Timer**




### ❍ Timebase timer counter

21-bit up-counter using the count clock of divide-by-two of the main clock oscillation. This counter stops operating when the main clock oscillation stops.

### ❍ Counter clear circuit

Clears the counter when, in addition to the setting (TBR=0) by the TBTC register, a transition to the main stop mode (STBC: STP=1) or subclock mode (SYCC: SCS=0), or a power-on reset occurs.

- **Interval timer selector**

  Circuit to select one bit for the interval timer from four bits of the timebase timer counter. The overflow of the selected bit causes an interrupt.

- **Timebase timer control register (TBTC)**

  This register is used to select the interval time, clear the counter, control interrupts, and check the states.

# 5.3 Timebase Timer Control Register (TBTC)

**The timebase timer control register (TBTC) is used to select the interval time, clear the counter, control interrupts, and check the state.**

## ■ Timebase Timer Control Register (TBTC)

**Figure 5.3-1 Timebase Timer Control Register (TBTC)**

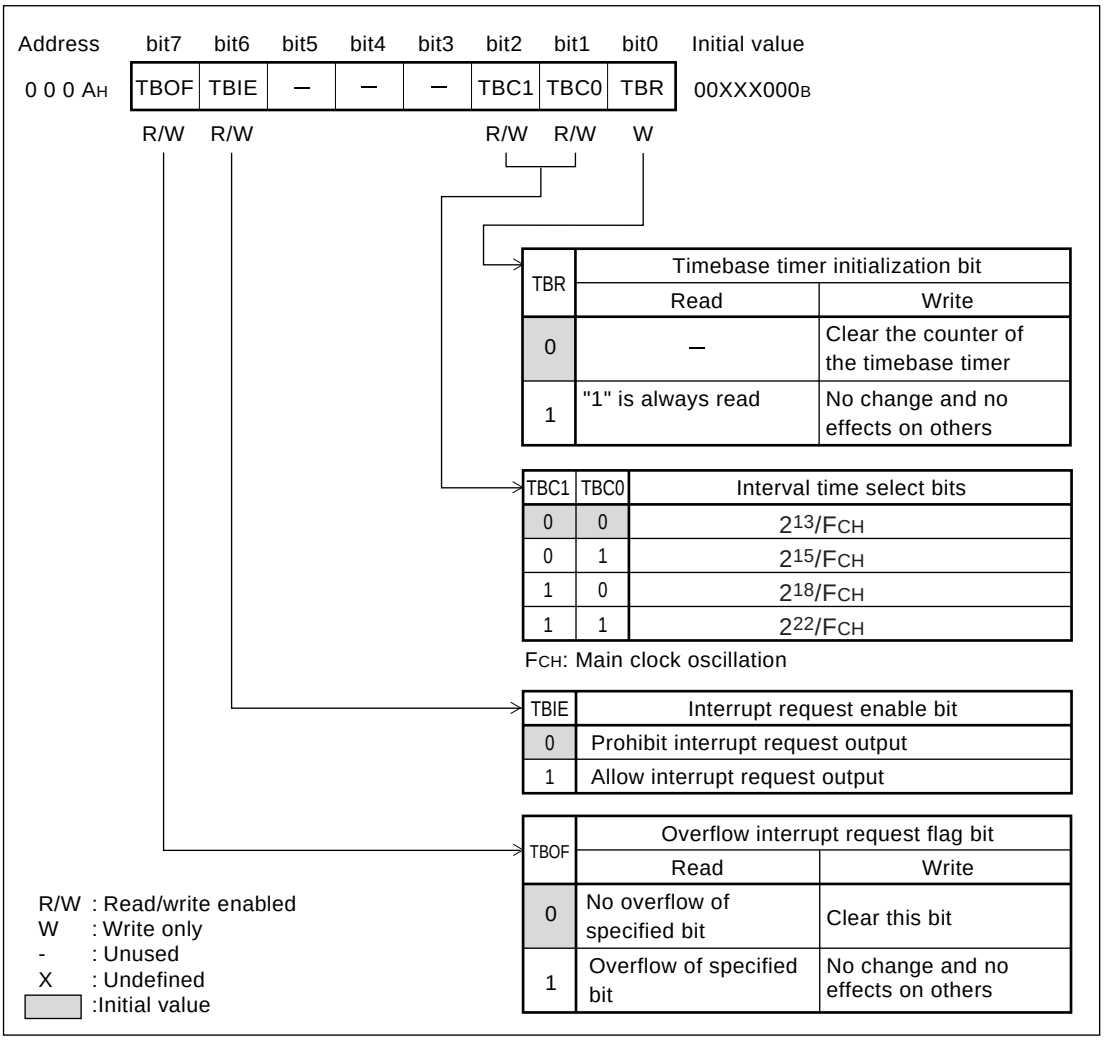


**Table 5.3-1 Explanation of Functions of Each Bit of the Timebase Timer Control Register (TBTC)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | TBOF:<br>Overflow interrupt request flag bit | • This bit is set to "1" if the specified bit of the counter of the timebase timer overflows.<br>• If both this bit and the interrupt enable bit (TBIE) are "1", an interrupt request is output.<br>• If "0" is written into this bit, the counter is cleared. If "1" is written, no change occurs and operations are not affected. |
| Bit 6 | TBIE:<br>Interrupt request enable bit | • Bit to allow/prohibit interrupt request output to CPU. If both this bit and the overflow interrupt request flag bit (TBOF) are "1", an interrupt request is output. |
| Bit 5<br>Bit 4<br>Bit 3 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 2<br>Bit 1 | TBC1, TBC0:<br>Interval time select bit | • Bits to select the interval timer cycle<br>• Bits for the interval timer of the counter of the timebase timer are specified.<br>• Four kinds of interval time can be selected. |
| Bit 0 | TBR:<br>Timebase timer initialization bit | • Bit to clear the counter of the timebase timer<br>• If "0" is written into this bit, the counter is cleared to "$000000_H$". If "1" is written, no change occurs and operations are not affected.<br>**Reference:**<br>"1" is always read. |

# 5.4 Timebase Timer Interrupt

**As an interrupt source of the timebase timer, an overflow of the specified bit of the timebase timer counter is available (interval timer function).**

## ■ Interrupt when the Interval Timer Function is Active

If an overflow of the selected interval timer bit occurs after the counter is counted up by the internal count clock, the overflow interrupt request flag bit (TBTC: TBOF) is set to "1". At this time, if the interrupt request enable bit is set (TBTC: TBIE=1), an interrupt request to CPU (IRQ7) is generated. Clear the interrupt request by writing "0" into the TBOF bit using an interrupt processing routine. The TBOF bit is set whenever an overflow of the specified bit occurs regardless of the value of the TBIE bit.

**Note:**

To allow interrupt request output (TBIE=1) after releasing a reset, clear (TBOF=0) the TBOF bit at the same time.

**Reference:**

If the TBIE bit is changed from prohibition to permission (0 --> 1) when the TBOF bit is "1", an interrupt request is issued immediately.

If the counter clear (TBTC: TBR=0) and an overflow of the selected bit occur at the same time, the TBOF bit is not set.

## ■ Oscillation Stabilization Wait Time and Timebase Timer Interrupts

If interval time shorter than the oscillation stabilization wait time of the main clock is set, an interval interrupt request (TBTC: TBOF=1) of the timebase timer is generated when the operation in main clock mode starts. In this case, prohibit (TBTC: TBIE=0) interrupts of the timebase timer when making a transition to a mode in which the oscillation of the main clock stops (main stop and subclock modes).

## ■ Register and Vector Table Related to the Timebase Timer Interrupts

**Table 5.4-1 Register and Vector Table Related to the Timebase Timer Interrupts**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ7 | ILR2 ($007C_H$) | L71 (bit 7) | L70 (bit 6) | $FFEC_H$ | $FFED_H$ |

For the interrupt operations, see Section 3.4.2 "Interrupt Processing".

# 5.5 Operation of the Timebase Timer

**The timebase timer provides the interval timer function and supplies the clock to part of the peripheral functions.**

## ■ Operation of the Interval Timer Function (Timebase Timer)

The setting in Figure 5.5-1 "Setting of the Interval Timer Function" is required for the operation of the interval timer function.

**Figure 5.5-1 Setting of the Interval Timer Function**

| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|---|
| TBTC | TBOF | TBIE | — | — | — | TBC1 | TBC0 | TBR |
| | 0 | 1 | | | | ◎ | ◎ | 0 |

◎ : Bit used
1 : 1 is set
0 : 0 is set

The counter of the timebase timer continues to count up provided the main clock oscillates in synchronization with the internal count clock (divide-by-two of the main clock oscillation).

If the counter is cleared (TBR=0), it starts counting up from "0". If an overflow of the bit for the interval timer occurs, "1" is set to the overflow interrupt request flag bit (TBOF). That is, starting when clearing occurs, an interrupt request is generated at regular intervals of the selected time.

## ■ Operation of the Clock Supply Function

The timebase timer is also used as a timer to generate the oscillation stabilization wait time of the main clock. Counting of the oscillation stabilization wait time starts when the counter of the timebase timer is cleared and ends when an overflow of the bit for oscillation stabilization wait time occurs. Three kinds of oscillation stabilization wait time can be selected by the setting of the oscillation stabilization wait time select bits (SYCC: WY1, WT0) of the system clock control register.

The timebase timer supplies the clock to the watchdog timer, A/D converter, and LCD controller/driver. When the counter of the timebase timer is cleared, operations of the continuous activation cycles of the A/D converter and those of the frame cycles of the LCD controller/driver are affected. The counter of the watchdog timer is cleared at the same time provided the timebase timer output is selected (WDTC: CS=0).

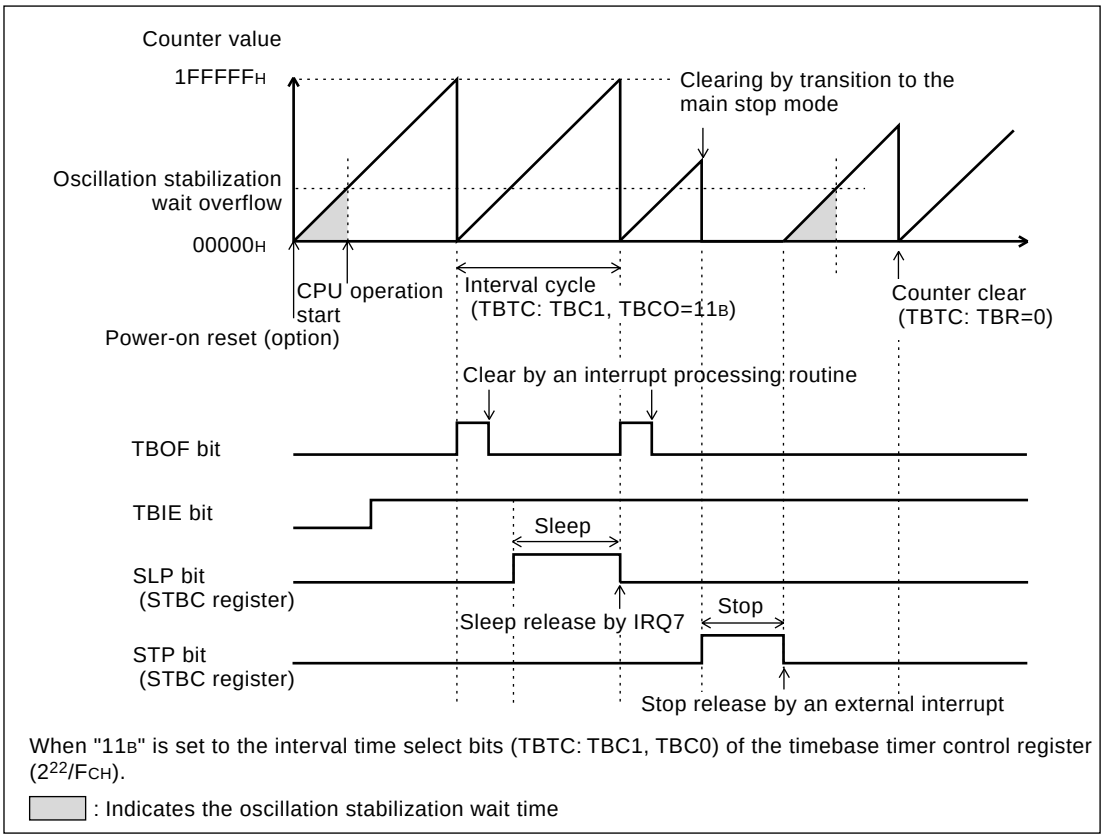
## ■ Operations of the Time-based Timer

Figure 5.5-2 "Operations of the Timebase Timer" shows the operations in the following states:

- When a power-on reset occurs
- When a transition to the sleep mode occurs during operation of the interval timer function in main clock mode
- When a transition to the main stop mode occurs
- When the counter clear is requested

In subclock mode and main stop mode, the timebase timer is cleared and its operation is stopped. When returning from the subclock mode or main stop mode, the oscillation stabilization wait time is counted by the timebase timer.

**Figure 5.5-2 Operations of the Timebase Timer**




When "11B" is set to the interval time select bits (TBTC: TBC1, TBC0) of the timebase timer control register ($2^{22}/F_{CH}$).

: Indicates the oscillation stabilization wait time

# 5.6 Notes on Using the Timebase Timer

**The following describes the precautions to take when using the timebase timer.**

## ■ Notes on Using the Timebase Timer

### ❍ Precautions when setting the timebase timer with programs

Because it is impossible to return from interrupt processing if the interrupt request flag bit (TBTC: TBOF) is "1" and the interrupt request enable bit (TBTC: TBIE=1) is allowed, the TBOF bit must be cleared.

### ❍ Clearing the timebase timer

The timebase timer is cleared, in addition to clearing by the timebase timer initialization bit (TBTC: TBR=0), when the oscillation stabilization wait time of the main clock is required. If the timebase timer is selected (WDTC: CS=0) as the count clock of the watchdog timer, the watchdog timer is cleared when the timebase timer is cleared.

### ❍ Using the timebase timer as a timer for the oscillation stabilization wait time

Since the main clock oscillation is stopped when the power is turned on or is in main stop mode or subclock mode, the oscillator takes the oscillation stabilization wait time of the main clock. The appropriate oscillation stabilization wait time must be selected according to the type of resonator connected to the oscillator (clock generator) of the main clock.

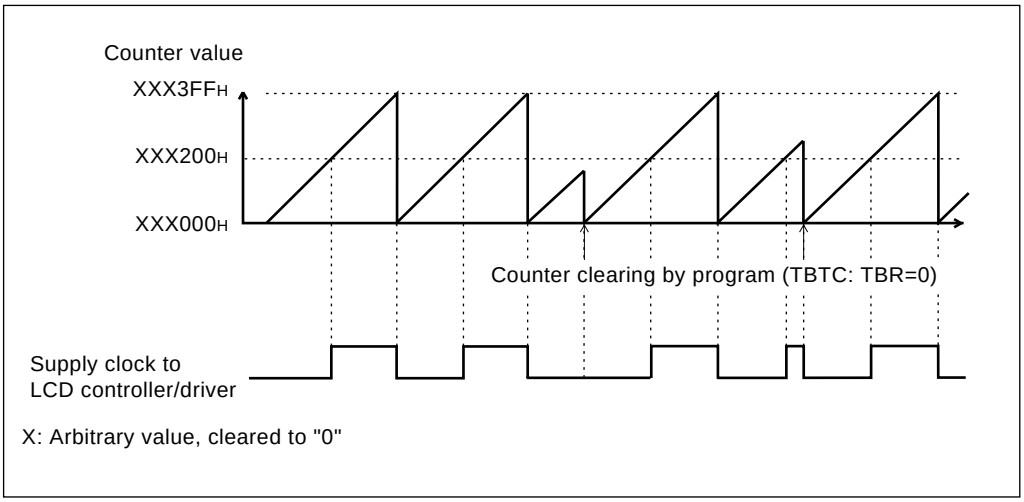
For details, see Section 3.7.5 "Oscillation Stabilization Wait Time"

### ❍ Precautions for peripheral functions to which the clock is supplied from the timebase timer

In a mode in which the main clock oscillation stops, the counter is cleared and the timebase timer stops its operation. If the counter of the timebase timer is cleared, the "H" level of the clock supplied by the timebase timer is short and its "H" level may be longer by a 1/2 cycle at the most because the output originates from the initial state. Though the clock for the watchdog timer is also output from the initial state, the watchdog timer works in normal cycles because the counter of the watchdog timer is cleared simultaneously.

Figure 5.6-1 "Effects on the LCD Controller/Driver when the Timebase Timer is Cleared" shows the effects on the LCD controller/driver when the timebase timer is cleared.

**Figure 5.6-1 Effects on the LCD Controller/Driver when the Timebase Timer is Cleared**



# 5.7 Program Example of the Timebase Timer

**The following shows a program example of the timebase timer.**

## ■ Program Example of the Timebase Timer

### ❍ Processing specifications

Generate the interval timer interrupt of $2^{18}/F_{CH}$ ($F_{CH}$: main clock oscillation) repeatedly. The interval time at this time is about 26 ms (for 10 MHz operations).

❍ **Coding example**

```
TBTC    .EQU     0000AH    ; Address of the timebase timer
                             control register
TBOF    .EQU     TBTC:7    ; Definition of interrupt
                             request flag bit
ILR2    .EQU     007CH     ; Address of the interrupt level
                             setting register 2
        .SECTION INT_V, DATA, LOCATE=0    ; [DATA SEGMENT]
        .ORG     0FFECH
IRQ7    .DATA.H  WARI      ; Setting interrupt vector
;INT_V  ENDS
;----Main program-----------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1      ; [CODE SEGMENT]
                             ; Stack pointer (SP) and other
                               are assumed to have been initialized
          :
        CLRI                 ; Interrupt disable
        MOV      ILR2,#01111111B  ; Setting interrupt level(level 1)
        MOV      TBTC,#01000100B  ; Clearing interrupt request
                                    flag, enabling interrupt
                                    request output, selecting 2^18/F_CH,
                                    and clearing timebase timer
        SETI                 ; Interrupt enable
          :
;----Interrupt program------------------------------------------------
WARI    CLRB     TBOF      ; Clearing interrupt request flag
        PUSHW    A
        XCHW     A,T
        PUSHW    A
          :
         User processing
          :
        POPW    A
        XCHW    A, T
        POPW    A
        RETI
        ENDS
; --------------------------------------------------------------------
        .END
```

# CHAPTER 6 WATCHDOG TIMER

**This chapter describes the functions and operations of the watchdog timer.**


6.1 "Overview of the Watchdog Timer"

6.2 "Configuration of the Watchdog Timer"

6.3 "Watchdog Timer Control Register (WDTC)"

6.4 "Operation of the Watchdog Timer"

6.5 "Notes on Using the Watchdog Timer"

6.6 "Program Example of the Watchdog Timer"

# 6.1 Overview of the Watchdog Timer

**The watchdog timer is a 1-bit timer which accepts the output of either the timebase timer operating with the main clock or the watch prescaler operating with the subclock as the count clock. If the watchdog timer is not cleared for a specified period of time after activation, CPU is reset.**

## ■ Watchdog Timer Function

The watchdog timer is a counter against program runaway. Once the watchdog timer is activated, it is necessary to continue clearing it periodically within a specified period of time. If the watchdog timer is not cleared for a specified period of time, for example, because the program slips into an endless loop, a watchdog reset of the four instruction cycles is generated to CPU.

As the count clock of the watchdog timer, the output of either the timebase timer or watch prescaler can be selected.

The interval time of the watchdog timer is as listed in Table 6.1-1 "Interval Time of the Watchdog Timer". If the watchdog timer is not cleared, a watchdog reset occurs between the minimum and maximum times. Clear the counter within the minimum time of this table.

**Table 6.1-1 Interval Time of the Watchdog Timer**

| | Count clock | |
|---|---|---|
| | **Timebase timer output (for main clock oscillation 10 MHz)** | **watch prescaler output (for subclock oscillation 32. 768 kHz)** |
| Minimum time | Approx. 209.7 ms(*1) | 500 ms(*2) |
| maximum time | Approx. 419.4 ms | 1000 ms |

*1: Divide-by-two of the main clock oscillation ($F_{CH}$) x count of the timebase timer ($2^{20}$)
*2: Cycle of the subclock oscillation ($F_{CL}$) x count of the watch prescaler ($2^{14}$)

For the minimum and maximum times of the interval time of the watchdog timer, see Section 6.4 "Operation of the Watchdog Timer"

**Note:**

The counter of the watchdog timer is cleared at the same time the timebase timer is cleared (TBTC: TBR=0) in a state in which the output of the timebase timer is selected as the count clock, or it is cleared at the same time the watch prescaler is cleared (WPCR: WCLR=0) in a state in which the output of the watch prescaler is selected as the count clock. Thus, if the counter (timebase timer or watch prescaler) used as the count clock is cleared repeatedly within an interval time of the watchdog timer, the watchdog timer will not function.

**Reference:**

If a transition to the sleep mode, stop mode, or watch mode occurs, the counter of the watchdog timer is cleared and will not operate until normal operation (RUN state) is resumed.

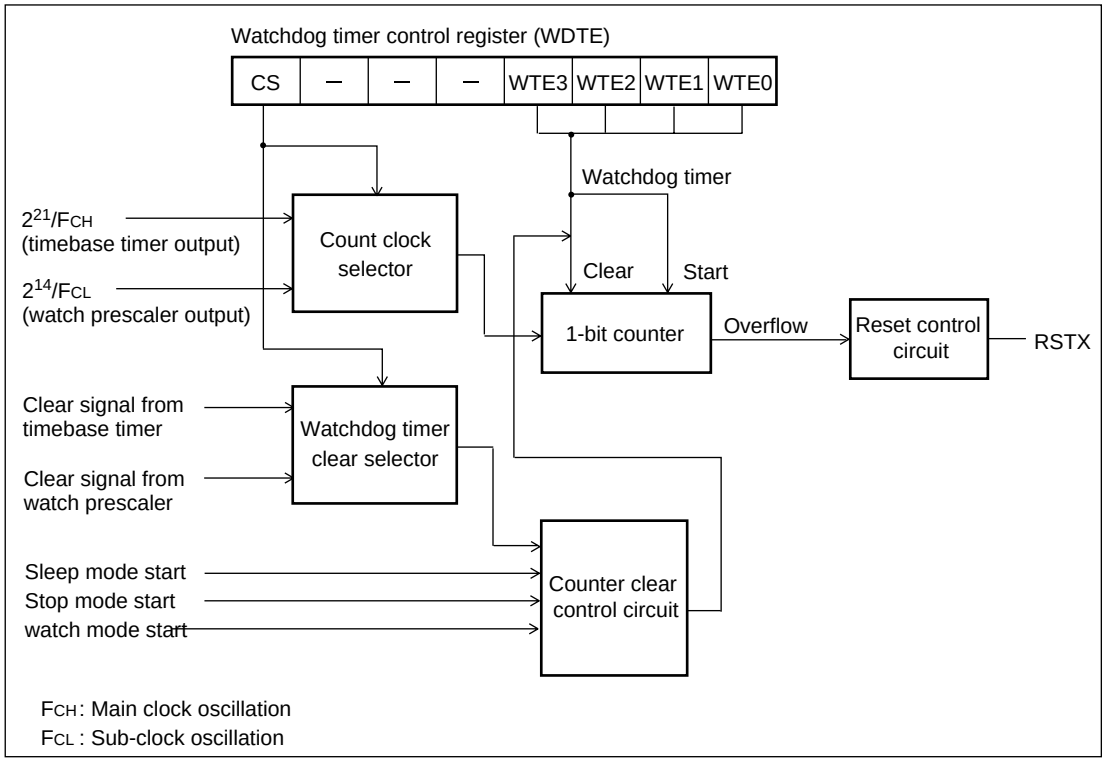
# 6.2 Configuration of the Watchdog Timer

**The watchdog timer is made up of the following six blocks:**

- **Count clock selector**
- **Watchdog timer counter**
- **Reset control circuit**
- **Watchdog timer clear selector**
- **Counter clear control circuit**
- **Watchdog timer control register (WDTC)**

## ■ Block Diagram of the Watchdog Timer

**Figure 6.2-1 Block Diagram of the Watchdog Timer**




### ❍ Count clock selector

The count clock selector selects the count clock of the watchdog timer counter. As the count clock, the output of either the timebase timer or the watch prescaler can be selected.

### ❍ Watchdog timer counter (1-bit counter)

The watchdog timer counter is a 1-bit counter whose count clock is the output of either the timebase timer or the watch prescaler.

- **Reset control circuit**

  The reset control circuit generates a reset signal to CPU when an overflow of the watchdog timer counter occurs.

- **Watchdog timer clear selector**

  The watchdog timer clear selector selects the watchdog timer clear signal from the timebase timer or watch prescaler simultaneously with the count clock selector.

- **Counter clear control circuit**

  The counter clear control circuit controls the watchdog timer counter clearing and operation stop.

- **Watchdog timer control register (WDTC)**

  The watchdog timer control register is used to select the count clock and activate/clear the watchdog timer counter. Since this register is write only, bit manipulation instructions cannot be used.

# 6.3 Watchdog Timer Control Register (WDTC)

**The watchdog timer Control Register (WDTC) is used to activate/clear the watchdog timer.**

## ■ Watchdog Timer Control Register (WDTC)

**Figure 6.3-1 Watchdog Timer Control Register (WDTC)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 0 9H | CS | – | – | – | WTE3 | WTE2 | WTE1 | WTE0 | 0XXXXXXXB |
| R/W | | | | | W | W | W | W | |

| WTE3 | WTE2 | WTE1 | WTE0 | Watchdog control bit |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | - Activate watchdog timer (for the 1st write after reset)<br>- Clear watchdog timer (for the 2nd and later write after reset) |
| Otherwise | | | | No operation |

| CS | Count clock switch bit |
|---|---|
| 0 | Timebase timer output cycle ($2^{21}$/FCH *1) |
| 1 | Watch prescaler output cycle ($2^{14}$/FCL *2) |

R/W : Read/write enabled
W : write only
- : Unused
X : Undefined
□ : Initial value

(Note) Since this register is write only, bit manipulation instructions cannot be used.
*1: FCH : Main clock oscillation
*2: FCL : Sub-clock oscillation

**Table 6.3-1 Explanation of the Functions of Each Bit of the Watchdog Timer Control Register (WDTC)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | CS:<br>Count clock switch bit | • Select the count clock of the watchdog timer when activating the watchdog timer.<br>• As the count clock, the output of either the timebase timer or the watch prescaler can be selected.<br>**Note:**<br>• To use the subclock mode, select the output of the watch prescaler.<br>• Select the count clock simultaneously with activation of the watchdog timer and do not change it after the activation.<br>• Bit manipulation instructions cannot be used. |
| Bit 6<br>Bit 5<br>Bit 4 | Unused bist | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 3<br>Bit 2<br>Bit 1<br>Bit 0 | WTE3, WTE2, WTE1, WTE0:<br>Watchdog control bit | • If "$0101_B$" is written into these bits, the watchdog timer is activated (the 1st write after reset) or cleared (the 2nd or later write after reset).<br>• Writing anything other than "$0101_B$" does not affect operations.<br>**Note:**<br>"$1111_B$" is read. Bit manipulation instructions cannot be used. |

# 6.4 Operation of the Watchdog Timer

**The watchdog timer generates a watchdog reset when the watchdog timer counter overflows.**

## ■ Operation of the Watchdog Timer

### ❍ Activating the watchdog timer

- The watchdog timer can be activated by writing the 1st "$0101_B$" into the watchdog control bits (WDTC: WTE3 to 0) of the watchdog timer control register after a reset. At this time, specify the count clock switch bit (WDTC: CS) simultaneously.
- A watchdog timer that is activated can only be stopped by a reset.

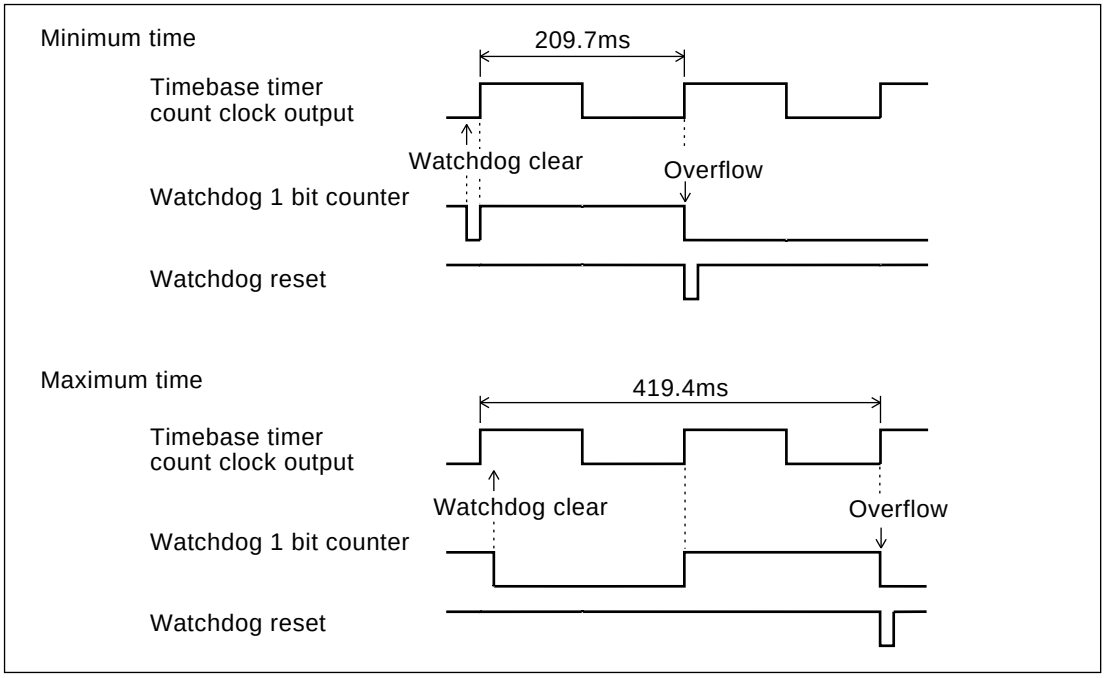
### ❍ Clearing the watchdog timer

- The counter of the watchdog timer can be cleared by writing the 2nd or subsequent "$0101_B$" into the watchdog control bits (WDTC: WTE3 to 0) of the watchdog timer control register after a reset.
- If the counter is not cleared within the interval time of the watchdog timer, an overflow of the counter occurs and an internal reset signal of the four instruction cycles is generated.

### ❍ Watchdog timer interval time

The interval time is changed by the timing of clearing the watchdog timer. Figure 6.4-1 "Watchdog Timer Clearing and Interval Time" shows the relations between the clearing timing of the watchdog timer and the interval time when the output of the timebase timer is selected as the count clock (if the main clock oscillation is 10 MHz).

**Figure 6.4-1 Watchdog Timer Clearing and Interval Time**



# 6.5 Notes on Using the Watchdog Timer

**The following describes the precautions to take when using the watchdog timer.**

## ■ Notes on Using the Watchdog Timer

### ❍ Stopping the watchdog timer

A watchdog timer that is activated can only be stopped by a reset.

### ❍ Selecting the count clock

The count clock switch bit (WDTC: CS) can be rewritten only if "$0101_B$" is written into the watchdog control bits (WDTC: WTE3 to 0) when the watchdog timer is activated. Thus, a write operation by bit manipulation instructions is not possible. Do not change the settings after activation.

Since the main clock oscillation stops in subclock mode, the timebase timer does not operate.

To enable operation of the watchdog timer in subclock mode, the watch prescaler (WDTC: CS=1) must be selected as the count clock in advance.

### ❍ Clearing the watchdog timer

- If the counter (timebase timer or watch prescaler) used as the count clock of the watchdog timer is cleared, the counter of the watchdog timer is cleared at the same time.
- If a transition to the sleep mode, stop mode, or watch mode occurs, the counter of the watchdog timer is cleared.

### ❍ Precautions when creating a program

When creating a program in which the watchdog timer is cleared repeatedly in the main loop, the processing time of the main loop including interrupt processing must be equal to or less than the minimum watchdog timer interval time.

### ❍ Operations in subclock mode

If a watchdog reset occurs in subclock mode, operation starts in main clock mode after taking the oscillation stabilization wait time. At this time, a reset signal is output during oscillation stabilization wait time.

# 6.6 Program Example of the Watchdog Timer

**The following shows a program example in which the watchdog timer is used.**

## ■ Program Example of the Watchdog Timer

### ❍ Processing specifications

- Select the watch prescaler just after starting the program to activate the watchdog timer.
- Clear the watchdog timer each time in a loop of the main program.
- The main loop must make a round in less than the interval minimum time (about 335.5 ms for 12.5 kHz operation), including the interrupt processing time, of the watchdog timer.

❍ **Coding example**

```
WDTC    .EQU     0009H        ; Address of the watchdog timer
                                control register
WDT_CLR .EQU     10000101B
        .SECTION VECT, DATA, LOCATE=0    ; [DATA SEGMENT]
        .ORG     0FFFEH
RST_V   .DATA.H  PROG         ; Setting reset vector
;VECT   ENDS
;----Main program-----------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1     ; [CODE SEGMENT]
PROG                          ; Initialization routine for reset
        MOVW     SP,#0280H    ; Setting initial value of stack pointer
                                (for interrupt)
          :
         Initializing interrupt or other peripheral functions
          :
INIT
        MOV      WDTC,#WDT_CLR   ; Activating watchdog timer
                                   Selection of the watchdog prescaler
                                   as the count clock
          :
MAIN
        MOV      WDTC,#WDT_CLR   ; Clearing watchdog timer
          :
         User processing (interrupt may occur in this processing.)
          :
        JMP      MAIN         ; Ensure that the time necessary
                                for running the loop is shorter
                                than the minimum time interval
                                of the watchdog timer.
        ENDS
;---------------------------------------------------------------------
        .END
```

# CHAPTER 7 WATCH PRESCALER

**This chapter describes the functions and operations of the watch prescaler.**


7.1 "Overview of the Watch Prescaler"

7.2 "Configuration of the Watch Prescaler"

7.3 "Watch Prescaler Control Register (WPCR)"

7.4 "Watch Prescaler Interrupt"

7.5 "Operation of the Watch Prescaler"

7.6 "Notes on Using the Watch Prescaler"

7.7 "Program Example of the Watch Prescaler"

# 7.1 Overview of the Watch Prescaler

**The watch prescaler is a 17-bit free-run counter that counts up in synchronization with the subclock generated in the clock generator and has an interval timer function that provides for the selection of six kinds of interval time.**
**The watch prescaler also supplies the timer output of subclock oscillation stabilization wait time and the operating clock of the watchdog and other timers.**

## ■ Interval Timer Function (Watch Interrupt)

The interval timer function is a function used to generate an interrupt repeatedly at regular intervals using the subclock as the count clock.

- An interrupt is generated by divide-by output for the interval timer of the watch prescaler.
- Four kinds of divide-by output (interval time) for the interval timer can be selected.
- The counter of the watch prescaler can be cleared.

Table 7.1-1 "Interval Time of the Watch Prescaler" lists the interval time of the watch prescaler.

**Table 7.1-1 Interval Time of the Watch Prescaler**

| Subclock cycle | Interval time |
|---|---|
| $1/F_{CL}$ (Approx. 30.5 μs) | $2^{10}/F_{CL}$ (31.25 ms) |
| | $2^{13}/F_{CL}$ (0.25 s) |
| | $2^{14}/F_{CL}$ (0.50 s) |
| | $2^{15}/F_{CL}$ (1.00 s) |
| | $2^{16}/F_{CL}$ (2.00 s) |
| | $2^{17}/F_{CL}$ (4.00 s) |

$F_{CL}$: Subclock oscillation
Values in ( ) represent the interval time when the subclock oscillation is operating at 32.768 kHz.

**Note:**

If a resonator is not connected to the subclock, the watch prescaler cannot be used.

## ■ Clock Supply Function

The clock supply function of the watch prescaler is a function used to supply the timer output (one) for oscillation stabilization wait time of the subclock and the clock for the watchdog timer.

Table 7.1-2 "Clocks Supplied from the Watch Prescaler" lists the clock cycles supplied to each

peripheral function from the watch prescaler.

**Table 7.1-2 Clocks Supplied from the Watch Prescaler**

| Subclock supply destination | Subclock cycle | Remarks |
|---|---|---|
| Subclock oscillation stabilization wait time | $2^{15}/F_{CL}$ (1.00 s) | Do not make a transition to the subclock mode during oscillation stabilization wait time |
| Watchdog timer | $2^{14}/F_{CL}$ (0.5 s) | Count-up clock of the watchdog timer |

$F_{CL}$: Subclock oscillation
Values in ( ) represent the subclock cycles when the subclock oscillation is operating at 32.768 kHz.

**Reference:**

Because the oscillation cycles are unstable just after the oscillation starts, the oscillation stabilization wait timer serves as a guideline.

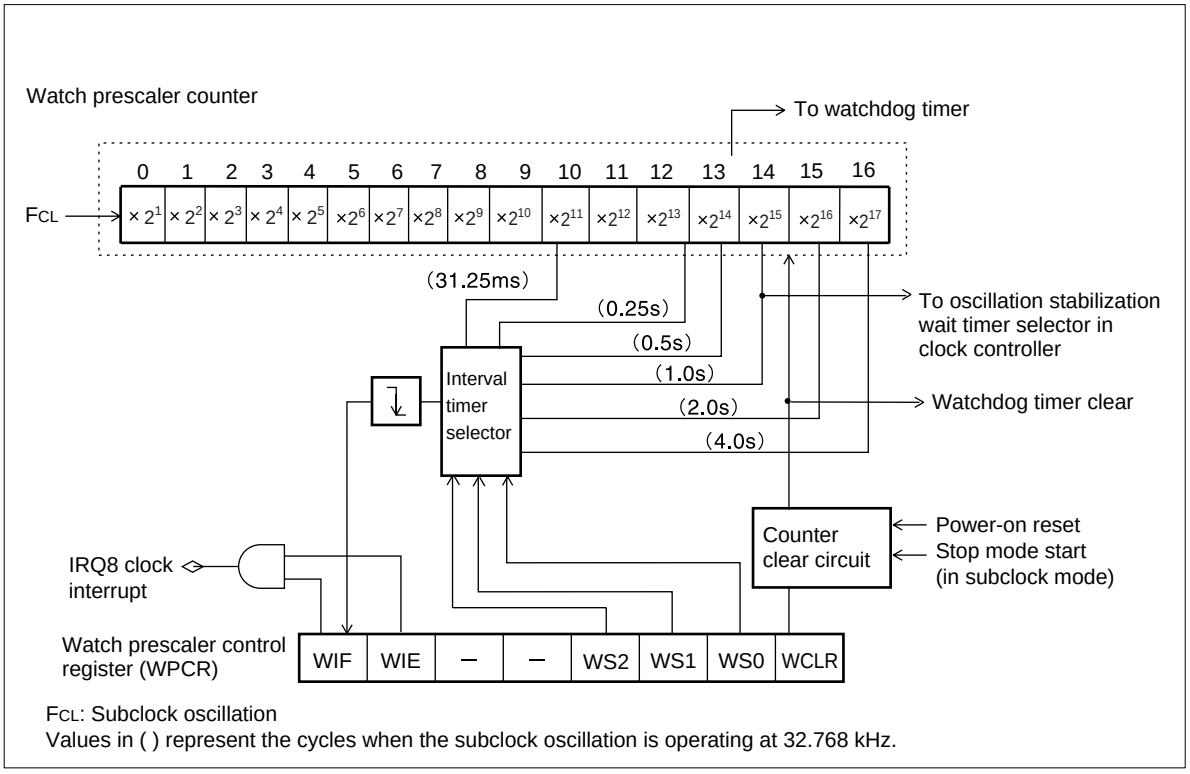
# 7.2 Configuration of the Watch Prescaler

**The watch prescaler comprises the following blocks:**

- **watch prescaler counter**
- **Counter clear circuit**
- **Interval timer selector**
- **watch prescaler control register (WPCR)**

## ■ Block Diagram of the Watch Prescaler

**Figure 7.2-1 Block Diagram of the Watch Prescaler**




FCL: Subclock oscillation
Values in ( ) represent the cycles when the subclock oscillation is operating at 32.768 kHz.

❍ **Watch prescaler counter**

17-bit up-counter using the subclock oscillation as the count clock.

❍ **Counter clear circuit**

The counter clear circuit clears the counter when, in addition to the setting by the WPCR register (WCLR=0), a transition to the sub-stop mode (STBC: STP=1) or a power-on reset occurs.

❍ **Interval timer selector**

Circuit to select one divide-by output from six kinds of divide-by output from the watch prescaler counter. The falling edges of the selected divide-by output become an interrupt source.

❍ **Watch prescaler control register (WPCR)**

This register is used to select the interval time, clear the counter, control interrupts, and check status.

# 7.3 Watch Prescaler Control Register (WPCR)

**The watch prescaler control register (WPCR) is a register used to select the interval time, clear the counter, control interrupts, and check status.**

## ■ Watch Prescaler Control Register (WPCR)

**Figure 7.3-1 Watch Prescaler Control Register (WPCR)**

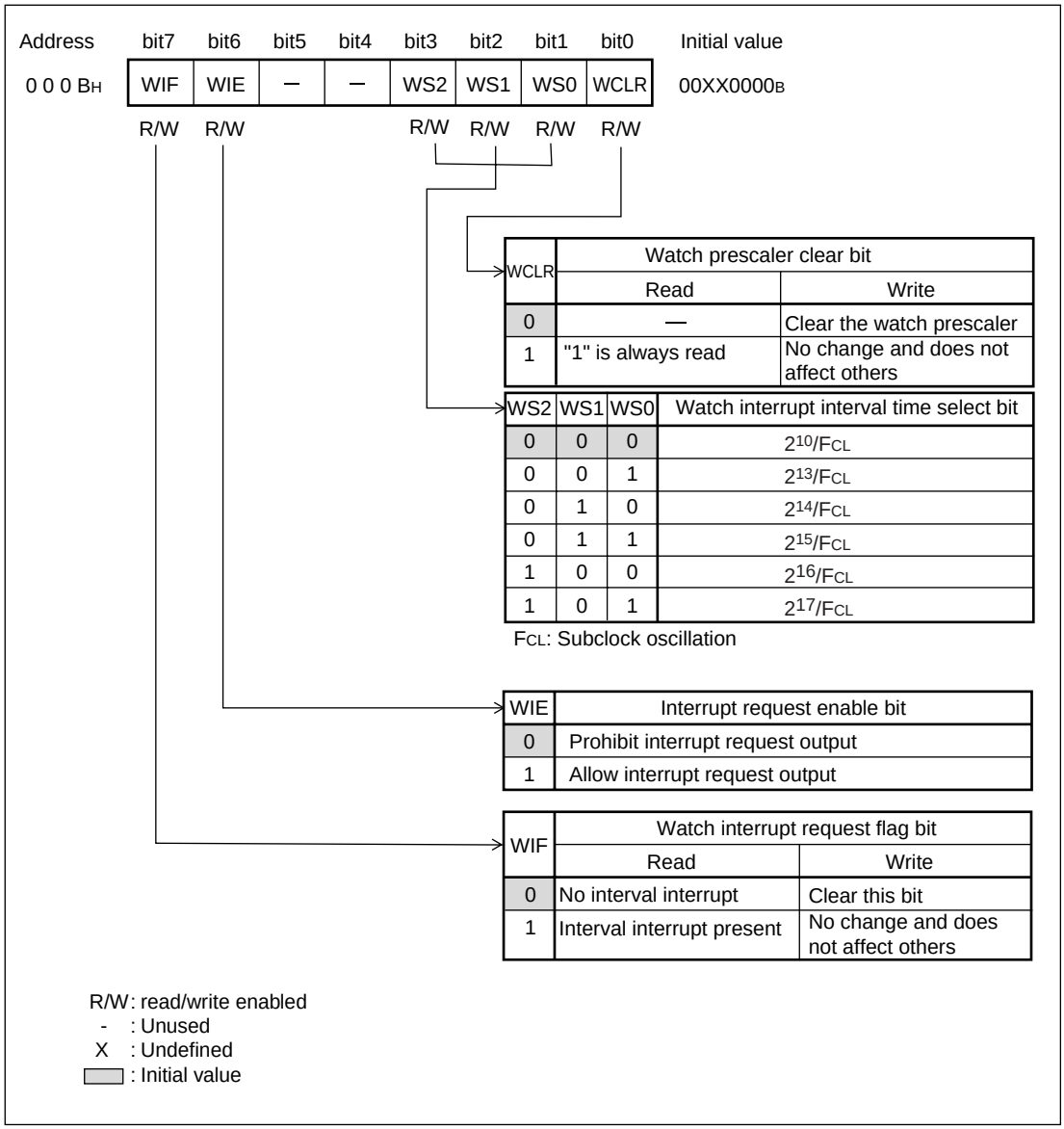


**Table 7.3-1 Explanation of the Functions of Each Bit of the Watch Prescaler Control Register (WPCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | WIF:<br>Watch interrupt request flag bit | • "1" is set by the falling edges of the selected divide-by output for interval timer.<br>• If both this bit and the interrupt request enable bit (WIE) are "1", an interrupt request is output.<br>• This bit is cleared if "0" is written into this bit. If "1" is written, no change occurs and no operation is affected. |
| Bit 6 | WIE:<br>Interrupt request enable bit | • Bit to allow/prohibit interrupt request output to CPU.<br>• If both this bit and the watch interrupt request flag bit (WIE) are "1", an interrupt request is output. |
| Bit 5<br>Bit 4 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 3<br>Bit 2<br>Bit 1 | WS2, WS1, WS0:<br>Watch interrupt interval time select bit | • Bits to select the interval timer cycle<br>• Bits for the interval timer of the counter of the watch prescaler are specified.<br>• Six kinds of interval time can be selected. |
| Bit 0 | WCLR:<br>watch prescaler clear bit | • Bit to clear the counter of the watch prescaler<br>• If "0" is written into this bit, the counter is cleared to "$000000_H$". If "1" is written, no change occurs and no operation is affected.<br>**Reference:**<br>"1" is always read. |

# 7.4 Watch Prescaler Interrupt

**The watch prescaler generates interrupt requests using the falling edges of the selected divide-by output (interval timer function).**

## ■ Interrupt when the Interval Timer Function is Active (Watch Interrupt)

The counter for the watch prescaler counts up using the subclock oscillation. When the specified interval time passes, if not in main stop mode, the watch interrupt request flag bit (WPCR: WIF=1) is set to "1". At this time, if the interrupt request enable bit is set (WPCR: WIE=1), an interrupt request to CPU (IRQ8) is issued. Clear the interrupt request to "0" by writing "0" into the WIF bit using an interrupt processing routine. The WIF bit is set whenever the specified divide-by output falls regardless of the value of the WIE bit.

**Note:**

To allow interrupt request output (WIE=1) after releasing a reset, clear (WIF=0) the WIF bit at the same time.

If the WIE bit is changed from prohibition to permission (0 --> 1) when the WIF bit is "1", an interrupt request is issued immediately.

If the counter clear (WPCR: WCLR=0) and an overflow of the selected bit occur at the same time, the WIF bit is not set.

## ■ Oscillation Stabilization Wait Time and Watch Interrupts

If an interval time period shorter than the oscillation stabilization wait time of the subclock is set, a watch interrupt request (WPCR: WIF=1) of the watch prescaler is issued when returning from the sub-stop mode following an external interrupt. In this case, prohibit (WPCR: WIE=0) interrupts of the watch prescaler when making a transition to the sub-stop mode.

## ■ Register and Vector Table Related to the Watch Prescaler Interrupts

Table 7.4-1 "Register and Vector Table Related to the Watch Prescaler Interrupts" lists the register and vector table related to the watch prescaler interrupts.

**Table 7.4-1 Register and Vector Table Related to the Watch Prescaler Interrupts**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ8 | ILR3 ($007D_H$) | L81 (bit 1) | L80 (bit 0) | $FFEA_H$ | $FFEB_H$ |

For interrupt operations, see Section 3.4.2 "Interrupt Processing".

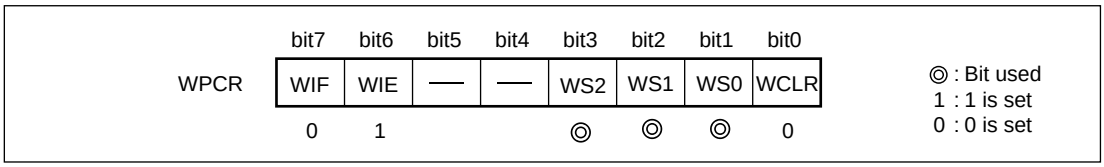
# 7.5 Operation of the Watch Prescaler

**The watch prescaler operates to provide the interval timer function and clock supply function.**

## ■ Operation of the Interval Timer Function (Watch Prescaler)

The setting in Figure 7.5-1 "Setting of the Interval Timer Function" is required for the operation of the interval timer function.

**Figure 7.5-1 Setting of the Interval Timer Function**




The 15-bit counter of the watch prescaler continues to count up the subclock provided the subclock oscillates.

If the counter is cleared (WCLR=0), it starts to count up from "$00000_H$". When "$1FFFF_H$" is reached, counting continues starting from "$00000_H$". When a falling edge is generated in the selected divide-by output for the interval timer, if not in main stop mode, "1" is set to the watch interrupt request flag bit (WIF). That is, starting with the time when cleared, a watch interrupt request is generated at regular intervals of the selected time.

## ■ Operation of the Clock Supply Function

The watch prescaler is also used as a timer to generate the oscillation stabilization wait time of the subclock. Counting of the oscillation stabilization wait time of the subclock ($2^{15}/F_{CL}$, $F_{CL}$: subclock oscillation) starts when the watch prescaler is cleared and ends when the highest bit falls.

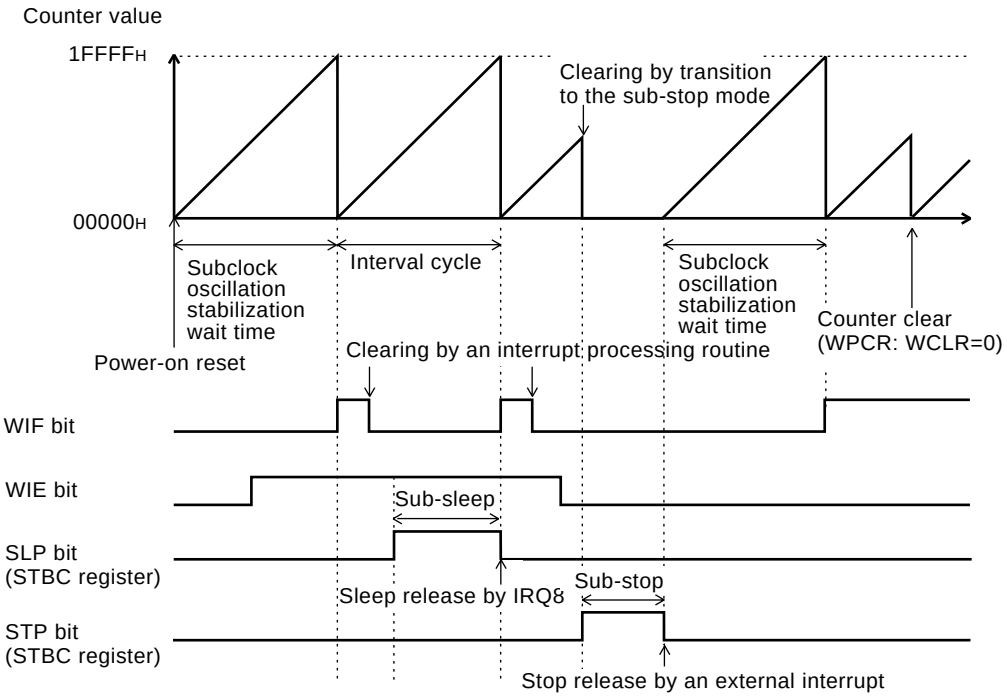
The watch prescaler supplies the clock to the watchdog timer and buzzer output. When the counter of the watch prescaler is cleared, operations of the buzzer output are affected. The counter of the watchdog timer is cleared at the same time if the watch prescaler output is selected (WDTC: CS=1).

## ■ Operations of the Watch Prescaler

Figure 7.5-2 "Operations of the Watch Prescaler" shows the counter values if a transition to the sleep mode or stop mode occurs, or the counter clearing is requested when the interval timer function is operating in subclock mode.

The transition to the watch mode is the same as that to the sub-sleep mode.

**Figure 7.5-2 Operations of the Watch Prescaler**




If "101B" is set to the interrupt interval time select bits (WPCR: WS2, WS1, WS0) of the watch prescaler control register ($2^{17}$/FCL)

# 7.6 Notes on Using the Watch Prescaler

**The following describes the precautions when using the watch prescaler.**
**The watch prescaler cannot be used when a single clock source is specified with the option setting.**

## ■ Notes on Using the Watch Prescaler

### ❍ Precautions when setting the watch prescaler in programs

It is impossible to return from interrupt processing if the interrupt request flag bit (WPCR: WIF) is "1" and the interrupt request enable bit is set (WPCR: WIF=1). The WIF bit must be cleared.

### ❍ Clearing the watch prescaler

The watch prescaler is cleared, in addition to clearing by the watch prescaler clear bit (WPCR: WCLR=0), when the oscillation stabilization wait time of the subclock is required.

If the watch prescaler is selected (WDTC: CS=1) as the count clock of the watchdog timer, the watchdog timer is also cleared when the watch prescaler is cleared.

### ❍ Using the watch prescaler as a timer for the oscillation stabilization wait time

Since the subclock oscillation is stopped when the power is turned on or operating in sub-stop mode, the oscillator takes the oscillation stabilization wait time using the watch prescaler after activating operations.

Do not make a transition from the main clock mode to the subclock mode during oscillation stabilization wait time, such as just after power-on.

The oscillation stabilization wait time of the subclock is fixed.

For details, see Section 3.7.5 "Oscillation Stabilization Wait Time".

### ❍ Precautions when using watch interrupts

In main stop mode, the watch prescaler performs a count operation but a watch interrupt (IRQ8)

does not occur.

### ❍ Precautions when using the peripheral functions that use clocks supplied from the prescaler.

If the counter of the watch prescaler is cleared, the "H" level of the clock supplied by the watch prescaler is short and its "L" level may be longer by a maximum of 1/2 cycle because the output originates from the initial state.

Though the clock for the watchdog timer is also output from the initial state, the watchdog timer works in normal cycles because the counter of the watchdog timer is cleared simultaneously.

# 7.7 Program Example of the Watch Prescaler

**The following shows a program example of the watch prescaler.**

## ■ Program Example of the Watch Prescaler

### ❍ Processing specifications

Generate the watch interrupt of $2^{15}/F_{CL}$ ($F_{CL}$: subclock oscillation) repeatedly. The interval time is about 1 s (for 32.768 kHz operation).

❍ **Coding example**

```
WPCR    .EQU     000BH        ; Address of the watch prescaler
                                control register
WIF     .EQU     WPCR:7       ; Definition of watch interrupt request
                                flag bit
ILR3    .EQU     007DH        ; Address of the interrupt level
                                setting register
          .SECTION INT_V, DATA, LOCATE=0    ; [DATA SEGMENT]
        .ORG     0FFEAH
IRQ8    .DATA.H  WARI         ; Setting interrupt vector
;INT_V  ENDS
;----Main program-------------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1     ; [CODE SEGMENT]
                              ; Stack pointer (SP) and other
                                are assumed to have been initialized
          :
        CLRI                  ; Interrupt disable
        MOV      ILR3,#11111110B  ; Setting interrupt level (level 2)
        MOV      WPCR,#01000110B  ; Clearing interrupt request
                                    flag, enabling interrupt request
                                    output, selecting 2^15/F_CL, and
                                    clearing watch prescaler
        SETI                  ; Interrupt enable
          :
;----Interrupt program--------------------------------------------------
WARI
        CLRB     WIF          ; Clearing interrupt request flag
        PUSHW    A
        XCHW     A,T
        PUSHW    A
          :
         User processing
          :
        POPW    A
        XCHW    A, T
        POPW    A
        RETI
        ENDS
;----------------------------------------------------------------------
        .END
```

# CHAPTER 8 8/16-BIT TIMER/COUNTER

**This chapter describes the functions and operations of the 8/16-bit timer/counter.**


8.1 "Overview of the 8/16-bit Timer/Counter"

8.2 "Configuration of the 8/16-bit Timer/Counter"

8.3 "Pins of the 8/16-bit Timer/Counter"

8.4 "Registers of the 8/16-bit Timer/Counter"

8.5 "8/16-bit Timer/Counter Interrupts"

8.6 "Operation of the Interval Timer Function"

8.7 "Operation of the Counter Function"

8.8 "Operation of the Square Wave Output Initial Setting Function"

8.9 "Operation of Stopping and Restarting the 8/16-bit Timer/Counter"

8.10 "Status of the 8/16-bit Timer/Counter in Each Mode"

8.11 "Notes on Using the 8/16-bit Timer/Counter"

# 8.1 Overview of the 8/16-bit Timer/Counter

**For Timer 1, either the interval timer function or the counter function can be selected. The former function counts up in synchronization with one of the three types of internal count clocks. The latter function counts up according to a clock that is input to the external pin. This output can be used to output square waves of a pre-specified frequency.**
**For Timer 2, only the interval timer function that counts up in synchronization with one of the three types of internal count clock can be selected. This output can be used to output square waves of a pre-specified frequency. Timer 2 is linked to Timer 1 in the 16-bit mode.**

## ■ Interval Timer Function

The interval timer function repeatedly generates an interrupt in pre-specified intervals.

Additionally, it can output square waves of a pre-specified frequency by reversing the output level of the pins at every interval.

- In the 8-bit mode, Timers 1 and 2 operate as two independent timers. Each of the timers can perform the interval timer operation from the count clock cycle to that times $2^8$.
- In the 16-bit mode, Timers 1 and 2 operate as one 16-bit timer with the former as the lower and the latter as the upper. The timer can perform the interval timer operation from the count clock cycle to that times $2^{16}$.
- The count clock can be selected from one of the three types of internal count clocks (Selecting the external clock for Timer 1 selects the operation as the counter function.)
- The output cycle of Timer 1 of the 8/16-bit timer/counter can be used as the consecutive start clock for the A/D converter.

Table 8.1-1 "Channels of the 8/16-bit Timer/Counter and the Pins that Output Square Waves" shows the channels of the 8/16-bit timer/counter and the pins that output square waves using this timer output.

Table 8.1-2 "Timer 1 Interval Time and Square Wave Output Range in the 8-bit Mode" to Table 8.1-4 "Interval Time and Square Wave Output Range in the 16-bit Mode" shows the interval time and square wave output range in each of the operation modes.

**Table 8.1-1 Channels of the 8/16-bit Timer/Counter and the Pins that Output Square Waves**

| Channel | | 8/16-bit timer/counter | |
|---|---|---|---|
| 8-bit timer | | Timer 1 | Timer 2 |
| Output pin | 8-bit mode | T01 | T02 |
| | 16-bit mode | T01 | |

**Table 8.1-2 Timer 1 Interval Time and Square Wave Output Range in the 8-bit Mode**

| Count clock cycle | | Interval time | Square wave output range (Hz) |
|---|---|---|---|
| Internal count clock | $2t_{inst}$ | $2t_{inst}$ to $2^9t_{inst}$ | $1/(2^2t_{inst})$ to $1/(2^{10}t_{inst})$ |
| | $32t_{inst}$ | $2^5t_{inst}$ to $2^{13}t_{inst}$ | $1/(2^6t_{inst})$ to $1/(2^{14}t_{inst})$ |
| | $512t_{inst}$ | $2^9t_{inst}$ to $2^{17}t_{inst}$ | $1/(2^{10}t_{inst})$ to $1/(2^{18}t_{inst})$ |
| External clock | $1t_{ext}$ | $1t_{ext}$ to $2^8t_{ext}$ | $1/(2t_{ext})$ to $1/(2^9t_{ext})$ |

**Table 8.1-3 Timer 2 Interval Time and Square Wave Output Range in the 8-bit Mode**

| Count clock cycle | | Interval time |
|---|---|---|
| Internal count clock | $2t_{inst}$ | $2t_{inst}$ to $2^9t_{inst}$ |
| | $32t_{inst}$ | $2^5t_{inst}$ to $2^{13}t_{inst}$ |
| | $512t_{inst}$ | $2^9t_{inst}$ to $2^{17}t_{inst}$ |

**Table 8.1-4 Interval Time and Square Wave Output Range in the 16-bit Mode**

| Count clock cycle | | Interval time | Square wave output range (Hz) |
|---|---|---|---|
| Internal count clock | $2t_{inst}$ | $2t_{inst}$ to $2^{17}t_{inst}$ | $1/(2^2t_{inst})$ to $1/(2^{18}t_{inst})$ |
| | $32t_{inst}$ | $2^5t_{inst}$ to $2^{21}t_{inst}$ | $1/(2^6t_{inst})$ to $1/(2^{22}t_{inst})$ |
| | $512t_{inst}$ | $2^9t_{inst}$ to $2^{25}t_{inst}$ | $1/(2^{10}t_{inst})$ to $1/(2^{26}t_{inst})$ |
| External clock | $1t_{ext}$ | $1t_{ext}$ to $2^{16}t_{ext}$ | $1/(2t_{ext})$ to $1/(2^{17}t_{ext})$ |

$t_{inst}$: Instruction cycle (influenced by the clock mode, etc.)
$t_{ext}$: External clock cycle (1 $t_{ext}$ greater than or equal to 2 $t_{inst}$)

❍ **Example of calculating the interval time and the square wave frequency**

Assuming the original oscillation of the main clock ($F_{CH}$) as 10 MHz and the Timer 1 data register (T1DR) value as "DDH (221)", calculate as follows the Timer 1 interval time as well as the frequency of square waves that are output from the T01 pin when Timer 1 continuously operates without changing this T1DR register value:

Note that the system clock control register (SYCC) is used to select the fastest clock (CS1, CS0 = $11_B$, 1 instruction cycle = $4/F_{CH}$) in the main clock mode (SCS=1).

```
Interval time = (2 x 4/FCH x (T1DR register value + 1)
              = (8/10 MHz) x (221 + 1)
              nearly equal to 177.6 µs
Output frequency = FCH / (2 x 8 x (T1DR register value + 1))
                 = 10 MHz / (16 x 221 + 1))
                 nearly equal to 2.815 kHz
```

## ■ Counter Function

The counter function counts how many times it detects the falling edge of the external clock that is input to the external pin. For the 8/16-bit counter, the EC pin is the external clock input pin. Since the external clock can be selected only for Timer 1, the counter function operates for Timer 1 in the 8-bit mode or for the 16-bit mode.

- The counter counts up according to the external clock and, if the counter value is equal to the setting value, generates an interrupt request and reverses the output level of the square wave output pin.
- In the 8-bit mode, Timer 1 can count up to $2^8$.
- In the 16-bit mode, the timer can count up to $2^{16}$.
- The counter can be used in the same way as for the interval timer function if an external clock with a fixed cycle is input.

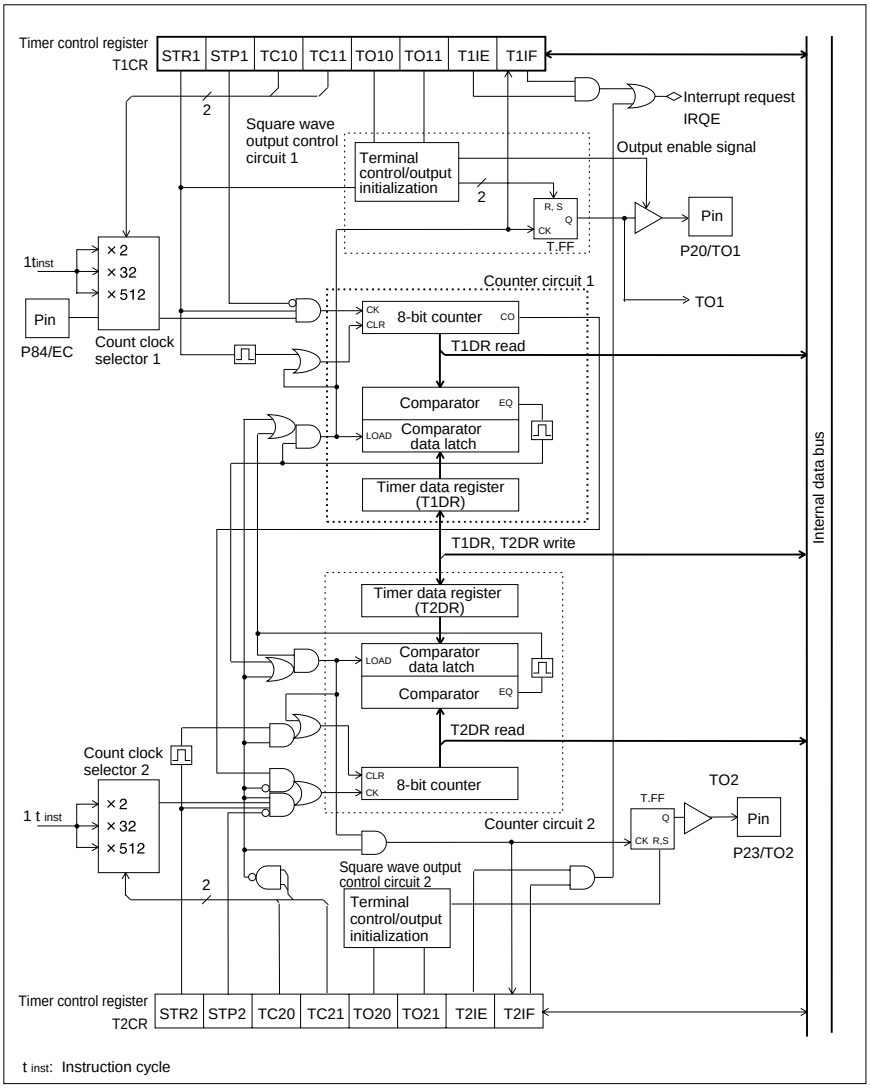
# 8.2 Configuration of the 8/16-bit Timer/Counter

**The 8/16-bit timer/counter consists of the following five blocks:**

- **Count clock selectors 1 and 2**
- **Counter circuits 1 and 2**
- **Square wave output control circuit**
- **Timer data registers (T1DR, T2DR)**
- **Timer control registers (T1CR, T2CR)**

## ■ Block Diagram of the 8/16-bit Timer/Counter

**Figure 8.2-1 Block Diagram of the 8/16-bit Timer/Counter**



❍ **Count clock selectors 1 and 2**

The count clock selectors 1 and 2 select the internal count clock. For Timer 1 in the 8-bit mode or for the 16-bit mode, three types of internal clocks and one type of external clock can be selected. For Timer 2 in the 8-bit mode, only three types of internal clocks can be selected.

❍ **Counter circuits 1 and 2**

Each of the counter circuits 1 and 2 consists of the 8-bit counter, comparator, comparator data latch, and data registers (T1DR, T2DR).

The 8-bit counter counts up according to the selected count clock. The comparator compares the counter value and the comparator data latch value and, if there is a match, clears the counter and sets (loads) the data register value in the comparator data latch.

In the 8-bit mode, the counter circuits 1 and 2 operate independently as Timers 1 and 2. In the 16-bit mode, the counter circuits 1 and 2 operate as one 16-bit counter with the former as the lower 8-bit and the latter as the upper 8-bit.

❍ **Square wave output control circuit**

If the comparator detects a match in the 8-bit or 16-bit mode, this circuit generates an interrupt request. If the square wave output is enabled at this time, a corresponding output control circuit reverses the output of the square wave output pin.

Additionally, the square wave output can be initialized either to the "L" or "H" level.

❍ **Timer data registers (T1DR, T2DR)**

While writing, these registers are used to set data to be compared with the values in the 8-bit counters. While reading, the current counter values in the counters are read.

❍ **Timer control registers (T1CR, T2CR)**

Select a function, enable and disable the operation, control an interrupt, and check the status.

❍ **Interrupt of the 8/16-bit timer/counter**

**IRQE:**

Generates an IRQE interrupt request if the interrupt request output is enabled (T1CR: T1IE=1 for Timer 1 in the 8-bit mode or for the 16-bit mode or T2CR: T2IE=1 for Timer 2 in the 8-bit mode) when the counter value becomes equal to the value in the data register either in the interval timer or the counter function mode.

# 8.3 Pins of the 8/16-bit Timer/Counter

**This section describes the pins related to the 8/16-bit timer/counter and shows the block diagram of the pins.**

## ■ Pins Related to the 8/16-bit Timer/Counter

The pins related to the 8/16-bit timer/counter are P84/EC, P20/T01, and P23/T02.

### ❍ P84/EC pin

The P84/EC pin functions to operate as the general-purpose input port (P84), the 8/16-bit timer external clock input pin (EC), and the 16-bit timer input pin.

**EC:**

Counts the clock that is input to this pin if an external clock input is selected (T1CR:TC11, TC10=11) for Timer 1 in the 8-bit mode or for the 16-bit mode.

### ❍ P20/T01 and P23/T02 pins

The P20/T01 and P23/T02 pins function to operate as the general-purpose input port (P20, 23) and the square wave output pin for the timer (T01, 02).

**T01:**

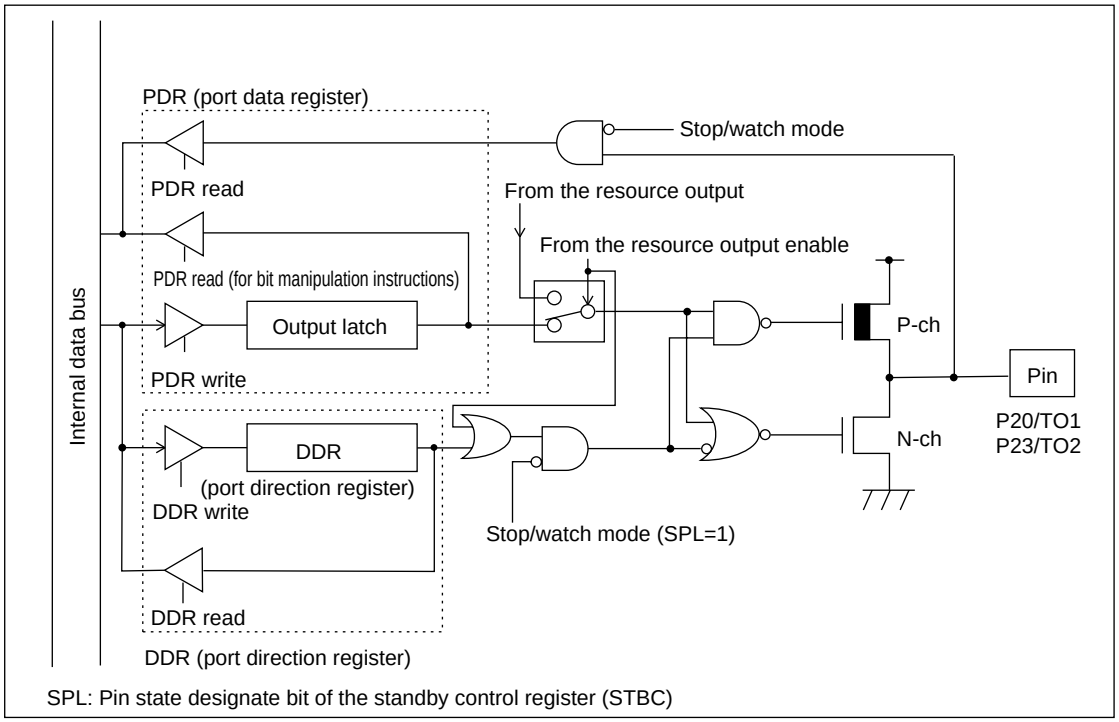
Outputs square waves from this pin for Timer 1 in the 8-bit mode or for the 16-bit mode. The P20/T01 pin automatically becomes the output pin regardless of the output latch value and operates as the T01 pin if the square wave output is enabled (T2CR: T011, T010=$00_B$).

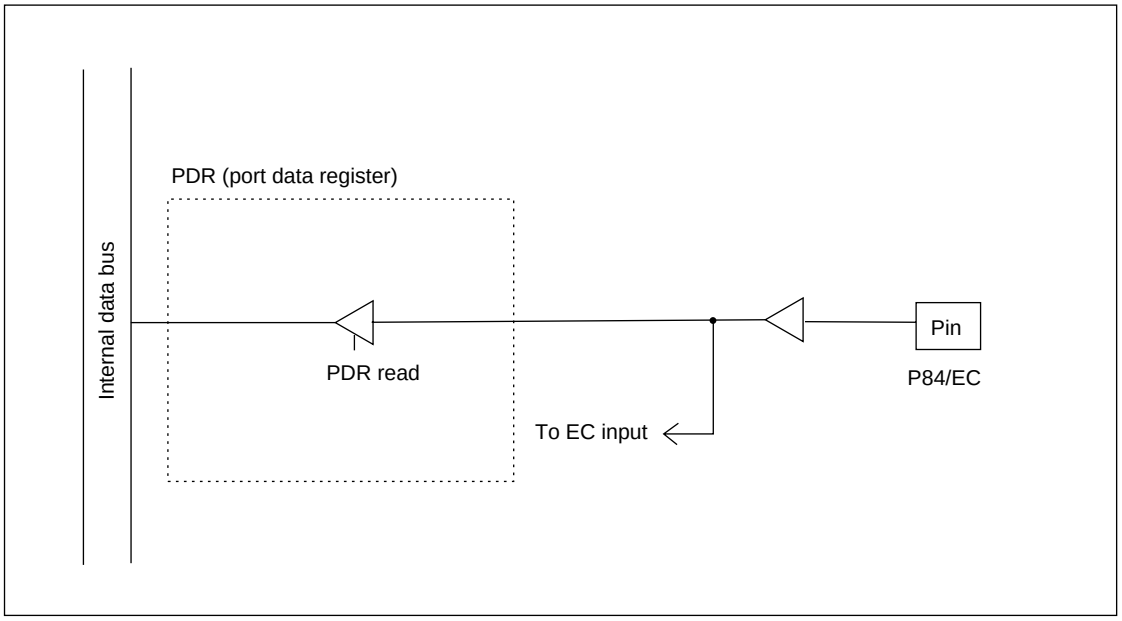
**T02:**

Outputs square waves from this pin for Timer 2 in the 8-bit mode. The P23/T02 pin automatically becomes the output pin regardless of the output latch value and operates as the T02 pin if the square wave output is enabled (T2CR: T021, T020=$00_B$).

## ■ Block Diagram of the Pins Related to the 8/16-bit Timer/Counter

**Figure 8.3-1 Block Diagram of the Pins Related to the 8/16-bit Timer/Counter**



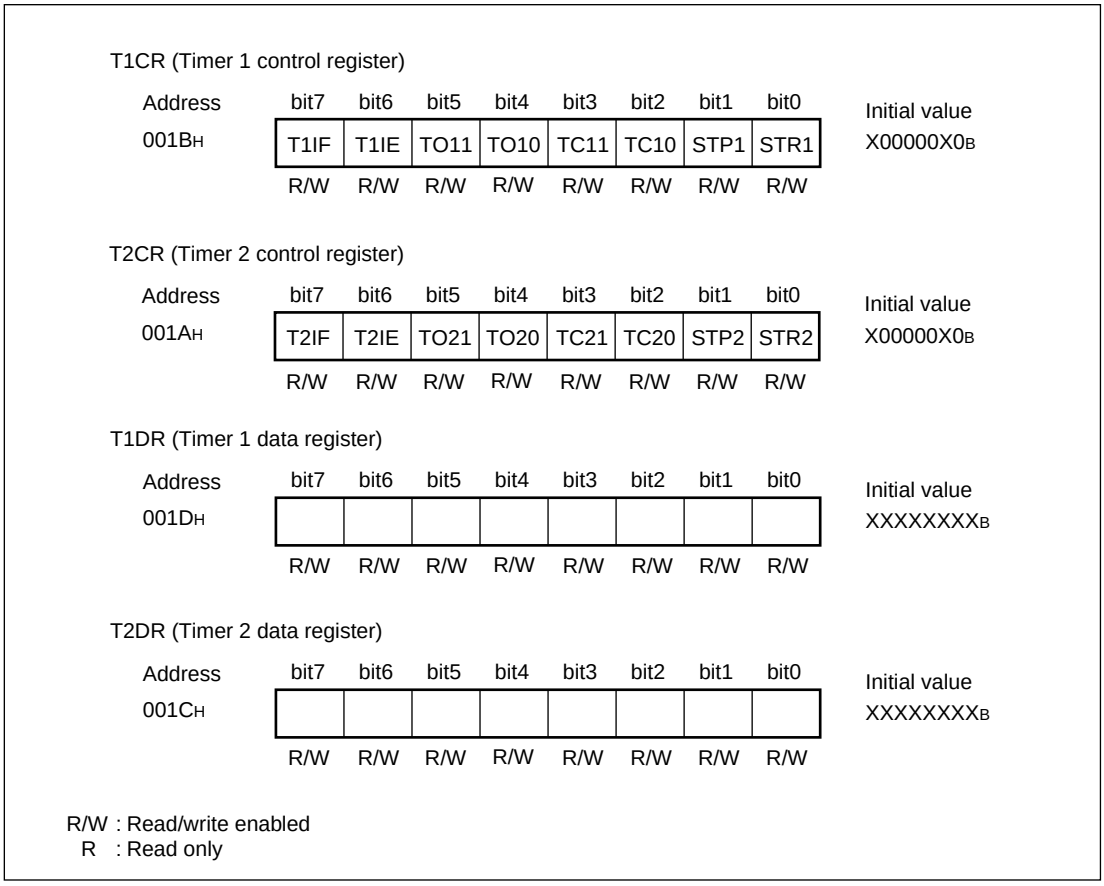



# 8.4 Registers of the 8/16-bit Timer/Counter

**This section describes the registers related to the 8/16-bit timer/counter.**

## ■ Registers Related to the 8/16-bit Timer/Counter

**Figure 8.4-1 Registers Related to the 8/16-bit Timer/Counter**



## 8.4.1 Timer 1 Control Register (T1CR)

**The Timer 1 control register (T1CR) selects a function, enables or disables an operation, controls an interrupt, and checks the status of Timer 1 in the 8-bit mode or of the 16-bit mode of the 8/16-bit timer/counter. To use only Timer 1 in the 8-bit mode, you must also initialize the Timer 2 control register (T2CR).**

■ **Timer 1 Control Register (T1CR)**

**Figure 8.4-2 Timer 1 Control Register (T1CR)**

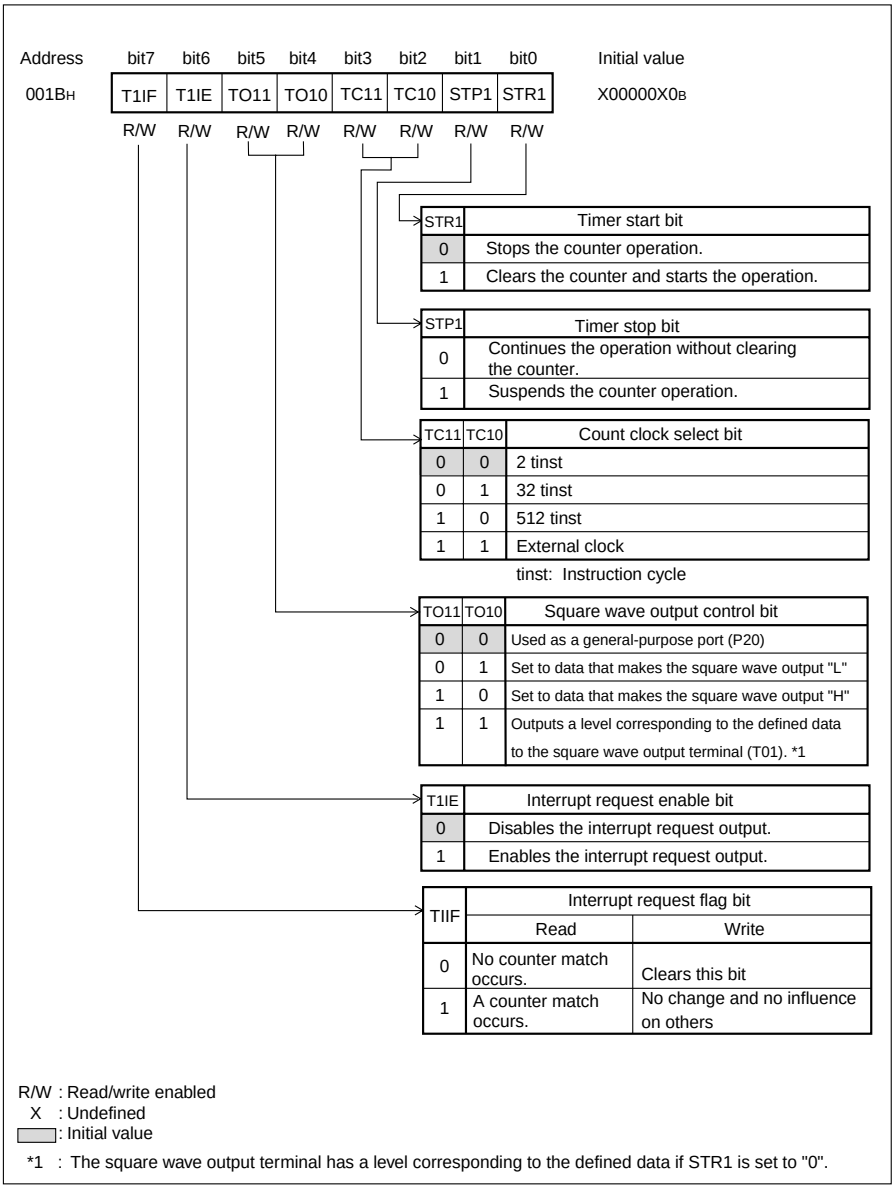


**Table 8.4-1 Functions of the Bits in the Timer 1 Control Register (T1CR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | T1IF:<br>Interrupt request flag bit | • In the 8-bit mode<br>Set to "1" if Timer 1 has the counter value that matches the Timer 1 data register (T1DR) setting value (comparator data latch).<br>• In the 16-bit mode<br>Set to "1" if Timers 1 and 2 have counter values that match the T1DR and T2DR setting values, respectively.<br>Setting this bit and the interrupt request enable bit (T1IE) to "1" outputs an interrupt request.<br>Cleared to "0" while writing. Setting this bit to "1" does not affect it and causes no change. |
| Bit 6 | T1IE:<br>Interrupt request enable bit | • Enables or disables the interrupt request output to the CPU.<br>Setting this bit and the interrupt request flag bit (T1IF) to "1" outputs an interrupt request. |
| Bit 5<br>Bit 4 | T011, T010:<br>Square wave output control bits | • Setting these bits to "$00_B$" makes the T20/T01 pin a general-purpose port (P20). Setting these bits to any other value makes it a square wave output pin (T01).<br>• Writing "$01_B$" or "$10_B$" in these bits sets the initialization data in the square wave output control circuit but does not output it to the T01 pin.<br>• If these bits are set to "$11_B$" and the timer is stopped (STR1=0), the T01 pin is set to a level corresponding to the initialization data. |
| Bit 3<br>Bit 2 | TC11, TC10:<br>Clock source select bit | • Selects the count clock to be supplied to the counter.<br>• Selects one of the three internal clocks and one external clock.<br>Setting these bits to "$11_B$" selects the external clock input and the operation as the counter function.<br>**Note:**<br>Selecting the external clock input (TC11, TC100=$11_{B)}$ uses the input of the P84/EC pin as the clock. |
| Bit 1 | STP1:<br>Timer stop bit | • Suspends the counter.<br>Writing "1" in this bit suspends the counter operation. While the timer is started (STR1=1), writing "0" causes the counter to continue the operation. |
| Bit 0 | STR1:<br>Timer start bit | • Starts or stops the counter.<br>• Changing this bit from "0" to "1" clears the counter. If the timer operation is continued (STP1=0) at this time, the counter starts the operation and counts up using the selected count clock. Writing "0" in this bit stops the counter operation.<br>• In the 16-bit mode, starting the timer (STP1=0 --> 1) clears Timer 1 and 2 counters. |

**Note:**

To use only Timer 1 of the 8/16-bit timer/counter in the 8-bit mode and to prevent a malfunction, set a value other than "11B" in the timer count clock select bits (T2CR: TC21, TC20) in the Timer 2 control register.

## 8.4.2 Timer 2 Control Register (T2CR)

**The Timer 2 control register (T2CR) selects a function, enables or disables an operation, controls an interrupt, and checks the status for Timer 2 in the 8-bit mode of the 8/16-bit timer/counter. To use Timer 2 in the 16-bit mode, the T2CR register must also be set although the Timer 1 control register (T1CR) is used to control Timer 2.**

### ■ Timer 2 Control Register (T2CR)

**Figure 8.4-3 Timer 2 Control Register (T2CR)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 001AH | T2IF | T2IE | TO21 | TO20 | TC21 | TC20 | STP2 | STR2 | X00000X0B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

| STR2 | Timer start bit |
|---|---|
| 0 | Stops the counter operation. |
| 1 | Clears the counter and starts the operation. |

| STP2 | Timer stop bit |
|---|---|
| 0 | Continues the operation without clearing the counter. |
| 1 | Suspends the counter operation. |

| TC21 | TC20 | Count clock select bit |
|---|---|---|
| 0 | 0 | 2 tinst |
| 0 | 1 | 32 tinst |
| 1 | 0 | 512 tinst |
| 1 | 1 | 16-bit mode |

tinst: Instruction cycle

| TO21 | TO20 | Square wave output control bit |
|---|---|---|
| 0 | 0 | Used as a general-purpose port (P23) |
| 0 | 1 | Set to data that makes the square wave output "L" |
| 1 | 0 | Set to data that makes the square wave output "H" |
| 1 | 1 | Outputs a level corresponding to the defined data to the square wave output terminal (T02). *1 |

| T1IE | Interrupt request enable bit |
|---|---|
| 0 | Disables the interrupt request output. |
| 1 | Enables the interrupt request output. |

| TIIF | Interrupt request flag bit | |
|---|---|---|
| | Read | Write |
| 0 | No counter match occurs. | Clears this bit |
| 1 | A counter match occurs. | No change and no influence on others |

R/W : Read/write enabled
X : Undefined
: Initial value

*1 : The square wave output terminal has a level corresponding to the defined data if STR2 is set to "0".

**Table 8.4-2 Functions of the Bits in the Timer 2 Control Register (T2CR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | T2IF:<br>Interrupt request flag bit | Set to "1" if Timer 2 has a counter value matching the Timer 2 data register (T2DR) setting value (comparator data latch).<br>• Setting this bit and the interrupt request enable bit (T2IE) to "1" outputs an interrupt request.<br>• Cleared to "0" while writing. Setting this bit to "1" does not affect it and causes no change.<br>**Note:**<br>In the 16-bit mode, the T1IF bit becomes valid and the T2IF bit becomes irrelevant to the operation. |
| Bit 6 | T2IE:<br>Interrupt request enable bit | • Enables or disables the interrupt request output to the CPU.<br>• Setting this bit and the interrupt request flag bit (T2IF) to "1" outputs an interrupt request.<br>**Note:**<br>In the 16-bit mode, the T1IE bit becomes valid and the T2IE bit becomes irrelevant to the operation. |
| Bit 5<br>Bit 4 | T021, T020:<br>Square wave output control bits | Setting these bits to "$00_B$" makes the T23/T02 pin a general-purpose port (P23). Setting these bits to any other value makes it a square wave output pin (T02). If the HCLK output is enabled at the same time, the square wave output (T02) is prioritized.<br>• Writing "$01_B$" or "$10_B$" in these bits sets the initialization data in the square wave output control circuit but does not output it to the T02 pin.<br>• If these bits are set to "$11_B$" and the timer is stopped (STR1=0), the T02 pin is set to a level corresponding to the initialization data. |
| Bit 3<br>Bit 2 | TC21, TC20:<br>Clock source select bit | • Selects the count clock to be supplied to the counter.<br>• Selects one of the three internal clocks.<br>• Setting these bits to "$11_B$" selects the 16-bit mode.<br>**Note:**<br>In the 16-bit mode, the TC11 and TC10 bits becomes valid and the TC21 and TC20 bits only select the 16-bit mode. |
| Bit 1 | STP2:<br>Timer stop bit | • Suspends the counter.<br>• Writing "1" in this bit suspends the counter operation. While the timer is started (STR2=1), writing "0" causes the counter to continue the operation.<br>**Note:**<br>In the 16-bit mode, the STP1 bit becomes valid and the STP2 bit becomes irrelevant to the operation. |

**Table 8.4-2 Functions of the Bits in the Timer 2 Control Register (T2CR) (Continued)**

| Bit name | | Function |
|---|---|---|
| Bit 0 | STR2:<br>Timer start bit | • Starts or stops the counter.<br>• Changing this bit from "0" to "1" clears the counter. If the timer operation is continued (STP2=0) at this time, the counter starts the operation and counts up using the selected count clock. Writing "0" in this bit stops the counter operation.<br>**Note:**<br>In the 16-bit mode, the STR1 bit becomes valid and the STR2 bit becomes irrelevant to the operation. |

**Note:**

To use the operation in the 16-bit mode, write "$11_B$" in the TC21 and TC20 bits and then perform control using the T1CR register.

## 8.4.3 Timer 1 Data Register (T1DR)

**The Timer 1 data register (T1DR) is used to set an interval timer value (in the interval timer function mode) or a counter value (in the counter function mode) for Timer 1 in the 8-bit mode or for the lower 8 bits in the 16-bit mode of the 8/16-bit timer/counter as well as read the counter value.**

### ■ Timer 1 Data Register (T1DR)

The value in this register is compared with the counter value. Reading this register reads the current counter value. The value in this register cannot be read.

Figure 8.4-4 "Timer 1 Data Register (T1DR)" shows the bit configuration of the Timer 1 data register.

**Figure 8.4-4 Timer 1 Data Register (T1DR)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 001D$_H$ | | | | | | | | | XXXXXXXX$_B$ |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

#### ❍ In the 8-bit mode (Timer 1)

The value in this register is compared with the counter value. This register is set to an interval time value in the interval timer function mode or a count value to be detected in the counter function mode. Enabling the count operation (T1CR: STR1=0 --> 1, STP1=0) sets (loads) the T1DR register value in the comparator data latch and starts the count-up.

If the comparator data latch value and the counter value match due to the count-up, the T1DR register value is reset in the comparator data latch, the counter is cleared, and the count operation is continued.

The comparator data latch is reset if a match is detected. Thus, a value written in the T1DR register while the counter operates becomes valid in the next cycle (after a match is detected) and thereafter.

You can calculate the value in the T1DR register in the interval timer mode as follows. Note that the instruction cycle is influenced by the clock mode or the gear function.

T1DR register value = Interval time / (Count clock cycle x Instruction cycle) -1

❍ **In the 16-bit mode**

The value in this register is compared with the lower 8 bits of the counter value of the 16-bit timer.

This register is set to the lower 8 bits of the interval time in the interval timer function mode or the lower 8 bits of a count value to be detected in the counter function mode. The T1DR register value is loaded to the lower 8 bits of the comparator data latch when the count operation is started or when a match with the 16-bit counter value is detected. A value written in the T1DR register while the 16-bit counter operates becomes valid when a match is detected.

For the setting values in the T1DR register in the interval timer function mode, see Section 8.4.4 "Timer 2 data register (T2DR)".

## 8.4.4 Timer 2 Data Register (T2DR)

**The Timer 2 data register (T2DR) is used to set an interval timer value (in the interval timer function mode) or a counter value (in the counter function mode) for Timer 2 in the 8-bit mode or for the upper 8 bits in the 16-bit mode of the 8/16-bit timer/counter and is also used to read the counter value.**

### ■ Timer 2 Data Register (T2DR)

The value in this register is compared with the counter value. Reading this register reads the current counter value. The value in this register cannot be read.

Figure 8.4-5 "Timer 2 Data Register (T2DR)" shows the bit configuration of the Timer 2 data register.

**Figure 8.4-5 Timer 2 Data Register (T2DR)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 001C$_H$ | | | | | | | | | XXXXXXXX$_B$ |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

❍ In the 8-bit mode (Timer 2)

The value in this register is compared with the Timer 2 counter value. This register is set to an interval time value in the interval timer function mode or a count value to be detected in the counter function mode. The T2DR register is reset (loaded) to the comparator data latch when the counter operation is started or a match with the counter value is detected.

A value written in the T2DR register while the counter operates becomes valid in the next cycle (after a match is detected) and thereafter.

The T2DR register in the interval timer mode can be calculated as follows. Note that the instruction cycle is affected by the clock mode or the gear function.

T2DR register value = Interval time / (Count clock cycle x Instruction cycle) -1

❍ **In the 16-bit mode**

The value in this register is compared with the upper 8 bits of the counter value of the 16-bit timer.

This register is set to the upper 8 bits of interval time in the interval timer function mode or the upper 8 bits of a count value to be detected in the counter function mode. The T2DR register is loaded to the upper 8 bits of the comparator data latch when the count operation is started or when a match with the 16-bit counter value is detected. A value written in the T2DR register while the 16-bit counter operates becomes valid when a match is detected. The count operation is controlled in the 16-bit mode using the Timer 1 control register (T1CR).

You can calculate the values in the T1DR and T2DR registers in the interval timer mode as follows. Note that the instruction cycle is influenced by the clock mode or the gear function.

16-bit data value = Interval time / (Count clock cycle x Instruction cycle) -1

The upper and lower 8 bits of the 16-bit data value are set to the T2DR and T1DR registers, respectively.

# 8.5 8/16-bit Timer/Counter Interrupts

**An interrupt of the 8/16-bit timer/counter is caused by a match of the data register setting value and the count value both in the interval timer and counter operation modes.**
**The 8/16-bit timer/counter generates an IRQE as an interrupt request.**

## ■ 8/16-bit Timer/Counter Interrupts

Table 8.5-1 "Interrupt Request Flag Bit and Interrupt Cause of the 8/16-bit Timer/Counter" shows the relationship between an interrupt request flag bit, an interrupt request output enable bit, and an interrupt cause of the 8/16-bit timer/counter.

**Table 8.5-1 Interrupt Request Flag Bit and Interrupt Cause of the 8/16-bit Timer/Counter**

| | 8-bit mode | | 16-bit mode |
|---|---|---|---|
| | Timer 1 | Timer 2 | Timers 1+2 |
| Interrupt request flag bit | T1CR:T1IF | T2CR:T2IF | T1CR:T1IF |
| Interrupt request enable bit | T1CR:T1IE | T2CR:T2IE | T1CR:T1IE |
| Interrupt cause | A match of the T1DR setting value and the 8-bit counter value | A match of the T2DR setting value and the 8-bit counter value | A match of the T1DR and T2DR setting value and the 16-bit counter value |

An interrupt request of the 8/16-bit timer/counter is generated by Timers 1 and 2 independently in the 8-bit mode and by Timer 1 in the 16-bit mode. However, the basic operation is the same in both modes. This section describes the interrupt operation of Timer 1 in the 8-bit mode.

### ❍ Interrupt operation of Timer 1 in the 8-bit mode

If the counter counts up from "$00_H$" according to the selected count clock and matches the setting value in the comparator data latch corresponding to the timer data register (T1DR), the interrupt request flag bit (T1CR: T1IF) is set to "1".

If the interrupt request flag bit is set to Enabled (T1CR: T1IF=1) at this time, an interrupt request (IRQE) occurs in the CPU. Use the interrupt processing routine to write "0" in the T1IF bit and clear the interrupt request.

The T1IF bit is set to "1" regardless of the value in the T1IE bit if the counter value and the setting value match.

Since Timers 1 and 2 operate independently in the 8-bit mode and generate the same interrupt request (IRQE), the software may have to evaluate the interrupt request flag bit.

**Note:**

If the counter value and the T1DR register value match and the counter is stopped (T1CR: STR1=0) at the same time, the T1IF bit is not set.

While the T1IF bit is "1", setting the T1IF bit from Disabled to Enabled (0 --> 1) immediately causes an interrupt request.

## ■ Register Related to the Interrupts of the 8/16-bit Timer/Counter and the Vector Table

**Table 8.5-2 Register Related to the Interrupts of the 8/16-bit Timer/Counter and the Vector Table**

| | Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|---|
| | | Register | Bit to be set | | Upper | Lower |
| 8/16-bit timer/counter | IRQE | ILR4 ($007E_H$) | LE1 (bit 5) | LE0 (bit 4) | $FFDE_H$ | $FFDF_H$ |

For the interrupt operation, see Section 3.4.2 "Interrupt Processing during".

# 8.6 Operation of the Interval Timer Function

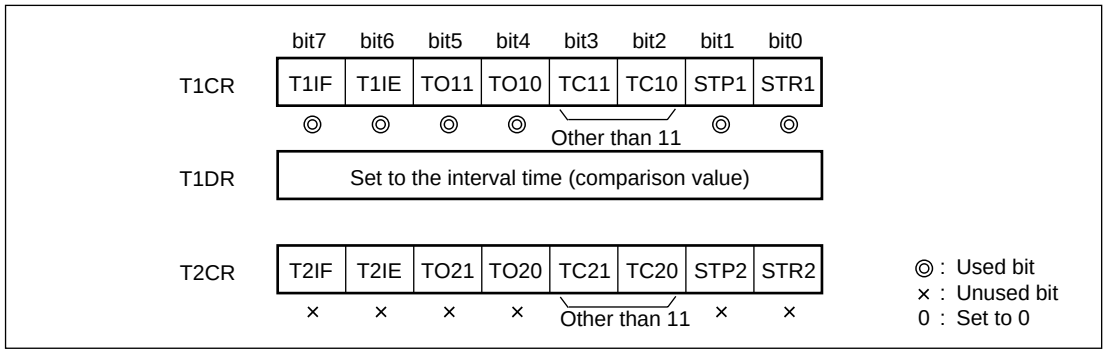
**This section describes the operation of the interval timer function of the 8/16-bit timer/counter.**

## ■ Operation of the Interval Timer Function

### ❍ In the 8-bit mode

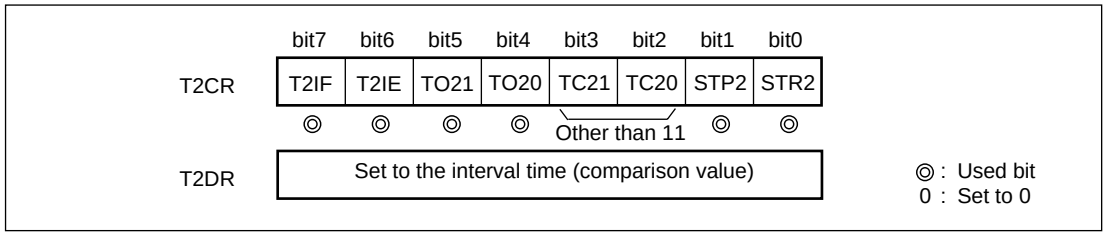
To use Timer 1 in the 8-bit mode as the interval timer function, the setting shown in Figure 8.6-1 "Setting the Interval Timer Function (Timer 1)" must be made.

**Figure 8.6-1 Setting the Interval Timer Function (Timer 1)**




To use Timer 2 in the 8-bit mode as the interval timer function, the setting shown in Figure 8.6-2 "Setting the Interval Timer Function (Timer 2)" must be made.

**Figure 8.6-2 Setting the Interval Timer Function (Timer 2)**

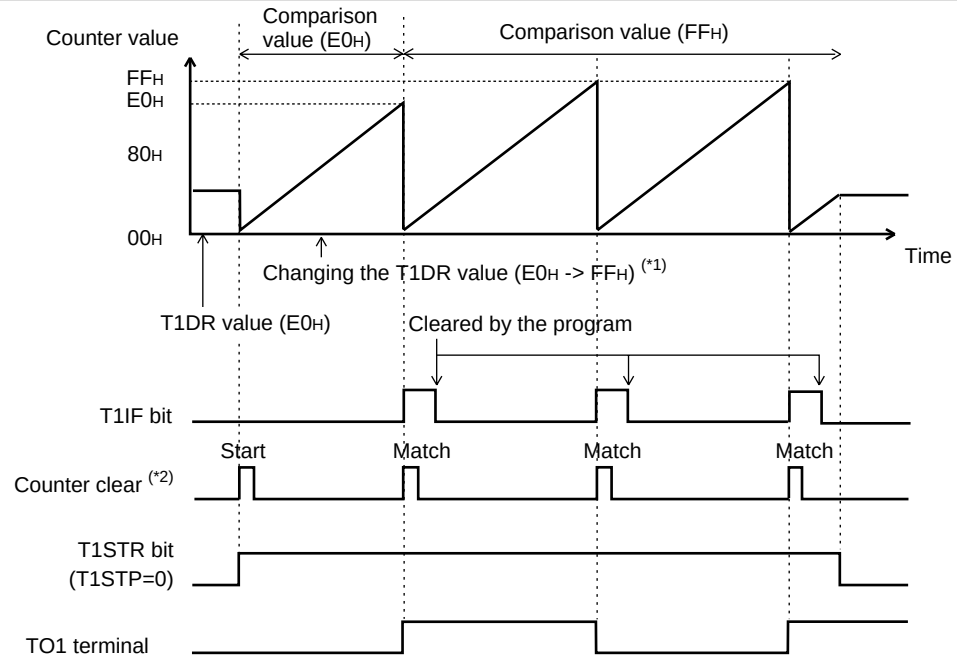



If the counter is started in the 8-bit mode, the counter starts counting up from "$00_H$" at every rising edge of the selected clock. If the counter value matches the value in the data register (comparator data latch), the interrupt request flag bit (T1CR: T1IF or T2IF) is set to "1" and the counter starts counting from "$00_H$". If the output of the square wave output control circuit is reversed and the square wave output is enabled (T011, T010, and T021, T020=Other than 00B), a corresponding timer output pin outputs square waves.

Timers 1 and 2 correspond to the T01 and T02 pins, respectively.

Figure 8.6-3 "Operation of the Interval Timer Function in the 8-bit Mode (Timer 1)" shows the operation of the interval timer function in the 8-bit mode.

**Figure 8.6-3 Operation of the Interval Timer Function in the 8-bit Mode (Timer 1)**

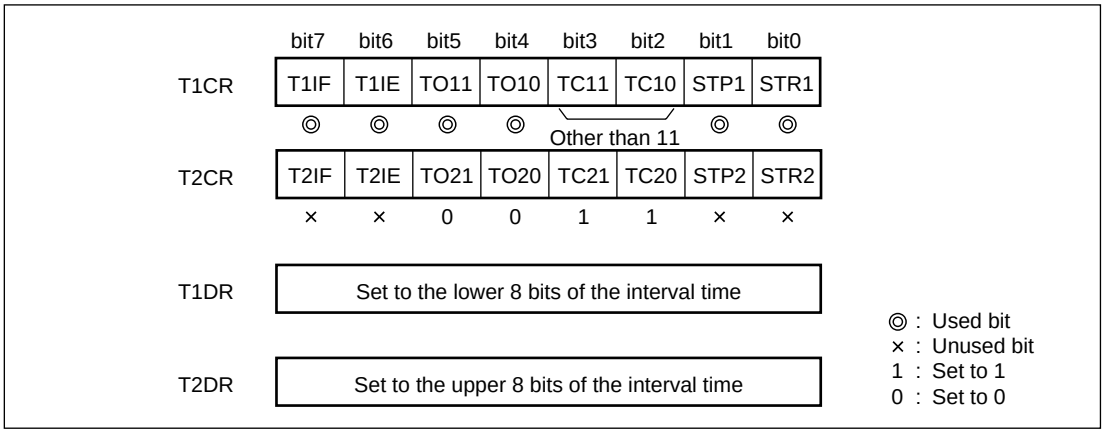



*1: A value rewritten in the data register during the counter operation is valid in the next cycle and thereafter.
*2: When the counter is started and when a match is detected, the counter is cleared and the data register setting value is loaded to the comparator data latch.

