

8-BIT MICROCONTROLLER  
**New 8FX**  
**MB95850K/860K/870K Series**  
**HARDWARE MANUAL**



**8-BIT MICROCONTROLLER**  
**New 8FX**  
**MB95850K/860K/870K Series**  
**HARDWARE MANUAL**

For the information for microcontroller supports, see the following website.

<http://edevice.fujitsu.com/micom/en-support/>

**FUJITSU SEMICONDUCTOR LIMITED**



# PREFACE

## ■ The Purpose and Intended Readership of This Manual

Thank you very much for your continued special support for Fujitsu Semiconductor products.

The MB95850K/860K/870K Series is a line of products developed as general-purpose products in the New 8FX family of proprietary 8-bit single-chip microcontrollers applicable as application-specific integrated circuits (ASICs). The MB95850K/860K/870K Series can be used for a wide range of applications from consumer products including portable devices to industrial equipment.

Intended for engineers who actually develop products using the MB95850K/860K/870K Series of microcontrollers, this manual describes its functions, features, and operations. You should read through the manual.

For details on individual instructions, refer to "F<sup>2</sup>MC-8FX Programming Manual".

This manual is written to explain the respective configurations and operations of peripheral functions, but not to provide specifications of a device.

For detailed specifications of a device, refer to its data sheet.

Note: F<sup>2</sup>MC is the abbreviation of FUJITSU Flexible Microcontroller.

## ■ Trademark

The company names and brand names in this document are the trademarks or registered trademarks of their respective owners.

## ■ Sample Programs

Fujitsu Semiconductor provides sample programs free of charge to operate the peripheral functions of the New 8FX family of microcontrollers. Feel free to use such sample programs to check the operational specifications and usages of Fujitsu microcontrollers.

Note that sample programs are subject to change without notice. As these pieces of software are offered to show standard operations and usages, evaluate them sufficiently before use with your system. Fujitsu Semiconductor assumes no liability for any damages whatsoever arising out of the use of sample programs.

# How to Use This Manual

## ■ Finding a function

The following methods can be used to search for details of a peripheral function in this manual:

- Searching from CONTENTS  
CONTENTS lists the contents in this manual in the order of description.
- Searching from registers

The address at which a register is located is not mentioned in this manual. To check the address of a register, refer to "■ I/O MAP" in the device data sheet.

## ■ Chapters

This manual explains one peripheral function in one chapter.

## ■ Terminology

This manual uses the following terminology.

| Term | Explanation                             |
|------|---|
| Word | Indicates an access in unit of 16 bits. |
| Byte | Indicates an access in unit of 8 bits.  |

## ■ Notations

The notations in "■ Register Configuration" in this manual are explained below:

- bit: bit number
- Field: bit field name
- Attribute: Attributes for read access and write access of each bit
  - R: Read-only
  - W: Write-only
  - R/W: Readable/Writable
  - —: Undefined
- Initial value: Initial value of a bit after a reset
  - 0: The initial value is "0".
  - 1: The initial value is "1".
  - X: The initial value is undefined.

Multiple bits are indicated in this manual in the following way.

- Example 1: bit7:0 represents bit7 to bit0.
- Example 2: SCM[2:0] represents SCM2 to SCM0.

The values such as those indicating addresses are written in this manual in the following ways:

- Hexadecimal number: The prefix "0x" is attached to the beginning of a value (e.g.: 0xFFFF).
- Binary number: The prefix "0b" is attached to the beginning of a value (e.g.: 0b1111).
- Decimal number: Only the number is used (e.g.: 1234).

In this manual, "n" in a pin name and a register abbreviation represents the channel number.