❍ **In the 16-bit mode**

To use the interval timer function in the 16-bit mode, the setting shown in Figure 8.6-4 "Setting the Interval Timer Function (in the 16-bit Mode)" must be made.

**Figure 8.6-4 Setting the Interval Timer Function (in the 16-bit Mode)**




Although the Timer 1 control register (T1CR) is used to control the timer in the 16-bit mode, the Timer 2 control register (T2CR) must be initialized. The data register is set to a 16-bit value with the T2DR and T1DR registers as upper and lower 8 bits, which is then compared with the 16-bit counter value. When the counter is cleared, all the 16 bits are cleared at the same time. Other operations in the 16-bit mode are the same as Timer 1 in the 8-bit mode.

# 8.7 Operation of the Counter Function

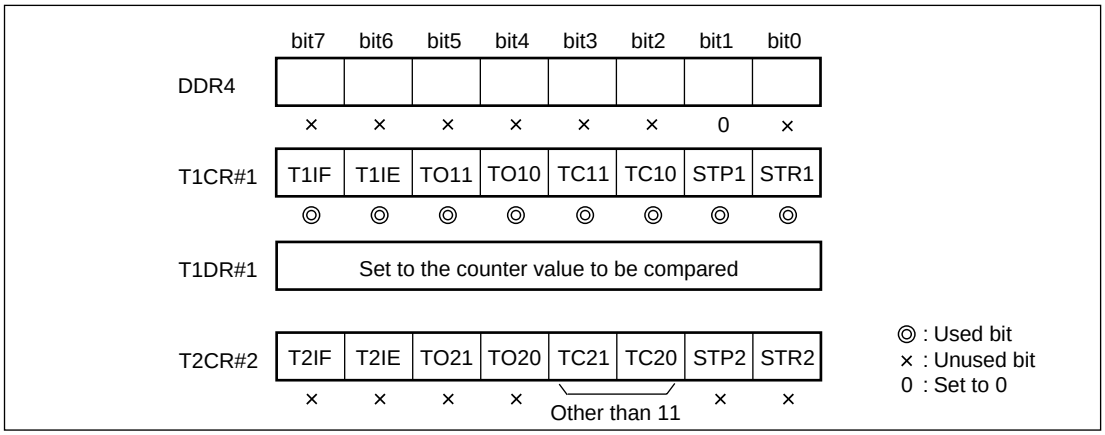
**This section describes the operation of the counter function of the 8/16-bit timer/counter.**

## ■ Operation of the Counter Function

### ❍ In the 8-bit mode

To use Timer 1 in the 8-bit mode as the counter function, the setting shown in Figure 8.7-1 "Setting the Counter Function (in the 8-bit Mode)" must be made.

**Figure 8.7-1 Setting the Counter Function (in the 8-bit Mode)**

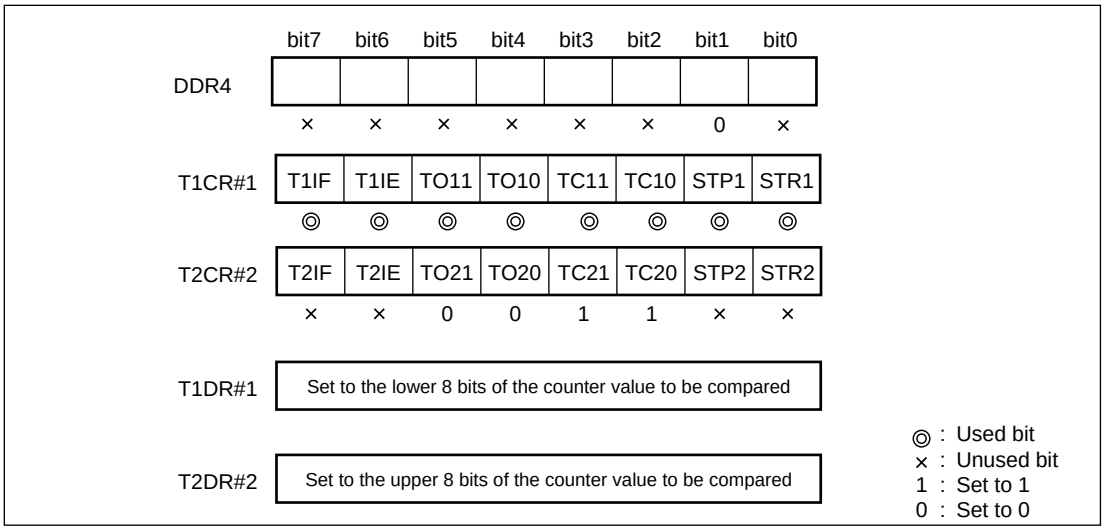



The counter function in the 8-bit mode operates in the same way as the interval timer function (Timer 1 in the 8-bit mode) except that the external clock is used instead of the internal clock.

### ❍ In the 16-bit mode

To use the counter function in the 16-bit mode, the setting shown in Figure 8.7-2 "Setting the Counter Function (in the 16-bit Mode)" must be made.

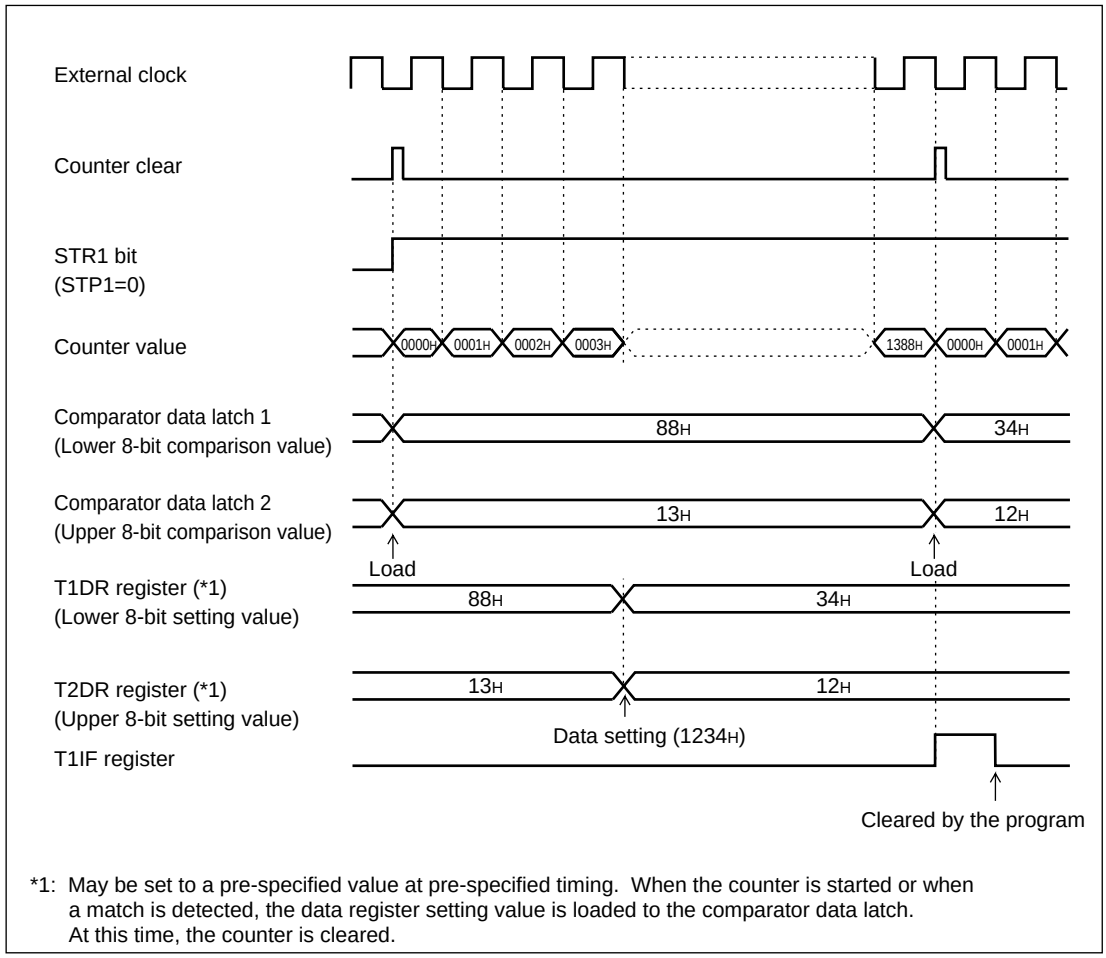
**Figure 8.7-2 Setting the Counter Function (in the 16-bit Mode)**




The counter function in the 16-bit mode operates in the same way as an interval timer function (in the 16-bit mode) except that the external clock is used instead of the internal clock.

Figure 8.7-3 "Operation of the Counter Function in the 16-bit Mode" shows the operation of the counter function in the 16-bit mode.

**Figure 8.7-3 Operation of the Counter Function in the 16-bit Mode**




*1: May be set to a pre-specified value at pre-specified timing. When the counter is started or when a match is detected, the data register setting value is loaded to the comparator data latch. At this time, the counter is cleared.

**Note:**

When reading a value in the operating counter in the 16-bit mode, always read it twice and check whether it is an appropriate value before using it.

# 8.8 Operation of the Square Wave Output Initial Setting Function

**The square wave output can be set to a pre-specified initial value using the Timer 1 control registers (T1CR, T1CR, T2CR, T2CR).**

## ■ Operation of the Square Wave Output Initial Setting Function

Although this section describes only the operation of Timer 1, the operation of Timer 2 is the same.

The square wave output can be set to a pre-specified initial value using the program only if the timer operation is stopped (T1CR: STR1=0).

Figure 8.8-1 "Square Wave Output Initial Setting Equivalent Circuit" shows the initial setting equivalent circuit in the square wave output control circuit. Make the initial setting as shown in Table 8.8-1 "Making the Initial Setting of the Square Wave Output (T1CR Register)". The square wave output operation at this time is shown in Figure 8.8-2 "Initial Setting Operation of the Square Wave Output".

**Figure 8.8-1 Square Wave Output Initial Setting Equivalent Circuit**

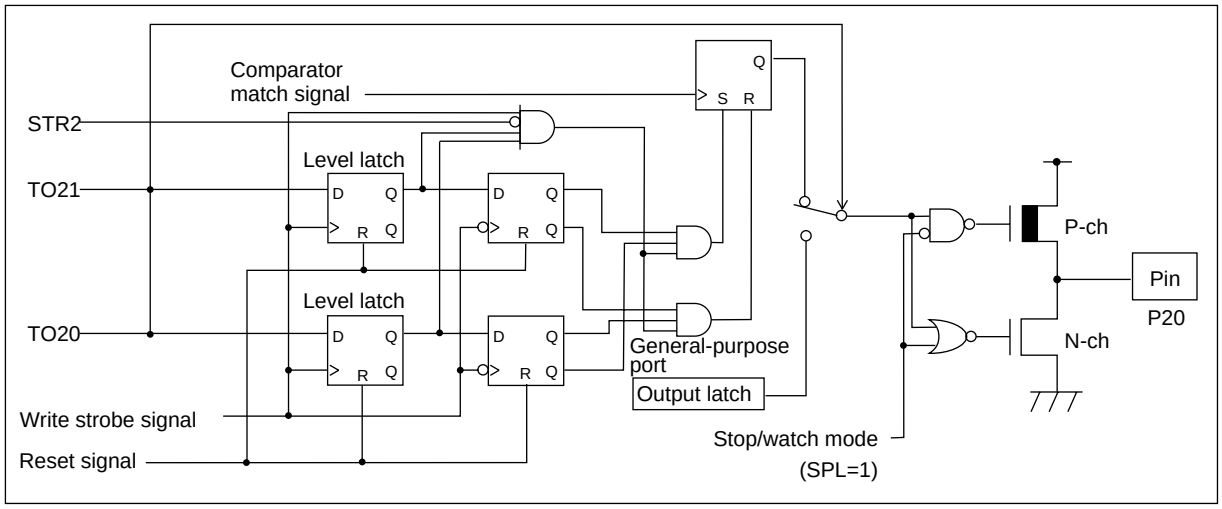


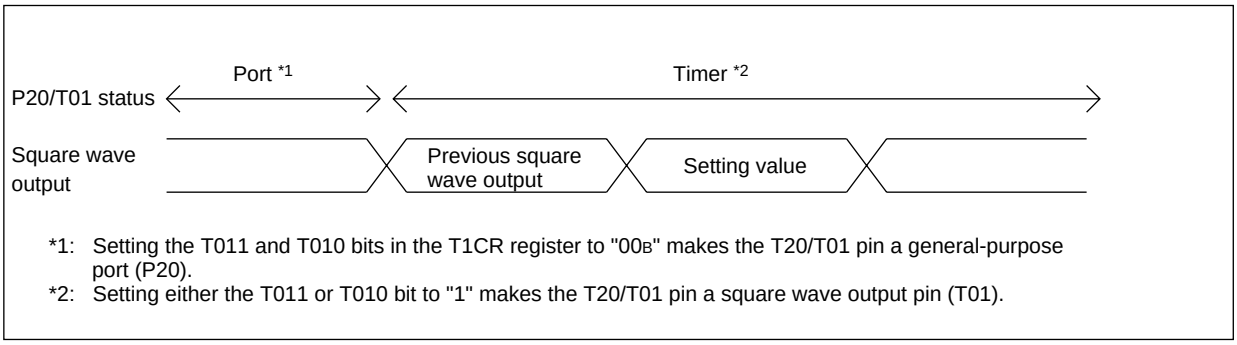

**Table 8.8-1 Making the Initial Setting of the Square Wave Output (T1CR Register)**

| Procedure | Setting and operation |
|---|---|
| (1) | To set the square wave output pin (T01) to the "L" level, write "$01_B$" in the square wave output control bits (T1CR: T011, T010) and then write "$11_B$". In addition, to set it to the "H" level, write "$10_B$" and "$11_B$" in this order.<br>**Note:**<br>Until "$11_B$" is written, only the level latch stores the value and the T01 pin has the current or previous level. |

**Table 8.8-1 Making the Initial Setting of the Square Wave Output (T1CR Register)**

| Procedure | Setting and operation |
|---|---|
| (2) | Writing "$11_B$" in the square wave output control bits (T011, T010) to stop the timer operation (STR1=0) outputs a level (initial value) corresponding to a value in the level latch to the T01 pin. |
| (3) | Setting the timer start bit (STR1=1) starts the counter operation. |
| (4) | The square wave output is reversed when the counter value and the data register setting value match. |

**Figure 8.8-2 Initial Setting Operation of the Square Wave Output**




*1: Setting the T011 and T010 bits in the T1CR register to "$00_B$" makes the T20/T01 pin a general-purpose port (P20).
*2: Setting either the T011 or T010 bit to "1" makes the T20/T01 pin a square wave output pin (T01).

# 8.9 Operation of Stopping and Restarting the 8/16-bit Timer/ Counter

**This section describes the operation of stopping and restarting the 8/16-bit timer/ counter.**

## ■ Operation of Stopping and Restarting the 8/16-bit Timer/Counter

Although this section describes only the operation of Timer 1, the operation of Timer 2 is the same.

The timer stop bit (STP1) and the timer start bit (STR1) in the Timer 1 control register are used to stop and restart Timer 1.

To clear the counter and start the count operation, set the STP1 and STR1 bits to "$01_B$" when the STR1 bit is "0". The timer is cleared and the count operation is started at a rising edge of the STR1 bit.

Suspending the timer and restarting the count operation without clearing the counter: To suspend the count operation, set the STP1 and STR1 bits to "$11_B$". To restart the count operation without clearing the counter, set the STP1 and STR1 bits to "$01_B$".

Table 8.9-1 "Stopping and Restarting the Timer" shows timer statuses depending on the STP1 and STR1 bits and the timer operation when it is started (STP1, STR1=01B) in each of these statuses

**Table 8.9-1 Stopping and Restarting the Timer**

| STP1 (STP2) | STR1 (STR2) | Timer status | Timer operation when it is started (STP1, STR1=$01_B$) in the status shown on the left |
|---|---|---|---|
| 0 | 0 | Count operation stopped | Clears the counter and starts the count operation. |
| 0 | 1 | Count operation in progress | Continues the count operation. |
| 1 | 0 | Count operation stopped | Clears the counter and starts the count operation. |
| 1 | 1 | Count operation suspended | Restarts the count operation without clearing the counter |

# 8.10 Status of the 8/16-bit Timer/Counter in Each Mode

**This section describes the operations of switching to the sleep and stop modes and receiving a suspend request during the operation of the 8/16-bit timer/counter.**

## ■ Operation in the Subclock and Standby Modes and when the Counter is Suspended

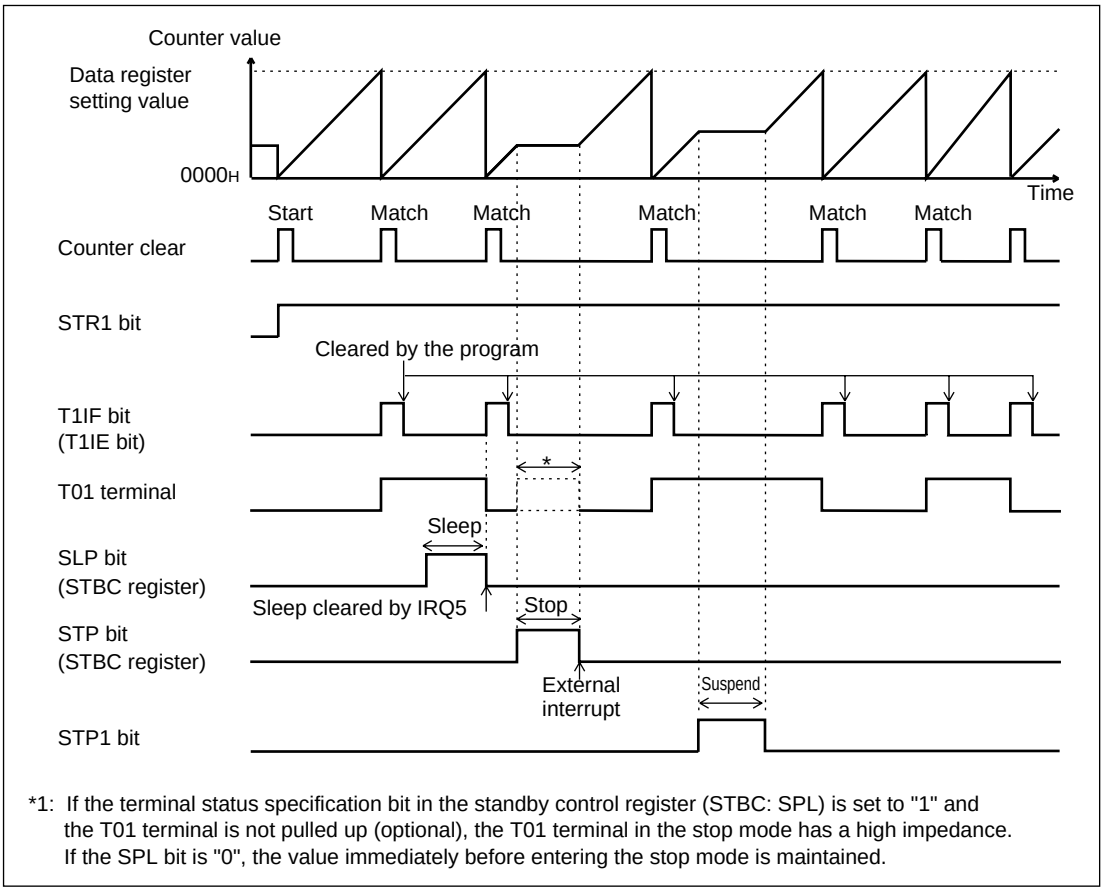
Figure 8.10-1 "Operation in the Subclock and Standby Modes and when the Counter is Suspended" shows the counter value statuses upon switching to the sleep and stop modes and receiving a suspend request while the interval timer or counter function is operating.

In the stop mode, the counter stops, maintaining the value. If the stop mode is cleared by an external interrupt, the counter starts counting from the maintained value. Therefore, the initial interval time and the external clock count cannot be correct values. After the stop mode is cleared, initialize the 8/16-bit timer/counter again.

The operations of switching to and clearing the watch mode (STBC: TMD=1) is the same as the operation of switching to and clearing the stop mode. The watch mode is cleared by a watch interrupt and an external interrupt.

If the counter is suspended (T1STP=1), the counter stops, maintaining the value. If the operation is continued (T1STP=0), the count operation is restarted.

**Figure 8.10-1 Operation in the Subclock and Standby Modes and when the Counter is Suspended**




*1: If the terminal status specification bit in the standby control register (STBC: SPL) is set to "1" and the T01 terminal is not pulled up (optional), the T01 terminal in the stop mode has a high impedance. If the SPL bit is "0", the value immediately before entering the stop mode is maintained.

# 8.11 Notes on Using the 8/16-bit Timer/Counter

**This section describes the precautions when using the 8/16-bit timer/counter.**

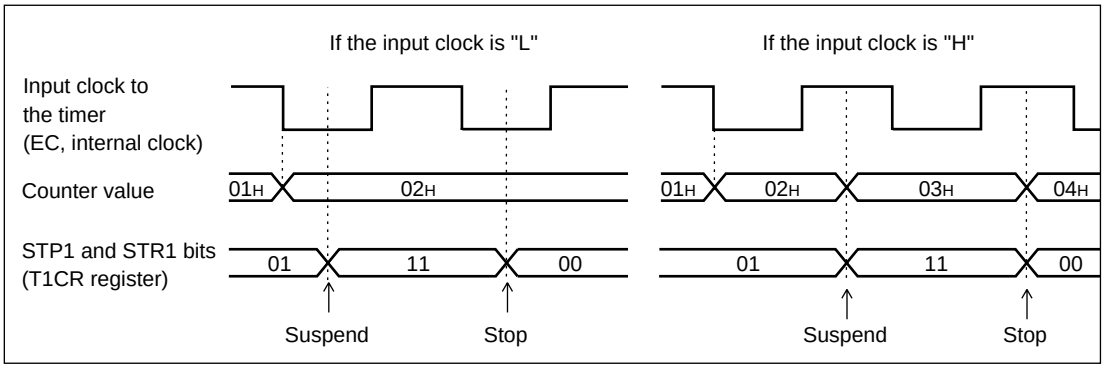
## ■ Notes on Using the 8/16-bit Timer/Counter

### ❍ Precautions when stopping the timer

Although this section describes only the operation of Timer 1, the operation of Timer 2 is the same.

If the STP1 bit is used to suspend the timer and the input clock is the "H" level, the counter value is incremented by "1". If, after the timer is suspended, "$00_B$" is written into the STP1 and STR1 bits at the same time and the input clock is the "L" level, the counter value is sometimes incremented by "1". If the STP1 bit is used to suspend the timer, read the counter value and then write "0" in the STR1 bit.

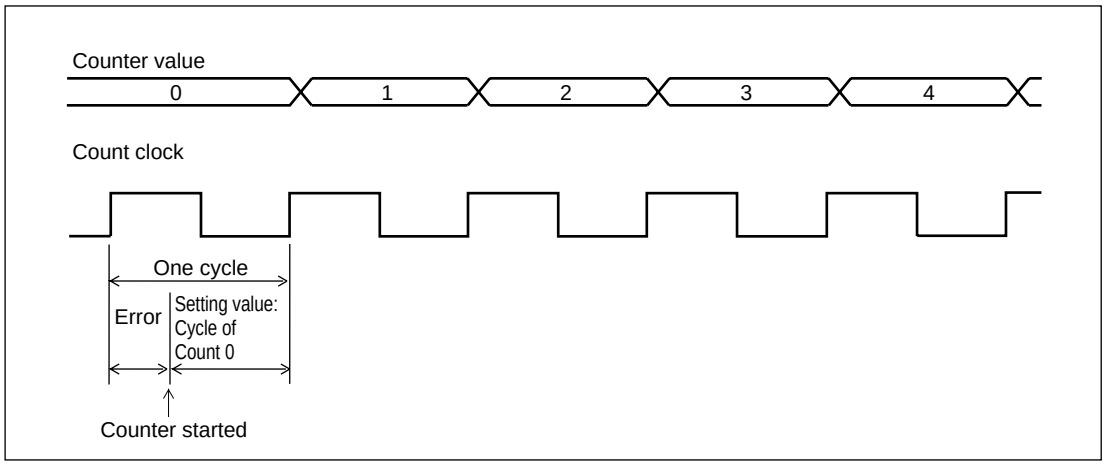
**Figure 8.11-1 Operation when the Timer Stop Bit is Used**




### ❍ Error

The counter start by the program and the count-up start by the selected count clock are asynchronous. Thus, an error of one instruction cycle shorter at the maximum may exist in the time elapsing until the count value and the setting data match. Figure 8.11-2 "Error Until the Count Operation is Started" shows an error until the count operation is started.

**Figure 8.11-2 Error Until the Count Operation is Started**




❍ **Using one 8-bit channel**

To use only Timer 1 of the 8/16-bit timer/counter in the 8-bit mode and to prevent a malfunction, set a value other than "$11_B$" in the timer count clock select bits (T2CR: TC21, TC20) in the Timer 2 control register.

❍ **Precautions when making a setting in a program**

- To use the 8/16-bit timer/counter in the 16-bit mode, set "$11_B$" in the count clock select bits in the Timer 2 control register (T2CR: TC21, TC20) and "$00_B$" in the unused bits of Bits 5 and 4 (T2CR: T021, T020).
- When reading a value in the operating counter in the 16-bit mode, read it twice and check whether it is an appropriate value before using.
- Initializing the square wave output while the timer is operating (T1CR: STR1=1) does not change the output value. The output is initialized while the timer operation is stopped.
- If the interrupt request flag bits (T1CR: T1IF, T2CR: T2IF) are "1" and the interrupt request enable bits (T1CR: T1IE=1, T2CR: T2IE=1) are enabled, a recovery from an interrupt cannot be made. Always clear the interrupt request flag bit.
- If the timer start bits are used to stop the counter operation (T1CR: STR1=0, T2CR: STR2=0) and an interrupt cause occurs at the same time, the interrupt request flag bits (T1CR: T1IF, T2CR: T2IF) are not set.

❍ **Writing a value to the data register in the 16-bit mode**

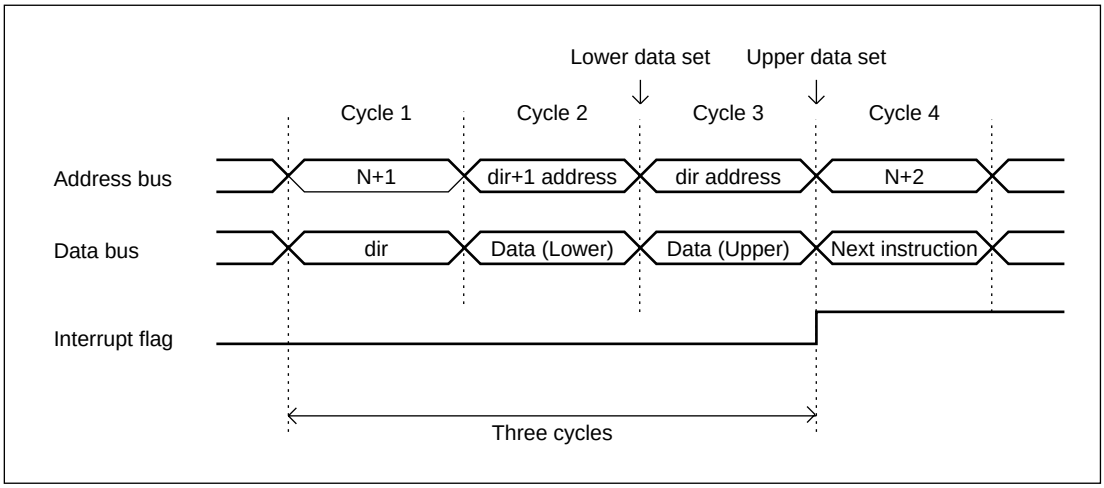
If a word write instruction is used to write a value to the T1DR and T2DR registers in the 16-bit mode, the CPU sets the data by splitting it into upper and lower 8 bits as shown in Table 8.11-1 "Word Data Transfer Operation", "Word data transfer operation". Thus, if an interrupt request occurs while executing a word write instruction, only the lower 8-bit data will be valid and the upper 8-bit data may not be valid until the next interrupt request is made.

To change the interval time during the timer operation, write data to the timer data registers (T1DR, T2DR) three cycles before an interrupt request occurs (see Figure 8.11-3 "Internal Bus Operation when the "MOVW dir, A" Instruction is Executed").

**Table 8.11-1 Word Data Transfer Operation**

| Instruction | Cycle count | Address bus | Data bus | RD | WR |
|---|---|---|---|---|---|
| MOVW A,dir | 1 | N+1 | dir | 0 | 1 |
| | 2 | Dir address | Data (Upper) | 0 | 1 |
| | 3 | Dir+1 address | Data (Lower) | 0 | 1 |
| | 4 | N+2 | Next instruction | 0 | 1 |
| MOVW dir,A | 1 | N+1 | dir | 0 | 1 |
| | 2 | Dir+1 address | Data (Lower) | 1 | 0 |
| | 3 | Dir address | Data (Upper) | 1 | 0 |
| | 4 | N+2 | Next instruction | 0 | 1 |

**Figure 8.11-3 Internal Bus Operation when the "MOVW dir, A" Instruction is Executed**



# CHAPTER 9 16-BIT TIMER/COUNTER

**This chapter describes the functions and operations of the 16-bit timer/counter.**


9.1 "Overview of the 16-bit Timer/Counter"

9.2 "Configuration of the16-bit Timer/Counter"

9.3 "Pin of the 16-bit Timer/Counter"

9.4 "Registers of the 16-bit Timer/Counter"

9.5 "16-bit Timer/Counter Interrupts"

9.6 "Operation of the Interval Timer Function"

9.7 "Operation of the Counter Function"

9.8 "Status of the 16-bit Timer/Counter in Each Mode"

9.9 "Notes on Using the 16-bit Timer/Counter"

9.10 "Program Example of the 16-bit Timer/Counter"

# 9.1 Overview of the 16-bit Timer/Counter

**The 16-bit timer/counter has an interval timer function and a counter function. The interval timer function counts up in synchronization with the internal count clock (the oscillator frequency divided into four cycles). The counter function counts up by detecting a prespecified edge of a pulse that is input to the external pin. One of these functions can be selected.**

## ■ Interval Timer Function

The interval timer function generates an interrupt at prespecified intervals.

The 16-bit counter counts up from a setting value in synchronization with the internal count clock that divides the oscillator frequency into four cycles and generates an interrupt if the counter value overflows.

- Interval timer operation up to an internal count clock times $2^{16}$ is possible.
- Use the interrupt processing routine to reset the interval time to generate an interrupt repeatedly.

Table 9.1-1 "Range for Interval Time" shows the range for the interval time.

**Table 9.1-1 Range for Interval Time**

| Internal count clock cycle | Interval time |
|---|---|
| $1t_{ins}$ | $1t_{inst}$ to $2^{16}t_{inst}$ |

$t_{inst}$: Instruction cycle (the oscillator frequency divided into four cycles)

The following shows an example of calculating the interval time.

Assuming the oscillator frequency (Fc) to be 10 MHz and the timer counter register (TCR) to be $0000_H$, calculate the interval time as follows:

Interval time = (4 / Fc) x ($2^{16}$ - TCR register value) = (4 / 10 MHz) x 65536
early equal to 26.214 ms

## ■ Counter Function

The counter function detects the edge of a pulse that is input to the external pin (EC pin) and counts it.

- Counts up every time a prespecified edge of the external input is detected and generates an interrupt if the counter value overflows.
- Can detect the pulse width of the external input at a minimum of two instruction cycles.
- Can be set to detect both rising and falling edges.

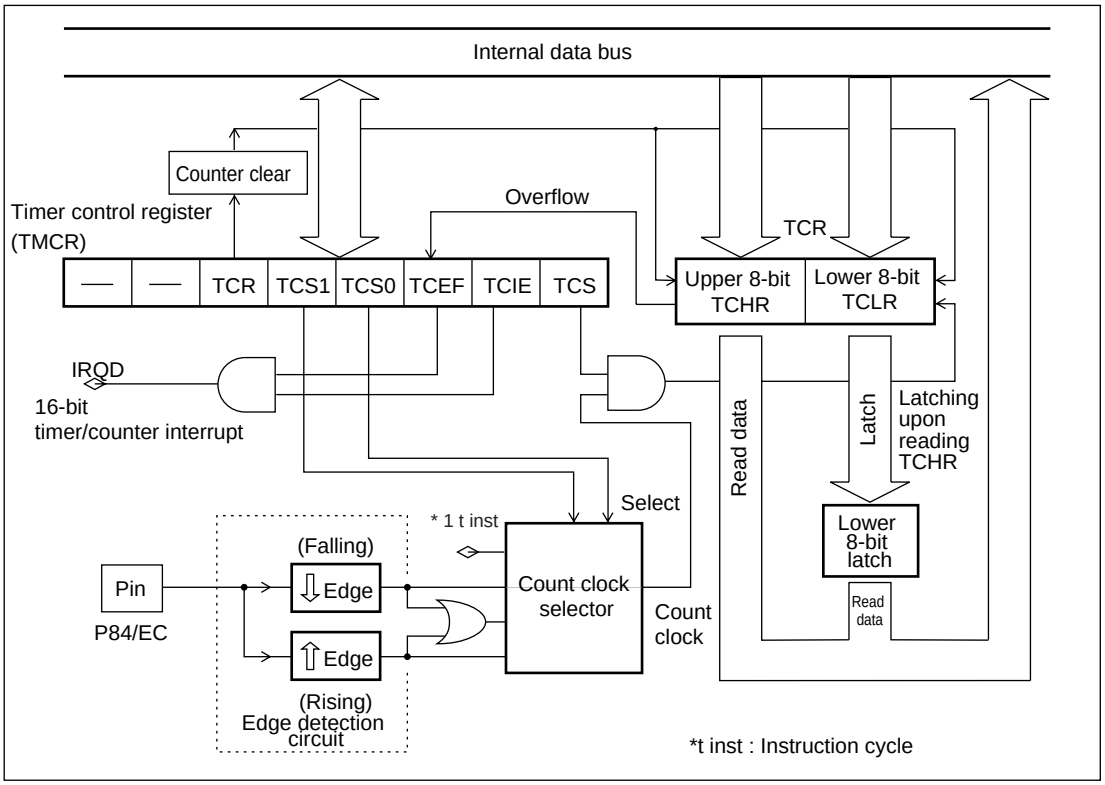
# 9.2 Configuration of the 16-bit Timer/Counter

**The 16-bit timer/counter consists of the following five blocks:**

- **Count clock selector**
- **Edge detection circuit**
- **Timer count register (TCR)**
- **Timer control register (TMCR)**
- **Lower 8-bit latch**

## ■ Block Diagram of the 16-bit Timer/Counter

**Figure 9.2-1 Block Diagram of the 16-bit Timer/Counter Section**




❍ **Count clock selector**

Selects the internal count clock (1 $t_{inst}$) in interval timer function mode. Selects the output of the edge detection circuit in counter function mode. The selected signal will be used as the clock by the 16-bit counter (TCR register) to count up.

❍ **Edge detection circuit**

Operates in counter function mode and detects the rising and/or falling edges of a pulse that is input from the EC3 pin.

- **Timer count register (TCR)**

  Stores a value from which the 16-bit counter counts up. If the counter value overflows, the TMCR register interrupt request flag bit is set (TCEF=1).

- **Timer control register (TMCR)**

  Selects a function, enables and disables operation, controls interrupts, and checks the status.

- **Lower 8-bit latch**

  Stores the lower 8 bits of the 16-bit counter when the TCR register upper 8-bit value (TCHR) is read. Since the lower 8-bit value of the counter (TCLR) is read from this lower 8-bit latch, a correct 16-bit counter value can be read even while the counter is counting up.

  If, while the counter is operating, the lower 8-bit value is read before the upper 8-bit value, a correct value may not be read, depending on the counter carry.

  Therefore, always use a word transfer instruction to read the TCR register.

- **Interrupt related to the timer/counter**

  **IRQD:**

  Generates an interrupt request if interrupt request output is enabled (TMCR: TCIE = 1) when the counter value overflows either in interval timer or counter function mode.

# 9.3 Pin of the 16-bit Timer/Counter

**This section describes the pin related to the 16-bit timer/counter and shows a block diagram.**

## ■ Pin Related to the 16-bit Timer/Counter

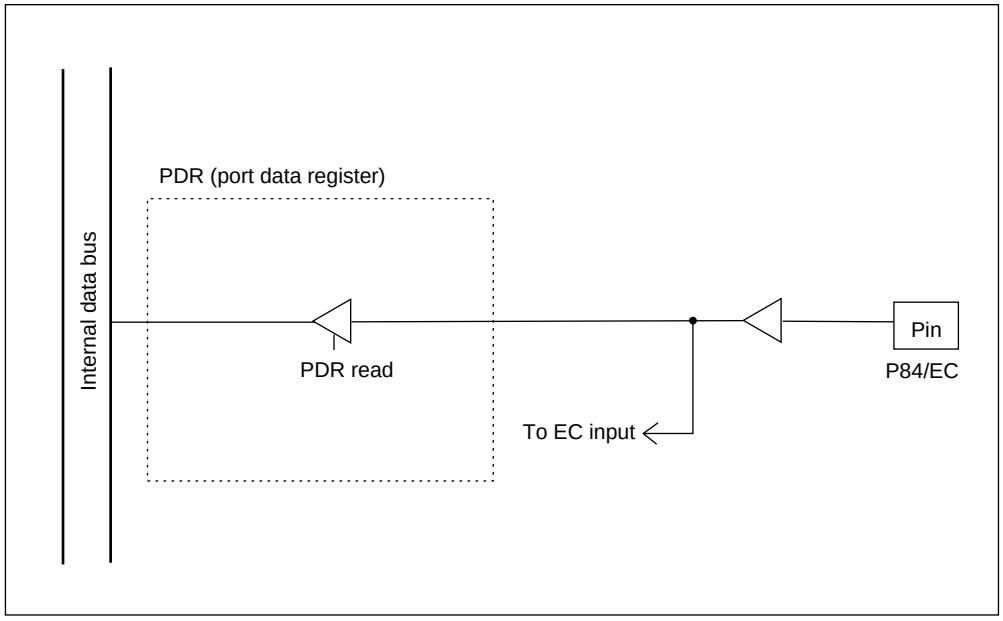
The pin related to the 16-bit timer/counter is the P84/EC pin. It functions both as a general-purpose input port (P84) and as an external pulse input pin for the counter (EC).

**EC:**

Counts a prespecified edge of a pulse that is input to this pin in counter function mode.

## ■ Block Diagram of the Pin Related to the 16-bit Timer/Counter

**Figure 9.3-1 Block Diagram of the Pin Related to the 16-bit Timer/Counter**

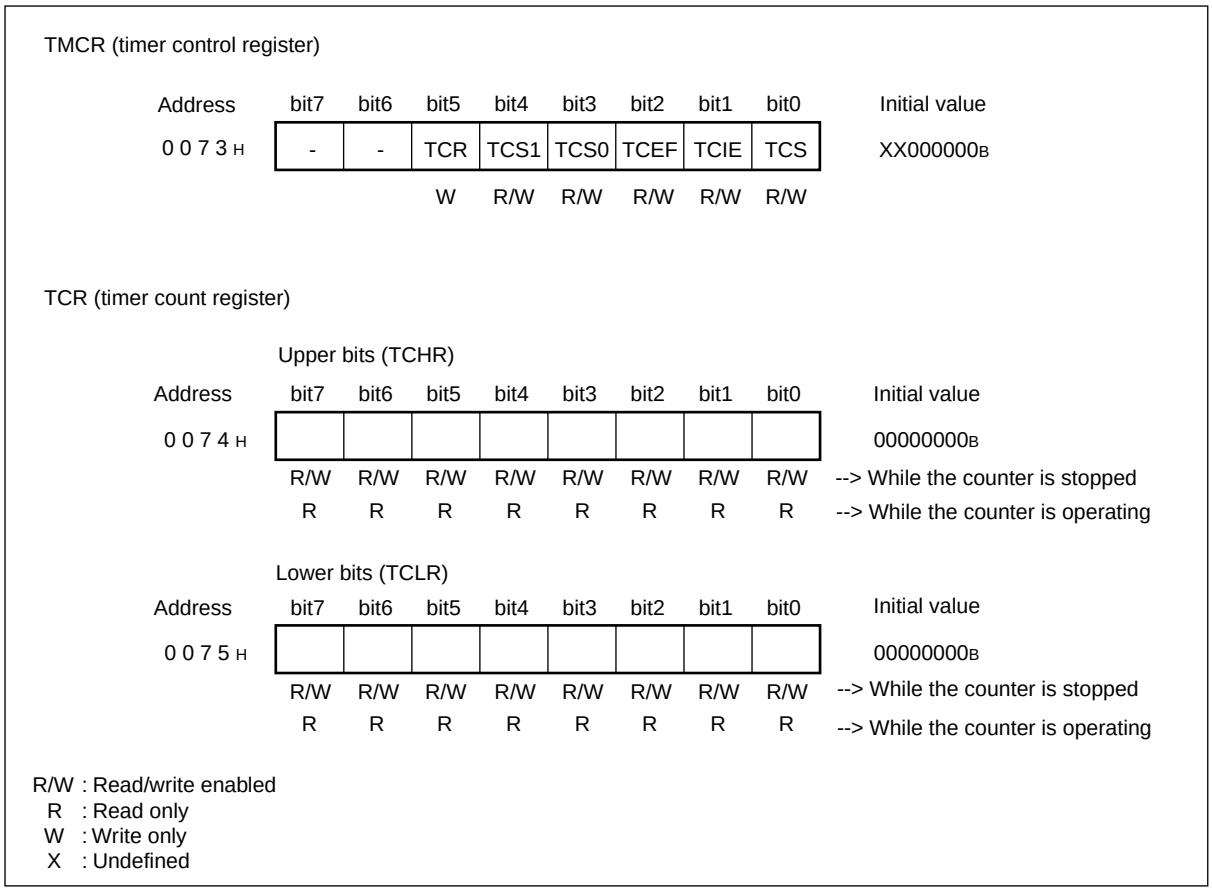


# 9.4 Registers of the 16-bit Timer/Counter

**This section describes the registers related to the 16-bit timer/counter.**

## ■ Registers Related to the 16-bit Timer/Counter

**Figure 9.4-1 Registers Related to the 16-bit Timer/Counter**



# 9.4.1 Timer Control Register (TMCR)

**The timer control register (TMCR) selects a 16-bit timer/counter function (either the interval timer or counter function), sets the operating conditions, enables or disables operation, clears the counter, controls interrupts, and checks the status.**

## ■ Timer Control Register (TMCR)

**Figure 9.4-2 Timer Control Register (TMCR)**

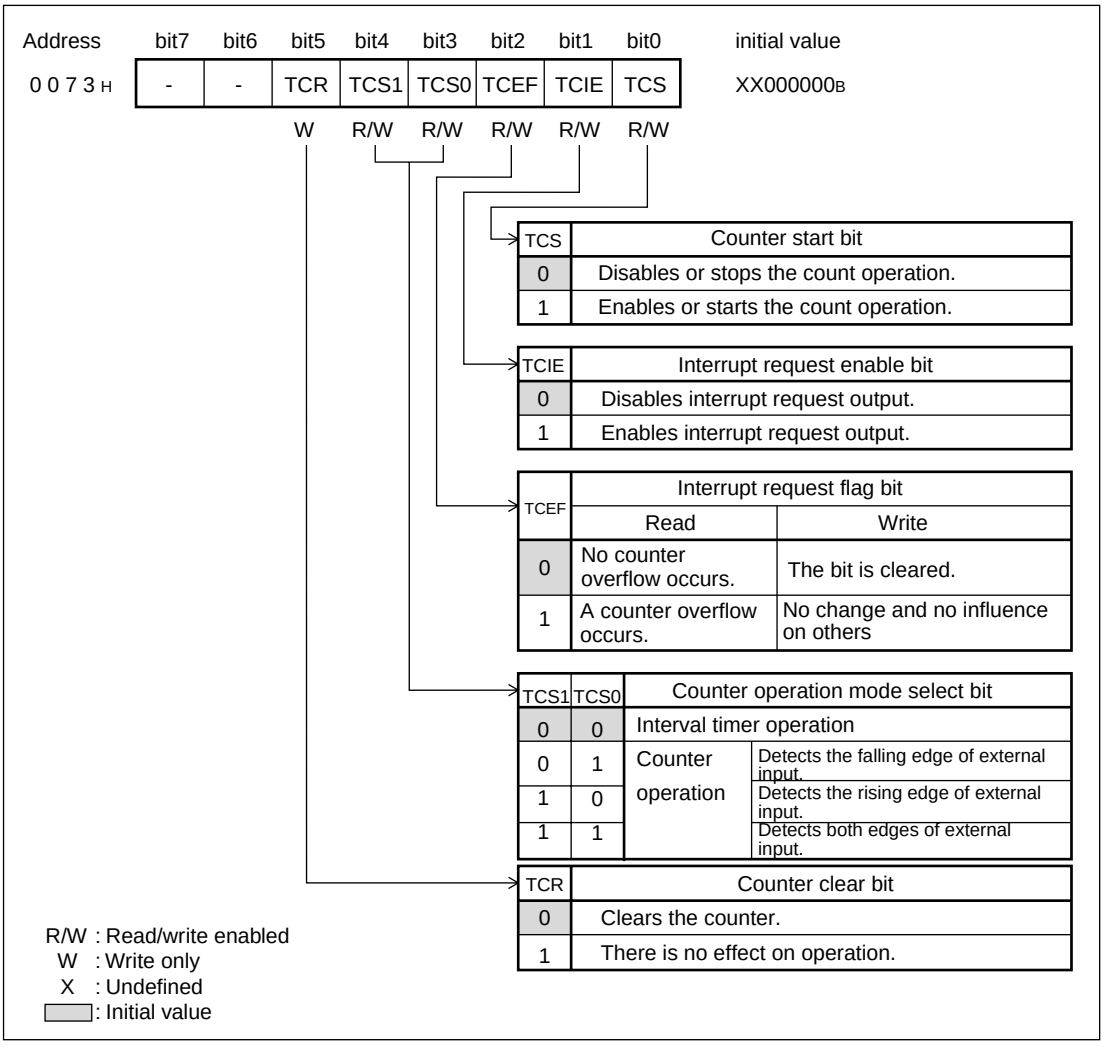


**Table 9.4-1 Functions of the Timer Control Register (TMCR) Bits**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 5 | TCR: Counter clear bit | • Clears the timer count register (TCR).<br>• Writing 0 to this bit clears the timer count register to $0000_H$. Writing 1 to this bit has no effect and has no effect on other bits.<br>**Reference:**<br>If this bit is read, the value is always 1. |
| Bit 4<br>Bit 3 | TCS1, TCS0: Counter operation mode select bits | • Switches the interval timer and counter functions. Setting these bits to $00_H$ selects the interval timer function that results in operation using the internal count clock.<br>• Selecting an edge to be detected (falling, rising, or both) from the external count clock causes operation as a 16-bit counter. |
| Bit 2 | TCEF: Interrupt request flag bit | • Set to 1 if the counter overflows.<br>• Setting this bit and the interrupt request enable bit (TCIE) to 1 outputs an interrupt request.<br>• Cleared to 0 for a write. Setting this bit to 1 has no effect and does not affect operation. |
| Bit 1 | TCIE: Interrupt request enable bit | • Enables or disables interrupt request output to the CPU.<br>• Setting this bit and the interrupt request flag bit (TCEF) to 1 outputs an interrupt request. |
| Bit 0 | TCS: Starts or stops the counter. | • Starts and stops the counter.<br>• Writing 1 to this bit starts the count operation of the timer count register (TCR), which counts up according to the count clock. Writing 0 to this bit stops the count operation, and the TCR retains the counter value. |

**Note:**

When clearing the interrupt request flag bit (TMCR: TCEF), do not overwrite the interrupt request enable bit (TMCR: TCIE) at the same time.

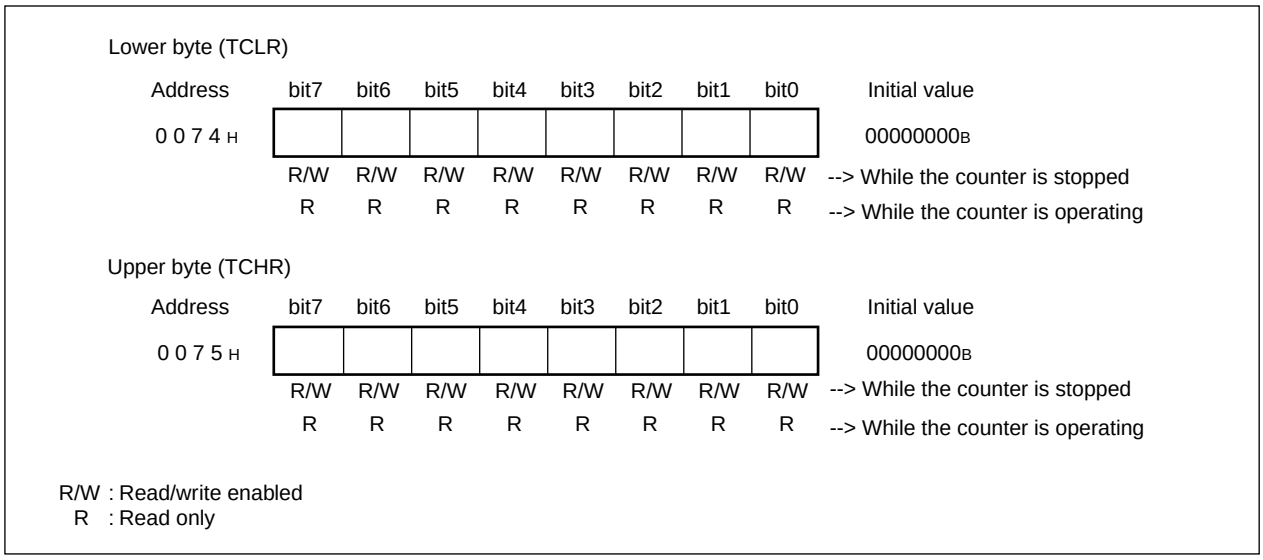
## 9.4.2 16-bit Timer Count Register (TCR)

**The timer count register (TCR) is a 16-bit up counter. The counter counts up from the setting value written in this register.**

### ■ Timer Count Register (TCR)

Figure 9.4-3 "16-bit Timer Count Register (TCR)" shows the bit configuration of the 16-bit timer count register.

**Figure 9.4-3 16-bit Timer Count Register (TCR)**




Both in the interval timer and counter function modes, set a counter initial value in this register while the counter operation is disabled (TMCR: TCS=0). When the counter operation is enabled (TCS=1), the counter counts up from the value written in this register. While the counter is stopped (TCS=0), the TCR register maintains its value. If the counter is cleared (TMCR:TCR=0), the TCR register (counter) becomes $0000_H$.

After the counter is cleared, writing a value in the TCR register sets the counter to the value written.

You can calculate the value in the TCR register in interval timer mode as follows. Note that the instruction cycle is the oscillator frequency divided into four cycles (4/Fc).

TCR register value = $2^{16}$ - (Interval time/Instruction cycle)

Set the upper 8 bits as the TCHR register and the lower 8 bits as the TCLR register.

**Note:**

The value that is set in this register is valid only when the counter is started for the first time. The counter, if it overflows, counts up from $0000_H$.

A value must be written in this register while the counter is stopped (TMCR: TCS=0).

A value can be read from this register even while the counter is operating.

Always use a word transfer instruction (such as MOVW A, 0019H) to read this register.

# 9.5 16-bit Timer/Counter Interrupts

**A 16-bit timer/counter interrupt is caused by:**

- **An overflow in interval timer function mode ($FFFF_H$ --> $0000_H$)**
- **An overflow in the 16-bit counter function mode ($FFFF_H$ --> $0000_H$)**

## ■ Interrupts in Interval Timer Function Mode

If the counter counts up from the defined counter value according to the internal count clock until it overflows, the interrupt request flag bit (TMCR: TCEF) is set to 1. If, at this time, the interrupt request enable bit is set to Enabled (TMCR: TCIE=1), an interrupt request (IRQD) occurs in the CPU.

Use the interrupt processing routine to write 0 to the TCEF bit and clear the interrupt request.

If the counter is cleared (TMCR: TCR=0) and the counter value overflows at the same time, the TCEF bit is not set. While the TCEF bit is 1, setting the TCIE bit from Disabled to Enabled (0 to 1) immediately causes an interrupt request.

The TCEF bit is set whenever the counter value overflows regardless of the value in the TCIE bit.

## ■ Interrupts in Counter Function Mode

If the counter counts up from the defined counter value each time preset edge is detected until it overflows, the interrupt request flag bit (TMCR: TCEF) is set to 1. If, at this time, the interrupt request enable bit is set to Enabled (TMCR: TCIE=1), an interrupt request (IRQD) occurs in the CPU.

Use the interrupt processing routine to write 0 to the TCEF bit and clear the interrupt request.

If the counter is cleared (TMCR: TCR=0) and the counter value overflows at the same time, the TCEF bit is not set. While the TCEF bit is 1, setting the TCIE bit from Disabled to Enabled (0 to 1) immediately causes an interrupt request.

The TCEF bit is set whenever the counter value overflows regardless of the value in the TCIE bit.

## ■ Register Related to the Interrupts of the 16-bit Timer/Counter and the Vector Table

**Table 9.5-1 Register Related to the Interrupts of the 16-bit Timer/Counter and the Vector Table**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQD | ILR4 ($007E_H$) | LD1 (bit 3) | LD0 (bit 2) | $FFE0_H$ | $FFE1_H$ |

For the operation of interrupts, see Section 3.4.2 "Interrupt Processing".

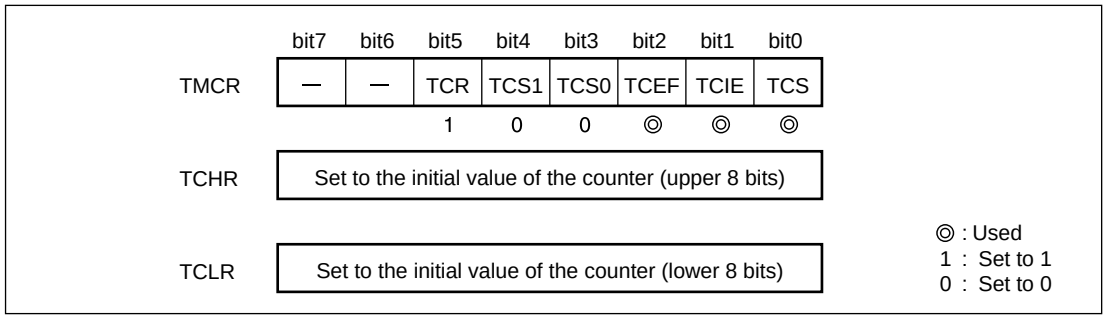
# 9.6 Operation of the Interval Timer Function

**This section describes the operation of the interval timer function of the 16-bit timer/ counter.**

## ■ Operation of the Interval Timer Function

For interval timer function operation, the setting shown in Figure 9.6-1 "Setting the Interval Timer Function" is necessary.

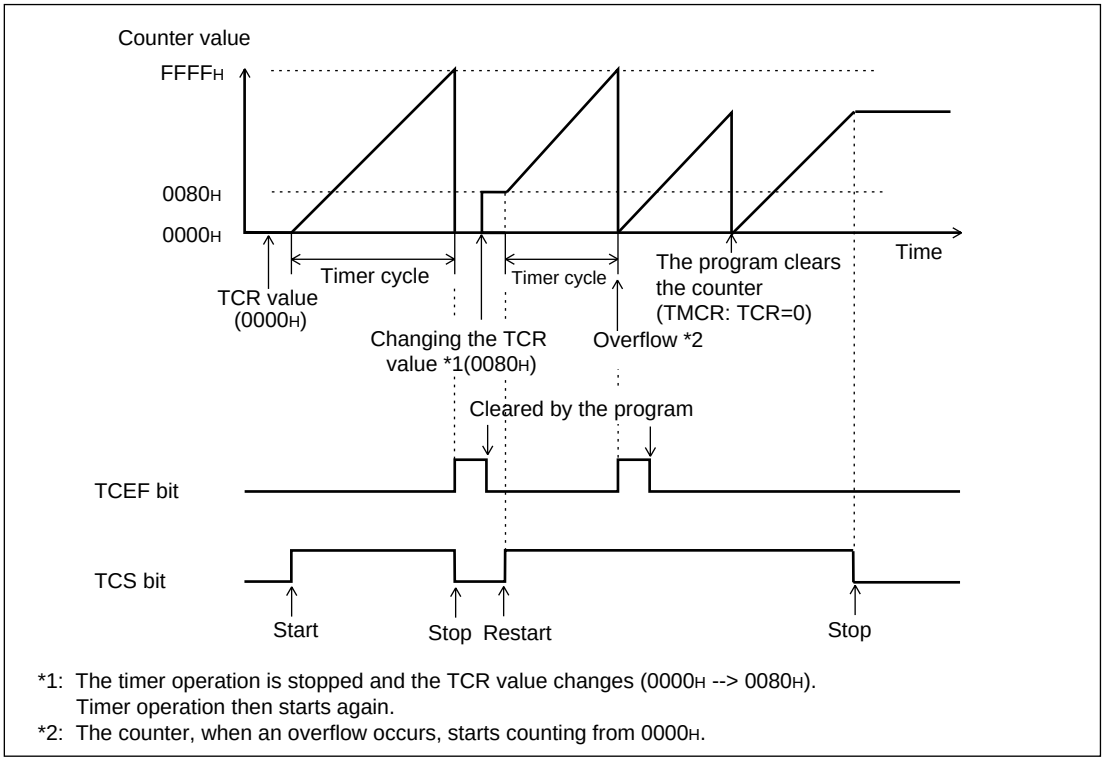
**Figure 9.6-1 Setting the Interval Timer Function**




If the counter is started (TMCR: TCS=1), the counter starts counting up from the value in the TCR register on every rising edge of the internal count clock (1 $t_{inst}$: the oscillator frequency divided into four cycles). If the counter overflows ($FFFF_H$ --> $0000_H$), the interrupt request flag bit is set (TMCR: TCEF=1). After an overflow, the counter starts counting up from $0000_H$.

Figure 9.6-2 "Operation of the Interval Timer" shows the operation of the interval timer.

**Figure 9.6-2 Operation of the Interval Timer**




*1: The timer operation is stopped and the TCR value changes (0000H --> 0080H). Timer operation then starts again.
*2: The counter, when an overflow occurs, starts counting from 0000H.

**Note:**

Do not write a value to the TCR register while the interval timer function is operating (TMCR: TCS=1)

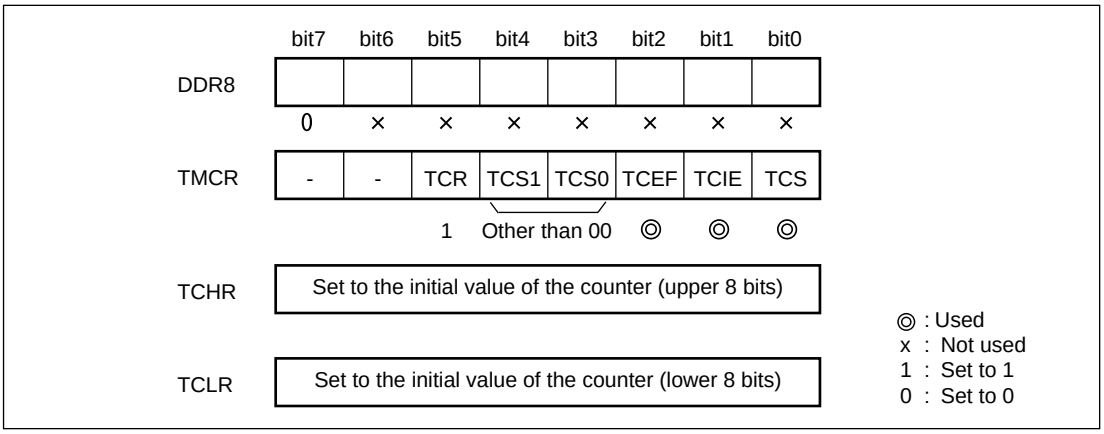
# 9.7 Operation of the Counter Function

**This section describes the operation of the counter function of the 16-bit timer/ counter.**

## ■ Operation of the Counter Function

To the counter function operation, the setting shown in Figure 9.7-1 "Setting the Counter Function", "Setting the counter function," is necessary

**Figure 9.7-1 Setting the Counter Function**

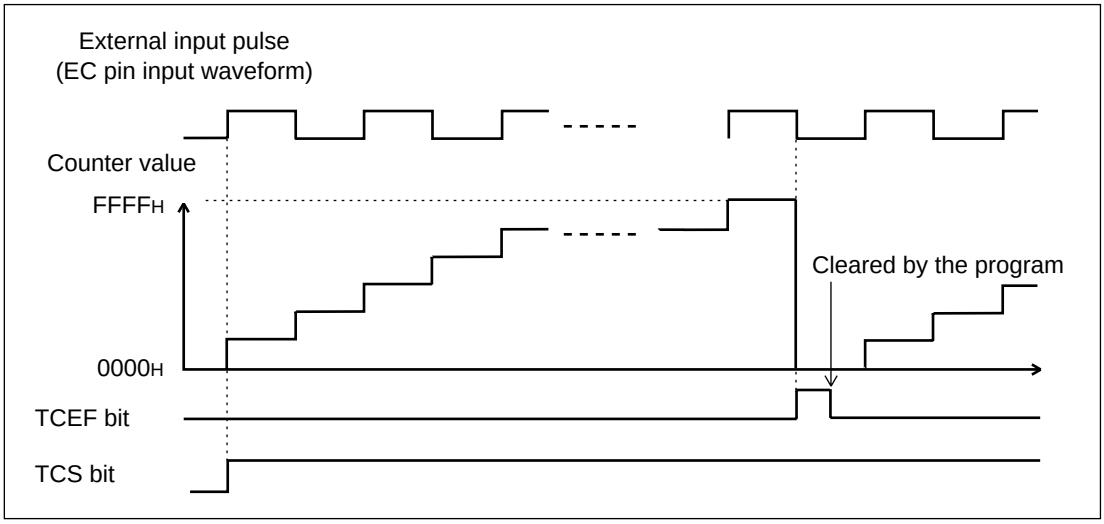



If the counter is started (TMCR: TCS=1), the counter starts counting up from the value in the TCR register whenever the prespecified edge of a pulse that is input to the EC pin (the external count clock) is detected.

If the counter overflows ($FFFF_H$ --> $0000_H$), the interrupt request flag bit is set (TMCR: TCEF=1).

Then, if the next prespecified edge is input, the counter starts counting up from $0000_H$.Figure 9.7-2 "16-bit Counter Operation" shows the operation when the counter operation mode select bits (TMCR: TCS1, TCS0) are set to $11_B$ (detecting both edges) and the TCR register to $0000_H$.

**Figure 9.7-2 16-bit Counter Operation**




**Note:**

Do not write a value to the TCR register while the counter function is operating (TMCR: TCS=1)

# 9.8 Status of the 16-bit Timer/Counter in Each Mode

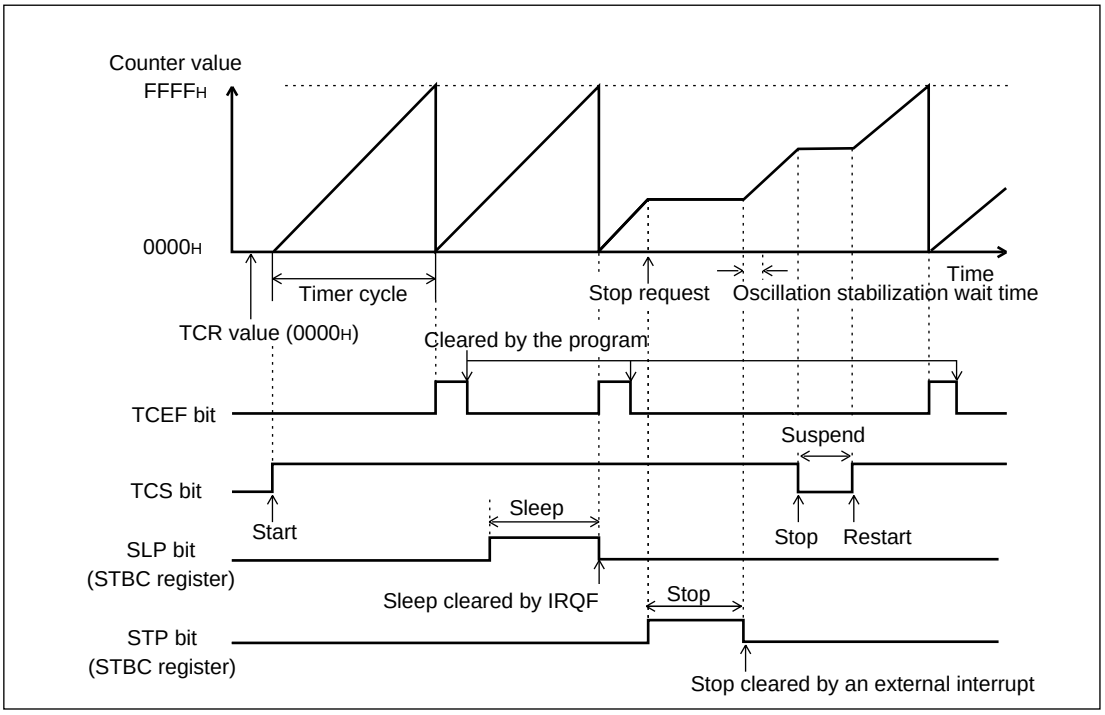
**This section describes the operations of switching to the sleep and stop modes and of receiving a suspend request while the 16-bit timer/counter is operating.**

## ■ Operation in Low Power Consumption (Standby) Mode and when the Counter is Suspended

Figure 9.8-1 "Operation of the Counter in Low Power Consumption (Standby) Mode and when the Counter is Suspended" shows the status of the counter value upon switching to the sleep and stop modes and upon receiving a suspend request while the interval timer or counter function is operating.

In stop mode, the counter stops, retaining the value. If stop mode is cleared by an external interrupt, the counter starts counting from the retained value. Therefore, the initial interval time and the input pulse edge count cannot be correct values. After stop mode is cleared, initialize the 16-bit timer/counter again.

**Figure 9.8-1 Operation of the Counter in Low Power Consumption (Standby) Mode and when the Counter is Suspended**




The counter value is retained while the counter is stopped (TMCR: TCS=0).

# 9.9 Notes on Using the 16-bit Timer/Counter

**This section contains notes on using the 16-bit timer/counter.**

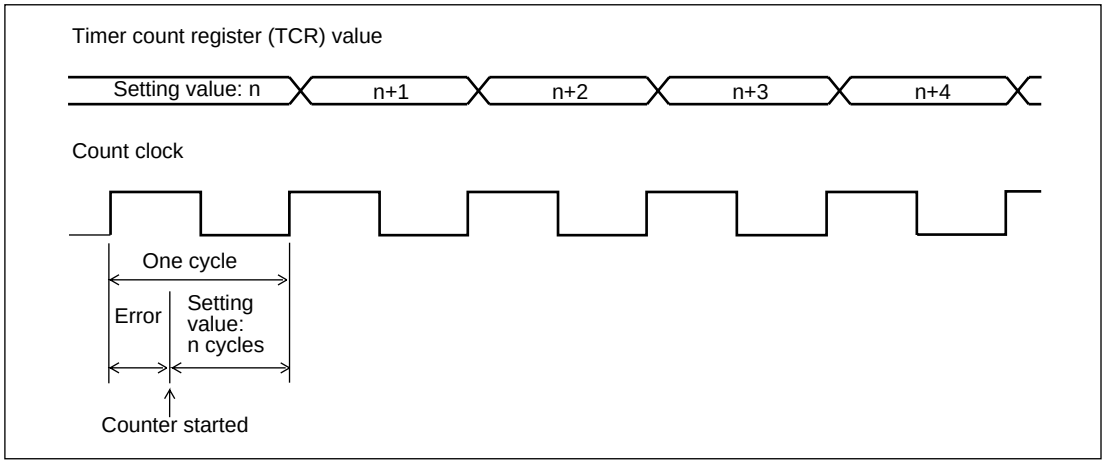
## ■ Notes on Using the 16-bit Timer/Counter

### ❍ Error

In interval timer function mode, starting of the counter by the program and starting of count-up by the internal count clock are asynchronous. Thus, an error of one less instruction cycle at the most may exist in the time elapsing until the counter overflows.

Figure 9.9-1 "Error Until the Count Operation is Started" shows an error until the count operation is started.

**Figure 9.9-1 Error Until the Count Operation is Started**




### ❍ Notes on setting the program

- Write a value to the TCR register while the counter operation is stopped (TMCR: TCS=0). A value can be read even while the counter is counting. However, always use a word transfer instruction (such as MOVW A, dir) to read this register.
- Change the counter operation mode select bits (TMCR: TCS1, TCS0) while the counter is stopped (TMCR: TCS=0), an interrupt is disabled (TCIE=0), and the interrupt request is cleared (TCEF=0).
- If the interrupt request flag bit (TMCR: TCEF) is 1 and the interrupt request is enabled (TMCR: TCIE=1), the counter cannot restored after interrupt processing. Always clear the TCEF bit.
- If the counter is cleared (TMCR: TCR=0) and the counter value overflows at the same time, the interrupt request flag bit (TMCR: TCEF) is not set.

# 9.10 Programe Example of the 16-bit Timer/Counter

**This section contains sample programs for the 16-bit timer/counter.**

## ■ Program Example of the Interval Timer Function

### ❍ Processing specification

- Generate a 20-ms interval timer interrupt.
- Use the interrupt processing routine to reset the TCR register and generate an interrupt repeatedly.
- The following shows the TCR register value for an interval time of 20 ms when the oscillator frequency is 10 MHz.
- TCR register value = $2^{16}$ - 20 ms / (4/10 MHz) = 15536 (3CB0H)

### ❍ Coding example

```
TMCR    .EQU     0073H        ; Address of the timer control reister
TCHR    .EQU     0074H        ; Upper address of the timer count
                                register
TCLR    .EQU     0075H        ; Lower address of the timer count
                                register
TCEF    .EQU     TMCR:2       ; Definition of the interrupt request
                                flag bit
TCS     .EQU     TMCR:0       ; Definition of the count start bit
ILR4    .EQU     007EH        ; Address of the interrupt level setting
                                register
        .SECTION INT_V, DATA, LOCATE=0    ; [DATA SEGMENT]
        .ORG     0FFDCH
IRQD    .DATA.H  WARI         ; Setting interrupt vector
;INT_V  ENDS
;----Main program-------------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1     ; [CODE SEGMENT]
                              ; Stack pointer (SP) and other
                                are assumed to have been initialized
          :
        CLRI                  ; Interrupt disable
        CLRB     TCS          ; Count operation stop
        MOV      ILR4,#11110111B  ; Setting interrupt level (level 1)
        MOV      TCHR,#03CH   ; Set the 20 ms timer data
        MOV      TCLR,#0B0H
        MOV      TMCR,#00100011B  ; Retain the counter value,
                                    set the interval timer operation,
                                    clearing the interrupt request
                                    flag, enabling interrupt request
                                    output, and start counter operation
        SETI                  ; Interrupt enable
          :
;----Interrupt program-------------------------------------------------
WARI
```

```
        MOV     TMCR,#00100000B    ; Clearing interrupt request flag
                                     and stop counter operatin
        PUSHW   A
        XCHW    A,T
        PUSHW   A
        MOVW    A,TCHR             ; Add the time from the overflow to the
                                     interrupt acceptance
        MOV     A,#3CB0H           ; 20 ms timer data (at 10 MHz)
        CLRC
        ADDCW   A                  ; Here, an overflow during addition is
                                     not considered
        MOVW    TCHR,A             ; Strictly, the time while the counter
                                     is stopped must be added
        MOV     TMCR,#00100011B    ; Enable the interrupt, and
                                     start counting
          :
         User processing
          :
        POPW    A
        XCHW    A, T
        POPW    A
        RETI
        ENDS
;-------------------------------------------------------------------------
        .END
```

## ■ Program Example of the Counter Function

### ❍ Processing specifications

- Generate an interrupt whenever the rising edge of a pulse being input to the EC pin is counted 10,000 times.
- Use the interrupt processing routine to reset the TCR register and generate an interrupt repeatedly.
- The following shows the TCR register value at which the counter overflows when a rising edge is detected 10,000 times.
  - TCR register value = $2^{16}$ - 10000 = 65536 - 10000 = 55536 = D8F0H

### ❍ Coding example

```
TMCR    .EQU    0073H         ; Address of the timer control reister
TCHR    .EQU    0074H         ; Upper address of the timer count
                                register
TCLR    .EQU    0075H         ; Lower address of the timer count
                                register
TCEF    .EQU    TMCR:2        ; Definition of the interrupt request
                                flag bit
TCS     .EQU    TMCR:0        ; Definition of the count start bit
ILR4    .EQU    007EH         ; Address of the interrupt level setting
                                register
        .SECTION INT_V, DATA, LOCATE=0    ; [DATA SEGMENT]
        .ORG    0FFDCH
IRQF    .DATA.H  WARI         ; Setting interrupt vector
;INT_V  ENDS
;----Main program-------------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1     ; [CODE SEGMENT]
                              ; Stack pointer (SP) and other
                                are assumed to have been initialized
          :
        CLRI                  ; Interrupt disable
        CLRB    TCS           ; Count operation stop
        MOV     ILR4,#01111111B  ; Setting interrupt level (level 1)
        MOV     TCHR,#0D8H    ; Initialize the counter value
        MOV     TCLR,#0F0H
        MOV     TMCR,#00110011B  ; Retain the counter value,
                                   set the counter function
                                   (selecting the rising edge of
                                   external input), clearing the
                                   interrupt request flag, enabling
                                   the interrupt request output, and
                                   enabling the counter operation
        SETI                  ; Interrupt enable
          :
;----Interrupt program-------------------------------------------------
WARI
        CLRB     TCEF         ; Clearing interrupt request flag
        PUSHW   A
        XCHW    A,T
        PUSHW   A
        CLRB    TCS           ; Stop counter operation
```

```
        MOV     A,0D8H      ; Initialize the counter value
        MOV     TCHR,A      ; Here, the pulse after overflow is
                              ignored
        MOV     A,#0F0H
        MOV     TCLR,A
        SETB    TCS         ; Restart the count operation, and
                              start counting 10,000 pulses from here
          :
         User processing
          :
        POPW    A
        XCHW    A, T
        POPW    A
        RETI
        ENDS
;-------------------------------------------------------------------
        .END
```

# CHAPTER 10 EXTERNAL INTERRUPTS (EDGES)

**This chapter describes the functions and operations of the external interrupt circuit (edge).**


10.1 "Overview of the External Interrupt Circuit"

10.2 "Configuration of the External Interrupt Circuit"

10.3 "Pins of the External Interrupt Circuit"

10.4 "Registers of the External Interrupt Circuit"

10.5 "External Interrupt Circuit Interrupts"

10.6 "Operation of the External Interrupt Circuit"

# 10.1 Overview of the External Interrupt Circuit

**The external interrupt circuit detects edges of the signal input into the four external interrupt pins to issue an interrupt request to CPU.**

## ■ External Interrupt Function

The external interrupt circuit has a function to detect edges of the signal input into the external interrupt pins to issue an interrupt request to CPU, which makes a return from the standby mode and a transition to a normal operation state (main RUN state) possible.

- External interrupt pin: Four (P80/INT0 to P83/INT3)
- External interrupt source: Signal input of any edge to the external interrupt pins
- Interrupt control: Permission and prohibition of the interrupt request output by the interrupt request enable bit of the external interrupt control registers (EIC1 to EIC2)
- Interrupt flag: Detection of the specified edges by the external interrupt request flag bit of the external interrupt control registers (EIC1 to EIC2)
- Interrupt request: Issued according to each external interrupt source (IRQ0, IRQ1)

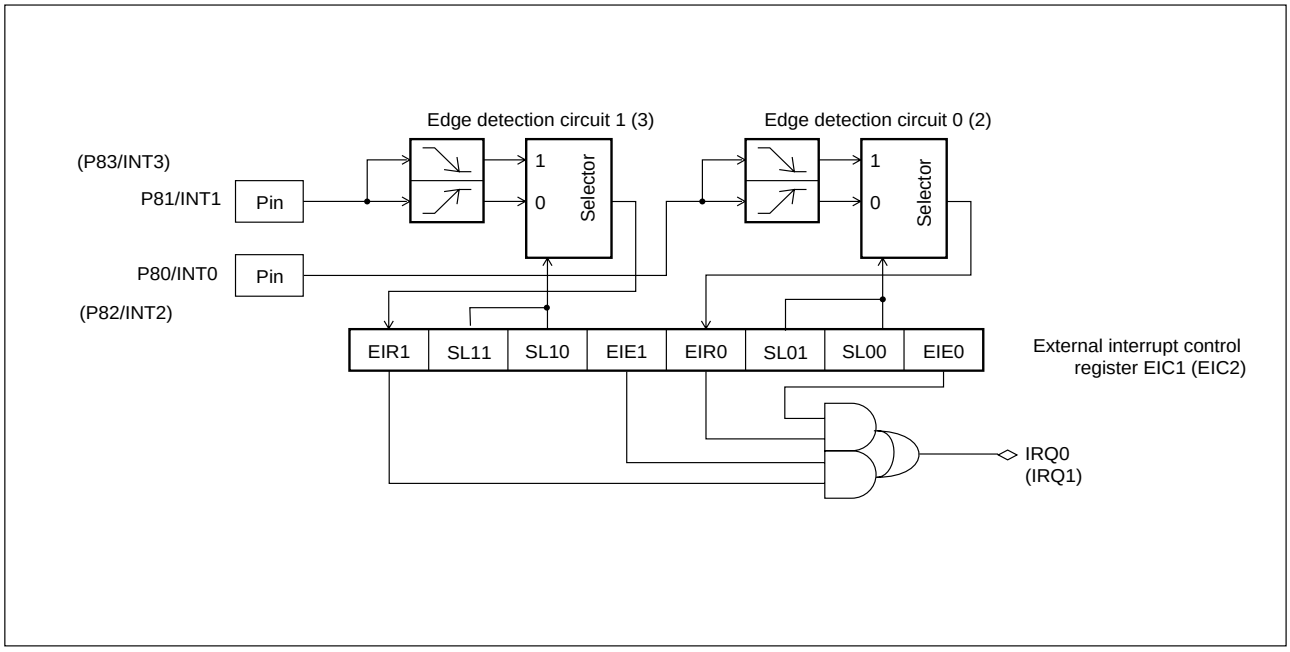
# 10.2 Configuration of the External Interrupt Circuit

**The external interrupt circuit comprises the following two elements:**

- **Edge detection circuit (0 to 3)**
- **External interrupt control register (EIC1 to 2)**

## ■ Block Diagram of the External Interrupt Circuit

**Figure 10.2-1 Block Diagram of the External Interrupt Circuit**




### ❍ Edge detection function

If the edge polarity of the signal input into the external interrupt pins (INT0 to INT3) and that (SL01 to SL31, SL00 to SL30) selected by the EIC1 to EIC2 registers match, the corresponding external interrupt request flag bit (EIR0 to EIR3) is set to "1".

### ❍ External interrupt control register (EIC1 to EIC2)

The EIC1 to EIC2 registers are used to select the edges, allow/prohibit interrupt requests, and check interrupt requests.

### ❍ Interrupt sources of external interrupts

**IRQ0:**

If the interrupt request output is allowed (EIC1: EIE0=1, EIE1=1), an interrupt request is issued when an edge of the selected polarity enters the external interrupt pin INT0 or INT1.

**IRQ1:**

If the interrupt request output is allowed (EIC2: EIE2=1, EIE3=1), an interrupt request is issued when an edge of the selected polarity enters the external interrupt pin INT2 or INT3.

# 10.3 Pins of the External Interrupt Circuit

**This section describes the pins related to the external interrupt circuit and provides a block diagram of the pins.**

## ■ Pins Related to the External Interrupt Circuit

The pins related to the external interrupt circuit are the P80/INT0 to P83/INT3 pins.

### ❍ P80/INT0 to P83/INT3 pins

These pins provide the function as a general-purpose I/O dedicated port (P80 to P83) and also serve for external interrupt input (hysteresis input) (INT0 to INT3).

If the interrupt request output is not allowed, an interrupt request is not output. The pin state can be read directly through the port data register (PDR8).

Table 10.3-1 "Pins Related to the External Interrupt Circuit" lists the pins related to the external interrupt circuit.

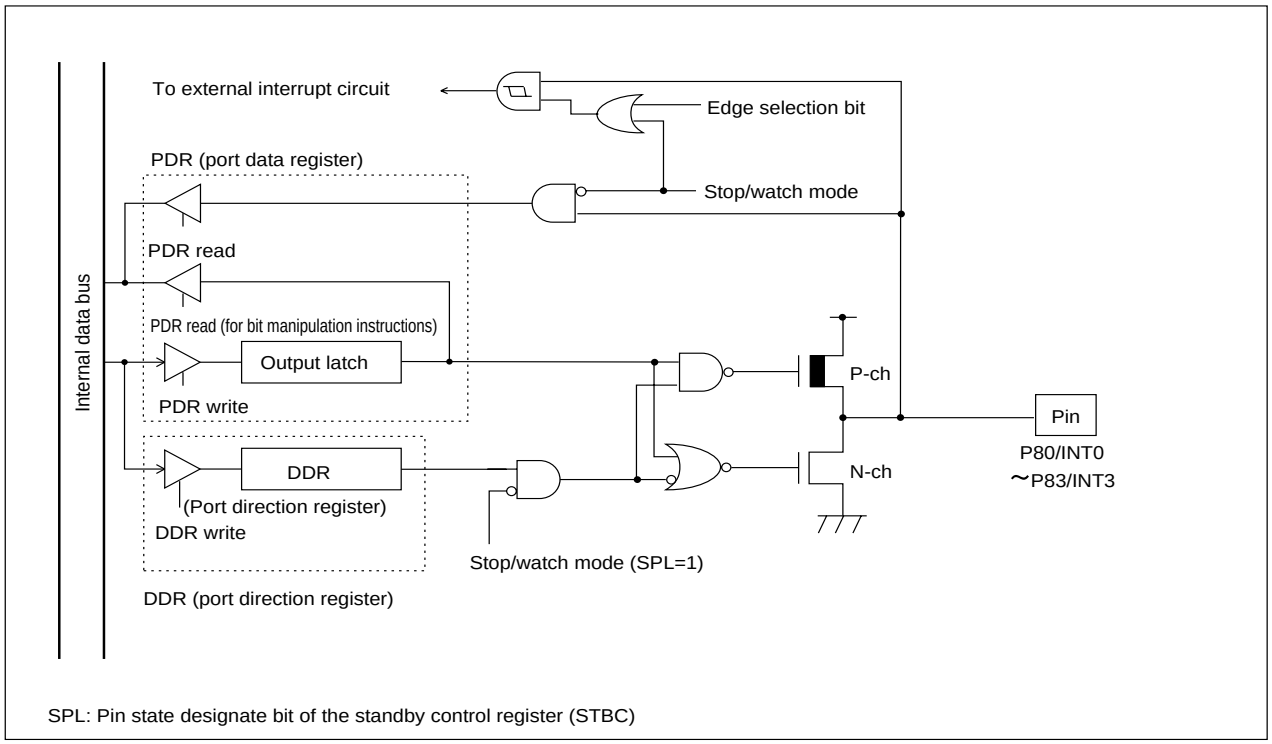
**Table 10.3-1 Pins Related to the External Interrupt Circuit**

| External interrupt pin | Used for external interrupt input (interrupt request output allowed) | Used as an input dedicated port (interrupt request output prohibited) |
|---|---|---|
| P80/INT0 | INT0 (EIC1:EIE0=1) | P80 (EIC1:EIE0=0) |
| P81/INT1 | INT1 (EIC1:EIE1=1) | P81 (EIC1:EIE1=0) |
| P82/INT2 | INT2 (EIC2:EIE2=1) | P82 (EIC2:EIE2=0) |
| P83/INT3 | INT3 (EIC2:EIE3=1) | P83 (EIC2:EIE3=0) |

INT0 to INT3: If an edge of the selected polarity enters these pins, an interrupt corresponding to the pin is generated.

## ■ Block Diagram of the Pins Related to the External Interrupt Circuit

**Figure 10.3-1 Block Diagram of the Pins Related to the External Interrupt Circuit**




SPL: Pin state designate bit of the standby control register (STBC)

**Note:**

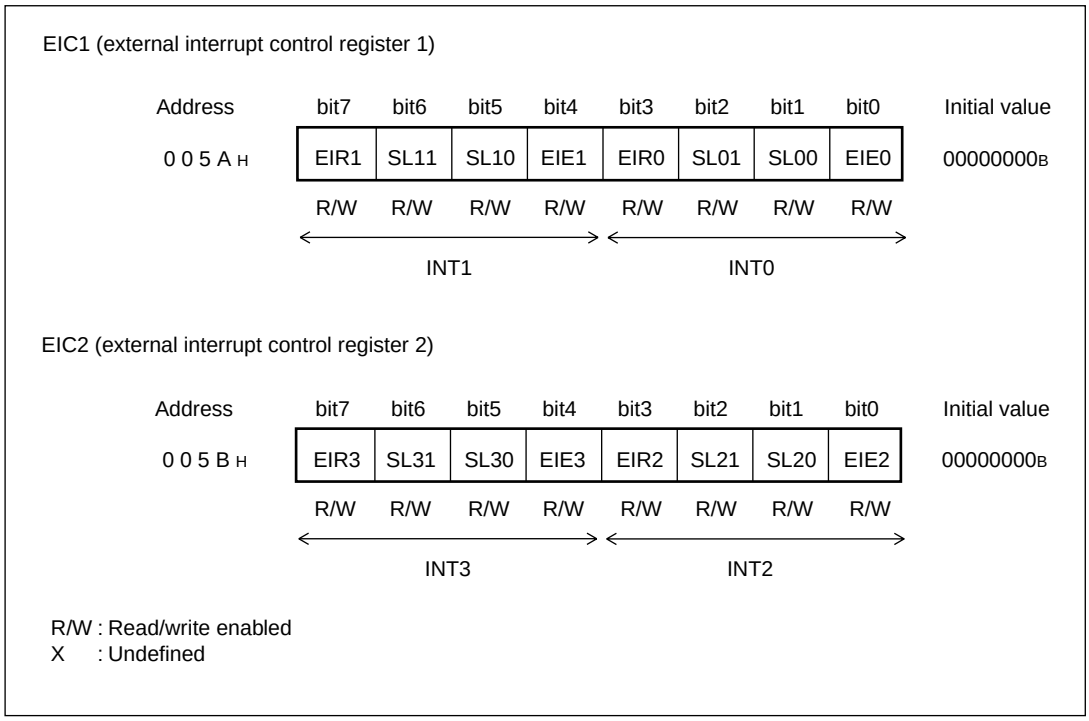
At the stop mode (SPL=1), the external interrupt input becomes input enable state and cutting off the external interrupt input buffer, even if the edge polarity selection bit was selected to the rising edge, falling edge, or both edges. In that case, it should be fix the voltage level by using the external pull-up resistor or pull-down resistor.

# 10.4 Registers of the External Interrupt Circuit

**This section describes the registers related to the external interrupt circuit.**

## ■ Registers Related to the External Interrupt Circuit

**Figure 10.4-1 Registers Related to the External Interrupt Circuit**



# 10.4.1 External Interrupt Control Register (EIC1 to EIC2)

**The external interrupt control registers (EIC1 to EIC2) are used to select the edge polarity for the external interrupt pins INT0 to INT3 and control interrupts. This section shows the register configuration using EIC1 as an example.**

## ■ External Interrupt Control Register (EIC1)

**Figure 10.4-2 External Interrupt Control Register (EIC1)**

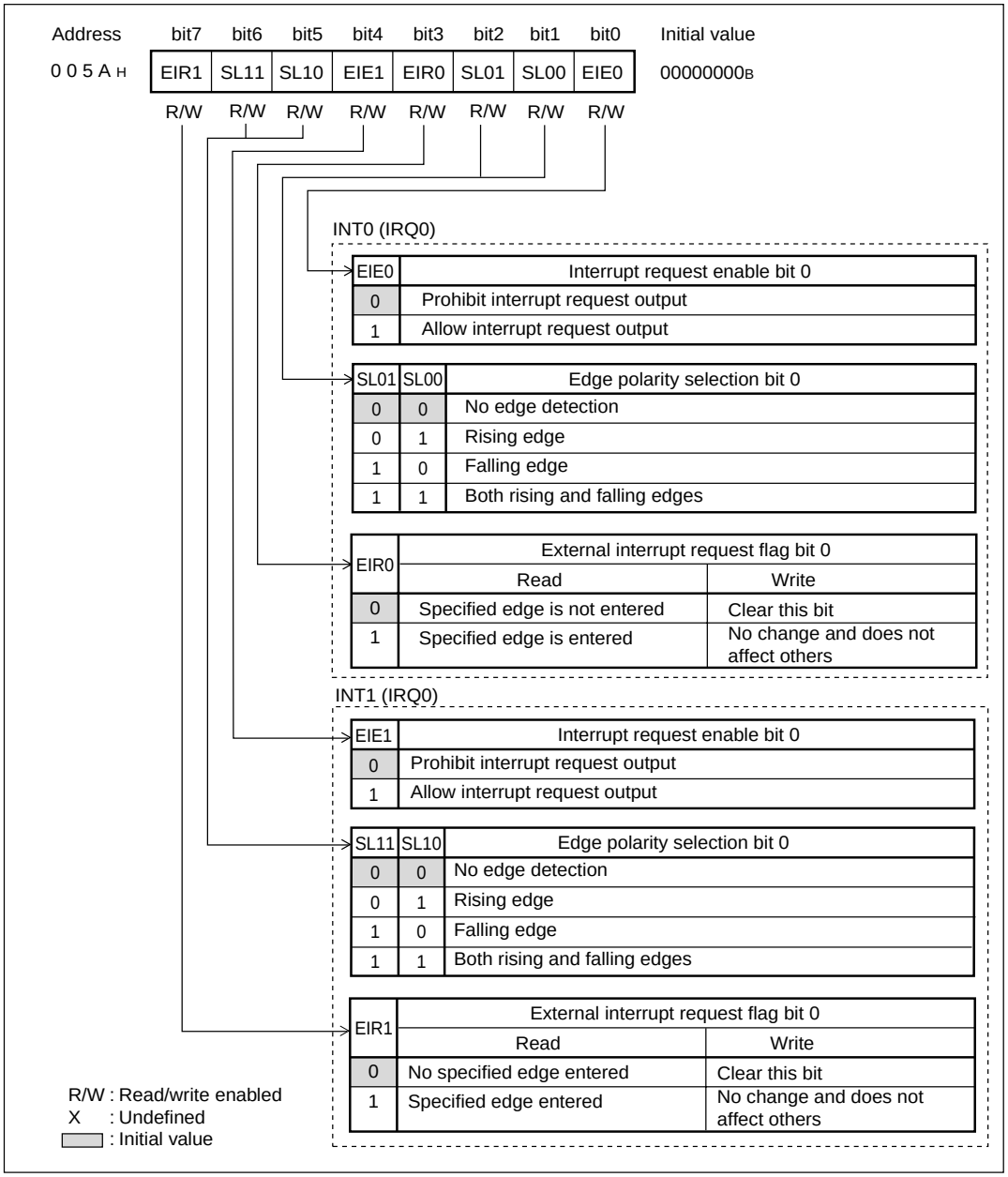


**Table 10.4-1 Explanation of Functions of Each Bit of the External Interrupt Control Register (EIC1)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | EIR1:<br>External interrupt request flag bit 1 | • "1" is set if the edge selected by the edge polarity selection bit 1 (SL11, SL10) is entered in the external interrupt pin INT1.<br>• If this bit and the interrupt request enable bit 1 (EIE1) are "1", an interrupt request is output.<br>• This bit is cleared by writing "0". If "1" is written, this bit is not affected and changed. |
| Bit 6<br>Bit 5 | SL11, SL10:<br>Edge polarity selection bit 1 | • Bits to select the polarity of edges to be an interrupt source for pulses input into the external interrupt pin INT1. |
| Bit 4 | EIE1:<br>Interrupt request enable bit 1 | • Bit to allow/prohibit interrupt request output to CPU. If this bit and the external interrupt request flag bit 1 (EIR1) are "1", an interrupt request is output. |
| Bit 3 | EIR0:<br>External interrupt request flag bit 0 | • "1" is set if the edge selected by the edge polarity selection bit 0 (SL01, SL00) is entered in the external interrupt pin INT0.<br>• If this bit and the interrupt request enable bit 0 (EIE0) are "1", an interrupt request is output.<br>• This bit is cleared by writing "0". If "1" is written, this bit is not affected and changed. |
| Bit 2<br>Bit 1 | SL01, SL00:<br>Edge polarity selection bit 0 | • Bits to select the polarity of edges to be an interrupt source for pulses input into the external interrupt pin INT0. |
| Bit 0 | EIE0:<br>Interrupt request enable bit 0 | • Bit to allow/prohibit interrupt request output to CPU. If this bit and the external interrupt request flag bit 0 (EIR0) are "1", an interrupt request is output. |

# 10.5 External Interrupt Circuit Interrupts

**As an interrupt source of the external interrupt circuit, the detection of the specified edge of a signal input into the external interrupt pin is available.**

## ■ Interrupts when the External Interrupt Circuit is Operating

If the specified edge of the external interrupt input is detected, "1" is set to the corresponding external interrupt request flag bit (EIC1: EIR0 to EIR1/EIC2: EIR2 to EIR3). At this time, if the corresponding interrupt request enable bit is set (EIC1: EIE0 to EIE1=1/EIC2: EIE2 to EIE3=1), an interrupt request (IRQ0, IRQ1) to CPU is issued.

The following table lists the correspondence of the interrupt requests (IRQ0 to IRQ1).

| Interrupt request (INT) | Interrupt request to CPU |
|---|---|
| INT0 | IRQ0 |
| INT1 | |
| INT2 | IRQ1 |
| INT3 | |

If no external interrupt is used to return from the stop mode, set "00" to the edge polarity bits and "0" to the interrupt enable bit.

**Note:**

To allow interrupts (EIE0 to EIE2=1) after releasing a reset, clear (EIR0 to EIR2=0) the external interrupt request flag bit at the same time.

It is not possible to return from interrupt processing if the external interrupt request flag bit is "1" and the interrupt request enable bit is set. The external interrupt request flag bit in interrupt processing routines must be cleared.

The release of the stop mode by an interrupt is possible only in the external interrupt circuit.

If the interrupt request enable bit is changed from prohibition to permission (0 --> 1), an interrupt occurs immediately.

## ■ Register and Vector Table Related to Interrupts of the External Interrupt Circuit

**Table 10.5-1 Register and Vector Table Related to Interrupts of the External Interrupt Circuit**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ0 | ILR1 ($007B_H$) | L01 (bit 1) | L00 (bit 0) | $FFFA_H$ | $FFFB_H$ |
| IRQ1 | ILR1 ($007B_H$) | L11 (bit 3) | L10 (bit 2) | $FFF8_H$ | $FFF9_H$ |

For interrupt operations, see Section 3.4.2 "Interrupt Processing".

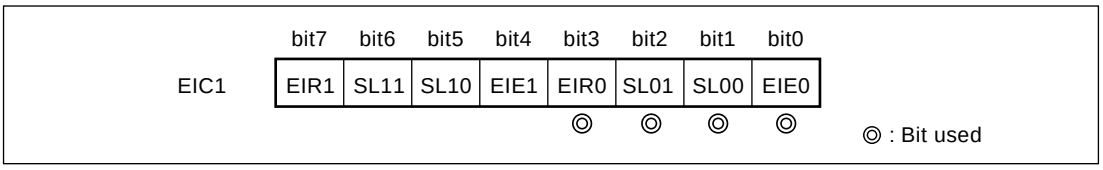
# 10.6 Operation of the External Interrupt Circuit

**The external interrupt circuit can detect the specified edge of signal input into the external interrupt pins. This section describes the operations, using INT0 as an example.**

## ■ Operations of the External Interrupt Circuit

The setting in Figure 10.6-1 "Setting of the External Interrupt Circuit" is required for the operation of INT0 of the external interrupt circuit.

**Figure 10.6-1 Setting of the External Interrupt Circuit**

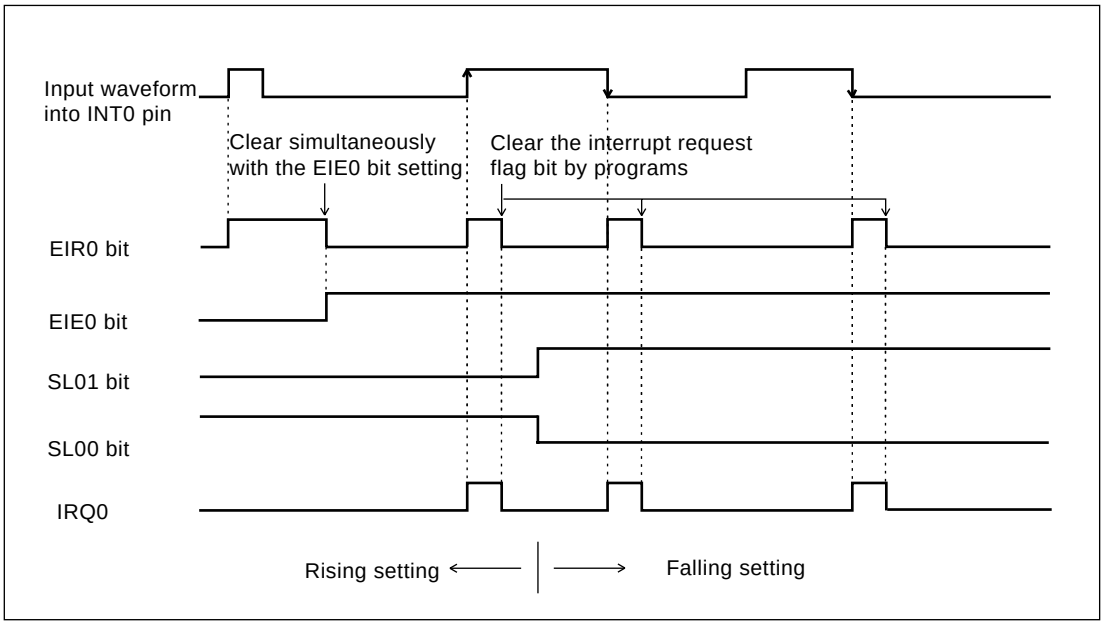



If the edge polarity of the signal input from the external interrupt pin (INT0) and that (EIC1: SL01, SL00) selected by the external interrupt control register match, the corresponding external interrupt request flag bit (EIC1: EIR0) is set to "1".

The external interrupt request flag bit is set if the polarity of edges match, regardless of the interrupt request enable bit (EIC1: EIE0).

Figure 10.6-2 "Operations of the External Interrupts (INT0)" shows the operations when the INT0 pin is used for external interrupt input.

**Figure 10.6-2 Operations of the External Interrupts (INT0)**




If the pins are used for external interrupt input, the pin states can be read directly from the port data register (PDR8) provided the analog input is not allowed.

# CHAPTER 11 A/D CONVERTER

**This chapter describes the functions and operations of the A/D converter.**


11.1 "Overview of the A/D Converter"

11.2 "Configuration of the A/D Converter"

11.3 "Pins of the A/D Converter"

11.4 "Registers of the A/D Converter"

11.5 "A/D Converter Interrupt"

11.6 "Operation of the A/D Converter"

11.7 "Notes on Using the A/D Converter"

11.8 "Program Example of the A/D Converter"

# 11.1 Overview of the A/D Converter

**The A/D converter is a 10-bit successive approximation type that selects one input signal from 12-channel analog pins. It can be started by software.**

## ■ A/D Conversion Function

The A/D conversion function converts an analog voltage entering at an analog input pin (input voltage) to a 10-bit digital value.

- One signal can be selected from 12 analog input pins.
- The conversion rate is 60 instruction cycles (24 μs at 10 MHz main clock oscillation).
- An interrupt occurs when A/D conversion is complete.
- The end of conversion can also be checked with software.

The A/D conversion function can be started by software.

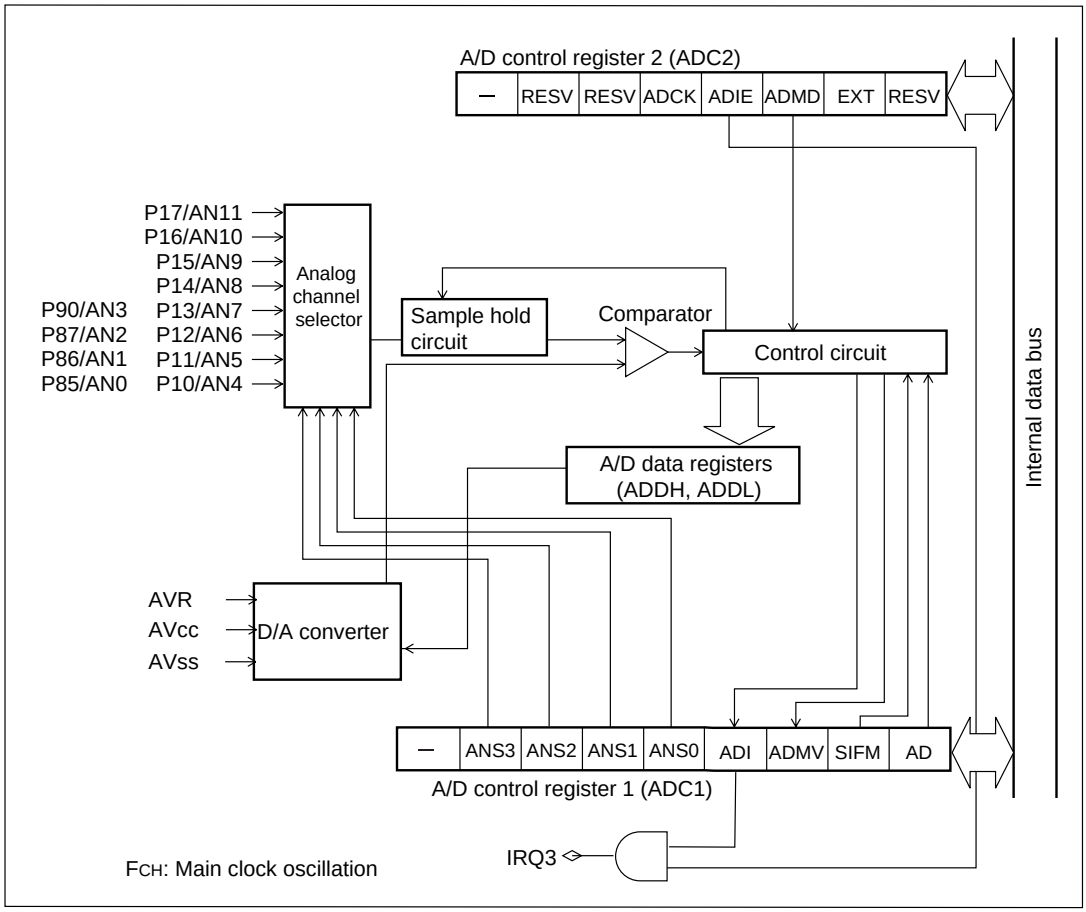
# 11.2 Configuration of the A/D Converter

**The A/D converter consists of the following nine blocks:**

- **Analog channel selector**
- **Sample hold circuit**
- **D/A converter**
- **Comparator**
- **Control circuit**
- **A/D data registers (ADDH, ADDL)**
- **A/D control registers 1, 2 (ADC1, 2)**

## ■ Block Diagram of the A/D Converter

**Figure 11.2-1 A/D Converter Block Diagram**




### ❍ Analog channel selector

The analog channel selector circuit selects one signal from 12 analog input pins.

❍ **Sample hold circuit**

The sample hold circuit holds the input voltage selected by the analog channel selector. By sampling and holding the input voltage immediately after A/D conversion, the analog voltage can be digitized without being affected by input voltage fluctuations during A/D conversion (comparison).

❍ **D/A converter**

Generates a voltage corresponding to the value set in the ADDH and ADDL registers.

❍ **Comparator**

Compares the sampled and held input voltage with the voltage output from the D/A converter and indicates whether the D/A output is greater or less than the input voltage.

❍ **Control circuit**

The control circuit has the following function:

- In the A/D conversion function, sets the bits in the 10-bit A/D data register from the most significant bit to the least significant bit according the signals from the comparator indicating whether the D/A output is greater or less than the input voltage. The control circuit sets the interrupt request flag bit (ADC1: ADI) when the conversion is complete.

❍ **A/D data registers (ADDH/ADDL)**

The upper two bits of the 10-bit A/D data are stored in the ADDH register, and the lower eight bits are stored in the ADDL register.

The result of A/D conversion is stored in the ADDH/ADDL registers.

❍ **A/D control register 1 (ADC1)**

Enables and disables the functions of the A/D converter, selects an analog input pin, checks the state, and controls interrupt.

❍ **A/D control register 2 (ADC2)**

Enables and disables interrupts.

❍ **AD converter interrupts**

**IRQ3:**

When A/D conversion is complete, an interrupt request is issued if interrupt request output is enabled (ADC2: ADIE = 1).

## ■ A/D Converter Power Supply Voltage

❍ **AVcc**

Power supply pin for the A/D converter. Use the same potential that is used for Vcc. If extreme A/D conversion accuracy is required, make sure that AVcc is free of Vcc noise or use another power supply. If the A/D converter is not used, still connect this pin to the power supply.

❍ **AVss**

Ground pin for the A/D converter. Use same potential as that is used for Vss. If extreme A/D conversion accuracy is required, make sure that AVss is free of Vss noise or use another power supply. If the A/D converter is not used, still connect this pin to ground (GND).

❍ **AVR**

Pin to inputting the reference voltage for the A/D converter. 10-bit A/D conversion is performed between AVR and AVss. If the A/D converter is not used, connect it to AVss.

# 11.3 Pins of the A/D Converter

**This section describes the pins related to the A/D converter and shows a block diagram.**

## ■ Pins Related to A/D Converter

The pins related to the A/D converter are P85/AN0 to P87/AN2, P90/AN3, and P10/AN4 to P17/AN11.

### ❍ P85/AN0/SW1 to P87/AN2/SW3

Pins P85/AN0/SW1 to P87/AN2/SW3 function as a general-purpose output port (P85 to P87), as analog input pins (AN0 to AN2), and as a comparator (SW1 to SW3).

### ❍ P90/AN3

Pin P90/AN3 functions as a general-purpose I/O port (P90) and as an analog input pin (AN3).

### ❍ P10/AN4 to P17/AN11

Pins P10/AN4 to P17/AN11 function as a general-purpose I/O port (P10 to P17) and as analog input pin (AN5 to AN11).

**AN0 to AN11**

To use the A/D conversion function, input the analog voltage to be converted to these pins. When the corresponding bits in the port data registers (DDR8, DDR9, DDR1) are set to 0, the output transistor is turned off, and a pin is selected with the analog input channel select bits (ADC2: ANS0 to ANS3), the selected pin functions as an analog input pin. The pins that are not used as analog input pins can be used as a general-purpose I/O port.

## ■ Block Diagram of Pins Related to the A/D Converter

**Figure 11.3-1 Block Diagram of Pins P85/AN0 to P87/AN2**

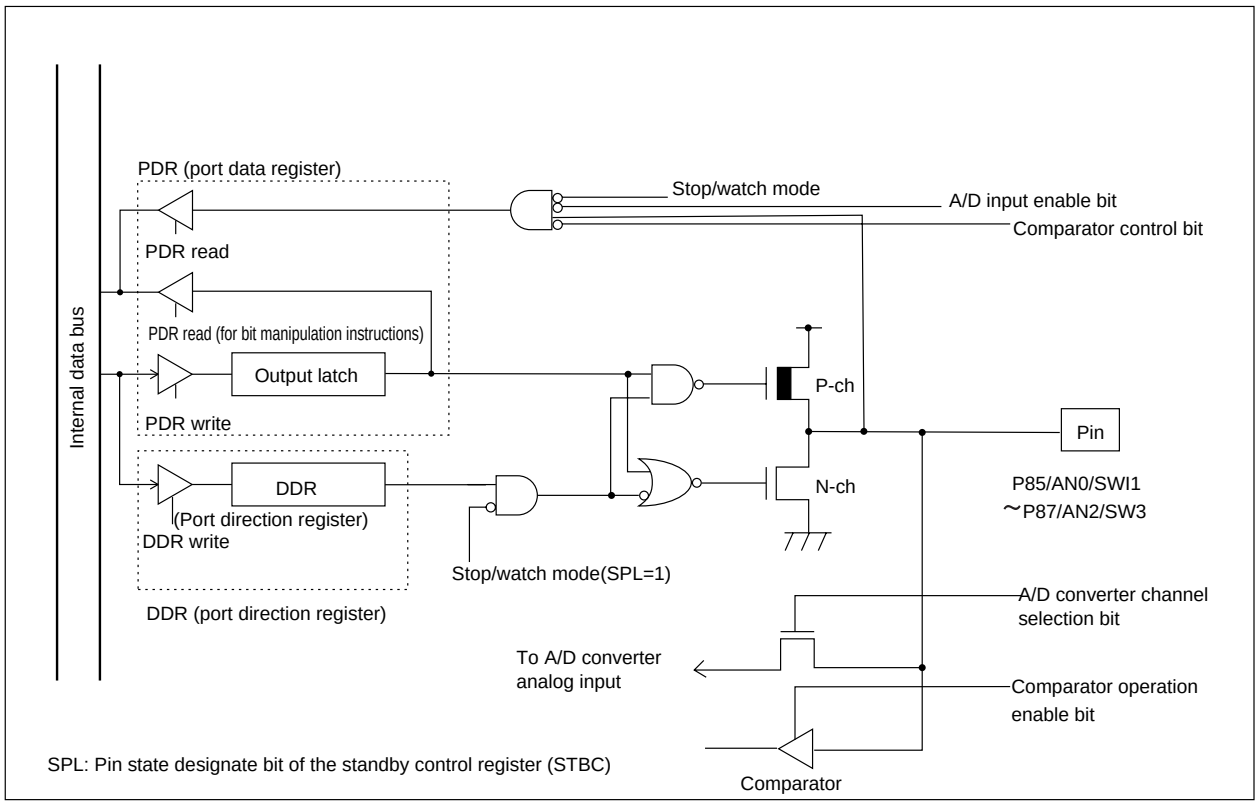
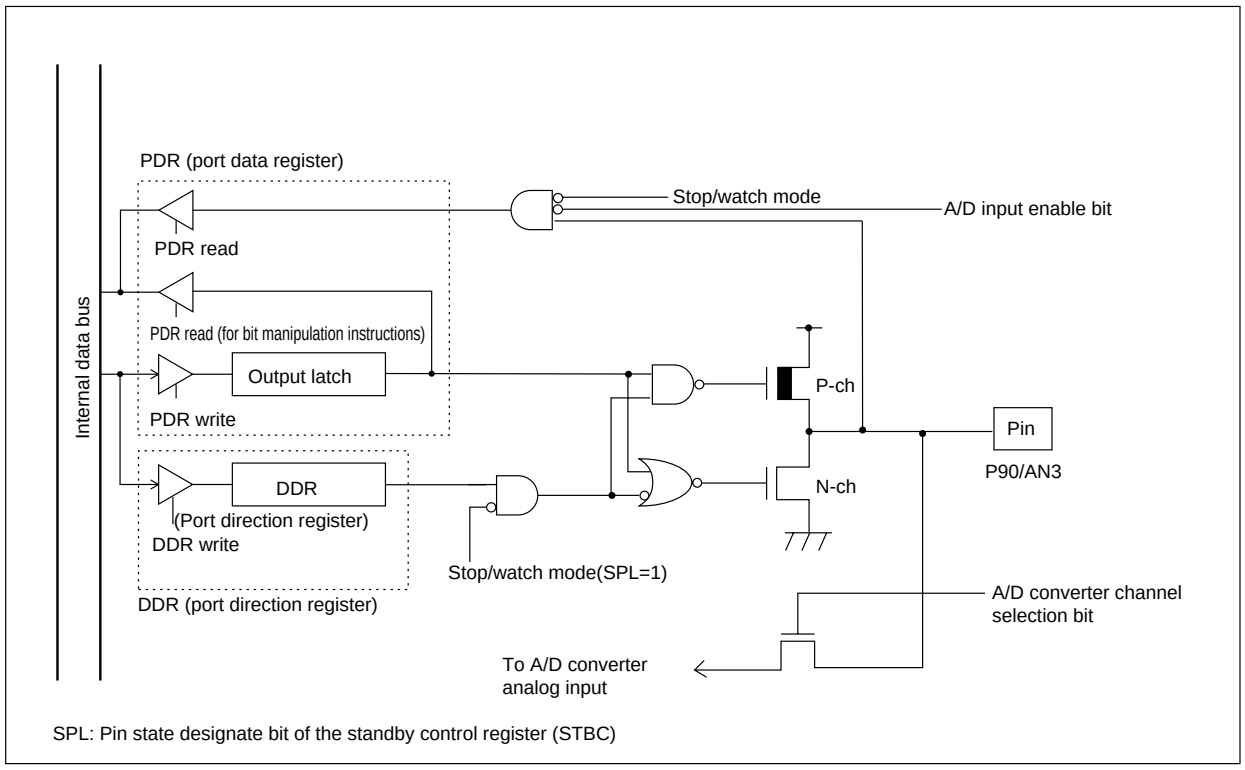
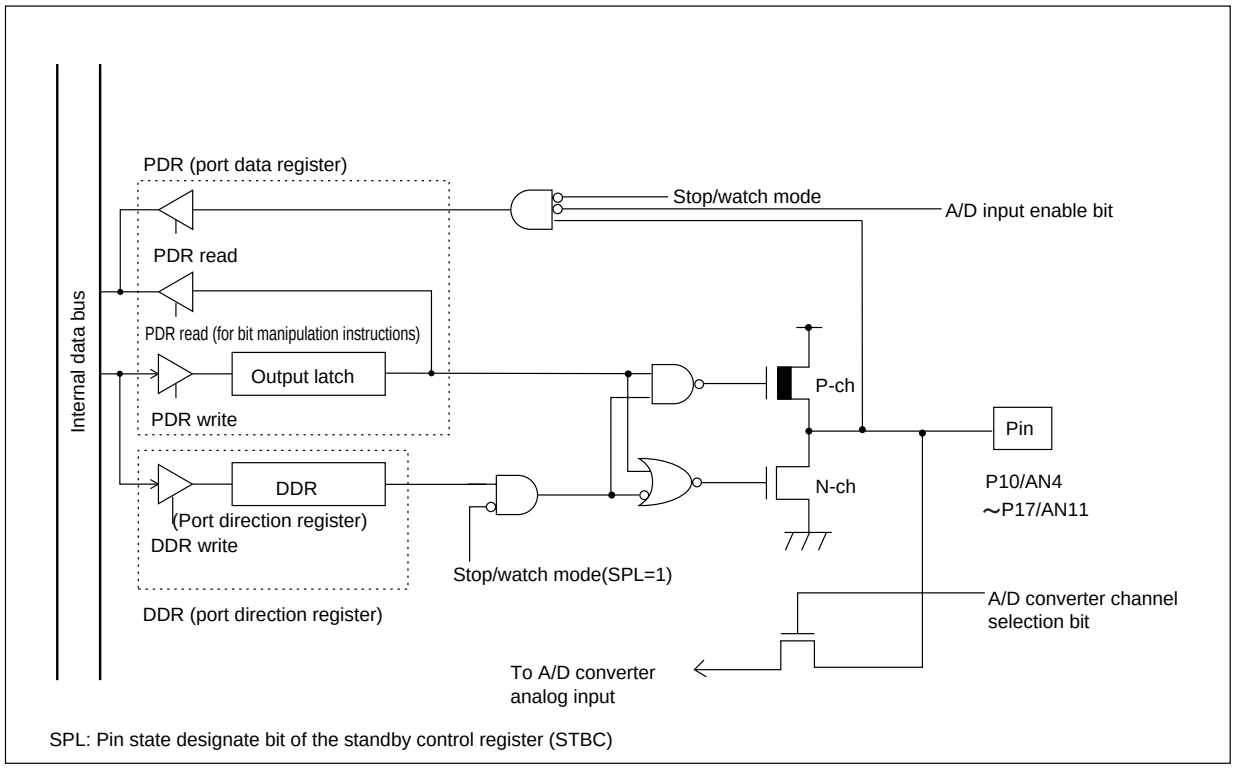


**Figure 11.3-2 Block Diagram of Pin P90/AN3**




SPL: Pin state designate bit of the standby control register (STBC)

**Figure 11.3-3 Block Diagram of Pins P10/AN4 to P17/AN11**




SPL: Pin state designate bit of the standby control register (STBC)

# 11.4 Registers of the A/D Converter

**This section shows the registers related to the A/D converter.**

## ■ Registers Related to A/D Converter

**Figure 11.4-1 Registers Related to the A/D Converter**

ADC1 (A/D control register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 2 F H | ANS3 | ANS2 | ANS1 | ANS0 | ADI | ADMV | - | AD | 000000X0B |
| | R/W | R/W | R/W | R/W | R/W | R | | R/W | |

ADC2 (A/D control register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 0 H | - | RESV | RESV | ADCK | ADIE | ADMD | EXT | RESV | X0000001B |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

ADDH, ADDL (A/D data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 2 H | - | - | - | - | - | - | D9 | D8 | XXXXXXXXB |
| | | | | | | | R/W | R/W | |
| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
| 0 0 3 3 H | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

ADEN1 (A/D enable register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 2 D H | - | - | - | - | ADE3 | ADE2 | ADE1 | ADE0 | XXXX1111B |
| | | | | | R/W | R/W | R/W | R/W | |

ADEN2 (A/D enable register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 2 E H | ADE11 | ADE10 | ADE9 | ADE8 | ADE7 | ADE6 | ADE5 | ADE4 | 11111111B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
R : Read only
X : Undefined

# 11.4.1 A/D Control Register 1 (ADC1)

**The A/D control register 1 (ADC 1) enables and disables the functions of the A/D converter and checks the state.**

## ■ A/D Control Register 1 (ADC1)

**Figure 11.4-2 A/D Control Register 1 (ADC1)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0002F H | ANS3 | ANS2 | ANS1 | ANS0 | ADI | ADMV | - | AD | 000000X0B |
| | R/W | R/W | R/W | R/W | R/W | R | | R/W | |

| AD | A/D conversion start bit |
|---|---|
| | Software start time |
| 0 | A/D conversion function has not started. |
| 1 | A/D conversion function has started. |

| ADMV | Conversion flag bit |
|---|---|
| 0 | Conversion is not in progress. |
| 1 | Conversion is in progress. |

| ADI | Interrupt request flag bit | |
|---|---|---|
| | During a read | During a write |
| 0 | Conversion has not been completed. | The bit is cleared. |
| 1 | Conversion is complete. | There is no change, and there is no effect on other bits. |

| ANS3 | ANS2 | ANS1 | ANS0 | Analog input channel select bit |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | AN0 pin |
| 0 | 0 | 0 | 1 | AN1 pin |
| 0 | 0 | 1 | 0 | AN2 pin |
| 0 | 0 | 1 | 1 | AN3 pin |
| 0 | 1 | 0 | 0 | AN4 pin |
| 0 | 1 | 0 | 1 | AN5 pin |
| 0 | 1 | 1 | 0 | AN6 pin |
| 0 | 1 | 1 | 1 | AN7 pin |
| 1 | 0 | 0 | 0 | AN8 pin |
| 1 | 0 | 0 | 1 | AN9 pin |
| 1 | 0 | 1 | 0 | AN10 pin |
| 1 | 0 | 1 | 1 | AN11 pin |

R/W : Read/write enabled
R : Read only
- : Unused
X : Undefined
□ : Initial value

**Table 11.4-1 Functions of the A/D Control Register 1 (ADC1) bits**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4 | ANS3, ANS2, ANS1, ANS0:<br>Analog input channel select bits | • These bits select a pin to be used as an analog input pin from AN0 to AN11.<br>• These bits can be rewritten when the A/D conversion function is started (AD = 1).<br>**Note:**<br>Do not rewrite these bits when the ADC1:ADMV bit is 1.<br>**Reference:**<br>The pins that are not used as analog input pins can be used as general-purpose ports. |
| Bit 3 | ADI:<br>Interrupt request flag bit | • In A/D conversion function operation, this bit is set to 1 when A/D conversion is completed.<br>• For a write operation, writing 0 clears this bit. Writing 1 has no effect and no effects on other bits. |
| Bit 2 | ADMV:<br>Conversion flag bit | • In the A/D conversion function operation, this bit indicates that conversion is being performed.<br>• During conversion, this bit is set to 1.<br>**Reference:**<br>This bit is read only. Writing to this bit has no meaning and has no effect on operation. |
| Bit 1 | Unused bit | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 0 | AD:<br>A/D conversion start bit | • This bit starts the A/D conversion function from software.<br>• Writing 1 to this bit starts the A/D conversion function.<br>**Note:**<br>• Even though 0 is written to this bit, the operation of the A/D conversion function cannot be stopped. The read value is always 0. |

## 11.4.2 A/D Control Register 2 (ADC2)

**The A/D control register 2 (ADC 2) selects the functions of the A/D converter, selects the input clock, enables and disables interrupts and continuous start, and checks the state.**

### ■ A/D Control Register 2 (ADC2)

**Figure 11.4-3 A/D Control Register 2 (ADC2)**

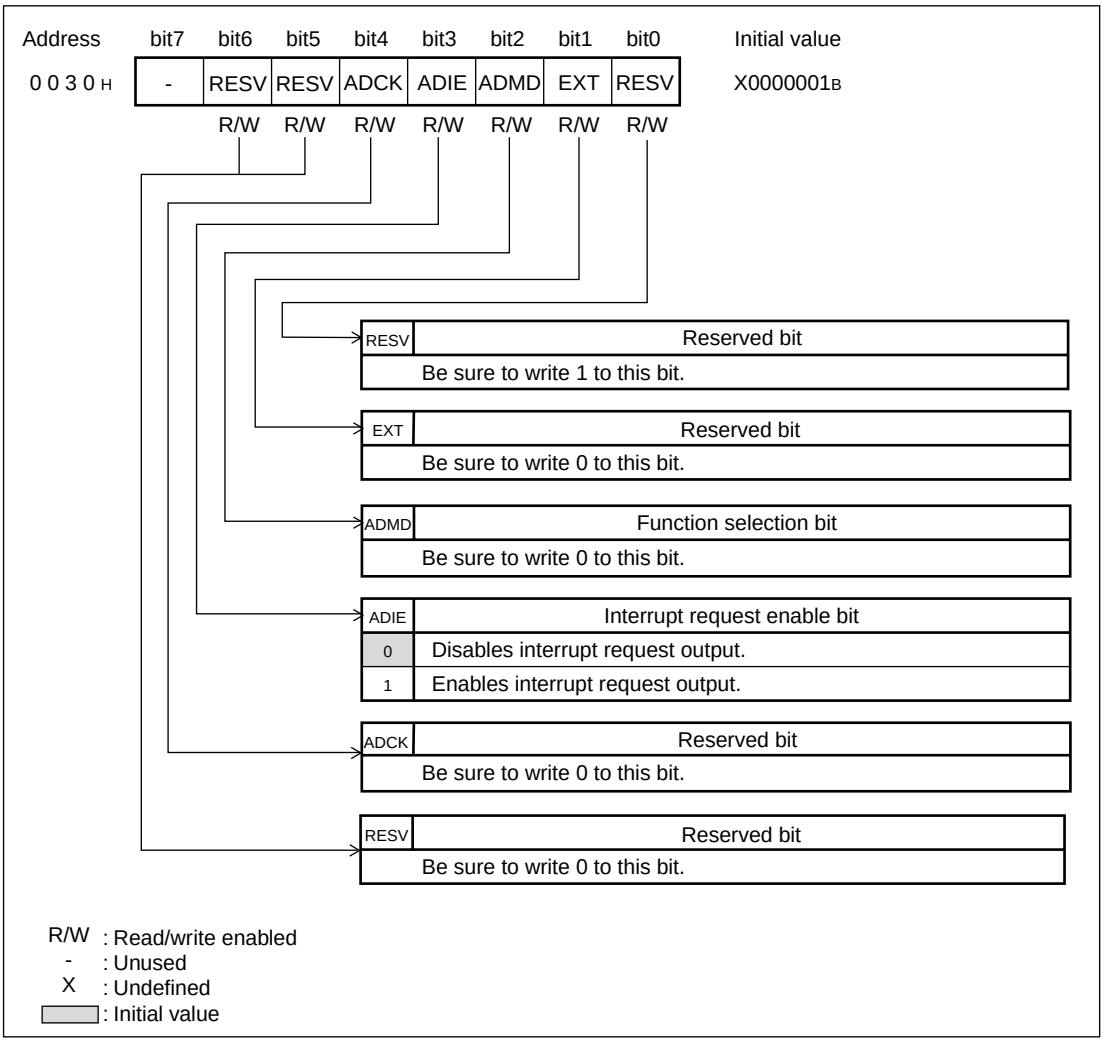


**Table 11.4-2 Functions the A/D Control Register 2 (ADC2) bits**

| Bit name | | Function |
|---|---|---|
| Bit 7 | Unused bit | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 6<br>Bit 5 | RESV:<br>Reserved bit | • Be sure to write 0 to this bit. |
| Bit 4 | ADCK:<br>Reserved bit | • Be sure to write 0 to this bit. |
| Bit 3 | ADIE:<br>Interrupt request enable bit | • This bit enables and disables interrupt output to the CPU.<br>• When this bit and the interrupt request flag bit (ADC1: ADI) are 1, an interrupt request is output. |
| Bit 2 | ADMD:<br>Reserved bit | • Be sure to write 0 to this bit. |
| Bit 1 | EXT:<br>Reserved bit | • Be sure to write 0 to this bit. |
| Bit 0 | RESV:<br>Reserved bit | **Note:**<br>• Be sure to write 1 to this bit.<br>• The read value is always 1. |

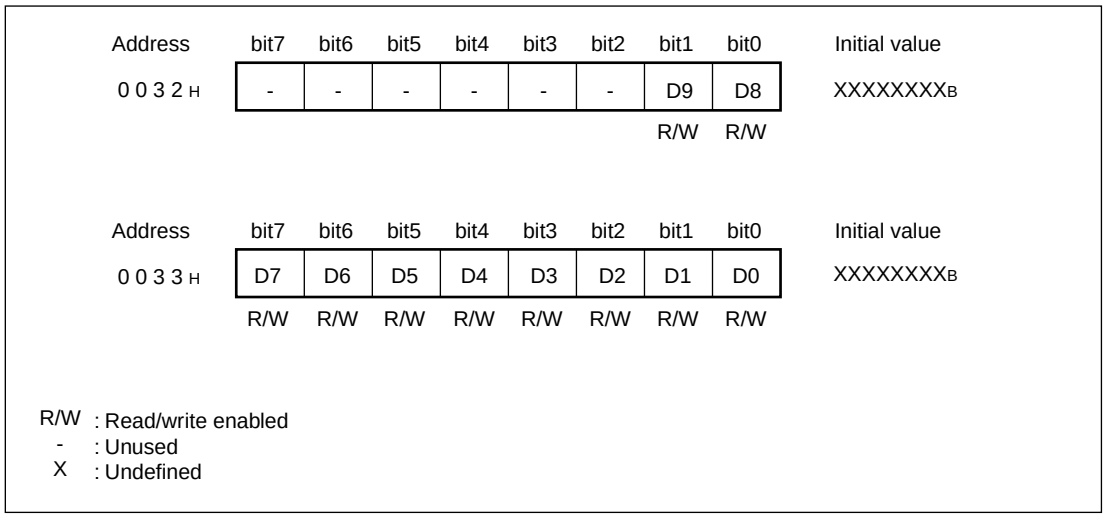
# 11.4.3 A/D Data Registers (ADDH, ADDL)

**These data registers store the result of A/D conversion.**
**The upper two bits of the 10-bit data correspond to the ADDH register, and the lower eight bits correspond to the ADDL register.**

## ■ A/D Data Registers (ADDH, ADDL)

Figure 11.4-4 "A/D Data Registers (ADDH, ADDL" shows the bit configuration of the A/D data register.

**Figure 11.4-4 A/D Data Registers (ADDH, ADDL)**




The upper two bits of the 10-bit A/D data correspond to bits 1 and 0 of the ADDH register, and the lower eight bits correspond to bits 7 to 0 of the ADDL register.

### ❍ A/D conversion cycle

When A/D conversion is started, the conversion result data is determined after about 60 instruction cycles and is stored in these registers. Between the completion of A/D conversion and the start of the next A/D conversion cycle, read the contents of these registers (conversion result), write 0 to ADI (bit 3) of the ADC1 register, and clear the flag when conversion is complete.

# 11.4.4 A/D Enable Registers 1 to 2 (ADEN 1 to 2)

**These registers specify the port used for performing A/D conversion.**

## ■ A/D Enable Registers 1 to 2 (ADEN1 to 2)

Figure 11.4-5 "A/D Enable Registers 1 to 2 (ADEN 1 to 2)" shows the bit configuration of the A/D enable registers.

**Figure 11.4-5 A/D Enable Registers 1 to 2 (ADEN 1 to 2)**

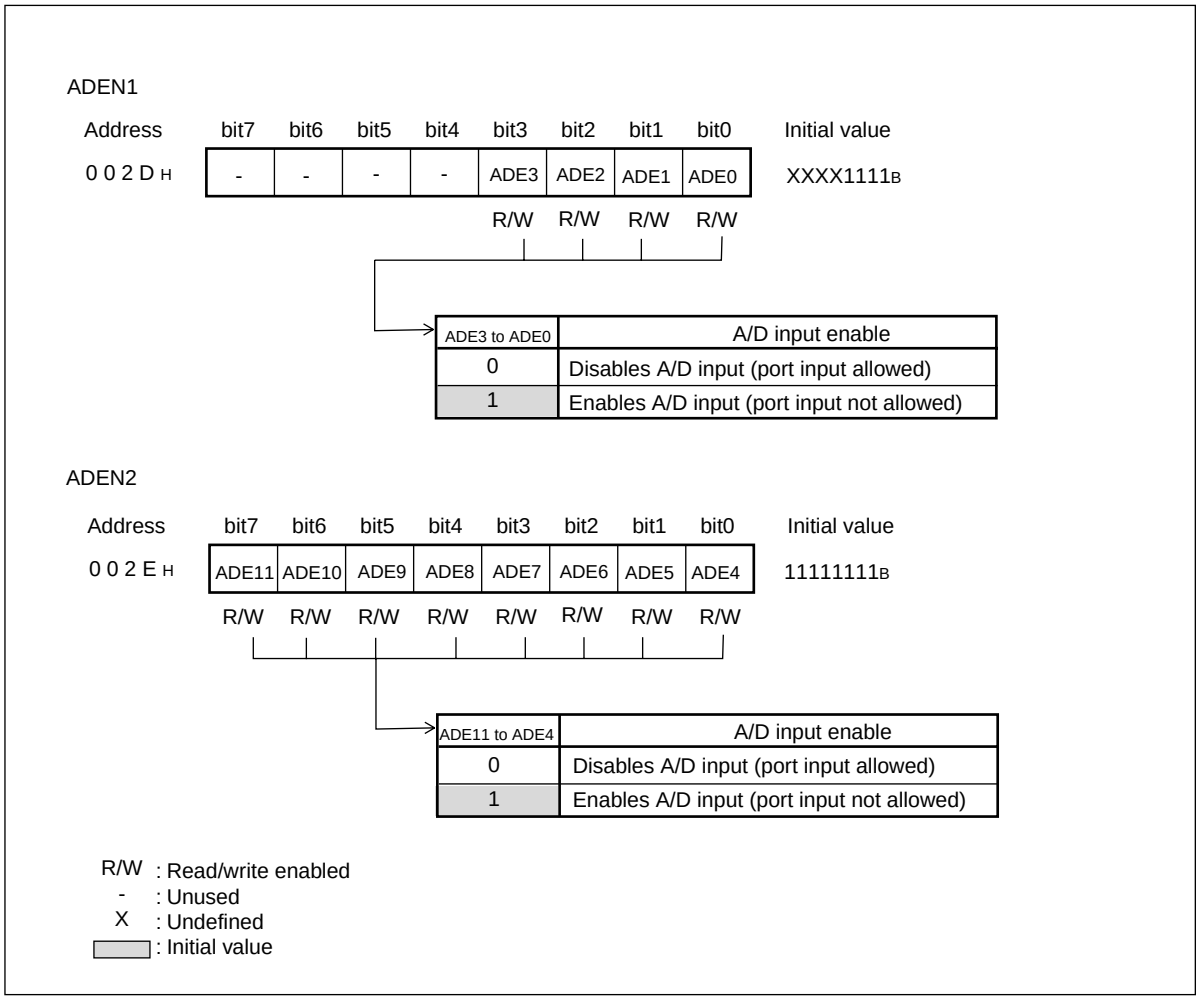


**Table 11.4-3 Functions of the A/D Enable Register 1 (ADEN1) Bits**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 3 | ADE3: AN3 enable bit | • Selects P90 for analog input (AN3). |
| Bit 2 | ADE2: AN2 enable bit | • Selects P87 for analog input (AN2). |
| Bit 1 | ADE1: AN1 enable bit | • Selects P86 for analog input (AN1). |
| Bit 0 | ADE0: AN0 enable bit | • Selects P85 for analog input (AN0). |

**Table 11.4-4 Functions of the A/D Enable Register 2 (ADEN2) Bits**

| Bit name | | Function |
|---|---|---|
| Bit 7 | ADE11: AN11 enable bit | • Selects P17 for analog input (AN11). |
| Bit 6 | ADE10: AN10 enable bit | • Selects P16 for analog input (AN10). |
| Bit 5 | ADE9: AN9 enable bit | • Selects P15 for analog input (AN9). |
| Bit 4 | ADE8: AN8 enable bit | • Selects P14 for analog input (AN8). |
| Bit 3 | ADE7: AN7 enable bit | • Selects P13 for analog input (AN7). |
| Bit 2 | ADE6: AN6 enable bit | • Selects P12 for analog input (AN6). |
| Bit 1 | ADE5: AN5 enable bit | • Selects P11 for analog input (AN5). |
| Bit 0 | ADE4: AN4 enable bit | • Selects P10 for analog input (AN4). |

The A/D input ports are also used as general-purpose I/O ports.

For the ports used for analog input, enter a 0 for the corresponding bits in the ADEN register. This setting prevents the DC path from being made on the A/D input port when a middle level voltage is applied to the A/D input port.

# 11.5 A/D Converter Interrupt

## The following function is provided as the A/D converter.

- **End of conversion in A/D conversion function operation**

### ■ Interrupt for A/D Conversion Function

When A/D conversion is completed, the interrupt request flag bit (ADC: ADI) is set to 1. If the interrupt request enable bit is then set to enable (ADC: ADIE=1), an interrupt request to the CPU (IRQ3) occurs. Clear the interrupt request by writing 0 to the ADI bit with the interrupt processing routine.

The ADI bit is set when A/D conversion is completed irrespective of the value of the ADIE bit.

If the ADIE bit is changed from "disable" to "enable" (0 --> 1) when the ADI bit is set to 1, an interrupt request occurs immediately.

### ■ Register and Vector Table Related to A/D Converter Interrupt

**Table 11.5-1 Register and Vector Table Related to the A/D Converter Interrupt**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Higher | Lower |
| IRQ3 | ILR1 (007B) | L31 (bit 7) | L30 (bit 6) | $FFF4_H$ | $FFF5_H$ |

For the operation of interrupts, see Section 3.4.2 "Interrupt Processing".

# 11.6 Operation of the A/D Converter

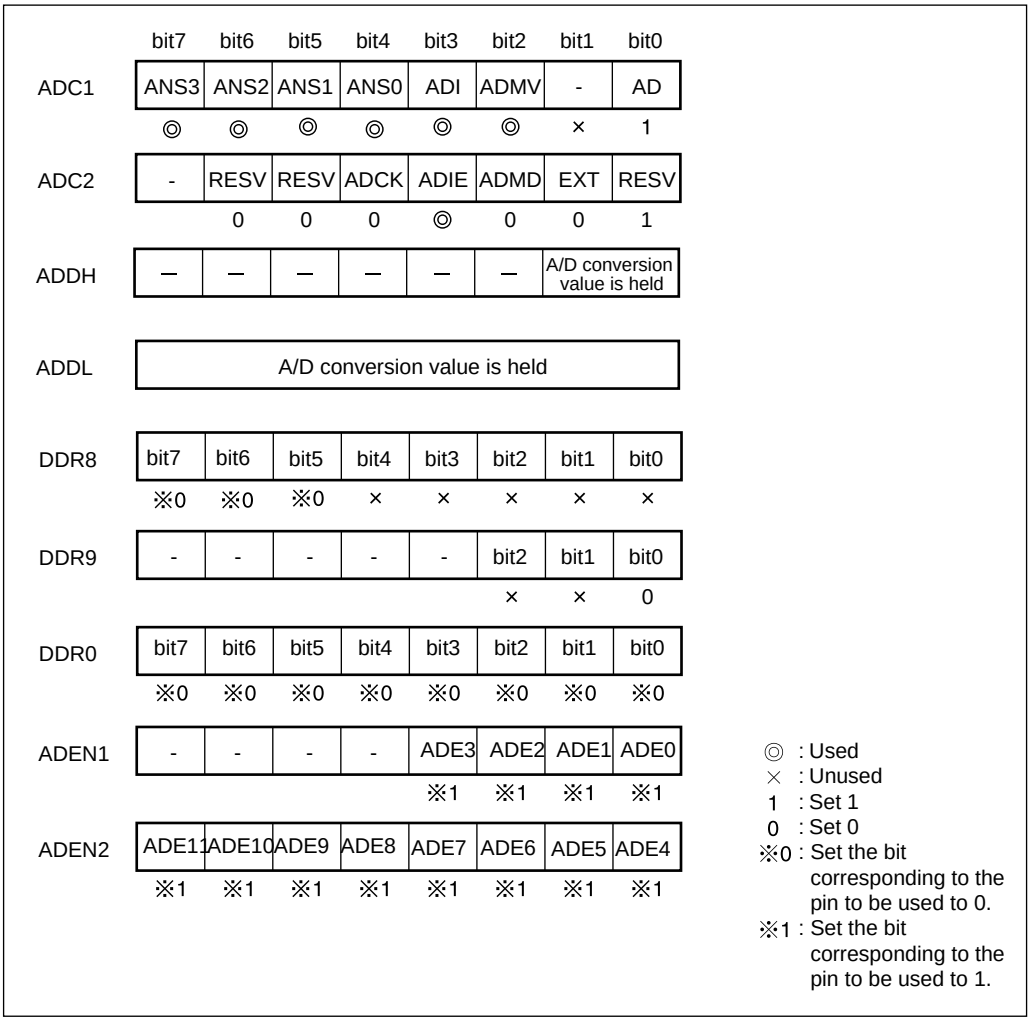
## The A/D converter is started by software.

### ■ Starting the A/D Conversion Function

#### ❍ Starting software

Starting of the software for the A/D conversion function requires the setting shown in Figure 11.6-1 "Setting the A/D Conversion Function (When Software Starts".

**Figure 11.6-1 Setting the A/D Conversion Function (When Software Starts)**




When A/D conversion is started, the A/D conversion function starts operation. The A/D conversion function can be restarted even during conversion.

## ■ Operation of A/D Conversion Function

A/D converter operation is described in the following. The time from start to end of the A/D conversion is about 60 instruction cycles.

When A/D conversion is started, the conversion flag bit is set (ADC1: ADMV = 1) and the set analog input pin is connected to the sample hold circuit.

The voltage on the analog input pin is added to the internal sample hold capacitor for about 16 instruction cycles. This voltage is held until A/D conversion is complete.

The comparator compares the voltage added to the sample hold capacitor with the reference voltage for A/D conversion in order from the most significant bit (MSB) to the least significant bit (LSB), and each comparison result is transferred on at a time to the ADDH and ADDL registers.

After the results have been transferred, the conversion flag bit is cleared (ADC1: ADMV = 0) and the interrupt request flag bit is set (ADC1: ADI = 1).

# 11.7 Notes on Using the A/D Converter

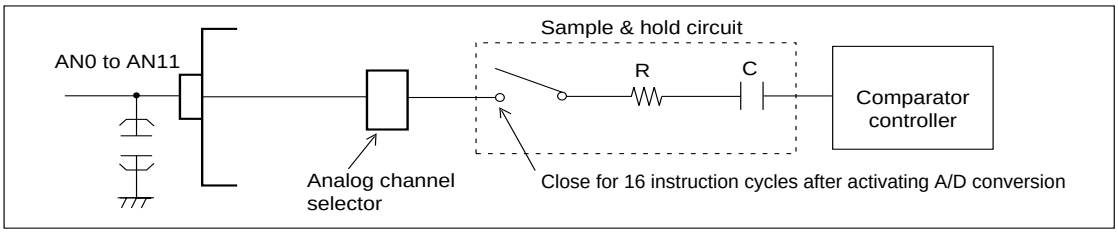
**This section contains notes on using the A/D converter.**

## ■ Notes on Using the A/D Converter

### ❍ Input impedance of the analog input pins

The A/D converter contains the sample hold circuit shown in Figure 11.7-1 "Equivalent Circuit for Analog Input" and adds the voltage of the analog input pin to the sample hold capacitor in about 16 instruction cycles after A/D conversion starts. Therefore, if the output impedance of the external circuit for analog input is high, the analog input voltage may not be stabilized within the analog input sampling period. To avoid this problem, keep the output impedance of the external circuit low enough to stabilize the voltage. If the output impedance of the external circuit cannot be kept low, it is recommended that an external capacitor of about 0.1 µF be added to the analog input pin.

**Figure 11.7-1 Equivalent Circuit for Analog Input**




It shows the R and C values of sample and hold circuit.

**Table 11.7-1 R and C values of sample and hold circuit**

| | MB89P579A | MB89577<br>MB89PV570 |
|---|---|---|
| R: Analog input equivalent resistance | 7.1 kΩ | 2.2 kΩ |
| C: Analog input equivalent capacitance | 48.3 pF | 45 pF |

### ❍ Notes on setting with a program

- In A/D conversion function operations, the ADDH and ADDL registers hold the previous values until A/D conversion is started. As soon as A/D conversion is started, the contents of the ADDH and ADDL registers become undefined.
- During operation of the A/D conversion function, do not reselect an analog input channel (ADC1: ANS3 to ANS0).
- Resetting, stopping, or starting the watch mode stops the A/D converter.
- If the interrupt request flag bit (ADC1: ADI) is set to 1 and the interrupt request is enabled (ADC2: ADIE = 1), it is not possible to return from interrupt processing.
  - Be sure to clear the ADI bit.

❍ **Note on interrupt requests**

When A/D conversion (ADC1: AD = 1) is started and completed at the same time, the interrupt request flag bit (ADC1: ADI) is not set.

❍ **About errors**

As |AVR-AVss| decreases, errors become relatively larger.

❍ **Order of applying A/D converter power and analog input**

Concurrently with or before turning on power to the A/D converter and applying the analog input (AN0 to AN11), turn on the digital power (Vcc).

Concurrently with or after turning off A/D converter power (AVcc, AVss) and disconnecting analog input (AN0 to AN11), turn off the digital power (Vcc).

When turning power to the A/D converter on and off, be careful that AVcc, AVss, and analog input do not exceed the voltage of the digital power.

❍ **Conversion rate**

The conversion rate of the A/D conversion function is influenced by the clock mode and the rate switching of the main clock (gear function).

# 11.8 Program Example of the A/D Converter

**This section contains a sample program for the 10-bit A/D converter.**

## ■ Program Example of the A/D Conversion Function

### ❍ Processing specification

The analog voltage input to the AN4 pin is digitized by starting of the software.

An interrupt is not used, and the completion of conversion is detected by the loop in the program.

❍ **Coding example**

```
DDR1    .EQU     0003H       ; Address of the port register

ADC1    .EQU     002FH       ; Address of the A/D control register 1
ADC2    .EQU     0030H       ; Address of the A/D control register 2
ADDH    .EQU     0032H       ; Address of the A/D data register H
ADDL    .EQU     0033H       ; Address of the A/D data register L
ADEN    .EQU     002EH       ; A/D port input enable register
AN4_PCR .EQU     DDR1:0      ; Definition of the AN5 analog input use
ADI     .EQU     ADC1:3      ; Definition of the interrupt request
                               flag bit
ADMV    .EQU     ADC1:2      ; Definition of the conversion flag bit
AD      .EQU     ADC1:0      ; Definition of the A/D conversion start
                               bit (software start)
;----Main program-------------------------------------------------------
        .SECTION CSEG, CODE, ALIGN=1     ; [CODE SEGMENT]
          :
        CLRI                ; Interrupt disable
        SETB     AN4_PCR    ; Set the P10/AN4 pin as an analog
                               input pin
AD_WAIT
        BBS      ADMV,AD_WAIT ; A/D converter stop check loop
        MOV      ADC1,#01001000B ; Select analog input channel 4
                                   (AN4), clearing the interrupt
                                   request flag, and set that
                                   software starting is not used
        MOV      ADEN,#11111110B ; Disable interrupt request output,
                                   and select the A/D conversion
                                   function.
                                   Conversion time: Approx. 24µs
                                   (at 10 MHz)
        SETI                ; Interrupt enable
        SETB     AD         ; Start the software
AD_CONV
        BBS      ADMV,AD_CONV ; A/D conversion end wait loop
                                 [Approx. 24µs (at 10 MHz)]
        CLRB     ADI        ; clearing the interrupt request flag
        MOVE     A,ADDH     ; Read the A/D conversion data
                               (upper 2 bits)
          :
        ENDS
;----------------------------------------------------------------------
        .END
```

# CHAPTER 12 D/A CONVERTER

**This chapter describes the functions and operations of the D/A converter.**


12.1 "Overview of the D/A Converter"

12.2 "Configuration of the D/A Converter"

12.3 "Pins of the D/A Converter"

12.4 "Registers of the D/A Converter"

12.5 "Operation of the D/A Converter"

# 12.1 Overview of the D/A Converter

**The D/A converter is an eight-bit converter that uses the R-2R method. Two built-in D/A converters for two channels are provided, and D/A control registers control the output of each independently.**

## ■ D/A Converter

Two built-in, eight-bit D/A converters for two channels are provided. The two converters can operate independently. The maximum output voltage value of the D/A converters varies depending on the reference voltage value input to the AVCC pin.

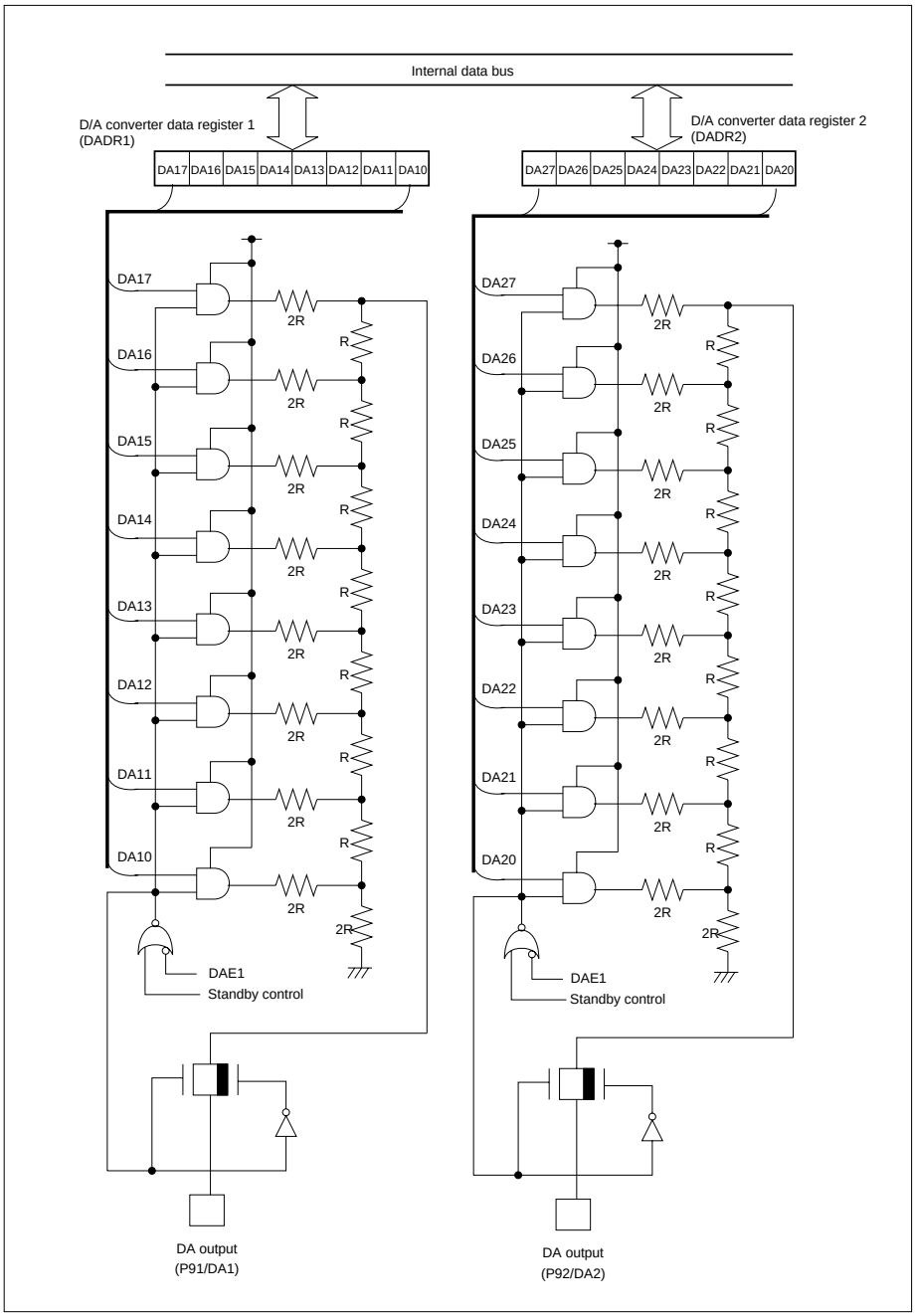
- D/A converter 1: DA1
- D/A converter 2: DA2

# 12.2 Configuration of the D/A Converter

**Figure 12.2-1 "D/A Converter Block Diagram" shows the block diagram of the D/A converters.**

## ■ Block Diagram of the D/A Converter Block

**Figure 12.2-1 D/A Converter Block Diagram**



# 12.3 Pins of the D/A Converter

**This section describes the pins related to the D/A converters (DA1, 2 and shows a block diagram.**

## ■ Pins Related to the D/A Converters (DA1, 2)

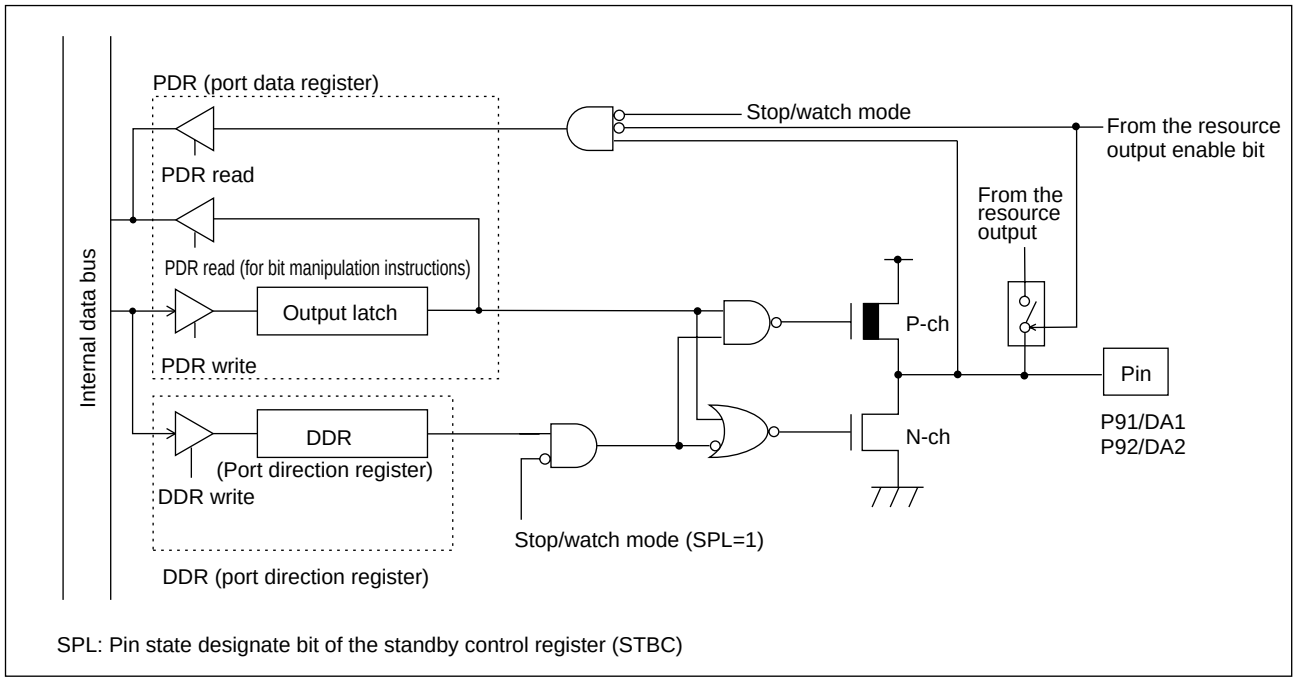
The pins related to the D/A converters include the DA1, DA2, $AV_{CC}$, and $AV_{SS}$ pins.

The $AV_{CC}$ and $AV_{SS}$ pins supply the analog power for the D/A converters as well as the analog power for the A/D converter.

- DA1: Output pin for D/A converter 1. Also used as a general-purpose input-output port (P91).
- DA2: Output pin for D/A converter 2. Also used as a general-purpose input-output port (P92).

## ■ Block Diagram of the Pins Related to the D/A Converters (DA1, 2)

**Figure 12.3-1 Block Diagram of the Pins Related to the D/A Converters (DA1, 2)**




SPL: Pin state designate bit of the standby control register (STBC)

# 12.4 Registers of the D/A Converter

**This section describes the registers related to the D/A converter.**

## ■ Registers Related to the D/A Converter

**Figure 12.4-1 D/A Converter Registers**

DACR (D/A control register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 7 0 H | - | - | - | - | - | - | DAE2 | DAE1 | XXXXXX00B |
| | | | | | | | R/W | R/W | |

DADR1 (D/A data register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 7 1 H | DA17 | DA16 | DA15 | DA14 | DA13 | DA12 | DA11 | DA10 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

DADR2 (D/A data register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 7 2 H | DA27 | DA26 | DA25 | DA24 | DA23 | DA22 | DA21 | DA20 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

# 12.4.1 D/A Control Register

**The D/A control register enables ands disables the output of the D/A converters 1 and 2.**

■ **D/A Control Register (DACR)**

**Figure 12.4-2 D/A Control Register (DACR)**

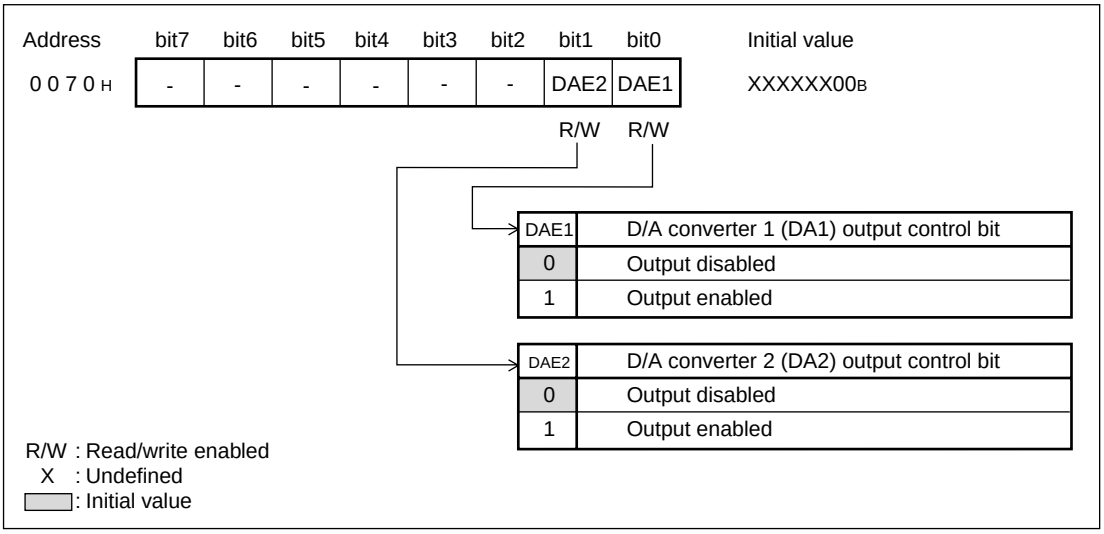



**Table 12.4-1 Functions of the D/A Control Register (DACR) Bits**

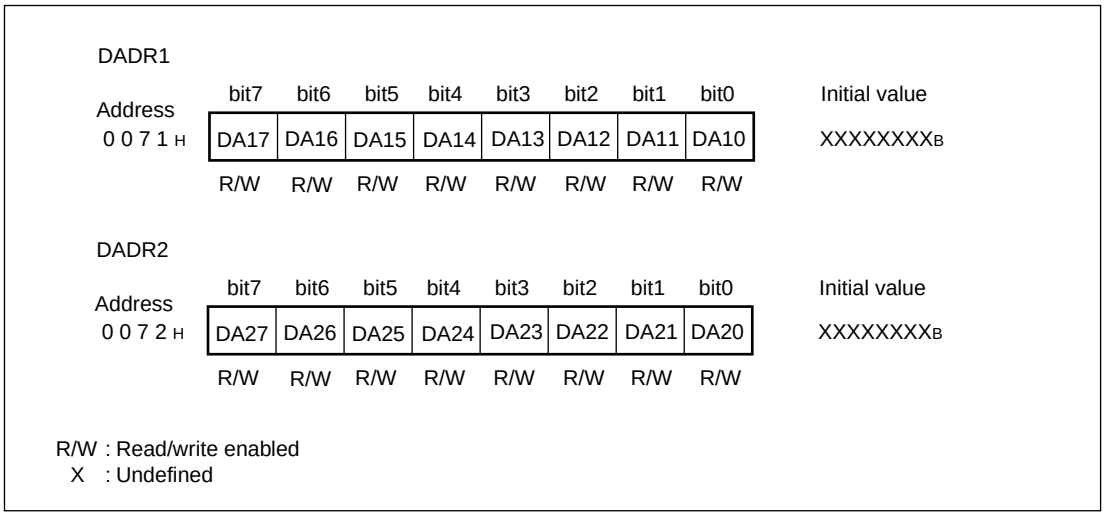
| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4<br>Bit 3<br>Bit 2 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 1 | DAE2: D/A converter 2 control bit | • Controls the output of D/A converter 2 (DA2)<br>• Enables D/A output if set to 1. Disables D/A output<br>• if set to 0.<br>• Initialized to 0 if reset.<br>• Can be read and written. |
| Bit 0 | DAE1: D/A converter 1 control bit | • Controls the output of D/A converter 1 (DA1)<br>• Enables D/A output if set to 1. Disables D/A output if set to 0.<br>• Initialized to 0 if reset.<br>• Can be read and written. |

## 12.4.2 D/A Data Registers 1 and 2 (DADR1, 2)

**The D/A data registers set the output voltage of the D/A converters.**

■ **D/A Data Registers 1 and 2 (DADR1, 2)**

**Figure 12.4-3 D/A Data Registers 1 and 2 (DADR1, 2)**




**DADR1 [DA17 to DA10]:**

Sets the output voltage of D/A converter 1 (DA1). Not initialized if reset.

**DADR2 [DA27 to DA20]:**

Sets the output voltage of D/A converter 2 (DA2). Not initialized if reset.

# 12.5 Operation of the D/A Converter

**To start D/A output, set the D/A data register (DADR) to a desired D/A output value and set the corresponding D/A output channel enable bit in the D/A control register (DACR) to 1.**

## ■ D/A Converter Operation

Disabling D/A output turns off the analog switch inserted in series into the output section of a channel of the D/A converter. It also clears the D/A converter to 0 output status and shuts off any channel where direct current runs. The same operation also occurs in stop mode.

The output of this D/A converter does not contain a buffer amplifier. Additionally, an analog switch is inserted in series into the output. Take sufficient precautions, therefore, regarding external output load in consideration of the required settling time.

The D/A converter has an output voltage range from 0 V to 255/256 x DVR. Externally adjust the DVR voltage to change the output voltage range.

Table 12.5-1 "Logical Values of the D/A Converter Output Voltage" shows the logical values of the D/A converter output voltage.

**Table 12.5-1 Logical Values of the D/A Converter Output Voltage**

| Setting value of DADR1 [DA17 to DA10] and DADR2 [DA27 to DA20] | Logical values of the output voltage |
|---|---|
| 00H | 0/256 x DVR (=0V) |
| 01H | 1/256 x DVR |
| 02H | 2/256 x DVR |
| to | to |
| FDH | 253/256 x DVR |
| FEH | 254/256 x DVR |
| FFH | 255/256 x DVR |

# CHAPTER 13 COMPARATOR

**This chapter describes the function and operation of the comparator.**


13.1 "Overview of the Comparator"

13.2 "Configuration of the Comparator"

13.3 "Pins of the Comparator"

13.4 "Registers of the Comparator"

13.5 "Comparator Interrupts"

13.6 "Operation of the Parallel Discharge Control"

13.7 "Operation of the Sequential Discharge Control"

13.8 "Sample Application"

# 13.1 Overview of the Comparator

**This comparator is a circuit that monitors voltage of up to three batteries and automatically controls electric discharge. Either parallel discharge or sequential discharge can be selected.**

## ■ Parallel Discharge Control

In parallel discharge control, all batteries are allowed to discharge when power is not being supplied from the AC adapter.

- If power is being supplied from the AC adapter, the permission/prohibition of discharge for batteries is controlled by software.

## ■ Sequential Discharge Control

In sequential discharge control, the comparator controls discharge in a specified order, while monitoring intermittent interruption of power, voltage level, and mount/dismount of batteries, when power is not being supplied from the AC adapter.

- If power is being supplied from the AC adapter, the permission/prohibition of discharge for batteries is controlled by software.
- Up to three batteries can be controlled, and the order of discharge can be selected.
  - The affect of intermittent interruption of power is automatically filtered.
  - Mount/dismount of batteries is automatically detected and discharge is controlled.
  - Battery voltage is monitored, and if battery voltage is below the specified voltage, change over to the next battery is automatically done.

See Section 13.8 "Sample Application".

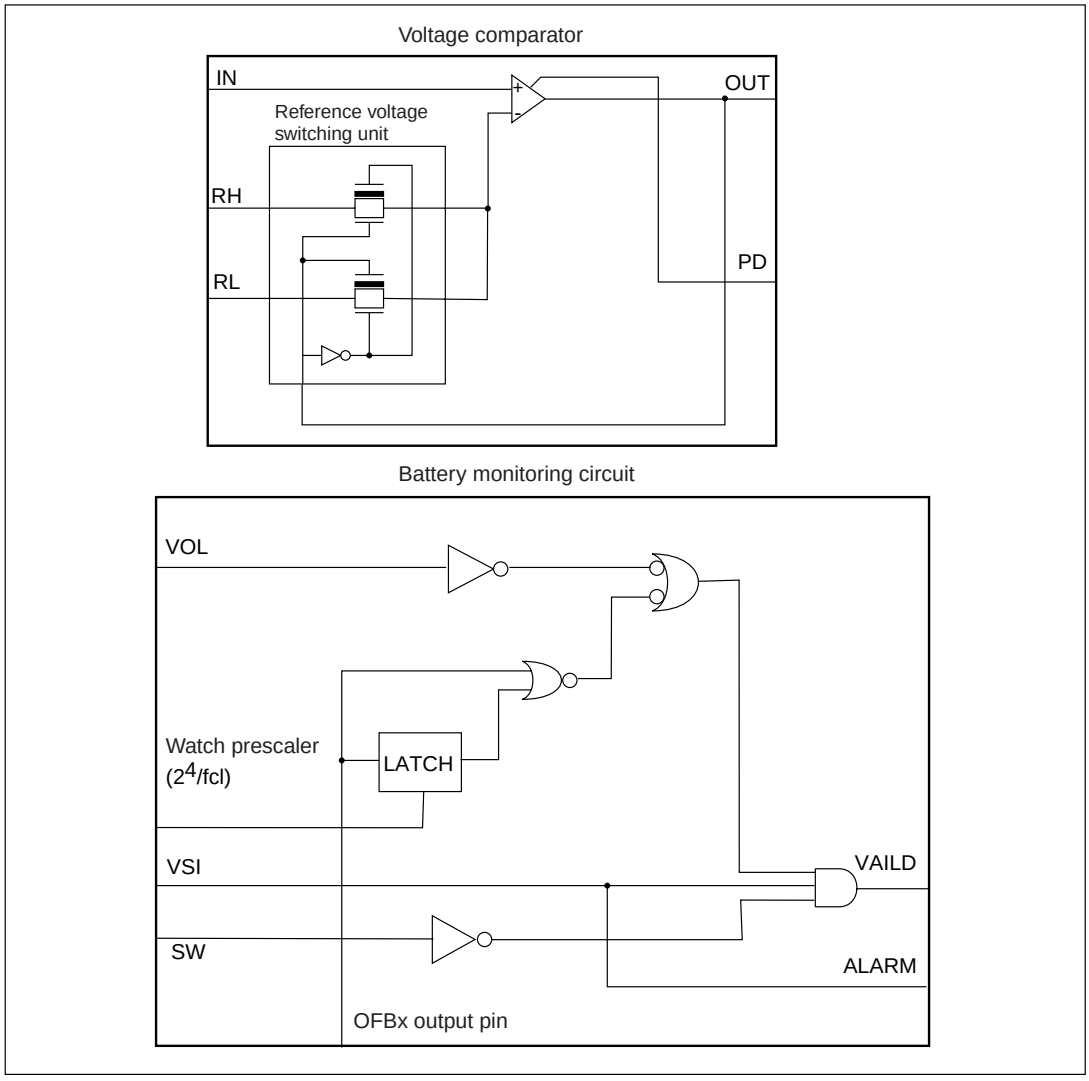
# 13.2 Configuration of the Comparator

**This comparator consists of following seven blocks:**

- **Voltage comparator**
- **Battery monitoring circuit**
- **Battery selection circuit**
- **Comparator control registers 1 to 2 (COCR2)**
- **Comparator status registers 1 to 3 (COSR1 to 3)**
- **Comparator interrupt control registers 1 to 2 (CICR1 to 2)**

## ■ Comparator Block Diagram

**Figure 13.2-1 Comparator Block Diagram**



**Voltage comparator**

The voltage comparator uses the RH and RL pins, as reference voltage input pins for the reference voltage switching units. This implements a comparator with RH-RL hysteresis width.

**Battery monitoring circuit**

The battery monitoring circuit monitors power supplied from the AC adapter and the intermittent interruption of power (VOL), voltage level (VSI), and mount/dismount of batteries. When battery discharge starts, the battery monitoring circuit automatically generates an approximately 250µs to 500µs of delay from change of VOL by the subclock.

**Battery selection circuit**

The battery selection circuit controls the selection of a battery discharge method and the order of discharge, based on the values set in the COCR register 1 (COCR1:SPM0, 1, 2).

**Comparator control register 1 (COCR1)**

This register is used to set a sequence for the battery selection circuit and to disable battery discharge.

**Comparator control register 2 (COCR2)**

This register is used to allow operation of the comparator and control output from ports.

**Comparator status register 1 (COSR1)**

This register holds and stores an edge change in signals that are output from comparator 1 and voltage comparators 2 to 8. This allows confirmation of a battery voltage change.

**Comparator interrupt control register 1 (CICR1)**

When an edge change is detected in an output signal by the COSR1 register, this register allows an interrupt to be generated.

**Comparator status register 2 (COSR2)**

This register holds and stores output from comparators 2 to 4 and an edge change in a VALID signal from the battery monitoring circuit. This allows the detection of mounting/ dismounting of batteries and the status changes of batteries.

**Comparator interrupt control register 2 (CICR2)**

The CICR2 controls enabling an interrupt when the outputs from comparators 2 to 4 together with the change of edges of VALID signal from the battery monitoring circuit are seized by the COSR2 register.

**Comparator status register 3 (COSR3)**

This register stores a VALID signal value that is output from the battery monitoring circuit . This allows a VALID signal output value to be confirmed.

**Comparator status register 4 (COSR4)**

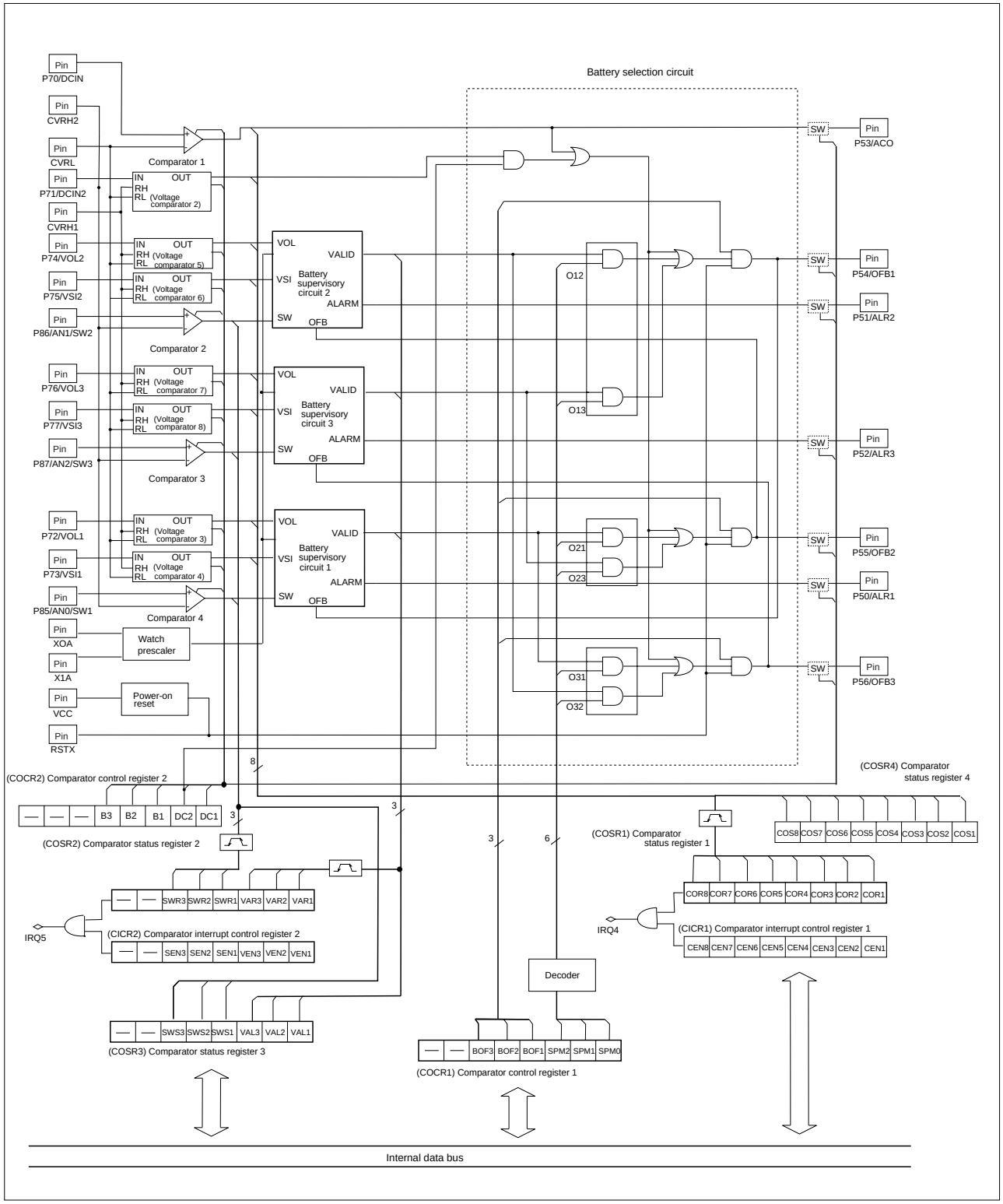
This register can directly read values that are output from comparator 1 and voltage comparators 2 to 8.

**Interrupts associated with the comparator**

IRQ4: When a change is detected in a output signal that is output from comparator 1 and voltage comparators 2 to 8, an interrupt is generated if interrupt generation has been allowed (CICR1).

IRQ5: When a change is detected in a VALID signal that is output from the battery monitoring circuit or when a change is detected in the output from the comparators for SW1 to SW3, an interrupt request is generated if generation of an interrupt has been allowed (CICR2).

## Figure 13.2-2 Comparator Block Diagram



# 13.3 Pins of the Comparator

**This section describes the pins associated with the comparator. It also shows a block diagram of pins.**

## ■ Pins Associated with the Comparator

The Pins associated with the comparator are as follows:

P50/ALR1, P51/ALR2, P52/ALR3, P53/ACO, P54/OFB1, P55/OFB2, P56/OFB3, P70/DCIN, P71/DCIN2, P72/VOL1, P73/VSI1, P74/VOL2, P75/VSI2, P76/VOL3, P77/VSI3, P85/AN0/SW1, P86/AN1/SW2, P87/AN2/SW3, CVRH1, CVRH2, CVRL.

Each of these pins acts as a general-purpose port or a comparator.

**P50/ALR1 to P52/ALR3:**

Each of these pins acts as a general-purpose I/O port (P50 to P52) or as an alarm signal output pin (ALR1 to ALR3) used when the battery charge is running low.

**P53/ACO:**

This pin acts as a general-purpose I/O port (P53) or as an output pin (ACO) used when ON/OFF of the comparator AC adapter power supply is detected.

**P54/OFB1 to P56/OFB3:**

Each of these pins acts as a general-purpose I/O port (P54 to P56) or as a battery discharge control signal output pin (OFB1 to OFB3). For "OFB1 to 3" output, discharge is allowed by "L".

**P85/AN0/SW1 to P87/AN2/SW3:**

Each of theses pins acts as a general-purpose I/O port (P85 to P87), as an input pin (SW1 to SW3) for the detection of the comparator voltage mount/dismount, or as an A/D input pin.

**P70/DCIN:**

This pin acts as a general-purpose I/O port (P70) or as an input pin (DCIN) for the detection of mount/dismount of the comparator AC adapter power supply.

**P71/DCIN2:**

Each of these pins acts as a general-purpose I/O port (P71) or as an input pin (DCIN2) for the detection of mount/dismount of the comparator AC adapter power supply.

**P72/VOL1, P74/VOL2, P76/VOL3:**

Each of these pins acts as a general-purpose I/O port (P72, P74, P76) or as an input pin (VOL1, VOL2, VOL3) for the detection of battery intermittence.

**P73/VSI1, P75/VSI2, P77/VSI3:**

Each of these pins acts as a general-purpose I/O port (P73, P75, P77) or as an input pin (VSI1, VSI2, VSI3) for comparator battery remainder monitoring.

**CVRH1, CVRH2, CVRL:**

These pins are used as the reference power supply pins for voltage comparators of the comparator. Hysteresis characteristics can be attached during voltage comparison, depending on the input voltage level.

**Note:**

These pins operate as general-purpose ports when the personal computer is activated. To use these pins as a comparator function, use the comparator control register 2 (COCR2) to allow these pins to function for the comparator.

## ■ Block Diagram of Pins Associated with the Comparator

**Figure 13.3-1 Block Diagram of Pins Associated with the Comparator**

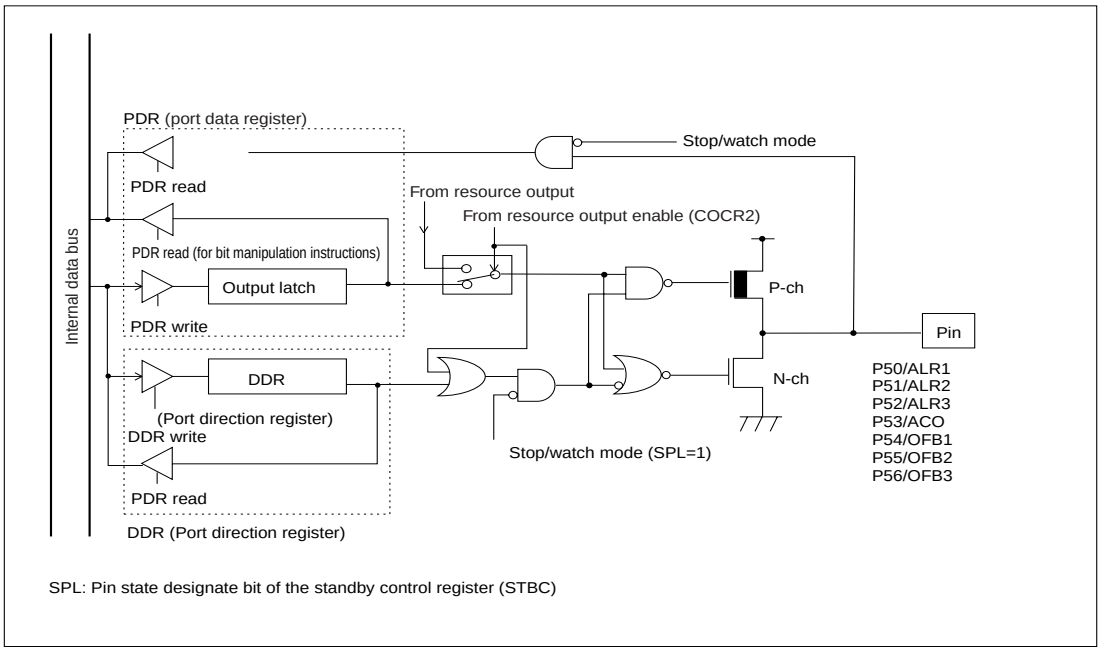



**Figure 13.3-2 Block Diagram of Pins Associated with the Comparator**

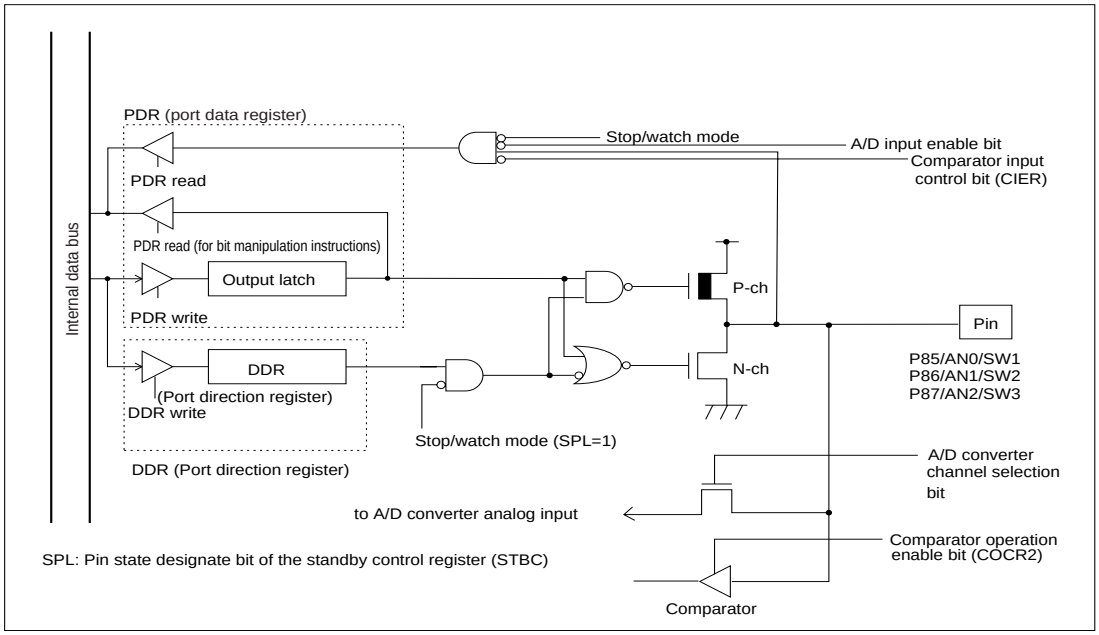
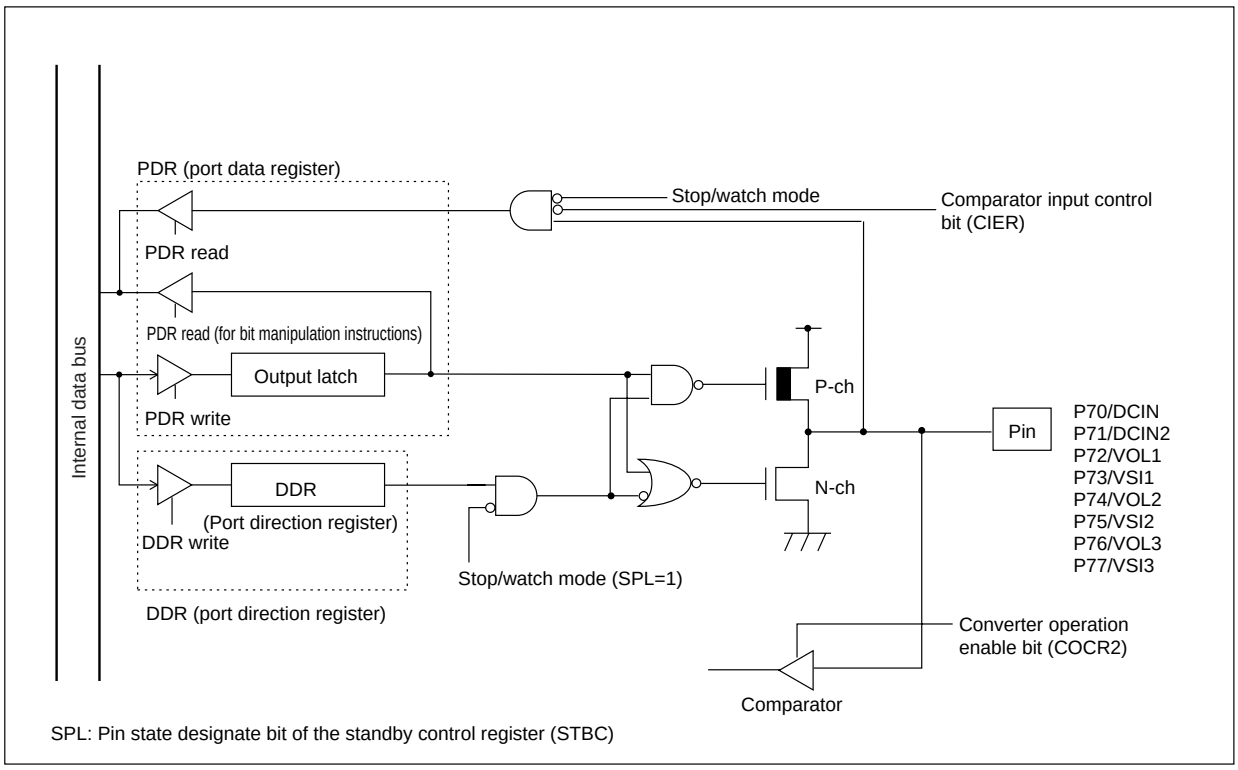


**Figure 13.3-3 Block Diagram of Pins Associated with the Comparator**




SPL: Pin state designate bit of the standby control register (STBC)

**Note:**

The comparator becomes operation enable state, even if the stop mode (SPL=1), when the comparator operation was enabled by the comparator control register 2 (COCR2).

# 13.4 Registers of the Comparator

**This section shows the registers associated with the comparator.**

## ■ Registers Associated with Comparator

**Figure 13.4-1 Block Diagram of Pins Associated with the Comparator**

COCR1(comparator control register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0051H | - | - | BOF3 | BOF2 | BOF1 | SPM2 | SPM1 | SPM0 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

COCR2(comparator control register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0052H | - | - | - | B3 | B2 | B1 | DC2 | DC1 | XXX11111B |
| | | | | R/W | R/W | R/W | R/W | R/W | |

COSR1(comparator status register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0053H | COR8 | COR7 | COR6 | COR5 | COR4 | COR3 | COR2 | COR1 | 00000000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

CICR1(comparator interrupt control register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0054H | CEN8 | CEN7 | CEN6 | CEN5 | CEN4 | CEN3 | CEN2 | CEN1 | 00000000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

COSR2(comparator status register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0055H | - | - | SWR3 | SWR2 | SWR1 | VAR3 | VAR2 | VAR1 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

CICR2(comparator interrupt control register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0056H | - | - | SEN3 | SEN2 | SEN1 | VEN3 | VEN2 | VEN1 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

COSR3(comparator status register 3)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0057H | - | - | SWS3 | SWS2 | SWS1 | VAL3 | VAL2 | VAL1 | XXXXXXXXB |
| | | | R | R | R | R | R | R | |

COSR4(comparator status register 4)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0058H | COS8 | COS7 | COS6 | COS5 | COS4 | COS3 | COS2 | COS1 | XXXXXXXXB |
| | R | R | R | R | R | R | R | R | |

CIER(comparator input enable register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0059H | - | - | - | BIE3 | BIE2 | BIE1 | DIE2 | DIE1 | XXX11111B |
| | | | | R/W | R/W | R/W | R/W | R/W | |

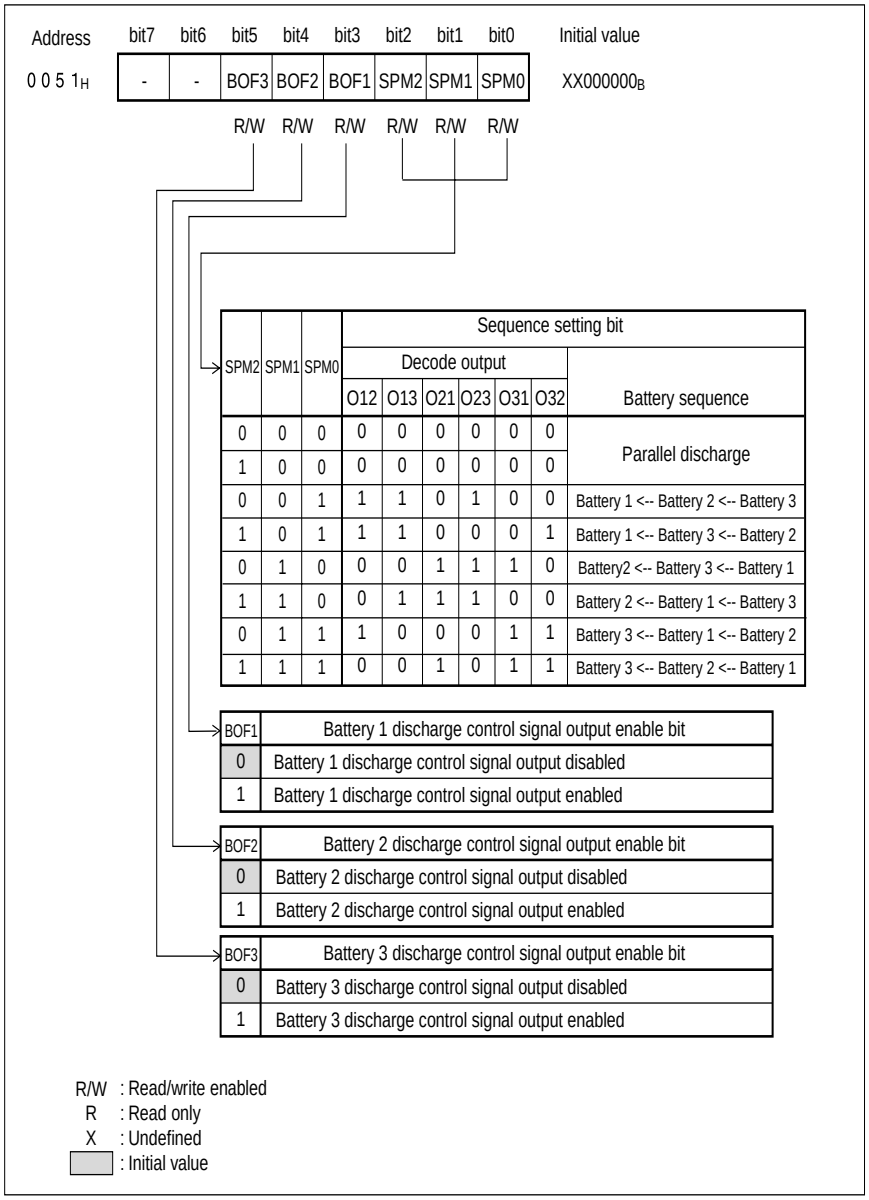
R/W : Read/write enabled
R : Read only
X : Undefined

# 13.4.1 Comparator Control Register 1 (COCR1)

**The comparator control register (COCR1) is used to confirm the sequence setting value for the battery selection circuit and to set discharge control signal output for batteries.**

## ■ Comparator Control Register 1 (COCR1)

**Figure 13.4-2 Comparator Control Register 1 (COCR1)**



| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0051H | - | - | BOF3 | BOF2 | BOF1 | SPM2 | SPM1 | SPM0 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

| SPM2 | SPM1 | SPM0 | Sequence setting bit: Decode output O12 | O13 | O21 | O23 | O31 | O32 | Battery sequence |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Parallel discharge |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | Battery 1 <-- Battery 2 <-- Battery 3 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | Battery 1 <-- Battery 3 <-- Battery 2 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | Battery2 <-- Battery 3 <-- Battery 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | Battery 2 <-- Battery 1 <-- Battery 3 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | Battery 3 <-- Battery 1 <-- Battery 2 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | Battery 3 <-- Battery 2 <-- Battery 1 |

| BOF1 | Battery 1 discharge control signal output enable bit |
|---|---|
| 0 | Battery 1 discharge control signal output disabled |
| 1 | Battery 1 discharge control signal output enabled |

| BOF2 | Battery 2 discharge control signal output enable bit |
|---|---|
| 0 | Battery 2 discharge control signal output disabled |
| 1 | Battery 2 discharge control signal output enabled |

| BOF3 | Battery 3 discharge control signal output enable bit |
|---|---|
| 0 | Battery 3 discharge control signal output disabled |
| 1 | Battery 3 discharge control signal output enabled |

R/W : Read/write enabled
R : Read only
X : Undefined
: Initial value

**Table 13.4-1 Function of Each Bit of the Comparator Control Register 1 (COCR1)**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 5 | BOF3:<br>Battery 3 discharge control signal output enable bit | This bit is used to allow discharge control signal output to battery 3.<br>When "1" is written in this bit, output of discharge control signal is enabled. When "0" is written, output of discharge control signal is disabled. |
| Bit 4 | BOF2:<br>Battery 2 discharge control signal output enable bit | This bit is used to allow discharge control signal output to battery 2.<br>When "1" is written in this bit, output of discharge control signal is enabled. When "0" is written, output of discharge control signal is disabled. |
| Bit 3 | BOF1:<br>Battery 1 discharge control signal output enable bit | This bit is used to allow discharge control signal output to battery 1.<br>When "1" is written in this bit, output of discharge control signal is enabled. When "0" is written, output of discharge control signal is disabled. |
| Bit 2<br>Bit 1<br>Bit 0 | SPM2, SPM1, SOMO:<br>Sequence setting bit | These bits are used to set sequence data for the battery selection circuit. Depending on the set value, either parallel discharge or sequential discharge is selected.<br>By reading this bit, the sequence setting value can be confirmed. |

**Reference:**

When the OFB1 to 3 pin output is "H", battery discharge is disabled. When it is "L", battery discharge is allowed.

**Note:**

Do not rewrite the sequence control bit (COCR1:SPM0 to 2) when the comparator is operating.

# 13.4.2 Comparator Control Register 2 (COCR2)

**Comparator control register 2 is used to control I/O of the comparator.**

## ■ Comparator Control Register 2 (COCR2)

**Figure 13.4-3 Comparator Control Register 2 (COCR2)**

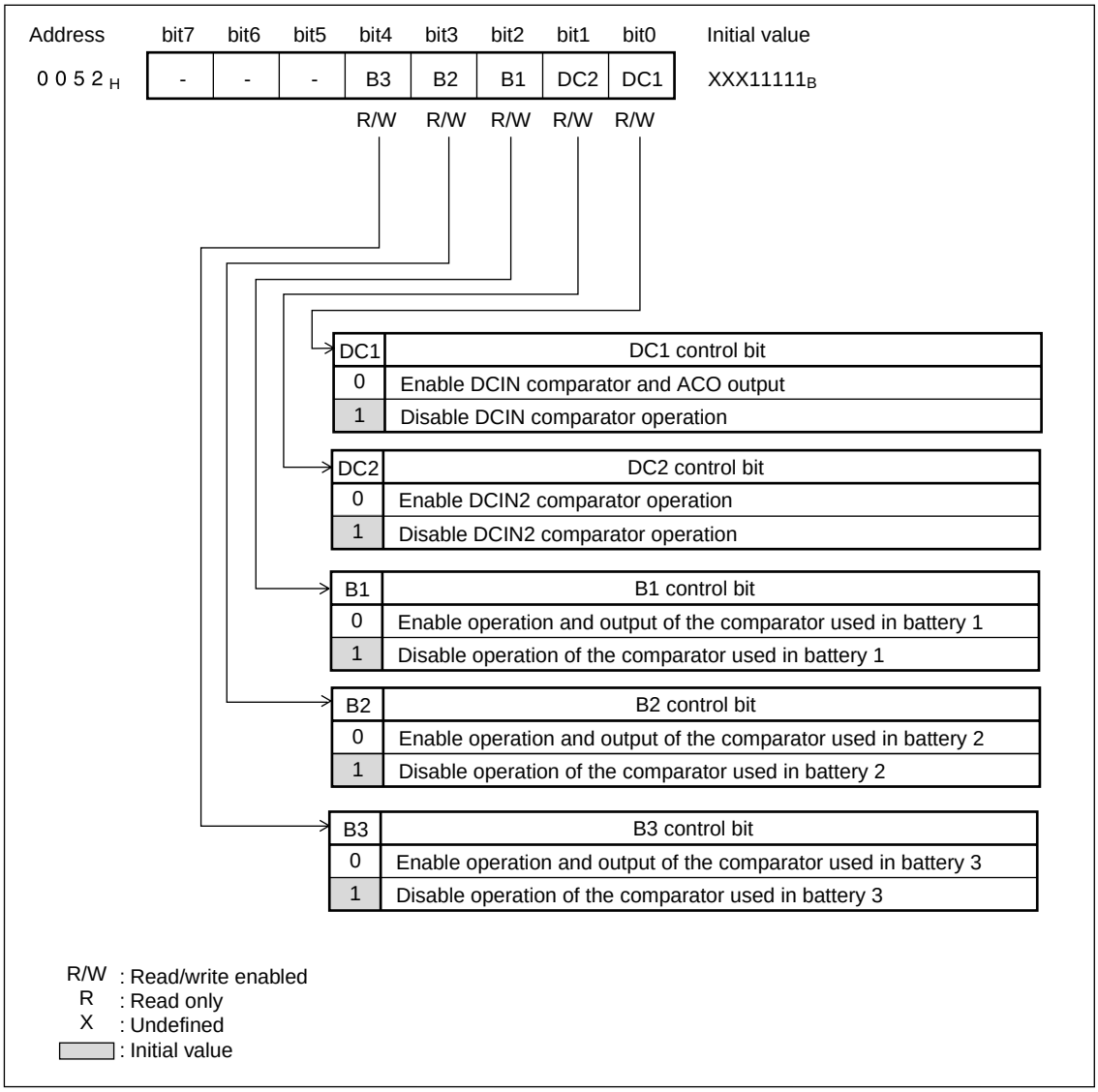


**Table 13.4-2 Function of Each Bit of the Comparator Control Register 2 (COCR2)**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 4 | B3:<br>B3 control bit | This bit is used to allow comparator operation of the pin used for battery 3.<br>• When this bit is set to "0", the comparators for the VOL3, VSI3, and SW3 pins operate and OFB3 and ALR3 pin output is allowed.<br>• When this bits is set to "1", the comparators for the VOL3, VSI3, and SW3 pins stop. OFB3 and ALR3 pin output becomes port output. |
| Bit 3 | B2:<br>B2 control bit | This bit is used to allow comparator operation of the pin used for battery 2.<br>• When this bit is set to "0", the comparators for the VOL2, VSI2, and SW2 pins operate and OFB2 and ALR2 pin output is allowed.<br>• When this bits is set to "1", the comparators for the VOL2, VSI2, and SW2 pins stop. OFB2 and ALR2 pin output becomes port output. |
| Bit 2 | B1:<br>B1 control bit | This bit is used to allow comparator operation of the pin used for battery 1.<br>• When this bit is set to "0", the comparators for the VOL1, VSI1, and SW1 pins operate and OFB1 and ALR1 pin output is allowed.<br>• When this bit is set to "1", the comparators for the VOL1, VSI1, and SW1 pins stop. OFB1 and ALR1 pin output becomes port output. |
| Bit 1 | DC2:<br>DC2 control register | This bit is used to allow comparator operation of the DC2 pin.<br>• When this bit is set to "0", the comparator for the DCIN2 pin operates<br>• When this bit is set to "1", the comparator for the DCIN2 pin stops. |
| Bit 0 | DC1:<br>DC1 control bit | This bit is used to allow comparator operation of the DC1 pin.<br>• When this bit is set to "0", the comparator for the DCIN pin operates and ACO pin output is allowed.<br>• When this bit is set to "1", the comparator for the DCIN pin stops and ACO pin output becomes port output. |

# 13.4.3 Comparator Status Register 1 (COSR1)

**Comparator status register 1 (COSR1) holds and stores an edge change in output voltage from the comparator 1 and an OUT signal output from voltage comparators. This allows change of battery voltage to be confirmed.**

## ■ Comparator Status Register 1 (COSR1)

**Figure 13.4-4 Comparator Status Register 1 (COSR1)**

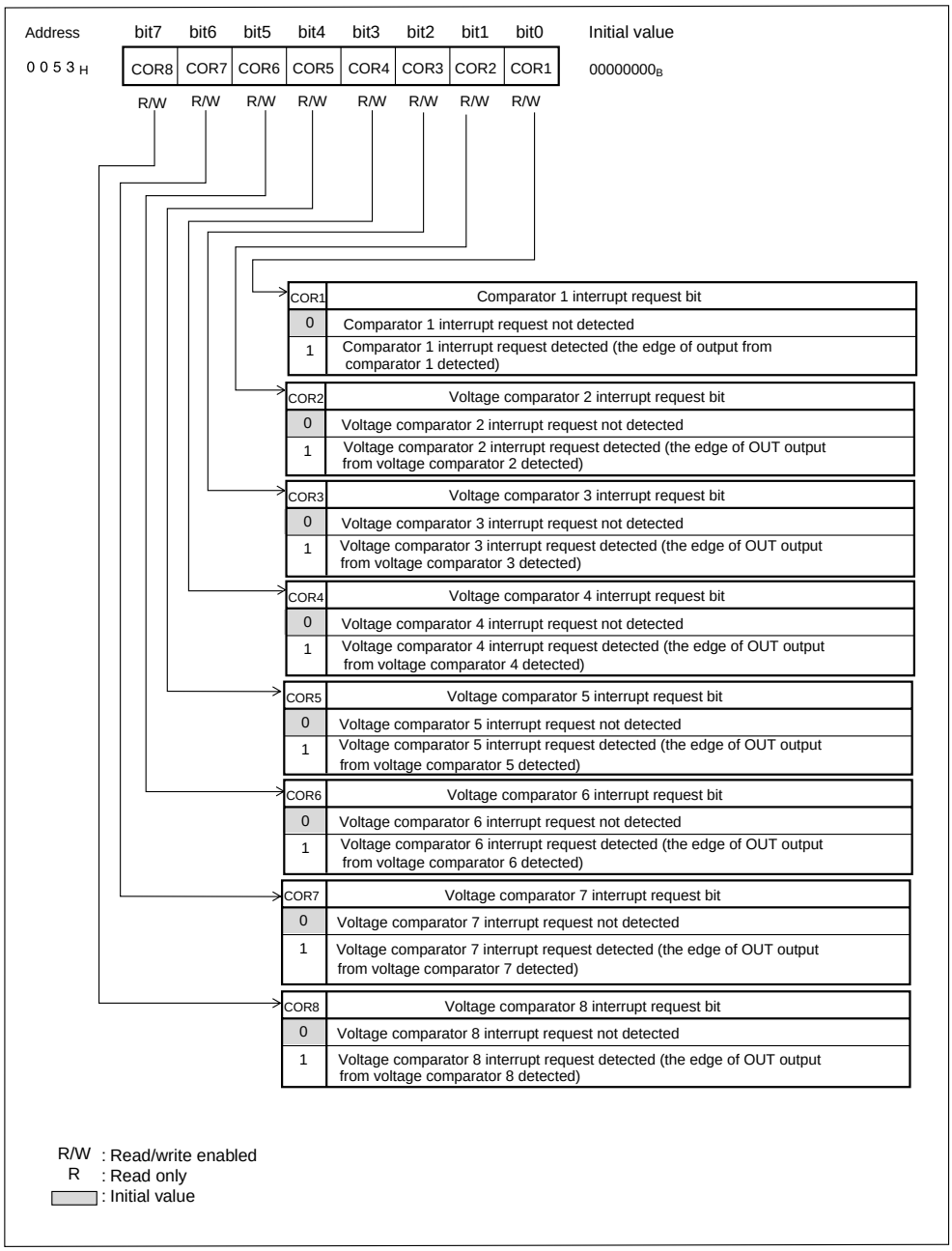


**Table 13.4-3 Comparator Status Register 1 (COSR1) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7 | COR8:<br>Voltage comparator 8 interrupt request bit | This bit holds and stores an edge change in OUT output from voltage comparator 8 (the result of P77/VSI3 pin input comparison by voltage comparator 8).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN8) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect on this bit and does not change the bit. |
| Bit 6 | COR7:<br>Voltage comparator 7 interrupt request bit | This bit holds and stores an edge change in OUT output from voltage comparator 7 (the result of P76/VOL3 pin input comparison by voltage comparator 7).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN7) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 5 | COR6:<br>Voltage comparator 6 interrupt request bit | 8. This bit holds and stores an edge change in OUT output from voltage comparator 6 (the result of P75/VSI2 pin input comparison by voltage comparator 6).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN6) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 4 | COR5:<br>Voltage comparator 5 interrupt request bit | This bit holds and stores an edge change in OUT output from voltage comparator 5 (the result of P74/VOL2 pin input comparison by voltage comparator 5).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN5) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 3 | COR4:<br>Voltage comparator 4 interrupt request bit | This bit holds and stores an edge change in the OUT output from voltage comparator 4 (the result of P73/VSI1 pin input comparison by voltage comparator 4).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN4) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 2 | COR3:<br>Voltage comparator 3 interrupt request bit | 14. This bit holds and stores an edge change in the OUT output from voltage comparator 3 (the result of P72/VOL1 pin input comparison by voltage comparator 3).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN3) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |

**Table 13.4-3 Comparator Status Register 1 (COSR1) Bit Functions (Continued)**

| Bit name | | Function |
|---|---|---|
| Bit 1 | COR2:<br>Voltage comparator 2 interrupt request bit | This bit holds and stores an edge change in OUT output from voltage comparator 2 (the result of P71/DCIN2 pin input comparison by voltage comparator 2).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN2) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 0 | COR1:<br>Voltage comparator 1 interrupt request bit | This bit holds and stores an edge change in OUT output from voltage comparator 1 (the result of P70/DCIN pin input comparison by comparator 1).<br>(This bit is set to "1" when the rising/falling edge of OUT output is detected. At this time, if the interrupt request enable bit (CICR1:CEN1) is "1," an interrupt request is output. This bit is cleared when "0" is written. Writing "1" in this bit has no effect and does not change the bit. |

**Note:**

To clear each bit, first read data from the bit, then write "0" in it.