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





# CONTENTS

|                  |   |           |
|------------------|---|-----------|
| <b>CHAPTER 1</b> | <b>MEMORY ACCESS MODE</b>                                   | <b>1</b>  |
| 1.1              | Memory Access Mode  | 2         |
| <b>CHAPTER 2</b> | <b>CPU</b>  | <b>3</b>  |
| 2.1              | Dedicated Registers   | 4         |
| 2.1.1            | Register Bank Pointer (RP)                                  | 6         |
| 2.1.2            | Direct Bank Pointer (DP)                                    | 7         |
| 2.1.3            | Condition Code Register (CCR)                               | 9         |
| 2.2              | General-purpose Register                                    | 11        |
| 2.3              | Placement of 16-bit Data in Memory                          | 13        |
| <b>CHAPTER 3</b> | <b>CLOCK CONTROLLER</b>                                     | <b>15</b> |
| 3.1              | Overview  | 16        |
| 3.2              | Oscillation Stabilization Wait Time                         | 24        |
| 3.3              | Registers   | 27        |
| 3.3.1            | System Clock Control Register (SYCC)                        | 28        |
| 3.3.2            | PLL Control Register (PLLC)                                 | 29        |
| 3.3.3            | Oscillation Stabilization Wait Time Setting Register (WATR) | 30        |
| 3.3.4            | Standby Control Register (STBC)                             | 32        |
| 3.3.5            | System Clock Control Register 2 (SYCC2)                     | 34        |
| 3.3.6            | Standby Control Register 2 (STBC2)                          | 36        |
| 3.4              | Clock Modes   | 37        |
| 3.5              | Operations in Low Power Consumption Mode (Standby Mode)     | 41        |
| 3.5.1            | Notes on Using Standby Mode                                 | 42        |
| 3.5.2            | Sleep Mode  | 48        |
| 3.5.3            | Stop Mode   | 49        |
| 3.5.4            | Time-base Timer Mode  | 51        |
| 3.5.5            | Watch Mode  | 53        |
| 3.6              | Clock Oscillator Circuit                                    | 54        |
| 3.7              | Overview of Prescaler                                       | 55        |
| 3.8              | Configuration of Prescaler                                  | 56        |
| 3.9              | Operation of Prescaler                                      | 57        |
| 3.10             | Notes on Using Prescaler                                    | 59        |
| <b>CHAPTER 4</b> | <b>RESET</b>  | <b>61</b> |
| 4.1              | Reset Operation   | 62        |
| 4.2              | Register  | 66        |
| 4.2.1            | Reset Source Register (RSRR)                                | 67        |
| 4.3              | Notes on Using Reset  | 70        |
| <b>CHAPTER 5</b> | <b>INTERRUPTS</b>   | <b>71</b> |
| 5.1              | Interrupts  | 72        |
| 5.1.1            | Interrupt Level Setting Registers (ILR0 to ILR5)            | 73        |
| 5.1.2            | Interrupt Processing  | 75        |

|                   |   |            |
|-------------------|---|------------|
| 5.1.3             | Nested Interrupts .....                           | 77         |
| 5.1.4             | Interrupt Processing Time .....                   | 78         |
| 5.1.5             | Stack Operation During Interrupt Processing ..... | 79         |
| 5.1.6             | Interrupt Processing Stack Area .....             | 80         |
| <b>CHAPTER 6</b>  | <b>I/O PORT .....</b>                             | <b>81</b>  |
| 6.1               | Overview .....                                    | 82         |
| 6.2               | Configuration and Operations of I/O Port .....    | 83         |
| <b>CHAPTER 7</b>  | <b>TIME-BASE TIMER .....</b>                      | <b>87</b>  |
| 7.1               | Overview .....                                    | 88         |
| 7.2               | Configuration .....                               | 89         |
| 7.3               | Interrupt .....                                   | 91         |
| 7.4               | Operations and Setting Procedure Example .....    | 92         |
| 7.5               | Register .....                                    | 95         |
| 7.5.1             | Time-base Timer Control Register (TBTC) .....     | 96         |
| 7.6               | Notes on Using Time-base Timer .....              | 98         |
| <b>CHAPTER 8</b>  | <b>HARDWARE/SOFTWARE WATCHDOG TIMER .....</b>     | <b>99</b>  |
| 8.1               | Overview .....                                    | 100        |
| 8.2               | Configuration .....                               | 101        |
| 8.3               | Operations and Setting Procedure Example .....    | 103        |
| 8.4               | Register .....                                    | 106        |
| 8.4.1             | Watchdog Timer Control Register (WDTC) .....      | 107        |
| 8.5               | Notes on Using Watchdog Timer .....               | 109        |
| <b>CHAPTER 9</b>  | <b>WATCH PRESCALER .....</b>                      | <b>111</b> |
| 9.1               | Overview .....                                    | 112        |
| 9.2               | Configuration .....                               | 113        |
| 9.3               | Interrupt .....                                   | 115        |
| 9.4               | Operations and Setting Procedure Example .....    | 116        |
| 9.5               | Register .....                                    | 119        |
| 9.5.1             | Watch Prescaler Control Register (WPCR) .....     | 120        |
| 9.6               | Notes on Using Watch Prescaler .....              | 122        |
| <b>CHAPTER 10</b> | <b>WATCH COUNTER .....</b>                        | <b>123</b> |
| 10.1              | Overview .....                                    | 124        |
| 10.2              | Configuration .....                               | 125        |
| 10.3              | Interrupt .....                                   | 127        |
| 10.4              | Operations and Setting Procedure Example .....    | 128        |
| 10.5              | Registers .....                                   | 130        |
| 10.5.1            | Watch Counter Data Register (WCDR) .....          | 131        |
| 10.5.2            | Watch Counter Control Register (WCSR) .....       | 132        |
| 10.6              | Notes on Using Watch Counter .....                | 133        |
| <b>CHAPTER 11</b> | <b>WILD REGISTER FUNCTION .....</b>               | <b>135</b> |
| 11.1              | Overview .....                                    | 136        |
| 11.2              | Configuration .....                               | 137        |
| 11.3              | Operations .....                                  | 139        |