# 13.4.4 Comparator Interrupt Control Register 1 (CICR1)

**Comparator interrupt control register 1 (CICR1) is used to allow an interrupt to be generated when the comparator status register 1 (COSR1) holds an edge change.**

## ■ Comparator Interrupt Control Register 1 (CICR1)

**Figure 13.4-5 Comparator Interrupt Control Register 1 (CICR1)**

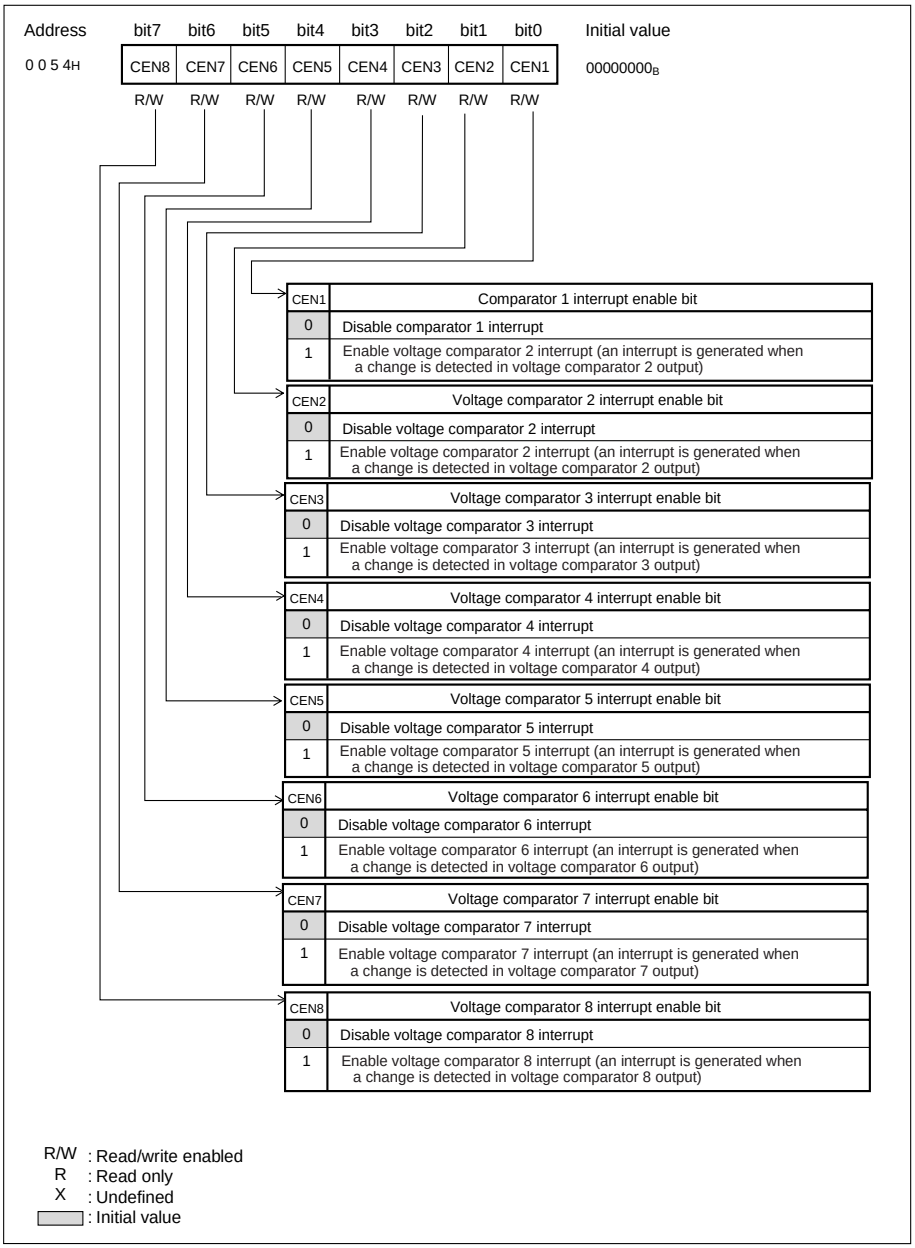


**Table 13.4-4 Comparator Interrupt Control Register 1 (CICR1) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7 | CEN8:<br>Voltage comparator 8 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in comparator 8 OUT output. When the voltage comparator 8 interrupt request bit (COSR1:COR8) is "1" an interrupt request is posted to the CPU. |
| Bit 6 | CEN7:<br>Voltage comparator 7 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 7 OUT output. When the voltage comparator 7 interrupt request bit (COSR1:COR7) is "1" an interrupt request is posted to the CPU. |
| Bit 5 | CEN6:<br>Voltage comparator 6 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 6 OUT output. When the voltage comparator 6 interrupt request bit (COSR1:COR6) is "1" an interrupt request is posted to the CPU. |
| Bit 4 | CEN5:<br>Voltage comparator 5 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 5 OUT output. When the voltage comparator 5 interrupt request bit (COSR1:COR5) is "1" an interrupt request is posted to the CPU. |
| Bit 3 | CEN4:<br>Voltage comparator 4 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 4 OUT output. When the voltage comparator 4 interrupt request bit (CORS1:COR4) is "1" an interrupt request is posted to the CPU. |
| Bit 2 | CEN3:<br>Voltage comparator 3 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 3 OUT output. When the voltage comparator 3 interrupt request bit (COSR1:COR3) is "1" an interrupt request is posted to the CPU. |
| Bit 1 | CEN2:<br>Voltage comparator 2 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 2 OUT output. When the voltage comparator 2 interrupt request bit (COSR1:COR2) is "1" an interrupt request is posted to the CPU. |
| Bit 0 | CEN1:<br>Voltage comparator 1 interrupt enable bit | This bit is used to allow an interrupt when a change is detected in voltage comparator 1 OUT output. When the voltage comparator 1 interrupt request bit (COSR1:COR1) is "1" an interrupt request is posted to the CPU. |

# 13.4.5 Comparator Status Register 2 (COSR2)

**Comparator status register 2 holds and stores an edge change in comparator output from SW1 to SW3 and in a VALID signal from the battery monitoring circuit. This allows the valid status of each battery to be detected.**

## ■ Comparator Status Register 2 (COSR2)

**Figure 13.4-6 Comparator Status Register 2 (COSR2)**

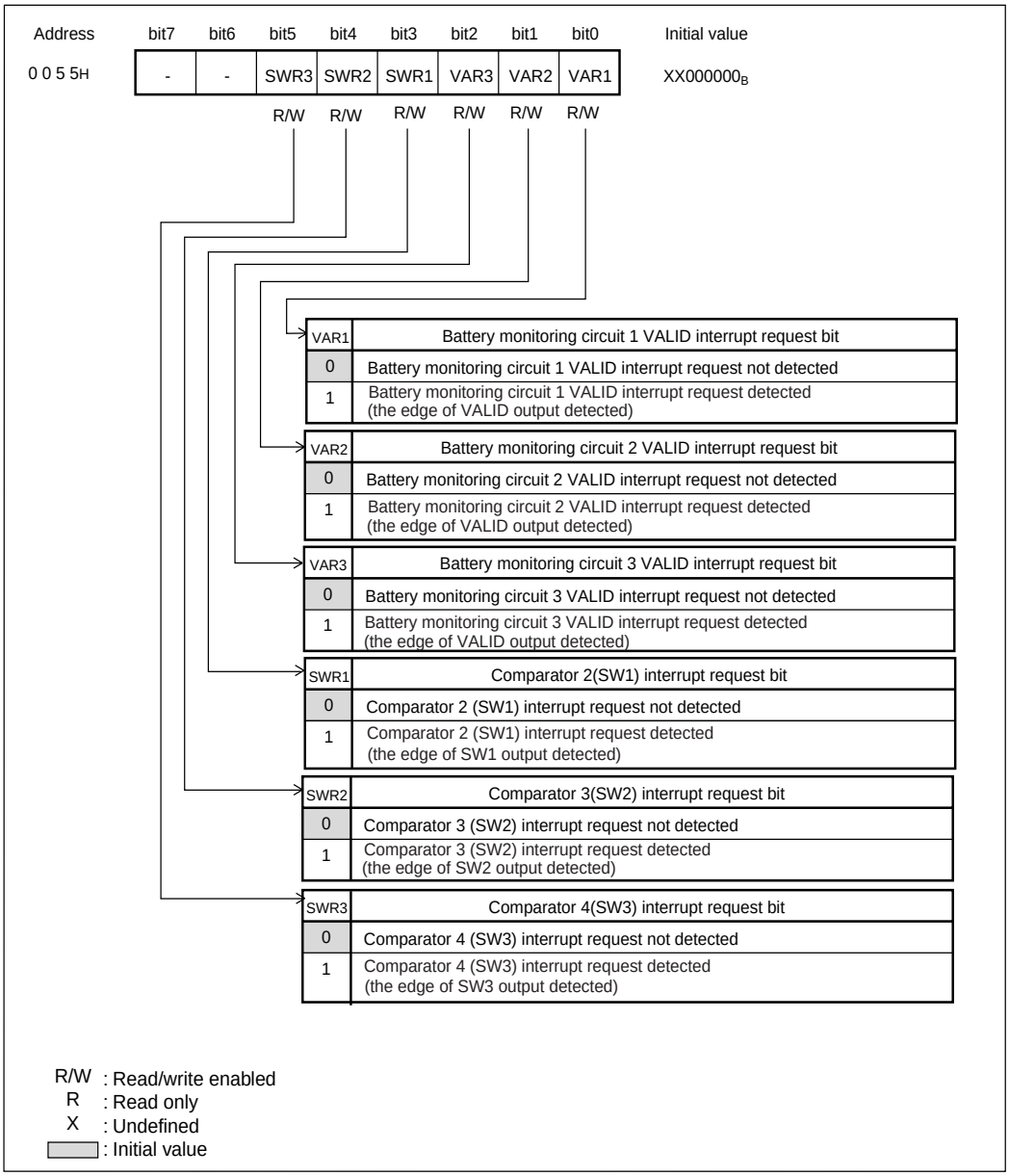


**Table 13.4-5 Comparator Status Register 2 (COSR2) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 5 | SWR3:<br>Comparator 4 interrupt request bit | This bit is set to "1" when an edge change is detected in comparator 4 output (the results of P87/SW3 pin input comparison by comparator 4). When this bit and the SW3 interrupt enable bit (CICR2:SEN3) are "1" an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 4 | SWR2:<br>Comparator 3 interrupt request bit | This bit is set to "1" when an edge change is detected in comparator 3 output (the results of P86/SW2 pin input comparison by comparator 4). When this bit and the SW2 interrupt enable bit (CICR2:SEN2) are "1" an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 3 | SWR1:<br>Comparator 2 interrupt request bit | This bit is set to "1" when an edge change is detected in comparator 2 output (the results of P85/SW1 pin input comparison by comparator 4). When this bit and the SW1 interrupt enable bit (CICR2:SEN1) are "1," an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 2 | VAR3:<br>Battery monitoring circuit 3 VALID interrupt request bit | This bit is set to "1" when a change is detected in battery monitoring circuit 3 VALID output. When this bit and the battery monitoring circuit 3 VALID interrupt enable bit (CICR2:VEN3) are "1," an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change this bit. |
| Bit 1 | VAR2:<br>Battery monitoring circuit 2 VALID interrupt request bit | This bit is set to "1" when a change is detected in battery monitoring circuit 2 VALID output. When this bit and the battery monitoring circuit 2 VALID interrupt enable bit (CICR2:VEN2) are "1," an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change the bit. |
| Bit 0 | VAR1:<br>Battery monitoring circuit 1 VALID interrupt request bit | This bit is set to "1" when a change is detected in battery monitoring circuit 1 VALID output. When this bit and the battery monitoring circuit 1 VALID interrupt enable bit (CICR2:VEN1) are "1," an interrupt request is output.<br>This bit is cleared by writing "0" in it. Writing "1" in this bit has no effect and does not change the bit. |

**Note:**

To clear each bit, first read data from the bit then write "0" in it.

# 13.4.6 Comparator Interrupt Control Register (CICR2)

**Comparator interrupt control register 2 (CICR2) is used to allow interrupt generation when an edge change is seized by the comparator status register 2 (COSR2).**

## ■ Comparator Interrupt Control Register 2 (CICR2)

**Figure 13.4-7 Comparator Interrupt Control Register 2 (CICR2)**

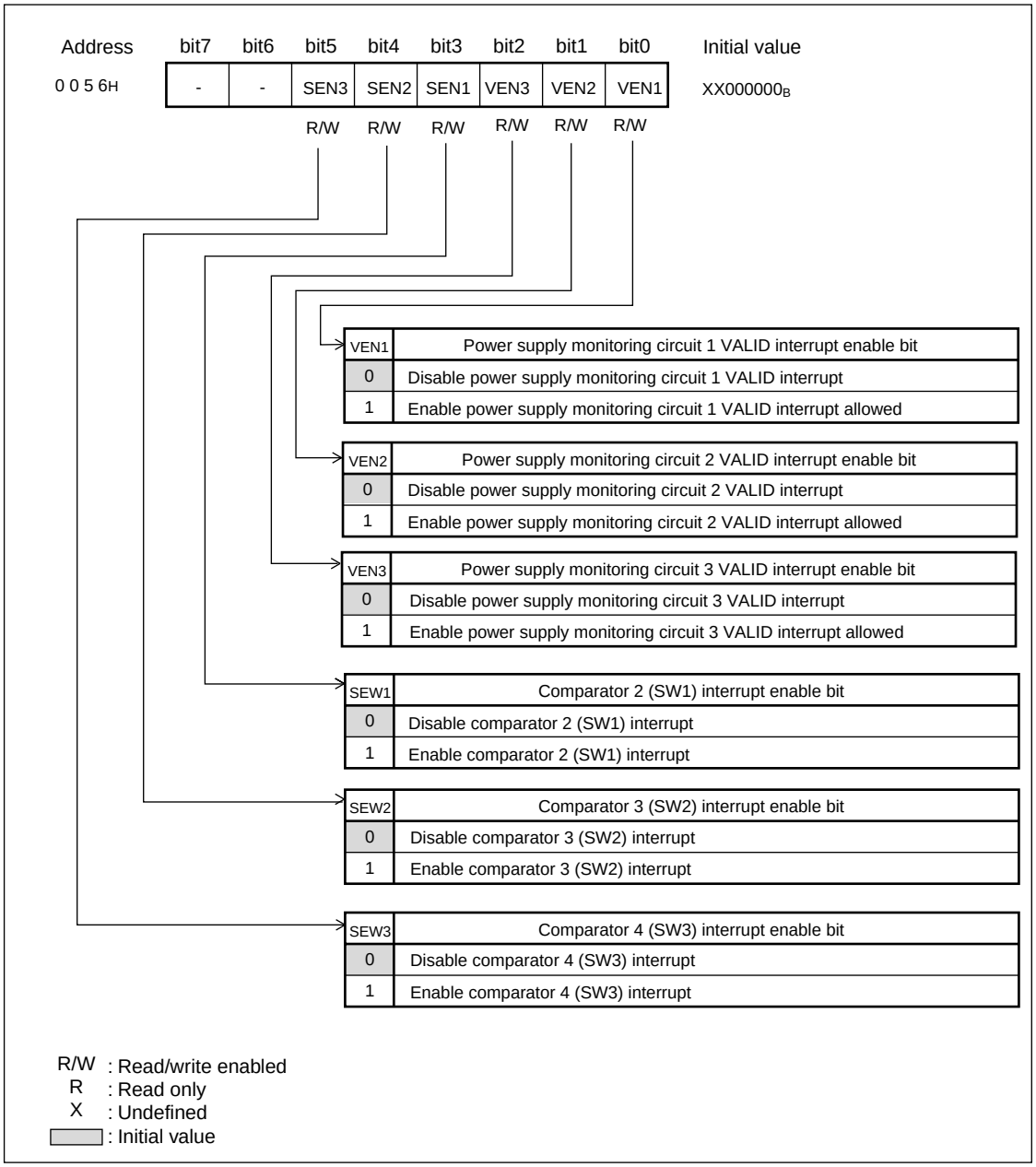


**Table 13.4-6 Comparator Interrupt Control Register 2 (CICR2) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 5 | SW3:<br>Comparator 4 interrupt enable bit | This bit is used to allow a comparator 4 interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the comparator 4 interrupt request bit is set to "1" (COSR2:SW3=1). |
| Bit 4 | SW2:<br>Comparator 3 interrupt enable bit | This bit is used to allow a comparator 3 interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the comparator 3 interrupt request bit is set to 1" (COSR2:SW2=1). |
| Bit 3 | SW1:<br>Comparator 2 interrupt enable bit | This bit is used to allow a comparator 2 interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the comparator 2 interrupt request bit is set to 1" (COSR2:SW1=1). |
| Bit 2 | VEN3:<br>Battery monitoring circuit 3 VALID interrupt enable bit | This bit is used to allow a battery monitoring circuit 3 VALID interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the battery monitoring circuit 3 interrupt request bit is set to "1" (COSR2:VAR3=1). |
| Bit 1 | VEN2:<br>Battery monitoring circuit 2 VALID interrupt enable bit | This bit is used to allow a battery monitoring circuit 2 VALID interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the battery monitoring circuit 2 interrupt request bit is set to "1" (COSR2:VAR2=1). |
| Bit 0 | VEN1:<br>Battery monitoring circuit 1 VALID interrupt enable bit | This bit is used to allow a battery monitoring circuit 1 VALID interrupt.<br>If this bit is set to "1," an interrupt to the CPU will be generated when the battery monitoring circuit 1 interrupt request bit is set to "1" (COSR2:VAR1=1). |

# 13.4.7 Comparator Status Register 3 (COSR3)

**Comparator status register 3 stores comparator output from SW1 to SW3 and a VALID signal from the battery monitoring circuit. The status of each battery can be confirmed by reading this register.**

## ■ Comparator Status Register 3 (COSR3)

**Figure 13.4-8 Comparator Status Register 3 (COSR3)**

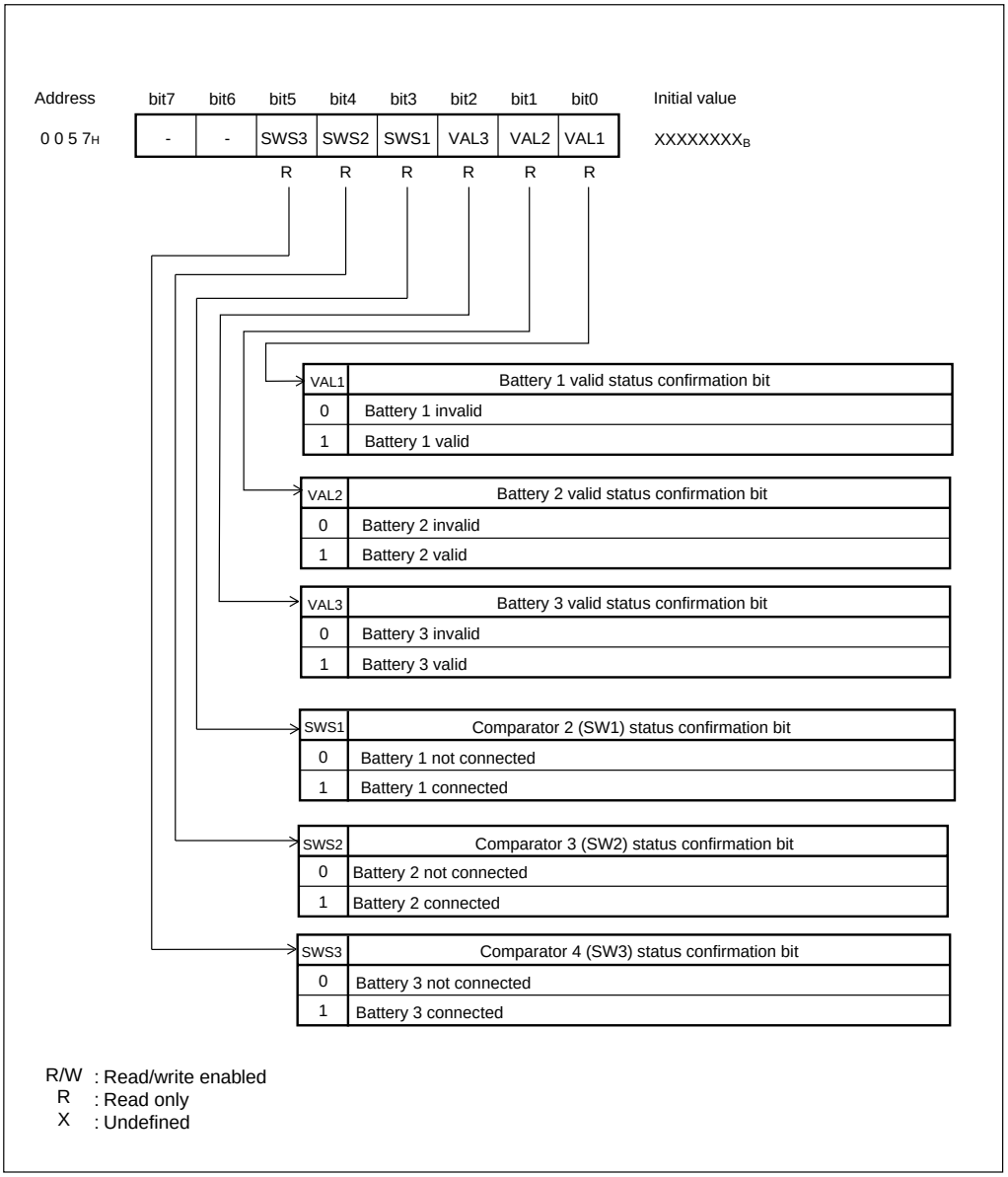


**Table 13.4-7 Comparator Status Register 3 (COSR3) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 5 | SWS3:<br>Comparator 4 status confirmation bit | Read this bit to confirm the output value from comparator 4.<br>When this bit is "1", it indicates the battery is connected.<br>When this bit is "0", it indicates the battery is not connected.<br>Writing in this bit has no effect. |
| Bit 4 | SWS2:<br>Comparator 3 status confirmation bit | Read this bit to confirm the output value from comparator 3.<br>When this bit is "1", it indicates the battery is connected.<br>When this bit is "0", it indicates the battery is not connected.<br>Writing in this bit has no effect. |
| Bit 3 | SWS1:<br>Comparator 2 status confirmation bit | Read this bit to confirm the output value from comparator 2.<br>When this bit is "1", it indicates the battery is connected.<br>When this bit is "0", it indicates the battery is not connected.<br>Writing in this bit has no effect. |
| Bit 2 | VAL3:<br>Battery 3 valid status confirmation bit 3 | Read this bit to confirm the VALID signal output value from battery monitoring circuit 3.<br>When this bit is "1", it indicates battery 3 is valid. When this bit is "0", it indicates battery 3 is invalid.<br>Writing in this bit has no effect. |
| Bit 1 | VAL2:<br>Battery 2 valid status confirmation bit 2 | Read this bit to confirm the VALID signal output value from battery monitoring circuit 2. When this bit is "1", it indicates battery 2 is valid. When this bit is "0", it indicates battery 2 is invalid.<br>Writing in this bit has no effect. |
| Bit 0 | VAL1:<br>Battery 1 valid status confirmation bit 1 | Read this bit to confirm the VALID signal output value from battery monitoring circuit 1.<br>When this bit is "1", it indicates battery 1 is valid. When this bit is "0", it indicates battery 1 is invalid.<br>Writing in this bit has no effect. |

# 13.4.8 Comparator Status Register 4 (COSR4)

**Comparator status register 4 (COISR4) stores output from comparator 1 and the OUT signals that are output from the battery monitoring circuits. The status of each battery can be confirmed by reading this register.**

## ■ Comparator Status Register 4 (COSR4)

**Figure 13.4-9 Comparator Status Register 4 (COSR4)**

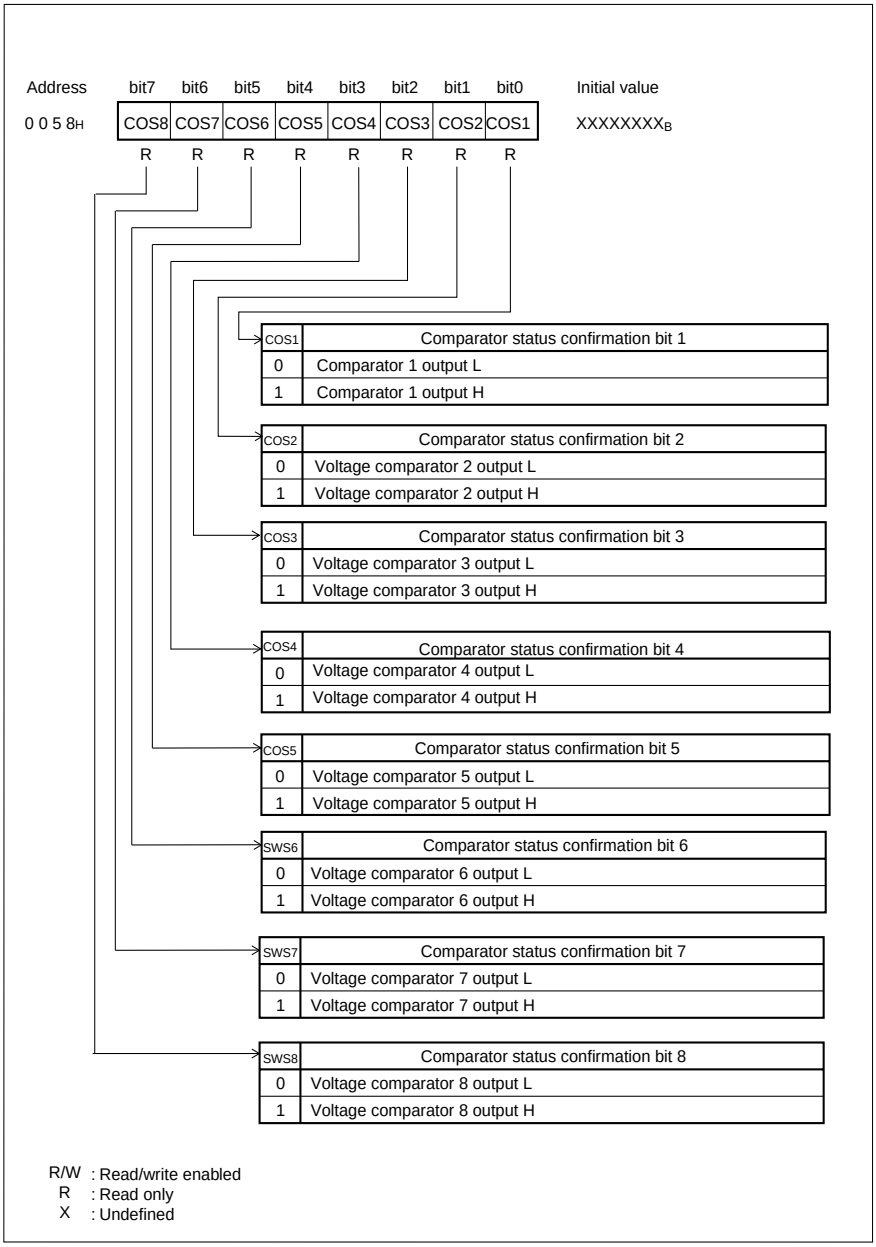


**Table 13.4-8 Comparator Status Register 4 (COSR4) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7 | COS8:<br>Comparator status confirmation bit 8 | Reading this bit to confirm OUT output from voltage comparator 8. When this bit is "1", it indicates the power level of battery 3 is enough. When this bit is "0", it indicates the power level of battery 3 is in short. |
| Bit 6 | COS7:<br>Comparator status confirmation bit 7 | Reading this bit to confirm OUT output from voltage comparator 7. When this bit is "1", it indicates power is being supplied from battery 3. When this bit is "0", it indicates power supply from battery 3 has dropped. |
| Bit 5 | COS6:<br>Comparator status confirmation bit 6 | Reading this bit to confirm OUT output from voltage comparator 6. When this bit is "1", it indicates the power level of battery 2 is enough. When this bit is "0", it indicates the power level of battery 2 is in short. |
| Bit 4 | COS5:<br>Comparator status confirmation bit 5 | Read this bit to confirm OUT output from voltage comparator 5. When this bit is "1", it indicates power is being supplied from battery 2. When this bit is "0", it indicates power supply from battery 2 has dropped. |
| Bit 3 | COS4:<br>Comparator status confirmation bit 4 | Read this bit to confirm OUT output from voltage comparator 4. When this bit is "1", it indicates the power level of battery 1 is enough. When this bit is "0", it indicates the power level of battery 1 is in short. |
| Bit 2 | COS3:<br>Comparator status confirmation bit 3 | Reading this bit to confirm OUT output from voltage comparator 3. When this bit is "1", it indicates power is being supplied from battery 1. When this bit is "0", it indicates power supply from battery 1 has dropped. |
| Bit 1 | COS2:<br>Comparator status confirmation bit 2 | Read this bit to confirm OUT output from voltage comparator 2. When this bit is "1", it indicates power is being supplied from the AC adapter connected to DCIN2. When this bit is "0", it indicates power supply from the AC adapter connected to DCIN2 has dropped. |
| Bit 0 | COS1:<br>Comparator status confirmation bit 1 | Read this bit to confirm OUT output from voltage comparator 1. When this bit is "1", it indicates power is being supplied from the AC adapter connected to DCIN. When this bit is "0", it indicates power supply from the AC adapter connected to DCIN has dropped. |

# 13.4.9 Comparator Input Allow Register (CIER)

**Comparator input enable register is used to cut the DC path when the comparator is used.**

## ■ Comparator Input Enable Register (CIER)

**Figure 13.4-10 Comparator Input Enable Register**

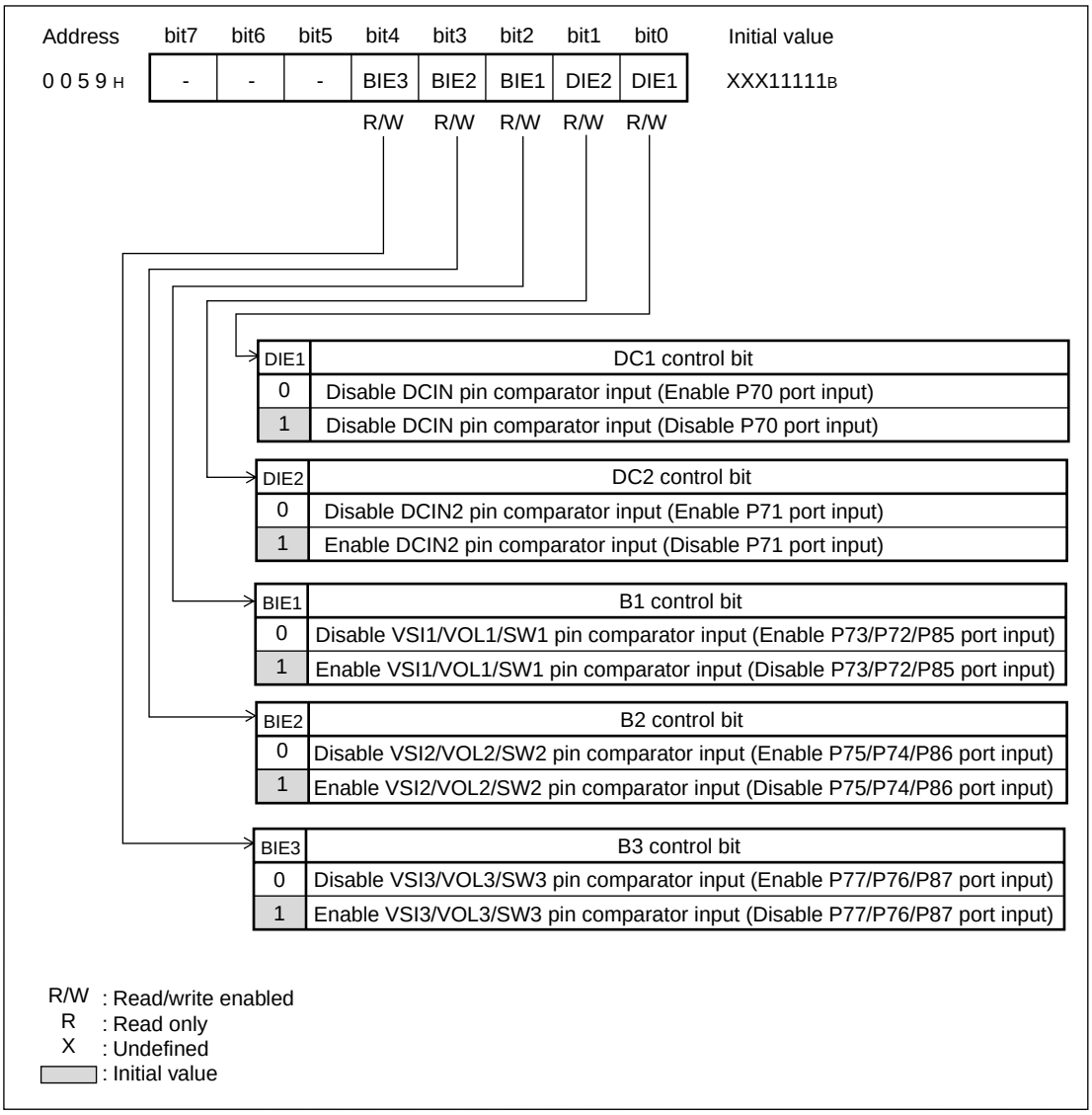


**Table 13.4-9 Comparator Input Enable Register (CIER) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5 | Unused bits | • The read value is undefined.<br>• Writing has no efect on operation. |
| Bit 4 | BIE3:<br>B3 control bit | This bit is used to allow input from the VOL3/VSI3/SW3 pin to the comparator.<br>• By setting "1" in this bit, the DC path can be cut when the intermediate level is input to the general-purpose port.<br>• To use the corresponding port as a general-purpose port, set "0" in this bit. |
| Bit 3 | BIE2:<br>B2 control bit | This bit is used to allow input from the VOL2/VSI2/SW2 pin to the comparator.<br>• By setting "1" in this bit, the DC path can be cut when the intermediate level is input to the general-purpose port.<br>• To use the corresponding port as a general-purpose port, set "0" in this bit. |
| Bit 2 | BIE1:<br>B1 control bit | This bit is used to allow input from the VOL1/VSI1/SW1 pin to the comparator.<br>• By setting "1" in this bit, the DC path can be cut when the intermediate level is input to the general-purpose port.<br>• To use the corresponding port as a general-purpose port, set "0" in this bit. |
| Bit 1 | DIE2:<br>DC2 control bit | This bit is used to allow input from the DCIN2 pin to the comparator.<br>• By setting "1" in this bit, the DC path can be cut when the intermediate level is input to the general-purpose port.<br>• To use the corresponding port as a general-purpose port, set "0" in this bit. |
| Bit 0 | DIE1:<br>DC1 control bit | This bit allows input from the DCIN pin to the comparator.<br>• By setting "1" in this bit, the DC path can be cut when the intermediate level is input to the general-purpose port.<br>• et "0" in the bit to use the corresponding port as a general-purpose port. |

# 13.5 Comparator Interrupts

**There are 14 different causes for comparator interrupts:**

- **Comparator 1, voltage comparators 2 to 8 interrupts**
- **Batteries 1 to 3 VALID interrupts. Comparators 2 to 4 interrupts**

### ■ Comparator 1, Voltage Comparators 2 to 7 Interrupts

Voltage input to the P70/DCIN, P71/DCIN2, P72/VOL1, P73/VSI1, P74/VOL2, P75/VS12, P76/VOL3, and P77/VSI3 pins is compared by comparator 1 and voltage comparators 2 to 8 (which have reference voltage input from the CVRH1 and CVRL pins, as hysteresis width). If a change is detected in the output of these comparators (edge detection), the interrupt request bits for comparator 1 and voltage comparators 2 to 8 (COSR1:COR1 to COR8) are set to "1". At this time, if the interrupt enable bit for the comparator 1 and voltage comparators 2 to 8 has been set to "enabled" (CICR1:"CEN to CEN8"=1), an interrupt request (IRQ4) to the CPU is generated.

In the interrupt processing routine, write "0" in the interrupt enable bits for comparator 1 and voltage comparators 2 to 8 to clear the interrupt requests.

### ■ Batteries 1 to 3 VALID Interrupts

The battery VALID interrupt request bits (COSR2:VAR1 to VAR3) are set to "1" under the following conditions:

- In comparison of voltage input to the P72/VOL1, P73/VSI1, and P84/AN0/SW1 pins by the battery monitoring circuit 1, a change has been detected in the VALID output result (edge detection).
- In comparison of voltage input to the P74/VOL2, P75/VSI2, and P85/AN1/SW2 pins by the battery monitoring circuit 2, a change has been detected in the VALID output result (edge detection).
- In comparison of voltage input to the P76/VOL3, P77/VSI3, and P86/AN2/SW3 pins by the battery monitoring circuit 3, a change has been detected in the VALID output result (edge detection).

  When any of the above conditions is true, an interrupt request (IRQ5) to the CPU is generated if the battery VALID interrupt enable bit has been set to enabled.(CICR2:VEN1 to VEN3)

  Write "0" in the battery interrupt request bits (COSR2:VAR1 to VAR3) in the interrupt processing routine to clear the interrupt requests.

### ■ Comparators 2 to 4 Interrupts

Voltage input to the P85/AN0/SW1, P86/AN1/SW2, and P87/AN2/SW3 pins is compared by comparators 2 to 4. If a change is detected in the output of these comparators (edge detection), an interrupt request (IRQ5) to the CPU is generated if the interrupt enable bits for comparators 2 to 4 have been set to "enabled" (CICR2: SEN1 to SEN3=1).

In the interrupt processing routine, write "0" in the interrupt enable bits for comparator 1 and voltage comparators 2 to 8 to clear the interrupt requests.

## ■ Registers and Vector Tables Associated with Comparator Interrupts

**Table 13.5-1 Registers and Vector Tables Associated with Comparator Interrupts**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ4 | ILR2 ($007C_H$) | L41 (bit 1) | L40 (bit 0) | $FFF2_H$ | $FFF3_H$ |
| IRQ5 | ILR2 ($007C_H$) | L51 (bit 3) | L50 (bit 2) | $FFF0_H$ | $FFF1_H$ |

For interrupt operation, see Section 3.4.2 "Interrupt Processing".

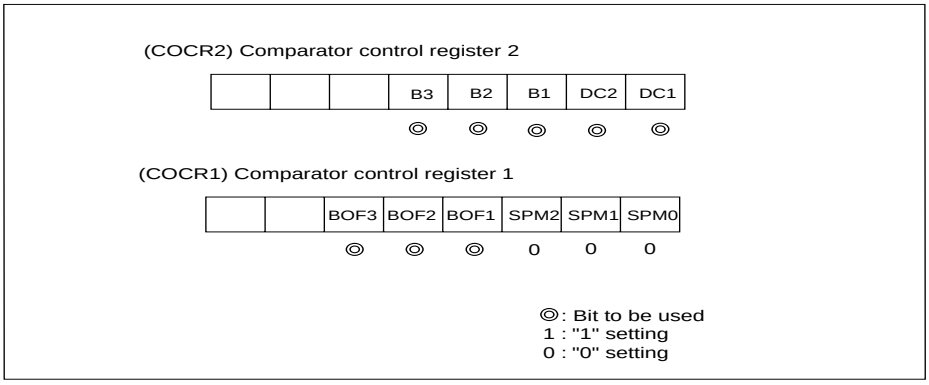
# 13.6 Operation of the Parallel Discharge Control

**This section describes operation in parallel discharge control.**

## ■ Operation of the Parallel Discharge Control

To operate in parallel discharge control, setting shown in Figure 13.6-1 "Setting for Parallel Discharge Control" is required.

**Figure 13.6-1 Setting for Parallel Discharge Control**

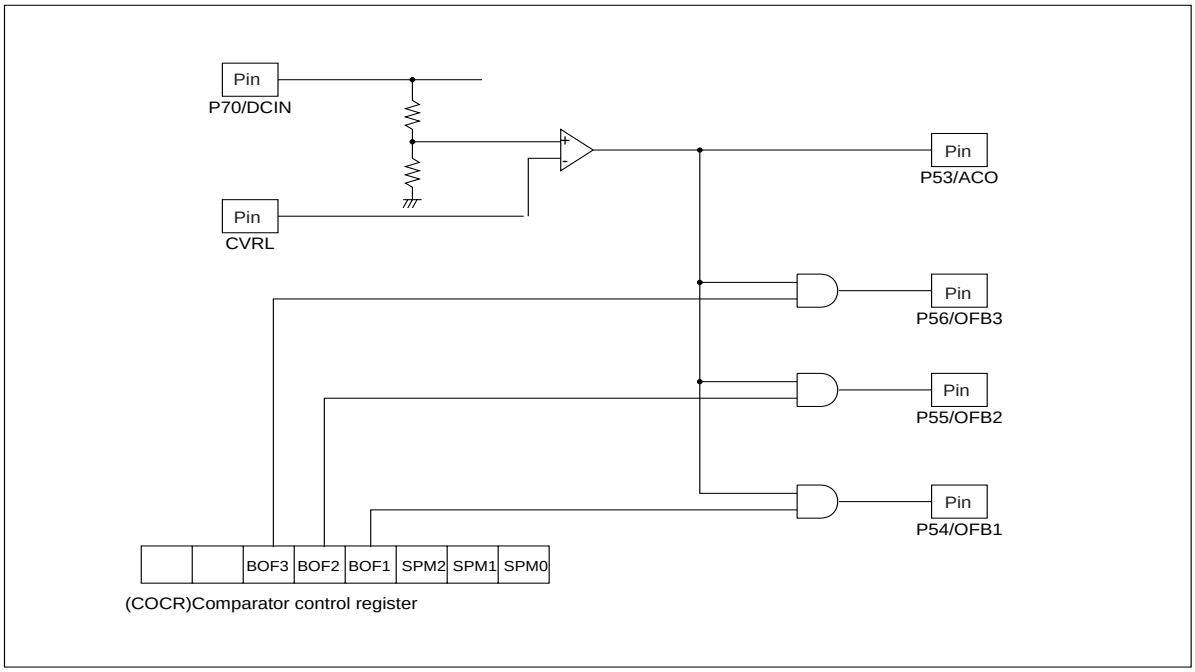



In parallel discharge control, discharge is allowed for all batteries unless power is not being supplied from the AC adapter.

If power is being supplied from the AC adapter, enable/disable of battery discharge is controlled by the battery discharge control signal output enable bits (COCR1: BOF1 to BOF3).

Figure 13.6-2 "Circuit for Parallel Discharge" shows the circuit for parallel discharge.

**Figure 13.6-2 Circuit for Parallel Discharge**



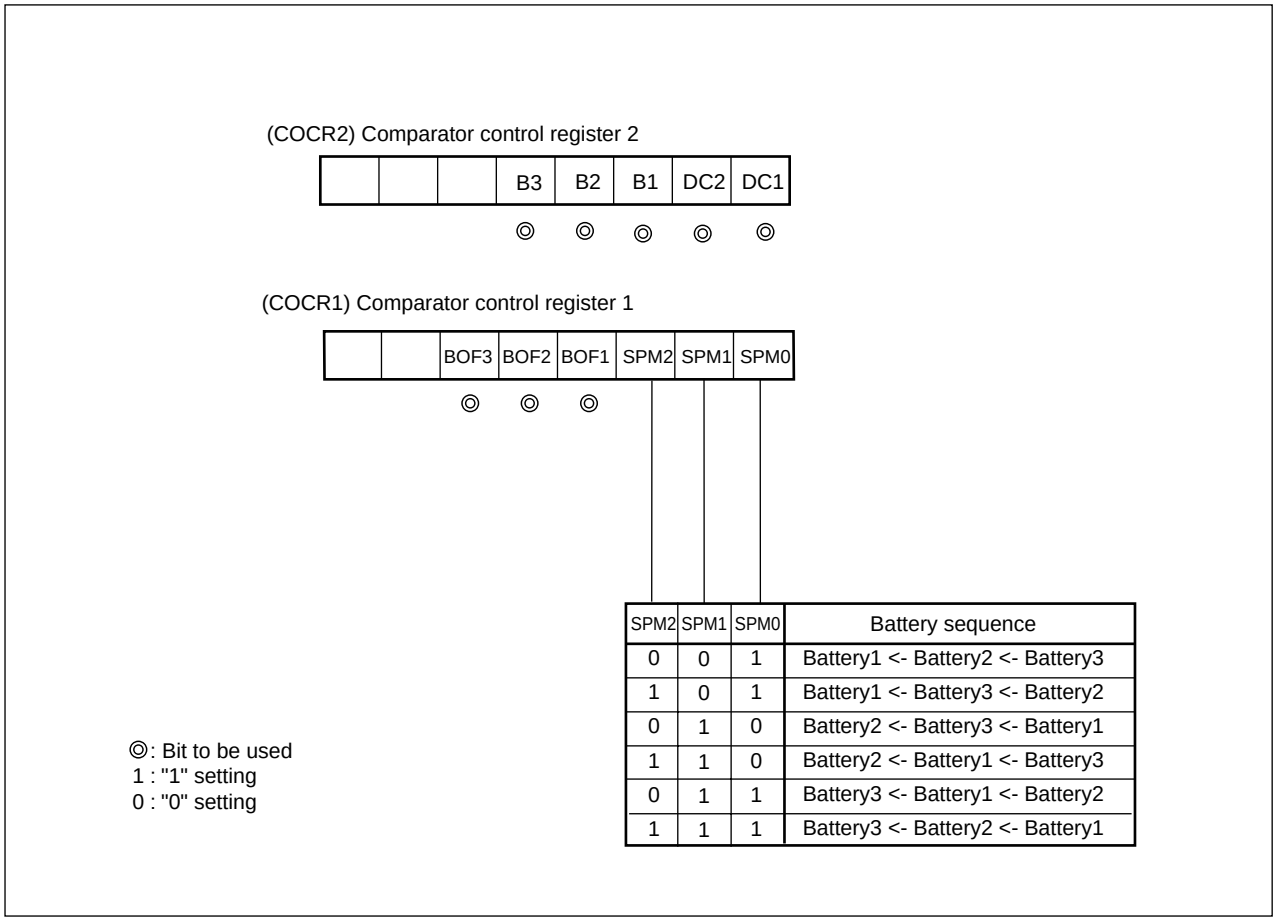
# 13.7 Operation of the Sequential Discharge Control

**This section describes operation in sequential discharge control.**

## ■ Operation of the Sequential Discharge Control

The setting shown in Figure 13.7-1 "Setting for Sequential Discharge Control" is required to perform the operation in the sequential discharge control.

**Figure 13.7-1 Setting for Sequential Discharge Control**




| SPM2 | SPM1 | SPM0 | Battery sequence |
|---|---|---|---|
| 0 | 0 | 1 | Battery1 <- Battery2 <- Battery3 |
| 1 | 0 | 1 | Battery1 <- Battery3 <- Battery2 |
| 0 | 1 | 0 | Battery2 <- Battery3 <- Battery1 |
| 1 | 1 | 0 | Battery2 <- Battery1 <- Battery3 |
| 0 | 1 | 1 | Battery3 <- Battery1 <- Battery2 |
| 1 | 1 | 1 | Battery3 <- Battery2 <- Battery1 |

In sequential discharge control, discharge is controlled in the order that is set by the sequence setting bits (COCR1:SPM0 to SPM2) when power is not being supplied from the AC adapter.

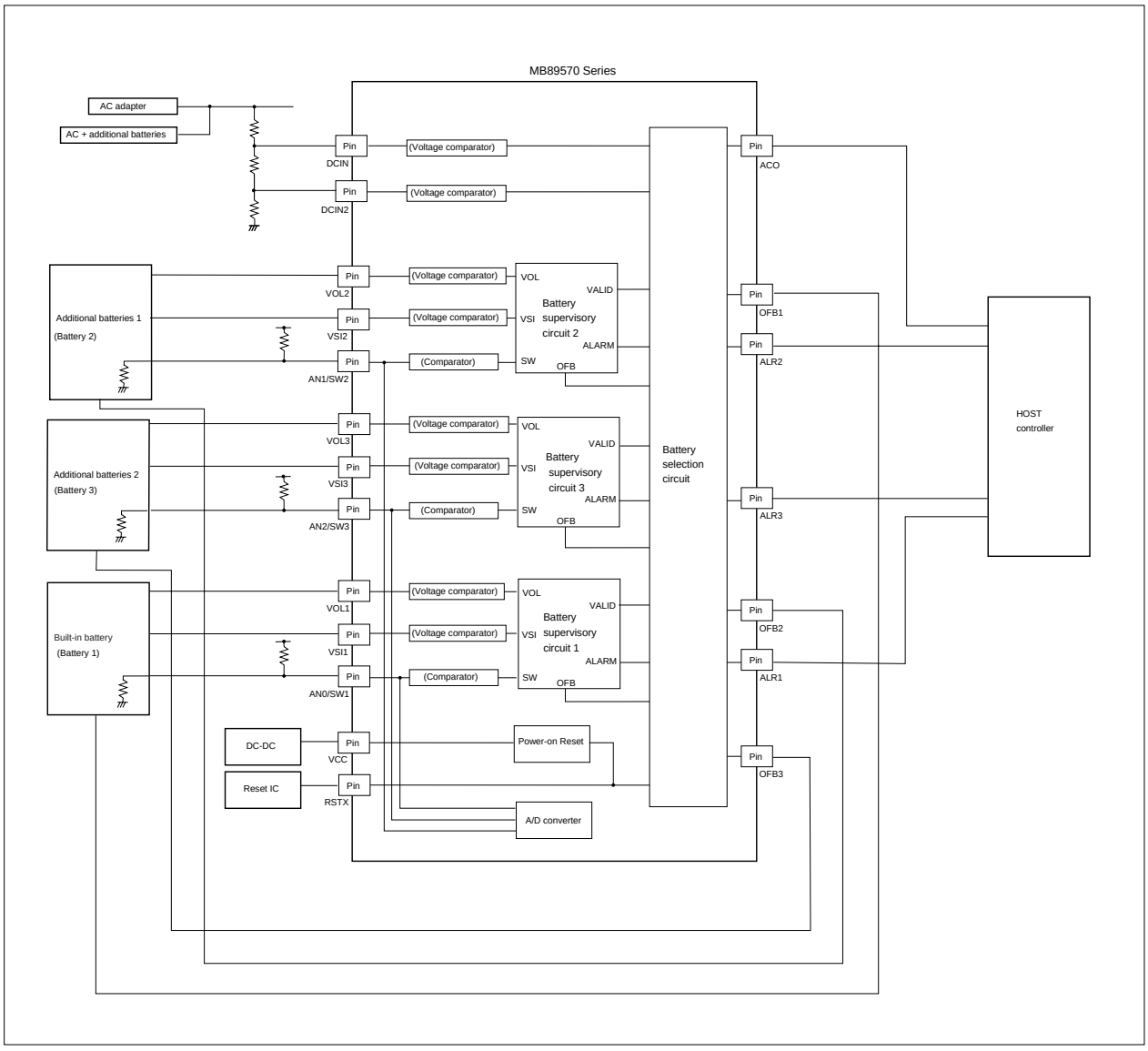
Also, discharge can forcibly be allowed by the battery discharge control signal output enable bits (COCR1:BOF1 to BOF3) even if power is being supplied from the AC adapter.

# 13.8 Sample Application

**This section shows an example of a comparator application.**

## ■ Sample Application

**Figure 13.8-1 Sample Application**



# CHAPTER 14 UART/SIO

**This chapter describes the functions and operations of the UART/SIO.**


14.1 "Overview of the UART/SIO"

14.2 "Configuration of the UART/SIO"

14.3 "Pins of the UART/SIO"

14.4 "Registers of the UART/SIO"

14.5 "UART/SIO Interrupt"

14.6 "Operation of the UART/SIO"

14.7 "Operation of the Operation Mode 0"

14.8 "Operation of the Operation Mode 1"

# 14.1 Overview of the UART/SIO

**The UART/SIO is a general-purpose serial data communication interface. Variable-length serial data can be transferred in clock synchronous or asynchronous mode. The NRZ transfer format is adopted and the transfer rate can be set with the dedicated baud rate generator, the external clock, or the internal timer.**

## ■ Functions of UART/SIO

The UART/SIO functions to transmit/receive serial data (serial I/O) to/from other CPUs and peripheral devices.

- Its full-duplex double buffer allows bidirectional transmission in full-duplex mode.
- A synchronous transfer mode or asynchronous transfer mode can be selected.
- With the built-in baud rate generator, 14 types of baud rates can be selected. In addition, free baud rates can be set using the externally input clock.
- The data length is variable. Seven to eight bits can be set when a parity bit is not attached, and eight to nine bits can be set when a parity bit is attached (See Table 14.1-1 "Operation Mode of UART/SIO").
- The NRZ (Non Return to Zero) method is adopted as the data transfer format.

**Table 14.1-1 Operation Mode of UART/SIO**

| Operation mode | Data length | | Synchronous mode | Stop bit length |
|---|---|---|---|---|
| | No parity | With parity | | |
| 0 | 7 | 8 | Asynchronous | 1 bit or 2 bits |
| | 8 | 9 | | |
| 1 | 8 | | Synchronous | – |

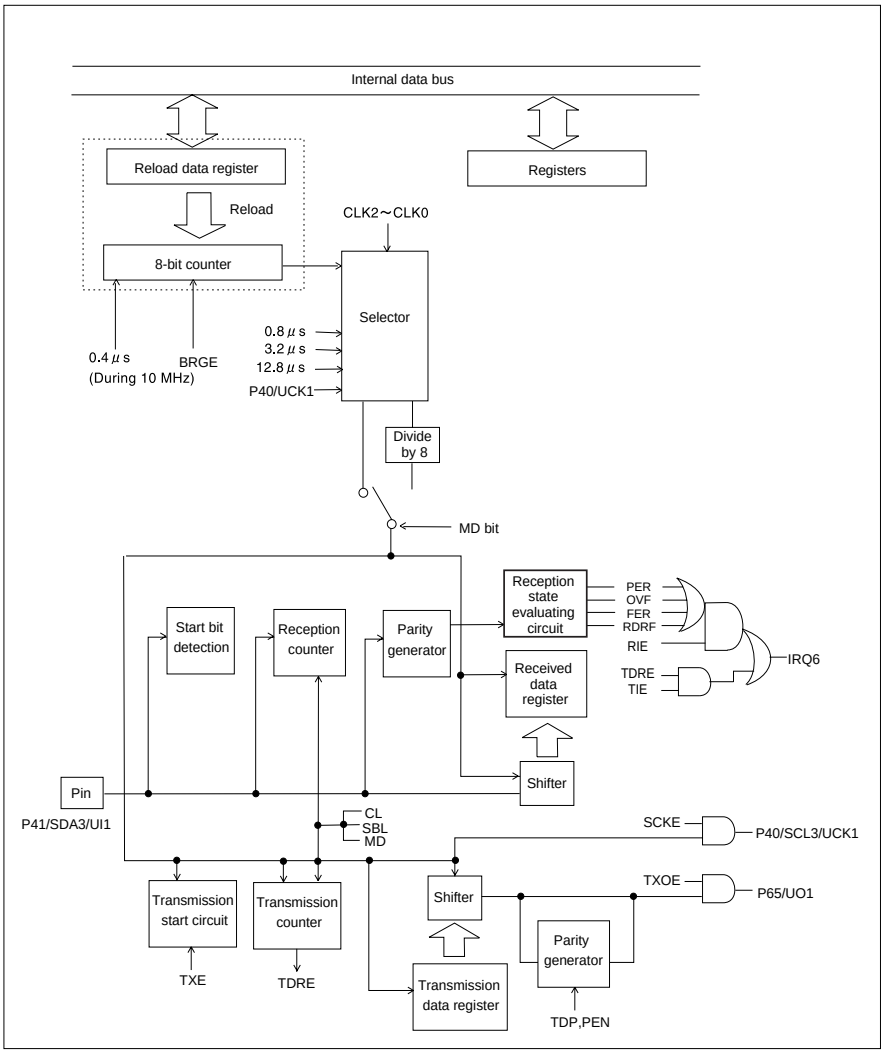
# 14.2 Configuration of the UART/SIO

**The UART/SIO consists of the following six blocks.**

- **Serial mode control register 1 (SMC1)**
- **Serial mode control register 2 (SMC2)**
- **Port generator reload register (SRC)**
- **Serial status and data register (SSD)**
- **Serial input data register (SIDR)**
- **Serial output data register (SODR)**

## ■ Block Diagram of UART/SIO

**Figure 14.2-1 UART/SIO Block Diagram**



- **Serial mode control register 1 (SMC1)**

  A register to control the operation mode of the UART/SIO. This register sets the presence/absence of parity, stop bit length, operation mode (data length), synchronous/asynchronous mode, and serial clock.

- **Serial mode control register 2 (SMC2)**

  A register to control the operation mode of the UART/SIO. This register sets the permission/prohibition of serial clock output, permission/prohibition of serial data output, switching between the serial port and the general-purpose port, and permission/prohibition of interrupts.

- **Baud rate generator reload register (SRC)**

  A register to control the UART/SIO data transfer rate (baud rate).

- **Serial status and data register (SSD)**

  A register that indicates the state of transmission/reception of the UART/SIO and errors.

- **Serial input data register (SIDR)**

  A register that holds received data. Serial input is converted and stored in this register.

- **Serial output data register (SODR)**

  A register that sets transmission data. The data written to this register is converted into serial data and output.

# 14.3 Pins of the UART/SIO

**This section shows the pins related to the UART/SIO and shows a block diagram of the pins.**

## ■ Pins Related to the UART/SIO

The pins related to UART/SIO include the clock I/O pin (P40/SCL3/UCK1), the serial data output pin (P65/U01), and the serial data input pin (P41/SDA3/UI1). All of which can be switched by the bridge circuit selection register (BRSR3) and the operating port selection bit (SMC2: SCKE).

### P40/SCL3/UCK1 pin

The P40/SCL3/UCK1 pin serves as a general-purpose I/O port (P40), a clock I/O pin (hysteresis input) of the UART/SIO (UCK1), and the clock line of the $I^2C$ bus (SCL3). When clock output is permitted (SMC2: SCKE = 1), this pin serves as a clock I/O pin of the UART/SIO (UCK1) irrespective of the value of the corresponding port direction register. At this time, do not select the external clock (SMC1: CLK2. CLK1, CLK0 is other than $100_B$). When this pin is used as a clock input pin of the UART/SIO, prohibit clock output (SMC: SCKE = 0) and set it to Hiz with the corresponding port data register (PDR4: bit0 = 1). Also at this time, select the external clock (SMC1: CLK2. CLK1, CLK0 = $100_B$).

### P65/U01

The P65/U01 pin serves as a general-purpose I/O port (P65) and a serial data output pin of the UART/SIO (U01). When serial data output is permitted (SMC2: TXOE = 1), this pin serves as a serial data output pin of the UART/SIO (U01) irrespective of the value of the corresponding port direction register.

### P41/SDA3/UI1 pin

The P41/SDA3/UI1 pin serves as a general-purpose I/O port (P41), a serial data input pin (hysteresis input) of the UART/SIO (UCK1), and the data line of the $I^2C$ bus (SDA3). When this pin is used as a serial data input pin of the UART/SIO, set it to Hiz with the corresponding port data register (PDR4: bit1 = 1).

## ■ Block Diagram of Pins Related to UART/SIO

**Figure 14.3-1 Block Diagram of Pins Related to UART/SIO**

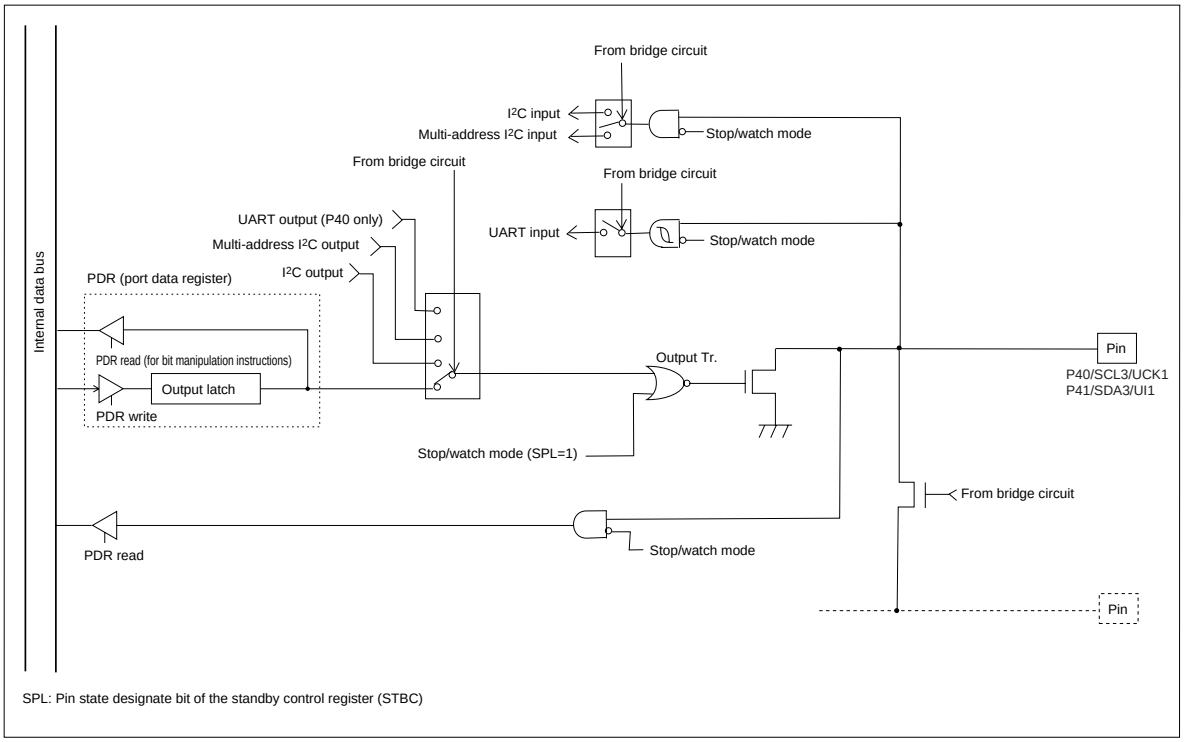
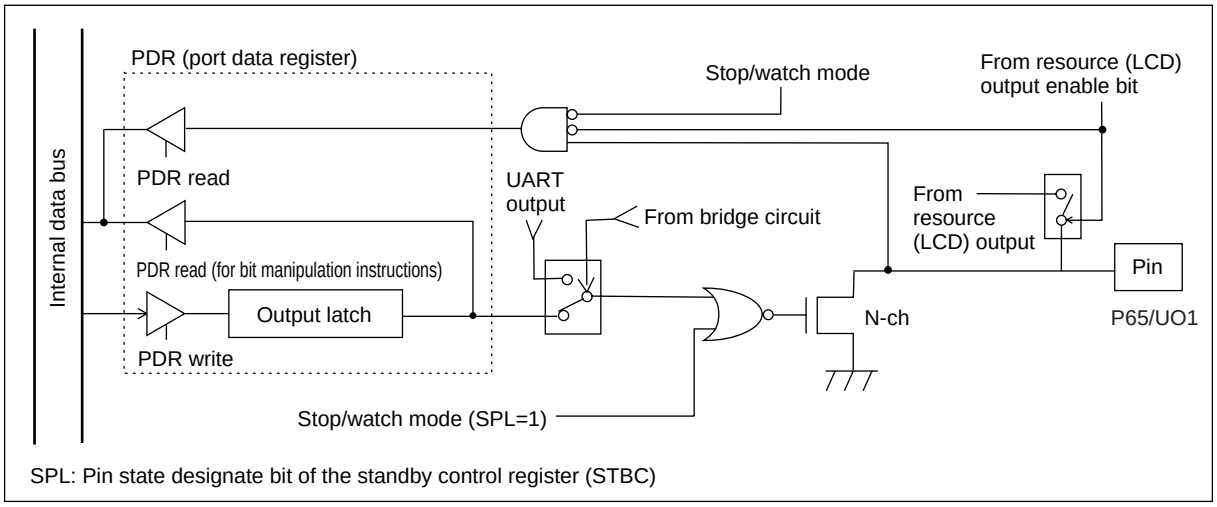



**Figure 14.3-2 Block Diagram of Pins Related to UART/SIO**




When the UART/SIO function is used, P33/UCK and P35/U0 must be pulled up externally.

# 14.4 Registers of the UART/SIO

## This section shows the registers related to the UART/SIO.

### ■ Registers Related to UART/SIO

**Figure 14.4-1 Registers Related to UART/SIO**

SMC1 (Serial mode control register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0010H | MD | PEN | TDP | SBL | CL | CLK2 | CLK1 | CLK0 | 00000000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

SMC2 (Serial mode control register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0011H | RERC | RXE | TXE | BRGE | TXOE | SCKE | RIE | TIE | 00000000B |
| | W | R/W | R/W | W | R/W | R/W | R/W | R/W | |

SSD (Serial status and data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0012H | PER | OVE | FER | RDRF | TDRE | - | - | - | 00001XXXB |
| | R | R | R | R | R | | | | |

SIDR (Serial input data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0013H | | | | | | | | | XXXXXXXXB |
| | R | R | R | R | R | R | R | R | |

SODR (Serial output data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0013H | | | | | | | | | XXXXXXXXB |
| | W | W | W | W | W | W | W | W | |

SRC (Baud rate generator reload register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0014H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
R : Read only
W : Write only
X : Undefined

# 14.4.1 Serial Mode Control Register 1 (SMC1)

**The serial mode control register 1 (SMC1) controls the operation mode of the UART/SIO. This register sets the presence/absence of parity, stop bit length, operation mode (data length), synchronous/asynchronous mode, and serial clock.**

## ■ Serial Mode Control Register 1 (SMC1)

**Figure 14.4-2 Serial Mode Control Register 1 (SMC1)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0010H | MD | PEN | TDP | SBL | CL | CLK2 | CLK1 | CLK0 | 00000000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

| CLK2 | CLK1 | CLK0 | Clock selection bit |
|---|---|---|---|
| 0 | 0 | 0 | 2-instruction cycle (0.8 μs/10MHz) |
| 0 | 0 | 1 | 8-instruction cycle (3.2 μs/10MHz) |
| 0 | 1 | 0 | 32-instruction cycle (12.8 μs/10MHz) |
| 0 | 1 | 1 | Dedicated baud rate generator |
| 1 | 0 | 0 | - |

| CL | Character bit length control bit |
|---|---|
| 0 | 7-bit length |
| 1 | 8-bit length |

| SBL | Stop bit length control bit |
|---|---|
| 0 | 1-bit length |
| 1 | 2-bit length |

| TDP | Parity polarity bit |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

| PEN | Parity control bit |
|---|---|
| 0 | No parity |
| 1 | With parity |

| MD | Mode control bit |
|---|---|
| 0 | Clock asynchronous mode (UART) |
| 1 | Clock synchronous mode (SIO) |

R/W : Read/write enabled
X : Undefined
□ : Initial value

**Table 14.4-1 Functions of Each Bit in Serial Mode Control Register 1 (SMC1)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | MD:<br>Mode control bit | • This bit specifies the operation mode of the UART/SIO. In asynchronous mode, the UART/SIO operates with the serial clock divided by eight. In clock synchronous mode, the UART/SIO operates with the selected serial clock. |
| Bit 6 | PEN:<br>Parity control bit | • This bit specifies whether there is parity in clock asynchronous mode. |
| Bit 5 | TDP:<br>Parity polarity bit | • This bit specifies the parity data attached at the time of serial transmission in clock asynchronous mode. At the time of serial reception, this bit checks the parity data. |
| Bit 4 | SBL:<br>Stop bit length control bit | • This bit specifies the stop bit length in clock asynchronous mode. At the time of serial transmission, this bit attaches a stop bit of the specified bit length. At the time of serial reception, this bit evaluates the stop bit with one bit length irrespective of the set value. |
| Bit 3 | CL:<br>Character bit length control bit | • This bit specifies the character bit length in clock asynchronous mode. |
| Bit 2<br>Bit 1<br>Bit 0 | CLK2 CLK1 CLK0:<br>Clock selection bits | • These bits select a serial clock. |

## 14.4.2 Serial Mode Control Register 2 (SMC2)

**The serial mode control register 2 (SMC2) controls the operation mode of the UART/SIO. This register sets the permission/prohibition of serial clock output, permission/prohibition of serial data output, switching between the serial port and the general-purpose port, and permission/prohibition of interrupts.**

### ■ Serial Mode Control Register 2 (SMC2)

**Figure 14.4-3 Serial Mode Control Register 2 (SMC2)**

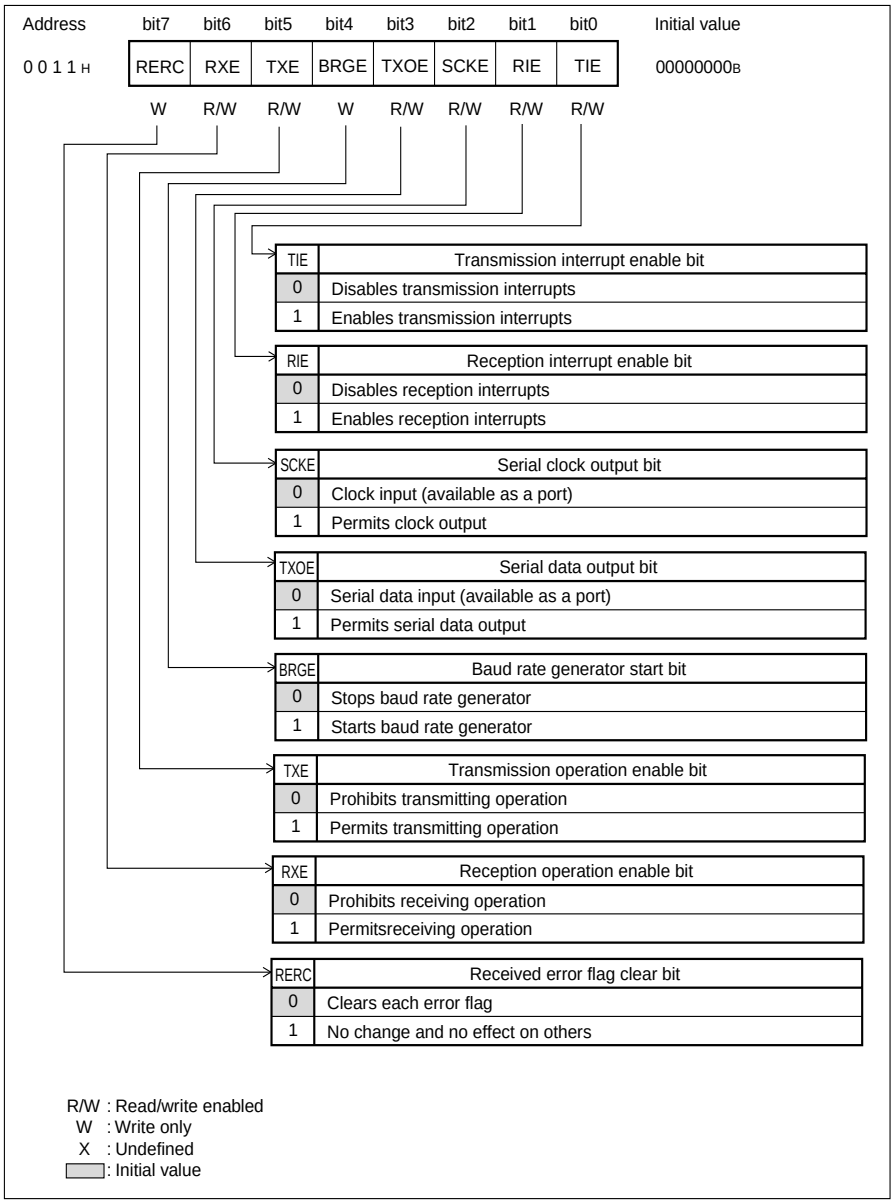


**Table 14.4-2 Functions of Each Bit in Serial Mode Control Register 2 (SMC2)**

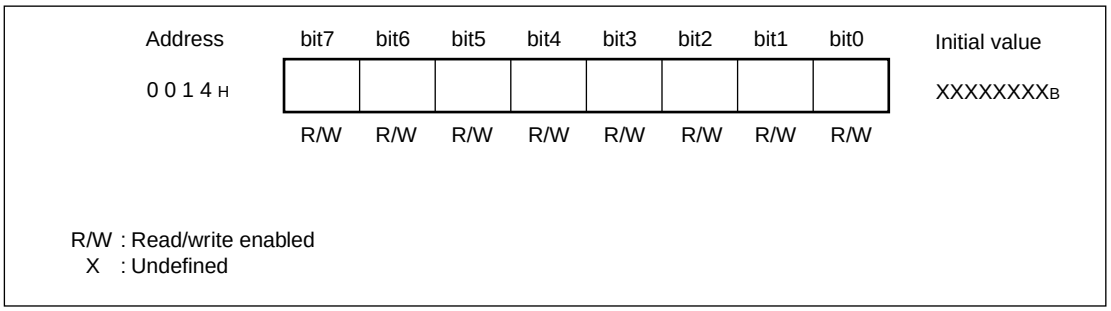
| Bit name | | Function |
|---|---|---|
| Bit 7 | RERC:<br>Received error flag clear bit | • When "0" is written to this bit, each error flag (PER/OVR/FER) in the SSD register is cleared. In read cycle, the value is always "1." |
| Bit 6 | RXE:<br>Receiving operation enable bit | • This bit permits the reception of serial data. When "0" is written to this bit during a receiving operation, the operation stops after data reception is completed and the receiving operation is prohibited. |
| Bit 5 | TXE:<br>Transmitting operation enable bit | • This bit permits the transmission of serial data. When "0" is written to this bit during a transmitting operation, the operation stops after data transmission is completed and the transmitting operation is prohibited. |
| Bit 4 | BRGE:<br>Baud rate generator start bit | • This bit starts the baud rate generator. |
| Bit 3 | TXOE:<br>Serial data output bit | • This bit controls the permission/prohibition of serial data output. |
| Bit 2 | SCKE:<br>Serial clock output bit | • This bit controls the I/O of the serial clock in clock synchronous mode. To enter the external clock into the P40/UCK pin, set it to input (PDR4: bit0 = 1). |
| Bit 1 | RIE:<br>Reception interrupt enable bit | • This bit enables reception interrupts. If reception interrupts are enabled when the RDRF bit is "1" or when each error flag is "1," a reception interrupt occurs immediately. |
| Bit 0 | TIE:<br>Transmission interrupt enable bit | • This bit enables transmission interrupts. If transmission interrupts are enabled when the TDRE bit is "1," a transmission interrupt occurs immediately. |

# 14.4.3 Baud Rate Generator Reload Register (SRC)

**The baud rate generator reload register (SRC) controls the UART/SIO data transfer rate (baud rate).**

## ■ Baud Rate Generator Reload Register (SRC)

**Figure 14.4-4 Baud Rate Generator Reload Register (SRC)**




When the clock selection bits CLK2-CLK0 are set to "011," the dedicated baud rate generator is selected as a serial clock. With this register, clocks of any baud rate can be set. A value can be written to this register only when the UART is stopped.

# 14.4.4 Serial Status and Data Register (SSD)

**The serial status and data register (SSD) indicate the state of transmission/reception of the UART/SIO and errors.**

■ **Serial Status and Data Register (SSD)**

**Figure 14.4-5 Serial Status and Data Register (SSD)**

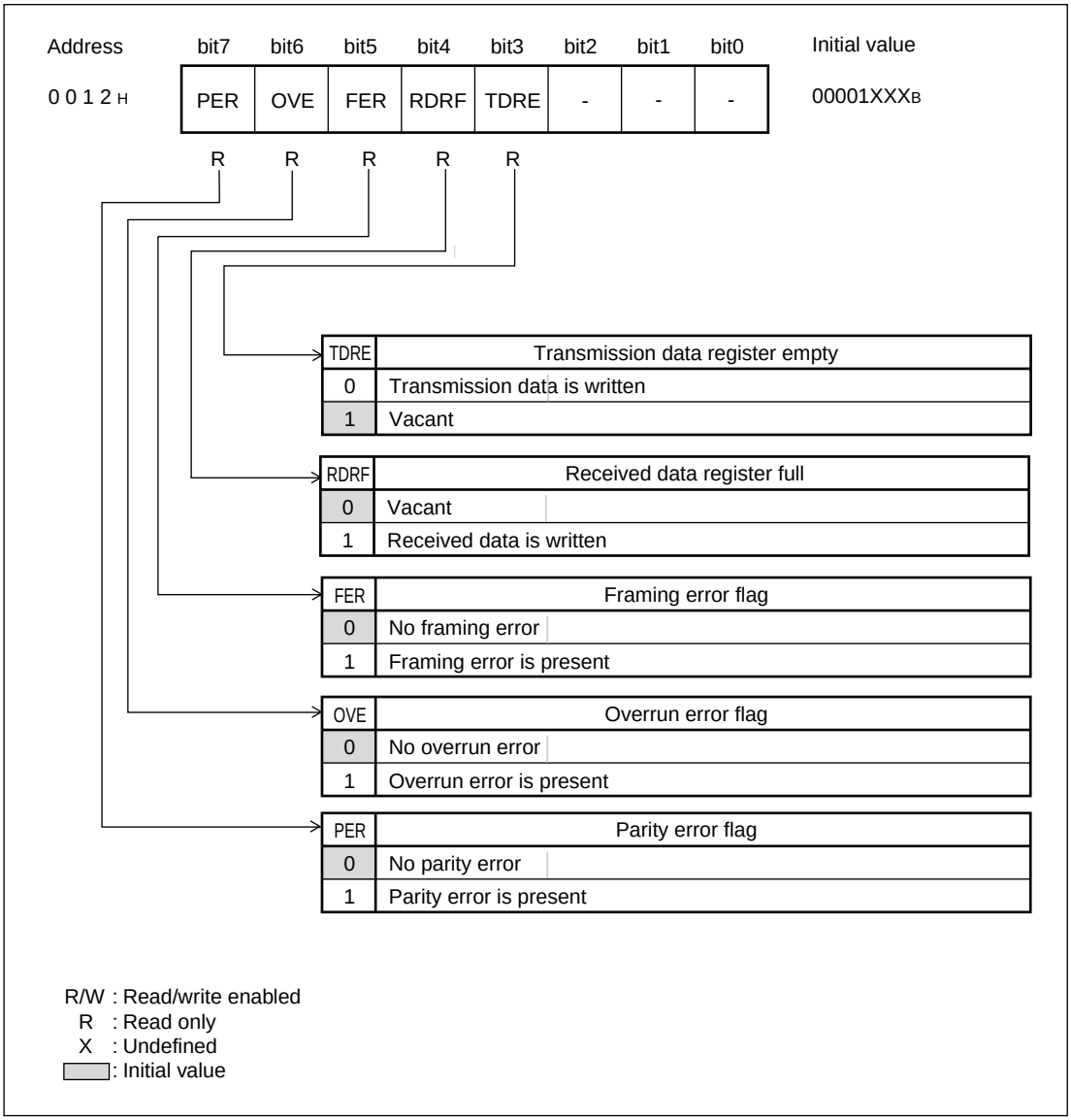


**Table 14.4-3 Functions of Each Bit in Serial Status and Data Register (SSD)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | PER:<br>Parity error flag | • This bit is set if a parity error occurs during reception and is cleared when "0" is written to the RERC bit in the SMC2 register. When this flag is set, the data in SIDR becomes invalid. If the PER bit is set when the RIE bit is set to "1," an interrupt occurs. |
| Bit 6 | OVE:<br>Overrun error flag | • This bit is set if an overrun error occurs during reception and is cleared when "0" is written to the RERC bit in the SMC2 register. When this flag is set, the data in SIDR becomes invalid. If the OVE bit is set when the RIE bit is set to "1," an interrupt occurs. |
| Bit 5 | FER:<br>Framing error flag | • This bit is set if an framing error occurs during reception and is cleared when "0" is written to the RERC bit in the SMC2 register. When this flag is set, the data in SIDR becomes invalid. If the FER bit is set when the RIE bit is set to "1," an interrupt occurs. |
| Bit 4 | RDRF:<br>Received data register full | • This bit is a flag indicating the state of the received data register (SIDR). This bit is set when the received data is loaded to the SIDR register and is cleared when the SIDR register is read. If the RDRF bit is set when the RIE bit is set to "1," an interrupt occurs. |
| Bit 3 | TDRE:<br>Transmission data register empty | • This bit is a flag indicating the state of the serial transmission data register (SODR). This bit is cleared when the transmission data is written to the SODR register and is set when the data is loaded to the shifter for transmission and transmission of the data starts. If the TDRE bit is set, an interrupt occurs. |
| Bit2<br>Bit1<br>Bit0 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |

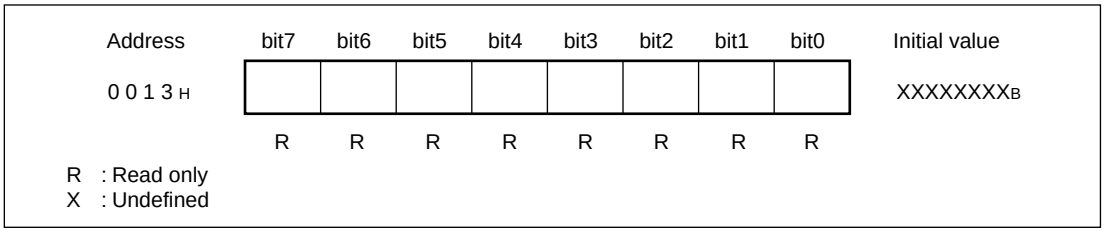
# 14.4.5 Serial Input Data Register (SIDR)

**The serial input data register (SIDR) is a register for inputting (receiving) serial data.**

## ■ Serial Input Data Register (SIDR)

Figure 14.4-6 "Serial Input Data Register (SIDR)" shows the bit configuration of the serial input data register.

**Figure 14.4-6 Serial Input Data Register (SIDR)**




The SIDR is a register for storing the received data. The serial data signal sent to the serial input pin (UI pin) is converted in the shift register and stored in this register.

- When the received data is set to this register successfully, the received data flag bit (RDRF) is set to "1." If the reception interrupt request is enabled, an interrupt occurs. When the received data is stored in this register in an interrupt or when checking the RDRF bit with the program, the RDRF flag is cleared by reading the description in this register.

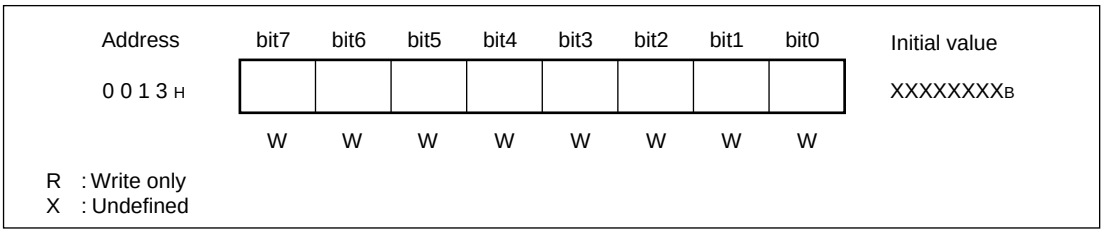
# 14.4.6 Serial Output Data Register (SODR)

**The serial output data register (SODR) is a register for outputting (transmitting) serial data.**

## ■ Serial Output Data Register (SODR)

Figure 14.4-7 "Serial Output Data Register (SODR)" shows the bit configuration of the serial output data register.

**Figure 14.4-7 Serial Output Data Register (SODR)**




When the data to be transmitted is written to this register after reading the SSD register in the transmission permitted state, the transmission data is transferred to the shift register for transmission, converted into serial data, and transmitted from the serial data output pin (UO pin).

When the transmission data is written to the SODR register, the transmission data flag bit is set to "0." After the transmission data is transferred to the shift register for transmission, the transmission data flag bit is set to "1" so that the next transmission data can be written in the register. If the interrupt request is enabled at this time, an interrupt occurs. The next transmission data can be written by generating an interrupt or when the transmission data flag bit is set to "1."

# 14.5 UART/SIO Interrupt

**The UART/SIO has three flags related to interrupts, the error flag bits (PER, OVE, FER), the received data flag bit (RDRF), and the transmission data flag bit (TDRE), as well as the following two interrupt sources.**

- **When the received data is transferred from the shift register for reception to the serial input data register (SIDR)**
- **When the transmission data is transferred from the serial output data register (SODR) to the shift register for transmission.**

### ■ Transmission Interrupt

When the output data is written to the SODR register, the data written to the SODR register is transferred to the shift register for internal transmission. When the register is ready to accept the next data, the TDRE bit is set to "1." If the transmission interrupt is enabled (SMC2: TIE = 1), an interrupt request to the CPU (IRQ6) occurs.

### ■ Reception Interrupt

After data is input up to the stop bit successfully, the RDRF bit is set to "1." If an overrun, parity, or error framing error has occurred, the bit of the corresponding error flag is set to "1."

These bits are set when the stop bit is detected. If the reception interrupt is enabled (SSD: RIE = 1), an interrupt request to the CPU (IRQ6) occurs.

### ■ Register and Vector Table Address Related to Interrupt of UART/SIO

**Table 14.5-1 Register and Vector Table Address Related to Interrupt of UART/SIO**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ6 | ILR2 (007CH) | L61 (bit 5) | L60 (bit 4) | $FFEE_H$ | $FFEF_H$ |

For interrupt operation, see Section 3.4.2 "Interrupt Processing".

# 14.6 Operation of the UART/SIO

**This section describes the operation of the UART/SIO.**
**The UART/SIO has ordinary serial communication functions (operation modes 0 and 1).**

## ■ Operation of UART/SIO

### ❍ Operation modes

The UART/SIO has two operation modes: clock synchronous mode (SIO) and clock asynchronous mode (UART). (See Table 14.1-1 "Operation mode of UART/SIO".)

# 14.7 Operation of the Operation Mode 0

**Operation mode 0 operates in clock asynchronous mode.**

## ■ Explanation of Operation Mode 0 of UART/SIO

The serial clock is selected, with bits CLK2 to CLK0 in the SMC1 register, from among three types of internal clocks, an external clock, and a baud rate generator output. When the external clock is selected, the clock must be entered.

In CLK asynchronous mode, the shift clock selected with bits CLK2 to CLK0 is divided by eight and data can be transferred in the range between -2% and +2% of the selected baud rate. The baud rate calculation expressions for the internal and external clocks and the baud rate generator are shown in the following.

**Figure 14.7-1 Baud Rate Calculation Expression for Internal and External Clocks**

$$\text{Baud rate value} = \frac{1}{8 \times \boxed{\text{Clock selected with the bits CLK2 to CLK0}}} \text{ [bps]}$$

**Figure 14.7-2 Baud Rate Calculation Expression when the Dedicated Baud Rate Generator is Used**

$$\text{Baud rate value} = \frac{1}{8 \times 2 \times \boxed{\begin{matrix}64/F_{CH}\\16/F_{CH}\\8/F_{CH}\\4/F_{CH}\end{matrix}} \times \boxed{\text{SRC register value (SRC)}}} \text{ [bps]}$$

Clock gear selection

$F_{CH}$: Main clock oscillation frequency

**Table 14.7-1 Example of the Asynchronous Transfer Rate with the Baud Rate Generator (in 4/ Fch Clock Gear)**

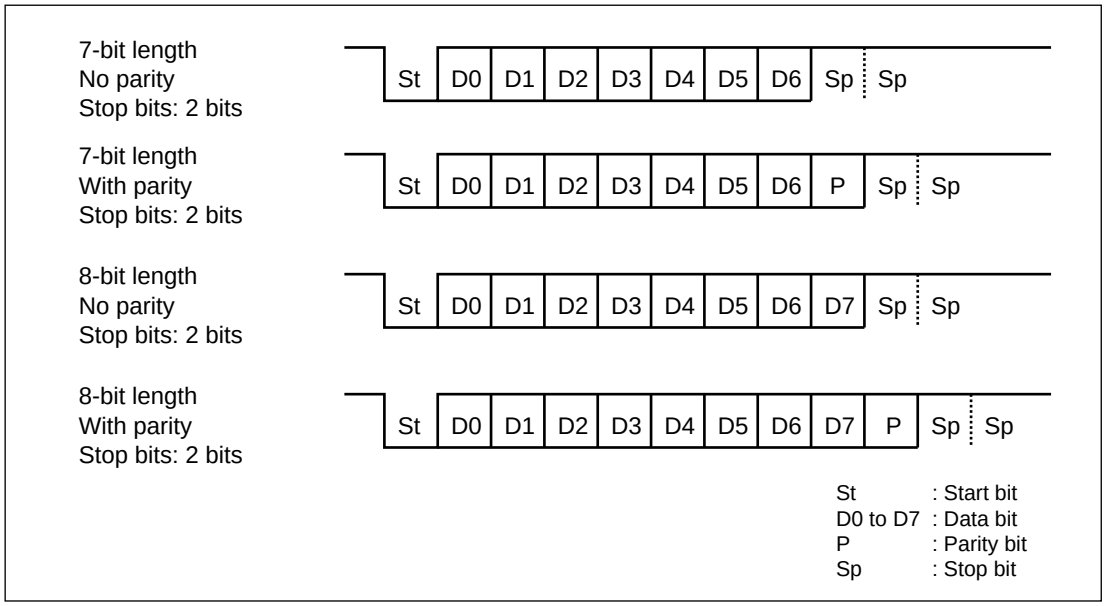
| Operating frequency | 10 MHz | 8 MHz | 7.3728 MHz | 4.9152 MHz |
|---|---|---|---|---|
| **Instruction cycle** | **0.4 µs** | **0.5 µs** | **0.54 µs** | **0.81 µs** |
| Baud rate<br><br>Values in parentheses indicate SRC register set values | 78125(n=2) | 62500(n=2) | – | 76800(n=1) |
| | 39062(n=4) | 31250(n=4) | 38400(n=3) | 38400(n=2) |
| | 19531(n=8) | 17857(n=7) | 19200(n=6) | 19200(n=4) |
| | 9765(n=16) | 9615(n=13) | 9600(n=12) | 9600(n=8) |
| | 4882(n=32) | 4807(n=26) | 4800(n=24) | 4800(n=16) |
| | 2403(n=65) | 2403(n=52) | 2400(n=48) | 2400(n=32) |
| | 1201(n=130) | 1201(n=104) | 1200(n=96) | 1200(n=64) |
| | – | 600(n=208) | 600(n=192) | 600(n=128) |
| | – | – | – | 300(n=0) |

## ■ Transfer Data Format

The UART/SIO can only use the NRZ (Non Return to Zero) format data. Figure 14.7-3 "Transfer Data Format" shows the data format. In the following example, the stop bit length is two bits.

As shown in Figure 14.7-3 "Transfer Data Format", data transfer always starts with the start bit ("L" level), followed by the data bit length specified as the LSB first, and ends with the stop bit ("H" level). In an idle state, it is at the "H" level.

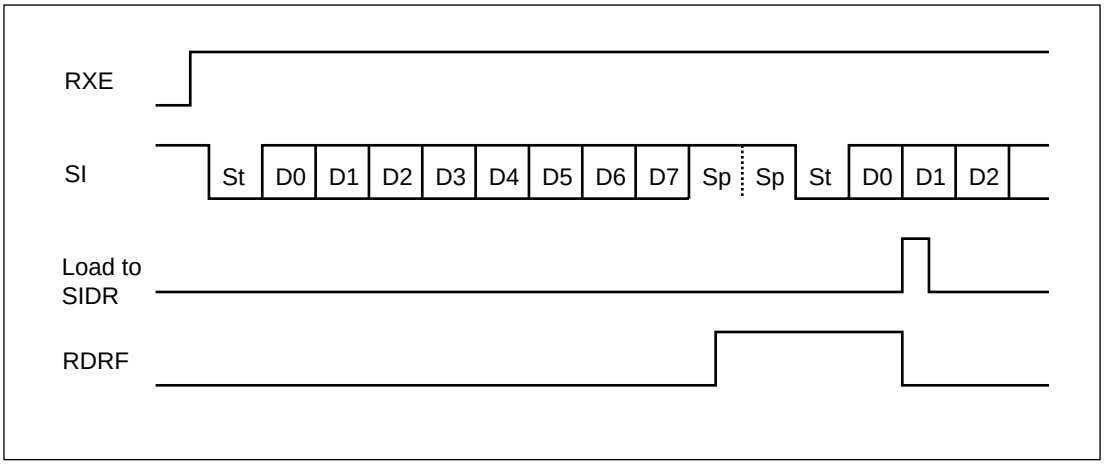
**Figure 14.7-3 Transfer Data Format**




## ■ Receiving Operation in CLK Asynchronous Mode

Select the baud rate clock with bits CLK2 to CLK0 in the SMC2 register. For the baud rate clock, see Figure 14.7-1 "Baud Rate Calculation Expression for Internal and External Clocks" and Figure 14.7-2 "Baud Rate Calculation Expression when the Dedicated Baud Rate Generator is Used". In a receiving operation, reception is permitted when the RXE bit in the SMC1 register is "1" and the receiving operation starts at the first falling edge of the input data (detection of the start bit). When the receiving operation is completed, the RDRF bit in the SSD register is set to "1" and the received data is loaded to the SIDR register. If the RDRF bit is set to "1" when the RIE bit is "1," a reception interrupt to the CPU is generated. If any of the three errors (PER/OVE/FER) is detected when reception is completed, the RDRF bit is not set to "1" and the received data is not loaded to the SIDR register. Therefore, the value in the SIDR register is the previously received data. Unless the RXE bit is set to "0," the receiving operation is continued whenever a start bit is detected even if an error flag is set.

**Figure 14.7-4 Receiving Operation of CLK Asynchronous Mode**



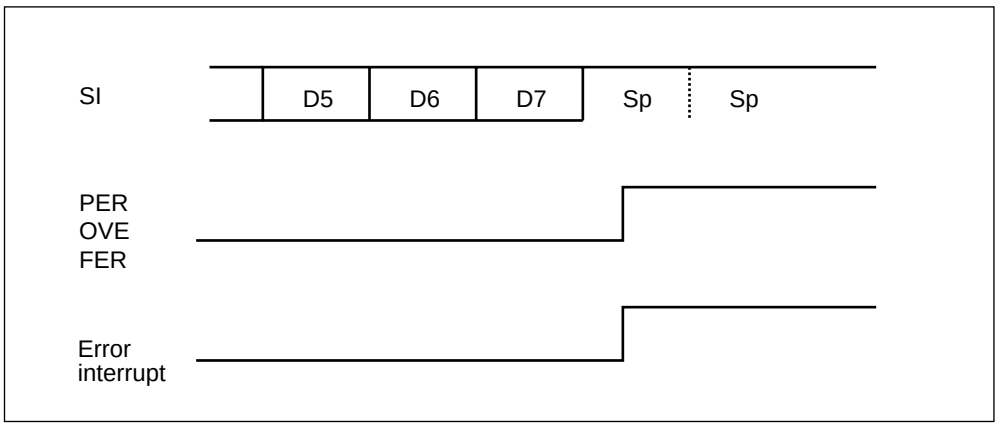

If "0" is written to the RXE bit of the SMC2 register during a receiving operation, the receiving operation is prohibited after data reception is completed.

## ■ Reception Error in CLK Asynchronous Mode

In CLK asynchronous mode, three types of errors are detected. When a parity error, overrun error, or framing error is detected, the PER, OVE, or FER bit in the SSD register is set to "1," respectively.

The detection of these errors are performed at the end of reception as shown in the following. When any of these errors is detected, RDRF is not set and the received data is not loaded to the SIDR register. Therefore, the value in the SIDR register is the previously received data. By writing "0" to the RERC bit in the SCM2 register, all of the three error flags are cleared.

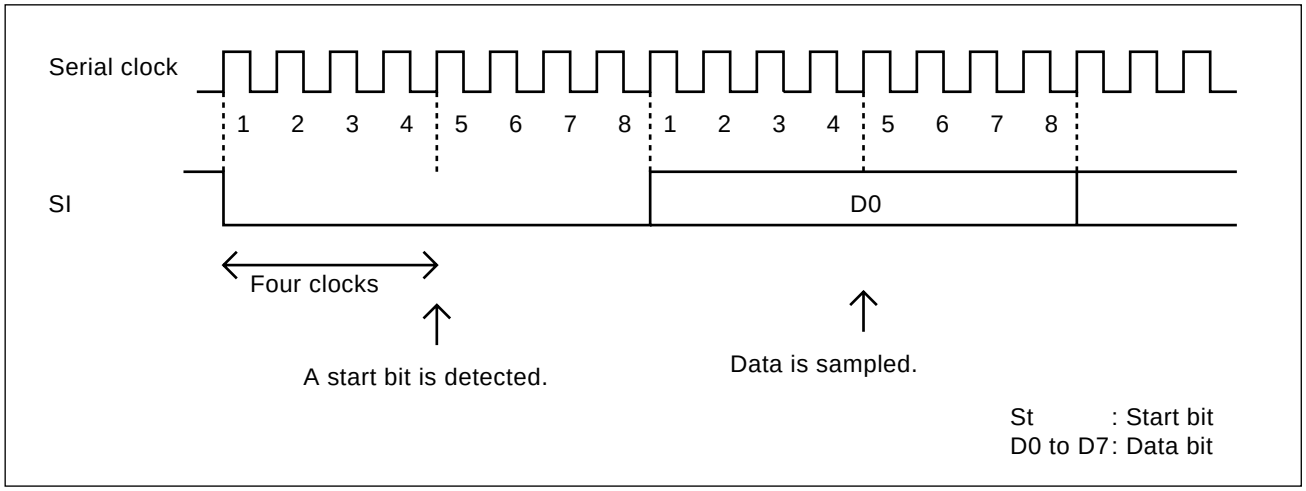
**Figure 14.7-5 Reception Error Setting Timing**




## ■ Detecting the Start Bit At Receiving Operation

When the "L" level remains for four clocks with the selected serial clock (generator output, etc.) after the first falling edge of the input data, the UART/SIO regards it as a start bit. After the start bit is detected, data is sampled at the rising edge of the fifth clock of the serial clock after the start bit is detected.

**Figure 14.7-6 Detecting a Start Bit**

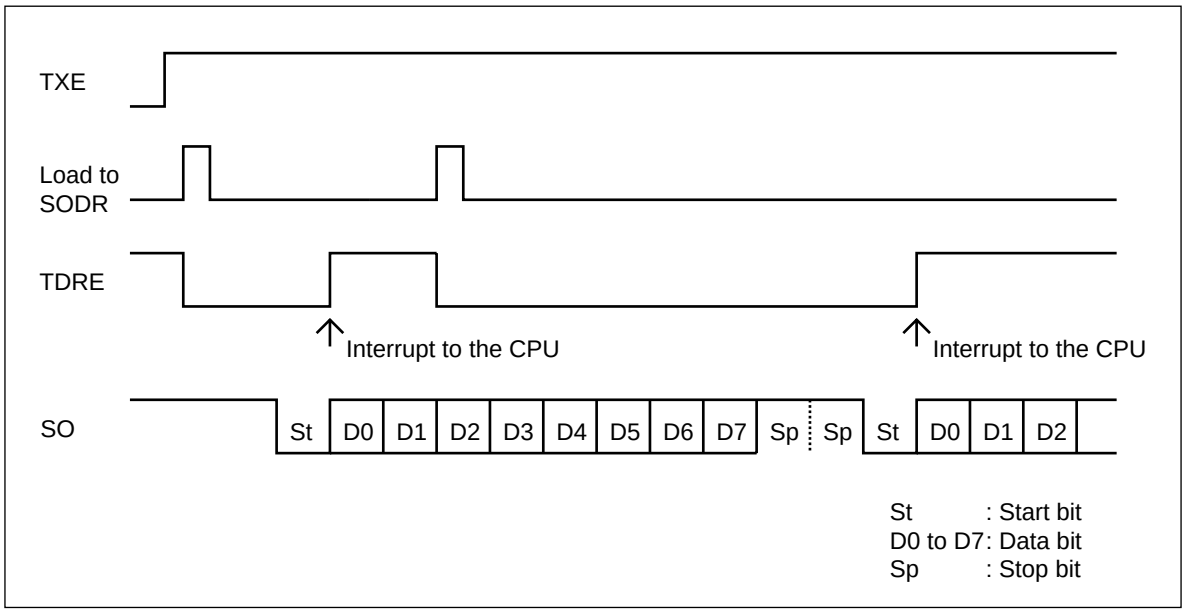


## ■ Transmitting Operation in CLK Asynchronous Mode

If transmission data is written to the SODR register when the TXE bit in the SMC2 register is "1," the TDRE bit in the SSD register is cleared and a transmitting operation starts. When the data in the SODR register is loaded to the shifter and the output of transmission data starts, the TDRE bit in the SSD register is set. If data is written to the SODR register when data is being transmitted (when the TDRE bit is set to "1"), the TDRE bit is cleared and data is transmitted continuously following the transmission of the specified bit length data.

If "0" is written to the TXE bit in the SMC2 register during a transmitting operation, the transmitting operation is prohibited following the transmission of the specified bit length data when the SODR register is vacant (when the TDRE bit is set to "1"). When there is data in the SODR register (when the TDRE bit is set to "1"), the transmitting operation is prohibited after the data in the SODR register is transmitted.

**Figure 14.7-7 Transmission in CLK Asynchronous Mode**



# 14.8 Operation of the Operation Mode 1

## Operation mode 1 operates in clock synchronous mode.

### ■ Explanation of UART/SIO Operation Mode

In CLK synchronous mode, the clock is selected, with bits CLK2 to CLK0 in the SMC1 register, from among three types of internal clocks, an external clock, and a baud rate generator output. Shift operation is performed with the selected clock as a shift clock. When the external clock is entered, set the SCKE bit to "0".

When the internal clock or the output of the baud rate generator is output as a shift clock, set the SCKE bit to "1". The baud rate calculation expressions for the internal and external clocks and the baud rate generator are shown in the following

**Figure 14.8-1 Baud Rate Calculation Expression for the Internal and External Clocks (Operation Mode 1)**

$$\text{Baud rate value} = \frac{1}{\boxed{\text{Clock selected with bits CLK2 to CLK0}}} \text{ [bps]}$$

**Figure 14.8-2 Baud Rate Calculation Expression when the Dedicated Baud Rate Generator is Used (Operation Mode 1)**

$$\text{Baud rate value} = \frac{F_{CH}}{2 \times \boxed{\begin{matrix}64/F_{CH}\\16/F_{CH}\\8/F_{CH}\\4/F_{CH}\end{matrix}} \times \boxed{\text{SRC register value (SRC)}}} \text{ [bps]}$$

Clock gear selection

$F_{CH}$: Main clock oscillation frequency

## ■ 8-bit Receiving Operation at Operation Mode 1

In reception in operation mode 1, use each register as shown in the following.

**Figure 14.8-3 Registers Used During Reception in Operation Mode 1**

SMC 1 (Serial mode control register 1)

| bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|
| MD | PEN | TDP | SBL | CL | CLK2 | CLK1 | CLK0 |
| 1 | 0 | 0 | 0 | 1 | ◎ | ◎ | ◎ |

SMC 2 (Serial mode control register 2)

| bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|
| RERC | RXE | TXE | BRGE | TXOE | SCKE | RIE | TIE |
| ◎ | ◎ | ◎ | ◎ | 0 | ◎ | ◎ | × |

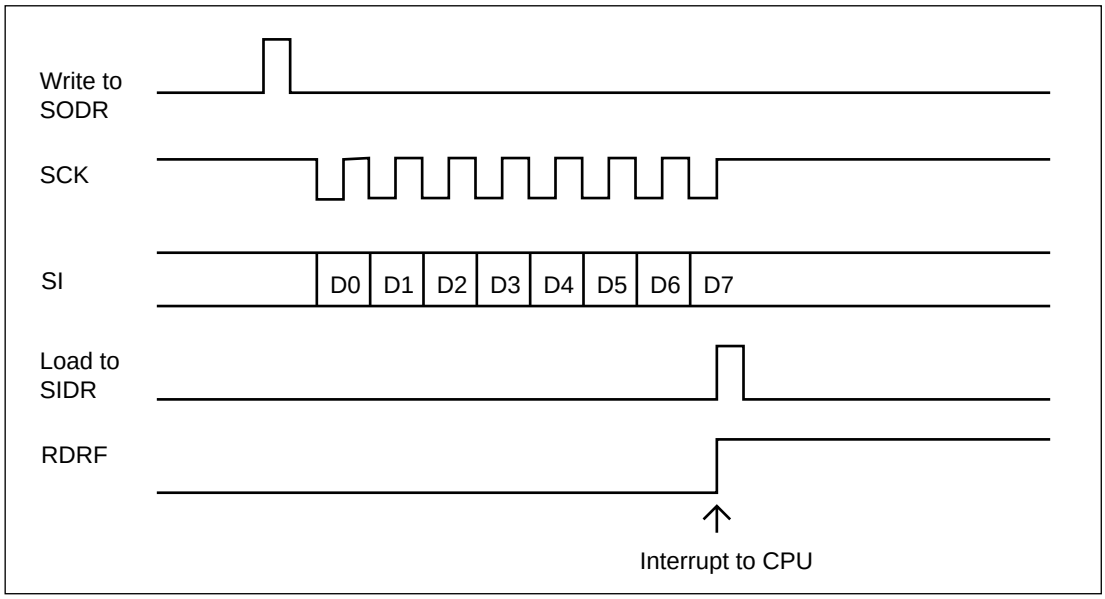
SSD (Serial status and data register)

| bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|
| PER | OVE | FER | RDRF | TDRE | - | - | - |
| × | ◎ | × | ◎ | × | | | |

◎ : Used bit
X : Unused bit
1 : 1 is set
0 : 0 is set

A receiving operation is permitted by setting the TXE/RXE bits to "11" and started by writing to the SODR register. The receiving operation is performed in synch with the rising edge of the shift clock. When the reception of 8-bit data is completed, the data in the shifter is loaded to the SIDR register and the RDRF flag is set to "1." When RIE is "1" at this time, an interrupt request to the CPU is generated. If an overrun error is detected at the end of reception, data is not loaded to the SIDR register. If "0" is written to the RXE bit during the receiving operation, the receiving operation is stopped after reception of the 8-bit data. In the serial operation stop state, maintain the input of the serial clock at the "H" level (irrespective of the value of the RXE bit).

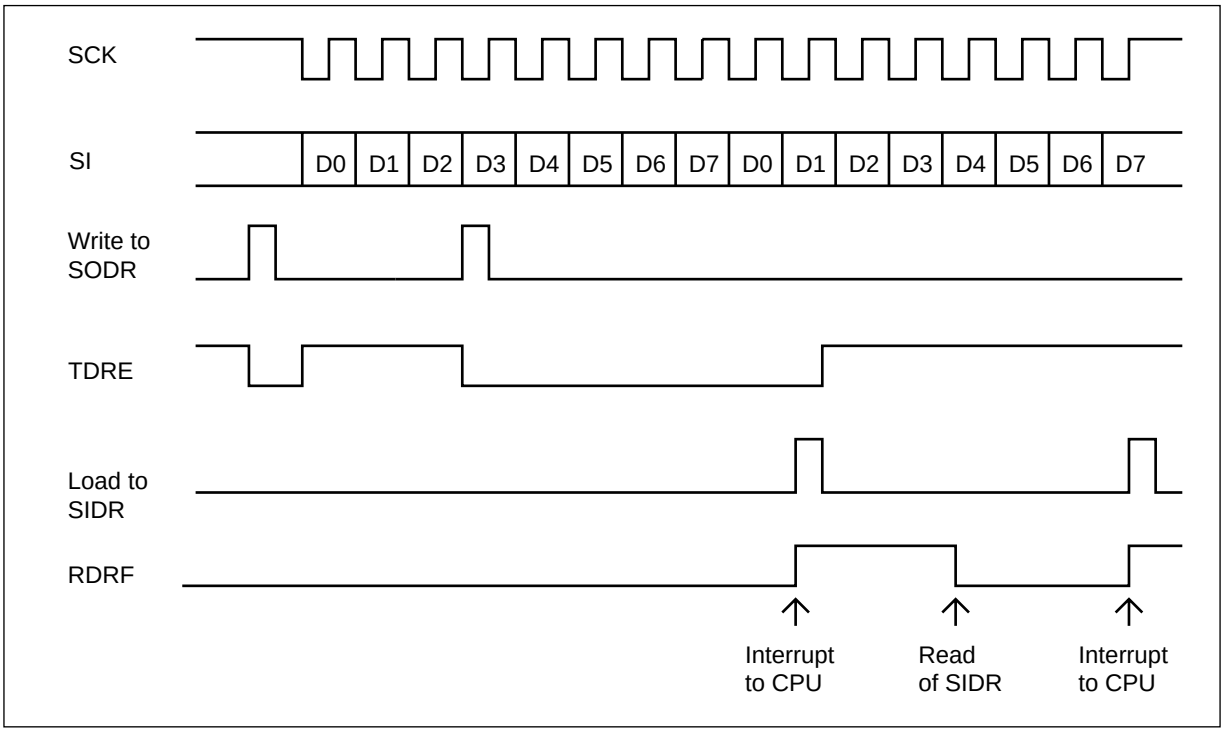
**Figure 14.8-4 8-bit Data Receiving Operation in CLK Synchronous Mode**




## ■ Continuous Receiving Operation

In CLK synchronous mode, not only an 8-bit data receiving operation but also a continuous receiving operation can be performed. In the continuous receiving operation, the TIE bit in the SMC2 register and the TDRE bit in the SSD register are used in addition to the registers used in the 8-bit data receiving operation. The receiving operation is permitted by setting the TXE/RXE bits to "11" and started by writing to the SODR register. The receiving operation is performed in synch with the rising edge of the shift clock. When a shift operation starts, the TDRE bit is set to "1." When TIE is "1" at this time, an interrupt to the CPU is generated. By writing to the SODR register before the shift operation of 8-bit data is completed, the next shift operation is permitted and the receiving operation is performed continuously after the reception of 8-bit data. When the reception of 8-bit data is completed, the data in the shifter is loaded to the SIDR register and the RDRF flag is set to "1." When RIE is "1" at this time, an interrupt request to the CPU is generated. If an overrun error is detected at the end of reception, data is not loaded to the SIDR register and the description in the SIDR register is the previously received data. By reading the SIDR register, a reception interrupt (RDRF) is cleared. By writing "0" to the RXE bit, the receiving operation is stopped. If "0" is written to the RXE bit during the receiving operation, the receiving operation is stopped after 8-bit data is received.

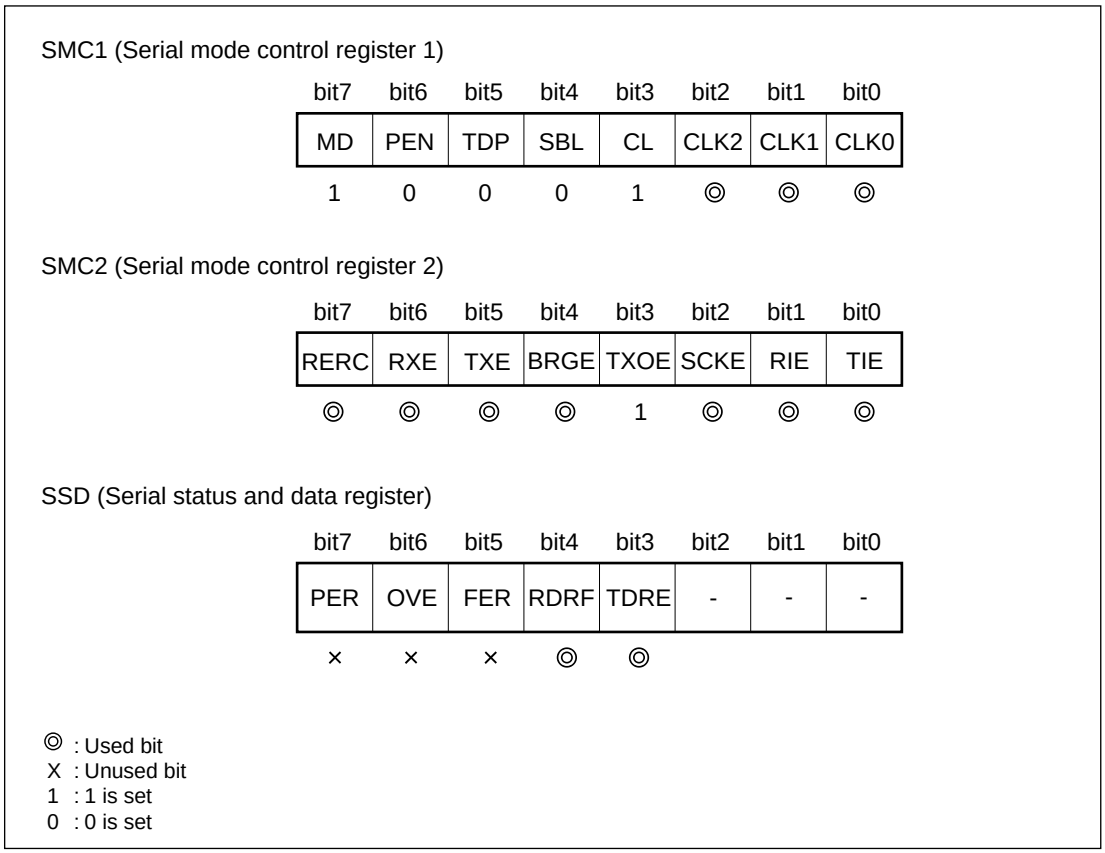
**Figure 14.8-5 Continuous Receiving Operation in CLK Synchronous Mode**




## ■ 8-bit Transmitting Operation at Operation Mode 1

In transmission in operation mode 1, use each register as shown in the following.

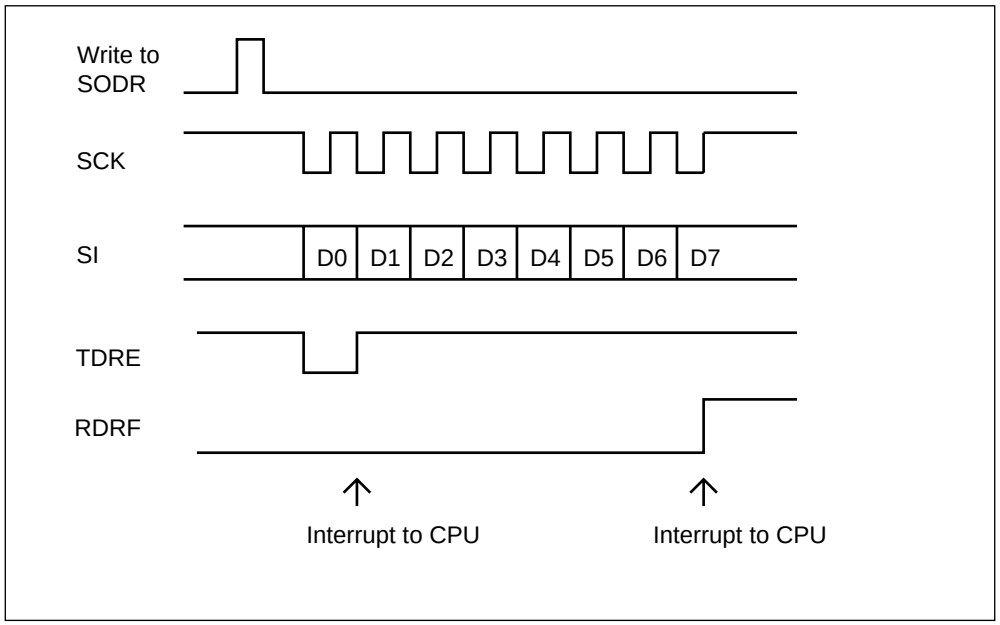
**Figure 14.8-6 Registers During Transmission in Operation Mode 1**




A transmitting operation is permitted by setting the TXE/RXE bits to "11" and started by writing data to the SODR register. When transmitting operation is started, the data written to the SODR register is loaded to the shifter and the shift operation is performed. When the data in the SODR register is loaded to the shifter, the TDRE flag is set to "1." When TIE is "1" at this time, an interrupt request to the CPU is generated. Output of the serial data is permitted with TXOE = "1" and is output in synch with the falling edge of the shift clock.

If "0" is written to the TXE bit during the transmitting operation, the operation is stopped after 8-bit data is transmitted. After the 8-bit data is transmitted, the RDRF bit is set to "1." If RIE is "1" at this time, an interrupt request to the CPU is generated. Data transmission starts with bit 0 and ends with bit 7. In the serial operation stop state, maintain the input of the serial clock at the "H" level (irrespective of the value of the TXE bit).

**Figure 14.8-7 8-bit Transmitting Operation in CLK Synchronous Mode**

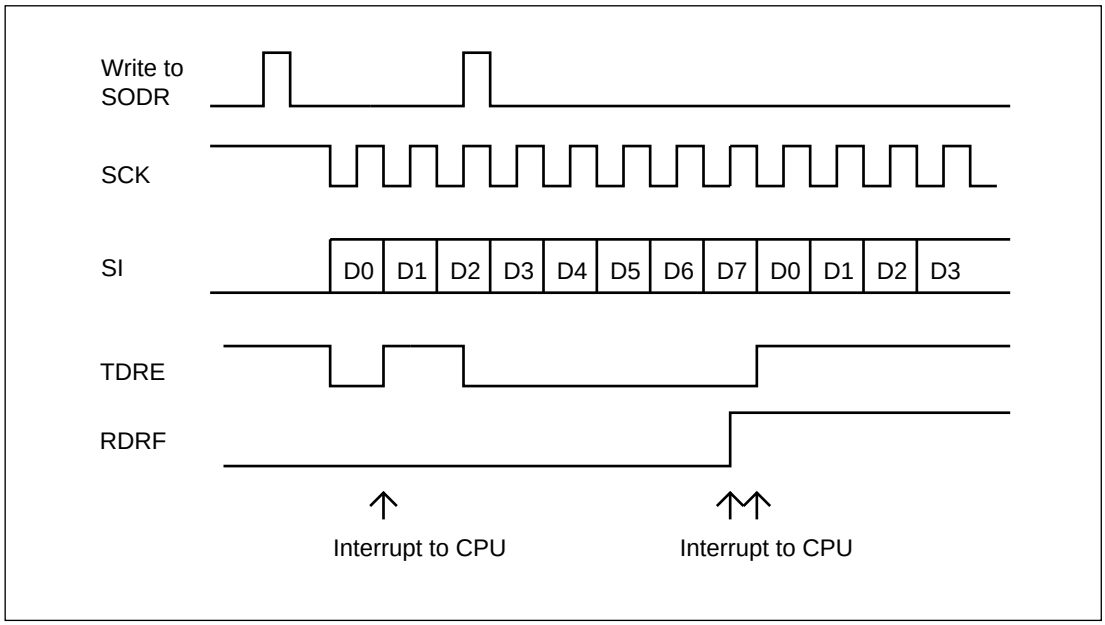



## ■ Continuous Transmission at Operation Mode 1

In CLK synchronous mode, not only 8-bit data transmission but also continuous transmission can be performed. The transmitting operation is performed by setting the TXE/RXE bits to "11" and writing data to the SODR register. When transmission starts, the data written to the SODR register is loaded to the shifter and the shift operation is performed. When the data in the SODR register is loaded to the shifter, the TDRE flag is set to "1." When TIE is set to "1" at this time, an interrupt request to the CPU is generated.

A continuous operation is performed by writing the next transmission data to the SODR register during the transmitting operation when the TDRE bit is "1" (the SODR register is vacant). By writing data to the SODR register, the TDRE bit is cleared. After 8-bit data is transmitted, the data written to the SODR register is loaded to the shifter and the transmitting operation is performed continuously. By writing "0" to the TXE bit, the transmitting operation is stopped. If "0" is written to the TXE bit during the transmitting operation, the transmitting operation is stopped after 8-bit data is transmitted when the SODR register is vacant (when the TDRE bit is "1"). When there is data in the SODR register (when the TDRE bit is "0"), the transmitting operation is stopped after the data in the SODR register is transmitted. When 8-bit data transmission is completed, the RDRF bit is set to "1." When RIE is "1" at this time, an interrupt request to the CPU is generated.

**Figure 14.8-8 Continuous Receiving Operation in CLK Synchronous Mode**



# CHAPTER 15 $I^2C$

**This chapter describes the functions and operations of the $I^2C$.**


15.1 "Overview of the $I^2C$"

15.2 "Configuration of the $I^2C$"

15.3 "Pins of the $I^2C$"

15.4 "Registers of the $I^2C$"

15.5 "$I^2C$ Interrupts"

15.6 "Operation of the $I^2C$"

15.7 "Notes on Using the $I^2C$"

15.8 "Operation of Timeout Detection Function"

# 15.1 Overview of the I²C

**The I²C is a simple bidirectional bus consisting of two wires that transfer data among devices. These two I²C bus interfaces allow internal devices requiring address data to connect to one another with a minimum number of circuits, making it possible to construct less expensive hardware using a fewer number of PCBs.**

**The I²C interface that supports Philips's I²C bus specification and Intel's SM bus specification provides master/slave transmission and reception, arbitration lost detection, slave address/general call address detection, generation and detection of start/stop conditions, and buss error detection.**

## ■ I²C Functions

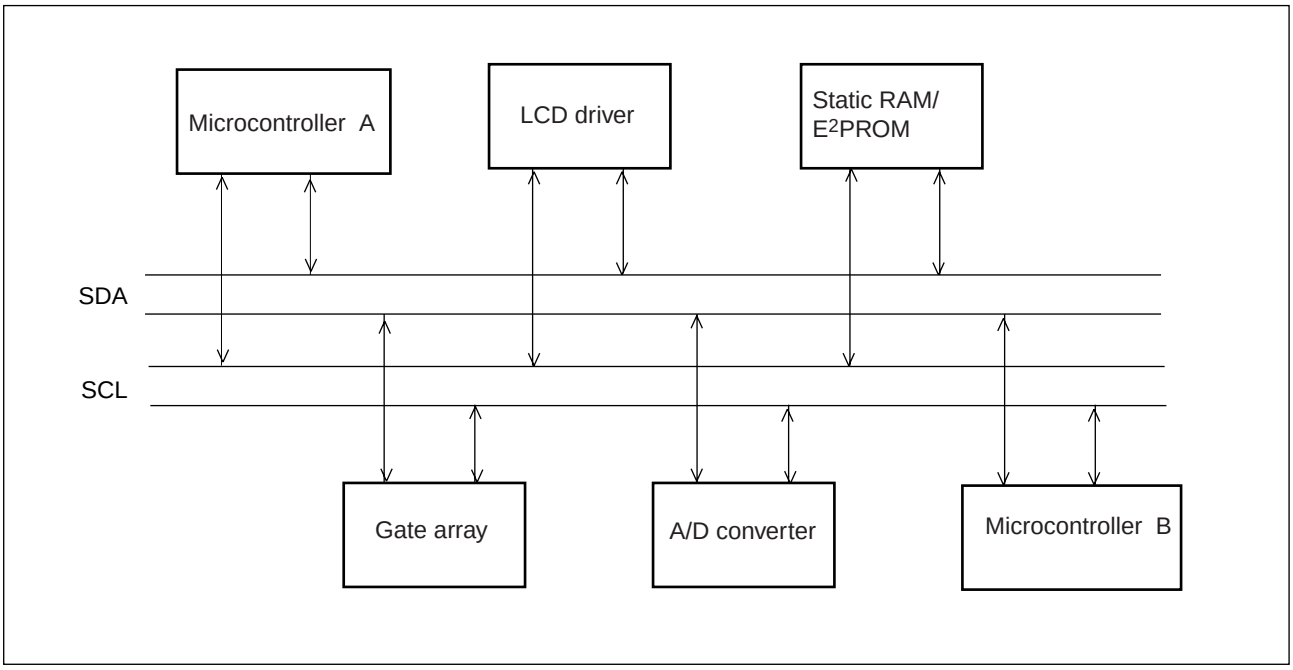
The I²C interface is a simple structure bidirectional bus consisting of two wires: a serial data line (SDA) and a serial clock line (SCL). Among the devices connected with these two wires, information is transmitted to one another. By recognizing the unique address of each device, it can operate as a transmitting or receiving device in accordance with the function of each device. Among these devices, the master/slave relation is established.

The I²C interface can connect two or more devices to the bus provided the upper limit of the bus capacitance does not exceed 400pF. It is a full-fledged multi-master bus equipped with collision detection and communication adjustment procedures designed to avoid the destruction of data if two or more masters attempt to start data transfer simultaneously.

A configuration example of the I²C interface is shown in Figure 15.1-1 "I²C Block Diagram".

The communication adjustment procedure permits only one master to control the bus when two or more masters attempt to control the bus so that messages are not lost or the contents of messages are not changed. Multi-master means that multiple masters attempt to control the bus simultaneously without losing messages.

**Figure 15.1-1 I²C Block Diagram**



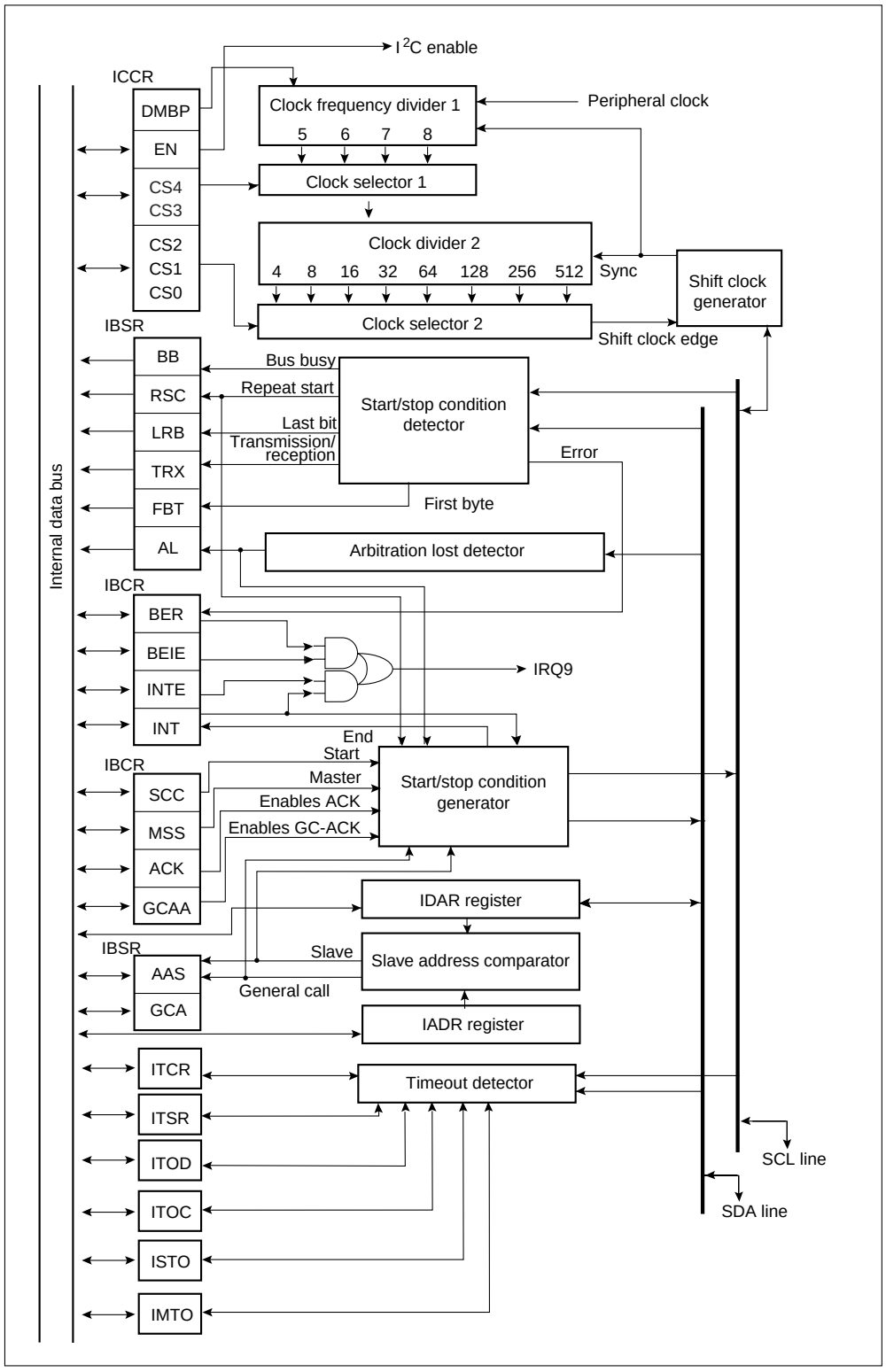
## 15.2 Configuration of the I²C

**The I²C consists of the following 14 blocks.**

- **Clock selector, clock divider, shift clock generator**
- **Start/stop condition generator**
- **Start/stop condition detector**
- **Arbitration lost detector**
- **Timeout detector**
- **Slave address comparator**
- **I²C bus status register (IBSR)**
- **I²C bus control register (IBCR)**
- **I²C clock control register (ICCR)**
- **I²C address register (IADR)**
- **I²C data register (IDAR)**
- **I²C timeout control register (ITCR)**
- **I²C timeout status register (ITSR)**
- **I²C timeout data register (ITOD)**
- **I²C timeout clock register (ITOC)**
- **I²C slave timeout register (ISTO)**
- **I²C master timeout register (IMTO)**

## ■ I²C Block Diagram

**Figure 15.2-1 I²C Block Diagram**



- **Clock selector, clock divider, shift clock generator**

  This circuit selects and generates a shift clock of the I²C bus based on the internal clock.

- **Start/stop condition generator**

  When the bus is released (when the SCL and SDA lines are at a "H" level), transmitting a start condition causes the master to start communication. When the SDA line is changed from "H" to "L" when SCL = H, a start condition is generated. When a stop condition is generated, the master can stop communication. The stop condition is generated when the SDA line is changed from "L" to "H" when SCL = H.

- **Start/stop condition detector**

  This circuit detects the start/stop condition for data transfer.

- **Arbitration lost detector**

  This interface circuit supports the multi-master system. If two or more masters transmit data simultaneously, arbitration lost is generated. When logic level "1" is transmitted when the SDA line is at level "L", this state is regarded as arbitration lost. At this time, IBSR:AL is set to "1" and the master is changed into a slave.

- **Slave address comparator**

  After a start condition is transmitted, a slave address is transmitted. This address is seven-bit data, followed by a data direction bit (R/W) as bit 8. ACK is returned only to the slave whose address matches the transmitted address.

- **Timeout detector**

  This circuit detects a timeout based on the value set in the ITOD, ITOC, ISTO, and IMTO registers.

- **IBSR register**

  The IBSR register indicates the status of the I²C interface. This register is read-only.

- **IBCR register**

  The IBCR register is used to select the operating mode, enables/disables interrupts, enables/disables acknowledge, and enables/disables general call acknowledge.

- **ICCR register**

  The ICCR register is used to permit the operation of the I²C interface and select the shift clock frequency.

- **IADR register**

  The IADR register is used to set the slave address.

- **IDAR register**

  The IDAR register is used to hold the shift data transmitted/received. In transmission, the data written in this register is transferred to the bus from the MSB in turn.

- **ITCR register**

  The ITCR register is used to enable/disable the operation of the timeout detector and to control interrupts.

- **ITSR register**

  The ITSR register is used to check the detection state of the timeout detector.

- **ITOD register**

  The ITOD register is used to set the count value for a I²C timeout in the data line.

- **ITOC register**

  The ITOC register is used to set the count value for a I²C timeout in the clock line.

- **ISTO register**

  The ISTO register is used to set the count value for a I²C slave timeout.

- **IMTO register**

  The IMTO register is used to set the count value for a I²C master timeout.

- **I²C interface interrupt source**

  IRQ9:

  An interrupt request is generated by the I²C interface when the bus error interrupt request bit is enabled (IBCR: BEIE = "1") and a bus error has occurred or when the transfer end interrupt enable bit is enabled (IBCR: INTE = "1") and data transfer is completed.

  IRQA:

  A timeout interrupt is generated if the set timeout is expired when the timeout detection function is enabled (ITCR: TS0 to TS2 are other than "000").

# 15.3 Pins of the I²C

**This section shows the pins related to the I²C and the block diagram of pins.**

## ■ Pins Related to the I²C

The pins related to the I²C include the clock I/O pin (P33/SCL2/UCK3) and the serial data I/O pin (P34/SDA2/UI3). They can be switched by the bridge circuit selection register (BRSR1 to 3), the operating port selection bit (SMC2: TXOE) of UART serial mode control register 2, and the I²C operation enable bit (ICCR: EN).

### P34/SDA2/UI3 pin

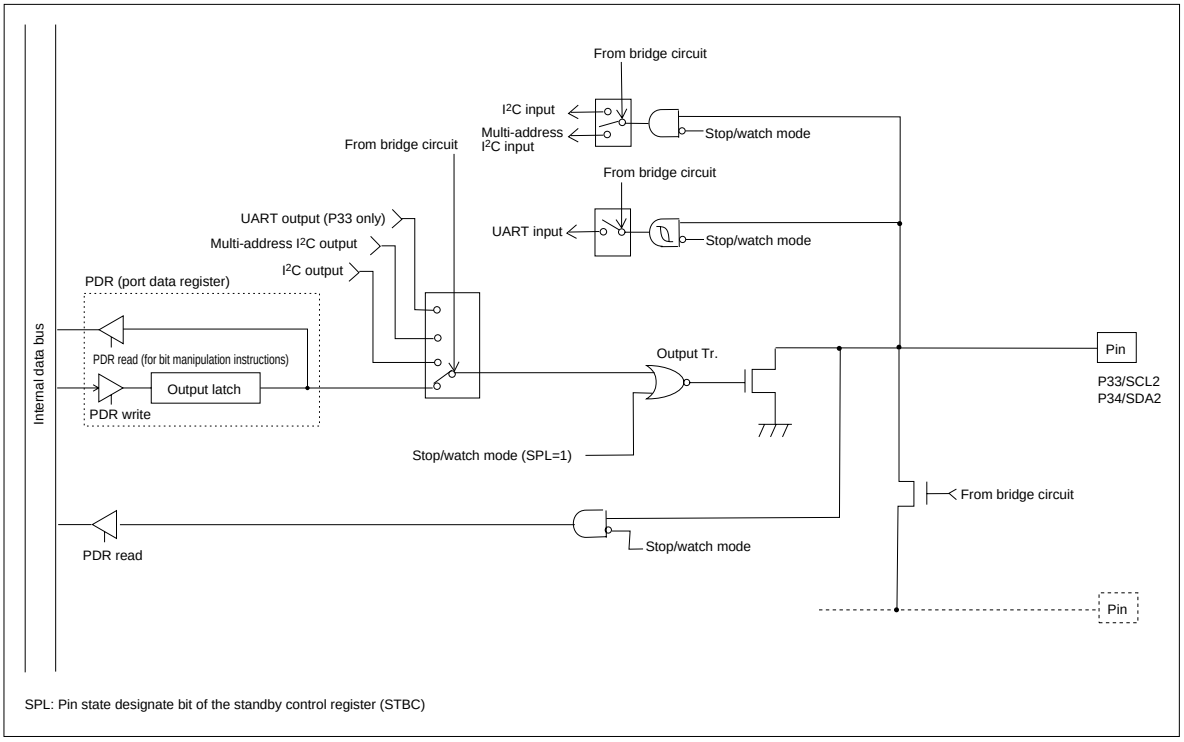
The P34/SDA2/U13 pin serves as an N-ch open-drain I/O port (P34), a I²C data I/O pin (SDA2), and a bridge circuit UART serial data input pin (UI3).

### P33/SCL2/UCK3 pin

The P33/SCL2/UCK3 pin serves as an N-ch open-drain I/O port (P33), a I²C shift clock I/O pin (SCL2), and a bridge circuit UART serial clock I/O pin (UCK3).

## ■ Block Diagram of Pins Related to I²C

**Figure 15.3-1 Block Diagram of Pins Related to I²C**




**Note:**

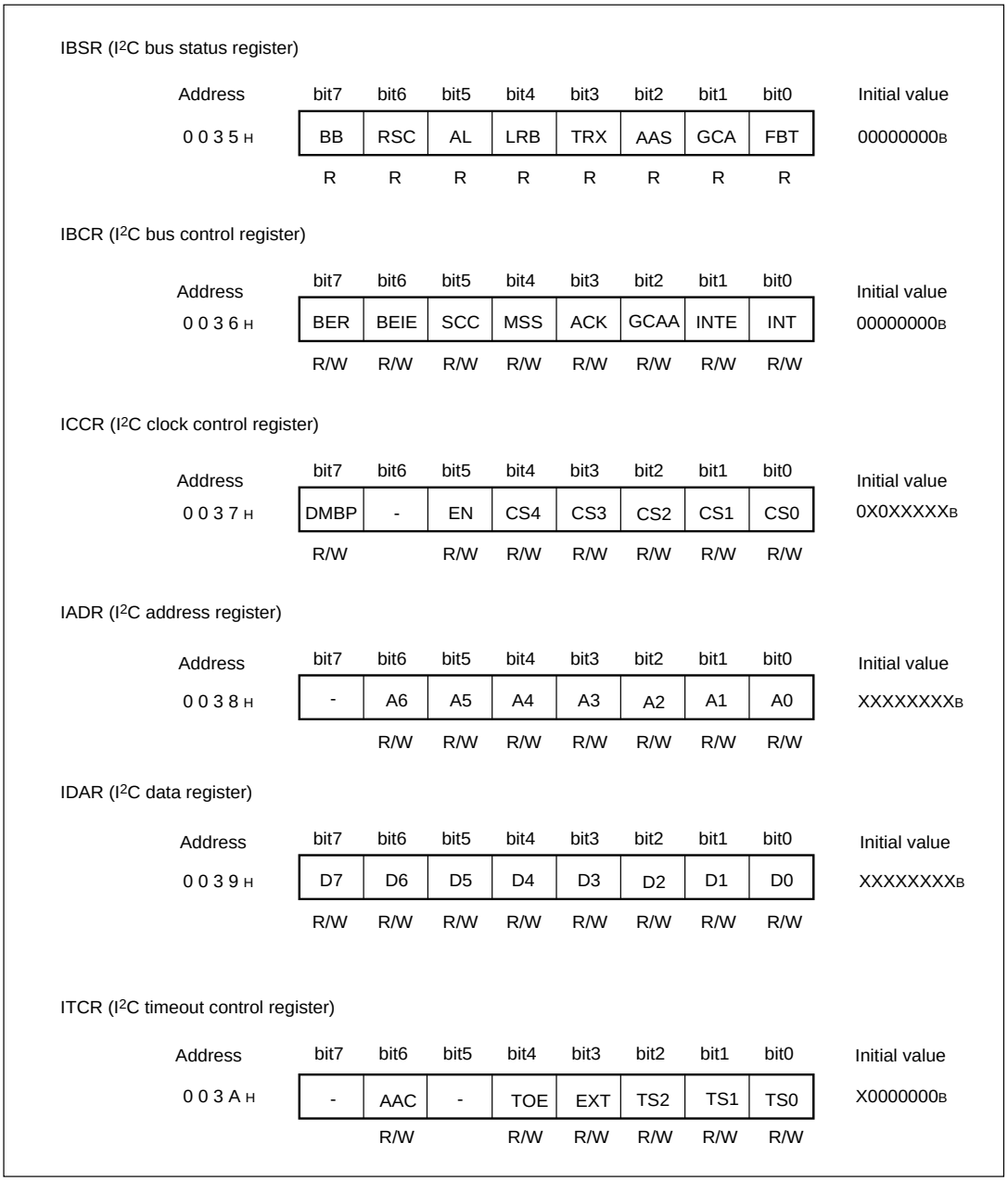
When the I²C function is used, P33/SCL2 and P34/SDA2 pins must be pulled up externally.

# 15.4 Registers of the I²C

## This section shows the registers related to the I²C.

### ■ Registers Related to I²C

**Figure 15.4-1 Registers Related to I²C**



ITSR (I²C timeout status register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 B H | - | - | - | - | TDR | TCR | MTR | STR | XXXX0000B |
| | | | | | R/W | R/W | R/W | R/W | |

ITOD (I²C timeout data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 C H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

ITOC (I²C timeout clock register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 D H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

IMTO (I²C master timeout register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 E H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

ISTO (I²C slave timeout register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 3 F H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W: Read/write enabled
R : Read only
X : Undefined

# 15.4.1 I²C Bus Status Register (IBSR)

**The IBSR register indicates the status of the interface.**

## ■ I²C Bus Status Register (IBSR)

**Figure 15.4-2 I²C Bus Status Register (IBSR)**

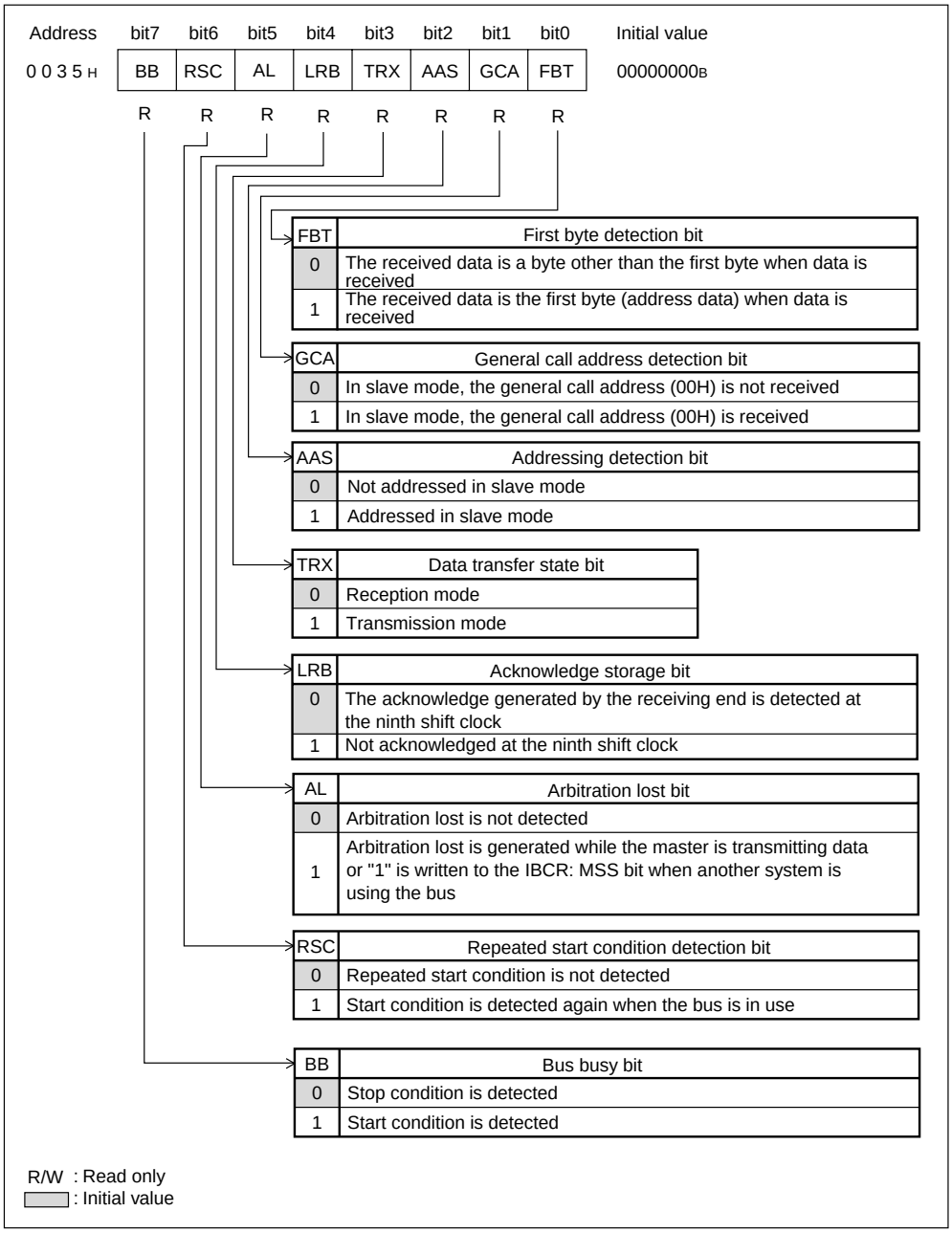


**Table 15.4-1 Functions of Each Bit in I²C Bus Status Register (IBSR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | BB:<br>Bus busy bit | This bit indicates the state of the bus. This bit is cleared when a stop condition is detected and set when a start condition is detected. |
| Bit 6 | RSC:<br>Repeated start condition detection bit | This bit detects the repeated start condition. This bit is set when a start condition is detected and cleared in the following state.<br>• "0" is written to the IBCR: INT bit<br>• The slave address does not match the set address<br>• A start condition is detected during bus stop<br>• A stop condition is detected |
| Bit 5 | AL:<br>Arbitration lost bit | This bit detects arbitration lost. This bit is set in the following states.<br>• Arbitration lost is detected when the master is transmitting data<br>• "1" is written to the IBCR: MSS bit when another system is using the bus<br>This bit is also cleared when "0" is written to the IBCR: INT bit |
| Bit 4 | LRB:<br>Acknowledge storage bit | This bit stores the SDA line value of the 9th clock when the data byte is transferred.<br>• Cleared when an acknowledge bit is detected. (SDA = L)<br>• Set when an acknowledge bit is not detected. (SDA = H)<br>• Cleared with "0" when a start or stop condition is detected. |
| Bit 3 | TRX:<br>Data transfer state bit | This bit indicates whether the data transfer is performed in the transmission mode or the reception mode. |
| Bit 2 | AAS:<br>Addressing detection bit | This bit indicates addressing is performed in slave mode.<br>This bit is set when addressing is performed in slave mode and cleared when a start or stop condition is detected. |
| Bit 1 | GCA:<br>General call address detection bit | This bit detects a general call address.<br>If this bit is set to "1" in slave mode, the general call address (00H) is received.<br>This bit is cleared when a start or stop condition is detected. |
| Bit 0 | FBT:<br>First byte detection bit | This bit detects the first byte<br>This bit is always set to "1" in the start condition.<br>This bit is set to "1" when a start condition is detected and cleared when "0" is written to the IBCR: INT bit or when the set address does not match its address in slave mode. |

# 15.4.2 I²C Bus Control Register (IBCR)

**The IBCR register is used to select the operating mode, enables/disables interrupts, enables/disables acknowledge, and enables/disables general call acknowledge.**

## ■ I²C Bus Control Register (IBCR)

**Figure 15.4-3 I²C Bus Control Register (IBCR)**

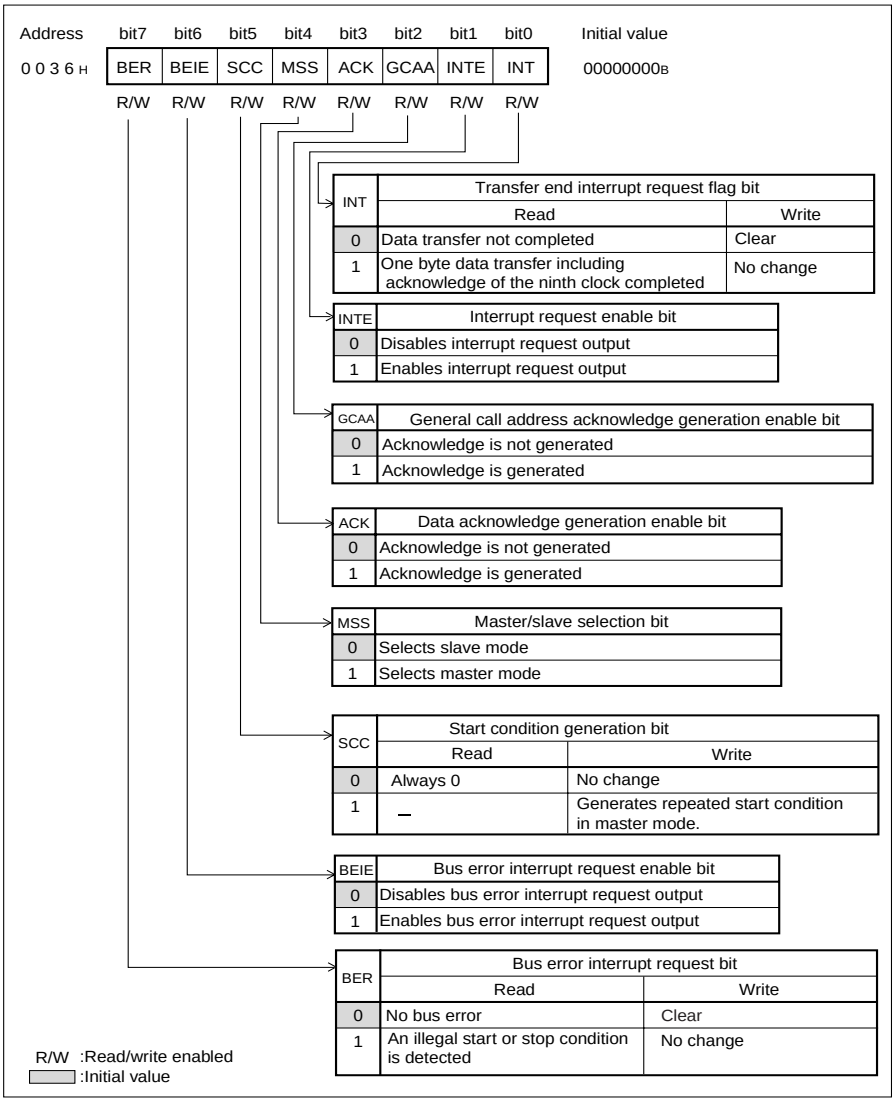


**Table 15.4-2 Functions of Each Bit in I²C Bus Control Register (IBCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | BER:<br>Bus error interrupt request flag bit | This bit clears a bus error interrupt and detects a bus error.<br>When a bus error is detected, "0" is written and the bus error interrupt is cleared.<br>When "1" is written, there is no change and no effect on others.<br>When an illegal start or stop condition is detected during data transfer, this bit is set to "1". For RMW instructions, "1" is always read.<br>When this bit is set, the operation enable bit in the ICCR register is cleared, the I²C enters the hold mode, and data transfer is terminated. |
| Bit 6 | BEIE:<br>Bus error interrupt request enable bit | This bit enables (BEIE = 1) or disables (BEIE = 0) the generation of a bus error interrupt request.<br>When this bit is set and BER = 1, an interrupt request is sent to the CPU. |
| Bit 5 | SCC:<br>Start condition generation bit | When this bit is set, a repeated start condition in master mode is generated. (SCC = 1)<br>No change when "0" is written.<br>The read value of this bit is always "0".<br>**Note:**<br>1) Do not write SCC = 1 and MSS = 0 simultaneously.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and a stop condition is generated. |
| Bit 4 | MSS:<br>Master/slave selection bit | This bit selects the slave mode (MSS = 0) or the master mode (MSS = 1).<br>When this bit is cleared to "0", a stop condition is generated and the master mode is switched to the slave mode after transfer is completed.<br>When this bit is set to "1", the slave mode is switched to the master mode, a start condition is generated, and transfer is started.<br>If arbitration lost is generated when the master is transmitting data, this bit is cleared and the master mode is switched to the slave mode.<br>**Note:**<br>1) Do not write SCC = 1 and MSS = 0 simultaneously.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and a stop condition is generated. |
| Bit 3 | ACK:<br>Data acknowledge generation enable bit | This bit enables (ACK = 1) or disables (ACK = 0) the output of the acknowledge bit in the 9th clock at data reception. |
| Bit 2 | GCAA:<br>General call address acknowledge generation enable bit | This bit permits the generation of acknowledge when a general call address is received.<br>When a general call address is received in slave mode when this bit is set to "1", output of acknowledge is permitted.<br>Even if a general call address is received when "0" is written to this bit, acknowledge is not output. |

**Table 15.4-2 Functions of Each Bit in I²C Bus Control Register (IBCR) (Continued)**

| Bit name | | Function |
|---|---|---|
| Bit 1 | INTE:<br>Transfer end interrupt request enable bit | This bit selects whether an interrupt at the end of transfer is enabled (INTE = 1) or disabled (INTE = 0).<br>When this bit is set and INT is set to "1", a transfer end interrupt request is sent to the CPU. |
| Bit 0 | INT:<br>Transfer end interrupt request flag bit | With this bit, the data transfer end interrupt request flag can be cleared. In addition, it can be determined whether the interrupt is detected.<br>When "0" is written, the transfer end interrupt request flag is cleared. When "1" is written, no change occurs.<br>If any of the following four conditions is met when one byte transfer including the acknowledge bit is completed (including the acknowledge bit in the 9th clock), this bit is set to "1".<br>• Bus master mode<br>• Addressed slave<br>• A general call address is received<br>• Arbitration lost is generated<br>When this bit is set to "1", the SCL line is kept at the "L" level. This bit is cleared when "0" is written to this bit. At this time, this macro releases the SCL line and transfers the next byte. This bit is also cleared to "0" when a start or stop condition is generated in master mode.<br>**Note:**<br>1) If "1" is written to SCC when INT = 0, "1" in the SCC bit has a higher priority and a start condition is generated.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and the stop condition is generated.<br>For RMW instructions, "1" is always read. |

**Note:**

When the interrupt request flag bit (IBCR: BER) is cleared, do not rewrite the interrupt request enable bit (IBCR: BEIE) simultaneously.

Only when the I²C enable bit (ICCR: EN) is set, values can be written to the ACK, GCAA, and INTE bits in the IBCR register.

# 15.4.3 I2C Clock Control Register (ICCR)

**The ICCR register is used to permit the operation of the I2C and select the shift clock frequency.**

## ■ I2C Clock Control Register (ICCR)

**Figure 15.4-4 I2C Clock Control Register (ICCR)**

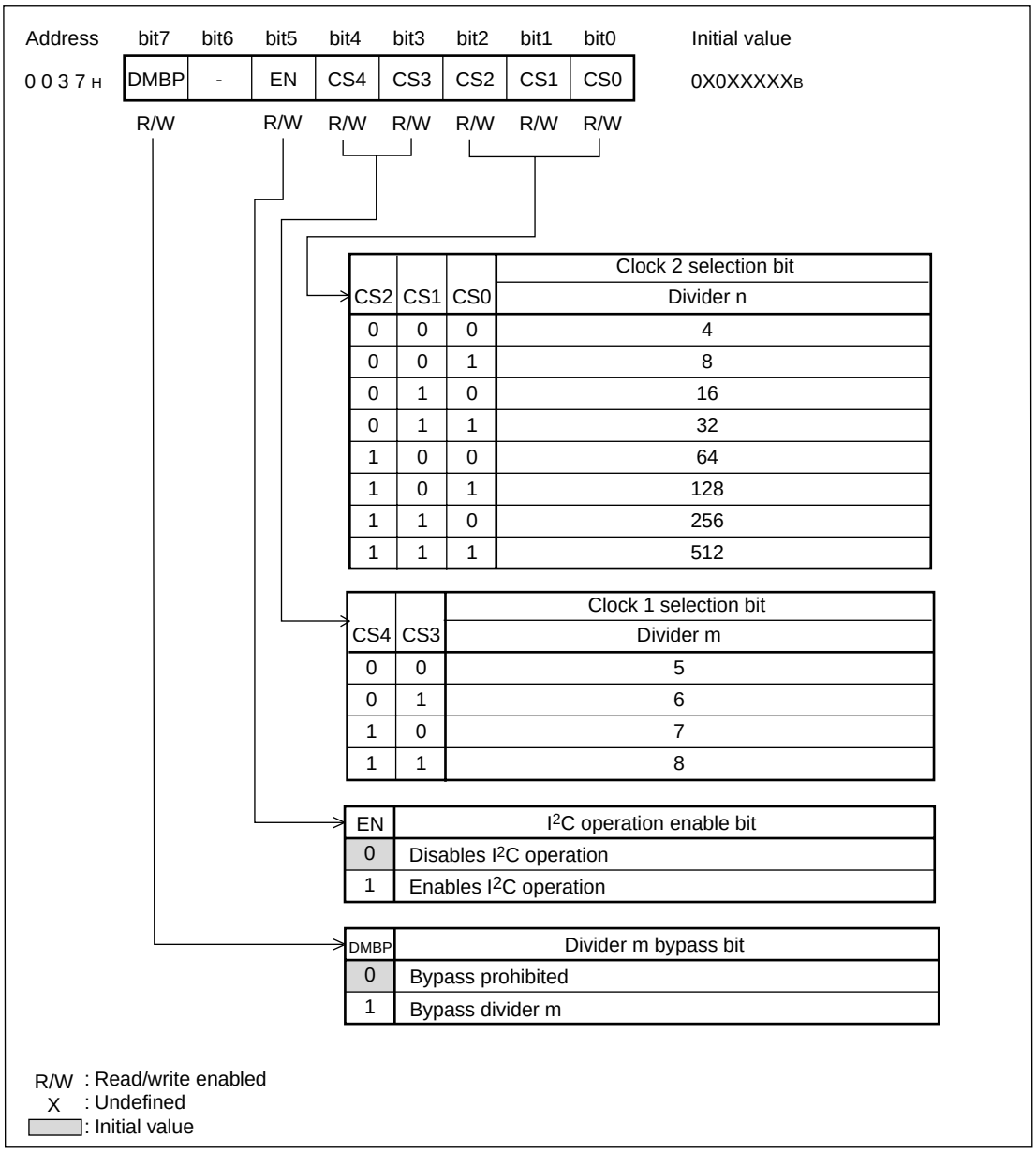


**Table 15.4-3 Functions of Each Bit in I²C Clock Control Register (ICCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | DMBP:<br>Divider m bypass bit | This bit is used to bypass the m divider for generating a shift clock frequency.<br>When "0" is written, the value set in CS3 and CS4 becomes the value of the m divider.<br>When "1" is written, the m divider is bypassed. This is equivalent to m = 1.<br>In read cycle, the present set value can be read.<br>When n = 0 (CS2 = CS1 = CS0 = 0), do not set this bit. |
| Bit 6 | Unused bit | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 5 | EN:<br>Multi-address I²C operation permission bit | This bit permits the operation of the multi-address I²C interface (EN = "1").<br>When the bit is "0", each bit of the MBSR and MBCR registers (excluding BER and BEIE bits) is cleared to "0".<br>When the MBCR:BER bit is set, the bit is cleared. |
| Bit 4<br>Bit 3 | CS4, CS3:<br>Clock 1 selection bit | This bit sets shift clock frequency.<br>Shift clock frequency Fsck is determined by the following formula. |
| Bit 2<br>Bit 1<br>Bit 0 | CS2, CS1, CS0:<br>Clock 2 selection bit | $Fsck = \frac{2/t_{inst}}{(m \times n + 2)}$ (1)<br>Where, $F_{inst}$ is an instruction cycle (the clock in the SYCC selected by the SCS bit). When DMBP is "0", m is selected by CS4 and CS3. When DMBP is "1," m is "1." n is selected by CS2, CS1, and CS0. |

# 15.4.4 I²C Address Register (IADR)

**The IADR register is used to set the slave address.**

## ■ I²C Address Register (IADR)

**Figure 15.4-5 I²C Address Register (IADR)**

IADR (I²C address register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0038H | - | A6 | A5 | A4 | A3 | A2 | A1 | A0 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

In slave mode, the value in this register is compared with the slave address stored in IADR after the requested address is received. When they match, acknowledge is transmitted to the master as the 9th shift clock.

# 15.4.5 I²C Data Register (IDAR)

**The IDAR register is used to set transmission data and to store received data.**

## ■ I²C Data Register (IDAR)

**Figure 15.4-6 I²C Data Register (IDAR)**

IDAR (I²C data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0039H | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

In master mode, the data written in the register is shifted to the SDA line bit by bit from the MSB bit.

The write side in this register is made up of a double buffer. When the bus is in use (IBSR: BB = 1), written data is loaded to the eight-bit shift register when the transfer of the present byte is completed. The data in the shift register is shifted and output to the SDA line bit by bit. The value written to this register has no effect on the present data transfer. Also in slave mode, the same function can be used after the address is determined.

When IBCR: INT = 1 at data reception (IBSR: TRX = 0), the received data can be read from this register. Since the register for serial transfer is read directly in read cycle, the received data is effective only when IBCR: INT = 1.

# 15.4.6 I2C Timeout Control Register (ITCR)

**The ITCR register is used to control the SM bus timeout detection.**

## ■ I2C Timeout Control Register (ITCR)

Figure 15.4-7 "I2C Timeout Control Register (ITCR)" shows the bit configuration of the I2C timeout control register (ITCR)

**Figure 15.4-7 I2C Timeout Control Register (ITCR)**

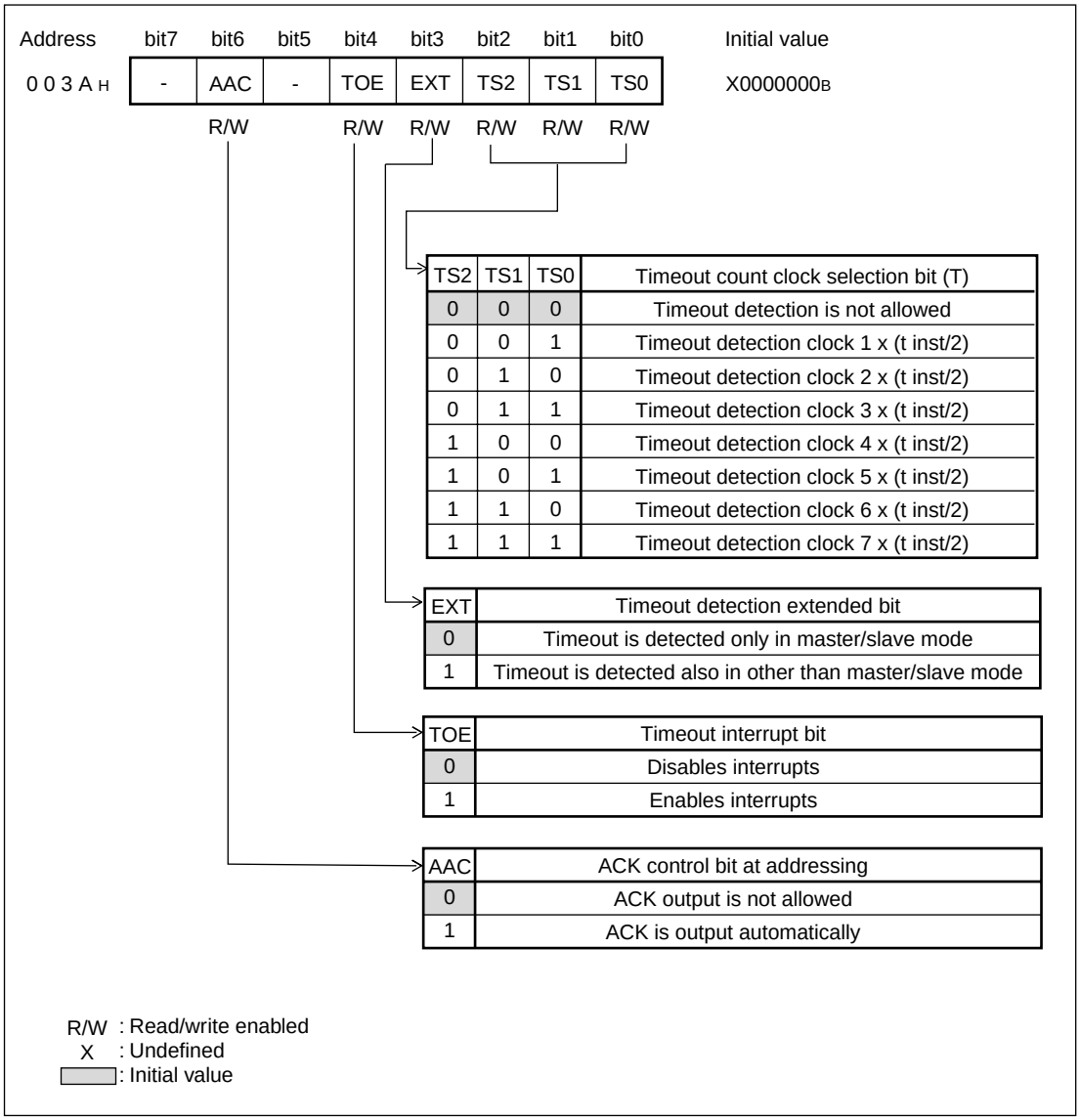


**Table 15.4-4 Functions of Each Bit in I²C Timeout Control Register (ITCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | Unused bit | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 6 | AAC:<br>ACK control bit at addressing | This bit controls ACK at address matching.<br>When this bit is set to "1", ACK control is performed automatically. (ACK is returned automatically when the addresses match.<br>When "0" is written to this bit, ACK at addressing is not output. |
| Bit 5 | Test bit | This bit is a test bit.<br>Write "1" when I²C is used.<br>**Note:**<br>Though the initial value of this bit is "0", write "1" to this bit when I²C is used. |
| Bit 4 | TOE:<br>Timeout interrupt enable bit | This bit enables/disables timeout interrupts.<br>When "1" is written to this bit, timeout interrupts are enabled and an interrupt request is sent to the CPU.<br>When "0" is written to this bit, timeout interrupts are disabled. |
| Bit 3 | EXT:<br>Timeout detection extended bit | This bit makes extended control for detecting a timeout.<br>When "1" is written to this bit, the timeout detection function also operates in a mode other than master/slave mode.<br>When "0" is written to this bit, the timeout detection function operates only in master/slave mode.<br>**Note:**<br>This bit is effective only when timeout detection is enabled (ITCR: TS0 to TS2 is other than "000".) |
| Bit 2<br>Bit 1<br>Bit 0 | TS2, TS1, TS0:<br>Timeout count selection bits (T) | These bits select the clock for detecting a timeout.<br>When TS2 = 0, TS1 = 0, and TS0 = 0, the timeout detection function is disabled.<br>With the clock selected in these bits [T x ($t_{inst}$/2) x 10], the low period in the SCL2 and SDA2 lines is counted. |

# 15.4.7 I²C Timeout Status Register (ITSR)

**The ITSR register indicates the SM bus timeout detection status.**

## ■ I²C Timeout Status Register (ITSR)

15.4-8 "I²C Timeout Status Register (ITSR)" shows the bit configuration of the I²C timeout status register (ITSR).

**Figure 15.4-8 I²C Timeout Status Register (ITSR)**

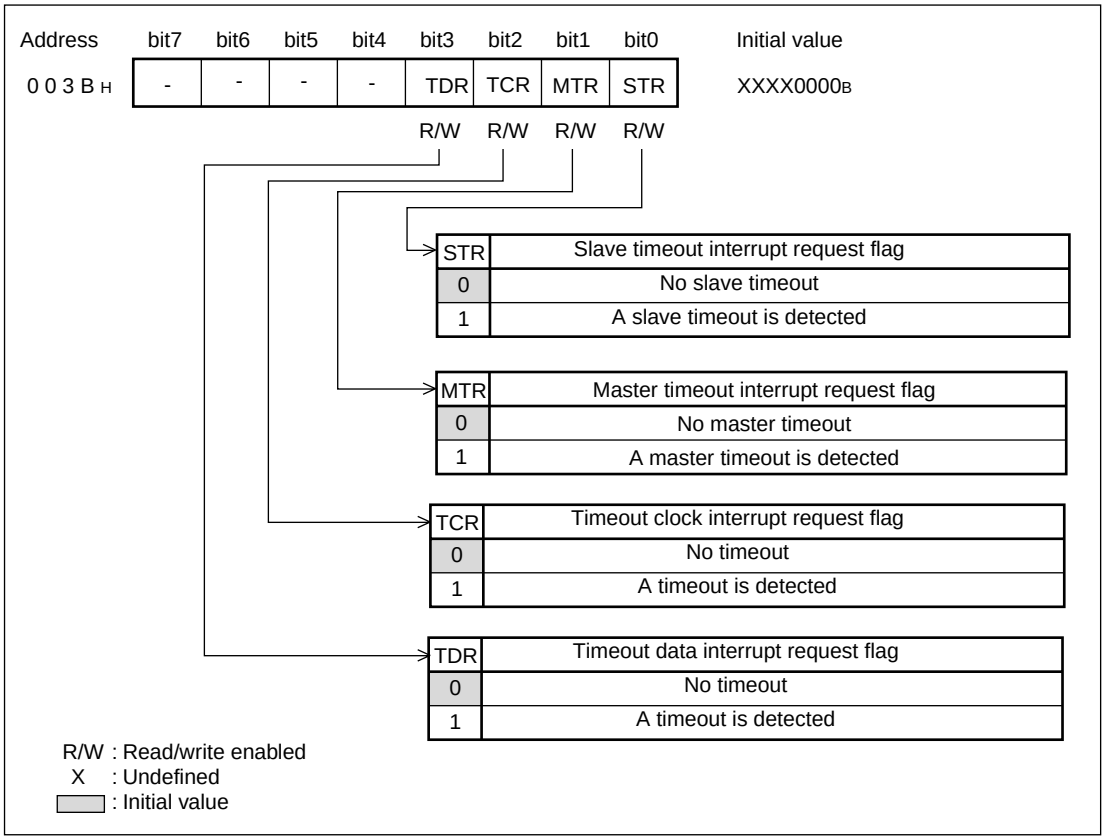


**Table 15.4-5 Functions of Each Bit in I²C Timeout Status Register (ITSR)**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4 | Unused bits | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 3 | TDR:<br>Timeout data interrupt request flag | This bit detects a timeout of the data line.<br>When timeout data is detected, this bit is set to "1".<br>If timeout interrupts (ITCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |
| Bit 2 | TCR:<br>Timeout clock interrupt request flag | This bit detects a timeout of the clock line.<br>When a timeout clock is detected, this bit is set to "1".<br>If timeout interrupts (ITCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |
| Bit 1 | MTR:<br>Master timeout interrupt request flag | This bit detects a master timeout.<br>When a master timeout is detected, this bit is set to "1".<br>If master timeout interrupts (ITCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no meaning. |
| Bit 0 | STR:<br>Slave timeout interrupt request flag | This bit detects a slave timeout.<br>When a slave timeout is detected, this bit is set to "1".<br>If slave timeout interrupts (ITCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |

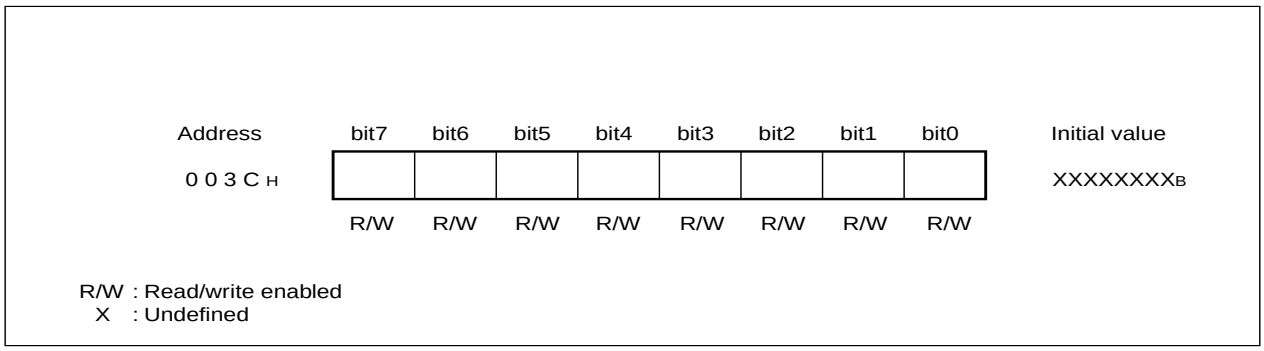
# 15.4.8 I2C Timeout Data Register (ITOD)

**The ITOD register is used to detect an SM bus timeout (data line).**

## ■ I2C Timeout Data Register (ITOD)

Figure 15.4-9 "I2C Timeout Data Register (ITOD)" shows the bit configuration of the I2C timeout data register (ITOD)

**Figure 15.4-9 I2C Timeout Data Register (ITOD)**




If the value written to this register plus one matches the counted value of the low time period of the SDA2 line when the timeout detection function is enabled (ITCR: TS0 to TS2 is other than "000"), the timeout data interrupt request flag (ITSR: TDR) is set to "1".

The low time period of the SDA2 line is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. When the SDA2 line is at a high level, the counter value is cleared to "0", and counting starts again when the SDA2 line is at a low level. When the counter value matches "the value set in the timeout data register (ITOD) + 1", a timeout is detected and the timeout data interrupt request flag is set.

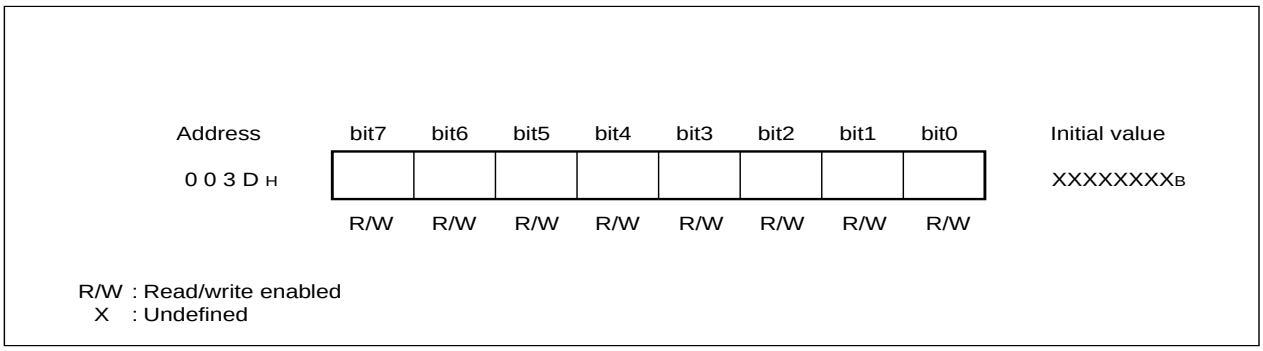
# 15.4.9 I²C Timeout Clock Register (ITOC)

**The ITOC register is used to detect an SM bus timeout (clock line).**

## ■ I²C Timeout Clock Register (ITOC)

Figure 15.4-10 "I²C Timeout Data Register (ITOD)" shows the bit configuration of the I²C timeout clock register (ITOC)

**Figure 15.4-10 I²C Timeout Data Register (ITOD)**




If the value written to this register plus one matches the counted value of the low time period of the SCL2 line when the timeout detection function is enabled (ITCR: TS0 to TS2 is other than "000"), the timeout clock interrupt request flag (ITSR: TCR) is set to "1".

The low time period of the SCL2 line is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. When the SCL2 line is at a high level, the counter value is cleared to "0" and counting starts again when the SCL2 line is at a low level. When the counter value matches "the value set in the timeout clock register (ITOC) + 1", a timeout is detected and the timeout clock interrupt request flag is set.

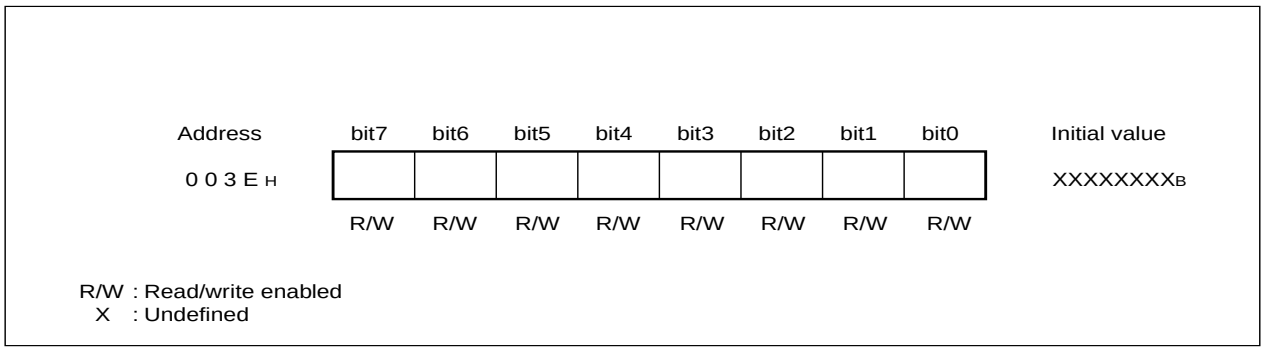
# 15.4.10 I2C Master Timeout Register (IMTO)

**The IMTO register is used to detect an SM bus timeout.**

## ■ I2C Master Timeout Register (IMTO)

Figure 15.4-11 "I2C Master Timeout Register (IMTO)" shows the bit configuration of the I2C master timeout register (IMTO)

**Figure 15.4-11 I2C Master Timeout Register (IMTO)**




When the value written to this register plus one matches the cumulative value of the low time period of the SCL2 line counted from start to acknowledge (or from acknowledge to acknowledge or from acknowledge to stop) when the timeout detection function is enabled (ITCR: TS0 to TS2 is other than "000"), the master timeout interrupt request flag (ITSR: MTR) is set to "1".

The low time period of the SCL2 line is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 10. In master mode, the low time period cumulative value of the SCL2 line is counted only when the SCL2 line is at a low level and counting is stopped when the SCL2 line is at a high level. At start, acknowledge or stop, or by exiting the master mode, the counter value is cleared to "0". When the counter value matches the value set in "the master timeout register (IMTO) + 1", a master timeout is detected and the master timeout interrupt request flag is set.

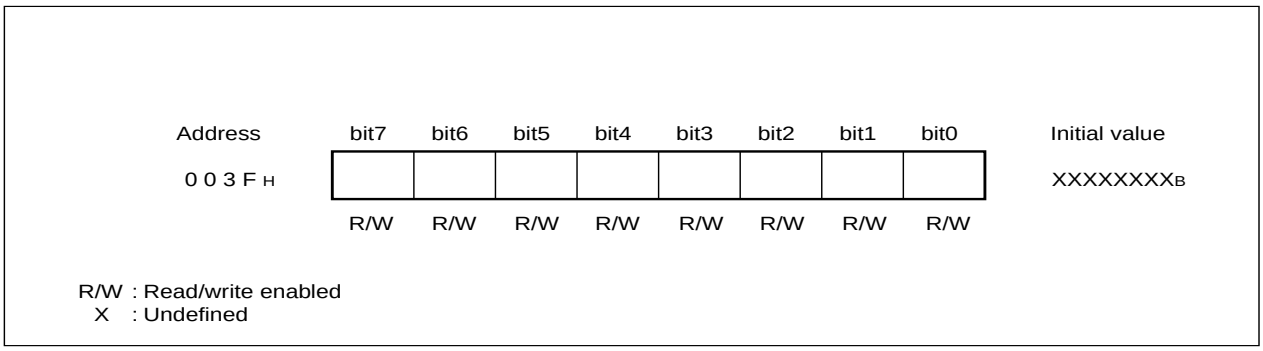
# 15.4.11 I2C Slave Timeout Register (ISTO)

**The ISTO register is used to detect an SM bus timeout.**

## ■ I2C Slave Timeout Register (ISTO)

Figure 15.4-12 "I2C Slave Timeout Register (ISTO)" shows the bit configuration of the I2C slave timeout register (ISTO)

**Figure 15.4-12 I2C Slave Timeout Register (ISTO)**




When the value written to this register plus one matches the cumulative value of the low time period of the SCL2 line counted from start to stop when the timeout detection function is enabled (ITCR: TS0 to TS2 is other than "000"), the slave timeout interrupt request flag (ITSR: STR) is set to "1".

The low-time period of the SCL2 line is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. In slave mode, the low-time period cumulative value of the SCL2 line is counted only when the SCL2 line is at a low level and counting is stopped when the SCL2 line is at a high level. At start or stop or by exiting the slave mode, the counter value is cleared to "0". When the counter value matches "the value set in the slave timeout register (ISTO) + 1", a slave timeout is detected and the slave timeout interrupt request flag is set.

# 15.5 I²C Interrupts

**The I²C interface may generate an interrupt request when the data transfer is completed, a bus error has occurred, or a timeout is detected.**

## ■ Interrupt at Bus Error

When the following conditions are met, a bus error is assumed to have occurred and the I²C interface is stopped.

1. When a stop condition is detected in master mode.
2. When a start or stop condition is detected when the first byte is being transmitted and received.
3. When a start or stop condition is detected when data (excluding the first bit of start, stop, and data) is being transmitted and received.

If the bus error interrupt request enable bit is enabled (IBCR: BEIE = 1) at this time, an interrupt request is output to the CPU. Clear the interrupt request by writing "0" to the BER bit in the interrupt processing routine.

If a bus error has occurred in spite of the BEIE bit value, the BER bit is set to "1".

## ■ Interrupt at Data Transfer Completion

When data transfer is completed and the transfer end interrupt request enable bit is enabled (IBCR: INTE = 1), an interrupt request (IRQ9) is output to the CPU. Clear the interrupt request by writing "0" to the INT bit in the interrupt processing routine.

If data transfer is completed in spite of the INTE bit value, the INT bit is set to "1".

## ■ Interrupt at Timeout Detection

If the specified timeout time has expired when the timeout detection function is enabled (ITCR: TS0 to TS2 is other than "000"), a timeout interrupt is generated (IRQA). The timeout can be checked with each interrupt request flag of the I²C bus status register (ITSR). When the timeout detection extended bit (ITOR: EXT) is set, the bus is also monitored in a mode other than master/slave mode.

## ■ Register and Vector Table Address Related to Interrupt of I²C

**Table 15.5-1 Register and Vector Table Address Related to Interrupt of I²C**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQ9 | ILR3 (007DH) | L91 (bit 3) | L90 (bit 2) | FFE8H | FFE9H |
| IRQA | ILR3 (007DH) | LA1 (bit 5) | LA0 (bit 4) | FFE6H | FFE7H |

For interrupt operation, see Section 3.4.2 "Interrupt Processing.

# 15.6 Operation of the I²C

**The I²C interface is a serial data base of 8-bit data synchronized with the shift clock.**

## ■ I²C System

### ❍ Operation mode

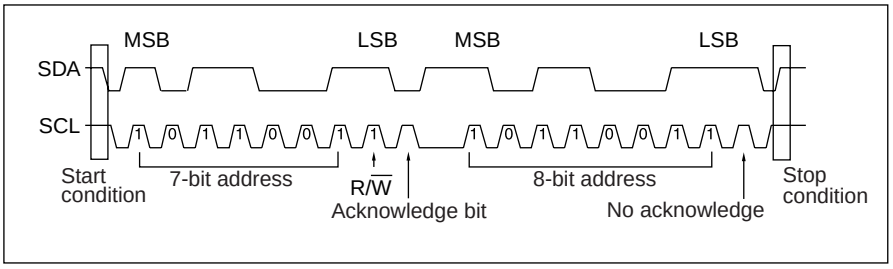
The I²C bus system uses a serial data line (SDA) and a serial clock line (SCL) to transfer data. All connected devices require an open-drain or open collector output. The logic function is used by connecting a pull-up resistor.

Each device connected to the bus has a unique address and can be set by software. Among the devices, simple master/slave relations are established and master devices function as master transmitters or master receivers. The I²C interface is a full-fledged multi-master bus equipped with collision detection and arbitration functions so that data destruction can be prevented even if two or more masters attempt to start data transfer simultaneously.

## ■ I²C Protocol

Figure 15.6-1 "Data Transfer Example" shows the format required for data transfer.

**Figure 15.6-1 Data Transfer Example**




After a start condition (S) is generated, a slave address is transmitted. This address is a seven-bit address followed by a data direction bit (R/W) as bit 8. Data transfer is always ended with the master stop condition (P). It is also possible to address to another slave without generating a stop condition by generating a repeated start condition (Sr).

### ■ Start Condition

When the master is not connected to the bus (the logic of SCL and SDA is "H") in states where the bus is released, the master generates a start condition. As indicated in Figure 15.6-1 "Data Transfer Example", a start condition is generated when the SDA line is changed from "H" to "L" in states where SCL is at the "H" level. At this time, new data transfer starts and master/slave operation starts. The two methods for generating a start condition are shown as follows.

- Writing "1" to the IBCR: MSS bit in states where the $I^2C$ bus is not used (IBCR: MSS = 0, IBSR: BB = 0, IBCR: INT = 0, IBSR: AL = 0). Thereafter, IBSR: BB is set to "1" to indicate bus busy.
- Writing "1" to the IBCR: SCC bit in interrupt states in bus master mode (IBCR: MSS = 1, IBSR: BB = 1, IBCR: INT = 1, IBSR: AL = 0) and generates a repeated start condition.

Even if "1" is written to the IBCR: MSS bit or "1" is written to the IBCR: SCC bit under conditions other than the above conditions, it is ignored. If "1" is written to the IBCR: MSS bit when another system is using the bus (in idle state), the IBSR: AL bit is set to "1".

### ■ Addressing

In master mode, the BB and TRX bits in the IBSR register are set to 1 after a start condition is generated and the contents of the IDAR register in the slave address are output from the MSB in turn. This address data consists of 8-bits with a seven-bit slave address, followed by a R/W bit indicating the data transfer direction (Bit 0 in IDAR).

After the address data is transmitted, the master receives acknowledge from the slave. The SDA line is set to "L" by the 9th clock and the master receives the acknowledge bit from the receiving end (see Figure 15.6-1 "Data Transfer Example"). At this time, the R/W bit (IDAR: bit 0) is reversed and stored in the IBSR: TRX bit.

In slave mode, the BB and TRX bits in the IBSR register are set to "1" and "0", respectively, after a start condition is detected and data from the master is received by the IDAR register. After receiving the address data, the IDAR and IADR registers are compared. If the values match, IBSR: AAS is set to "1" and acknowledge is transmitted to the master. Thereafter, bit 0 of the received data (bit 0 in the IDAR register) is stored in the IBSR: TRX bit.

### ■ Data Transfer

After addressing of the slave is achieved, data can be transmitted and received in byte units in the direction determined by the R/W bit sent by the master.

Each byte output to the SDA line is fixed to 8-bits. As shown in Figure 15.6-1 "Data Transfer Example", the receiving device transmits acknowledge to the transmitting device by stabilizing the SDA line to the "L" level when the acknowledge clock pulse is "H". With the MSB at the head, each bit of data is transmitted in one clock pulse. Each time a byte is transferred, acknowledge must be transmitted and received. Therefore, 9 clock pulses are required to transfer one complete data byte.

### ■ Acknowledge

Acknowledge is transmitted from the receiving end for the 9th clock of data byte transfer from the transmitting end.

When data is received, the acknowledge bit can be enabled (IBCR: ACK = 1) or disabled (IBCR: ACK = 0) with the IBCR: ACK bit.

When transmitting data, acknowledge from the receiving end is stored in the IBSR: LRB bit.

## ■ Stop Condition

By generating a stop condition, the master can release the bus to terminate communication. A stop condition can be generated by changing the SDA line from "L" to "H" when the SCL line is at the "H" level. It is a signal to notify the bus connection device of the end of communication (bus free) in master mode. The master can generate start conditions continuously without generating a stop condition. This is called the repeated start condition.

In bus master mode, a stop condition is generated by writing "0" to the IBCR: MSS bit in the interrupt state (IBCR: MSS = 1, IBSR: BB = 1, IBCR: INT = 1, IBSR: AL = 0) and the master mode is switched to the slave mode..

Even if "0" is written to the IBCR: MSS bit in other the above, it is ignored.

## ■ Arbitration

This interface circuit is a full-fledged multi-master bus that can connect two or more masters. If a master transfers data and another master transfers data simultaneously, an arbitration is generated.

An arbitration occurs in the SDA line when the SCL line is at the "H" level. The master recognizes the occurrence of an arbitration lost when its transmission data is "1" and data on the SDA line is at the "L" level, and then it sets data output to off and sets the IBSR: AL bit to "1". When the IBSR: AL is set to "1", "0" is written to IBCR: MSS and IBSR: TRX. As a result, the TRX is cleared and the master mode is switched to the slave reception mode.

# 15.7 Notes on Using the I²C

**This section describes precautions to take when using the I²C interface.**

## ■ Precaution in Setting the I²C Interface Register

Before writing to the bus control register (IBCR), the I²C interface must be enabled (ICCR: EN).

When the master slave selection bit (IBCR: MSS) is set, transfer starts.

## ■ Precaution in Setting the Shift Clock Frequency

To calculate the shift clock frequency using the $F_{sck}$ expression (1) in Table 15.4-3 "Functions of Each Bit in I²C Clock Control Register (ICCR)", it is necessary to know the values of m, n, and DMBP.

When n is 4 (ICCR: CS2 = CS1 = CS0 = 0), "DMBP = 1" cannot be selected. Other combinations do not present a problem.

## ■ Precaution on the Priority at Simultaneous Writing

- Contention of the next byte transfer and stop condition

  When "0" is written to IBCR: MSS in states where IBCR: INT is cleared, the MSS bit has a higher priority and a stop condition is generated.

- Contention of the next byte transfer and start condition

  When "1" is written to IBCR: SCC in states where IBCR: INT is cleared, the SCC bit has a higher priority and a start condition is generated.

## ■ Precaution on Setting with Software

Do not select the repeated start condition (IBCR: EN = 0) and the slave mode (IBCR: MSS = 0) at the same time.

In states where the interrupt request flag bits (BER and INT in the IBCR register) are set to "1" and the interrupt request enable bits are enabled (BEIE and INTE in the IBCR register are set to "1"), recovery from the interrupt processing cannot be performed. Clear the BER and INT bits in the IBCR register.

When the I²C operation is not permitted (ICCR: EN = 0), all bits of the bus status register IBSR and the bus control register IBCR (excluding the bus error BER bit and the bus error enable BEIE bit) are cleared.

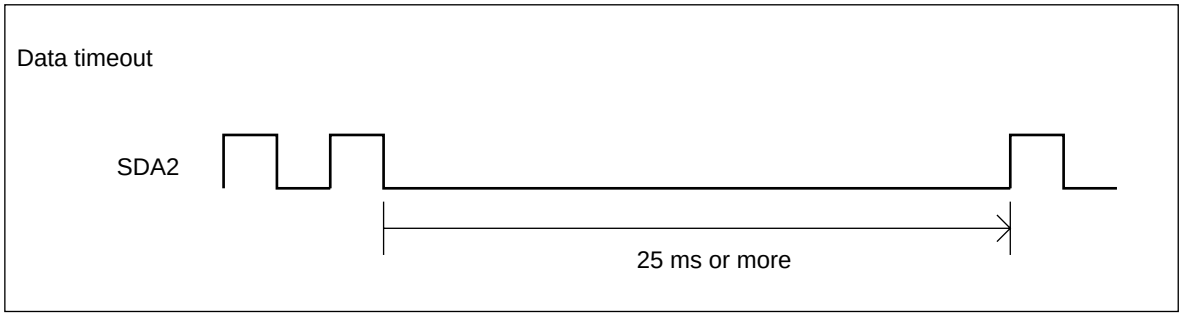
# 15.8 Operation of the Timeout Detection Function

**This section describes the operation of the timeout detection function when it is used as the SM bus.**

## ■ Data Timeout

When the "L" period of the SDA2 line exceeds 25 ms, this state is defined as a data timeout. The low time period is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the timeout data register (ITOD) + 1", a timeout is detected and the timeout data interrupt request flag (ITSR: TDR) is set.

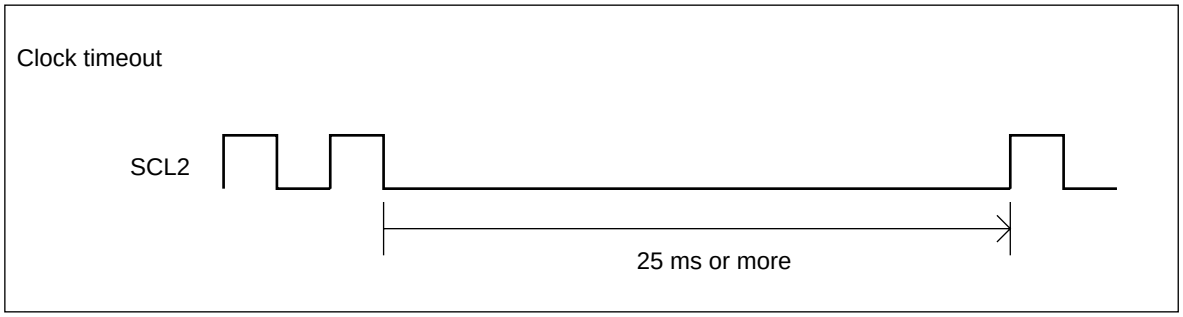
**Figure 15.8-1 Data Timeout**




## ■ Clock Timeout

When the "L" period of the SCL2 line exceeds 25 ms, this state is defined as a clock timeout. The low-time period is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the timeout clock register (ITOC) + 1", a timeout is detected and the timeout clock interrupt request flag (ITSR: TCR) is set.

**Figure 15.8-2 Clock Timeout**

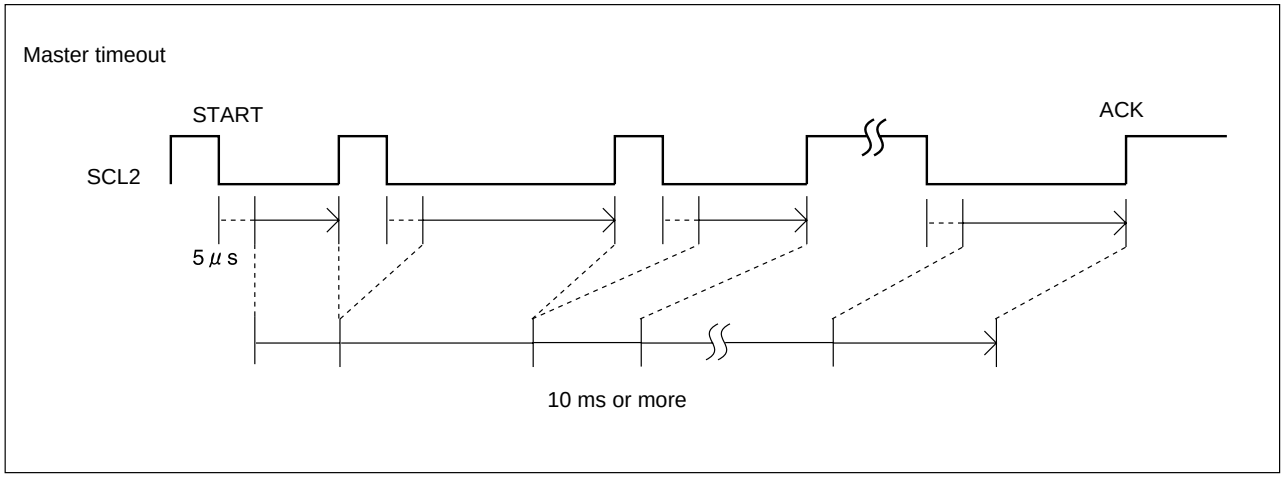


## ■ Master Timeout

When the cumulative "L" period of the SCL2 line between one byte data (START to ACK, ACK to ACK, ACK to STOP) exceeds 10 ms in master mode, this state is defined as a master timeout.

The low-time period is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 10. When the counter value matches "the value set in the master timeout register (IMTO) + 1", a master timeout is detected and the master timeout interrupt request flag (ITSR: MTR) is set.

**Figure 15.8-3 Master Timeout**

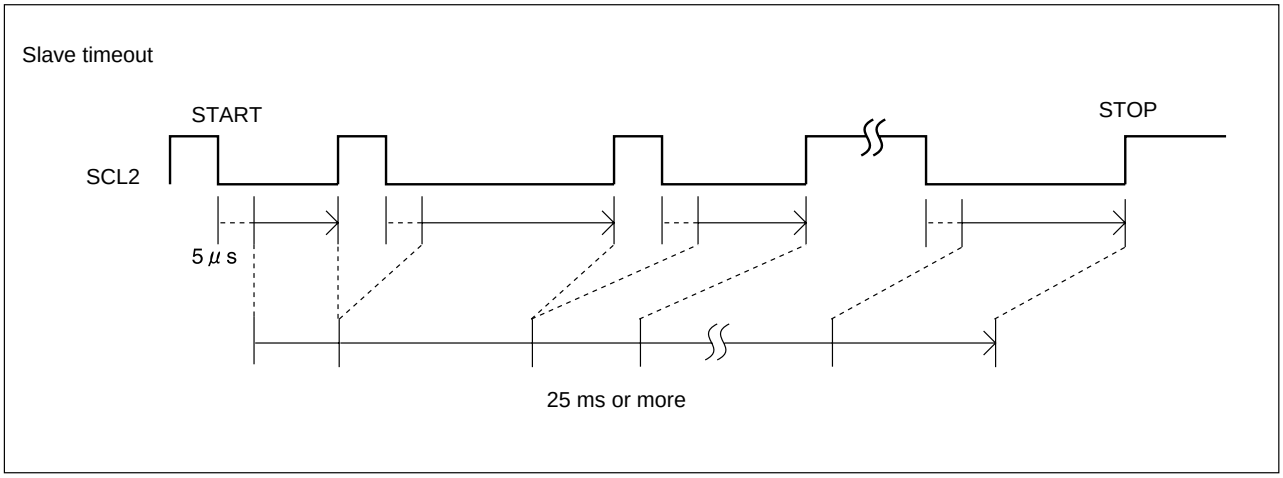


## ■ Slave Timeout

When the cumulative "L" period in the SCL2 line between START and STOP exceeds 10 ms in slave mode, this state is defined as a slave timeout.

The low-time period is counted with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the slave timeout register (ISTO) + 1", a slave timeout is detected and the slave timeout interrupt request flag (ITSR: STR) is set.

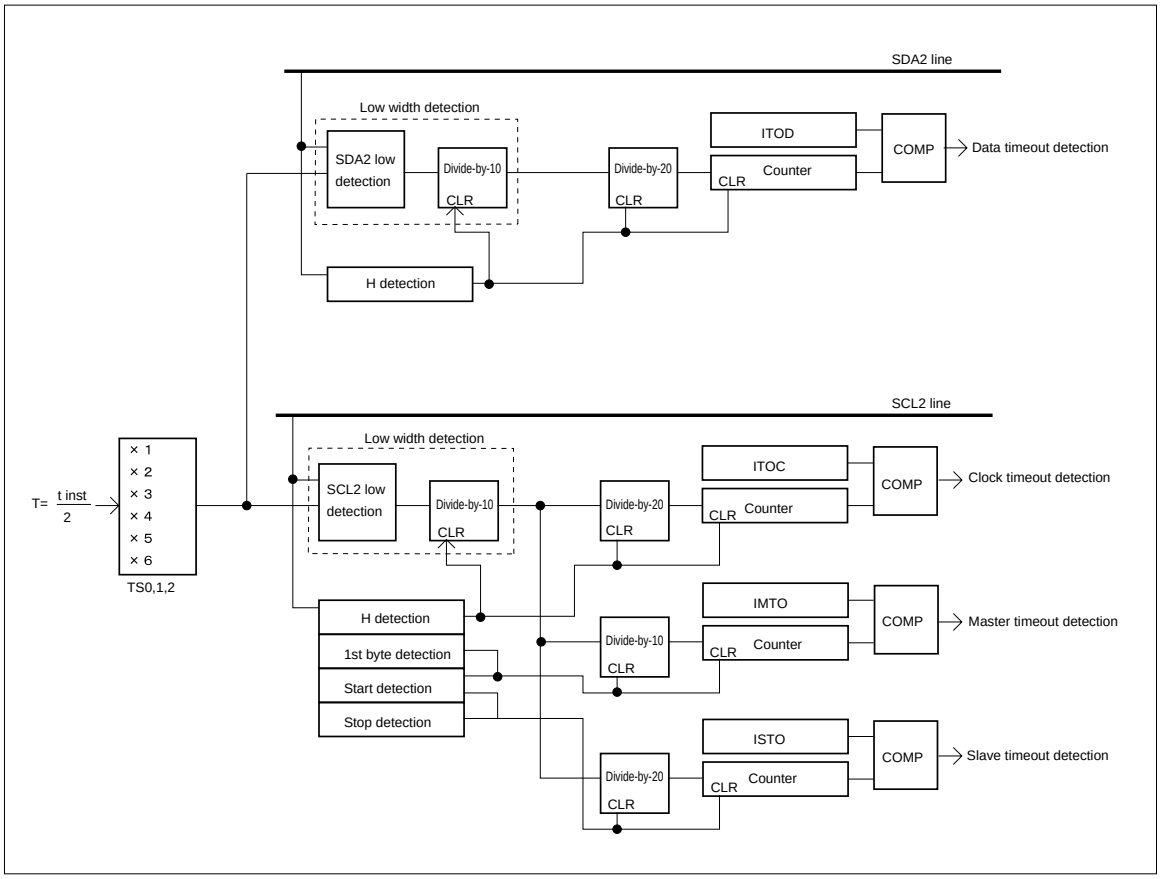
**Figure 15.8-4 Slave Timeout**



## ■ Timeout Clock Supply Block

Figure 15.8-5 "Timeout Clock Supply Block" shows the clock supply block of timeout.

**Figure 15.8-5 Timeout Clock Supply Block**




## ■ Errors

Timeout detection is performed with the clock selected with the timeout count clock detection bits (ITCR: TS0 to TS2) (T) divided by 10. Therefore, the following errors occur in the detection of an "L" width when, for example, T = 0.5 µs.

- Detects an "L" width of 5 to 5.5 µs or more in duration. (The sampling cycle is 0.5 µs)
- When the count is stopped (when low width detection is completed), an error of up to -5.5 µs in duration occurs. (In a cumulative count, errors are also accumulated.)

The counter for detecting a timeout is incremented with an L width or cumulative L width of 100 µs or more (50 µs or more for a master timeout).

# CHAPTER 16 MULTI-ADDRESS I$^2$C

**This chapter describes the functions and operations of the multi-address I$^2$C.**


16.1 "Overview of the Multi-address I$^2$C"

16.2 "Configuration of the Multi-address I$^2$C"

16.3 "Pins of the Multi-address I$^2$C"

16.4 "Registers of the Multi-address I$^2$C"

16.5 "Multi-address I$^2$C Interrupts"

16.6 "Operation of the Multi-address I$^2$C"

16.7 "Notes on Using the Multi-address I$^2$C"

16.8 "Operation of the Timeout Detection Function"

# 16.1 Overview of the Multi-address I2C

**The Multi-address I2C is a simple bidirectional bus consisting of two wires that transfer data among devices. These two Multi-address I2C bus interfaces allow internal devices requiring address data to connect to one another with a minimum number of circuits, making it possible to construct less expensive hardware using a fewer number of PCBs.**

**The Multi-address I2C interface that supports Philips's Multi-address I2C bus specification and Intel's SM bus specification provides master/slave transmission and reception, arbitration lost detection, slave address/general call address detection, generation and detection of start/stop conditions, and buss error detection.**

## ■ Multi-address I2C Functions

The multi-address I2C interface is a simple structure bidirectional bus consisting of two wires: a serial data line (SDA) and a serial clock line (SCL). Among the devices connected with these two wires, information is transmitted to one another. By recognizing the unique address of each device, it can operate as a transmitting or receiving device in accordance with the function of each device. Among these devices, the master/slave relation is established.

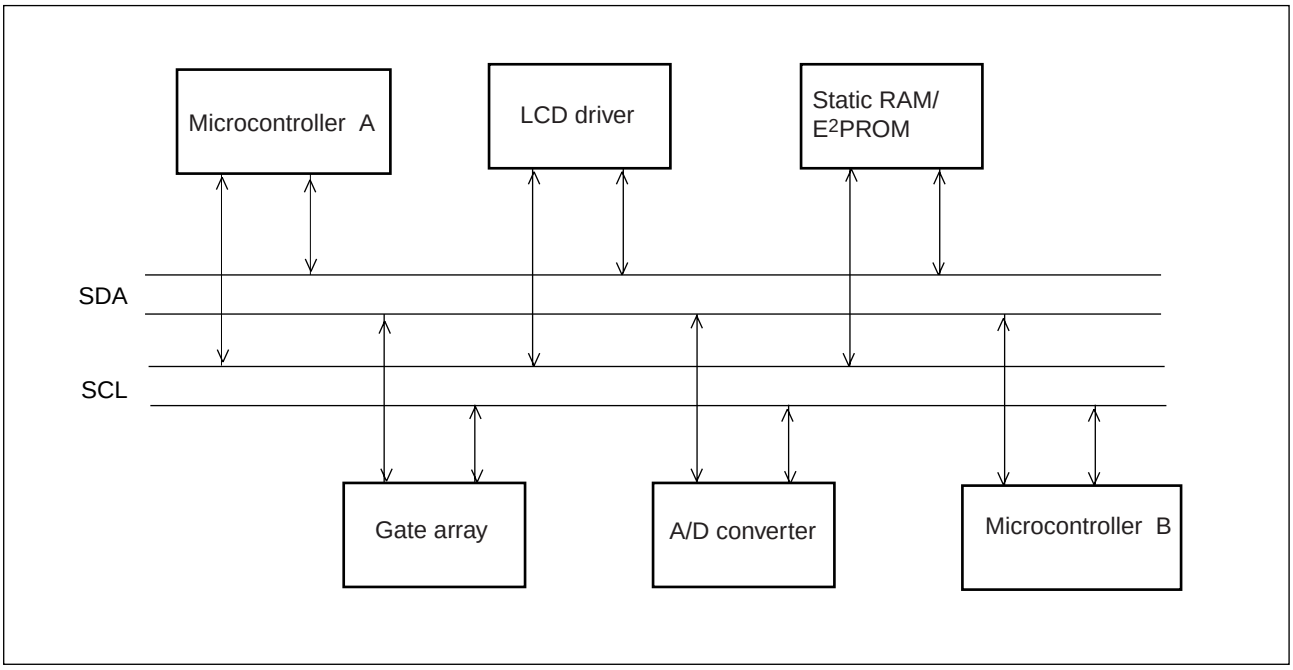
The multi-address I2C interface can connect two or more devices to the bus provided the upper limit of the bus capacitance does not exceed 400pF. It is a full-fledged multi-master bus equipped with collision detection and communication adjustment procedures designed to avoid the destruction of data if two or more masters attempt to start data transfer simultaneously.

This macro provides six addresses to implement the multi-address function.

A configuration example of the multi-address I2C interface is shown in Figure 16.1-1 "Multi-address I2C block Diagram".

The communication adjustment procedure permits only one master to control the bus when two or more masters attempt to control the bus so that messages are not lost or the contents of messages are not changed. Multi-master means that multiple masters attempt to control the bus simultaneously without losing messages.