|   |  |            |
|---|--|------------|
| 11.4  | Registers .....  | 140        |
| 11.4.1  | Wild Register Data Setting Registers (WRDR0 to WRDR2) .....              | 141        |
| 11.4.2  | Wild Register Address Setting Registers (WRAR0 to WRAR2) .....           | 142        |
| 11.4.3  | Wild Register Address Compare Enable Register (WREN) .....               | 143        |
| 11.4.4  | Wild Register Data Test Setting Register (WROR) .....                    | 144        |
| 11.5  | Typical Hardware Connection Example .....                                | 145        |
| <b>CHAPTER 12 8/16-BIT COMPOSITE TIMER .....</b>        |  | <b>147</b> |
| 12.1  | Overview .....   | 148        |
| 12.2  | Configuration .....  | 150        |
| 12.3  | Channel .....  | 153        |
| 12.4  | Pins .....   | 154        |
| 12.5  | Interrupts .....   | 155        |
| 12.6  | Operation of Interval Timer Function (One-shot Mode) .....               | 156        |
| 12.7  | Operation of Interval Timer Function (Continuous Mode) .....             | 158        |
| 12.8  | Operation of Interval Timer Function (Free-run Mode) .....               | 160        |
| 12.9  | Operation of PWM Timer Function (Fixed-cycle mode) .....                 | 162        |
| 12.10   | Operation of PWM Timer Function (Variable-cycle Mode) .....              | 164        |
| 12.11   | Operation of PWC Timer Function .....                                    | 166        |
| 12.12   | Operation of Input Capture Function .....                                | 168        |
| 12.13   | Operation of Noise Filter .....  | 170        |
| 12.14   | Registers .....  | 171        |
| 12.14.1   | 8/16-bit Composite Timer Status Control Register 0 (Tn0CR0/Tn1CR0) ..... | 172        |
| 12.14.2   | 8/16-bit Composite Timer Status Control Register 1 (Tn0CR1/Tn1CR1) ..... | 175        |
| 12.14.3   | 8/16-bit Composite Timer Timer Mode Control Register (TMCRn) .....       | 179        |
| 12.14.4   | 8/16-bit Composite Timer Data Register (Tn0DR/Tn1DR) .....               | 181        |
| 12.15   | Notes on Using 8/16-bit Composite Timer .....                            | 184        |
| <b>CHAPTER 13 EXTERNAL INTERRUPT CIRCUIT .....</b>      |  | <b>187</b> |
| 13.1  | Overview .....   | 188        |
| 13.2  | Configuration .....  | 189        |
| 13.3  | Channels .....   | 190        |
| 13.4  | Pin .....  | 191        |
| 13.5  | Interrupt .....  | 192        |
| 13.6  | Operations and Setting Procedure Example .....                           | 193        |
| 13.7  | Register .....   | 195        |
| 13.7.1  | External Interrupt Control Register (EIC) .....                          | 196        |
| 13.8  | Notes on Using External Interrupt Circuit .....                          | 198        |
| <b>CHAPTER 14 INTERRUPT PIN SELECTION CIRCUIT .....</b> |  | <b>199</b> |
| 14.1  | Overview .....   | 200        |
| 14.2  | Configuration .....  | 201        |
| 14.3  | Pins .....   | 204        |
| 14.4  | Operation .....  | 205        |
| 14.5  | Register .....   | 206        |
| 14.5.1  | Interrupt Pin Selection Circuit Control Register (WICR) .....            | 207        |
| 14.6  | Notes on Using Interrupt Pin Selection Circuit .....                     | 213        |