**Figure 16.1-1 Multi-address I2C block Diagram**



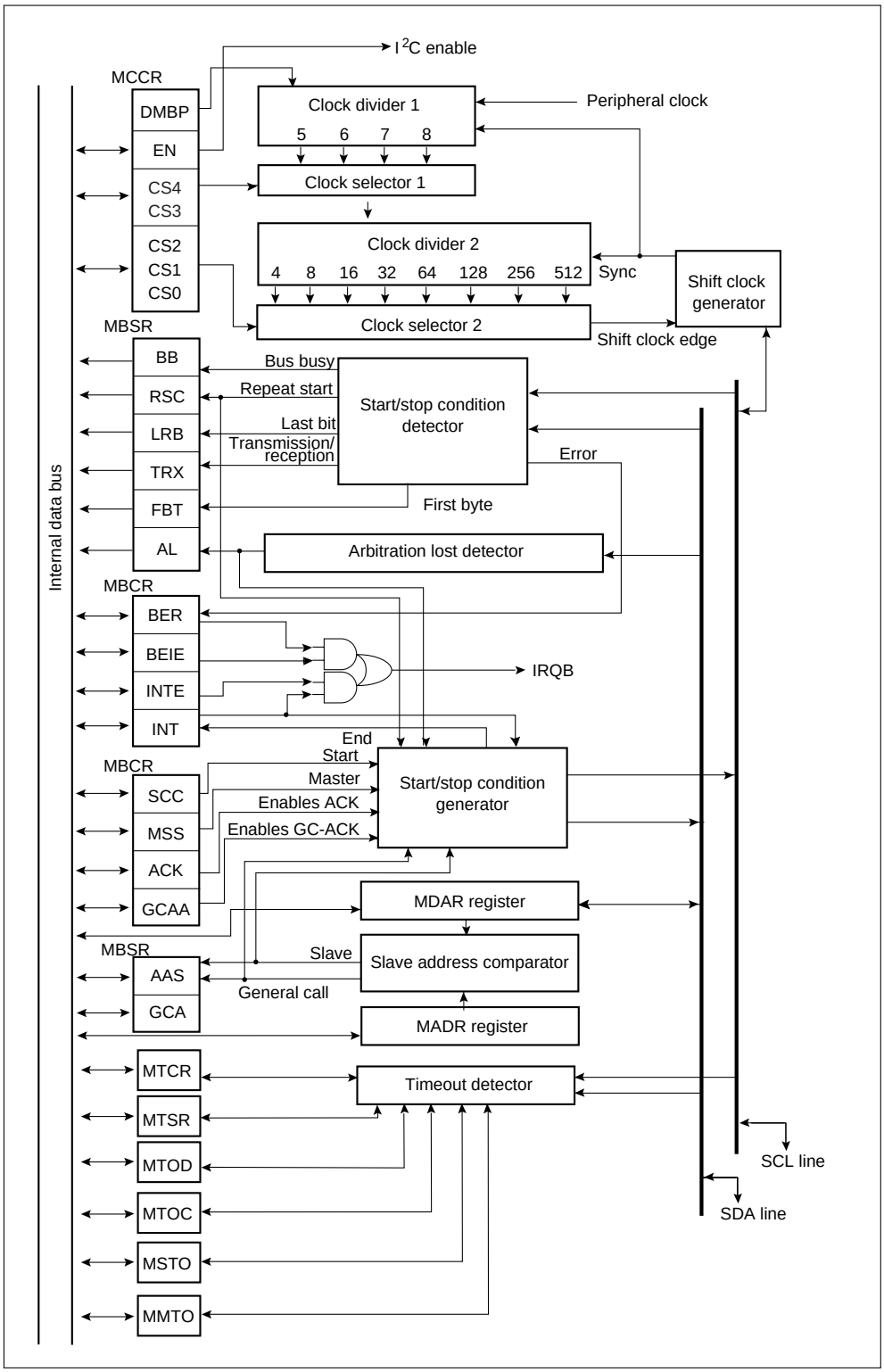
## 16.2 Configuration of the Multi-address I²C

**The multi-address I²C consists of the following 14 blocks.**

- **Clock selector, clock divider, shift clock generator**
- **Start/stop condition generator**
- **Start/stop condition detection**
- **Arbitration lost detection**
- **Timeout detection circuit**
- **Slave address comparison circuit**
- **Multi-address I²C bus status register (MBSR)**
- **Multi-address I²C bus control register (MBCR)**
- **Multi-address I²C clock control register (MCCR)**
- **Multi-address I²C address registers (MADR1 to 6)**
- **Multi-address I²C data register (MDAR)**
- **Multi-address I²C timeout control register (MTCR)**
- **Multi-address I²C timeout status register (MTSR)**
- **Multi-address I²C timeout data register (MTOD)**
- **Multi-address I²C timeout clock register (MTOC)**
- **Multi-address I²C slave timeout register (MSTO)**
- **Multi-address I²C master timeout register (MMTO)**

## ■ Multi-address I²C Block Diagram

**Figure 16.2-1 Multi-address I²C Block Diagram**



- **Clock selector, clock divider, shift clock generator**

  This circuit selects and generates shift clock for the multi-address I2C bus based on the internal clock.

- **Start/stop condition generator**

  When the bus is released (when the SCL and SDA lines are at a "H" level), transmitting a start condition causes the master to start communication . When the SDA line is changed from "H" to "L" when SCL = H, a start condition is generated. When a stop condition is generated, the master can stop communication. The stop condition is generated when the SDA line is changed from "L" to "H" when SCL = H.

- **Start/stop condition detector**

  This circuit detects the start/stop condition for data transfer.

- **Arbitration lost detector**

  This interface circuit supports the multi-master system. If two or more masters transmit data simultaneously, arbitration lost is generated. When logic level "1" is transmitted when the SDA line is at level "L", this state is regarded as arbitration lost. At this time, MBSR: AL is set to "1" and the master is changed into a slave.

- **Slave address comparator**

  After a start condition is transmitted, a slave address is transmitted. This address is seven-bit data, followed by a data direction bit (R/W) as bit 8. ACK is returned only to the slave whose address matches the transmitted address.

- **Timeout detector**

  This circuit detects a timeout, based on the value set in the MTOD, MTOC, MSTO, and MMTO registers.

- **MBSR register**

  The MBSR register indicates the status of the multi-address I2C interface. This register is read-only.

- **MBCR register**

  The MBCR register is used to select the operating mode, enables/disables interrupts, enables/disables acknowledge, and enables/disables general call acknowledge.

- **MCCR register**

  The MCCR register is used to permit the operation of the multi-address I2C interface and select the shift clock frequency.

- **MADR1 to 6 registers**

  The MADR1 to 6 registers is used to set the slave address.

- **MDAR register**

  The MDAR register stores shift data that is transmitted/received. For transmission, data written in this register is transmitted to the bus, starting from MSB. If this register is read during reception, "FF" is obtained.

❍ **MTCR register**

This MTCR egister is used to enable/disable the operation of the timeout detector and to control interrupts.

❍ **MTSR register**

This MTSR register is used to check the detection state of the timeout detector.

❍ **MTOD register**

The MTOD register is used to set the count value for a multi-address I²C timeout in the data line.

❍ **MTOC register**

The MTOC register is used to set the count value for a multi-address I²C timeout in the clock line.

❍ **MSTO register**

The MSTO register is used to set the count value for a multi-address I²C slave timeout.

❍ **MMTO register**

The MSTO register is used to set the count value for a multi-address I²C master timeout.

❍ **Multi-address I²C interface interrupt source**

**IRQB:**

An interrupt request is generated by the multi-address I²C interface when the bus error interrupt request bit is enabled (MBCR: BEIE = "1") and a bus error has occurred or when the transfer end interrupt enable bit is enabled (MBCR: INTE = "1") and data transfer is completed.

**IRQC:**

A timeout interrupt is generated if the set timeout is expired when the timeout detection function is enabled (MTCR: TS0 to TS2 are other than "000").

# 16.3 Pins of the Multi-address I2C

**The pins related to the multi-address I2C and a pins block diagram is shown below.**

## ■ Pins Related to the Multi-address I2C

The pins related to the multi-address I2C include the clock input/output pin (P30/SCL1), serial data input/output pin (P31/SDA1), and ALERT output pin (P32/ALERT). These pins are switched by the multi-address I2C operation enable bit (MCCR:EN) and multi-address I2C ALERT register.

**P31/SDA1 pin:**

The P31/SDA pin serves as an N-ch open-drain output port (P31) and a multi-address I2C data input/output pin (SDA1).

**P30/SCL1 pin:**

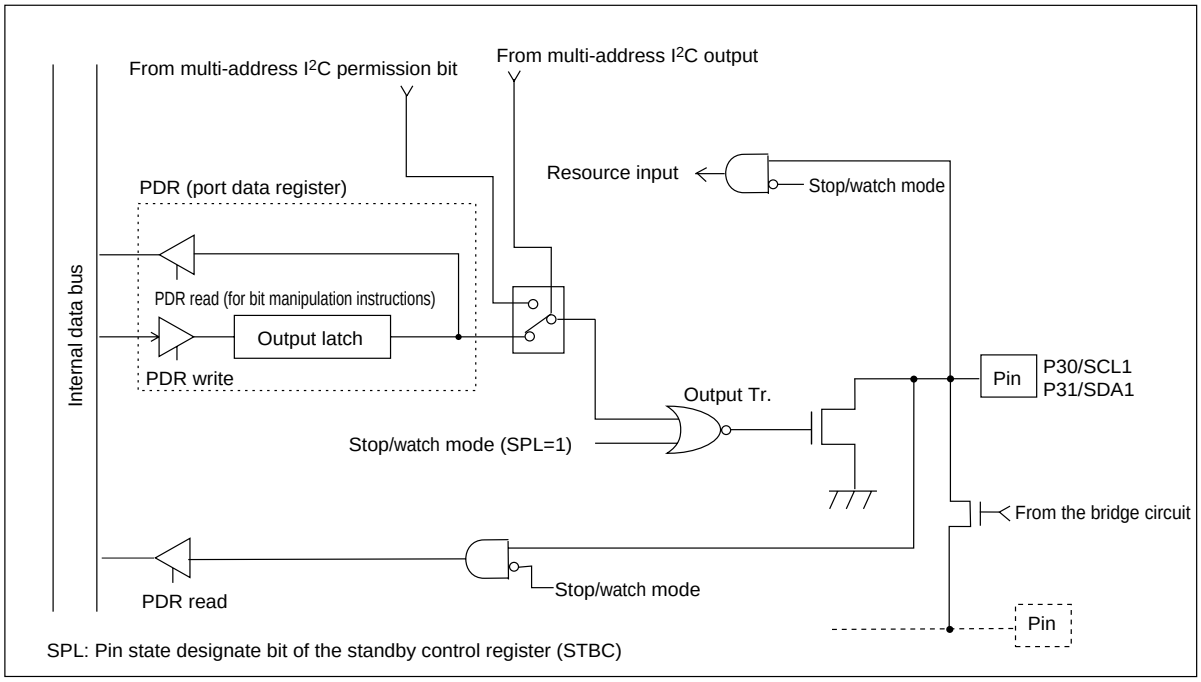
The P30/SCL1 pin serves as an N-ch open-drain output port (P30) and a multi-address I2C shift clock input/output pin (SCL1).

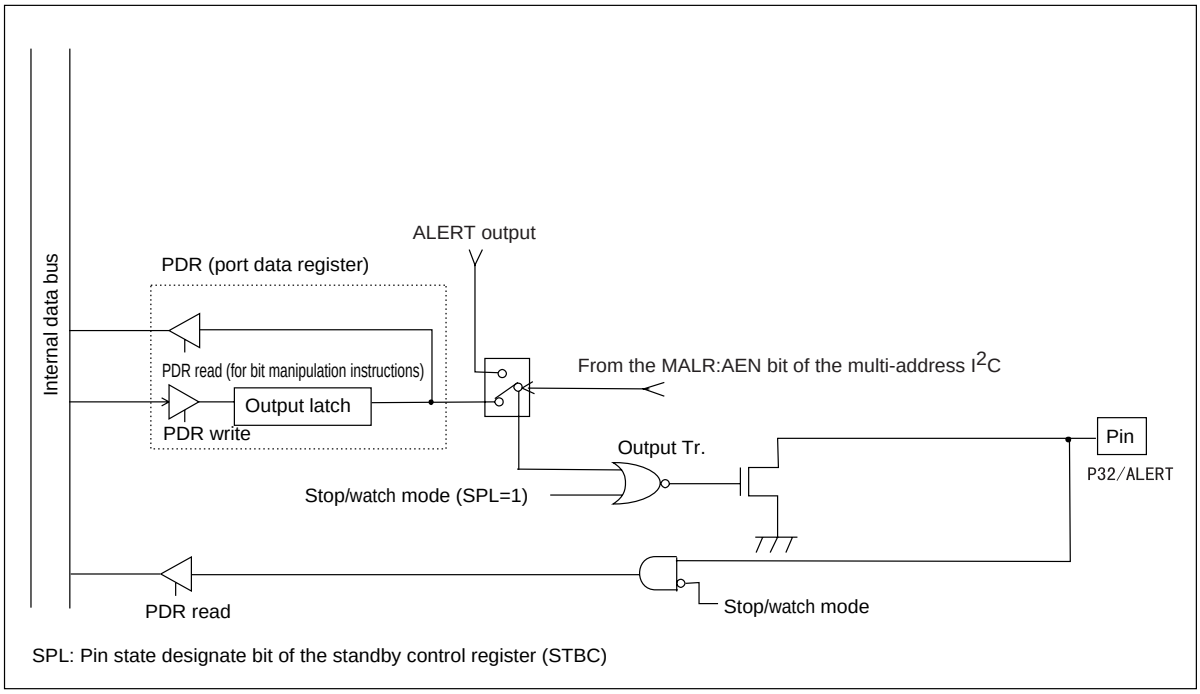
**P32/ALERT pin:**

The P32/ALERT pin serves as an N-ch open-drain output port (P32) and an ALERT output pin.

## ■ Block Diagram of Pins Related to Multi-address I²C

**Figure 16.3-1 Block Diagram of Pins Related to Multi-address I²C**




**Note:**

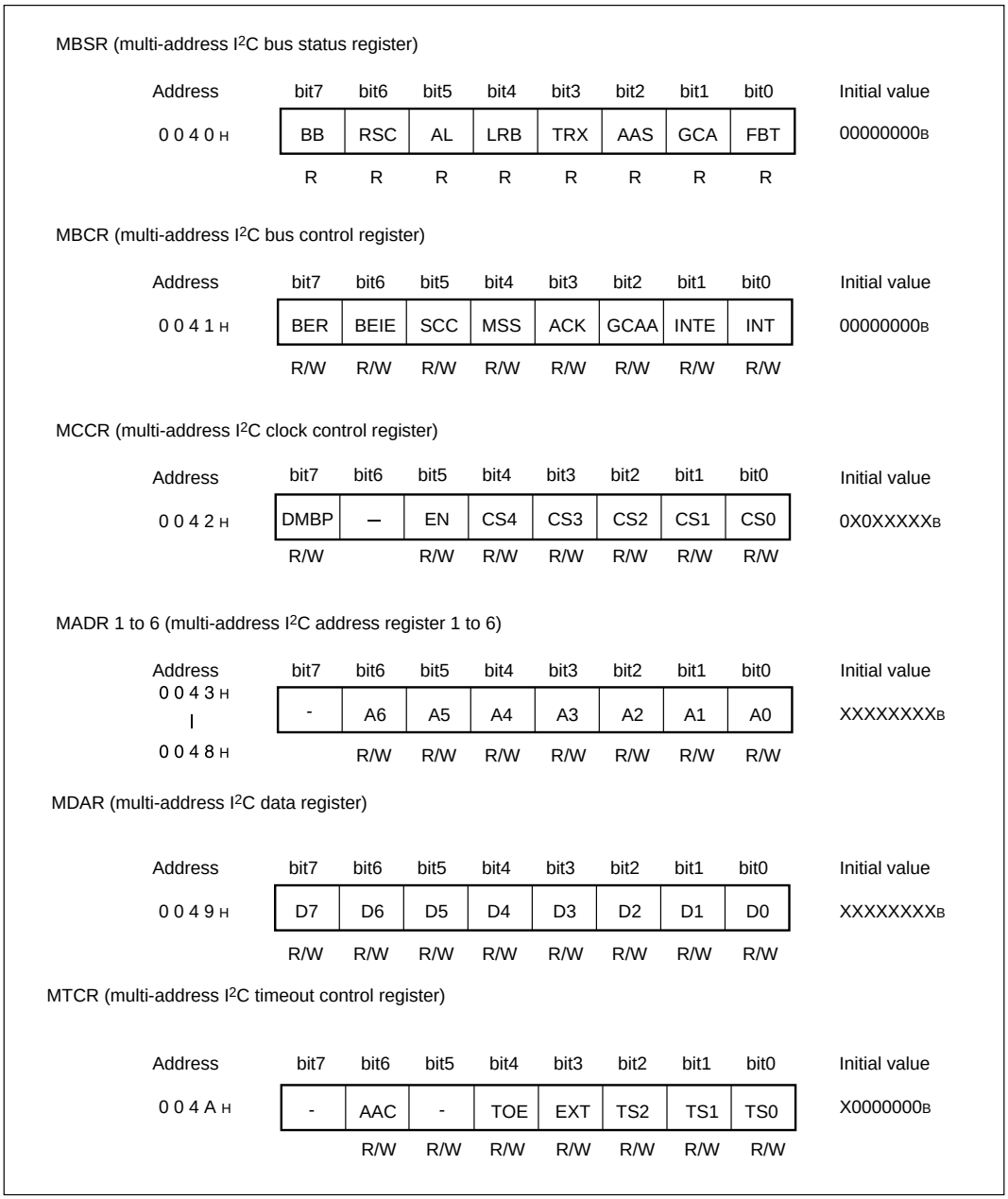
When the multi-address I²C function is used, P30/SCL1, P31/SDA1, and P32/ALERT pins must be pulled up externally.

# 16.4 Registers of the Multi-address I²C

**This section shows the registers related to the multi-address I²C.**

## ■ Registers Related to Multi-address I²C

**Figure 16.4-1 Registers Related to Multi-address I²C**



MTSR (multi-address I²C timeout status register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 4 B H | - | - | - | - | TDR | TCR | MTR | STR | XXXX0000B |
| | | | | | R/W | R/W | R/W | R/W | |

MTOD (multi-address I²C timeout data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 4 C H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

MTOC (multi-address I²C timeout clock register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 4 D H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

MMTO (multi-address I²C master timeout register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 4 E H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

MSTO (multi-address I²C slave timeout register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 4 F H | | | | | | | | | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

MALR (multi-address I²C ALART register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 5 0 H | - | - | - | - | ARAE | AR0 | ARF | AEN | XXXX0000B |
| | | | | | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
R : Read only
X : Undefined

# 16.4.1 Multi-address I²C Bus Status Register (MBSR)

**The MBSR register indicates the status of the interface.**

## ■ Multi-address I²C Bus Status Register (MBSR)

**Figure 16.4-2 Multi-address I²C Bus Status Register (MBSR)**

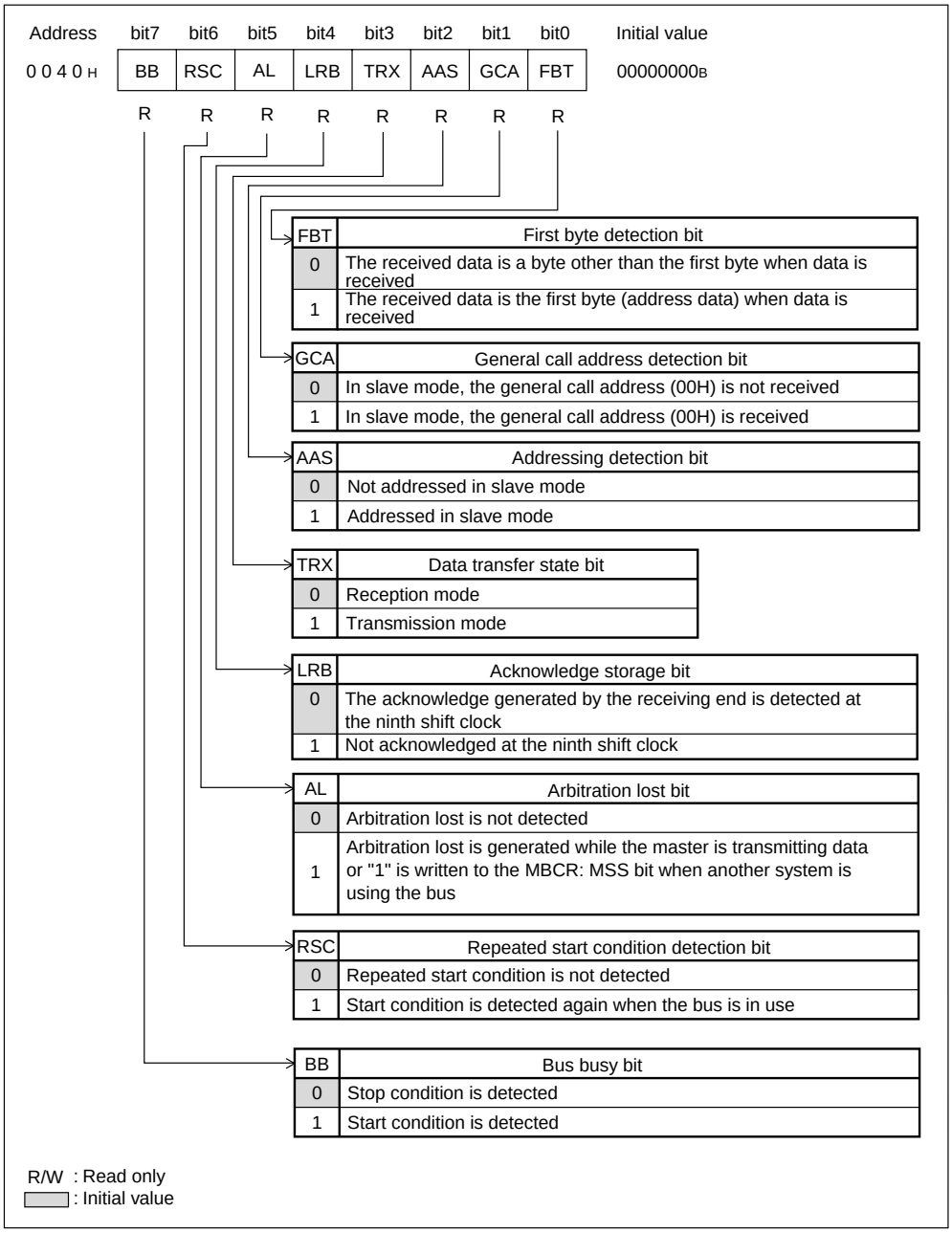


**Table 16.4-1 Function of Each Bit in Multi-address I²C Bus Status Register (MBSR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | BB:<br>Bus busy bit | This bit indicates the status of the bus. This bit is cleared when a stop condition is detected and set when a start condition or timeout is detected. |
| Bit 6 | RSC:<br>Repeated start condition detection bit | This bit detects the repeated start condition. This bit is set when a start condition is detected and cleared in the following state.<br>• "0" is written to the MBCR: INT bit,<br>• The slave address does not match the set address<br>• A start condition is detected during bus stop<br>• A stop condition is detected |
| Bit 5 | AL:<br>Arbitration lost bit | This bit detects arbitration lost. This bit is set in the following states.<br>• Arbitration lost is detected when the master is transmitting data<br>• "1" is written to the MBCR: MSS bit when another system is using the bus<br>This bit is also cleared when "0" is written to the MBCR: INT bit |
| Bit 4 | LRB:<br>Acknowledge storage bit | This bit stores the SDA line value of the 9th clock when the data byte is transferred.<br>• Cleared when an acknowledge bit is detected. (SDA = L)<br>• Set when an acknowledge bit is not detected. (SDA = H)<br>• Cleared with "0" when a start or stop condition is detected. |
| Bit 3 | TRX:<br>Data transfer state bit | This bit indicates whether the data transfer is performed in the transmission mode or the reception mode. |
| Bit 2 | AAS:<br>Addressing detection bit | This bit indicates addressing is performed in slave mode.<br>This bit is set when addressing is performed in slave mode and cleared when a start or stop condition is detected. |
| Bit 1 | GCA:<br>General call address detection bit | This bit detects a general call address.<br>If this bit is set to "1" in slave mode, the general call address (00H) is received.<br>This bit is cleared when a start or stop condition is detected. |
| Bit 0 | FBT:<br>First byte detection bit | This bit detects the first byte<br>This bit is always set to "1" in the start condition.<br>This bit is set to "1" when a start condition is detected and cleared when "0" is written to the MBCR: INT bit or when the set address does not match its address in slave mode. |

# 16.4.2 Multi-address I²C Bus Control Register (MBCR)

**The MBCR register is used to select the operating mode, enables/disables interrupts, enables/disables acknowledge, and enables/disables general call acknowledge.**

## ■ Multi-address I²C Bus Control Register (MBCR)

**Figure 16.4-3 Multi-address I²C Bus Controller Register (MBCR)**

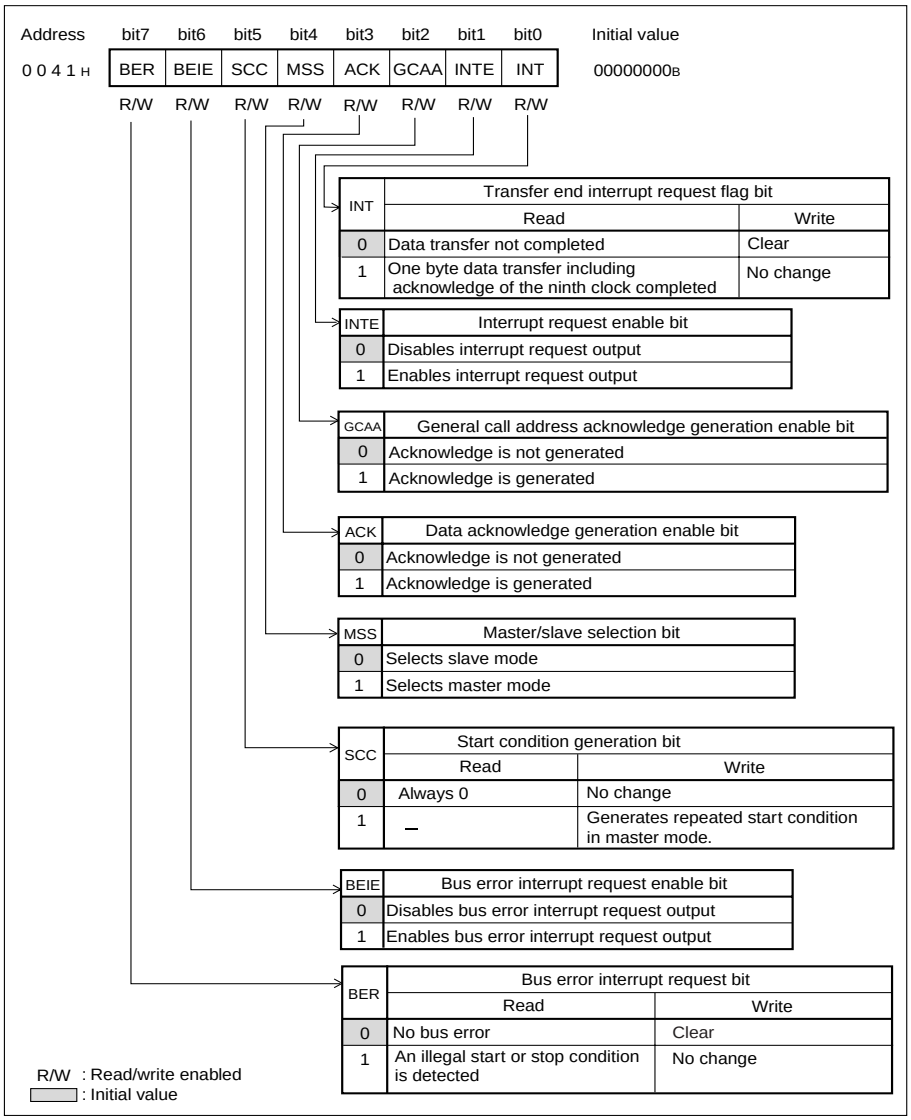


**Table 16.4-2 Function of Each Bit in Multi-address I2C Bus Controller Register (MBCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | BER:<br>Bus error interrupt request flag bit | This bit clears a bus error interrupt and detects a bus error.<br>When a bus error is detected, "0" is written and the bus error interrupt is cleared.<br>When "1" is written, there is no change and no effect on others.<br>When an illegal start or stop condition is detected during data transfer, this bit is set to "1". For RMW instructions, "1" is always read.<br>When this bit is set, the operation enable bit in the MCCR register is cleared, the multi-address I2C enters the hold mode, and data transfer is terminated. |
| Bit 6 | BEIE:<br>Bus error interrupt request enable bit | This bit enables (BEIE = 1) or disables (BEIE = 0) the generation of a bus error interrupt request.<br>When this bit is set and BER = 1, an interrupt request is sent to the CPU. |
| Bit 5 | SCC:<br>Start condition generation bit | When this bit is set, a repeated start condition in master mode is generated. (SCC = 1)<br>No change when "0" is written.<br>The read value of this bit is always "0."<br>**Note:**<br>1) Do not write SCC = 1 and MSS = 0 simultaneously.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and a stop condition is generated. |
| Bit 4 | MSS:<br>Master/slave selection bit | This bit selects the slave mode (MSS = 0) or the master mode (MSS = 1).<br>When this bit is cleared to "0," a stop condition is generated and the master mode is switched to the slave mode after transfer is completed.<br>When this bit is set to "1," the slave mode is switched to the master mode, a start condition is generated, and transfer is started.<br>If arbitration lost is generated when the master is transmitting data, this bit is cleared and the master mode is switched to the slave mode.<br>**Note:**<br>1) Do not write SCC = 1 and MSS = 0 simultaneously.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and a stop condition is generated. |
| Bit 3 | ACK:<br>Data acknowledge generation enable bit | This bit enables (ACK = 1) or disables (ACK = 0) the output of the acknowledge bit in the 9th clock at data reception. |
| Bit 2 | GCAA:<br>General call address acknowledge generation enable bit | This bit permits the generation of acknowledge when a general call address is received.<br>When a general call address is received in slave mode when this bit is set to "1," output of acknowledge is permitted.<br>Even if a general call address is received when "0" is written to this bit, acknowledge is not output. |

**Table 16.4-2 Function of Each Bit in Multi-address I²C Bus Controller Register (MBCR) (Continued)**

| Bit name | | Function |
|---|---|---|
| Bit 1 | INTE:<br>Transfer end interrupt request enable bit | This bit selects whether an interrupt at the end of transfer is enabled (INTE = 1) or disabled (INTE = 0).<br>When this bit is set and INT is set to "1," a transfer end interrupt request is sent to the CPU. |
| Bit 0 | INT:<br>Transfer end interrupt request flag bit | With this bit, the data transfer end interrupt request flag can be cleared. In addition, it can be determined whether the interrupt is detected.<br>When "0" is written, the transfer end interrupt request flag is cleared. When "1" is written, no change occurs.<br>If any of the following four conditions is met when one byte transfer including the acknowledge bit is completed (including the acknowledge bit in the 9th clock), this bit is set to "1."<br>• Bus master mode<br>• Addressed slave<br>• A general call address is received<br>• Arbitration lost is generated<br>When this bit is set to "1," the SCL line is kept at the "L" level. This bit is cleared when "0" is written to this bit. At this time, this macro releases the SCL line and transfers the next byte. This bit is also cleared to "0" when a start or stop condition is generated in master mode.<br>**Note:**<br>1) If "1" is written to SCC when INT = 0, "1" in the SCC bit has a higher priority and a start condition is generated.<br>2) If "0" is written to MSS when INT = 0, "0" in the MSS bit has a higher priority and the stop condition is generated.<br>For RMW instructions, "1" is always read. |

**Note:**

When the interrupt request flag bit (MBCR: BER) is cleared, do not rewrite the interrupt request enable bit (MBCR: BEIE) simultaneously.

Only when the multi-address I²C enable bit (MCCR: EN) is set, values can be written to the ACK, GCAA, and INTE bits in the MBCR register.

# 16.4.3 Multi-address I2C Clock Control Register (MCCR)

**The multi-address I2C clock control register is used to enable the operation of multi-address I2C and to select shift clock frequency.**

## ■ Multi-address I2C Clock Control Register (MCCR)

**Figure 16.4-4 Multi-address I2C Clock Control Register (MCCR)**

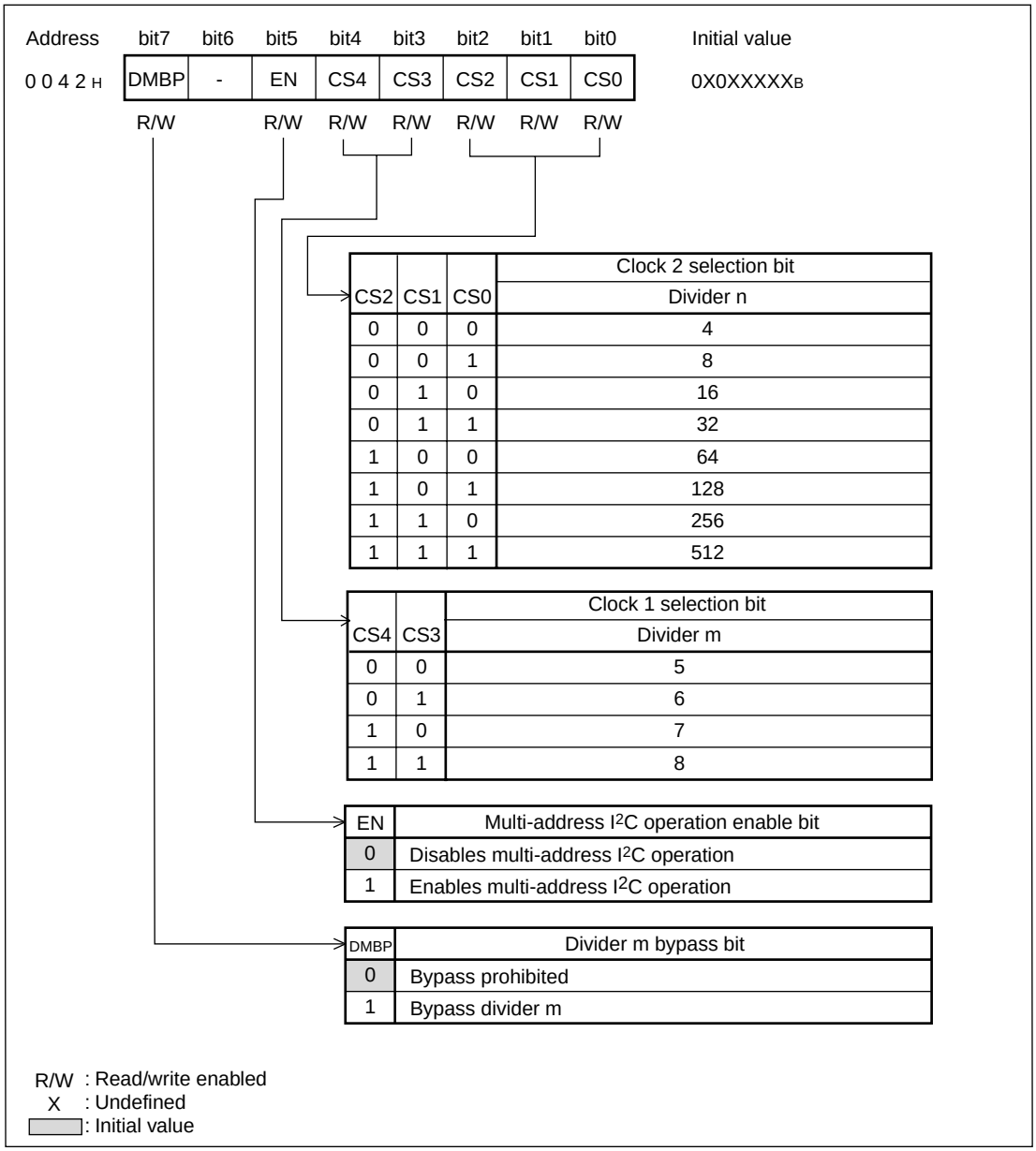


**Table 16.4-3 Function of Each Bit in Multi-address I²C Clock Control Register (MCCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | DMBP:<br>Divider m bypass bit | This bit is used to bypass the m divider for generating a shift clock frequency.<br>When "0" is written, the value set in CS3 and CS4 becomes the value of the m divider.<br>When "1" is written, the m divider is bypassed. This is equivalent to m = 1.<br>In read cycle, the present set value can be read.<br>When n = 0 (CS2 = CS1 = CS0 = 0), do not set this bit. |
| Bit 6 | Unused bit | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 5 | EN:<br>Multi-address I²C operation permission bit | This bit permits the operation of the multi-address I²C interface (EN = "1").<br>When the bit is "0", each bit of the MBSR and MBCR registers (excluding BER and BEIE bits) is cleared to "0".<br>When the MBCR:BER bit is set, the bit is cleared. |
| Bit 4<br>Bit 3 | CS4, CS3:<br>Clock 1 selection bit | This bit sets shift clock frequency.<br>Shift clock frequency Fsck is determined by the following formula. |
| Bit 2<br>Bit 1<br>Bit 0 | CS2, CS1, CS0:<br>Clock 2 selection bit | $Fsck = \frac{2/t_{inst}}{(m \times n + 2)}$ (1)<br>Where, $F_{inst}$ is an instruction cycle (the clock in the SYCC selected by the SCS bit). When DMBP is "0", m is selected by CS4 and CS3. When DMBP is "1," m is "1." n is selected by CS2, CS1, and CS0. |

# 16.4.4 Multi-address I2C Address Registers (MADR1 to 6)

**The MADR1 to 6 registers are used to set slave addresses.**

## ■ Multi-address I2C Address Registers (MADR1 to 6)

**Figure 16.4-5 Multi-address I2C Address Registers (MADR1 to 6)**

MADR 1 to 6 (multi-address I2 C address registers 1 to 6)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0043H<br>\|<br>0048H | - | A6 | A5 | A4 | A3 | A2 | A1 | A0 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

In slave mode, the value in this register is compared with the slave address stored in MADR1 to 6 after the requested address is received. When they match, acknowledge is transmitted to the master as the 9th shift clock.

# 16.4.5 Multi-address I²C Data Register (MDAR)

**The MDAR register is used to set transmission data and to store received data.**

## ■ Multi-address I²C Data Register (MDAR)

**Figure 16.4-6 Multi-address I²C Data Register (MDAR)**

MDAR (multi-address I²C data register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0049H | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

In master mode, the data written in the register is shifted to the SDA line bit by bit from the MSB bit.

The write side in this register is made up of a double buffer. When the bus is in use (MBSR: BB = 1), written data is loaded to the eight-bit shift register when the transfer of the present byte is completed. The data in the shift register is shifted and output to the SDA line bit by bit. The value written to this register has no effect on the present data transfer. Also in slave mode, the same function can be used after the address is determined.

When MBCR: INT = 1 at data reception (MBSR: TRX = 0), the received data can be read from this register. Since the register for serial transfer is read directly in read cycle, the received data is effective only when MBCR: INT = 1.

# 16.4.6 Multi-address I2C Timeout Control Register (MTCR)

**The MTCR register is used to control the SM bus timeout detection.**

## ■ Multi-address I2C Timeout Control Register (MTCR)

Figure 16.4-7 "Multi-address I2C Timeout Control Register (MTCR)" shows the bit configuration of the multi-address I2C timeout control register (MTCR)

**Figure 16.4-7 Multi-address I2C Timeout Control Register (MTCR)**

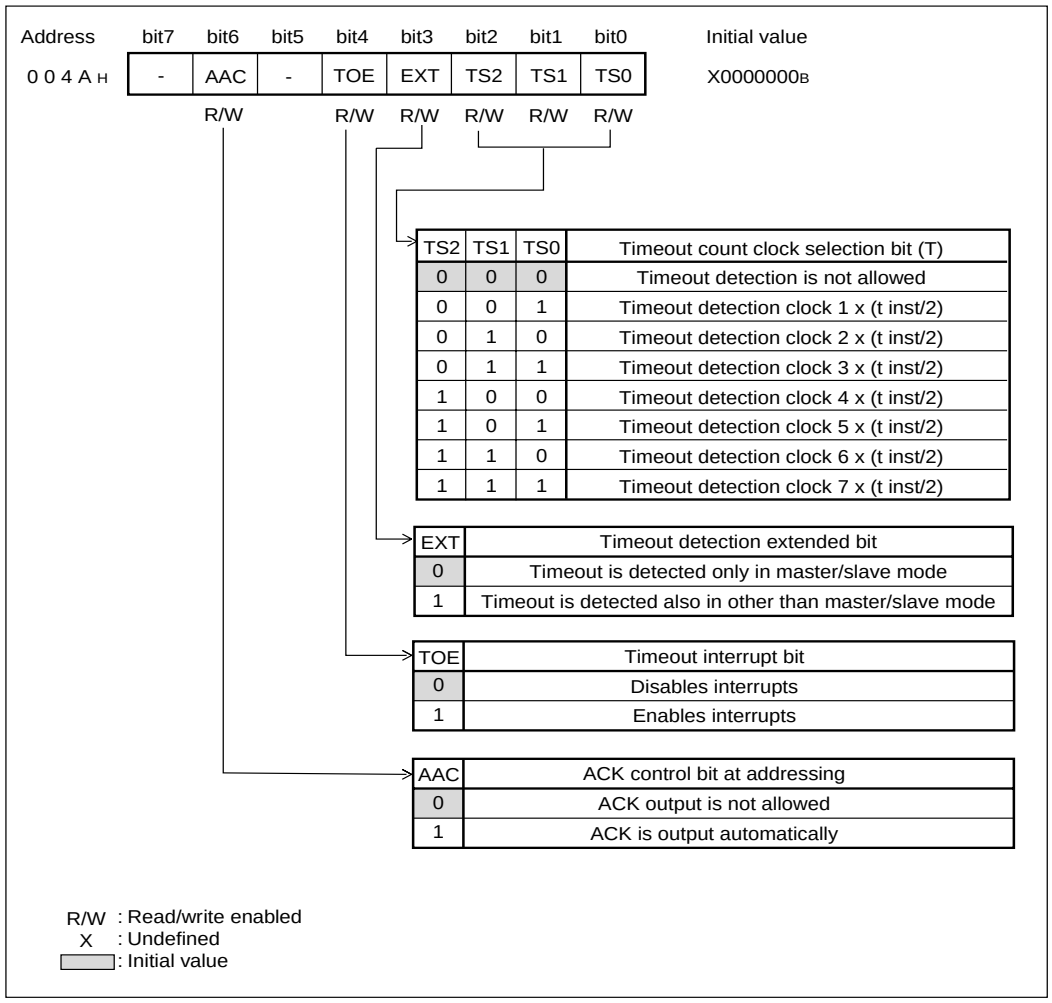


**Table 16.4-4 Function of Each Bit in Multi-address $I^2C$ Bus Status Register (MTCR)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | Unused bit | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 6 | AAC:<br>ACK control bit at addressing | This bit controls ACK at address matching.<br>When this bit is set to "1", ACK control is performed automatically. (ACK is returned automatically when the addresses 1 to 5 match.<br>When "0" is written to this bit, ACK at addressing is not output. |
| Bit 5 | Test bit | This bit is a test bit.<br>Write "1" when multi-address $I^2C$ is used.<br>**Note:**<br>Though the initial value of this bit is "0", write "1" to this bit when multi-address $I^2C$ is used. |
| Bit 4 | TOE:<br>Timeout interrupt enable bit | This bit enables/disables timeout interrupts.<br>When "1" is written to this bit, timeout interrupts are enabled and an interrupt request is sent to the CPU.<br>When "0" is written to this bit, timeout interrupts are disabled. |
| Bit 3 | EXT:<br>Timeout detection extended bit | This bit makes extended control for detecting a timeout.<br>When "1" is written to this bit, the timeout detection function also operates in a mode other than master/slave mode.<br>When "0" is written to this bit, the timeout detection function operates only in master/slave mode.<br>**Note:**<br>This bit is effective only when timeout detection is enabled (MTCR: TS0 to TS2 is other than "000".) |
| Bit 2<br>Bit 1<br>Bit 0 | TS2, TS1, TS0:<br>Timeout count selection bits (T) | These bits select the clock for detecting a timeout.<br>When TS2 = 0, TS1 = 0, and TS0 = 0, the timeout detection function is disabled.<br>With the clock selected in these bits [$T \times (t_{inst}/2) \times 10$], the low period in the SCL1 and SDA1 lines is counted. |

# 16.4.7 Multi-address I2C Timeout Status Register (MTSR)

**The MTSR register indicates the SM bus timeout detection status.**

## ■ Multi-address I2C Timeout Status Register (MTSR)

16.4-8 "Multi-address I2C Timeout Status Register (MTSR)" shows the bit configuration of the multi-address I2C timeout status register (MTSR).

**Figure 16.4-8 Multi-address I2C Timeout Status Register (MTSR)**

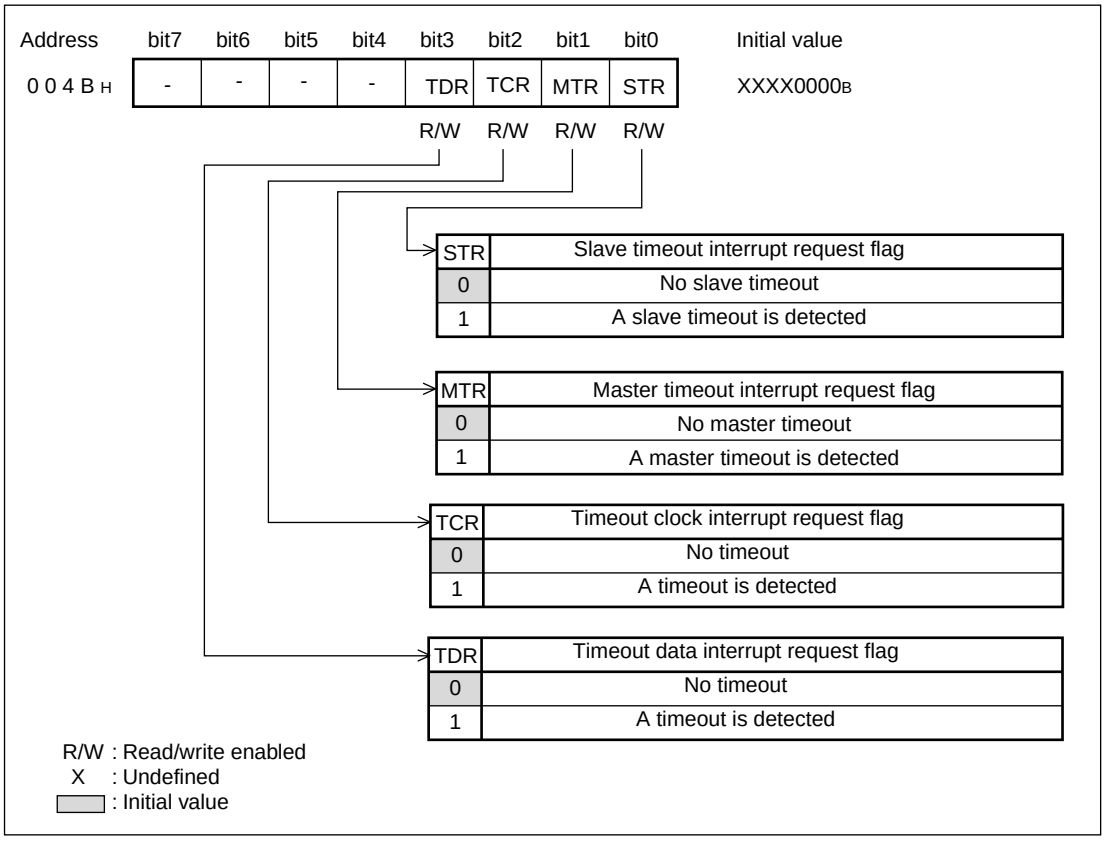


**Table 16.4-5 Function of Each Bit in Multi-address I²C Timeout Status Register (MTSR)**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4 | Unused bits | The read value is undefined.<br>Writing has no efect on operation. |
| Bit 3 | TDR:<br>Timeout data interrupt request flag | This bit detects a timeout of the data line.<br>When timeout data is detected, this bit is set to "1".<br>If timeout interrupts (MTCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |
| Bit 2 | TCR:<br>Timeout clock interrupt request flag | This bit detects a timeout of the clock line.<br>When a timeout clock is detected, this bit is set to "1".<br>If timeout interrupts (MTCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |
| Bit 1 | MTR:<br>Master timeout interrupt request flag | This bit detects a master timeout.<br>When a master timeout is detected, this bit is set to "1".<br>If master timeout interrupts (MTCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no meaning. |
| Bit 0 | STR:<br>Slave timeout interrupt request flag | This bit detects a slave timeout.<br>When a slave timeout is detected, this bit is set to "1".<br>If slave timeout interrupts (MTCR: TOE) are enabled at this time, an interrupt occurs.<br>• When a timeout is detected, this bit is cleared to "0".<br>• "1" written to this bit has no significance. |

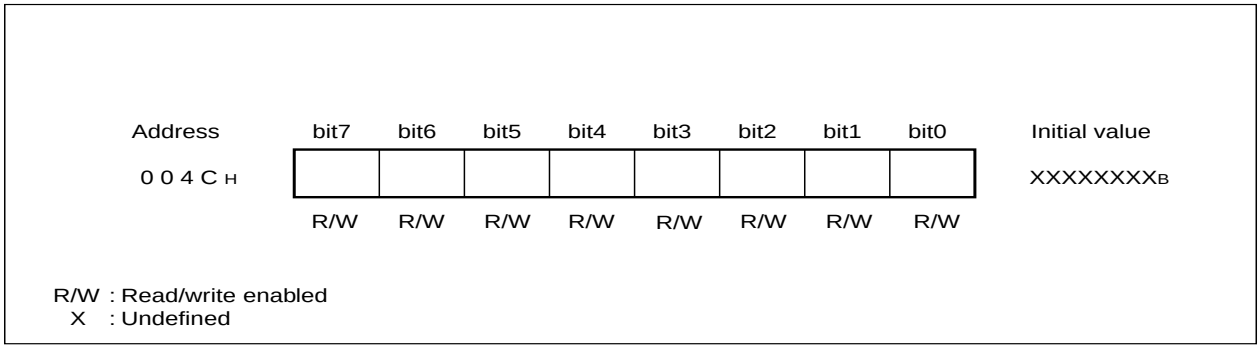
# 16.4.8 Multi-address I2C Timeout Data Register (MTOD)

**The MTOD register is used to detect an SM bus timeout (data line).**

## ■ Multi-address I2C Timeout Data Register (MTOD)

Figure 16.4-9 "Multi-address I2C Timeout Data Register (MTOD)" shows the bit configuration of the multi-address I2C timeout data register (MTOD)

**Figure 16.4-9 Multi-address I2C Timeout Data Register (MTOD)**




If the value written to this register plus one matches the counted value of the low time period of the SDA1 line when the timeout detection function is enabled (MTCR: TS0 to TS2 is other than "000"), the timeout data interrupt request flag (MTSR: TDR) is set to "1".

The low time period of the SDA1 line is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. When the SDA1 line is at a high level, the counter value is cleared to "0", and counting starts again when the SDA1 line is at a low level. When the counter value matches "the value set in the timeout data register (MTOD) + 1", a timeout is detected and the timeout data interrupt request flag is set.

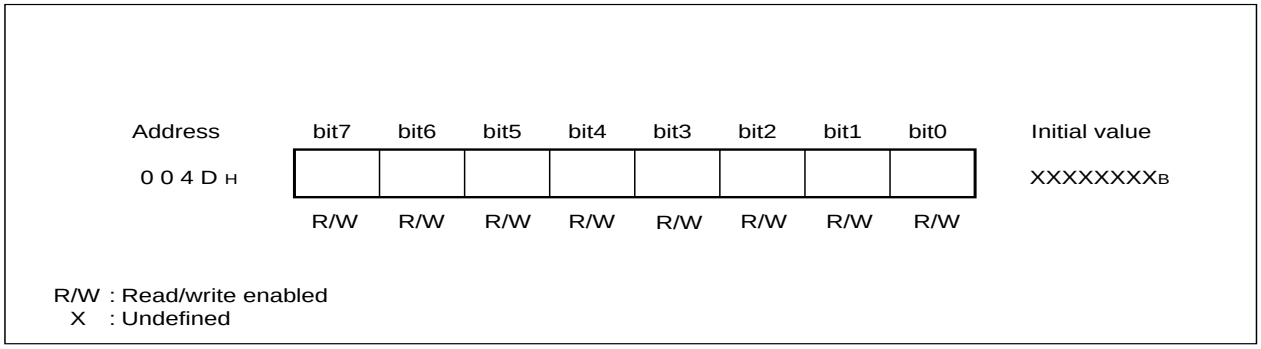
# 16.4.9 Multi-address I2C Timeout Clock Register (MTOC)

**The MTOC register is used to detect an SM bus timeout (clock line).**

## ■ Multi-address I2C Timeout Clock Register (MTOC)

Figure 16.4-10 "Multi-address I2C Timeout Clock Register (MTOC)" shows the bit configuration of the multi-address I2C timeout clock register (MTOC)

**Figure 16.4-10 Multi-address I2C Timeout Clock Register (MTOC)**




If the value written to this register plus one matches the counted value of the low time period of the SCL1 line when the timeout detection function is enabled (MTCR: TS0 to TS2 is other than "000"), the timeout clock interrupt request flag (MTSR: TCR) is set to "1".

The low time period of the SCL1 line is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. When the SCL1 line is at a high level, the counter value is cleared to "0" and counting starts again when the SCL1 line is at a low level. When the counter value matches "the value set in the timeout clock register (MTOC) + 1", a timeout is detected and the timeout clock interrupt request flag is set.

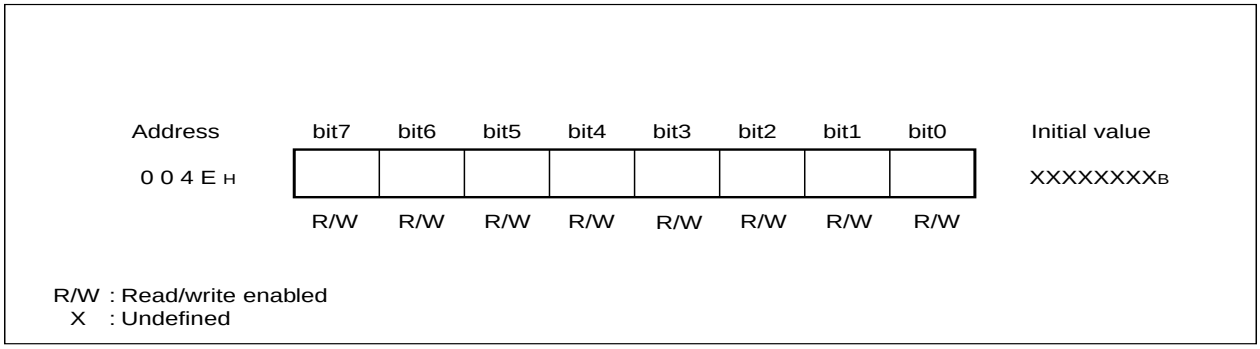
# 16.4.10 Multi-address I2C Master Timeout Register (MMTO)

**The MMTO register is used to detect an SM bus timeout.**

## ■ Multi-address I2C Master Timeout Register (MMTO)

Figure 16.4-11 "Multi-address I2C Master Timeout Register (MMTO)" shows the bit configuration of the multi-address I2C master timeout register (MMTO)

**Figure 16.4-11 Multi-address I2C Master Timeout Register (MMTO)**




When the value written to this register plus one matches the cumulative value of the low time period of the SCL1 line counted from start to acknowledge (or from acknowledge to acknowledge or from acknowledge to stop) when the timeout detection function is enabled (MTCR: TS0 to TS2 is other than "000"), the master timeout interrupt request flag (MTSR: MTR) is set to "1".

The low time period of the SCL1 line is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 10. In master mode, the low time period cumulative value of the SCL1 line is counted only when the SCL1 line is at a low level and counting is stopped when the SCL1 line is at a high level. At start, acknowledge or stop, or by exiting the master mode, the counter value is cleared to "0". When the counter value matches "the value set in the master timeout register (MMTO) + 1", a master timeout is detected and the master timeout interrupt request flag is set.

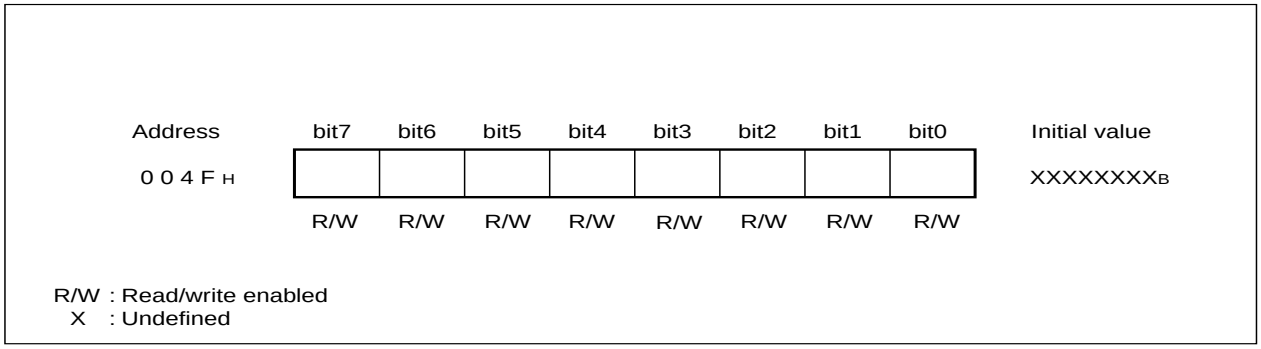
# 16.4.11 Multi-address I2C Slave Timeout Register (MSTO)

**The MSTO register is used to detect an SM bus timeout.**

## ■ Multi-address I2C Slave Timeout Register (MSTO)

Figure 16.4-12 "Multi-address I2C Slave Timeout Register (MSTO)" shows the bit configuration of the multi-address I2C slave timeout register (MSTO)

**Figure 16.4-12 Multi-address I2C Slave Timeout Register (MSTO)**




When the value written to this register plus one matches the cumulative value of the low time period of the SCL1 line counted from start to stop when the timeout detection function is enabled (MTCR: TS0 to TS2 is other than "000"), the slave timeout interrupt request flag (MTSR: STR) is set to "1".

The low-time period of the SCL1 line is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) divided by 10 and the counter is incremented by the clock further divided by 20. In slave mode, the low-time period cumulative value of the SCL1 line is counted only when the SCL1 line is at a low level and counting is stopped when the SCL1 line is at a high level. At start or stop or by exiting the slave mode, the counter value is cleared to "0". When the counter value matches "the value set in the slave timeout register (MSTO) + 1", a slave timeout is detected and the slave timeout interrupt request flag is set.

# 16.4.12 Multi-address I2C ALERT Register (MALR)

**The MALR register is used for an SM bus ALERT signal.**

## ■ Multi-address I2C ALERT Register (MALR)

Figure 16.4-13 "Multi-address I2C ALERT Register (MALR)" shows the bit configuration of the multi-address I2C ALERT register (MALR).

**Figure 16.4-13 Multi-address I2C ALERT Register (MALR)**

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0050H | - | - | - | - | ARAE | ARO | ARF | AEN | XXXX0000B |
| | | | | | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

When the ALERT detection permission bit is set to ON (MALR: AEN = 1) and the ALERT_ACK permission bit is set to "enabled" (MALR:ARAE = 1), ACK is returned and the ALERT request flag (MALR:ARF) is set when data set in the multi-address I2C address register 6 (ALERT command: 0001 100) is detected.

When the ALERT detection permission bit is ON (MALR: AEN = 1) and the ALERT request output bit is set to ON (MALR:ARO = 1), an ALERT request is output from the P32/ALERT pin ("L" output). "ALERT command-0001 100" should already be set to multi-address I2C address register 6.

The ALERT_ACK permission bit (MALR: ARAE) can control ACK when addressing for address 6. When "0" is written in the bit, ACK at addressing for address 6 is not output if the addresses are identical. If the bit is "1," ACK control at addressing time for address 6 is done automatically.

# 16.5 Multi-address I²C Interrupts

**The multi-address I²C interface may generate an interrupt request when the data transfer is completed, a bus error has occurred, or a timeout is detected.**

### ■ Interrupt at Bus Error

When the following conditions are met, a bus error is assumed to have occurred and the multi-address I²C interface is stopped.

1. When a stop condition is detected in master mode.
2. When a start or stop condition is detected when the first byte is being transmitted and received.
3. When a start or stop condition is detected when data (excluding the first bit of start, stop, and data) is being transmitted and received.

If the bus error interrupt request enable bit is enabled (MBCR: BEIE = 1) at this time, an interrupt request is output to the CPU. Clear the interrupt request by writing "0" to the BER bit in the interrupt processing routine.

If a bus error has occurred in spite of the BEIE bit value, the BER bit is set to "1".

Regardless of the BEIE bit value, the BER bit is set to "1" if a bus error occurs.

### ■ Interrupt at Data Transfer Completion

When data transfer is completed and the transfer end interrupt request enable bit is enabled (MBCR: INTE = 1), an interrupt request (IRQB) is output to the CPU. Clear the interrupt request by writing "0" to the INT bit in the interrupt processing routine.

If data transfer is completed in spite of the INTE bit value, the INT bit is set to "1".

### ■ Interrupt at Timeout Detection

If the specified timeout time has expired when the timeout detection function is enabled (MTCR: TS0 to TS2 is other than "000"), a timeout interrupt is generated (IRQC). The timeout can be checked with each interrupt request flag of the multi-address I²C bus status register (MTSR). When the timeout detection extended bit (MTOR: EXT) is set, the bus is also monitored in a mode other than master/slave mode.

## ■ Register and Vector Table Address Related to Interrupt of Multi-address I²C

**Table 16.5-1 Registers and Vector Table Addresses Related to Interrupt of Multi-address I²C**

| Interrupt name | Interrupt level setting register | | | Vector table address | |
|---|---|---|---|---|---|
| | Register | Bit to be set | | Upper | Lower |
| IRQB | ILR3 (007DH) | LB1 (bit 7) | LB90 (bit 6) | FFE4H | FFE5H |
| IRQC | ILR4 (007EH) | LC1 (bit 1) | LC0 (bit 0) | FFE2H | FFE3H |

For interrupt operation, see Section 3.4.2 "Interrupt Processing.

# 16.6 Operation of the Multi-address I²C

**The multi-address I²C interfaceis a serial data base of 8-bit data synchronized with the shift clock.**

## ■ Multi-address I²C System

### ❍ Operation mode

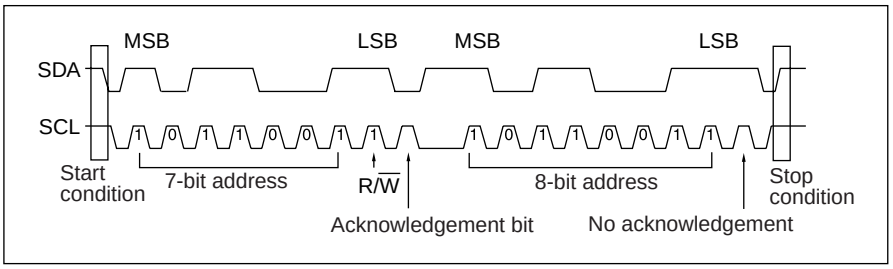
The multi-address I²C bus system uses a serial data line (SDA) and a serial clock line (SCL) to transfer data. All connected devices require an open-drain or open collector output. The logic function is used by connecting a pull-up resistor.

Each device connected to the bus has a unique address and can be set by software. Among the devices, simple master/slave relations are established and master devices function as master transmitters or master receivers. The multi-address I²C interface is a full-fledged multi-master bus equipped with collision detection and arbitration functions so that data destruction can be prevented even if two or more masters attempt to start data transfer simultaneously.

## ■ Multi-address I²C Protocol

Figure 16.6-1 "Data Transfer Example" shows the format required for data transfer.

**Figure 16.6-1 Data Transfer Example**




After a start condition (S) is generated, a slave address is transmitted. This address is a seven-bit address followed by a data direction bit (R/W) as 8th bit. Data transfer is always ended with the master stop condition (P). It is also possible to address to another slave without generating a stop condition by generating a repeated start condition (Sr).

### ■ Start Condition

When the master is not connected to the bus (the logic of SCL and SDA is "H") in states where the bus is released, the master generates a start condition. As indicated in Figure 16.6-1 "Data Transfer Example", a start condition is generated when the SDA line is changed from "H" to "L" in states where SCL is at the "H" level. At this time, new data transfer starts and master/slave operation starts. The two methods for generating a start condition are shown as follows.

- Writing "1" to the MBCR: MSS bit in states where the multi-address $I^2C$ bus is not used (MBCR: MSS = 0, MBSR: BB = 0, MBCR: INT = 0, MBSR: AL = 0). Thereafter, MBSR: BB is set to "1" to indicate bus busy.
- Writing "1" to the MBCR: SCC bit in interrupt states in bus master mode (MBCR: MSS = 1, MBSR: BB = 1, MBCR: INT = 1, MBSR: AL = 0) and generates a repeated start condition.

Even if "1" is written to the MBCR: MSS bit or "1" is written to the MBCR: SCC bit under conditions other than the above conditions, it is ignored. If "1" is written to the MBCR: MSS bit when another system is using the bus (in idle state), the MBSR: AL bit is set to "1".

### ■ Addressing

In master mode, the BB and TRX bits in the MBSR register are set to 1 after a start condition is generated and the contents of the MDAR register in the slave address are output from the MSB in turn. This address data consists of 8-bits with a seven-bit slave address, followed by a R/W bit indicating the data transfer direction (Bit 0 in MDAR).

After the address data is transmitted, the master receives acknowledge from the slave. The SDA line is set to "L" by the 9th clock and the master receives the acknowledge bit from the receiving end (see Figure 16.6-1 "Data Transfer Example"). At this time, the R/W bit (MDAR: bit 0) is reversed and stored in the MBSR: TRX bit.

In slave mode, the BB and TRX bits in the MBSR register are set to "1" and "0", respectively, after a start condition is detected and data from the master is received by the MDAR register. After receiving the address data, the MDAR and MADR1 to 6 registers are compared. If the values match, MBSR: AAS is set to "1" and acknowledge is transmitted to the master. Thereafter, bit 0 of the received data (bit 0 in the MDAR register) is stored in the MBSR: TRX bit.

### ■ Data Transfer

After addressing of the slave is achieved, data can be transmitted and received in byte units in the direction determined by the R/W bit sent by the master.

Each byte output to the SDA line is fixed to 8-bits. As shown in Figure 16.6-1 "Data Transfer Example", the receiving device transmits acknowledge to the transmitting device by stabilizing the SDA line to the "L" level when the acknowledge clock pulse is "H". With the MSB at the head, each bit of data is transmitted in one clock pulse. Each time a byte is transferred, acknowledge must be transmitted and received. Therefore, 9 clock pulses are required to transfer one complete data byte.

### ■ Acknowledge

Acknowledge is transmitted from the receiving end for the 9th clock of data byte transfer from the transmitting end.

When data is received, the acknowledge bit can be enabled (MBCR: ACK = 1) or disabled (MBCR: ACK = 0) with the MBCR: ACK bit.

When transmitting data, acknowledge from the receiving end is stored in the MBSR: LRB bit.

## ■ Stop Condition

By generating a stop condition, the master can release the bus to terminate communication. A stop condition can be generated by changing the SDA line from "L" to "H" when the SCL line is at the "H" level. It is a signal to notify the bus connection device of the end of communication (bus free) in master mode. The master can generate start conditions continuously without generating a stop condition. This is called the repeated start condition.

In bus master mode, a stop condition is generated by writing "0" to the MBCR: MSS bit in the interrupt state (MBCR: MSS = 1, MBSR: BB = 1, MBCR: INT = 1, MBSR: AL = 0) and the master mode is switched to the slave mode..

Even if "0" is written to the MBCR: MSS bit in other the above, it is ignored.

## ■ Arbitration

This interface circuit is a full-fledged multi-master bus that can connect two or more masters. If a master transfers data and another master transfers data simultaneously, an arbitration is generated.

An arbitration occurs in the SDA line when the SCL line is at the "H" level. The master recognizes the occurrence of an arbitration lost when its transmission data is "1" and data on the SDA line is at the "L" level, and then it sets data output to off and sets the MBSR: AL bit to "1". When the MBSR: AL is set to "1", "0" is written to MBCR: MSS and MBSR: TRX. As a result, the TRX is cleared and the master mode is switched to the slave reception mode.

# 16.7 Notes on Using the Multi-address $I^2C$

**This section describes precautions to take when using the multi-address $I^2C$ interface.**

## ■ Precaution in Setting the Multi-address $I^2C$ Interface Register

Before writing to the bus control register (MBCR), the multi-address $I^2C$ interface must be enabled (MCCR: EN).

When the master slave selection bit (MBCR: MSS) is set, transfer starts.

## ■ Precaution in Setting Shift Clock Frequency

To calculate the shift clock frequency using the $F_{sck}$ expression (1) in Table 16.4-3 "Function of Each Bit in Multi-address $I^2C$ Clock Control Register (MCCR)", it is necessary to know the values of m, n, and DMBP.

When the value of m is 5 (MCCR: CS4 = CS3 =0) and the value of n is 4 (MCCR: CS2 = CS1 = CS0 = 0), "DMBP = 1" cannot be selected. Other combinations do not present a problem.

## ■ Precaution on the Priority at Simultaneous Writing

- Contention of the next byte transfer and stop condition

  When "0" is written to MBCR: MSS in states where MBCR: INT is cleared, the MSS bit has a higher priority and a stop condition is generated.

- Contention of the next byte transfer and start condition

  When "1" is written to MBCR: SCC in states where MBCR: INT is cleared, the SCC bit has a higher priority and a start condition is generated.

## ■ Precaution on Setting with Software

Do not select the repeated start condition (MBCR: EN = 0) and the slave mode (MBCR: MSS = 0) at the same time.

In states where the interrupt request flag bits (BER and INT in the MBCR register) are set to "1" and the interrupt request enable bits are enabled (BEIE and INTE in the MBCR register are set to "1"), recovery from the interrupt processing cannot be performed. Clear the BER and INT bits in the MBCR register.

When the multi-address $I^2C$ operation is not permitted (MCCR: EN = 0), all bits of the bus status register MBSR and the bus control register MBCR (excluding the bus error BER bit and the bus error enable BEIE bit) are cleared.

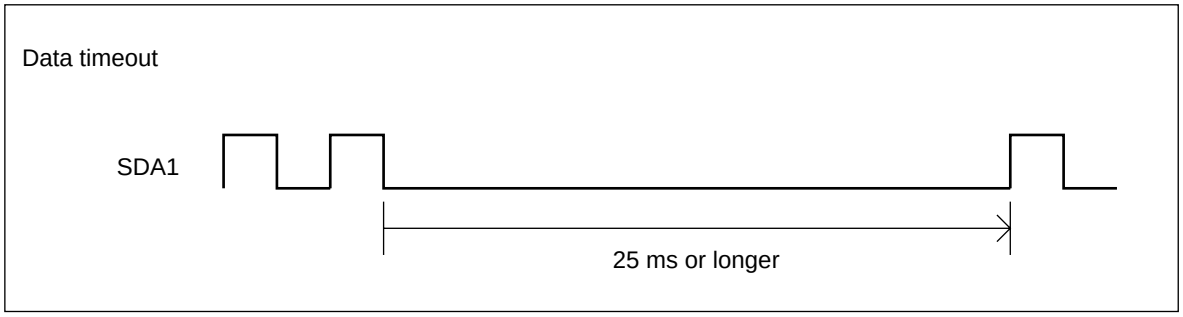
# 16.8 Operation of the Timeout Detection Function

**This section describes the operation of the timeout detection function when it is used as the SM bus.**

## ■ Data Timeout

When the "L" period of the SDA1 line exceeds 25 ms, this state is defined as a data timeout. The low time period is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the timeout data register (MTOD) + 1", a timeout is detected and the timeout data interrupt request flag (MTSR: TDR) is set.

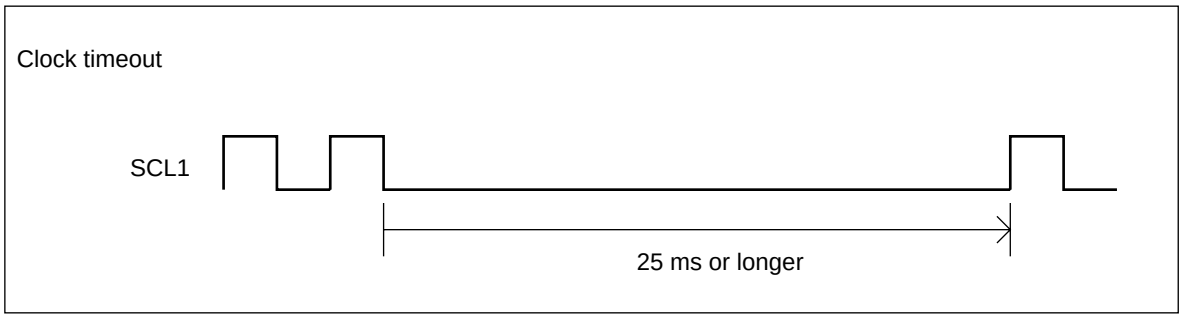
**Figure 16.8-1 Data Timeout**




## ■ Clock Timeout

When the "L" period of the SCL1 line exceeds 25 ms, this state is defined as a clock timeout. The low-time period is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the timeout clock register (MTOC) + 1", a timeout is detected and the timeout clock interrupt request flag (MTSR: TCR) is set.

**Figure 16.8-2 Clock Timeout**

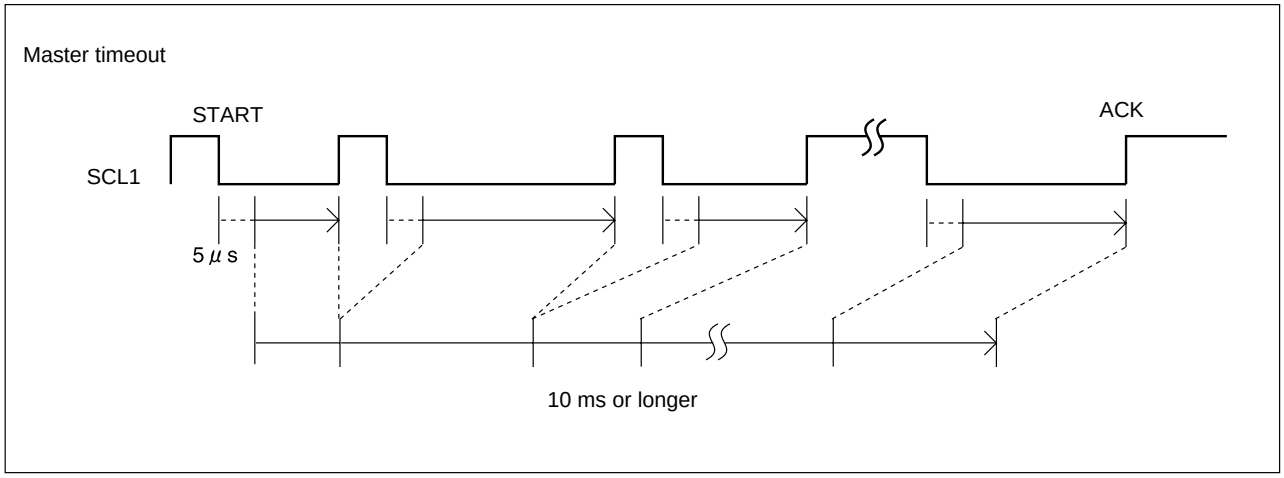


## ■ Master Timeout

When the cumulative "L" period of the SCL1 line between one byte data (START to ACK, ACK to ACK, ACK to STOP) exceeds 10 ms in master mode, this state is defined as a master timeout.

The low-time period is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 10. When the counter value matches "the value set in the master timeout register (MMTO) + 1", a master timeout is detected and the master timeout interrupt request flag (MTSR: MTR) is set.

**Figure 16.8-3 Master Timeout**

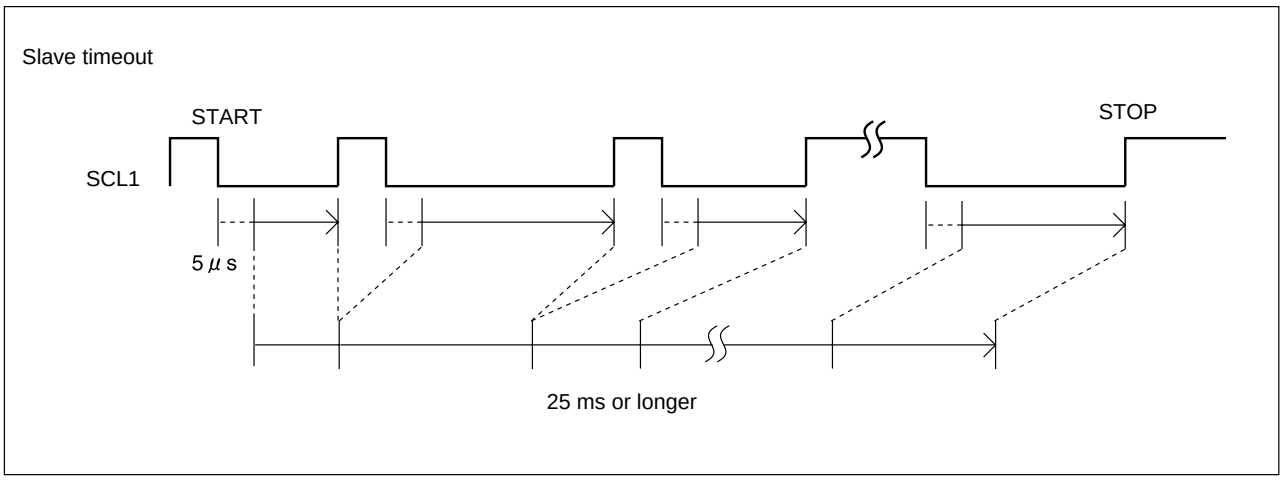


## ■ Slave Timeout

When the cumulative "L" period in the SCL1 line between START and STOP exceeds 10 ms in slave mode, this state is defined as a slave timeout.

The low-time period is counted with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) (T) divided by 10 and the counter is incremented by the clock further divided by 20. When the counter value matches "the value set in the slave timeout register (MSTO) + 1", a slave timeout is detected and the slave timeout interrupt request flag (MTSR: STR) is set.

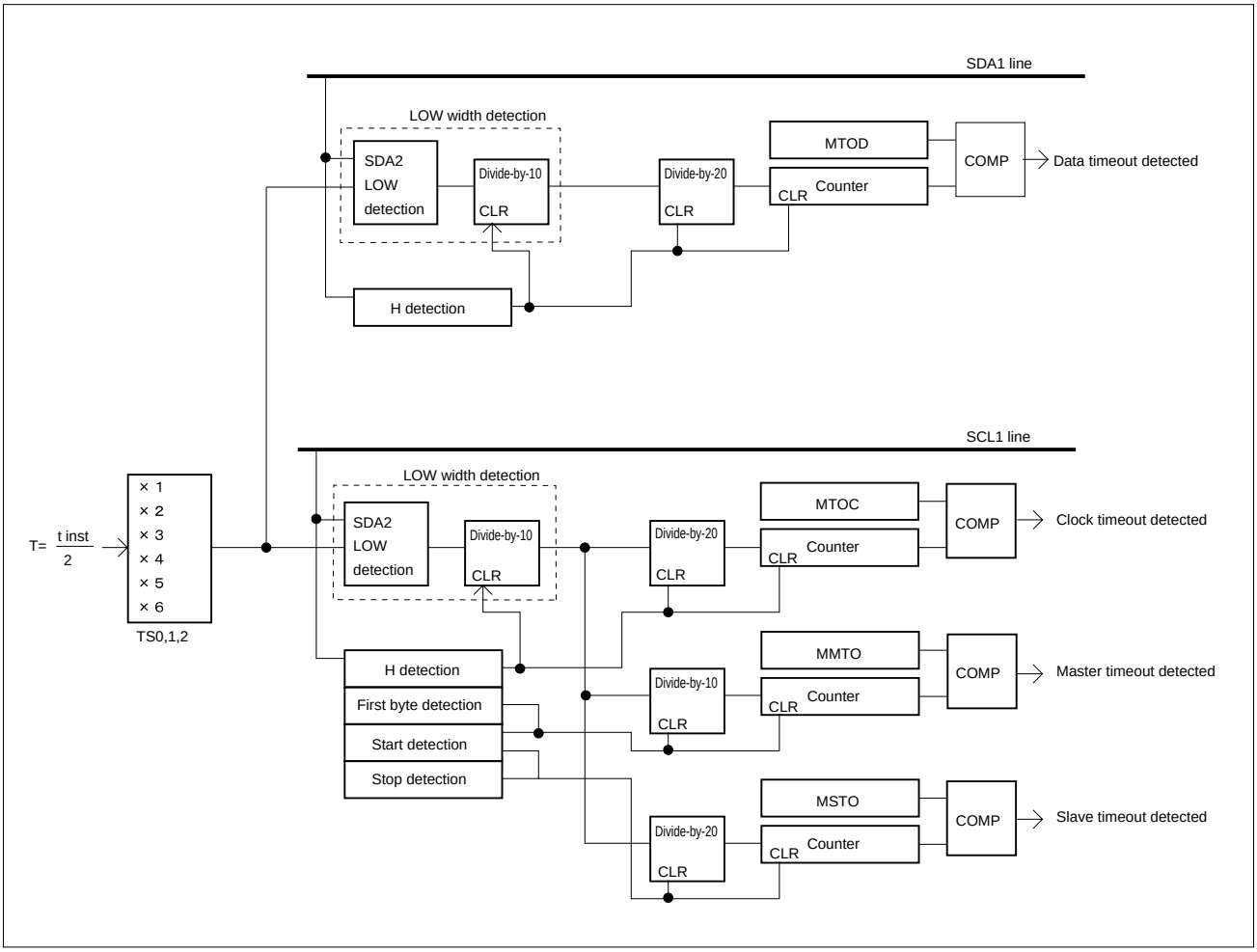
**Figure 16.8-4 Slave Timeout**



## ■ Timeout Clock Supply Block

Figure 16.8-5 "Timeout Clock Supply Block" shows the clock supply block of timeout.

**Figure 16.8-5 Timeout Clock Supply Block**




## ■ Errors

Timeout detection is performed with the clock selected with the timeout count clock detection bits (MTCR: TS0 to TS2) (T) divided by 10. Therefore, the following errors occur in the detection of an "L" width when, for example, T = 0.5 µs.

- Detects an "L" width of 5 to 5.5 µs or more in duration. (The sampling cycle is 0.5 µs)
- When the count is stopped (when low width detection is completed), an error of up to -5.5 µs in duration occurs. (In a cumulative count, errors are also accumulated.)

The counter for detecting a timeout is incremented with an L width or cumulative L width of 100 µs or more (50 µs or more for a master timeout).

# CHAPTER 17 BRIDGE CIRCUIT

**This chapter describes the functions and operations of the bridge circuit.**


17.1 "Overview of the Bridge Circuit"

17.2 "Configuration of the Bridge Circuit"

17.3 "Pins of the Bridge Circuit

17.4 "Registers of the Bridge Circuit"

# 17.1 Overview of the Bridge Circuit

**The bridge circuit is used to switch the I/O path of the $I^2C$/UART.**

## ■ Bridge Circuit

### ❍ Selection of "$I^2C$", multi-address $I^2C$, or "UART"

The bridge circuit can switch the I/O path of each port to "$I^2C$", "multi-address $I^2C$, and "UART".

Table 17.1-1 "Ports and Target Units that can be Selected by the Bridge Circuit" lists the ports and target units that can be selected by the bridge circuit.

**Table 17.1-1 Ports and Target Units that can be Selected by the Bridge Circuit**

| Ports that can be selected by the bridge circuit | Target Units |
|---|---|
| P33/SCL2/UCK3<br>P34/SDA2/UI3<br>(P35/U03) | $I^2C$ |
| | UART |
| | Multi-address $I^2C$ |
| P40/SCL3/UCK1<br>P41/SDA3/UI1<br>(P65/U01) | $I^2C$ |
| | UART |
| | Multi-address $I^2C$ |
| P42/SCL4/UCK2<br>P43/SDA4/UI2<br>(P64/U02) | $I^2C$ |
| | UART |
| | Multi-address $I^2C$ |

### ❍ Bypass with P30 - P31

In the bridge circuit, P30/SCL1 - P31/SDA1 can be bypassed with other buses (P33/SCL2 - P34/SDA2, P40/SCL3 - P41/SDA3, and P42/SCL4 - P43/SDA4).

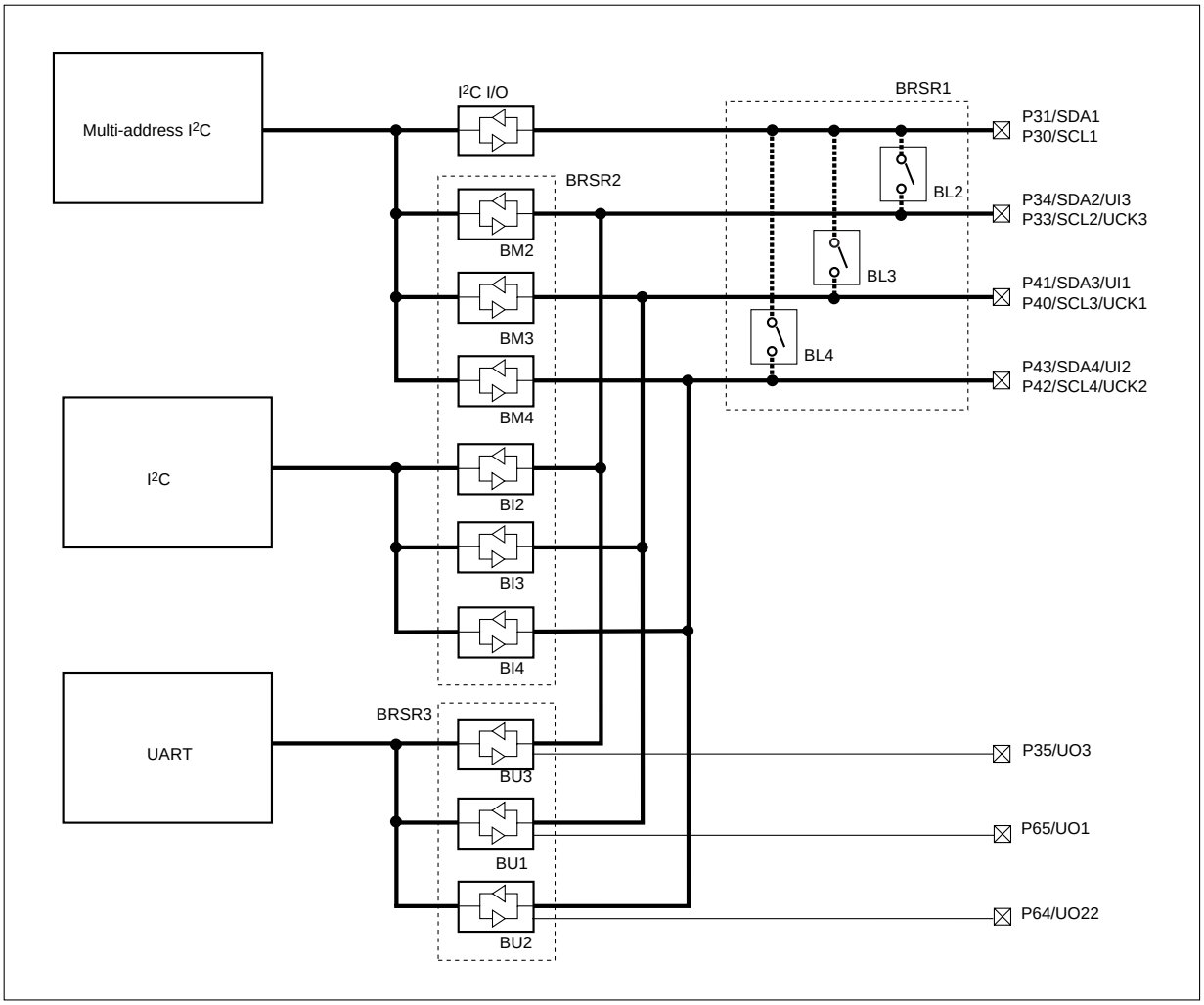
# 17.2 Configuration of the Bridge Circuit

**The bridge circuit consists of the following blocks.**

- **Bridge circuit selection registers (BRSR1 to 3)**

## ■ Bridge Circuit Block Diagram

**Figure 17.2-1 Bridge Circuit Block Diagram**




❍ **Bridge circuit selection registers 1 to 3 (BRSR 1 to 3)**

These registers are used for switching the bridge circuit.

# 17.3 Pins of the Bridge Circuit

**This section describes the pins related to the bridge circuit. It also shows a block diagram of pins.**

## ■ Pins Related to the Bridge Circuit

The pins related to the bridge circuit are as follows:

P33/SCL2/UCK3, P34/SDA2/UI3, P35/UO3, P40/SCL3/UCK1, P41/SDA3/UI1, P65/UO1, P42/SCL4/UCK2, P43/SDA4/UI2, and P64/UO2 pins.

### P33/SCL2/UCK3, P34/SDA2/UI3, and P35/UO3 pins

Each of these pins has multiple functions. Each pin can act as a general-purpose I/O-dedicated port (P33, P34, P35), $I^2C$ I/O pin (SCL2, SDA2), or UART/SIO I/O pin (UCK3, UI3, UO3).

The pin status can be read directly from the port data register (PDR3).

### P40/SCL3/UCK1, P41/SDA3/UI1, and P65/UO1 pins

Each of these pins has multiple functions. Each pin can act as a general-purpose I/O-dedicated port (P40, P41, P65), $I^2C$ I/O pin (SCL3, SDA3), or UART/SIO I/O pin (UCK1, UI1, UO1).

The pin status can be read directly from the port data registers (PDR4 and PDR6).

### P42/SCL4/UCK2, P43/SDA4/UI2, and P64/UO2 pins

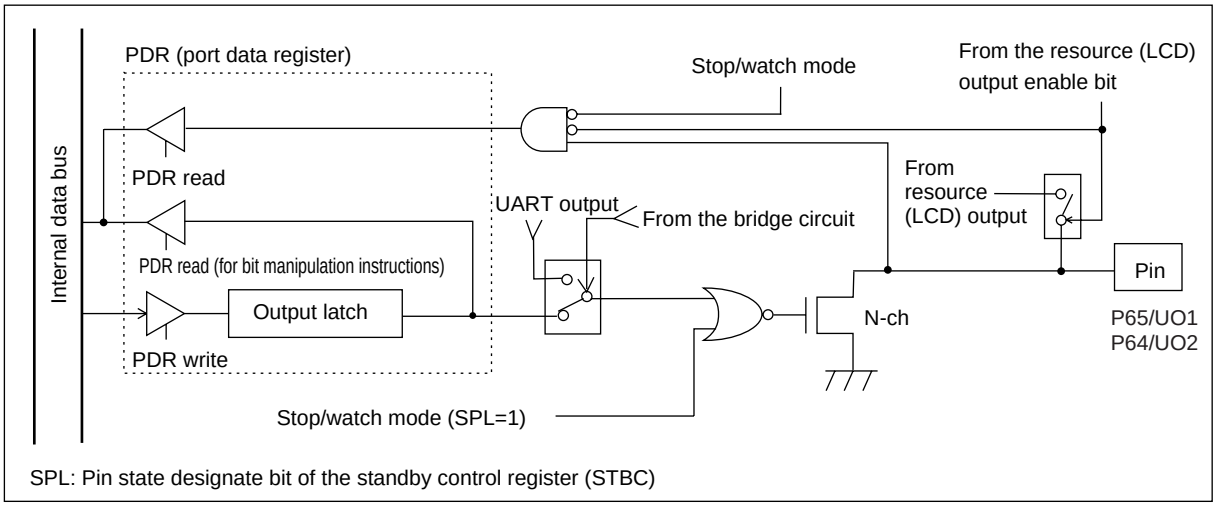
Each of these pins has multiple functions. Each pin can act as a general-purpose I/O-dedicated port (P42, P43, P64), $I^2C$ I/O pin (SCL4, SDA4), or UART/SIO I/O pin (UCK2, UI2, UO2).

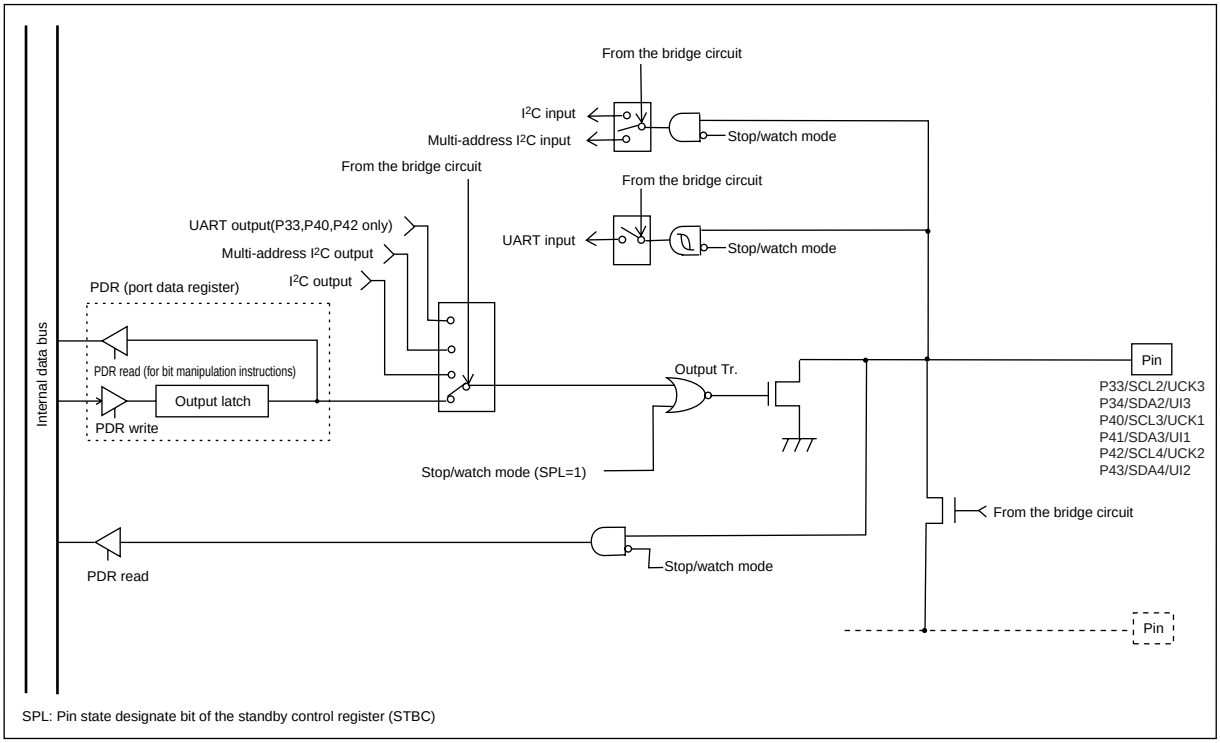
The pin status can be read directly from the port data registers (PDR4 and PDR6).

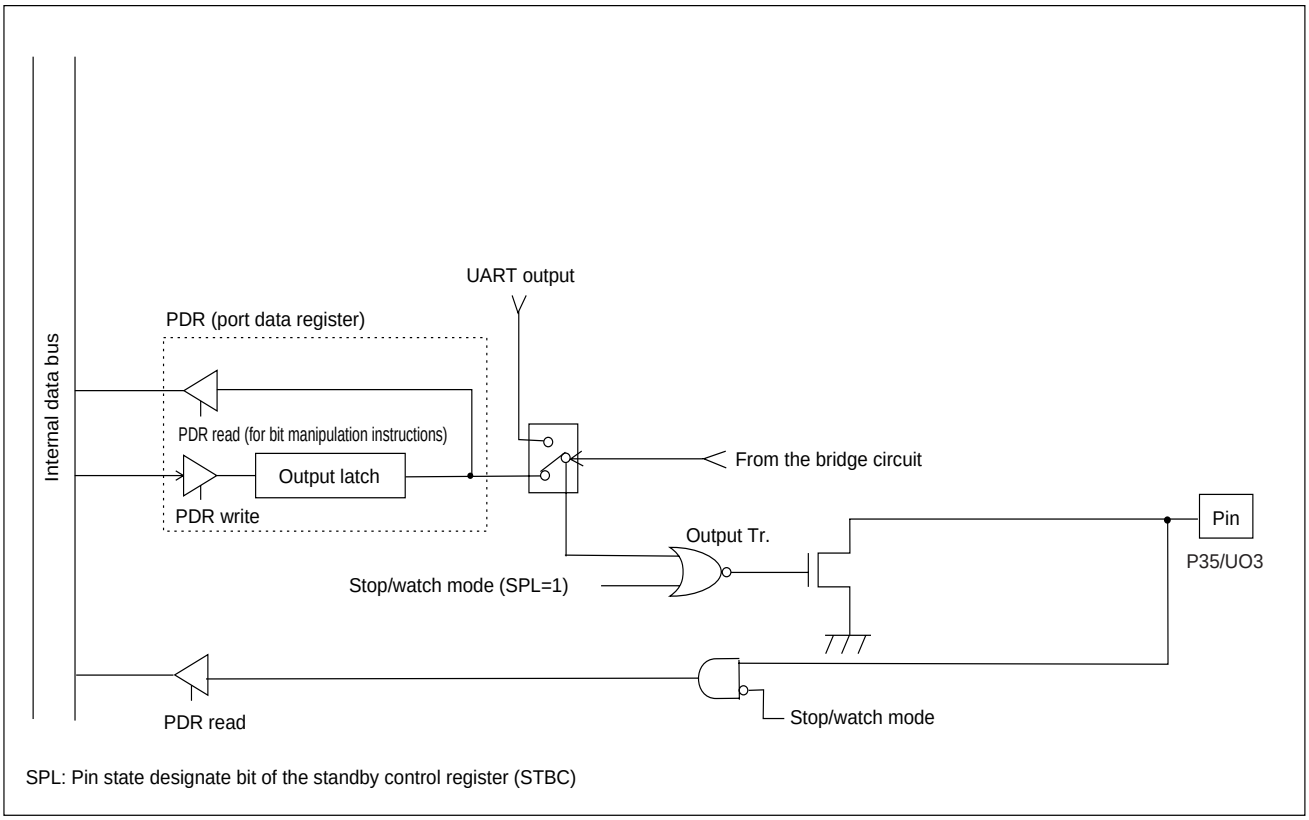
Figure 17.3-1 "A Block Diagram of the Pins Related to the Bridge Circuit" shows the pins related to the bridge circuit.