|                   |  |            |
|-------------------|--|------------|
| <b>CHAPTER 15</b> | <b>8/10-BIT A/D CONVERTER .....</b>                                      | <b>215</b> |
| 15.1              | Overview .....   | 216        |
| 15.2              | Configuration .....  | 217        |
| 15.3              | Pin .....  | 219        |
| 15.4              | Interrupt .....  | 220        |
| 15.5              | Operations and Setting Procedure Example .....                           | 221        |
| 15.6              | Registers .....  | 224        |
| 15.6.1            | 8/10-bit A/D Converter Control Register 1 (ADC1) .....                   | 225        |
| 15.6.2            | 8/10-bit A/D Converter Control Register 2 (ADC2) .....                   | 227        |
| 15.6.3            | 8/10-bit A/D Converter Data Register (Upper/Lower) (ADDH/ADDL) .....     | 229        |
| 15.7              | Notes on Using 8/10-bit A/D Converter .....                              | 230        |
| <br>              |  |            |
| <b>CHAPTER 16</b> | <b>LOW-VOLTAGE DETECTION RESET CIRCUIT .....</b>                         | <b>233</b> |
| 16.1              | Overview .....   | 234        |
| 16.2              | Configuration .....  | 235        |
| 16.3              | Pins .....   | 236        |
| 16.4              | Operation .....  | 237        |
| 16.5              | Registers .....  | 238        |
| 16.5.1            | LVD Reset Voltage Selection ID Register (LVDR) .....                     | 239        |
| 16.5.2            | LVD Reset Circuit Password Register (LVDPW) .....                        | 240        |
| 16.5.3            | LVD Reset Circuit Control Register (LVGCC) .....                         | 241        |
| <br>              |  |            |
| <b>CHAPTER 17</b> | <b>CLOCK SUPERVISOR COUNTER .....</b>                                    | <b>243</b> |
| 17.1              | Overview .....   | 244        |
| 17.2              | Configuration .....  | 245        |
| 17.3              | Operations .....   | 247        |
| 17.4              | Registers .....  | 252        |
| 17.4.1            | Clock Monitoring Data Register (CMDR) .....                              | 253        |
| 17.4.2            | Clock Monitoring Control Register (CMCR) .....                           | 254        |
| 17.5              | Notes on Using Clock Supervisor Counter .....                            | 256        |
| <br>              |  |            |
| <b>CHAPTER 18</b> | <b>8/16-BIT PPG .....</b>  | <b>259</b> |
| 18.1              | Overview .....   | 260        |
| 18.2              | Configuration .....  | 261        |
| 18.3              | Channel .....  | 263        |
| 18.4              | Pins .....   | 264        |
| 18.5              | Interrupt .....  | 265        |
| 18.6              | Operations and Setting Procedure Example .....                           | 266        |
| 18.6.1            | 8-bit PPG Independent Mode .....   | 267        |
| 18.6.2            | 8-bit Prescaler + 8-bit PPG Mode .....                                   | 269        |
| 18.6.3            | 16-bit PPG Mode .....  | 271        |
| 18.7              | Registers .....  | 274        |
| 18.7.1            | 8/16-bit PPG timer n1 Control Register (PCn1) .....                      | 275        |
| 18.7.2            | 8/16-bit PPG timer n0 Control Register (PCn0) .....                      | 277        |
| 18.7.3            | 8/16-bit PPG timer n1/n0 Cycle Setup Buffer Register (PPSn1/PPSn0) ..... | 279        |
| 18.7.4            | 8/16-bit PPG timer n1/n0 Duty Setup Buffer Register (PDSn1/PDSn0) .....  | 280        |
| 18.7.5            | 8/16-bit PPG Start Register (PPGS) .....                                 | 281        |
| 18.7.6            | 8/16-bit PPG Output Reverse Register (REVC) .....                        | 283        |
| 18.8              | Notes on Using 8/16-bit PPG .....  | 285        |