## ■ Block Diagram of Pins Related to the Bridge Circuit

**Figure 17.3-1 A Block Diagram of the Pins Related to the Bridge Circuit**






SPL: Pin state designate bit of the standby control register (STBC)

# 17.4 Registers of the Bridge Circuit

**This section shows the registers related to the bridge circuit.**

## ■ Registers Related to Bridge Circuit

**Figure 17.4-1 Registers Related to Bridge Circuit**

BRSR1 (Bridge circuit selection register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 5 C H | - | - | - | - | - | BL4 | BL3 | BL2 | XXXXX000B |
| | | | | | | R/W | R/W | R/W | |

BRSR2 (Bridge circuit selection register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 5 D H | — | — | BM4 | BI4 | BM3 | BI3 | BM2 | BI2 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

BRSR3 (Bridge circuit selection register 3)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 1 9 H | - | - | - | - | - | BU3 | BU2 | BU1 | XXXXX001B |
| | | | | | | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

# 17.4.1 Bridge Circuit Selection Register 1 ( BRSR1)

**Bridge circuit selection register 1 (BRSR1) is used to control bypass between the pins by the bridge circuit.**

## ■ Bridge Circuit Selection Register 1 (BRSR1)

**Figure 17.4-2 Bridge Circuit Selection Register 1 (BRSR1)**

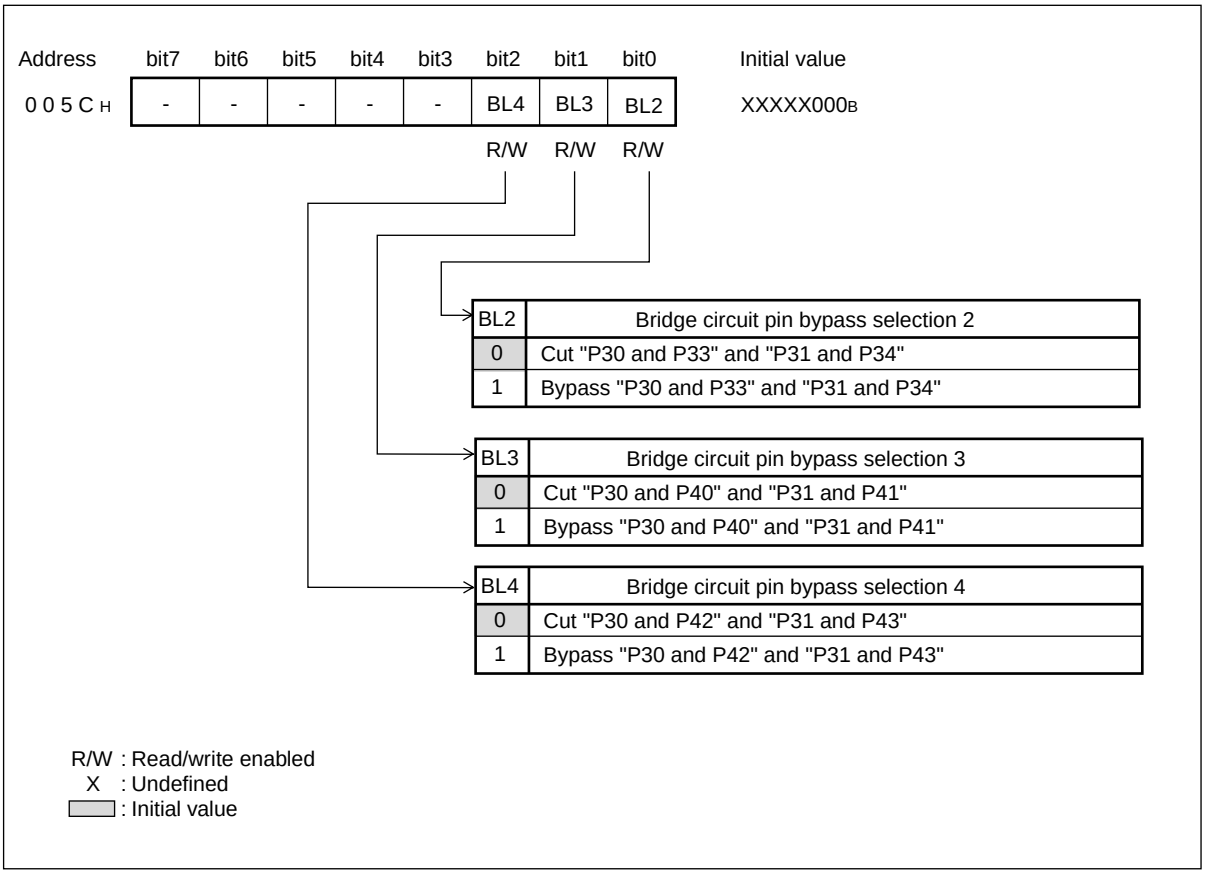



**Table 17.4-1 Bridge Circuit Register 1 (BRSR1) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4<br>Bit 3 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |

**Table 17.4-1 Bridge Circuit Register 1 (BRSR1) Bit Functions (Continued)**

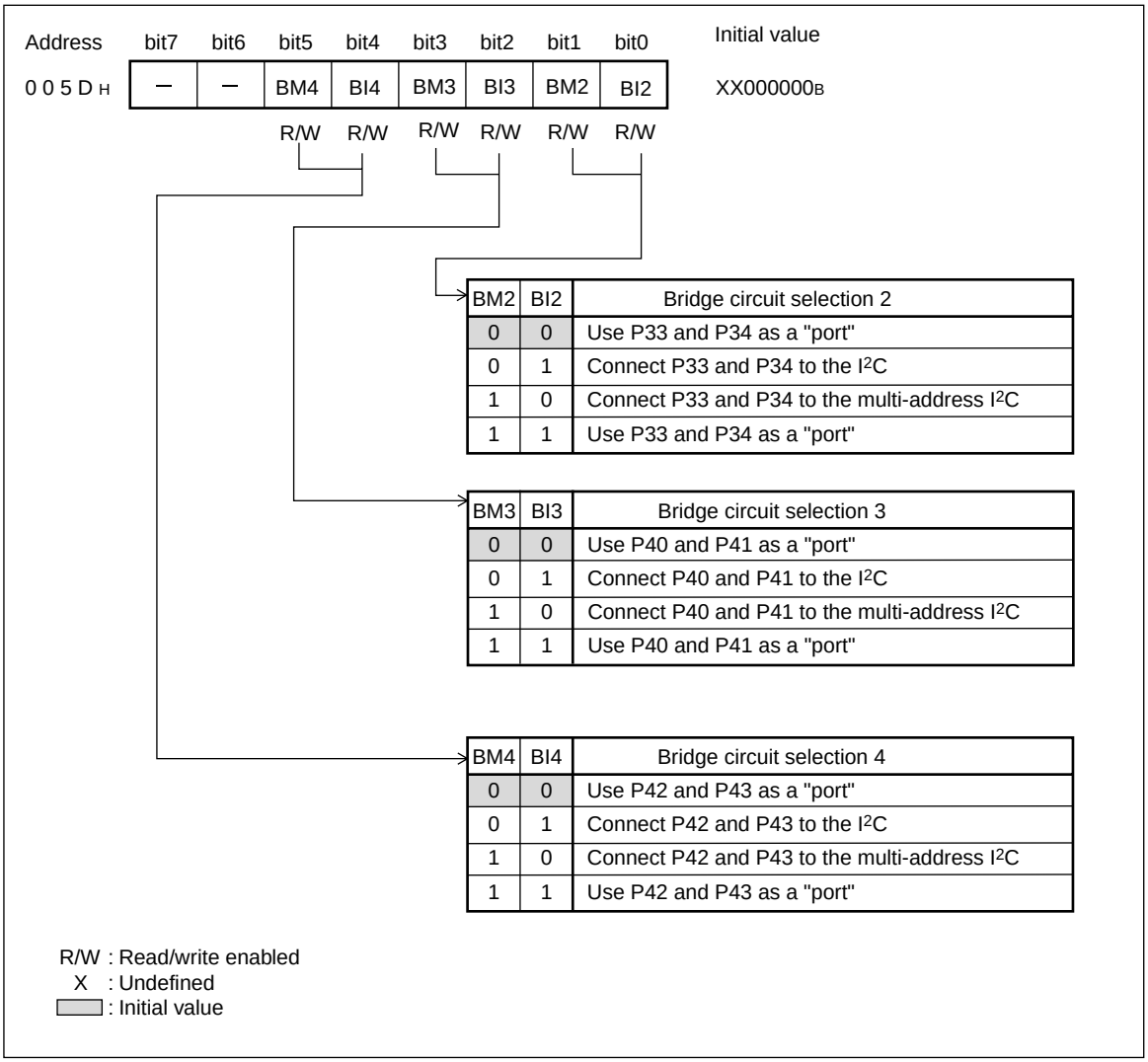
| Bit name | | Function |
|---|---|---|
| Bit 2 | BL4:<br>Bridge circuit pin bypass selection 4 | This bit controls bypass setting for "P30/SCL1 and P42/SCL4" and "P31/<br>SDA1 and P43/SDA4".<br>When "1" is written in this bit, bypass is set.<br>• P30/SCL1 = P42/SCL4/UCK2<br>• P31/SDA1 = P43/SDA4/UI2<br>When "0" is written in this bit, bypass is cut.<br>• P30/SCL1 not equal to P42/SCL4/UCK2<br>• P31/SDA1 not equal to P43/SDA4/UI2 |
| Bit 1 | BL3:<br>Bridge circuit pin bypass selection 3 | This bit controls bypass setting for "P30/SCL1 and P40/SCL3" and "P31/<br>SDA1 and P41/1SDA3".<br>When "1" is written in this bit, bypass is set.<br>• P30/SCL1 = P40/SCL3/UCK1<br>• P31/SDA1 = P41/1SDA3/UI1<br>When "0" is written in this bit, bypass is cut.<br>• P30/SCL1 not equal to P40/SCL3/UCK1<br>• P31/SDA1 not equal to P41/1SDA3/UI1 |
| Bit 0 | BL2:<br>Bridge circuit pin bypass selection 2 | This bit controls bypass setting for "P30/SCL1 and P31/SCL2" and "P31/<br>SDA1 and P34/SDA2".<br>When "1" is written in this bit, bypass is set.<br>• P30/SCL1=P33/SCL2/UCK3<br>• P31/SDA1=P34/SDA2/UI3<br>When "0" is written in this bit, bypass is cut.<br>• P30/SCL1 not equal to P33/SCL2/UCK3<br>• P31/SDA1 not equal to P34/SDA2/UI3 |

## 17.4.2 Bridge Circuit Selection Register 2 (BRSR2)

**Bridge circuit selection register 2 (BRSR2) controls the connection switching by the bridge circuit.**

### ■ Bridge Selection Register 2 (BRSR2)

**Figure 17.4-3 Bridge Circuit Selection Register 2 (BRSR2)**




| BM2 | BI2 | Bridge circuit selection 2 |
|---|---|---|
| 0 | 0 | Use P33 and P34 as a "port" |
| 0 | 1 | Connect P33 and P34 to the I²C |
| 1 | 0 | Connect P33 and P34 to the multi-address I²C |
| 1 | 1 | Use P33 and P34 as a "port" |

| BM3 | BI3 | Bridge circuit selection 3 |
|---|---|---|
| 0 | 0 | Use P40 and P41 as a "port" |
| 0 | 1 | Connect P40 and P41 to the I²C |
| 1 | 0 | Connect P40 and P41 to the multi-address I²C |
| 1 | 1 | Use P40 and P41 as a "port" |

| BM4 | BI4 | Bridge circuit selection 4 |
|---|---|---|
| 0 | 0 | Use P42 and P43 as a "port" |
| 0 | 1 | Connect P42 and P43 to the I²C |
| 1 | 0 | Connect P42 and P43 to the multi-address I²C |
| 1 | 1 | Use P42 and P43 as a "port" |

**Table 17.4-2 Bridge Circuit Selection Register 2 (BRSR2) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 5 | BM4:<br>Bridge circuit multi-$I^2C$ selection 4 | This bit specifies options for P42/SCL4/UCK2 and P43/SDA4/UI2 whether they are connected to the "multi-address $I^2C$."<br>When "1" is written in this bit, P42/SCL4 and P43/SDA4 are connected to the "multi-address $I^2C$."<br>When this bit is "0" or BM4=BI4=1, the port function is selected. |
| Bit 4 | BI4:<br>Bridge circuit $I^2C$ selection 4 | This bit specifies options for P42/SCL4/UCK2 and P43/SDA4/UI2 whether they are connected to the "$I^2C$."<br>When "1" is written in this bit, P42/SCL4 and P43/SDA4 are connected to the "$I^2C$."<br>When this bit is "0" or BM4=BI4=1, the port function is selected. |
| Bit 3 | BM3:<br>Bridge circuit multi-$I^2C$ selection 3 | This bit specifies options for P40/SCL3/UCK1 and P41/SDA3/UI1 whether they are connected to the "multi-address $I^2C$."<br>When "1" is written in this bit, P40/SCL3 and P41/SDA3 are connected to the "multi-address $I^2C$."<br>When this bit is "0" or BM3=BI3=1, the port function is selected. |
| Bit 2 | BI3:<br>Bridge circuit $I^2C$ selection 3 | This bit specifies options for P40/SCL3/UCK1 and P41/SDA3/UI1 whether they are connected to the "$I^2C$."<br>When "1" is written in this bit, P40/SCL3 and P41/SDA3 are connected to the "$I^2C$."<br>When this bit is "0" or BM3=BI3=1, the port function is selected. |
| Bit 1 | BM2:<br>Bridge circuit multi-$I^2C$ selection 2 | This bit specifies options for P33/SCL2/UCK3 and P34/SDA2/UI3 whether they are connected to the "multi-address $I^2C$."<br>When "1" is written in this bit, P33/SCL2 and P34/SDA2 are connected to the "multi-address $I^2C$."<br>When this bit is "0" or BM2=BI2=1, the port function is selected. |
| Bit 0 | BI2:<br>Bridge circuit $I^2C$ selection 2 | This bit specifies options for P33/SCL2/UCK3 and P34/SDA2/UI3 whether they are connected to the "$I^2C$."<br>When "1" is written in this bit, P33/SCL2 and P34/SDA2 are connected to the "$I^2C$."<br>When this bit is "0" or BM2=BI2=1, the port function is selected. |

**Note:**

Switching of the connection destination by this register is valid only when the resource function of the connection destination has been allowed. Be careful not to duplicate the port specification for switching the connection destination between this register and the BRSR3 register.

# 17.4.3 Bridge Circuit Selection Register 3 (BRSR3)

**Bridge circuit selection register (BRSR3) controls connection switching by the bridge circuit.**

## ■ Bridge Circuit Selection Register 3 (BRSR3)

**Figure 17.4-4 Bridge Circuit Selection Resister 3(BRSR3)**

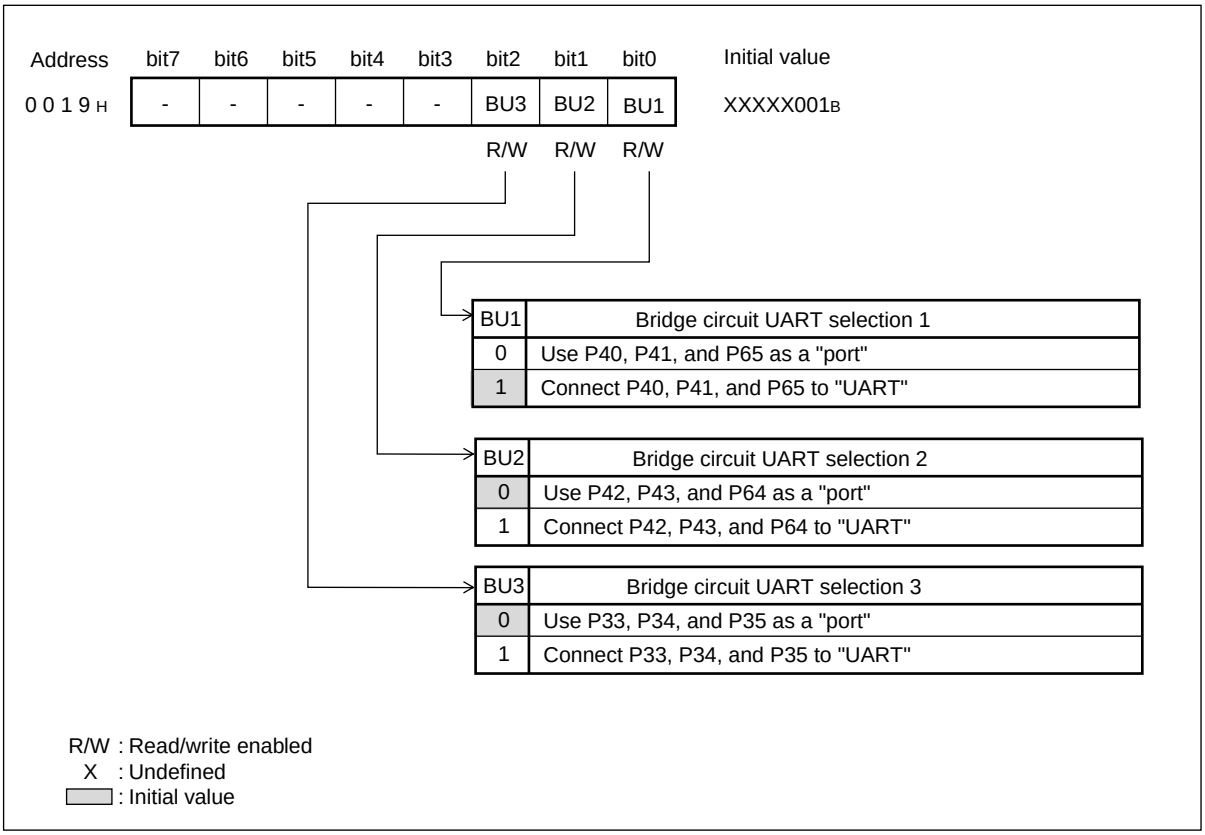


**Table 17.4-3 Bridge Circuit Selection Register 3 (BRSR3) Bit Functions**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6<br>Bit 5<br>Bit 4<br>Bit 3 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 2 | BU3:<br>Bridge circuit UART selection 3 | This bit specifies options for "P33/SCL2/UCK3, P34/SDA2/UI3, and P35/UO3" whether they are connected to UART.<br>When "1" is written in this bit, UART is selected.<br>• UCK3 (P33): Acts as UCK only when UART serial clock output is allowed (SMC2:SCKE = 1).<br>• UI3 (P34)<br>• UO3 (P35): Acts as UO only when UART serial data output is allowed (SMC2:TXOE = 1).<br>When "0" is written in this bit, the port function is selected. |
| Bit 1 | BU2:<br>Bridge circuit UART selection 2 | This bit specifies options for "P42/SCL4/UCK2, P43/SDA4/UI2, and P64/UO2" whether thy are connected to UART.<br>When "1" is written in this bit, UART is selected.<br>• UCK2 (P42): Acts as UCK only when UART serial clock output is allowed (SMC2:SCKE = 1).<br>• UI2 (P43)<br>• UO2 (P64): Acts as UO only when UART serial data output is allowed (SMC2:TXOE = 1).<br>When "0" is written in this bit, the port function is selected. |
| Bit 0 | BU1:<br>Bridge circuit UART selection 1 | This bit specifies options for "P40/SCL3/UCK1, P41/SDA3/UI1, and P65/UO1" whether they are connected to UART.<br>When "1" is written in this bit, UART is selected.<br>• UCK1 (P40): Acts as UCK only when UART serial clock output is allowed (SMC2:SCKE = 1).<br>• UI1 (P41)<br>• UO1 (P65): Acts as UO only when UART serial data output is allowed (SMC2:TXOE = 1).<br>When "0" is written in this bit, the port function is selected. |

**Note:**

Switching of the connection destination is valid only when the resource function of the connection destination has been allowed. Be careful not to duplicate the port specification for switching the connection destination between this register and the BRSR3 register.

# CHAPTER 18 LCD CONTROLLER DRIVER

**This chapter describes the functions and operations of the LCD controller driver.**


18.1 "Overview of the LCD Controller Driver"

18.2 "Configuration of the LCD Controller Driver"

18.3 "Pins of the LCD Controller Driver"

18.4 "Registers of the LCD Controller Driver"

18.5 "LCD Display RAM in the LCD Controller Driver"

18.6 "Operation of the LCD Controller Driver"

# 18.1 Overview of the LCD Controller Driver

**The LCD Controller driver, which contains 8-byte display data memory, controls the LCD display with four common output signals and 14 segment output signals. By selecting from three types of duty output, the LCD panel can be driven directly.**

## ■ LCD Controller Driver Functions

The LCD controller driver has a function for displaying the description of the display data memory (LCD display RAM) on the LCD panel directly with the segment output pins and the common output pins.

- The LCD controller driver contains voltage dividing resistors for LCD driving. The external dividing resistors can also be connected to the LCD controller driver.
- Up to four common output pins (COM0 to COM3) and 14 segment output pins (SEG0 to SEG13) can be used.
- The LCD controller driver contains a 7-byte (14 x 4 bit) LCD display RAM.
- As duty, 1/2, 1/3, or 1/4 can be selected. (It is restricted by bias setting.)
- As a driving clock, a main clock or a subclock can be selected.
- The LCD can be driven directly.

Table 18.1-1 "Combinations of Bias and Duty" shows available combinations of bias and duty.

**Table 18.1-1 Combinations of Bias and Duty**

| Bias | 1/2 duty | 1/3 duty | 1/4 duty |
|---|---|---|---|
| 1/2 bias | Yes | No | No |
| 1/3 bias | No | Yes | Yes |

Yes: Recommended mode
No: Unavailable

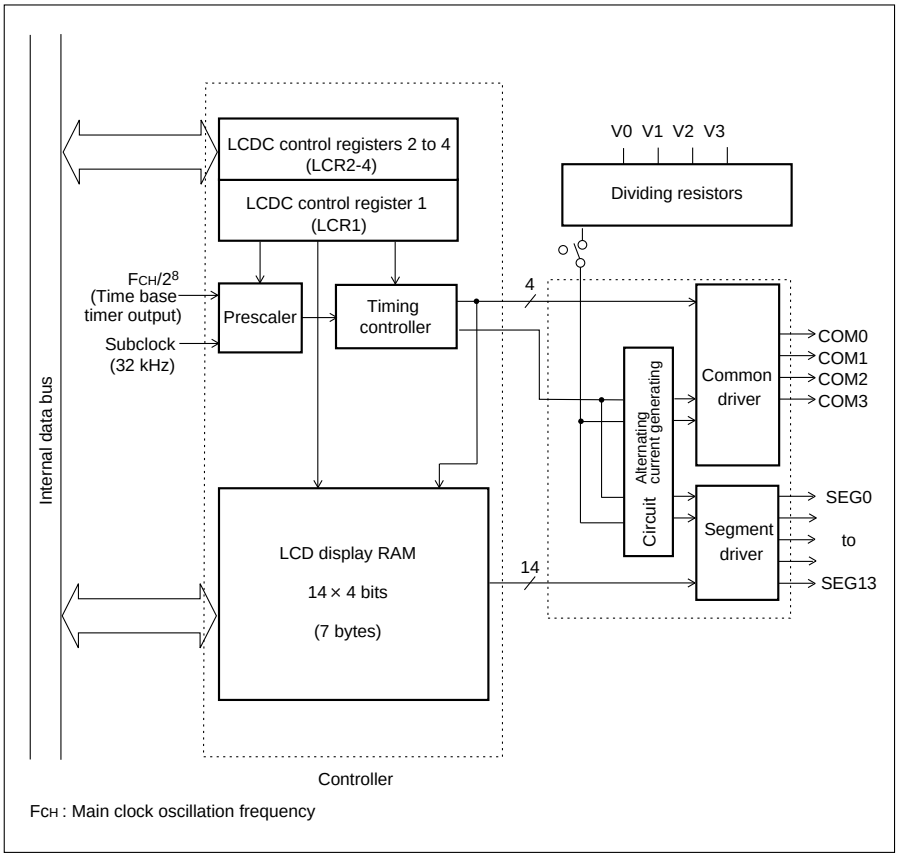
# 18.2 Configuration of the LCD Controller Driver

**The LCD controller driver consists of the following eight blocks, which can be divided into two sections in terms of functions: the controller section that generates segment and common signals in accordance with the description in the LCD display RAM and the driver section that drives the LCD.**

- **LCDC control register 1 (LCR1)**
- **LCD display RAM**
- **Prescaler**
- **Timing controller**
- **Alternating current generating circuit**
- **Common driver**
- **Segment driver**
- **Dividing resistors**

## ■ LCD Controller Driver Block Diagram

**Figure 18.2-1 LCD Controller Driver Block Diagram**



- **LCDC control register 1 (LCR1)**

  This register selects a clock for generating frame cycles (driving duty), controls the LCD driving power supply, selects display or display blanking, selects the display mode, and selects the cycle of the LCD clock (duty driving).

- **LCDC display RAM**

  A 16 x 4 bit RAM for generating segment output signals. The description in this RAM is read in synch with the common signal selection timing automatically and is output from segment output pins.

- **Prescaler**

  The prescaler generates a frame frequency in accordance with the setting selected from two types of clocks and four types of frequencies.

- **Timing controller**

  The timing controller controls common signals and segment signals based on the frame frequency and the setting in the LCR1 register.

- **Alternating current generating circuit**

  This circuit generates an alternating current waveform for driving the LCD from the signals from the timing controller.

- **Common driver**

  A driver for the common pins in the LCD.

- **Segment driver**

  A driver for the segment pins in the LCD.

- **Dividing resistors**

  Resistors for generating the LCD driving voltage by dividing a voltage. The dividing resistors can also be connected externally.

## ■ Power Supply Voltage of the LCD Controller Driver

The power supply voltage of the LCD driver is set using the internal dividing resistors or by connecting dividing resistors to the pins V0 to V3.

# 18.2.1 Internal Dividing Resistors of the LCD Controller Driver

**The supply voltage of the LCD controller driver is generated by the internal dividing resistors. (The dividing resistors can also be connected externally.)**

## ■ Internal Dividing Resistors

The LCD controller driver contains internal dividing resistors. The external dividing resistors can also be connected to pins V0 to V3.

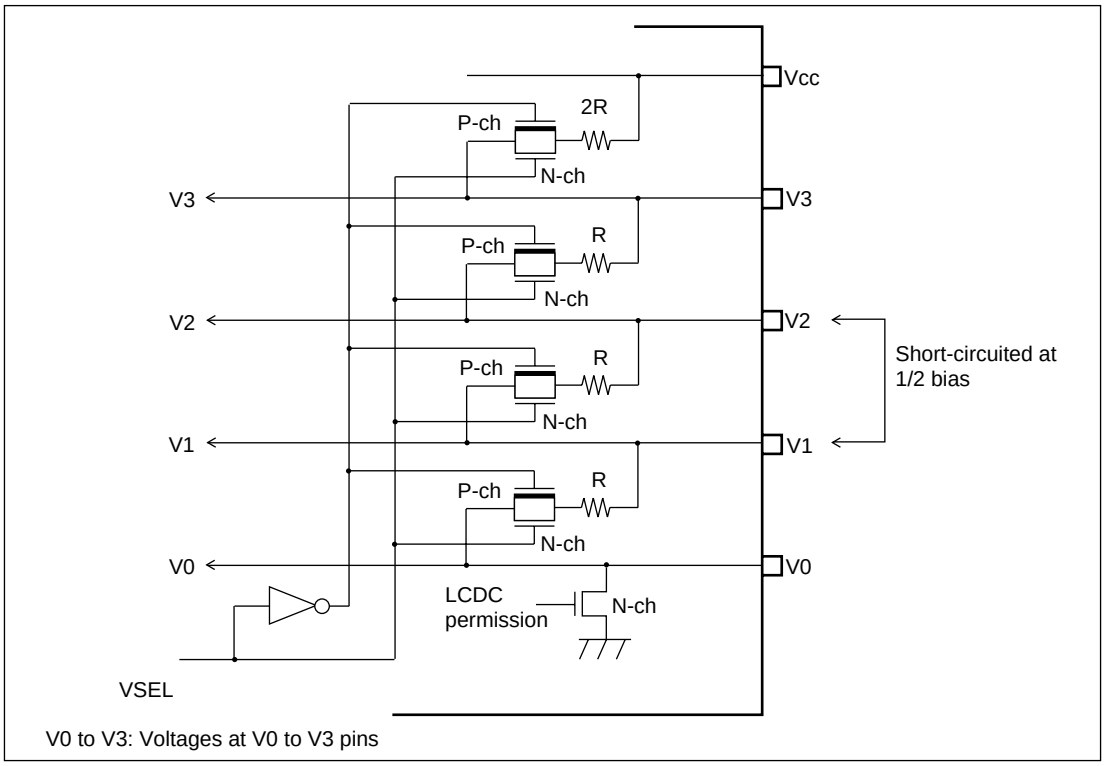
The internal dividing resistors or the external dividing resistors can be selected with the driving power supply control bit in the LCDC control register 1 (LCR1: VSEL). When the VSEL bit is set to "1," the internal dividing resistors are energized. To use only the internal dividing resistors without using external dividing resistors, set this bit to "1."

The LCD controller permission is inactivated at LCD operation stop (LCR1: MS1, MS0 = $00_B$) and in watch mode (STBC: TMD = 1) in states where operation in watch mode is prohibited (LCR: LCEN = 0).

To set 1/2 bias, short-circuit the V2 and V1 pins.

Figure 18.2-2 "Equivalent Circuit of the Internal Dividing Resistors" shows the equivalent circuit of the internal dividing resistors.

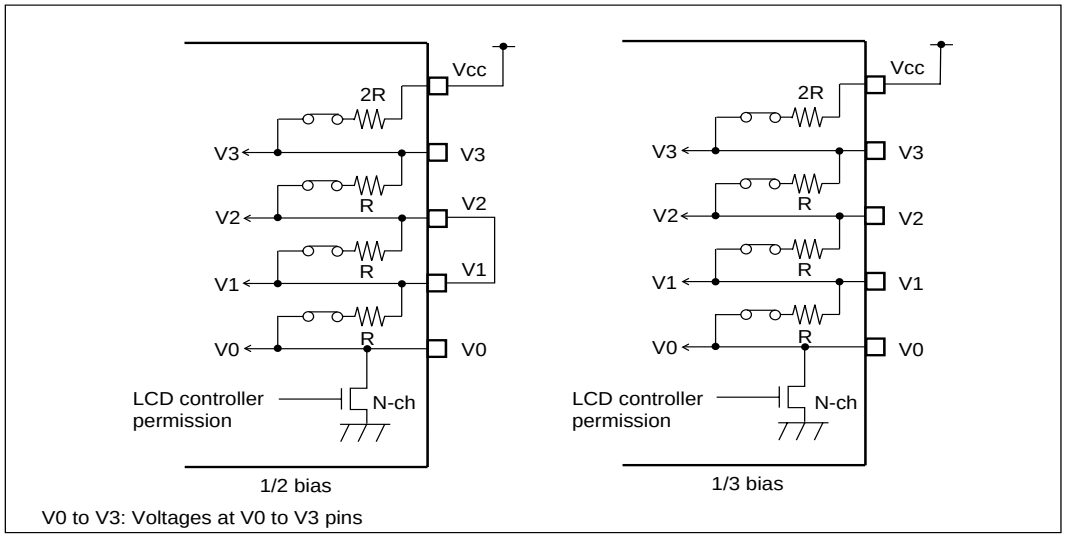
**Figure 18.2-2 Equivalent Circuit of the Internal Dividing Resistors**



## ■ Using the Internal Dividing Resistors

When internal dividing registers are used, the voltages at V1, V2, and V3 become lower by 2R because a resistor (2R) is included. Figure 18.2-3 "States in which the Internal Dividing Resistors are Used" shows the states in which the internal dividing resistors are used.

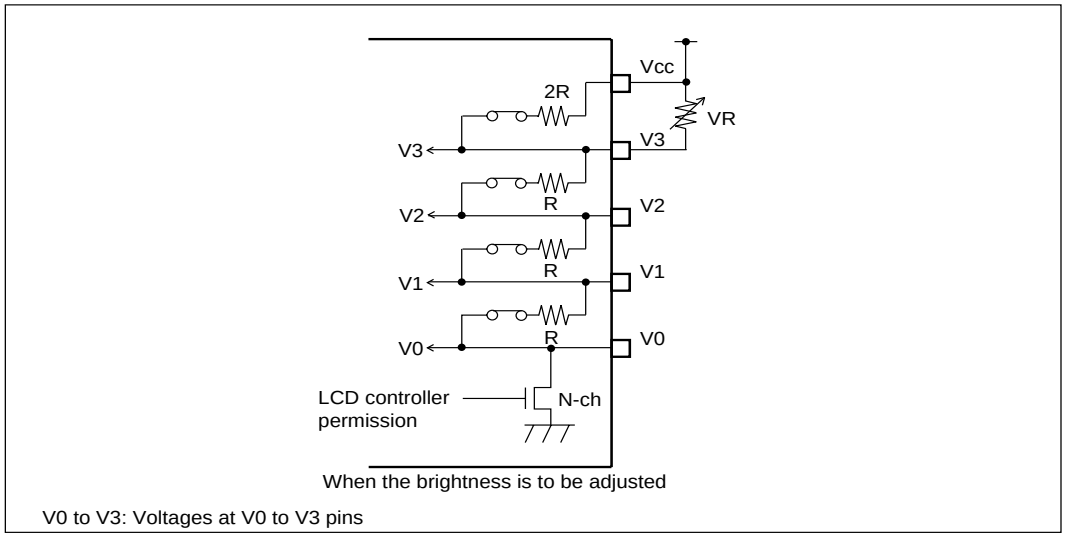
**Figure 18.2-3 States in which the Internal Dividing Resistors are Used**




## ■ Adjusting the Brightness of the LCD when the Internal Dividing Resistors are Used

If the brightness cannot be increased when the internal dividing resistors are used, connect a variable register (VR) externally (between Vcc and V3 pins) to adjust the voltage at 3V. Figure 18.2-4 "Brightness Adjustment when the Internal Dividing Resistors are Used" shows a connection example of the VR.

**Figure 18.2-4 Brightness Adjustment when the Internal Dividing Resistors are Used**




**Note:**

During LCD operation, the internal 2R is effective. Therefore, VR and 2R are connected in parallel.

## 18.2.2 External Dividing Resistor of LCD Controller Driver

**External dividing resistors can be connected to the pins V0 to V3.**
**By connecting a variable resistor between the Vcc and V3 pins, the brightness level can be adjusted.**

### ■ External Dividing Resistors

The external dividing resistors can be connected to the LCD driving power supply pins (V0 to V3) instead of using the internal dividing resistors. The connection of external dividing resistors based on the bias method and the LCD driving voltages are shown in Figure 18.2-5 "Connection Example of External Dividing Resistors" and Table 18.2-1 "Setting of LCD Driving Voltages", respectively.

**Figure 18.2-5 Connection Example of External Dividing Resistors**

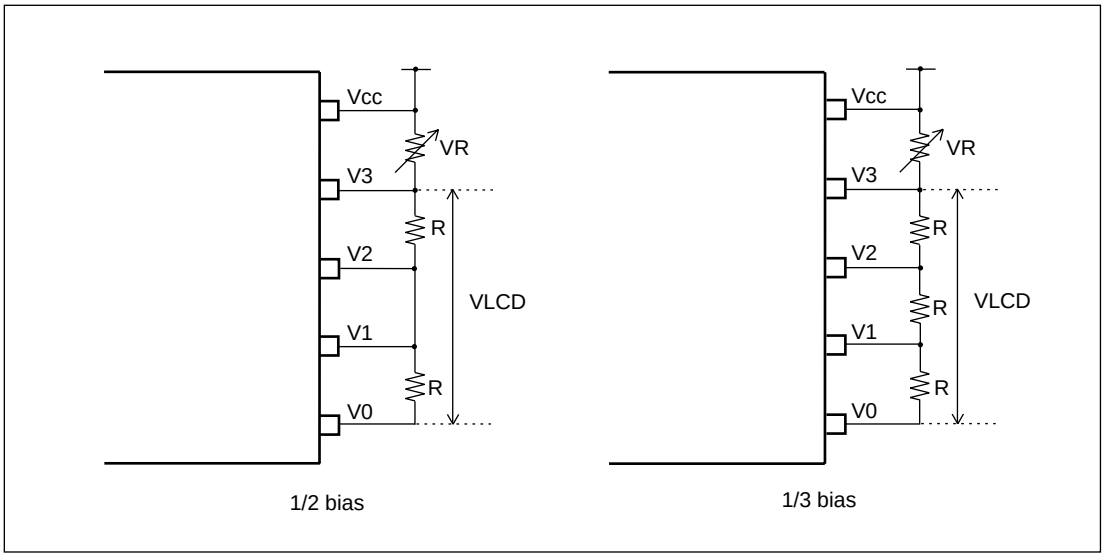



**Table 18.2-1 Setting of LCD Driving Voltages**

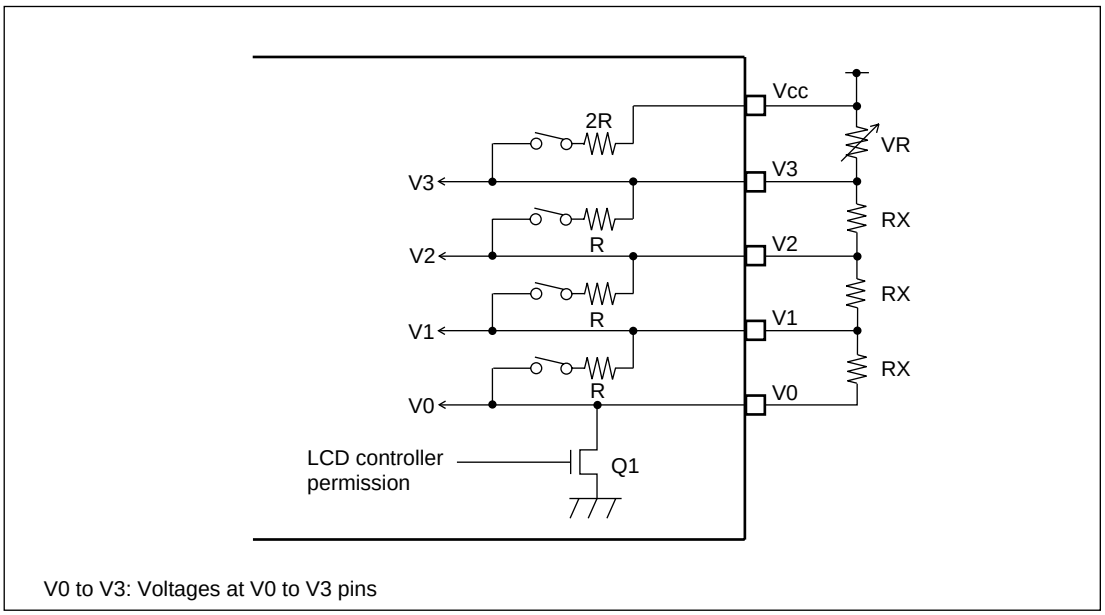
| | V3 | V2 | V1 | V0 |
|---|---|---|---|---|
| 1/2 bias | $V_{LCD}$ | $1/2V_{LCD}$ | $1/2V_{LCD}$ | GND |
| 1/3 bias | $V_{LCD}$ | $2/3V_{LCD}$ | $1/3V_{LCD}$ | GND |

V0 to V3: Voltages at V0 to V3 pins
$V_{LCD}$: Operating voltage of LCD

## ■ Using External Dividing Resistors

The V0 pin is internally connected to Vss (GND) through a transistor. Therefore, when external dividing resistors are used, the current flowing into the resistor can be shut off when the LCD controller driver is stopped by connecting the Vss of the dividing resistors to the V0 pin only. Figure 18.2-6 "State where External Dividing Resistors are Used" shows the external dividing resistors being used.

**Figure 18.2-6 State where External Dividing Resistors are Used**




V0 to V3: Voltages at V0 to V3 pins

1. To connect dividing resisters externally without being affected by the internal dividing resistors, the entire internal dividing resistors must first be separated by writing "0" to the driving voltage control bit (LCR: VSEL) in the LCD controller control register.
2. If a value other than "00B" is written to the display mode selection bits (MS1, MS0) in the LCR1 register in states where the internal dividing resistors are separated, the LCD controller enable transistor (Q1) is set to "ON" and current flows into the external dividing resistors.
3. If "00B" is written to the display mode selection bits (MS1, MS0) in the LCR1 register, the LCD controller enable transistor (Q1) is set to "OFF" and current does not flow into the external dividing resistors.

**Note:**

The RX connected externally differs depending on the LCD used. Select an appropriate resistance.

# 18.3 Pins of the LCD Controller Driver

**This section shows the pins related to the LCD controller driver and the block diagram of pins.**

## ■ Pins related to the LCD Controller Driver

The pins related to the LCD controller driver include four common output pins (COM0 to COM3), 14 segment output pins (SEG0 to SEG13), and four power supply pins for LCD driving (V0 to V3).

### ❍ PB4/COM0 to PB7/COM3 pins

The PB4/COM0 to PB7/COM3 pins serve as an I/O port (PB4 to PB7) and LCD common output pins (COM0 to COM3) that can be switched with the LCR4 register.

### ❍ PA0/SEG00 to PA7/SEG07 and P60/SEG08 to P65/SEG13 pins

The P60/SEG08 to P65/SEG13 pins serve as an I/O port (P60 to P65) and LCD segment output pins (SEG08 to SEG13) that can be switched with the LCR2 register. The PA0/SEG00 to PA7/SEG07 pins serve as I/O ports (PA0 to PA7) and LCD segment output pins (SEG00 to SEG07) that can be switched with the LCR3 register.

### ❍ PB0/V0 to PB3/V3 pins

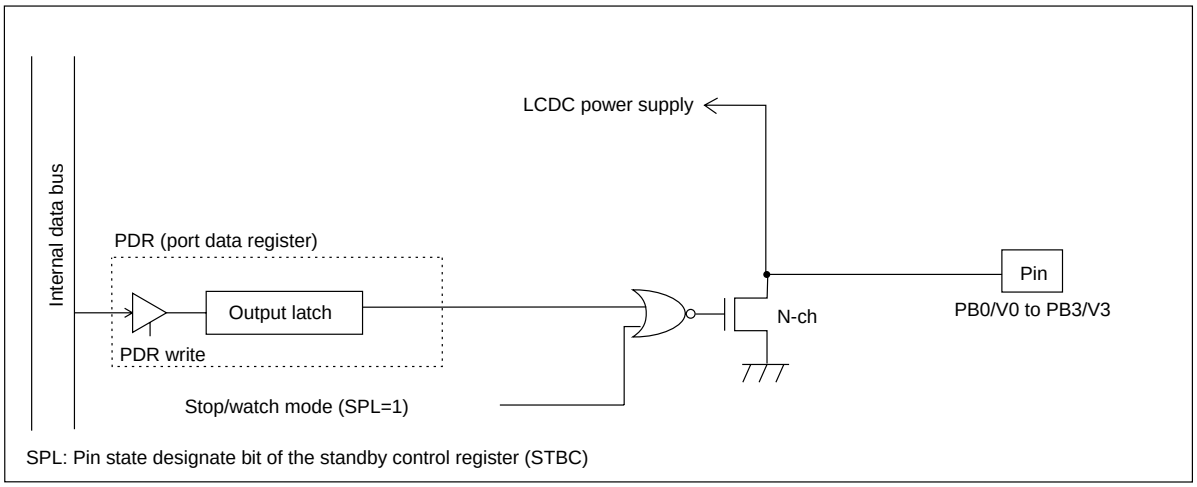
The PB0/V0 to PB3/V3 pins serve as power pins for LCD driving (V0 to V3) and output pins (PB0 to PB3) that can be switched with the LCR4 register.

**Note:**

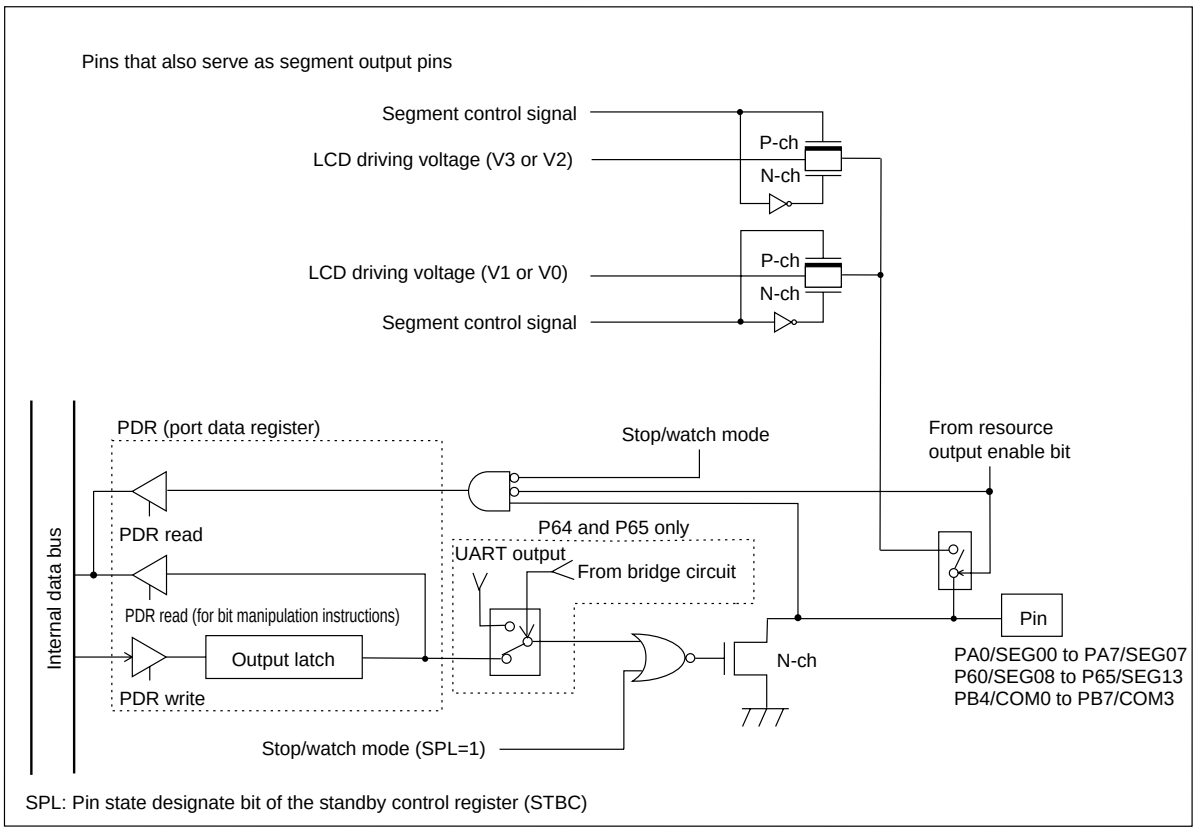
To use these bits as LCDC pins, set the corresponding bits to "LCD enabled" with the LCDC control registers (LCR2, LCR3, and LCR4) and set the output transistor to "OFF" by writing "1" to the corresponding port data registers (PDR6, PDRA, and PDRB).

## ■ Block Diagram of Pins Related to LCD Controller Driver

**Figure 18.3-1 Block Diagram of Pins Related to LCD Controller Driver**




SPL: Pin state designate bit of the standby control register (STBC)




SPL: Pin state designate bit of the standby control register (STBC)

# 18.4 Registers of the LCD Controller Driver

**This section shows the registers related to the LCD controller driver.**

## ■ Registers Related to LCD Controller Driver

**Figure 18.4-1 Registers Related to LCD Controller Driver**

LCR1 (LCDC control register 1)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 5 E H | CSS | LCDEN | VSEL | BK | MS1 | MS0 | FP1 | FP0 | 00010000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

LCR2 (LCDC control register 2)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 5 F H | - | SEG13 | SEG12 | SEG11 | SEG10 | SEG9 | SEG8 | - | X000000XB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | | |

LCR3 (LCDC control register 3)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 1 6 H | SEG7 | SEG6 | SEG5 | SEG4 | SEG3 | SEG2 | SEG1 | SEG0 | 00000000B |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

LCR4 (LCDC control register 4)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0 0 1 8 H | COM3 | COM2 | COM1 | COM0 | - | - | - | - | 0000XXXXB |
| | R/W | R/W | R/W | R/W | | | | | |

R/W : Read/write enabled

# 18.4.1 LCDC Control Register 1 (LCR1)

**The LCDC control register 1 (LCR1) selects a clock for generating frame cycles, controls the power supply for LCD driving, selects display or display blanking, selects the display mode, and selects the cycle of the LCD clock (duty driving).**

## ■ LCDC Control Register 1 (LCR1)

**Figure 18.4-2 LCDC Control Register 1 (LCR1)**

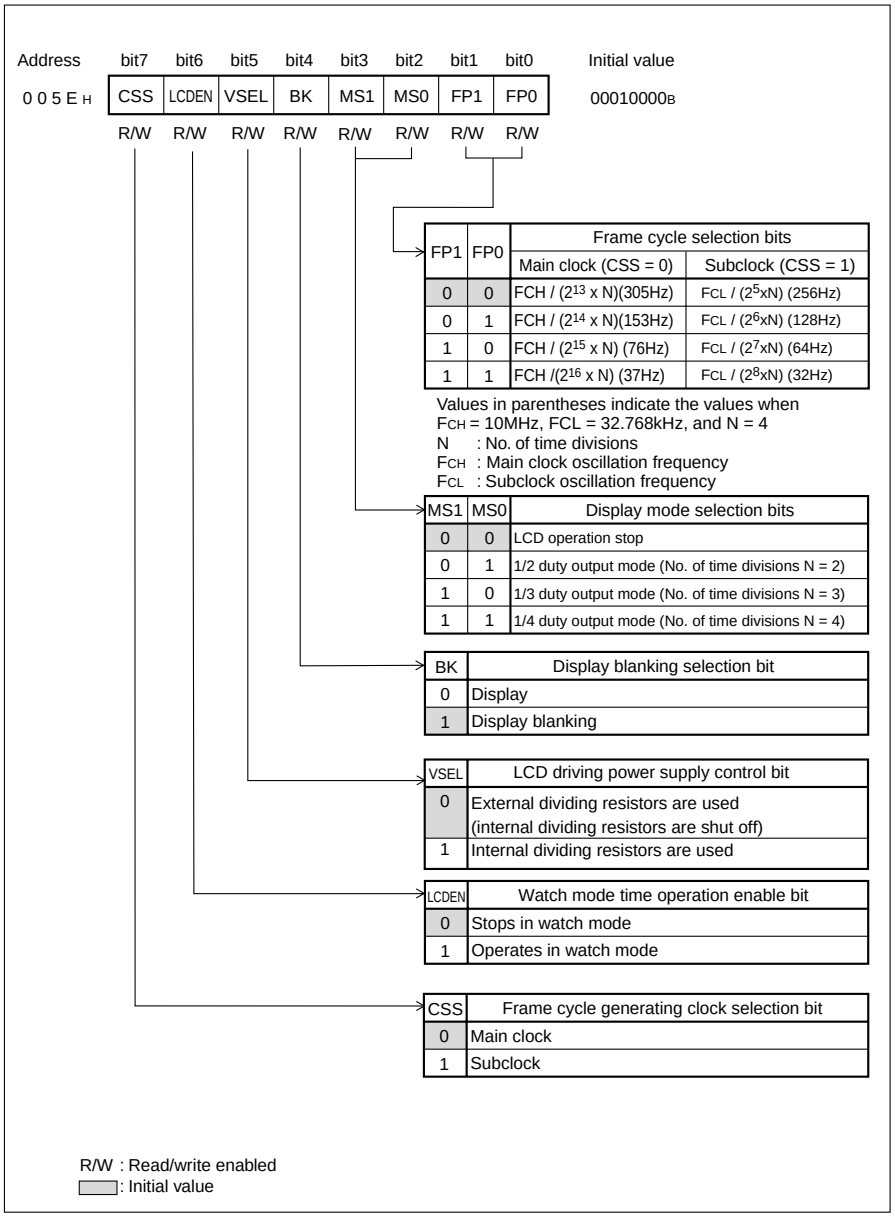


**Table 18.4-1 Functions of Each Bit in LCDC Control Register 1 (LCR1)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | CSS:<br>Frame cycle generating clock selection bit | • This bit selects the clock for generating the LCD display frame cycle (for driving duty).<br>• When this bit is "0," the LCD controller driver is operated on the output of the time base timer that is the main clock oscillation frequency divided by $2^8$. When this bit is "1," it is operated on the subclock<br>**Note:**<br>In main stop and subclock modes, the LCD controller driver cannot be operated on the output of the time base timer because oscillation of the main clock is stopped. To operate the LCD controller driver on the subclock, check that the subclock is fully stabilized before switching to the subclock.<br>**Reference:**<br>Even if the rate of the main clock is switched (gear function) when the LCD controller driver is operating on the timer base timer, the frame frequency is not affected. |
| Bit 6 | LCDEN:<br>Watch mode time operation enable bit | • This bit selects whether the LCD controller driver is operated in watch mode.<br>• When this bit is "1," LCD display continues even after moving to the watch mode.<br>• When this bit is "0," LCD display is stopped in watch mode.<br>**Note:**<br>To operate LCD display even in watch mode, the subclock (CSS = 1) must be selected. |
| Bit 5 | VSEL:<br>LCD driving power supply control bit | • This bit selects whether the internal dividing resistors are energized.<br>• When this bit is "0," the internal dividing resistors are shut off; when this bit is "1," the resistors are energized. To connect external dividing resistors, this bit must be set to "0." |
| Bit 4 | BK:<br>Display blanking selection bit | • This bit selects whether or not the LCD is displayed.<br>• In display blanking (non-display, BK = 1), the segment output indicates a non-selected waveform (a waveform that is out of the display condition). |
| Bit 3<br>Bit 2 | MS1, MS0:<br>Display mode selection bits | • These bits select the duty of the output waveform from three types.<br>• The operating common pins are determined in accordance with the selected duty output mode.<br>• When these bits are "00B," the LCD controller driver stops display.<br>**Note:**<br>When the selected frame cycle generating clock stops due to a shift to the top mode or otherwise, stop the display operation in advance. |

**Table 18.4-1 Functions of Each Bit in LCDC Control Register 1 (LCR1) (Continued)**

| Bit name | | Function |
|---|---|---|
| Bit 1<br>Bit 0 | FP1, FP0:<br>Frame cycle selection bits | These bits select the frame cycle of the LCD display (For duty driving) from four types.<br>**Note:**<br>Set the register after calculating the optimum frame frequency in accordance with the LCD module used.<br>The frame cycle is affected by the oscillation frequency. |

## 18.4.2 LCDC Control Register 2 (LCR2)

**The LCDC control register 2 (LCR2) selects whether Port 6 is used as segment output pins or as a general-purpose I/O port.**

### ■ LCDC Control Register 2 (LCR2)

**Figure 18.4-3 LCDC Control Register 2 (LCR2)**

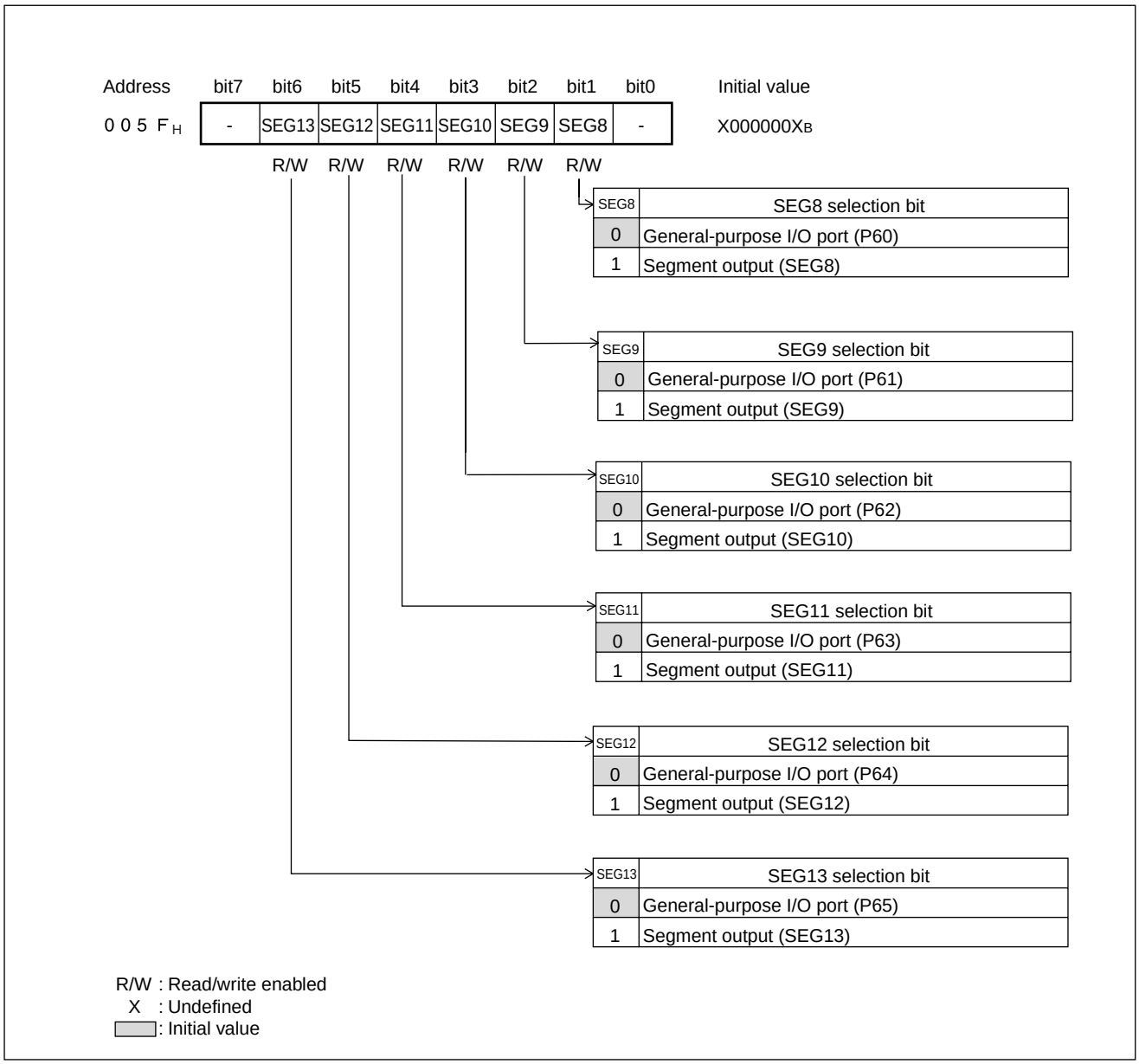


**Table 18.4-2 Functions of Each Bit in LCDC Control Register 2 (LCR2)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | Unused bit | • The read value is undefined.<br>• Writing has no effect on operation. |
| Bit 6 | SEG13 | SEG13 selection bit<br>When this bit is "1," P65 functions as segment output (SEG13). When this bit is "0," it functions as a general-purpose I/O port (P65). |
| Bit 5 | SEG12 | SEG12 selection bit<br>When this bit is "1," P64 functions as segment output (SEG12). When this bit is "0," it functions as a general-purpose I/O port (P64). |
| Bit 4 | SEG11 | SEG11 selection bit<br>When this bit is "1," P63 functions as segment output (SEG11). When this bit is "0," it functions as a general-purpose I/O port (P63). |
| Bit 3 | SEG10 | SEG10 selection bit<br>When this bit is "1," P62 functions as segment output (SEG10). When this bit is "0," it functions as a general-purpose I/O port (P62). |
| Bit 2 | SEG9 | SEG9 selection bit<br>When this bit is "1," P61 functions as segment output (SEG9). When this bit is "0," it functions as a general-purpose I/O port (P61). |
| Bit 1 | SEG8 | SEG8 selection bit<br>When this bit is "1," P60 functions as segment output (SEG8). When this bit is "0," it functions as a general-purpose I/O port (P60). |
| Bit 0 | Unused bit | • The read value is undefined.<br>• Writing has no effect on operation. |

# 18.4.3 LCDC Control Register 3 (LCR3)

**The LCDC control register 3 (LCR3) selects whether Port A is used as segment output pins or as a general-purpose I/O port.**

## ■ LCDC Control Register 3 (LCR3)

**Figure 18.4-4 LCDC Control Register 3 (LCR3)**

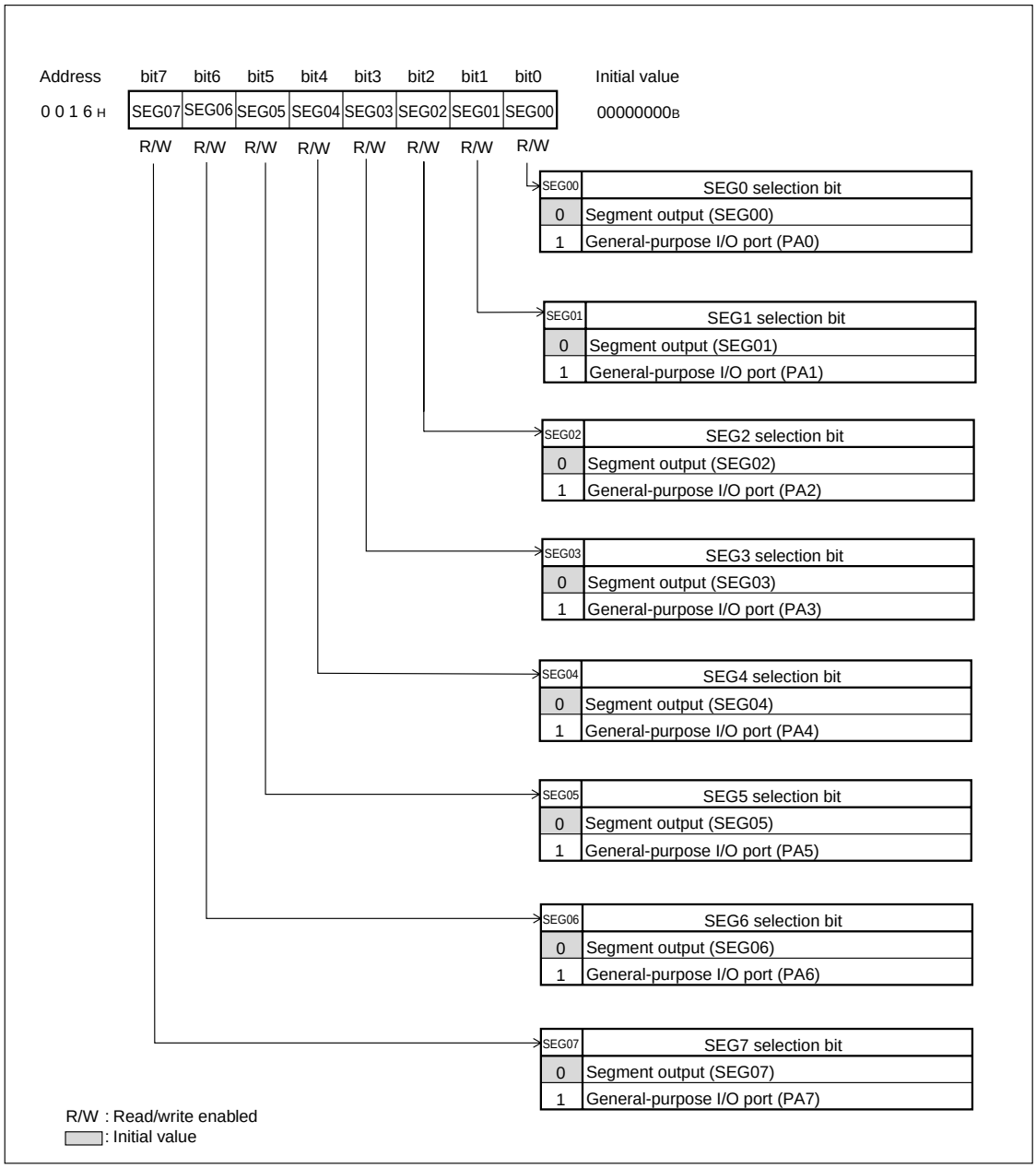


**Table 18.4-3 Functions of Each Bit in LCDC Control Register 3 (LCR3)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | SEG7 | SEG7 selection bit<br>When this bit is "0," PA7 functions as segment output (SEG07). When this bit is "1," it functions as a general-purpose I/O port (PA7). |
| Bit 6 | SEG6 | SEG6 selection bit<br>When this bit is "0," PA6 functions as segment output (SEG06). When this bit is "1," it functions as a general-purpose I/O port (PA6). |
| Bit 5 | SEG5 | SEG5 selection bit<br>When this bit is "0," PA5 functions as segment output (SEG05). When this bit is "1," it functions as a general-purpose I/O port (PA5). |
| Bit 4 | SEG4 | SEG4 selection bit<br>When this bit is "0," PA4 functions as segment output (SEG04). When this bit is "1," it functions as a general-purpose I/O port (PA4). |
| Bit 3 | SEG3 | SEG3 selection bit<br>When this bit is "0," PA3 functions as segment output (SEG03). When this bit is "1," it functions as a general-purpose I/O port (PA3). |
| Bit 2 | SEG2 | SEG2 selection bit<br>When this bit is "0," PA2 functions as segment output (SEG02). When this bit is "1," it functions as a general-purpose I/O port (PA2). |
| Bit 1 | SEG1 | SEG1 selection bit<br>When this bit is "0," PA1 functions as segment output (SEG01). When this bit is "1," it functions as a general-purpose I/O port (PA1). |
| Bit 0 | SEG0 | SEG0 selection bit<br>When this bit is "0," PA0 functions as segment output (SEG00). When this bit is "1," it functions as a general-purpose I/O port (PA0). |

# 18.4.4 LCDC Control Register 4 (LCR4)

**The LCDC control register 4 (LCR4) selects whether Port B is used as common output pins or as a general-purpose I/O port.**

## ■ LCDC Control Register 4 (LCR4)

**Figure 18.4-5 LCDC Control Register 4 (LCR4)**

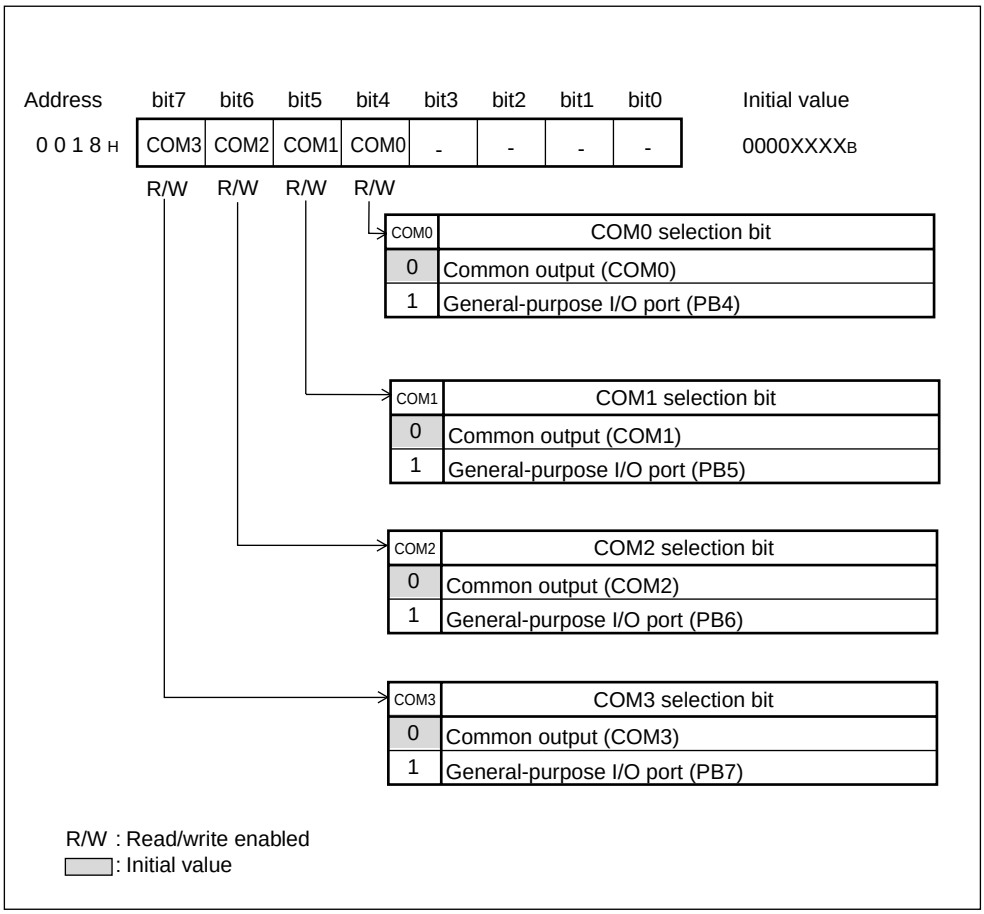


**Table 18.4-4 Functions of Each Bit in LCDC Control Register 4 (LCR4)**

| Bit name | | Function |
|---|---|---|
| Bit 7 | COM3 | COM3 selection bit<br>When this bit is "0," PB7 functions as common output (COM3). When this bit is "1," it functions as a general-purpose I/O port (PB7). |
| Bit 6 | COM2 | COM2 selection bit<br>When this bit is "0," PB6 functions as common output (COM2). When this bit is "1," it functions as a general-purpose I/O port (PB6). |
| Bit 5 | COM1 | COM1 selection bit<br>When this bit is "0," PB5 functions as common output (COM1). When this bit is "1," it functions as a general-purpose I/O port (PB5). |
| Bit 4 | COM0 | COM0 selection bit<br>When this bit is "0," PB4 functions as common output (COM0). When this bit is "1," it functions as a general-purpose I/O port (PB4). |
| Bit 3<br>Bit 2<br>Bit 1<br>Bit 0 | Unused bits | • The read value is undefined.<br>• Writing has no effect on operation. |

# 18.5 LCD Display RAM in the LCD Controller Driver

**The LCD display RAM is 14 x 4 bit (7 byte) display data memory for generating segment output signals.**

## ■ LCD Display RAM and Output Pins

The description in this RAM is read automatically in synch with the selection timing of the common signal and output from the segment output pins.

If the description in each bit is "1," it is output after being converted into the selected voltage (LCD is displayed). If the description in each bit is "0," it is output after being converted into the non-selected voltage (LCD is not displayed).

Since the LCD display operation is performed independently of the CPU operation, the LCD display RAM can be read and written in any timing.

Of the SEG8 to SEG15 pins, the pins not specified as segment output are used as an I/O port (general-purpose output) and the corresponding RAM can be used as an ordinary RAM.

Table 18.5-1 "Relationships between Duty and Common Output and the Bits Used in the LCD Display RAM" shows the relationships between duty and common output and the bits used in the

LCD display RAM. Figure 18.5-1 "Assignment of the LCD Display RAM and the Common/ Segment Output Pins" shows the assignment of the LCD display RAM and the common/ segment output pins.

**Figure 18.5-1 Assignment of the LCD Display RAM and the Common/Segment Output Pins**

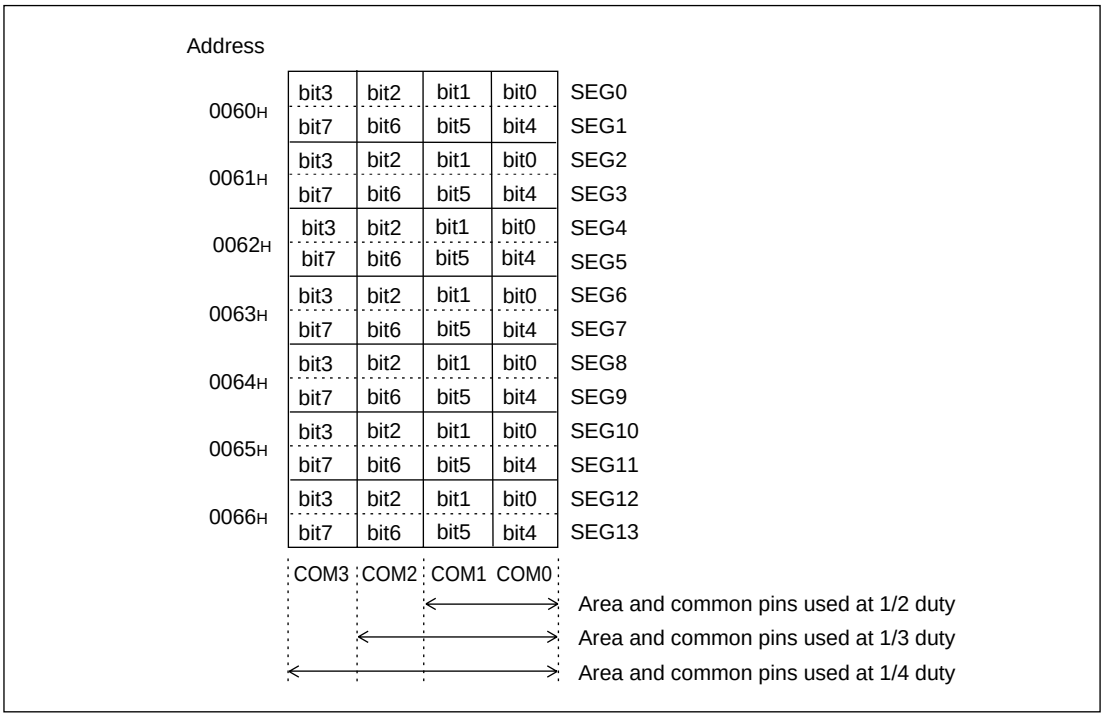


**Table 18.5-1 Relationships between Duty and Common Output and the Bits Used in the LCD Display RAM**

| Duty set value | Common output used | Bits of data for display used | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
| 1/2 | COM0 to COM1 (2 pins) | - | - | Yes | Yes | - | - | Yes | Yes |
| 1/3 | COM0 to COM2 (3 pins) | - | Yes | Yes | Yes | - | Yes | Yes | Yes |
| 1/4 | COM0 to COM3 (4 pins) | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

Yes: Used
-: Not used

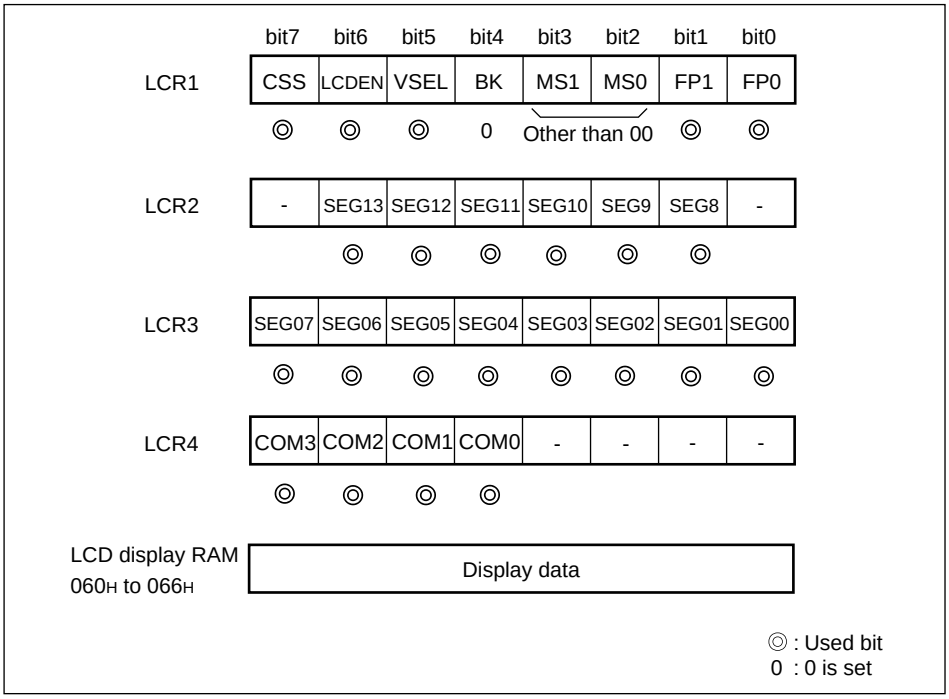
# 18.6 Operation of the LCD Controller Driver

**The LCD controller driver performs the control and driving required to display the LCD.**

## ■ Explanation of LCD Controller Driver Operation

To display the LCD, the settings in Figure 18.6-1 "LCD Controller Driver Settings" are required.

**Figure 18.6-1 LCD Controller Driver Settings**




When the above settings are made and the selected frame cycle generating clock is oscillating, the driving waveform of the LCD panel is output to the common and segment output pins (COM0 to COM3, SEG0 to SEG13) in accordance with the description in the LCD display RAM and LCRs 1 to 4.

Even during LCD display operation, the frame cycle generating clock can be switched. However, the display may blink at the time of switching. Stop the display temporarily by blanking (LCR1: BK = 1) before switching the clock.

The output for driving the display is two-frame alternating current waveform selected by the setting of bias and duty.

When LCD display operation is stopped (LCR: MS1, MS0 = 00B), both the common and segment output pins are set at the "L" level.

**Note:**

When the selected frame cycle oscillating clock is stopped during LCD display operation, a DC voltage is applied to the LCD elements because the alternating current generating circuit is stopped. In this case, the LCD display operation must be stopped in advance. The conditions for stopping the main clock (time base timer) and subclock are dependent on the selection of the clock mode and standby mode. When the timer base timer output is selected (LCR1: CSS = 0), clearing the time base timer affects the frame cycle.

## ■ LCD Driving Waveform

Driving the LCD with direct current creates a chemical reaction that degrades the LCD display elements. Therefore, the LCD controller driver contains an alternating current generating circuit and drives the LCD with the two frame alternating current waveform. The three types of output waveforms are as follows.

- 1/2 bias, 1/2 duty output waveform
- 1/3 bias, 1/3 duty output waveform
- 1/3 bias, 1/4 duty output waveform

## 18.6.1 Output Waveforms during LCD Controller Driver Operation (1/2 Duty)

**The display driving output uses two-frame alternating current waveforms in the multiplex driving method. In 1/2 duty, only COM0 and COM1 are used for display. COM2 and COM3 are not used.**

### ■ Example of 1/2 Bias, 1/2 Duty Output Waveforms

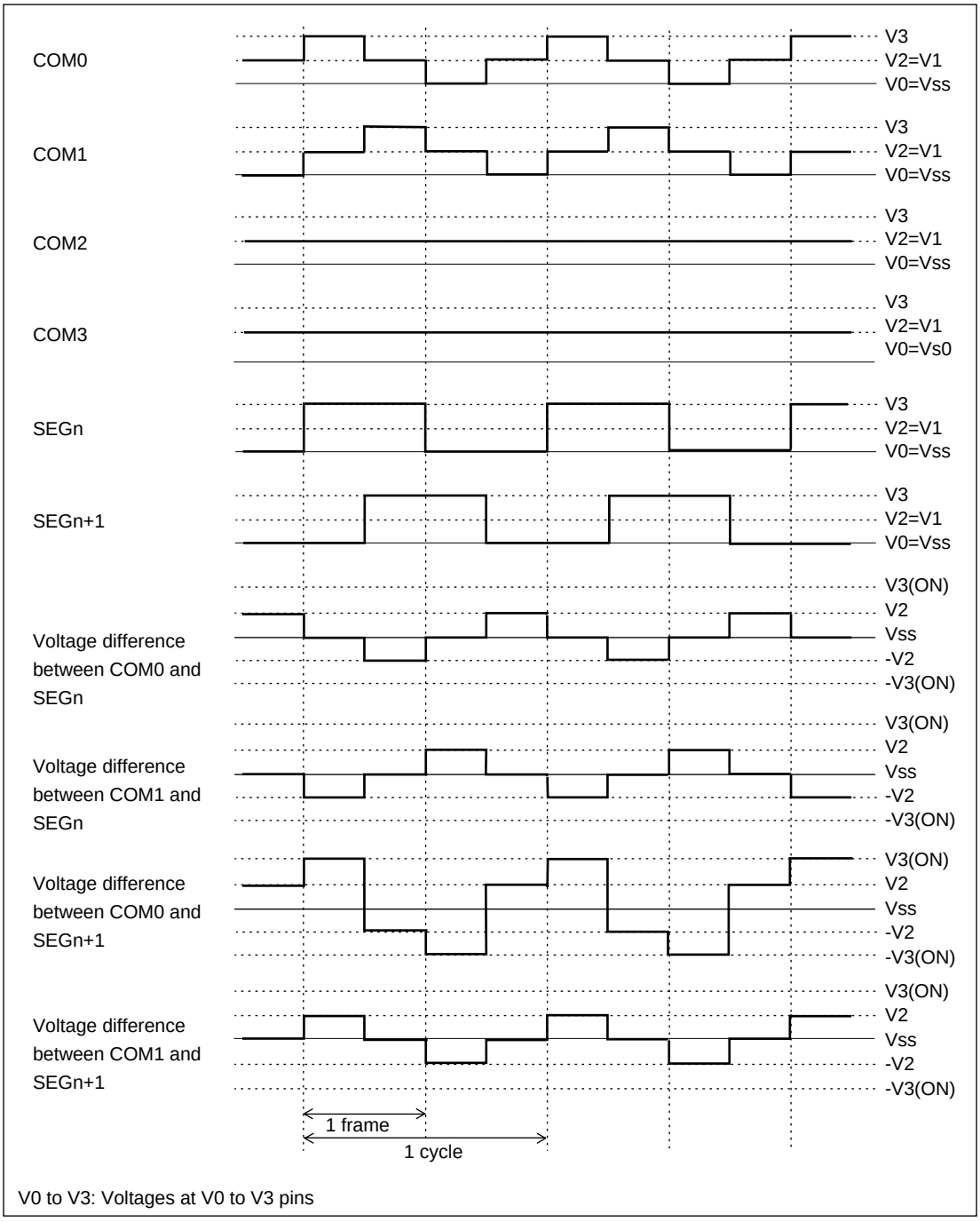
For display, the LCD elements with the maximum potential difference between the common output and the segment output are turned on.

Figure 18.6-2 "Example of 1/2 Bias, 1/2 Duty Output Waveform" shows the output waveforms when the description in the LCD display RAM is as shown in Table 18.6-1 "Description Example of LCD Display RAM".

**Table 18.6-1 Description Example of LCD Display RAM**

| Segment | Description in LCD display RAM | | | |
|---|---|---|---|---|
| | COM3 | COM2 | COM1 | DOM0 |
| $SEG_m$ | – | – | 0 | 0 |
| $SEG_{m+1}$ | – | – | 0 | 1 |

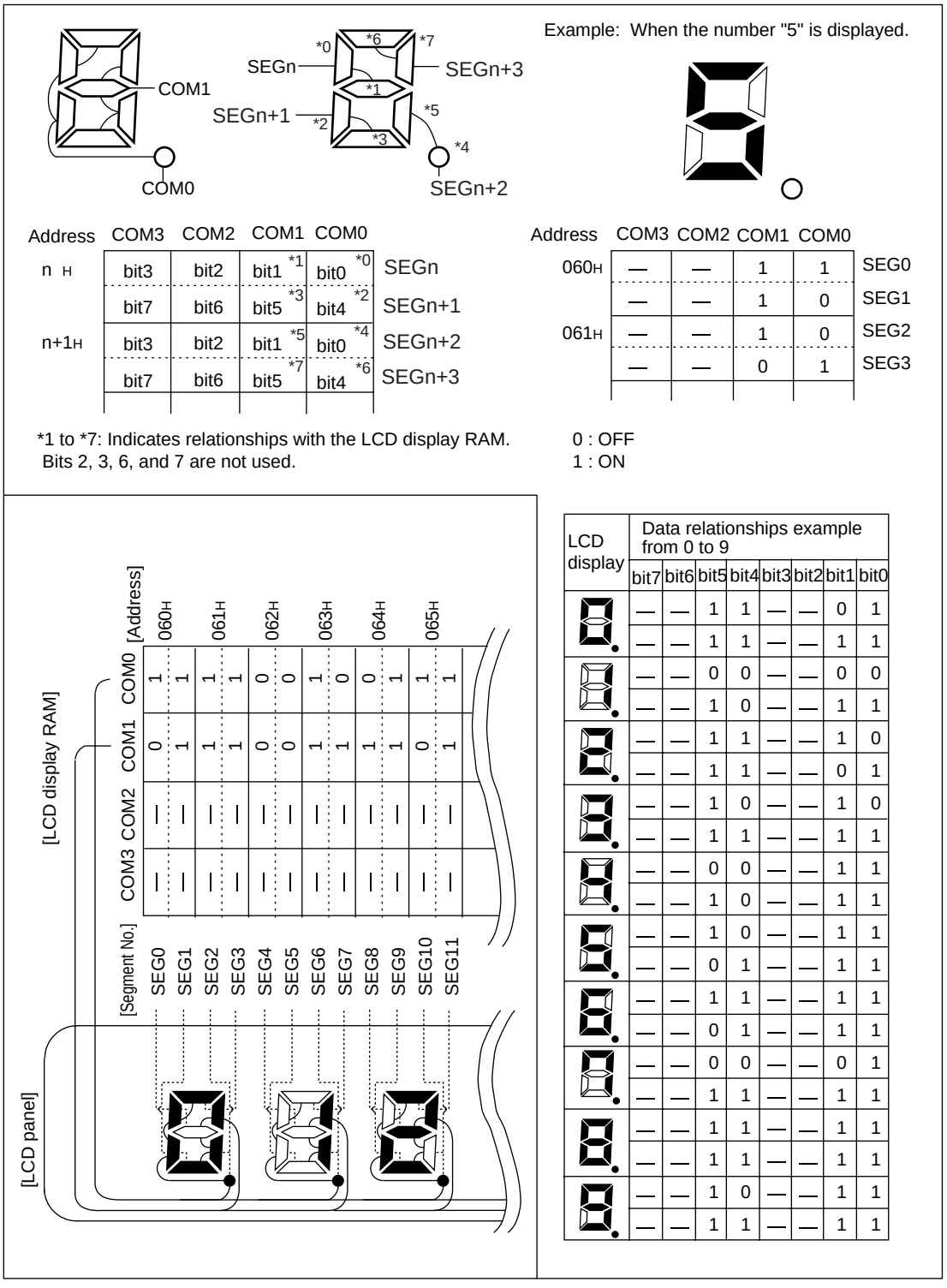
**Figure 18.6-2 Example of 1/2 Bias, 1/2 Duty Output Waveform**




V0 to V3: Voltages at V0 to V3 pins

## ■ LCD Panel Connection Example and Display Data Example (1/2 Duty Driving Method)

**Figure 18.6-3 Connection Example of Segment And Common Pins and Correspondence With Display Data**




| Address | COM3 | COM2 | COM1 | COM0 | |
|---|---|---|---|---|---|
| n H | bit3 | bit2 | bit1 *1 | bit0 *0 | SEGn |
| | bit7 | bit6 | bit5 *3 | bit4 *2 | SEGn+1 |
| n+1H | bit3 | bit2 | bit1 *5 | bit0 *4 | SEGn+2 |
| | bit7 | bit6 | bit5 *7 | bit4 *6 | SEGn+3 |

| Address | COM3 | COM2 | COM1 | COM0 | |
|---|---|---|---|---|---|
| 060H | — | — | 1 | 1 | SEG0 |
| | — | — | 1 | 0 | SEG1 |
| 061H | — | — | 1 | 0 | SEG2 |
| | — | — | 0 | 1 | SEG3 |

*1 to *7: Indicates relationships with the LCD display RAM.
Bits 2, 3, 6, and 7 are not used.

0 : OFF
1 : ON




| LCD display | Data relationships example from 0 to 9 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| 0 | — | — | 1 | 1 | — | — | 0 | 1 |
| | — | — | 1 | 1 | — | — | 1 | 1 |
| 1 | — | — | 0 | 0 | — | — | 0 | 0 |
| | — | — | 1 | 0 | — | — | 1 | 1 |
| 2 | — | — | 1 | 1 | — | — | 1 | 0 |
| | — | — | 1 | 1 | — | — | 0 | 1 |
| 3 | — | — | 1 | 0 | — | — | 1 | 0 |
| | — | — | 1 | 1 | — | — | 1 | 1 |
| 4 | — | — | 0 | 0 | — | — | 1 | 1 |
| | — | — | 1 | 0 | — | — | 1 | 1 |
| 5 | — | — | 1 | 0 | — | — | 1 | 1 |
| | — | — | 0 | 1 | — | — | 1 | 1 |
| 6 | — | — | 1 | 1 | — | — | 1 | 1 |
| | — | — | 0 | 1 | — | — | 1 | 1 |
| 7 | — | — | 0 | 0 | — | — | 0 | 1 |
| | — | — | 1 | 1 | — | — | 1 | 1 |
| 8 | — | — | 1 | 1 | — | — | 1 | 1 |
| | — | — | 1 | 1 | — | — | 1 | 1 |
| 9 | — | — | 1 | 0 | — | — | 1 | 1 |
| | — | — | 1 | 1 | — | — | 1 | 1 |

## 18.6.2 Output Waveforms During LCD Controller Driver Operation (1/3 Duty)

**In 1/3 duty, only COM0, COM1, and COM2 are used for display. COM3 is not used.**

### ■ Example of 1/3 Bias, 1/3 Duty Output Waveforms

For display, the LCD elements with the maximum potential difference between the common output and the segment output are turned on.

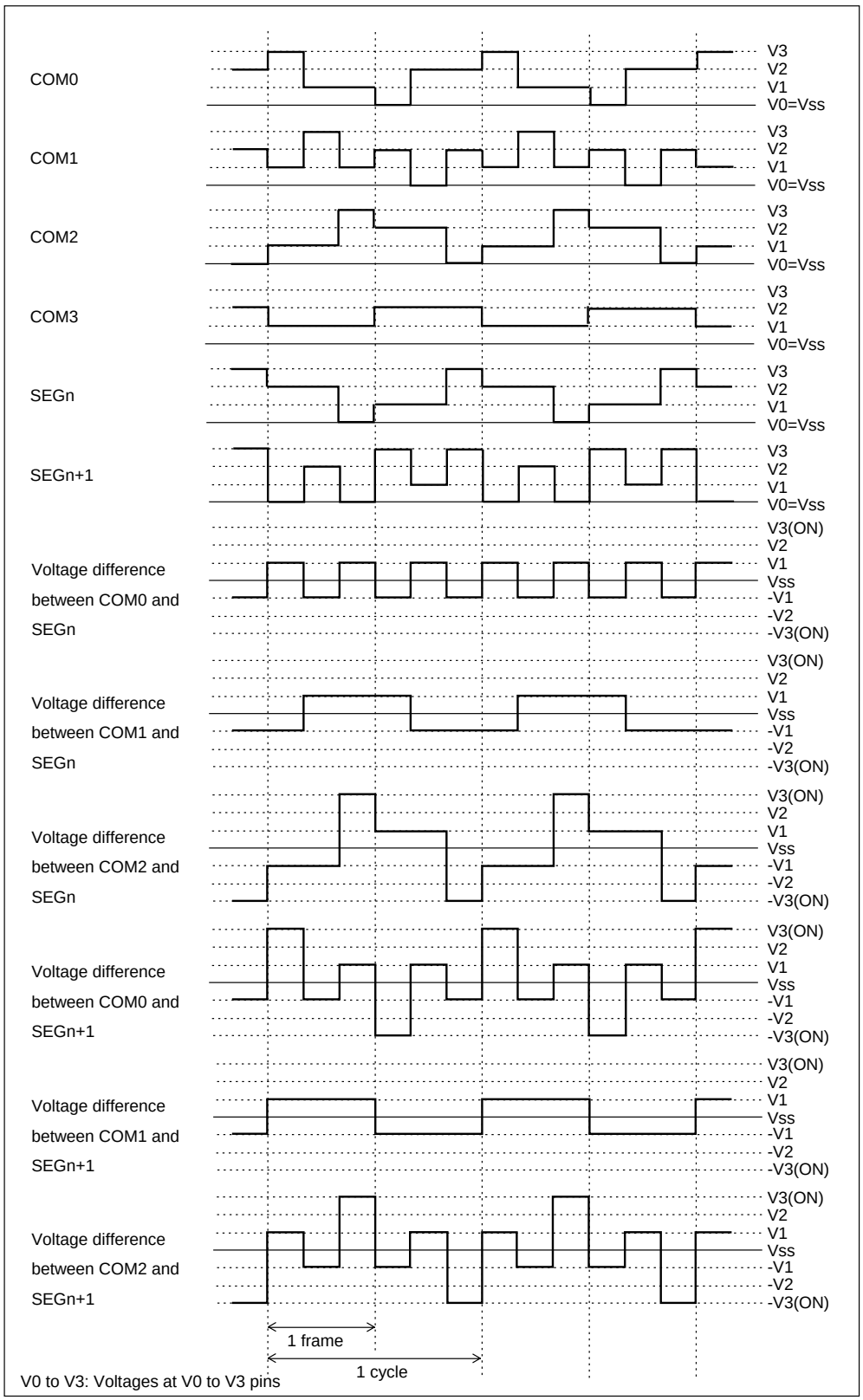
Figure 18.6-4 "Example of 1/3 Bias, 1/3 Duty Output Waveforms" shows the output waveforms when the description in the LCD display RAM is as shown in Table 18.6-2 "Description Example of LCD display RAM".

**Table 18.6-2 Description Example of LCD display RAM**

| Segment | Description in LCD display RAM | | | |
|---|---|---|---|---|
| | COM3 | COM2 | COM1 | DOM0 |
| $SEG_m$ | - | 1 | 0 | 0 |
| $SEG_{m+1}$ | - | 1 | 0 | 1 |

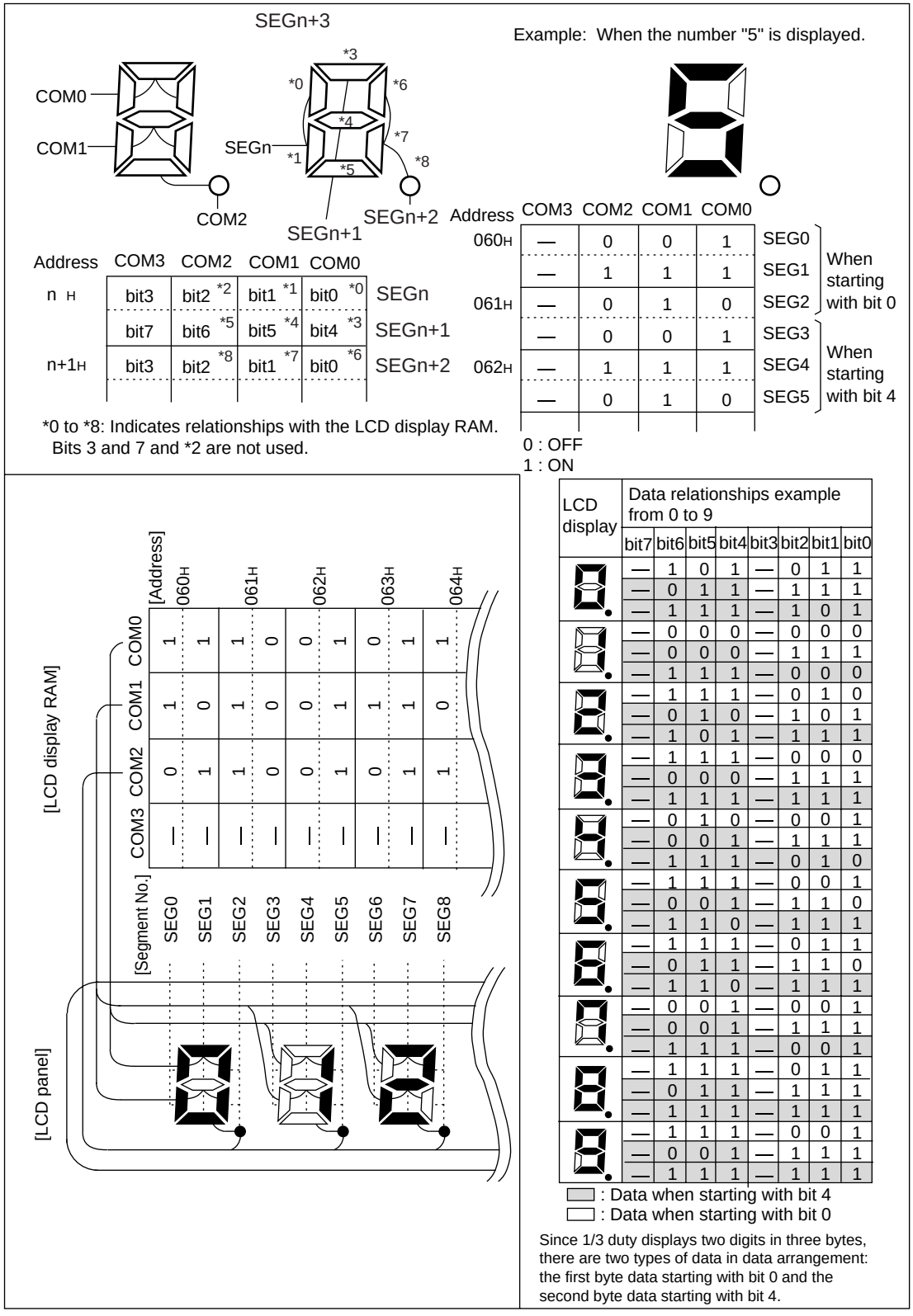
-: Not used

**Figure 18.6-4 Example of 1/3 Bias, 1/3 Duty Output Waveforms**



## ■ LCD Panel Connection Example and Display Data Example (1/3 Duty Driving Method)

### Figure 18.6-5 Connection Example of Segment And Common Pins and Relationships to Display Data




| LCD display | Data relationships example from 0 to 9 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| 0 | — | 1 | 0 | 1 | — | 0 | 1 | 1 |
| | — | 0 | 1 | 1 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 1 | 0 | 1 |
| 1 | — | 0 | 0 | 0 | — | 0 | 0 | 0 |
| | — | 0 | 0 | 0 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 0 | 0 | 0 |
| 2 | — | 1 | 1 | 1 | — | 0 | 1 | 0 |
| | — | 0 | 1 | 0 | — | 1 | 0 | 1 |
| | — | 1 | 0 | 1 | — | 1 | 1 | 1 |
| 3 | — | 1 | 1 | 1 | — | 0 | 0 | 0 |
| | — | 0 | 0 | 0 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 1 | 1 | 1 |
| 4 | — | 0 | 1 | 0 | — | 0 | 0 | 1 |
| | — | 0 | 0 | 1 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 0 | 1 | 0 |
| 5 | — | 1 | 1 | 1 | — | 0 | 0 | 1 |
| | — | 0 | 0 | 1 | — | 1 | 1 | 0 |
| | — | 1 | 1 | 0 | — | 1 | 1 | 1 |
| 6 | — | 1 | 1 | 1 | — | 0 | 1 | 1 |
| | — | 0 | 1 | 1 | — | 1 | 1 | 0 |
| | — | 1 | 1 | 0 | — | 1 | 1 | 1 |
| 7 | — | 0 | 0 | 1 | — | 0 | 0 | 1 |
| | — | 0 | 0 | 1 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 0 | 0 | 1 |
| 8 | — | 1 | 1 | 1 | — | 0 | 1 | 1 |
| | — | 0 | 1 | 1 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 1 | 1 | 1 |
| 9 | — | 1 | 1 | 1 | — | 0 | 0 | 1 |
| | — | 0 | 0 | 1 | — | 1 | 1 | 1 |
| | — | 1 | 1 | 1 | — | 1 | 1 | 1 |

# 18.6.3 Output Waveforms During LCD Controller Driver Operation (1/4 Duty)

**In 1/4 duty, all of COM0, COM1, COM2 and COM3 are used for display.**

## ■ Example of 1/3 Bias, 1/4 Duty Output Waveforms

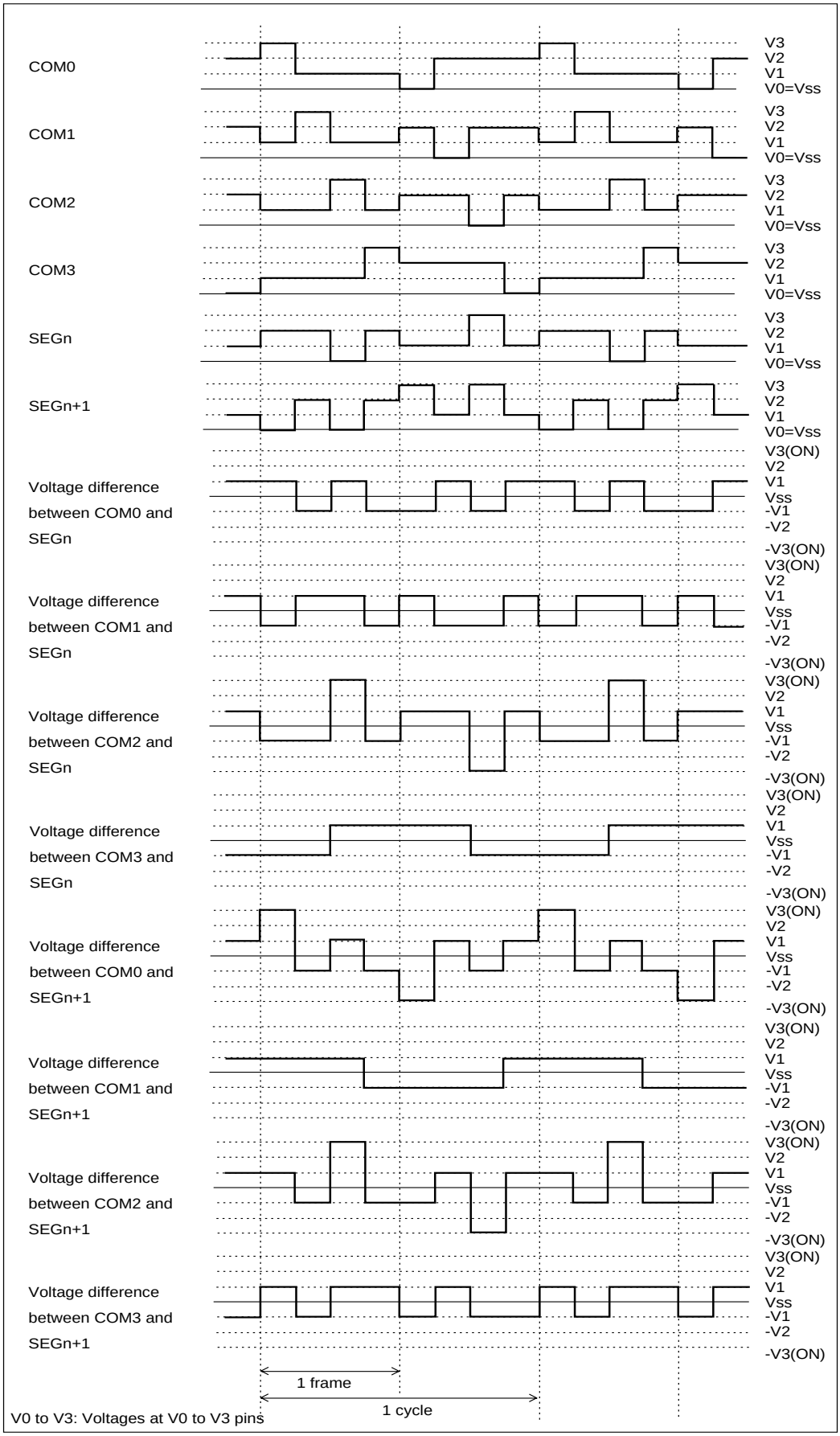
For display, the LCD elements with the maximum potential difference between the common output and the segment output are turned on.

Figure 18.6-6 "Example of 1/3 Bias, 1/4 Duty Output Waveforms" shows the output waveforms when the description in the LCD display RAM is as shown in Table 18.6-3 "Description Example of LCD display RAM".

**Table 18.6-3 Description Example of LCD display RAM**

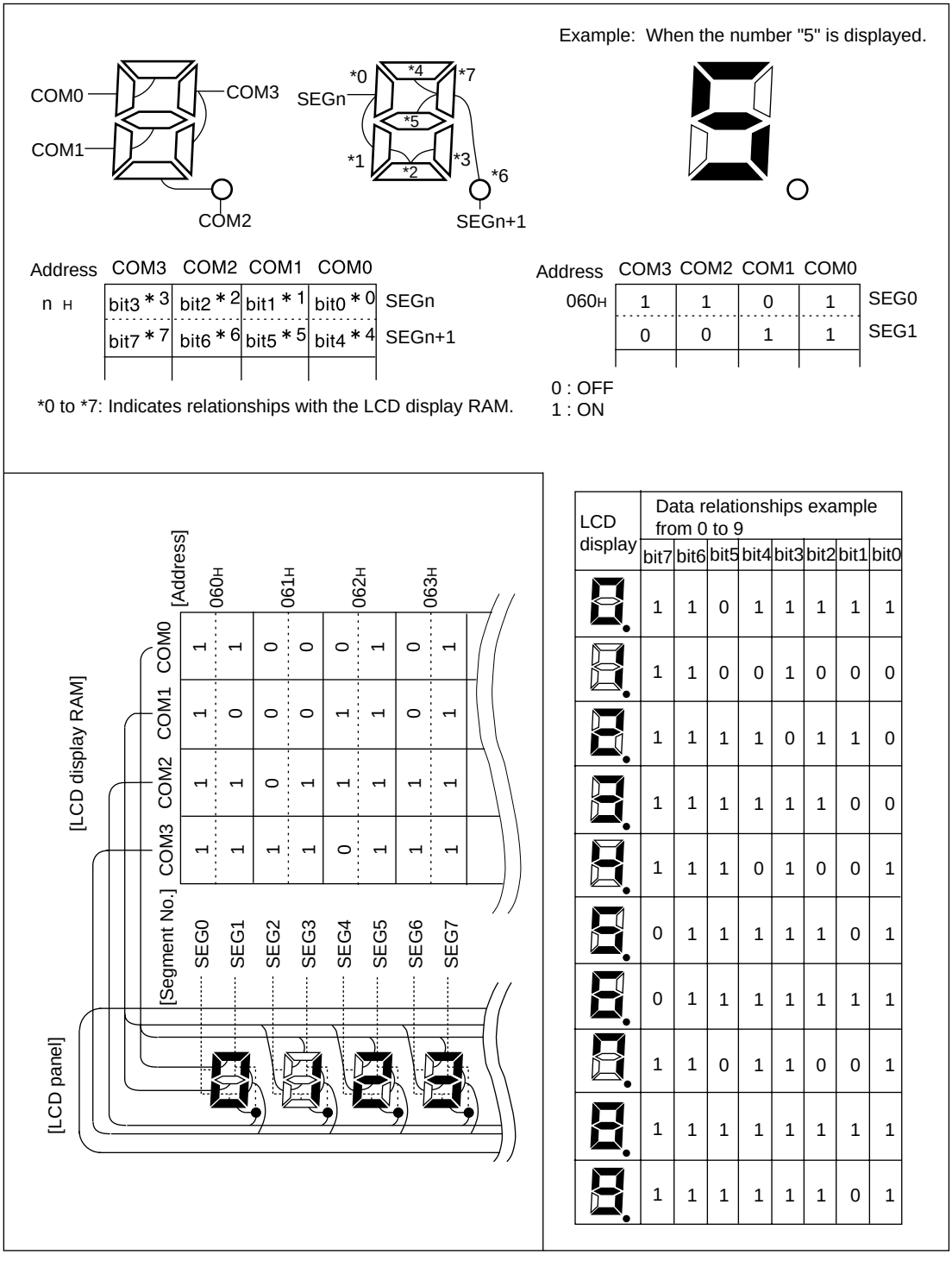
| Segment | Description in LCD display RAM | | | |
|---|---|---|---|---|
| | COM3 | COM2 | COM1 | DOM0 |
| $SEG_m$ | 0 | 1 | 0 | 0 |
| $SEG_{m+1}$ | 0 | 1 | 0 | 1 |

**Figure 18.6-6 Example of 1/3 Bias, 1/4 Duty Output Waveforms**



## ■ Connection Example of 8-segment LCD Panel and Display Data Example (1/4 Duty Driving Method)

**Figure 18.6-7 Connection Example of Segment and Common Pins and Relationships to Display Data**




| LCD display | Data relationships example from 0 to 9 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 3 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 4 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 5 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 7 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |

# CHAPTER 19 WILD REGISTER FUNCTION

**This chapter describes the functions and operations of the wild register function.**


19.1 "Overview of the Wild Register Function"

19.2 "Configuration of the Wild Register Function"

19.3 "Registers of the Wild Register Function"

19.4 "Operation of the Wild Register Function"

# 19.1 Overview of the Wild Register Function

**The wild register function applies a patch for a program fault by setting an address and correction data in an internal register. Up to six bytes of data can be corrected.**

## ■ Wild Register Function

The wild register function assigns an address in the ROM space of the microcontroller and replaces the existing data at the address with new data. Thus, if a program has a fault, you can correct the faulty data by setting its address and the address if the correction data in the register.