|                   |   |            |
|-------------------|---|------------|
| <b>CHAPTER 19</b> | <b>UART/SIO</b>   | <b>287</b> |
| 19.1              | Overview  | 288        |
| 19.2              | Configuration   | 289        |
| 19.3              | Channel   | 291        |
| 19.4              | Pins  | 292        |
| 19.5              | Interrupts  | 293        |
| 19.6              | Operations and Setting Procedure Example  | 294        |
| 19.6.1            | Operations in Operation Mode 0  | 295        |
| 19.6.2            | Operations in Operation Mode 1  | 302        |
| 19.7              | Registers   | 308        |
| 19.7.1            | UART/SIO Serial Mode Control Register 1 ch. n (SMC1n)                           | 309        |
| 19.7.2            | UART/SIO Serial Mode Control Register 2 ch. n (SMC2n)                           | 311        |
| 19.7.3            | UART/SIO Serial Status and Data Register ch. n (SSRn)                           | 313        |
| 19.7.4            | UART/SIO Serial Input Data Register ch. n (RDRn)                                | 315        |
| 19.7.5            | UART/SIO Serial Output Data Register ch. n (TDRn)                               | 316        |
| <br>              |   |            |
| <b>CHAPTER 20</b> | <b>UART/SIO DEDICATED BAUD RATE GENERATOR</b>                                   | <b>317</b> |
| 20.1              | Overview  | 318        |
| 20.2              | Channel   | 319        |
| 20.3              | Operations  | 320        |
| 20.4              | Registers   | 321        |
| 20.4.1            | UART/SIO Dedicated Baud Rate Generator Prescaler Select Register ch. n (PSSRn)  | 322        |
| 20.4.2            | UART/SIO Dedicated Baud Rate Generator Baud Rate Setting Register ch. n (BRSRn) | 323        |
| <br>              |   |            |
| <b>CHAPTER 21</b> | <b>I<sup>2</sup>C BUS INTERFACE</b>   | <b>325</b> |
| 21.1              | Overview  | 326        |
| 21.2              | Configuration   | 327        |
| 21.3              | Channel   | 330        |
| 21.4              | Pins  | 331        |
| 21.5              | Interrupts  | 332        |
| 21.6              | Operations and Setting Procedure Example  | 334        |
| 21.6.1            | I <sup>2</sup> C Bus Interface  | 335        |
| 21.6.2            | Function to Wake up the MCU from Standby Mode                                   | 343        |
| 21.7              | Registers   | 345        |
| 21.7.1            | I <sup>2</sup> C Bus Control Register 0 ch. n (IBCR0n)                          | 346        |
| 21.7.2            | I <sup>2</sup> C Bus Control Register 1 ch. n (IBCR1n)                          | 349        |
| 21.7.3            | I <sup>2</sup> C Bus Status Register ch. n (IBSRn)                              | 352        |
| 21.7.4            | I <sup>2</sup> C Data Register ch. n (IDDRn)                                    | 355        |
| 21.7.5            | I <sup>2</sup> C Address Register ch. n (IAARn)                                 | 356        |
| 21.7.6            | I <sup>2</sup> C Clock Control Register ch. n (ICCRn)                           | 357        |
| 21.8              | Notes on Using I <sup>2</sup> C Bus Interface                                   | 359        |
| <br>              |   |            |
| <b>CHAPTER 22</b> | <b>EXAMPLE OF SERIAL PROGRAMMING CONNECTION</b>                                 | <b>361</b> |
| 22.1              | Basic Configuration of Serial Programming Connection                            | 362        |
| 22.2              | Example of Serial Programming Connection  | 363        |

|  |            |
|--|------------|
| <b>CHAPTER 23 DUAL OPERATION FLASH MEMORY .....</b>                          | <b>365</b> |
| 23.1 Overview .....  | 366        |
| 23.2 Sector/Bank Configuration .....   | 368        |
| 23.3 Invoking Flash Memory Automatic Algorithm .....                         | 369        |
| 23.4 Checking Automatic Algorithm Execution Status .....                     | 371        |
| 23.4.1 Data Polling Flag (DQ7) .....   | 373        |
| 23.4.2 Toggle Bit Flag (DQ6) .....   | 375        |
| 23.4.3 Execution Timeout Flag (DQ5) .....                                    | 376        |
| 23.4.4 Sector Erase Timer Flag (DQ3) .....                                   | 377        |
| 23.4.5 Toggle Bit2 Flag (DQ2) .....  | 378        |
| 23.5 Programming/Erasing Flash Memory .....                                  | 379        |
| 23.5.1 Placing Flash Memory in Read/Reset State .....                        | 380        |
| 23.5.2 Programming Data to Flash Memory .....                                | 381        |
| 23.5.3 Erasing All Data from Flash Memory (Chip Erase) .....                 | 383        |
| 23.5.4 Erasing Specific Data from Flash Memory (Sector Erase) .....          | 384        |
| 23.5.5 Suspending Sector Erase from Flash Memory .....                       | 386        |
| 23.5.6 Resuming Sector Erase of Flash Memory .....                           | 387        |
| 23.5.7 Unlock Bypass Program .....   | 388        |
| 23.6 Operations .....  | 389        |
| 23.7 Flash Security .....  | 391        |
| 23.8 Registers .....   | 392        |
| 23.8.1 Flash Memory Status Register 2 (FSR2) .....                           | 393        |
| 23.8.2 Flash Memory Status Register (FSR) .....                              | 396        |
| 23.8.3 Flash Memory Sector Write Control Register 0 (SWRE0) .....            | 399        |
| 23.8.4 Flash Memory Status Register 3 (FSR3) .....                           | 401        |
| 23.8.5 Flash Memory Status Register 4 (FSR4) .....                           | 402        |
| 23.9 Notes on Using Dual Operation Flash Memory .....                        | 410        |
| <br>   |            |
| <b>CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE .....</b>                | <b>411</b> |
| 24.1 Overview .....  | 412        |
| 24.2 Configuration .....   | 413        |
| 24.3 Registers .....   | 414        |
| 24.3.1 Main CR Clock Trimming Register (Upper) (CRTH) .....                  | 415        |
| 24.3.2 Main CR Clock Trimming Register (Lower) (CRTL) .....                  | 416        |
| 24.3.3 Main CR Clock Temperature Dependent Adjustment Register (CRTDA) ..... | 417        |
| 24.3.4 Watchdog Timer Selection ID Register (Upper/Lower) (WDTH/WDTL) .....  | 418        |
| 24.4 Notes on Main CR Clock Trimming .....                                   | 419        |
| 24.5 Notes on Using NVR Interface .....                                      | 421        |
| <br>   |            |
| <b>CHAPTER 25 COMPARATOR .....</b>   | <b>423</b> |
| 25.1 Overview .....  | 424        |
| 25.2 Configuration .....   | 425        |
| 25.3 Dedicated BGR .....   | 427        |
| 25.4 Pins .....  | 428        |
| 25.5 Interrupt .....   | 429        |
| 25.6 Operations and Setting Procedure Example .....                          | 430        |
| 25.7 Register .....  | 432        |
| 25.7.1 Comparator Control Register ch. n (CMRn) .....                        | 433        |