## ■ Wild Register Application Address

The address space where the wild register function can be used depends on the model. Table 19.1-1 "Applicable Addresses for the Wild Register Function" shows the applicable addresses for the wild register function for each model.

**Table 19.1-1 Applicable Addresses for the Wild Register Function**

| Model name | ROM space |
|---|---|
| MB89PV570 | $0C92_H$ to $FFFF_H$ |
| MB89P579 | $1000_H$ to $FFFF_H$ |
| MB89577 | $8000_H$ to $FFFF_H$ |

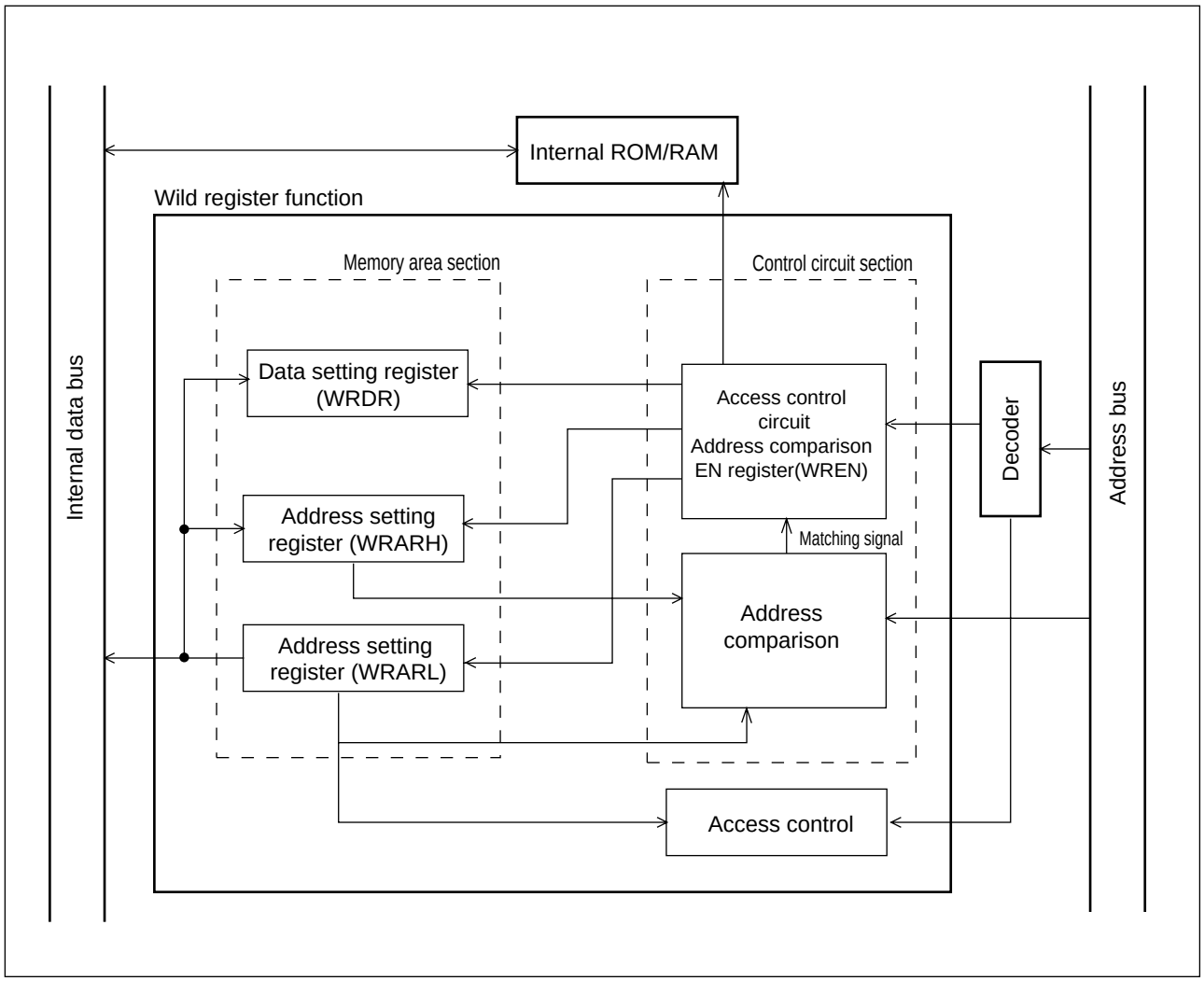
# 19.2 Configuration of the Wild Register Function

**The wild register function consists of the following two blocks:**

- **Memory area section**
  **Data setting register (WRDR)**
  **Upper address setting register (WRARH)**
  **Lower address setting register (WRARL)**
- **Control circuit section**
  **Address comparison enable register (WREN)**

## ■ Block Diagram of the Wild Register Function

**Figure 19.2-1 Block Diagram of the Wild Register Function**



❍ **Memory area section**

The memory area section consists of the data setting registers, the address setting register (H address), and the address setting register (L address). In the memory area section, set the address and the data that will be used to replace the fault using the wild register function. Each MB89570 series model has these registers, each of which is six bytes.

❍ **Control circuit section**

The control circuit section compares the data set in the address setting registers and the actual data on the address bus. If it finds a match, it sets the data in the data setting register on the data bus. The operation of the control circuit section is controlled by the address comparison enable register.

# 19.3 Registers of the Wild Register Function

**This section describes the registers related to the wild register function.**

## ■ Registers Related to the Wild Register Function

**Figure 19.3-1 Registers Related to the Wild Register Function**

WRDR0 to WRDR5 [Data setting registers]

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0C82H<br>0C85H<br>0C88H<br>0C8BH<br>0C8EH<br>0C91H | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | W | W | W | W | W | W | W | W | |

WRARH0 to WRARH5 [Address setting registers (upper bytes)]

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0C80H<br>0C83H<br>0C86H<br>0C89H<br>0C8CH<br>0C8FH | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

WRARL0 to WRARL5 [Address setting registers (lower bytes)]

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0C81H<br>0C84H<br>0C87H<br>0C8AH<br>0C8DH<br>0C90H | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

WREN [Address comparison enable register]

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0077H | - | - | EN05 | EN04 | EN03 | EN02 | EN01 | EN00 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
W : Write only
X : Undefined

## 19.3.1 Data Setting Registers (WRDR0 to 5)

**The data setting registers (WRDR0 to 5) contain correction data to be set using the wild register function.**

■ **Data Setting Registers (WRDR)**

**Figure 19.3-2 Data Setting Registers (WRDR)**

| | Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|---|
| WRDR0 | 0C82H | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |
| WRDR1 | 0C85H | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |
| WRDR2 | 0C88H | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |
| WRDR3 | 0C8BH | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |
| WRDR4 | 0C8EH | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |
| WRDR5 | 0C91H | RD07 | RD06 | RD05 | RD04 | RD03 | RD02 | RD01 | RD00 | XXXXXXXXB |
| | | W | W | W | W | W | W | W | W | |

W : Write only
X : Undefined

**Table 19.3-1 Functions of the Data Setting Registers (WRDR)**

| Wild register number | Register name | Function |
|---|---|---|
| 0 | WRDR0 | One-byte register that stores data at the address assigned by WRARL and WRARH. This data is valid at the address (WRARL and WRARH) corresponding to each wild register number. |
| 1 | WRDR1 | |
| 2 | WRDR2 | |
| 3 | WRDR3 | |
| 4 | WRDR4 | |
| 5 | WRDR5 | |

**Note:**

While the WREN register is set, reading the address set in WRAH (address H) and WRAL (address L) reads the value in the WRDR (data) register.

## 19.3.2 Upper Address Setting Registers (WRARH0 to 5)

**The upper address setting registers (WRARH0 to 5) contain the upper part of an address where data is corrected.**

### ■ Upper Address Setting Registers (WRARH)

**Figure 19.3-3 Upper Address Setting Registers (WRARH)**

| | Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|---|
| WRARH0 | 0C80H | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARH1 | 0C83H | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARH2 | 0C86H | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARH3 | 0C89H | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARH4 | 0C8CH | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARH5 | 0C8FH | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA09 | RA08 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

**Table 19.3-2 Functions of the Upper Address Setting Registers (WRARH)**

| Wild register number | Register name | Function |
|---|---|---|
| 0 | WRARH0 | One-byte register that specifies the upper address of memory to be assigned. Specifies an address corresponding to the wild register number. |
| 1 | WRARH1 | |
| 2 | WRARH2 | |
| 3 | WRARH3 | |
| 4 | WRARH4 | |
| 5 | WRARH5 | |

## 19.3.3 Lower Address Setting Registers (WRARL 0 to 5)

**The lower address setting registers (WRARL0 to 5) contain the lower part of an address where data is corrected.**

### ■ Lower Address Setting Registers (WRARL)

**Figure 19.3-4 Lower Address Setting Registers (WRARL)**

| | Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|---|
| WRARL0 | 0C81H | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARL1 | 0C84H | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARL2 | 0C87H | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARL3 | 0C8AH | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARL4 | 0C8DH | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| WRARL5 | 0C90H | RA07 | RA06 | RA05 | RA04 | RA03 | RA02 | RA01 | RA00 | XXXXXXXXB |
| | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

**Table 19.3-3 Functions of the Upper Address Setting Registers (WRARH)**

| Wild register number | Register name | Function |
|---|---|---|
| 0 | WRARL0 | One-byte register that specifies the lower address of memory to be assigned. Specifies an address corresponding to a wild register number. |
| 1 | WRARL1 | |
| 2 | WRARL2 | |
| 3 | WRARL3 | |
| 4 | WRARL4 | |
| 5 | WRARL5 | |

# 19.3.4 Address Comparison Enable Register (WREN)

**The address comparison enable register (WREN) enables or disables the operation of the wild register function for each wild register number.**

## ■ Address Comparison Enable Register (WREN)

**Figure 19.3-5 Address Comparison Enable Register (WREN)**

WREN (Address comparison enable register)

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value |
|---|---|---|---|---|---|---|---|---|---|
| 0077H | - | - | EN05 | EN04 | EN03 | EN02 | EN01 | EN00 | XX000000B |
| | | | R/W | R/W | R/W | R/W | R/W | R/W | |

R/W : Read/write enabled
X : Undefined

**Table 19.3-4 Functions of the Address Comparison Enable Register (WREN)**

| Bit name | | Function |
|---|---|---|
| Bit 7<br>Bit 6 | Unused bits | • The read value is undefined.<br>• For a write, always write 0. |
| Bit 5 | EN05: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH5 and WRARL5, the value in the WRDR5 is output to the internal bus instead of the value in ROM. |
| Bit 4 | EN04: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH4 and WRARL4, the value in the WRDR4 is output to the internal bus instead of the value in ROM. |
| Bit 3 | EN03: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH3 and WRARL3, the value in WRDR3 is output to the internal bus instead of the value in ROM. |
| Bit 2 | EN02: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH2 and WRARL2, the value in the WRDR2 is output to the internal bus instead of the value in ROM. |
| Bit 1 | EN01: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH1 and WRARL1, the value in the WRDR1 is output to the internal bus instead of the value in ROM. |
| Bit 0 | EN00: | • Setting this bit to 0 disables the wild register function.<br>• Setting this bit to 1 enables the wild register function. If a match is then found for the address set in WRARH0 and WRARL0, the value in the WRDR0 is output to the internal bus instead of the value in ROM. |

# 19.4 Operation of the Wild Register Function

**This section describes the sequence of wild register function operations.**

## ■ Sequence of Wild Register Function Operations

The following shows the sequence of wild register function operations. In the operation example, the data $FF_H$ at the address $FC36_H$ is corrected to $B5_H$.

**Table 19.4-1 Sequence of Wild Register Function Operations**

| | Operation | Operation example |
|---|---|---|
| 1 | Set the address of the wild register support area in the address setting registers. | Address: $FC36_H$, Data: $FF_H$<br>WRARL0=$36_H$<br>WRARH0=$FC_H$ |
| 2 | Set the correction data in the data setting registers. | WRDR0=$B5_H$ |
| 3 | Set 1 in the address comparison enable register. | WREN=$01_H$ |
| 4 | If the addresses match, the wild register function is enabled. | If the address $FC36_H$ is accessed<br>↓<br>Data = $B5_H$ |

## ■ List of Wild Register Addresses

The following shows the list of addresses corresponding to the wild register numbers.

**Table 19.4-2 List of Wild Register Addresses**

| | Upper address | | Lower address | | Data | |
|---|---|---|---|---|---|---|
| | Register name | Address | Register name | Address | Register name | Address |
| 1 | WRARH1 | $0C80_H$ | WRARL1 | $0C81_H$ | WRDR1 | $0C82_H$ |
| 2 | WRARH2 | $0C83_H$ | WRARL2 | $0C84_H$ | WRDR2 | $0C85_H$ |
| 3 | WRARH3 | $0C86_H$ | WRARL3 | $0C87_H$ | WRDR3 | $0C88_H$ |
| 4 | WRARH4 | $0C89_H$ | WRARL4 | $0C8A_H$ | WRDR4 | $0C8B_H$ |
| 5 | WRARH5 | $0C8C_H$ | WRARL5 | $0C8D_H$ | WRDR5 | $0C8E_H$ |
| 6 | WRARH6 | $0C8F_H$ | WRARL6 | $0C90_H$ | WRDR6 | $0C91_H$ |

# APPENDIX

**This appendix includes I/O maps, instruction lists, and other information.**

APPENDIX A "I/O Maps"

APPENDIX B "Overview of Instructions"

APPENDIX C "Mask Options"

APPENDIX D "One-time PROM and EPROM Microcontroller Write Specification"

APPENDIX E "Pin Statuses of the MB89570 Series"

# APPENDIX A I/O Maps

**The addresses shown in Table A-1 "I/O Map" are assigned to the registers of peripheral functions contained in the MB89570 series.**

## ■ I/O Maps

**Table A-1 I/O Map**

| Address | Abbreviation of register | Register name | Read and write | Initial value |
|---|---|---|---|---|
| $00_H$ | PDR0 | Port 0 data register | R/W | $XXXXXXXX_B$ |
| $01_H$ | DDR0 | Port 0 direction register | W | $00000000_B$ |
| $02_H$ | PDR1 | Port 1 data register | R/W | $XXXXXXXX_B$ |
| $03_H$ | DDR1 | Port 1 direction register | W | $00000000_B$ |
| $04_H$ | PDR2 | Port 2 data register | R/W | $XXXXXXXX_B$ |
| $05_H$ | (Unused area) | | | |
| $06_H$ | DDR2 | Port 2 direction register | R/W | $00000000_B$ |
| $07_H$ | SYCC | System clock control register | R/W | $XXXMM100_B$ |
| $08_H$ | STBC | Standby control register | R/W | $00010XXX_B$ |
| $09_H$ | WDTC | Watchdog control register | R/W | $0XXXXXXX_B$ |
| $0A_H$ | TBTC | Timebase timer control register | R/W | $X0XXX000_B$ |
| $0B_H$ | WPCR | Watch prescaler control register | R/W | $X0XX0000_B$ |
| $0C_H$ | (Unused area) | | | |
| $0D_H$ | (Unused area) | | | |
| $0E_H$ | RSFR | Reset flag register | R | $XXXXXXXX_B$ |
| $0F_H$ | (Unused area) | | | |
| $10_H$ | SMC1 | Serial mode control register 1 | R/W | $00000000_B$ |
| $11_H$ | SMM2 | Serial mode control register 2 | R/W | $00000000_B$ |
| $12_H$ | SSD | Serial status and data register | R | $00001XXX_B$ |
| $13_H$ | SIDR/SODR | Serial input/serial output data register | R/W | $XXXXXXXX_B$ |
| $14_H$ | SRC | Baud rate generator reload register | R/W | $XXXXXXXX_B$ |
| $15_H$ | PDRA | Port A data register | R/W | $11111111_B$ |
| $16_H$ | LCR3 | LCDC control register 3 | R/W | $00000000_B$ |

**Table A-1 I/O Map (Continued)**

| Address | Abbreviation of register | Register name | Read and write | Initial value |
|---|---|---|---|---|
| $17_H$ | PDRB | Port B data register | R/W | $11111111_B$ |
| $18_H$ | LCR4 | LCDC control register 4 | R/W | $0000XXXX_B$ |
| $19_H$ | BRSR3 | Bridge circuit selection register 3 | R/W | $XXXXX001_B$ |
| $1A_H$ | T2CR | Timer 2 control register | R/W | $X00000X0_B$ |
| $1B_H$ | T1CR | Timer 1 control register | R/W | $X00000X0_B$ |
| $1C_H$ | T2DR | Timer 2 data register | R/W | $XXXXXXXX_B$ |
| $1D_H$ | T1DR | Timer 1 data register | R/W | $XXXXXXXX_B$ |
| $1E_H$ | (Unused area) | | | |
| $1F_H$ | (Unused area) | | | |
| $20_H$ | PDR3 | Port 3 data register | R/W | $XX111111_B$ |
| $21_H$ | PDR4 | Port 4 data register | R/W | $XXXX1111_B$ |
| $22_H$ | PDR5 | Port 5 data register | R/W | $XXXXXXXX_B$ |
| $23_H$ | DDR5 | Port 5 direction register | R/W | $X0000000_B$ |
| $24_H$ | PDR6 | Port 6 data register | R/W | $XX111111_B$ |
| $25_H$ | PDR7 | Port 7 data register | R/W | $XXXXXXXX_B$ |
| $26_H$ | DDR7 | Port 7 direction register | R/W | $00000000_B$ |
| $27_H$ | PDR8 | Port 8 data register | R/W | $XXXXXXXX_B$ |
| $28_H$ | DDR8 | Port 8 direction register | R/W | $000X0000_B$ |
| $29_H$ | PDR9 | Port 9 data register | R/W | $XXXXXXXX_B$ |
| $2A_H$ | DDR9 | Port 9 direction register | R/W | $XXXXX000_B$ |
| $2B_H$ | (Unused area) | | | |
| $2C_H$ | (Unused area) | | | |
| $2D_H$ | ADEN1 | A/D enable register 1 | R/W | $XXXX1111_B$ |
| $2E_H$ | ADEN2 | A/D enable register 2 | R/W | $11111111_B$ |
| $2F_H$ | ADC1 | A/D control register 1 | R/W | $00000000_B$ |
| $30_H$ | ADC2 | A/D control register 2 | R/W | $X000001_B$ |
| $31_H$ | (Unused area) | | | |
| $32_H$ | ADDH | A/D data register (upper bytes) | R/W | $XXXXXXXX_B$ |
| $33_H$ | ADDL | A/D data register (lower bytes) | R/W | $XXXXXXXX_B$ |
| $34_H$ | (Unused area) | | | |
| $35_H$ | IBSR | $I^2C$ bus status register | R | $00000000_B$ |

**Table A-1 I/O Map (Continued)**

| Address | Abbreviation of register | Register name | Read and write | Initial value |
|---|---|---|---|---|
| $36_H$ | IBCR | $I^2C$ bus control register | R/W | $00000000_B$ |
| $37_H$ | ICCR | $I^2C$ clock control register | R/W | $0X0XXXXX_B$ |
| $38_H$ | IADR | $I^2C$ address register | R/W | $XXXXXXXX_B$ |
| $39_H$ | IDAR | $I^2C$ data register | R/W | $XXXXXXXX_B$ |
| $3A_H$ | ITCR | $I^2C$ timeout control register | R/W | $X0000000_B$ |
| $3B_H$ | ITSR | $I^2C$ timeout status register | R/W | $XXXX0000_B$ |
| $3C_H$ | ITOD | $I^2C$ timeout data register | R/W | $XXXXXXXX_B$ |
| $3D_H$ | ITOC | $I^2C$ timeout clock register | R/W | $XXXXXXXX_B$ |
| $3E_H$ | IMTO | $I^2C$ master timeout register | R/W | $XXXXXXXX_B$ |
| $3F_H$ | ISTO | $I^2C$ slave timeout register | R/W | $XXXXXXXX_B$ |
| $40_H$ | MBSR | Multi-address $I^2C$ bus status register | R | $00000000_B$ |
| $41_H$ | MBCR | Multi-address $I^2C$ bus control register | R/W | $00000000_B$ |
| $42_H$ | MCCR | Multi-address $I^2C$ bus clock control register | R/W | $0X0XXXXX_B$ |
| $43_H$ | MADR1 | Multi-address $I^2C$ bus address register 1 | R/W | $XXXXXXXX_B$ |
| $44_H$ | MADR2 | Multi-address $I^2C$ bus address register 2 | R/W | $XXXXXXXX_B$ |
| $45_H$ | MADR3 | Multi-address $I^2C$ bus address register 3 | R/W | $XXXXXXXX_B$ |
| $46_H$ | MADR4 | Multi-address $I^2C$ bus address register 4 | R/W | $XXXXXXXX_B$ |
| $47_H$ | MADR5 | Multi-address $I^2C$ bus address register 5 | R/W | $XXXXXXXX_B$ |
| $48_H$ | MADR6 | Multi-address $I^2C$ bus address register 6 | R/W | $XXXXXXXX_B$ |
| $49_H$ | MADR | Multi-address $I^2C$ bus data register | R/W | $XXXXXXXX_B$ |
| $4A_H$ | MTCR | Multi-address $I^2C$ bus timeout control register | R/W | $X0000000_B$ |
| $4B_H$ | MTSR | Multi-address $I^2C$ bus timeout status register | R/W | $XXXX0000_B$ |
| $4C_H$ | MTOD | Multi-address $I^2C$ bus timeout data register | R/W | $XXXXXXXX_B$ |
| $4D_H$ | MTOC | Multi-address $I^2C$ bus timeout clock register | R/W | $XXXXXXXX_B$ |
| $4E_H$ | MMTO | Multi-address $I^2C$ bus master timeout register | R/W | $XXXXXXXX_B$ |
| $4F_H$ | MSTO | Multi-address $I^2C$ bus slave timeout register | R/W | $XXXXXXXX_B$ |
| $50_H$ | MALR | Multi-address $I^2C$ bus ALART register | R/W | $XXXX0000_B$ |
| $51_H$ | COCR1 | Comparator control register 1 | R/W | $XX000000_B$ |
| $52_H$ | COCR2 | Comparator control register 2 | R/W | $XXX11111_B$ |
| $53_H$ | COSR1 | Comparator status register 1 | R/W | $00000000_B$ |

**Table A-1 I/O Map (Continued)**

| Address | Abbreviation of register | Register name | Read and write | Initial value |
|---|---|---|---|---|
| $54_H$ | CICR1 | Comparator interrupt control register 1 | R/W | $00000000_B$ |
| $55_H$ | COSR2 | Comparator status register 2 | R/W | $XX000000_B$ |
| $56_H$ | CICR2 | Comparator interrupt control register 2 | R/W | $XX000000_B$ |
| $57_H$ | COSR3 | Comparator status register 3 | R | $XXXXXXXX_B$ |
| $58_H$ | COSR4 | Comparator status register 4 | R | $XXXXXXXX_B$ |
| $59_H$ | CIER | Comparator input enable register | R/W | $XXX11111_B$ |
| $5A_H$ | EIC1 | External interrupt control register 1 | R/W | $00000000_B$ |
| $5B_H$ | EIC2 | External interrupt control register 2 | R/W | $00000000_B$ |
| $5C_H$ | BRSR1 | Bridge circuit selection register 1 | R/W | $XXXXX000_B$ |
| $5D_H$ | BRSR2 | Bridge circuit selection register 2 | R/W | $XX000000_B$ |
| $5E_H$ | LCR1 | LCDC control register 1 | R/W | $00010000_B$ |
| $5F_H$ | LCR2 | LCDC control register 2 | R/W | $X000000X_B$ |
| $60_H$ to $66_H$ | VRAM | LCD display RAM | R/W | $XXXXXXXX_B$ |
| $67_H$ to $6F_H$ | (Unused area) | | | |
| $70_H$ | DACR | D/A control register | R/W | $XXXXXX00_B$ |
| $71_H$ | DADR1 | D/A data register 1 | R/W | $XXXXXXXX_B$ |
| $72_H$ | DADR2 | D/A data register 2 | R/W | $XXXXXXXX_B$ |
| $73_H$ | TMCR | Timer control register | R/W | $XX000000_B$ |
| $74_H$ | TCHR | Timer count register (upper bytes) | R/W | $00000000_B$ |
| $75_H$ | TCLR | Timer count register (lower bytes) | R/W | $00000000_B$ |
| $76_H$ | (Unused area) | | | |
| $77_H$ | WREN | Wild register address comparison enable register | R/W | $XX000000_B$ |
| $78_H$ | Wild register test register | | | |
| $79_H$ | (Unused area) | | | |
| $7A_H$ | (Unused area) | | | |
| $7B_H$ | ILR1 | Interrupt level setting register 1 | W | $11111111_B$ |
| $7C_H$ | ILR2 | Interrupt level setting register 2 | W | $11111111_B$ |
| $7D_H$ | ILR3 | Interrupt level setting register 3 | W | $11111111_B$ |

**Table A-1 I/O Map (Continued)**

| Address | Abbreviation of register | Register name | Read and write | Initial value |
|---|---|---|---|---|
| $7E_H$ | ILR4 | Interrupt level setting register 4 | W | $11111111_B$ |
| $7F_H$ | (Unused area) | | | |
| $0C80_H$ | WRARH1 | Wild register address setting register 1 (upper bytes) | R/W | XXXXXXXX |
| $0C81_H$ | WRARL1 | Wild register address setting register 1 (lower bytes) | R/W | XXXXXXXX |
| $0C82_H$ | WRDR1 | Wild register data setting register 1 | W | XXXXXXXX |
| $0C83_H$ | WRARH2 | Wild register address setting register 2 (upper bytes) | R/W | XXXXXXXX |
| $0C84_H$ | WRARL2 | Wild register address setting register 2 (lower bytes) | R/W | XXXXXXXX |
| $0C85_H$ | WRDR2 | Wild register Data setting register 2 | W | XXXXXXXX |
| $0C86_H$ | WRARH3 | Wild register address setting register 3 (upper bytes) | R/W | XXXXXXXX |
| $0C87_H$ | WRARL3 | Wild register address setting register 3 (lower bytes) | R/W | XXXXXXXX |
| $0C88_H$ | WRDR3 | Wild register Data setting register 3 | W | XXXXXXXX |
| $0C89_H$ | WRARH4 | Wild register address setting register 4 (upper bytes) | R/W | XXXXXXXX |
| $0C8A_H$ | WRARL4 | Wild register address setting register 4 (lower bytes) | R/W | XXXXXXXX |
| $0C8B_H$ | WRDR4 | Wild register Data setting register 4 | W | XXXXXXXX |
| $0C8C_H$ | WRARH5 | Wild register address setting register 5 (upper bytes) | R/W | XXXXXXXX |
| $0C8D_H$ | WRARL5 | Wild register address setting register 5 (lower bytes) | R/W | XXXXXXXX |
| $0C8E_H$ | WRDR5 | Wild register Data setting register 5 | W | XXXXXXXX |
| $0C8F_H$ | WRARH6 | Wild register address setting register 6 (upper bytes) | R/W | XXXXXXXX |
| $0C90_H$ | WRARL6 | Wild register address setting register 6 (lower bytes) | R/W | XXXXXXXX |
| $0C91_H$ | WRDR6 | Wild register Data setting register 6 | W | XXXXXXXX |

❍ **Description of "Read and write"**

- R/W: Read/write enabled
- R: Read only

- W: Write only

❍ **Description of "Initial value"**

- 0: This bit has the initial value of "0".
- 1: This bit has the initial value of "1".
- X: This bit has an undefined initial value.

**Note:**

Do not use the unused area.

# APPENDIX B Overview of Instructions

**Appendix B describes the instructions used by the F$^2$MC-8L.**

B.1 "Overview of F$^2$MC-8L instructions"

B.2 "Addressing"

B.3 "Special Instructions"

B.4 "Bit Manipulation Instructions (SETB, CLRB)"

B.5 "F$^2$MC-8L Instructions"

B.6 "Instruction Map"

# B.1 Overview of F$^2$MC-8L instructions

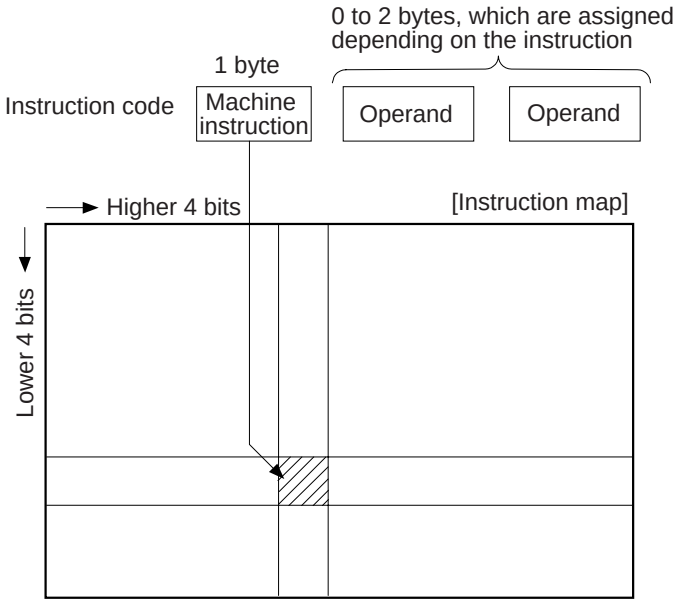
## The F$^2$MC-8L supports 140 types of instructions.

### ■ Overview of F2MC-8L instructions

The F$^2$MC-8L has 140 1-byte machine instructions (256-byte instruction map). An instruction code consists of an instruction and zero or more operands that follow.

Figure B.1-1 "Relationship between the instruction codes and the instruction map" shows the relationship between the instruction codes and the instruction map.

**Figure B.1-1 Relationship between the instruction codes and the instruction map**




- The instructions are classified into four types: transfer, arithmetic, branch, and other.
- A variety of addressing methods is available. One of ten addressing modes can be selected depending on the selected instruction and specified operand(s).
- Bit manipulation instructions are provided. They can be used for read-modify-write operations.
- Some instructions are used for special operations.

## ■ Symbols used with instructions

Table B.1-1 "Symbols in the instruction list" lists the symbols used in the instruction code descriptions in Appendix B.

**Table B.1-1 Symbols in the instruction list**

| Symbol | Meaning |
|---|---|
| dir | Direct address (8 bits) |
| off | Offset (8 bits) |
| ext | Extended address (16 bits) |
| #vct | Vector table number (3 bits) |
| #d8 | Immediate data (8 bits) |
| #d16 | Immediate data (16 bits) |
| dir:16 | Bit direct address (8 bits:3 bits) |
| rel | Branch relative address (8 bits) |
| @ | Register indirect addressing (examples: @A, @IX, @EP) |
| A | Accumulator (8 or 16 bits, which are determined depending on the instruction being used) |
| AH | Higher 8 bits of the accumulator (8 bits) |
| AL | Lower 8 bits of the accumulator (8 bits) |
| T | Temporary accumulator (8 or 16 bits, which are determined depending on the instruction being used) |
| TH | Higher 8 bits of the temporary accumulator (8 bits) |
| TL | Lower 8 bits of the temporary accumulator (8 bits) |
| IX | Index register (16 bits) |
| EP | Extra pointer (16 bits) |
| PC | Program counter (16 bits) |
| SP | Stack pointer (16 bits) |
| PS | Program status (16 bits) |
| dr | Either accumulator or index register (16 bits) |
| CCR | Condition code register (8 bits) |
| RP | Register bank pointer (5 bits) |
| Ri | General-purpose register (8 bits, i = 0 to 7) |
| X | X is immediate data (8 or 16 bits, which are determined depending on the instruction being used). |
| (X) | The content of X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used). |
| ((X)) | The address indicated by the X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used). |

## ■ Items in the instruction list

**Table B.1-2 Items in the instruction list**

| Item | Description |
|---|---|
| MNEMONIC | This column shows the instruction in assembly language. |
| to | This column shows the number of cycles required by the instruction (instruction cycle count). |
| # | This column shows the number of bytes for the instruction. |
| Operation | This column shows the operation performed by the instruction. |
| TL, TH, AH | These columns indicate a change in the contents of TL, TH, and AH (automatic transfer from A to T) upon the execution of the instruction.<br>The meanings of symbols in each column are as follows:<br>• "-" indicates that no change is made.<br>• "dH" indicates the higher 8 bits of data in the operation column.<br>• "AL" and "AH" indicate that the contents of AL and AH immediately before the execution of the instruction are set.<br>• "00" indicates that 00 is set. |
| N, Z, V, C | These columns indicate whether their respective flags are changed upon the execution of the instruction.<br>A plus (+) sign indicates that the instruction changes the corresponding flag. |
| OP CODE | This column shows the operation code(s) of the instruction. When the instruction uses two or more operation codes, the following notation is used:<br>[Example] 48 to 4F: This represents from 48 to 4F. |

# B.2 Addressing

**The F2MC-8L has the following ten addressing modes:**

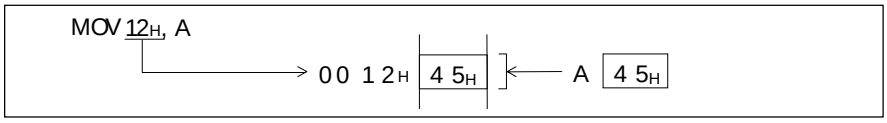
- **Direct addressing**
- **Extended addressing**
- **Bit direct addressing**
- **Index addressing**
- **Pointer addressing**
- **General-purpose register addressing**
- **Immediate addressing**
- **Vector addressing**
- **Relative addressing**
- **Inherent addressing**

## ■ Explanation of addressing

### ❍ Direct addressing

Direct addressing is indicated by dir in the instruction list. This addressing is used to access the area between $0000_H$ and $00FF_H$. In this addressing mode, the higher byte of the address is $00_H$ and the lower byte is specified by the operand. Figure B.2-1 "Example of direct addressing" shows an example.

**Figure B.2-1 Example of direct addressing**

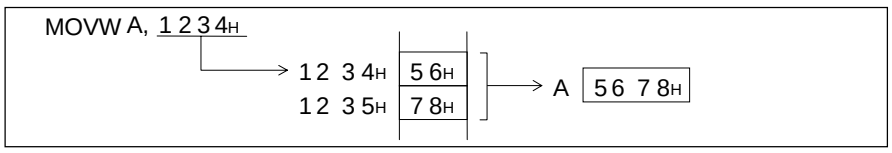



### ❍ Extended addressing

Extended addressing is indicated by ext in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the first operand specifies the higher byte of the address, and the second operand specifies the lower byte.

Figure B.2-2 "Example of extended addressing" shows an example.

**Figure B.2-2 Example of extended addressing**



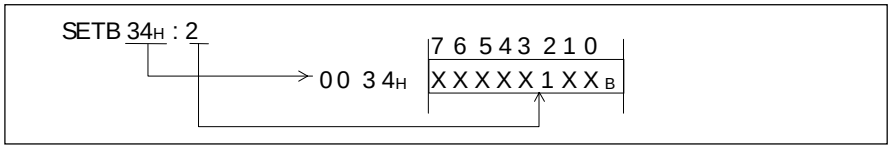

### ❍ Bit direct addressing

Bit direct addressing is indicated by dir:b in the instruction list. This addressing is used to access a particular bit in the area between $0000_H$ and $00FF_H$. In this addressing mode, the

higher byte of the address is $00_H$ and the lower byte is specified by the operand. The bit position at the address is specified by the lower three bits of the operation code.

Figure B.2-3 "Example of bit direct addressing" shows an example.

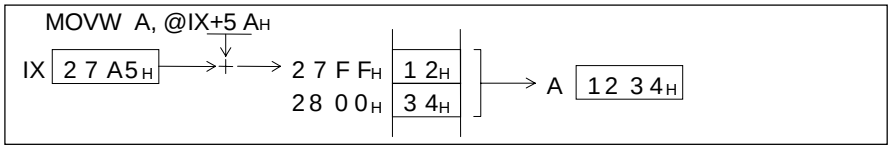
**Figure B.2-3 Example of bit direct addressing**




❍ **Index addressing**

Index addressing is indicated by @IX+off in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is the value resulting from sign-extending the contents of the first operand and adding them to IX (index register). Figure B.2-4 "Example of index addressing" shows an example.

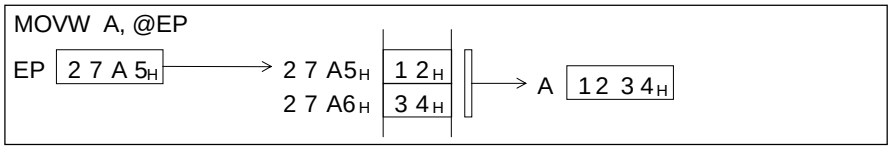
**Figure B.2-4 Example of index addressing**




❍ **Pointer addressing**

Pointer addressing is indicated by @EP in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is contained in EP (extra pointer). Figure B.2-5 "Example of pointer addressing" shows an example.

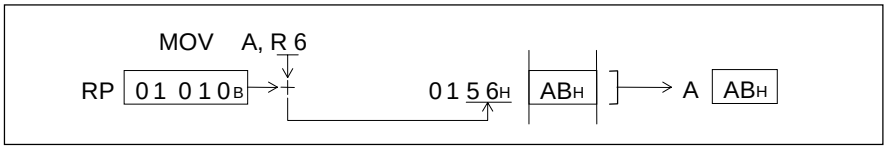
**Figure B.2-5 Example of pointer addressing**




❍ **General-purpose register addressing**

General-purpose register addressing is indicated by Ri in the instruction list. This addressing is used to access a register bank in the general-purpose register area. In this addressing mode, the higher byte of the address is always 01 and the lower byte is specified based on the contents of RP (register bank pointer) and the lower three bits of the operation code. Figure B.2-6 "Example of general-purpose register addressing" shows an example.

**Figure B.2-6 Example of general-purpose register addressing**

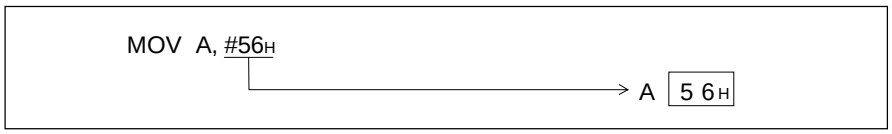



❍ **Immediate addressing**

Immediate addressing is indicated by #d8 in the instruction list. This addressing is used when

immediate data is required. In this addressing mode, the operand is used as immediate data. Whether the data is specified in bytes or words is determined by the operation code. Figure B.2-7 "Example of immediate addressing" shows an example.

**Figure B.2-7 Example of immediate addressing**




❍ **Vector addressing**

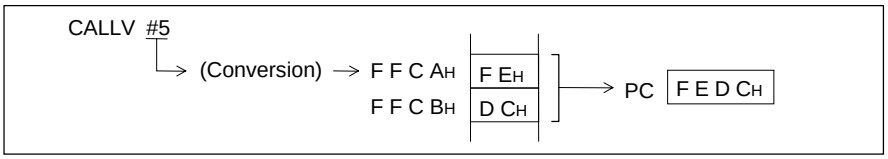
Vector addressing is indicated by vct in the instruction list. This addressing is used to branch to a subroutine address stored in the vector table. In this addressing mode, vct information is contained in the operation codes, and the corresponding table addresses are created as shown in Table B.2-1 "Vector table addresses corresponding to vct".

**Table B.2-1 Vector table addresses corresponding to vct**

| #vct | Vector table address (higher address:lower address of branch destination) |
|---|---|
| 0 | $FFC0_H$ : $FFC1_H$ |
| 1 | $FFC2_H$ : $FFC3_H$ |
| 2 | $FFC4_H$ : $FFC5_H$ |
| 3 | $FFC6_H$ : $FFC7_H$ |
| 4 | $FFC8_H$ : $FFC9_H$ |
| 5 | $FFCA_H$ : $FFCB_H$ |
| 6 | $FFCC_H$ : $FFCD_H$ |
| 7 | $FFCE_H$ : $FFCF_H$ |

Figure B.2-8 "Example of vector addressing" shows an example.

**Figure B.2-8 Example of vector addressing**

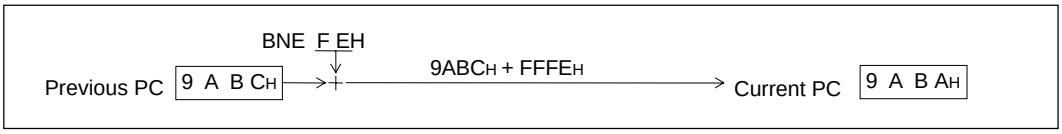



❍ **Relative addressing**

Relative addressing is indicated by rel in the instruction list. This addressing is used to branch to within the area between the address 128 bytes higher and that 128 bytes lower relative to the address contained in the PC (program counter). In this addressing mode, the result of a signed addition of the contents of the operand to the PC is stored in the PC. Figure B.2-9 "Example of

relative addressing" shows an example.

**Figure B.2-9 Example of relative addressing**

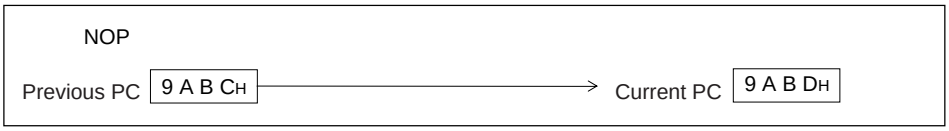



In this example, a branch to the address of the BNE operation code occurs, thus resulting in an infinite loop.

❍ **Inherent addressing**

Inherent addressing is indicated as the addressing without operands in the instruction list. This addressing is used to perform the operation determined by the operation code. In this addressing mode, different operations are performed via different instructions. Figure B.2-10 "Example of inherent addressing" shows an example.

**Figure B.2-10 Example of inherent addressing**



# B.3 Special Instructions

**This section describes the special instructions used for other than addressing.**

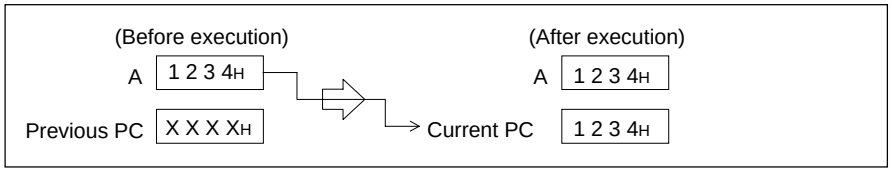
## ■ Special instructions

### ❍ JMP @A

This instruction sets the contents of A (accumulator) to PC (program counter) as the address, and causes a branch to that address. One of the N branch destination addresses is selected from a table, and then transferred to A. The instruction can be executed to perform N-branch processing.

Figure B.3-1 "JMP @A" shows a summary of the instruction.

**Figure B.3-1 JMP @A**

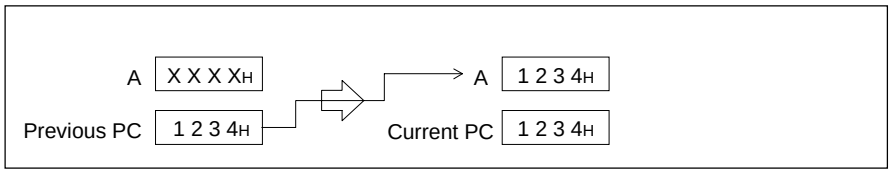



### ❍ MOVW A, PC

This instruction performs the operation which is the reverse of that performed by JMP @A. That is, the instruction stores the contents of PC in A. When the instruction is executed in the main routine, so that a specific subroutine is called, whether A contains a predetermined value can be checked by the subroutine. This can be used to determine that the branch source is not any unexpected section of the program and to check for program runaway.

Figure B.3-2 "MOVW A, PC" shows a summary of the instruction.

**Figure B.3-2 MOVW A, PC**




After the MOVW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of MOVW A, PC. Accordingly, Figure B.3-2 "MOVW A, PC" shows that A contains $1234_H$, which is the address of the operation code of the instruction that follows MOVW A, PC.

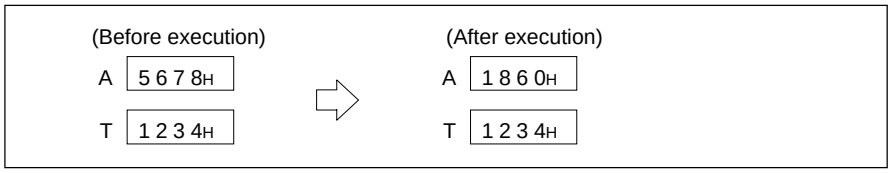
### ❍ MULU A

This instruction performs an unsigned multiplication of AL (lower eight bits of the accumulator) and TL (lower eight bits of the temporary accumulator), and stores the 16-bit result in A. The contents of T (temporary accumulator) do not change. The contents of AH (higher eight bits of the accumulator) and TH (higher eight bits of the temporary accumulator) before execution of the instruction are not used for the operation. The instruction does not change the flags, and

therefore care must be taken when a branch may occur depending on the result of a multiplication.

Figure B.3-3 "MULU" shows a summary of the instruction.

**Figure B.3-3 MULU**



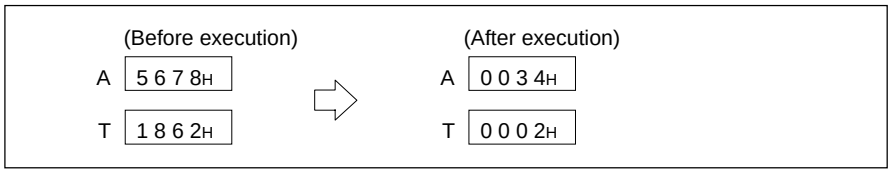

❍ **DIVU A**

This instruction divides the 16-bit value in T by the unsigned 8-bit value in AL, and stores the 8-bit result and the 8-bit remainder in AL and TL, respectively. A value of 0 is set to both AH and TH. The contents of AH before execution of the instruction are not used for the operation. An unpredictable result is produced from data that results in more than eight bits. In addition, there is no indication of the result having more than eight bits. Therefore, if it is likely that data will cause a result of more than eight bits, the data must be checked to ensure that the result will not have more than eight bits before it is used.

The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a division.

Figure B.3-4 "DIVU A" shows a summary of the instruction.

**Figure B.3-4 DIVU A**

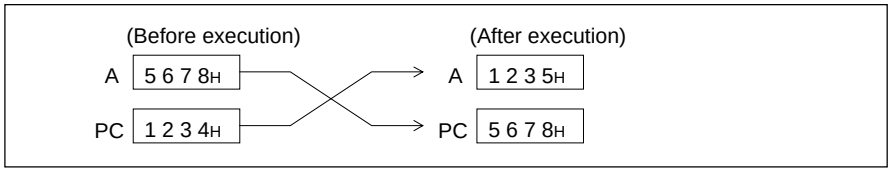



❍ **XCHW A, PC**

This instruction swaps the contents of A and PC, resulting in a branch to the address contained in A before execution of the instruction. After the instruction is executed, A contains the address that follows the address of the operation code of MOVW A, PC. This instruction is effective especially when it is used in the main routine to specify a table for use in a subroutine.

Figure B.3-5 "XCHW A, PC" shows a summary of the instruction.

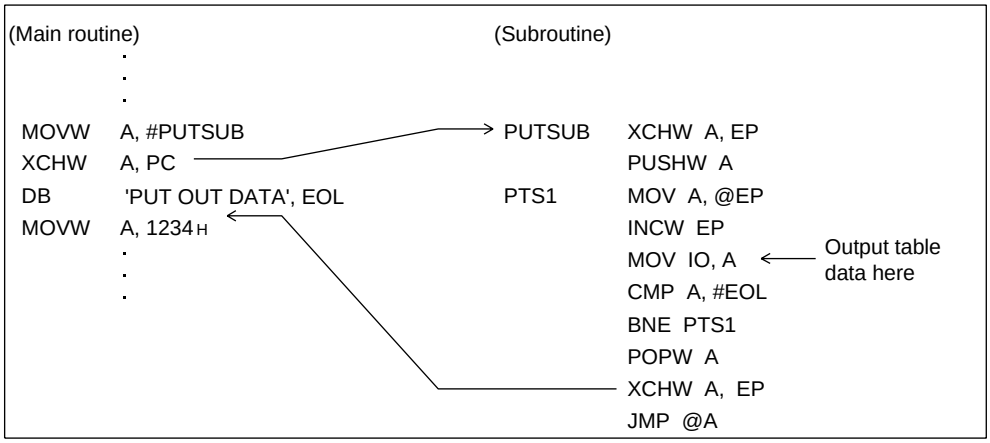
**Figure B.3-5 XCHW A, PC**




After the XCHW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of XCHW A, PC. Accordingly, Figure B.3-5 "XCHW A, PC" shows that A contains $1235_H$, which is the address of the operation code of the instruction that follows XCHW A, PC. This is why $1235_H$ is stored instead of $1234_H$.

Figure B.3-6 "Example of using XCHW A, PC" shows an assembly language example.

**Figure B.3-6 Example of using XCHW A, PC**

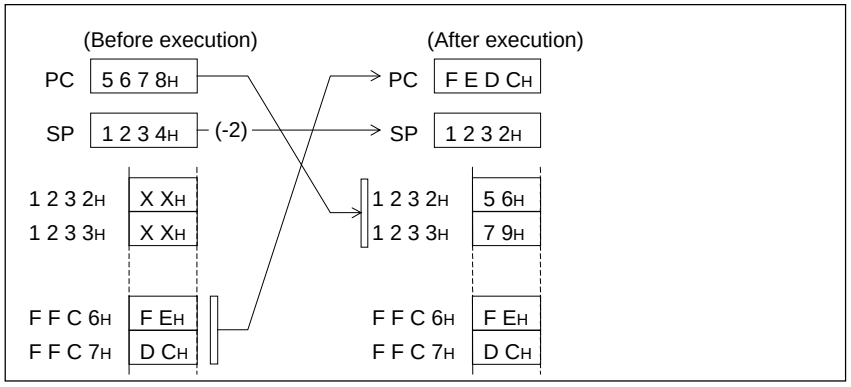



❍ **CALLV #vct**

This instruction is used to branch to a subroutine address stored in the vector table. The instruction saves the return address (contents of PC) in the location at the address contained in SP (stack pointer), and uses vector addressing to cause a branch to the address stored in the vector table. Because CALLV #vct is a 1-byte instruction, the use of this instruction for frequently used subroutines can reduce the entire program size.

Figure B.3-7 "Example of executing CALLV #3" shows a summary of the instruction.

**Figure B.3-7 Example of executing CALLV #3**




After the CALLV #vct instruction is executed, the contents of PC saved on the stack area are the address of the operation code of the next instruction, rather than the address of the operation code of CALLV #vct. Accordingly, Figure B.3-7 "Example of executing CALLV #3" shows that the value saved in the stack ($1232_H$ and $1233_H$) is $5679_H$, which is the address of the operation code of the instruction that follows CALLV #vct (return address).

# B.4 Bit Manipulation Instructions (SETB, CLRB)

**Some bits of peripheral function registers include bits that are read by a bit manipulation instruction differently than usual.**

## ■ Read-modify-write operation

By using these bit manipulation instructions, only the specified bit in a register or RAM location can be set to 1 (SETB) or cleared to 0 (CLRB). However, as the CPU operates on data in 8-bit units, the actual operation (read-modify-write operation) involves a sequence of steps: 8-bit data is read, the specified bit is changed, and the data is written back to the location at the original address.

Table B.4-1 "Bus operation for bit manipulation instructions" shows bus operation for bit manipulation instructions.

**Table B.4-1 Bus operation for bit manipulation instructions**

| CODE | MNEMONIC | TO | Cycle | Address bus | Data bus | $\overline{RD}$ | $\overline{WR}$ | RMW |
|---|---|---|---|---|---|---|---|---|
| A0 to A7 | CLRB dir:b | 4 | 1 | N+1 | Dir | 0 | 1 | 0 |
| | | | 2 | dir address | Data | 0 | 1 | 1 |
| A8 to AF | SETB dir:b | | 3 | dir address | Data | 1 | 0 | 0 |
| | | | 4 | N+2 | Next instruction | 0 | 1 | 0 |

## ■ Read operation upon the execution of bit manipulation instructions

For some I/O ports and for the interrupt request flag bits, the value to be read differs between a normal read operation and a read-modify-write operation.

### ❍ I/O ports (during a bit manipulation)

From some I/O ports, an I/O pin value is read during a normal read operation, while an output latch value is read during a bit manipulation. This prevents the other output latch bits from being changed accidentally, regardless of the I/O directions and states of the pins.

### ❍ Interrupt request flag bits (during a bit manipulation)

An interrupt request flag bit functions as a flag bit indicating whether an interrupt request exists during a normal read operation. However, 1 is always read from this bit during a bit manipulation. This prevents the flag from being cleared accidentally by a value of 0 which would otherwise be written to the interrupt request flag bit when another bit is manipulated.

# B.5 F²MC-8L Instructions

**Table B.5-1 "Transfer Instructions" to Table B.5-4 "Other instructions" list the instructions used with the F²MC-8L.**

## ■ Transfer instructions

**Table B.5-1 Transfer Instructions**

| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | MOV dir, A | 3 | 2 | (dir)<--(A) | - | - | - | - | - | - | - | 45 |
| 2 | MOV @IX+off, A | 4 | 2 | ((IX)+off)<--(A) | - | - | - | - | - | - | - | 46 |
| 3 | MOV ext, A | 4 | 3 | (ext)<--(A) | - | - | - | - | - | - | - | 61 |
| 4 | MOV @EP, A | 3 | 1 | ((EP))<--(A) | - | - | - | - | - | - | - | 47 |
| 5 | MOV Ri, A | 3 | 1 | (Ri)<--(A) | - | - | - | - | - | - | - | 48 to 4F |
| 6 | MOV A, #d8 | 2 | 2 | (A)<--d8 | AL | - | - | + | + | - | - | 04 |
| 7 | MOV A, dir | 3 | 2 | (A)<--(dir) | AL | - | - | + | + | - | - | 05 |
| 8 | MOV A, @IX+off | 4 | 2 | (A)<--((IX)+off) | AL | - | - | + | + | - | - | 06 |
| 9 | MOV A, ext | 4 | 3 | (A)<--(ext) | AL | - | - | + | + | - | - | 60 |
| 10 | MOV A, @A | 3 | 1 | (A)<--((A)) | AL | - | - | + | + | - | - | 92 |
| 11 | MOV A, @EP | 3 | 1 | (A)<--((EP)) | AL | - | - | + | + | - | - | 07 |
| 12 | MOV A, Ri | 3 | 1 | (A)<--(Ri) | AL | - | - | + | + | - | - | 08 to 0F |
| 13 | MOV dir, #d8 | 4 | 3 | (dir)<--d8 | - | - | - | - | - | - | - | 85 |
| 14 | MOV @IX+off, #d8 | 5 | 3 | ((IX)+off)<--d8 | - | - | - | - | - | - | - | 86 |
| 15 | MOV @EP, #d8 | 4 | 2 | ((EP))<--d8 | - | - | - | - | - | - | - | 87 |
| 16 | MOV Ri, #d8 | 4 | 2 | (Ri)<--d8 | - | - | - | - | - | - | - | 88 to 8F |
| 17 | MOVW dir, A | 4 | 2 | (dir)<--(AH), (dir+1)<--(AL) | - | - | - | - | - | - | - | D5 |
| 18 | MOVW @IX+off, A | 5 | 2 | ((IX)+off )<--(AH),<br>((IX)+off+1)<--(AL) | - | - | - | - | - | - | - | D6 |
| 19 | MOVW ext, A | 5 | 3 | (ext)<--(AH), (ext+1)<--(AL) | - | - | - | - | - | - | - | D4 |
| 20 | MOVW @EP, A | 4 | 1 | ((EP))<--(AH),<br>((EP)+1)<--(AL) | - | - | - | - | - | - | - | D7 |
| 21 | MOVW EP, A | 2 | 1 | (EP)<--(A) | - | - | - | - | - | - | - | E3 |
| 22 | MOVW A, #d16 | 3 | 3 | (A)<--d16 | AL | AH | dH | + | + | - | - | E4 |
| 23 | MOVW A, dir | 4 | 2 | (AH)<--(dir), (AL)<--(dir+1) | AL | AH | dH | + | + | - | - | C5 |

**Table B.5-1 Transfer Instructions (Continued)**

| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24 | MOVW A, @IX+off | 5 | 2 | (AH)<--((IX)+off),<br>(AL)<--((IX)+off+1) | AL | AH | dH | + | + | - | - | C6 |
| 25 | MOVW A, ext | 5 | 3 | (AH)<--(ext), (AL)<--(ext+1) | AL | AH | dH | + | + | - | - | C4 |
| 26 | MOVW A, @A | 4 | 1 | (AH)<--((A)), (AL)<--((A)+1) | AL | AH | dH | + | + | - | - | 93 |
| 27 | MOVW A, @EP | 4 | 1 | (AH)<--((EP)),<br>(AL)<--((EP)+1) | AL | AH | dH | + | + | - | - | C7 |
| 28 | MOVW A, EP | 2 | 1 | (A)<--(EP) | - | - | dH | - | - | - | - | F3 |
| 29 | MOVW EP, #d16 | 3 | 3 | (EP)<--d16 | - | - | - | - | - | - | - | E7 |
| 30 | MOVW IX, A | 2 | 1 | (IX)<--(A) | - | - | - | - | - | - | - | E2 |
| 31 | MOVW A, IX | 2 | 1 | (A)<--(IX) | - | - | dH | - | - | - | - | F2 |
| 32 | MOVW SP, A | 2 | 1 | (SP)<--(A) | - | - | - | - | - | - | - | E1 |
| 33 | MOVW A, SP | 2 | 1 | (A)<--(SP) | - | - | dH | - | - | - | - | F1 |
| 34 | MOV @A, T | 3 | 1 | ((A))<--(T) | - | - | - | - | - | - | - | 82 |
| 35 | MOVW @A, T | 4 | 1 | ((A))<--(TH), ((A)+1)<--(TL) | - | - | - | - | - | - | - | 83 |
| 36 | MOVW IX, #d16 | 3 | 3 | (IX)<--d16 | - | - | - | - | - | - | - | E6 |
| 37 | MOVW A, PS | 2 | 1 | (A)<--(PS) | - | - | dH | - | - | - | - | 70 |
| 38 | MOVW PS, A | 2 | 1 | (PS)<--(A) | - | - | - | + | + | + | + | 71 |
| 39 | MOVW SP, #d16 | 3 | 3 | (SP)<--d16 | - | - | - | - | - | - | - | E5 |
| 40 | SWAP | 2 | 1 | (AH)<-- -->(AL) | - | - | AL | - | - | - | - | 10 |
| 41 | SETB dir:b | 4 | 2 | (dir):b <--1 | - | - | - | - | - | - | - | A8 to AF |
| 42 | CLRB dir:b | 4 | 2 | (dir):b <--0 | - | - | - | - | - | - | - | A0 to A7 |
| 43 | XCH A, T | 2 | 1 | (AL)<-- -->(TL) | AL | - | - | - | - | - | - | 42 |
| 44 | XCHW A, T | 3 | 1 | (A)<-- -->(T) | AL | AH | dH | - | - | - | - | 43 |
| 45 | XCHW A, EP | 3 | 1 | (A)<-- -->(EP) | - | - | dH | - | - | - | - | F7 |
| 46 | XCHW A, IX | 3 | 1 | (A)<-- -->(IX) | - | - | dH | - | - | - | - | F6 |
| 47 | XCHW A, SP | 3 | 1 | (A)<-- -->(SP) | - | - | dH | - | - | - | - | F5 |
| 48 | MOVW A, PC | 2 | 1 | (A)<--(PC) | - | - | dH | - | - | - | - | F0 |

**Caution:**

In automatic transfer to T during byte transfer to A, AL is transferred to TL.

If an instruction has two or more operands, they are assumed to be saved in the order indicated by MNEMONIC.

## ■ Arithmetic instructions

### Table B.5-2 Arithmetic Operation Instructions

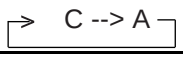
| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ADDC A, Ri | 3 | 1 | (A)<--(A)+(Ri)+C | - | - | - | + | + | + | + | 28 to 2F |
| 2 | ADDC A, #d8 | 2 | 2 | (A)<--(A)+d8+C | - | - | - | + | + | + | + | 24 |
| 3 | ADDC A, dir | 3 | 2 | (A)<--(A)+(dir)+C | - | - | - | + | + | + | + | 25 |
| 4 | ADDC A, @IX+off | 4 | 2 | (A)<--(A)+((IX)+off)+C | - | - | - | + | + | + | + | 26 |
| 5 | ADDC A, @EP | 3 | 1 | (A)<--(A)+((EP))+C | - | - | - | + | + | + | + | 27 |
| 6 | ADDCW A | 3 | 1 | (A)<--(A)+(T)+C | - | - | dH | + | + | + | + | 23 |
| 7 | ADDC A | 2 | 1 | (AL)<--(AL)+(TL)+C | - | - | - | + | + | + | + | 22 |
| 8 | SUBC A, Ri | 3 | 1 | (A)<--(A)-(Ri)-C | - | - | - | + | + | + | + | 38 to 3F |
| 9 | SUBC A, #d8 | 2 | 2 | (A)<--(A)-d8-C | - | - | - | + | + | + | + | 34 |
| 10 | SUBC A, dir | 3 | 2 | (A)<--(A)-(dir)-C | - | - | - | + | + | + | + | 35 |
| 11 | SUBC A, @IX+off | 4 | 2 | (A)<--(A)-((IX)+off)-C | - | - | - | + | + | + | + | 36 |
| 12 | SUBC A, @EP | 3 | 1 | (A)<--(A)-((EP))-C | - | - | - | + | + | + | + | 37 |
| 13 | SUBCW A | 3 | 1 | (A)<--(T)-(A)-C | - | - | dH | + | + | + | + | 33 |
| 14 | SUBC A | 2 | 1 | (AL)<--(TL)-(AL)-C | - | - | - | + | + | + | + | 32 |
| 15 | INC Ri | 4 | 1 | (Ri)<--(Ri)+1 | - | - | - | + | + | + | - | C8 to CF |
| 16 | INCW EP | 3 | 1 | (EP)<--(EP)+1 | - | - | - | - | - | - | - | C3 |
| 17 | INCW IX | 3 | 1 | (IX)<--(IX)+1 | - | - | - | - | - | - | - | C2 |
| 18 | INCW A | 3 | 1 | (A)<--(A)+1 | - | - | dH | + | + | - | - | C0 |
| 19 | DEC Ri | 4 | 1 | (Ri)<--(Ri)-1 | - | - | - | + | + | + | - | D8 to DF |
| 20 | DECW EP | 3 | 1 | (EP)<--(EP)-1 | - | - | - | - | - | - | - | D3 |
| 21 | DECW IX | 3 | 1 | (IX)<--(IX)-1 | - | - | - | - | - | - | - | D2 |
| 22 | DECW A | 3 | 1 | (A)<--(A)-1 | - | - | dH | + | + | - | - | D0 |
| 23 | MULU A | 19 | 1 | (A)<--(AL)x(TL) | - | - | dH | - | - | - | - | 01 |
| 24 | DIVU A | 21 | 1 | (A)<--(T)/(AL), MOD -->(T) | dL | 00 | 00 | - | - | - | - | 11 |
| 25 | ANDW A | 3 | 1 | (A)<--(A) ∧ (T) | - | - | dH | + | + | R | - | 63 |
| 26 | ORW A | 3 | 1 | (A)<--(A) ∨ (T) | - | - | dH | + | + | R | - | 73 |
| 27 | XORW A | 3 | 1 | (A)<--(A) ∀ (T) | - | - | dH | + | + | R | - | 53 |
| 28 | CMP A | 2 | 1 | (TL)-(AL) | - | - | - | + | + | + | + | 12 |
| 29 | CMPW A | 3 | 1 | (T)-(A) | - | - | - | + | + | + | + | 13 |
| 30 | RORC A | 2 | 1 |   | - | - | - | + | + | - | + | 03 |

**Table B.5-2 Arithmetic Operation Instructions (Continued)**

| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | ROLC A | 2 | 1 | C <-- A | - | - | - | + | + | - | + | 02 |
| 32 | CMP A, #d8 | 2 | 2 | (A)-d8 | - | - | - | + | + | + | + | 14 |
| 33 | CMP A, dir | 3 | 2 | (A)-(dir) | - | - | - | + | + | + | + | 15 |
| 34 | CMP A, @EP | 3 | 1 | (A)-((EP)) | - | - | - | + | + | + | + | 17 |
| 35 | CMP A, @IX+off | 4 | 2 | (A)-((IX)+off) | - | - | - | + | + | + | + | 16 |
| 36 | CMP A, Ri | 3 | 1 | (A)-(Ri) | - | - | - | + | + | + | + | 18 to 1F |
| 37 | DAA | 2 | 1 | decimal adjust for addition | - | - | - | + | + | + | + | 84 |
| 38 | DAS | 2 | 1 | decimal adjust for subtraction | - | - | - | + | + | + | + | 94 |
| 39 | XOR A | 2 | 1 | (A)<--(AL) ∀ (TL) | - | - | - | + | + | R | - | 52 |
| 40 | XOR A, #d8 | 2 | 2 | (A)<--(AL) ∀ d8 | - | - | - | + | + | R | - | 54 |
| 41 | XOR A, dir | 3 | 2 | (A)<--(AL) ∀ (dir) | - | - | - | + | + | R | - | 55 |
| 42 | XOR A, @EP | 3 | 1 | (A)<--(AL) ∀ ((EP)) | - | - | - | + | + | R | - | 57 |
| 43 | XOR A, @IX+off | 4 | 2 | (A)<--(AL) ∀ ((IX)+off) | - | - | - | + | + | R | - | 56 |
| 44 | XOR A, Ri | 3 | 1 | (A)<--(AL) ∀ (Ri) | - | - | - | + | + | R | - | 58 to 5F |
| 45 | AND A | 2 | 1 | (A)<--(AL) ∧ (TL) | - | - | - | + | + | R | - | 62 |
| 46 | AND A, #d8 | 2 | 2 | (A)<--(AL) ∧ d8 | - | - | - | + | + | R | - | 64 |
| 47 | AND A, dir | 3 | 2 | (A)<--(AL) ∧ (dir) | - | - | - | + | + | R | - | 65 |
| 48 | AND A, @EP | 3 | 1 | (A)<--(AL) ∧ ((EP)) | - | - | - | + | + | R | - | 67 |
| 49 | AND A, @IX+off | 4 | 2 | (A)<--(AL) ∧ ((IX)+off) | - | - | - | + | + | R | - | 66 |
| 50 | AND A, Ri | 3 | 1 | (A)<--(AL) ∧ (Ri) | - | - | - | + | + | R | - | 68 to 6F |
| 51 | OR A | 2 | 1 | (A)<--(AL) ∨ (TL) | - | - | - | + | + | R | - | 72 |
| 52 | OR A, #d8 | 2 | 2 | (A)<--(AL) ∨ d8 | - | - | - | + | + | R | - | 74 |
| 53 | OR A, dir | 3 | 2 | (A)<--(AL) ∨ (dir) | - | - | - | + | + | R | - | 75 |
| 54 | OR A, @EP | 3 | 1 | (A)<--(AL) ∨ ((EP)) | - | - | - | + | + | R | - | 77 |
| 55 | OR A, @IX+off | 4 | 2 | (A)<--(AL) ∨ ((IX)+off) | - | - | - | + | + | R | - | 76 |
| 56 | OR A, Ri | 3 | 1 | (A)<--(AL) ∨ (Ri) | - | - | - | + | + | R | - | 78 to 7F |
| 57 | CMP dir, #d8 | 5 | 3 | (dir)-d8 | - | - | - | + | + | + | + | 95 |
| 58 | CMP @EP, #d8 | 4 | 2 | ((EP))-d8 | - | - | - | + | + | + | + | 97 |
| 59 | CMP @IX+off, #d8 | 5 | 3 | ((IX)+off)-d8 | - | - | - | + | + | + | + | 96 |
| 60 | CMP Ri, #d8 | 4 | 2 | (Ri)-d8 | - | - | - | + | + | + | + | 98 to 9F |
| 61 | INCW SP | 3 | 1 | (SP)<--(SP)+1 | - | - | - | - | - | - | - | C1 |
| 62 | DECW SP | 3 | 1 | (SP)<--(SP)-1 | - | - | - | - | - | - | - | D1 |

## ■ Branch instructions

**Table B.5-3 Branch instructions**

| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | BZ/BEQ rel | 3 | 2 | if Z=1 then PC<--PC+rel | - | - | - | - | - | - | - | FD |
| 2 | BNZ/BNE rel | 3 | 2 | if Z=0 then PC<--PC+rel | - | - | - | - | - | - | - | FC |
| 3 | BC/BLO rel | 3 | 2 | if C=1 then PC<--PC+rel | - | - | - | - | - | - | - | F9 |
| 4 | BNC/BHS rel | 3 | 2 | if C=0 then PC<--PC+rel | - | - | - | - | - | - | - | F8 |
| 5 | BN rel | 3 | 2 | if N=1 then PC<--PC+rel | - | - | - | - | - | - | - | FB |
| 6 | BP rel | 3 | 2 | if N=0 then PC<--PC+rel | - | - | - | - | - | - | - | FA |
| 7 | BLT rel | 3 | 2 | if V ∀ N=1 then PC<--PC+rel | - | - | - | - | - | - | - | FF |
| 8 | BGE rel | 3 | 2 | if V ∀ N=0 then PC<--PC+rel | - | - | - | - | - | - | - | FE |
| 9 | BBC dir:b, rel | 5 | 3 | if (dir:b)=0 then PC<--PC+rel | - | - | - | - | + | - | - | B0 to B7 |
| 10 | BBS dir:b, rel | 5 | 3 | if (dir:b)=1 then PC<--PC+rel | - | - | - | - | + | - | - | B8 to BF |
| 11 | JMP @A | 2 | 1 | (PC)<--(A) | - | - | - | - | - | - | - | E0 |
| 12 | JMP ext | 3 | 3 | (PC)<--ext | - | - | - | - | - | - | - | 21 |
| 13 | CALLV #vct | 6 | 1 | vector call | - | - | - | - | - | - | - | E8 to EF |
| 14 | CALL ext | 6 | 3 | subroutine call | - | - | - | - | - | - | - | 31 |
| 15 | XCHW A, PC | 3 | 1 | (PC)<--(A), (A)<--(PC)+1 | - | - | dH | - | - | - | - | F4 |
| 16 | RET | 4 | 1 | return from subroutine | - | - | - | - | - | - | - | 20 |
| 17 | RETI | 6 | 1 | return from interrupt | - | - | - | restore | | | | 30 |

## ■ Other instructions

**Table B.5-4 Other Instructions**

| No. | MNEMONIC | ~ | # | Operation | TL | TH | AH | N | Z | V | C | OP CODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | PUSHW A | 4 | 1 | | - | - | - | - | - | - | - | 40 |
| 2 | POPW A | 4 | 1 | | - | - | dH | - | - | - | - | 50 |
| 3 | PUSHW IX | 4 | 1 | | - | - | - | - | - | - | - | 41 |
| 4 | POPW IX | 4 | 1 | | - | - | - | - | - | - | - | 51 |
| 5 | NOP | 1 | 1 | | - | - | - | - | - | - | - | 00 |
| 6 | CLRC | 1 | 1 | | - | - | - | - | - | - | R | 81 |
| 7 | SETC | 1 | 1 | | - | - | - | - | - | - | S | 91 |
| 8 | CLRI | 1 | 1 | | - | - | - | - | - | - | - | 80 |
| 9 | SETI | 1 | 1 | | - | - | - | - | - | - | - | 90 |

# B.6 Instruction Map

## Table B.6-1 "F$^2$MC-8L Instruction Map" shows the F$^2$MC-8L instruction map.

■ **Instruction map**

**Table B.6-1 F$^2$MC-8L Instruction Map**

| H / L | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NOP | SWAP | RET | RETI | PUSHW A | POPW A | MOV A, ext | MOVW A, PS | CLRI | SETI | CLRB dir : 0 | BBC dir : 0, rel | INCW A | DECW A | JMP @A | MOVW A, PC |
| 1 | MULU A | DIVU A | JMP addr16 | CALL addr16 | PUSHW IX | POPW IX | MOV ext, A | MOVW PS, A | CLRC | SETC | CLRB dir : 1 | BBC dir : 1, rel | INCW SP | DECW SP | MOVW SP, A | MOVW A, SP |
| 2 | ROLC A | CMP A | ADDC A | SUBC A | XCH A, T | XOR A | AND A | OR A | MOV @A, T | MOV A, @A | CLRB dir : 2 | BBC dir : 2, rel | INCW IX | DECW IX | MOVW IX, A | MOVW A, IX |
| 3 | RORC A | CMPW A | ADDCW A | SUBCW A | XCHW A, T | XORW A | ANDW A | ORW A | MOVW @A, T | MOVW A, @A | CLRB dir : 3 | BBC dir : 3, rel | INCW EP | DECW EP | MOVW EP, A | MOVW A, EP |
| 4 | MOV A, #d8 | CMP A, #d8 | ADDC A, #d8 | SUBC A, #d8 | / | XOR A, #d8 | AND A, #d8 | OR A, #d8 | DAA | DAS | CLRB dir : 4 | BBC dir : 4, rel | MOVW A, ext | MOVW ext, A | MOVW A, #d16 | XCHW A, PC |
| 5 | MOV A, dir | CMP A, dir | ADDC A, dir | SUBC A, dir | MOV dir, A | XOR A, dir | AND A, dir | OR A, dir | MOV dir, #d8 | CMP dir, #d8 | CLRB dir : 5 | BBC dir : 5, rel | MOVW A, dir | MOVW dir, A | MOVW SP, #d16 | XCHW A, SP |
| 6 | MOV A, @IX+d | CMP A, @IX+d | ADDC A, @IX+d | SUBC A, @IX+d | MOV @IX+d, A | XOR A, @IX+d | AND A, @IX+d | OR A, @IX+d | MOV @IX+d, #d8 | CMP @IX+d, #d8 | CLRB dir : 6 | BBC dir : 6, rel | MOVW A, @IX+d | MOVW @IX+d, A | MOVW IX, #d16 | XCHW A, IX |
| 7 | MOV A, @EP | CMP A, @EP | ADDC A, @EP | SUBC A, @EP | MOV @EP, A | XOR A, @EP | AND A, @EP | OR A, @EP | MOV @EP#, d8 | CMP @EP#, d8 | CLRB dir : 7 | BBC dir : 7, rel | MOVW A, @EP | MOVW @EP, A | MOVW EP, #d16 | XCHW A, EP |
| 8 | MOV A, R0 | CMP A, R0 | ADDC A, R0 | SUBC A, R0 | MOV R0, A | XOR A, R0 | AND A, R0 | OR A, R0 | MOV R0, #d8 | CMP R0, #d8 | SETB dir : 0 | BBS dir : 0, rel | INC R0 | DEC R0 | CALLV #0 | BNC rel |
| 9 | MOV A, R1 | CMP A, R1 | ADDC A, R1 | SUBC A, R1 | MOV R1, A | XOR A, R1 | AND A, R1 | OR A, R1 | MOV R1, #d8 | CMP R1, #d8 | SETB dir : 1 | BBS dir : 1, rel | INC R1 | DEC R1 | CALLV #1 | BC rel |
| A | MOV A, R2 | CMP A, R2 | ADDC A, R2 | SUBC A, R2 | MOV R2, A | XOR A, R2 | AND A, R2 | OR A, R2 | MOV R2, #d8 | CMP R2, #d8 | SETB dir : 2 | BBS dir : 2, rel | INC R2 | DEC R2 | CALLV #2 | BP rel |
| B | MOV A, R3 | CMP A, R3 | ADDC A, R3 | SUBC A, R3 | MOV R3, A | XOR A, R3 | AND A, R3 | OR A, R3 | MOV R3, #d8 | CMP R3, #d8 | SETB dir : 3 | BBS dir : 3, rel | INC R3 | DEC R3 | CALLV #3 | BN rel |
| C | MOV A, R4 | CMP A, R4 | ADDC A, R4 | SUBC A, R4 | MOV R4, A | XOR A, R4 | AND A, R4 | OR A, R4 | MOV R4, #d8 | CMP R4, #d8 | SETB dir : 4 | BBS dir : 4, rel | INC R4 | DEC R4 | CALLV #4 | BNZ rel |
| D | MOV A, R5 | CMP A, R5 | ADDC A, R5 | SUBC A, R5 | MOV R5, A | XOR A, R5 | AND A, R5 | OR A, R5 | MOV R5, #d8 | CMP R5, #d8 | SETB dir : 5 | BBS dir : 5, rel | INC R5 | DEC R5 | CALLV #5 | BZ rel |
| E | MOV A, R6 | CMP A, R6 | ADDC A, R6 | SUBC A, R6 | MOV R6, A | XOR A, R6 | AND A, R6 | OR A, R6 | MOV R6, #d8 | CMP R6, #d8 | SETB dir : 6 | BBS dir : 6, rel | INC R6 | DEC R6 | CALLV #6 | BGE rel |
| F | MOV A, R7 | CMP A, R7 | ADDC A, R7 | SUBC A, R7 | MOV R7, A | XOR A, R7 | AND A, R7 | OR A, R7 | MOV R7, #d8 | CMP R7, #d8 | SETB dir : 7 | BBS dir : 7, rel | INC R7 | DEC R7 | CALLV #7 | BLT rel |

# APPENDIX C Mask Options

**This appendix shows a list of mask options for the MB89570 series.**

## ■ List of Mask Options

**Table C-1 List of Mask Options**

| No. | Model | MB89577 | MB89P579A | MB89PV570 |
|---|---|---|---|---|
| | Specification method | To be specified when ordering a mask | To be specified when ordering | To be specified when ordering |
| 1 | Selecting the initial value(*1) for the main clock oscillation stabilization wait time (Fch = 10 MHz)<br>• 01: $2^{14}$/Fch (approx. 1.63 ms)<br>• 10: $2^{17}$/Fch (approx. 13.1 ms)<br>• 11: $2^{18}$/Fch (approx. 26.2 ms) | Selectable | $2^{18}$/Fch (approx. 26.2 ms) | $2^{18}$/Fch (approx. 26.2 ms) |

*1: Shall be used as the initial value upon reset for the oscillation stabilization wait time select bit of the system clock control register (SYCC: WT1, WT0).

# APPENDIX D One-time PROM and EPROM Microcontroller Write Specification

**The MB89P579, equipped with the PROM mode, allows a general-purpose ROM programmer to write data using a dedicated adapter.**

## ■ ROM Programmer Adapters and Recommended ROM Programmers

The following shows ROM programmer adapters and recommended ROM programmers.

**Table D-1 ROM Programmer Adapters and Recommended ROM Programmers**

| Package name | Applicable adapter model | Recommended programmer maker and programmer | |
|---|---|---|---|
| | Sun Hayato Co., Ltd. | Minato electronics Co., Ltd. | |
| | | MODEL1890A | |
| FPT-100P-M05 | ROM2-100LQF-32DP-8LA | Under evaluation | |
| FPT-100P-M18 | ROM2-100TQF2-32DP-8LA | Under evaluation | |

- Contact information

  Sun Hayato Co., Ltd.: Phone (81) 3-3986-0403

  Minato electronics Co., Ltd.: Phone (81) 45-591-5611

Writing data to the EPROM

1. Set the EPROM programmer for the CU50-OTP (device code: cdB86DC).
2. Load the program data to the EPROM programmer.
3. Write data using the EPROM programmer.

# APPENDIX E Pin Statuses of the MB89570 Series

**This appendix shows the pin statuses during various operations of the MB89570 series.**

## ■ Pin Statuses during Various Operations

### Table E-1 Pin Statuses during Various Operations

| Pin name | Normal operation | Sleep mode | Stop mode SPL=0 | Stop mode SPL=1 | During reset |
|---|---|---|---|---|---|
| P80/INTO to P83/INT3 | Port input-output/ Resource input-output | Hold/External interrupt input | Hold/External interrupt input | Hi-Z/External interrupt input | Hi-Z |
| X0, X0A | Oscillation input | Oscillation input | Hi-Z | Hi-Z | Oscillation input |
| X1, X1A | Oscillation output | Oscillation output | "H" output | "H" output | Oscillation output |
| MODA | Mode input | Mode input | Mode input | Mode input | Mode input |
| $\overline{\text{RST}}$ | Reset input-output | Reset input-output | Reset input-output | Reset input-output | Reset input-output |
| P00 to P07 | Port input-output/ Resource input-output | Hold/Resource input-output | Hold | Hi-Z | Hi-Z |
| P10/AN4 to P17/AN11 | | | | | |
| P20/TO1 | | | | Hold | |
| P23/TO2 | | | | | |
| P21,P22,P24 to P27 | | | | Hi-Z | |
| P30/SLC1 | | | | | |
| P31/SDA1 | | | | | |
| P32/ALERT | | | | | |
| P33/SCL2/UCK3 | | | | | |
| P34/SDA2/UI3 | | | | | |
| P35/UO3 | | | | | |
| P40/SCL3/UCK1 | | | | | |
| P41/SDA3/UI1 | | | | | |
| P42/SCL4/UCK2 | | | | | |
| P43/SDA4/UI2 | | | | | |
| P50/ALR1 to P52/ALR3 | | | | | |
| P53/ACO | | | | | |
| P54/OFB1 to P56/OFB3 | | | | | |
| P70/DCIN | | | | | |
| P71/DCIN2 | | | | | |
| P72/VOL1 | | | | | |
| P73/VSI1 | | | | | |

## Table E-1 Pin Statuses during Various Operations (Continued)

<table>
<tr><th>Pin name</th><th>Normal operation</th><th>Sleep mode</th><th>Stop mode<br>SPL=0</th><th>Stop mode<br>SPL=1</th><th>During reset</th></tr>
<tr><td>P74/VOL2</td><td rowspan="11">Port input-output/<br>Resource input-<br>output</td><td rowspan="11">Hold/Resource<br>input-output</td><td rowspan="11">Hold</td><td rowspan="6">Hi-Z</td><td rowspan="6">Hi-Z</td></tr>
<tr><td>P75/VSI2</td></tr>
<tr><td>P76/VOL3</td></tr>
<tr><td>P77/VSI3</td></tr>
<tr><td>P85/AN0/SW1 to P87/AN2/SW3</td></tr>
<tr><td>P90/AN3</td></tr>
<tr><td>P91/DA1 to P92/DA2</td><td rowspan="5">Hi-Z(*1)</td></tr>
<tr><td>P60/SEG08 to P65/SEG13/UO1</td><td>Hi-Z</td></tr>
<tr><td>PB4/COM0 to PB7/COM3</td><td rowspan="3">"L" output pin</td></tr>
<tr><td>PA0/SEG00 to PA7/SEG07</td></tr>
<tr><td>P84/EC</td><td colspan="2">Port input/Resource input</td><td colspan="3">Hi-Z</td></tr>
<tr><td>PB0/V0 to PB3/V3</td><td>Port output/<br>Resource input</td><td>Hold/Resource input</td><td>Hold/Resource input</td><td>Hi-Z(*1)</td><td>Resource input</td></tr>
</table>

Hi-Z: High impedance
SPL: Pin status specification bit of the standby control register (STBC)
Hold: The pin specified for output holds the pin status (level) immediately before the mode is entered.
*1: Hi-Z does not occur while LCDC is selected.

# INDEX

**The index follows on the next page.**
**This is listed in alphabetic order.**

# Index


**Numerics**

1/2 bias, 1/2 duty output waveform, example of... 479
1/3 bias, 1/3 duty output waveform, example of... 482
1/3 bias, 1/4 duty output waveform, example of... 485
16-bit data on RAM, storage of ..... 30
16-bit data on stack, storage of ..... 31
16-bit operand, storage of ..... 30
16-bit timer/counter and vector table, register related to interrupts of ..... 243
16-bit timer/counter, block diagram of ..... 236
16-bit timer/counter, block diagram of pin related to ..... 238
16-bit timer/counter, note on using ..... 249
16-bit timer/counter, pin related to ..... 238
16-bit timer/counter, register related to ..... 239
8/16-bit timer/counter interrupt ..... 218
8/16-bit timer/counter, block diagram of ..... 203
8/16-bit timer/counter, note on using ..... 229
8-bit receiving operation at operation mode 1 ..... 357
8-bit transmitting operation at operation mode 1.. 359
8-segment LCD panel and display data example (1/4 duty driving method), connection example of ..... 487

**A**

A/D control register 1 (ADC1) ..... 274
A/D control register 2 (ADC2) ..... 276
A/D conversion function ..... 266
A/D conversion function, interrupt for ..... 281
A/D conversion function, operation of ..... 283
A/D conversion function, program example of ..... 286
A/D conversion function, starting ..... 282
A/D converter interrupt, register and vector table related to ..... 281
A/D converter, block diagram of ..... 267
A/D converter, note on using ..... 284
A/D data register (ADDH, ADDL) ..... 278
A/D enable register 1 to 2 (ADEN1 to 2) ..... 279
acknowledge ..... 393, 433
address comparison enable register (WREN) ..... 500
addressing ..... 393, 433
adjusting brightness of LCD when internal dividing resistor is used ..... 460
arbitration ..... 394, 434
arithmetic instruction ..... 524
arithmetic operation result bit ..... 34

**B**

battery 1 to 3 VALID interrupt ..... 327
baud rate generator reload register (SRC) ..... 344
bit manipulation instruction, read operation upon the execution of ..... 521
block diagram of 8/16-bit timer/counter ..... 203
block diagram of pin related to 8/16-bit timer/counter ..... 206
block diagram of pin related to I2C ..... 371
block diagram of pin related to multi-address I2C ..... 409
block diagram, I2C ..... 367
block diagram, multi-address I2C ..... 405
branch instruction ..... 526
bridge circuit ..... 442
bridge circuit block diagram ..... 443
bridge circuit selection register 1 (BRSR1) ..... 448
bridge circuit selection register 3 (BRSR3) ..... 452
bridge circuit, block diagram of pin related to ..... 445
bridge circuit, pin related to ..... 444
bridge circuit, registers related to ..... 447
bridge selection register 2 (BRSR2) ..... 450

**C**

clock controller, block diagram of ..... 62
clock generator ..... 60
clock mode, operating state of ..... 67
clock supply function ..... 161, 186
clock supply function, operation of ..... 167, 193
clock supply map ..... 58
clock timeout ..... 396, 436
comparator 1, voltage comparators 2 to 7 interrupt ..... 327
comparator 2 to 4 interrupt ..... 327
comparator block diagram ..... 299
comparator control register 1 (COCR1) ..... 308
comparator control register 2 (COCR2) ..... 310
comparator input enable register (CIER) ..... 325
comparator interrupt control register 1 (CICR1) ..... 315
comparator interrupt control register 2 (CICR2) ..... 319
comparator status register 1 (COSR1) ..... 312
comparator status register 2 (COSR2) ..... 317
comparator status register 3 (COSR3) ..... 321

comparator status register 4 (COSR4) ..... 323
condition code register (CCR), structure of..... 34
continuous receiving operation ..... 358
continuous transmission at operation mode 1 ..... 361
counter function ..... 202, 235
counter function mode, interrupt in ..... 243
counter function, operation of ..... 222, 246
counter function, program example of ..... 252

**D**

D/A control register (DACR)..... 294
D/A converter..... 290
D/A converter (DA1, 2), block diagram of pin related to..... 292
D/A converter (DA1, 2), pin related to..... 292
D/A converter block, block diagram of ..... 291
D/A converter operation..... 296
D/A converter, register related to..... 293
D/A data register 1 and 2 (DADR1, 2) ..... 295
data setting register (WRDR)..... 494
data timeout ..... 396, 436
data transfer..... 393, 433
dedicated register configuration..... 32
dedicated register function..... 32
detecting start bit at receiving operation ..... 354

**E**

error ..... 399, 439
explanation of addressing..... 514
external dividing resistor ..... 461
external interrupt circuit, block diagram of ..... 257
external interrupt circuit, operation of..... 264
external interrupt control register (EIC1)..... 261
external interrupt function ..... 256
external reset pin function..... 54
external reset pin, block diagram of..... 54

**F**

FPT-100P-M05, package dimensions of..... 10
FPT-100P-M18, package dimension of ..... 11
function of I/O port ..... 88
function of register of port 0 ..... 93
function of register of port 1 ..... 98
function of register of port 2 ..... 104
function of register of port 3 ..... 111
function of register of port 4 ..... 115
function of register of port 5 ..... 119
function of register of port 6 ..... 125
function of register of port 7.....130
function of register of port 8.....137
function of register of port 9.....144
function of register of port A .....150
function of register of port B .....155
function of UART/SIO .....334

**G**

gear function (function for switching speed of main clock) .....68
general-purpose register area .....28
general-purpose register, feature of .....39
general-purpose register, structure of .....38

**I**

I/O circuit type.....18
I/O map.....504
I/O port, function of.....88
I/O port, program example of.....157
I2C address register (IADR) .....381
I2C block diagram.....367
I2C bus control register (IBCR) .....376
I2C bus status register (IBSR).....374
I2C clock control register (ICCR).....379
I2C data register (IDAR).....382
I2C function .....364
I2C interface register, precaution in setting .....395
I2C master timeout register (IMTO).....389
I2C protocol .....392
I2C slave timeout register (ISTO) .....390
I2C system.....392
I2C timeout clock register (ITOC) .....388
I2C timeout control register (ITCR).....383
I2C timeout data register (ITOD) .....387
I2C timeout status register (ITSR) .....385
I2C, block diagram of pin related to.....371
I2C, pin related to .....370
I2C, register and vector table address related to interrupt of .....391
I2C, register related to .....372
instruction cycle (tinst) .....66
instruction map .....528
internal dividing resistor.....459
interrupt acceptance control bit .....36
interrupt at bus error .....391, 430
interrupt at data transfer completion.....391, 430
interrupt at timeout detection .....391, 430
interrupt level setting register (ILR1, ILR2, ILR3, ILR4), structure of .....42

interrupt processing ..... 44
interrupt processing time ..... 47
interrupt request from peripheral function ..... 40
interrupt when external interrupt circuit is operating ..... 263
interrupt when interval timer function is active ..... 166
interrupt when interval timer function is active (watch interrupt) ..... 192
interval timer function ..... 160, 200, 234
interval timer function (timebase timer), operation of ..... 167
interval timer function (watch interrupt) ..... 186
interval timer function (watch prescaler), operation of ..... 193
interval timer function mode, interrupt in ..... 243
interval timer function, operation of ..... 244
interval timer function, program example of ..... 250
items in the instruction list ..... 513

**L**

LCD controller driver block diagram ..... 457
LCD controller driver function ..... 456
LCD controller driver operation, explanation of ..... 477
LCD controller driver, power supply voltage of ..... 458
LCD display RAM and output pin ..... 475
LCD driving waveform ..... 478
LCD panel connection example and display data example (1/2 duty driving method) ..... 481
LCD panel connection example and display data example (1/3 duty driving method) ..... 484
LCDC control register 1 (LCR1) ..... 466
LCDC control register 2 (LCR2) ..... 469
LCDC control register 3 (LCR3) ..... 471
LCDC control register 4 (LCR4) ..... 473
low power consumption (standby) mode and when counter is suspended, operation in ..... 248
lower address setting register (WRARL) ..... 498

**M**

main clock mode, operation in ..... 68
main clock oscillation stabilization delay time and reset source ..... 51
main clock, oscillation stabilization wait time of ..... 70
mask option, list of ..... 529
master timeout ..... 397, 437
MB89570 series (FPT-100P-M05, FPT-100P-M18, MQP-100C-P02), pin assignment of ..... 9
MB89570 series, block diagram of ..... 8
MB89570 series, feature ..... 2
MB89570 series, product lineup in ..... 5
memory access mode, selection of ..... 86
memory map ..... 27
memory space, configuration of ..... 26
mode data ..... 85
mode fetch ..... 56
mode pin ..... 56
mode pin (MODA) ..... 85
MQP-100C-P02, package dimension of ..... 12
multi-address I2C address register (MADR1 to 6) ..... 419
multi-address I2C ALERT register (MALR) ..... 429
multi-address I2C block diagram ..... 405
multi-address I2C bus control register (MBCR) ..... 414
multi-address I2C bus status register (MBSR) ..... 412
multi-address I2C clock control register (MCCR) ..... 417
multi-address I2C data register (MDAR) ..... 420
multi-address I2C function ..... 402
multi-address I2C master timeout register (MMTO) ..... 427
multi-address I2C protocol ..... 432
multi-address I2C slave timeout register (MSTO) ..... 428
multi-address I2C system ..... 432
multi-address I2C timeout clock register (MTOC) ..... 426
multi-address I2C timeout control register (MTCR) ..... 421
multi-address I2C timeout data register (MTOD) ..... 425
multi-address I2C timeout status register (MTSR) ..... 423
multi-address I2C, block diagram of pin related to ..... 409
multi-address I2C, pin related to ..... 408
multi-address I2C, register and vector table address related to interrupt of ..... 431
multi-address I2C, register related to ..... 410
multiple interrupt ..... 46

**N**

Note on Handling Device ..... 24
note on using 16-bit timer/counter ..... 249
note on using 8/16-bit timer/counter ..... 229
note on using timebase timer ..... 169
note on using watch prescaler ..... 195
note on using watchdog timer ..... 181

**O**

operation in low power consumption (standby) mode and when counter is suspended ..... 248

operation in main clock mode ..... 68
operation in sleep mode..... 74
operation in stop mode ..... 75
operation in subclock and standby mode and when counter is suspended ..... 228
operation in subclock mode ..... 69
operation in watch mode..... 77
operation mode 0 of UART/SIO, explanation of... 351
operation of A/D conversion function ..... 283
operation of clock supply function..... 193
operation of counter function ..... 222, 246
operation of external interrupt circuit..... 264
operation of interval timer function..... 220, 244
operation of interval timer function (watch prescaler) ..... 193
operation of parallel discharge control ..... 329
operation of port 0..... 94
operation of port 1..... 100
operation of port 2..... 106
operation of port 3..... 112
operation of port 4..... 116
operation of port 5..... 121
operation of port 6..... 127
operation of port 7..... 132
operation of port 8..... 139
operation of port 9..... 146
operation of port A ..... 151
operation of port B ..... 156
operation of square wave output initial setting function ..... 225
operation of stopping and restarting 8/16-bit timer/counter..... 227
operation of UART/SIO ..... 350
operation of watch prescaler..... 193
operation of watchdog timer..... 179
oscillation stabilization delay reset state ..... 56
oscillation stabilization wait time ..... 70, 84
oscillation stabilization wait time and timebase timer interrupt..... 166
oscillation stabilization wait time and watch interrupt ..... 192
oscillation stabilization wait time, selection of ..... 5
other instruction ..... 527
overview of F2MC-8L instruction ..... 511

**P**

parallel discharge control..... 298
parallel discharge control, operation of..... 329
pin associated with comparator ..... 303
pin associated with comparator, block diagram of .....304
pin description .....13
pin related to 8/16-bit timer/counter.....205
pin related to A/D converter.....270
pin related to A/D converter, block diagram of .....271
pin related to external interrupt circuit .....258
pin related to external interrupt circuit, block diagram of .....259
pin related to LCD controller driver.....463
pin related to LCD controller driver , block diagram of .....464
pin related to UART/SIO.....337
pin related to UART/SIO, block diagram of .....338
pin state during reset.....57
pin states after reading mode data.....57
pinl status during various operation.....531
port 0, block diagram of.....92
port 0, configuration of.....91
port 0, operation of .....94
port 0, pin of.....91
port 1, block diagram of.....97
port 1, configuration of.....96
port 1, operation of .....100
port 1, pin of.....96
port 1, register of .....97
port 2, block diagram of.....103
port 2, configuration of.....102
port 2, operation of .....106
port 2, pin of.....102
port 2, register of .....103
port 3, block diagram of.....109
port 3, configuration of.....108
port 3, operation of .....112
port 3, pin of.....108
port 3, register of .....110
port 4, block diagram of.....114
port 4, configuration of.....113
port 4, operation of .....116
port 4, pin of.....113
port 4, register of .....114
port 5, block diagram of.....118
port 5, configuration of.....117
port 5, operation of .....121
port 5, pin of.....117
port 5, register of .....118
port 6, block diagram of.....124
port 6, configuration of.....123
port 6, operation of .....127

port 6, pin of ..... 123
port 6, register of ..... 124
port 7, block diagram of ..... 129
port 7, configuration of ..... 128
port 7, operation of ..... 132
port 7, pin of ..... 128
port 8, block diagram of ..... 135
port 8, configuration of ..... 134
port 8, operation of ..... 139
port 8, pins of ..... 134
port 8, register of ..... 136
port 9, block diagram of ..... 142
port 9, configuration of ..... 141
port 9, operation of ..... 146
port 9, pin of ..... 141
port A, block diagram of ..... 149
port A, configuration of ..... 148
port A, operation of ..... 151
port A, register of ..... 149
port B, block diagram of ..... 153
port B, configuration of ..... 152
port B, operation of ..... 156
port B, pin of ..... 152
port B, register of ..... 153
precaution in setting I2C interface register ..... 395
precaution in setting multi-address I2C interface register ..... 435
precaution in setting shift clock frequency .... 395, 435
precaution on priority at simultaneous writing ..... 395, 435
precaution on setting with software ..... 395, 435

**R**

read-modify-write operation ..... 521
receiving operation in CLK asynchronous mode .. 353
reception error in CLK asynchronous mode ..... 354
reception interrupt ..... 349
register and vector table address related to interrupt of UART/SIO ..... 349
register and vector table associated with comparator interrupts ..... 328
register and vector table related to interrupt of external interrupt circuit ..... 263
register and vector table related to timebase timer interrupt ..... 166
register and vector table related to watch prescaler interrupt ..... 192
register associated with comparator ..... 306
register bank pointer (RP), structure of ..... 37
register of port 0, function of ..... 93
register of port 1, function of ..... 98
register of port 2, function of ..... 104
register of port 3, function of ..... 111
register of port 4, function of ..... 115
register of port 5, function of ..... 119
register of port 6, function of ..... 125
register of port 7, function of ..... 130
register of port 8, function of ..... 137
register of port 9, function of ..... 144
register of port A, function of ..... 150
register of port B, function of ..... 155
register PDR0 and DDR0 of port 0 ..... 92
register PDR7 and DDR7 of port 7 ..... 129
register PDR9 and DDR9 of port 9 ..... 143
register related to 8/16-bit timer/counter ..... 207
register related to A/D converter ..... 273
register related to external interrupt circuit ..... 260
register related to interrupt of 8/16-bit timer/counter and vector table ..... 219
register related to LCD controller driver ..... 465
register related to port 1 ..... 99
register related to port 3 ..... 111
register related to port 4 ..... 115
register related to port 6 ..... 126
register related to port 7 ..... 131
register related to port 8 ..... 138
register related to port 9 ..... 145
register related to port A ..... 150
register related to port B ..... 155
register related to UART/SIO ..... 339
reset flag register (RSFR) ..... 52
reset on RAM content, effect of ..... 56
reset operation, overview of ..... 55
reset source ..... 50
ROM programmer adapter and recommended ROM programmer ..... 530

**S**

sample application ..... 331
sequential discharge control ..... 298
sequential discharge control, operation of ..... 330
serial input data register (SIDR) ..... 347
serial mode control register 1 (SMC1) ..... 340
serial mode control register 2 (SMC2) ..... 342
serial output data register (SODR) ..... 348
serial status and data register (SSD) ..... 345
setting with software, caution on ..... 395, 435
shift clock frequency, note on setting ..... 435
shift clock frequency, precaution in setting ..... 395

simultaneous writing, precaution on priority at ..... 395, 435
single chip mode ..... 85
slave timeout ..... 398, 438
sleep mode, operation in ..... 74
special instruction ..... 518
square wave output initial setting function, operation of ..... 225
stack area for interrupt processing ..... 49
stack operation at interrupt return ..... 48
stack operation at start of interrupt processing ..... 48
standby control register (STBC) ..... 78
standby mode ..... 72
standby mode and interrupt, transition to ..... 83
standby mode by interrupt, release of ..... 83
standby mode, operating state in ..... 73
standby mode, precaution in setting ..... 84
start condition ..... 393, 433
state transition diagram 1 (dual clock) ..... 80
stop condition ..... 394, 434
stop mode, operation in ..... 75
stopping and restarting 8/16-bit timer/counter, operation of ..... 227
subclock and standby mode and when counter is suspended, operation in ..... 228
subclock mode, operation in ..... 69
subclock, oscillation stabilization wait time of ..... 71
symbols used with instruction ..... 512
system clock control register (SYCC), structure of ..... 64

**T**

timebase timer control register (TBTC) ..... 164
timebase timer interrupt, oscillation stabilization wait time and ..... 166
timebase timer, block diagram of ..... 162
timebase timer, note on using ..... 169
timebase timer, program example of ..... 171
time-based timer, operation of ..... 167
timeout clock supply block ..... 399, 439
timer 1 control register (T1CR) ..... 208
timer 1 data register (T1DR) ..... 214
timer 2 control register (T2CR) ..... 211
timer 2 data register (T2DR) ..... 216
timer control register (TMCR) ..... 240
timer count register (TCR) ..... 242
transfer data format ..... 352
transfer instruction ..... 522
transmission interrupt ..... 349
transmitting operation in CLK asynchronous mode ..... 355

**U**

UART/SIO operation mode, explanation of ..... 356
UART/SIO, block diagram of ..... 335
UART/SIO, function of ..... 334
UART/SIO, operation of ..... 350
upper address setting register (WRARH) ..... 496
using external dividing resistor ..... 462
using internal dividing resistor ..... 460

**V**

various product and precaution for selecting productl, difference of ..... 7
vector table area ..... 28

**W**

watch mode, operation in ..... 77
watch prescaler control register (WPCR) ..... 190
watch prescaler, block diagram of ..... 188
watch prescaler, note on using ..... 195
watch prescaler, operation of ..... 193
watch prescaler, program example of ..... 196
watchdog timer control register (WDTC) ..... 177
watchdog timer function ..... 174
watchdog timer, block diagram of ..... 175
watchdog timer, note on using ..... 181
watchdog timer, operation of ..... 179
watchdog timer, program example of ..... 182
wild register address, list of ..... 502
wild register application address ..... 490
wild register function ..... 490
wild register function operation, sequence of ..... 502
wild register function, block diagram of ..... 491
wild register function, register related to ..... 493

CM25-10140-1E

**FUJITSU SEMICONDUCTOR** • CONTROLLER MANUAL

# $F^2$MC-8L
# 8-BIT MICROCONTROLLER
# MB89570 Series
# HARDWARE MANUAL

February 2001 the first edition

Published **FUJITSU LIMITED** Electronic Devices

Edited Technical Communication Dept.

FUJITSU

*CM25-10140-1E*

Copyright © Each Manufacturing Company.

All Datasheets cannot be modified without permission.

This datasheet has been download from :

www.AllDataSheet.com

100% Free DataSheet Search Site.

Free Download.

No Register.

Fast Search System.

www.AllDataSheet.com