|   |            |
|---|------------|
| <b>CHAPTER 26 TOUCH SENSOR .....</b>                                      | <b>437</b> |
| 26.1 Overview .....   | 438        |
| 26.2 Configuration .....  | 439        |
| 26.3 Pins .....   | 442        |
| 26.4 Interrupts .....   | 443        |
| 26.5 Operations and Setting Procedure Example .....                       | 445        |
| 26.6 Registers .....  | 455        |
| 26.6.1 TS Touch Channel Enable Register 0 (TER0) .....                    | 456        |
| 26.6.2 TS Touch Channel Enable Register 1 (TER1) .....                    | 458        |
| 26.6.3 TS Prescaler Control Register (PSC) .....                          | 460        |
| 26.6.4 TS Warm Reset Register (WRESET) .....                              | 463        |
| 26.6.5 TS Sensitivity Select Register 0 (RSEL0) .....                     | 464        |
| 26.6.6 TS Sensitivity Select Register 1 (RSEL1) .....                     | 465        |
| 26.6.7 TS Sensitivity Select Register 2 (RSEL2) .....                     | 466        |
| 26.6.8 TS Sensitivity Select Register 3 (RSEL3) .....                     | 467        |
| 26.6.9 TS Sensitivity Select Register 4 (RSEL4) .....                     | 468        |
| 26.6.10 TS Sensitivity Select Register 5 (RSEL5) .....                    | 469        |
| 26.6.11 TS Sensitivity Select Register 6 (RSEL6) .....                    | 470        |
| 26.6.12 TS Beep Duration Setting Register (BPDUR) .....                   | 471        |
| 26.6.13 TS Direct Output Control Register 1 (DIOR1) .....                 | 472        |
| 26.6.14 TS Direct Output Control Register 2 (DIOR2) .....                 | 474        |
| 26.6.15 TS Direct Output Control Register 3 (DIOR3) .....                 | 476        |
| 26.6.16 TS Feature Select Register (FTSEL) .....                          | 477        |
| 26.6.17 TS AIC Wait Time Setting Register (AICWAT) .....                  | 478        |
| 26.6.18 TS Calibration Interval Setting Register (CALITV) .....           | 479        |
| 26.6.19 TS Integration Time Setting Register (ITGTM) .....                | 480        |
| 26.6.20 TS Idle Time Setting Register (IDLETM) .....                      | 482        |
| 26.6.21 TS Control Register (CONTROL) .....                               | 483        |
| 26.6.22 TS Interrupt Mask Register (INTMR) .....                          | 484        |
| 26.6.23 TS Interrupt Clear Register (INTCR) .....                         | 485        |
| 26.6.24 TS Interrupt Pending Register (INTPR) .....                       | 487        |
| 26.6.25 TS Filter Period Setting Register (FLTP) .....                    | 489        |
| 26.6.26 TS Filter Threshold Setting Register (FLTTH) .....                | 490        |
| 26.6.27 TS Reference Delay Setting Register (REFDLY) .....                | 491        |
| 26.6.28 TS Alpha Value Setting Register ch. n (ALPHn) .....               | 492        |
| 26.6.29 TS Beta Value Setting Register (BETA) .....                       | 493        |
| 26.6.30 TS Touch Strength Threshold Setting Register ch. n (STRTHn) ..... | 494        |
| 26.6.31 TS Touch Strength Register ch. n (STRn) .....                     | 495        |
| 26.6.32 TS Calibrated Impedance Register ch. n (CALIPn) .....             | 496        |
| 26.6.33 TS Impedance Register ch. n (IMPEn) .....                         | 497        |
| 26.6.34 TS Touch Data Register (Lower) (TOUCHL) .....                     | 498        |
| 26.6.35 TS Touch Data Register (Upper) (TOUCHH) .....                     | 500        |
| <br>  |            |
| <b>CHAPTER 27 BEEP OUTPUT UNIT .....</b>                                  | <b>503</b> |
| 27.1 Overview .....   | 504        |
| 27.2 Configuration .....  | 505        |
| 27.3 Pin .....  | 506        |
| 27.4 Operations and Setting Procedure Example .....                       | 507        |
| 27.5 Register .....   | 509        |

|   |  |            |
|---|--|------------|
| 27.5.1  | Beep Output Frequency Register (BPFREQ)    | 510        |
| <b>CHAPTER 28 SYSTEM CONFIGURATION CONTROLLER</b> |  | <b>513</b> |
| 28.1  | Overview                                   | 514        |
| 28.2  | Register                                   | 515        |
| 28.2.1  | System Configuration Register (SYSC)       | 516        |
| 28.3  | Notes on Using Controller                  | 518        |
| <b>APPENDIX</b>                                   |  | <b>519</b> |
| APPENDIX A Instruction Overview                   |  | 520        |
| A.1   | Addressing                                 | 523        |
| A.2   | Special Instruction                        | 527        |
| A.3   | Bit Manipulation Instructions (SETB, CLRB) | 531        |
| A.4   | F <sup>2</sup> MC-8FX Instructions         | 532        |
| A.5   | Instruction Map                            | 535        |



# Major revisions in this edition

| Page | Revisions (For details, see their respective pages.)   |   |
|------|--|---|
| 67   | CHAPTER 4 RESET<br>4.2.1 Reset Source Register (RSRR)<br>■ Register Functions  | <p>Revised the following statement in details of the EXTTS bit. This bit reads "0" when read by a read access. A write access (writing "0" or "1") to this bit sets it to "0".<br/>           →<br/>           A read access or a write access (writing "0" or "1") to this bit sets it to "0".</p> <p>Revised the following statement in details of the WDTR bit. This bit reads "0" when read by a read access. A write access (writing "0" or "1") to this bit sets it to "0".<br/>           →<br/>           A read access or a write access (writing "0" or "1") to this bit sets it to "0".</p> <p>Revised the following statement in details of the PONR bit. This bit reads "0" when read by a read access. A write access (writing "0" or "1") to this bit sets it to "0".<br/>           →<br/>           A read access or a write access (writing "0" or "1") to this bit sets it to "0".</p> |
| 68   |  | <p>Revised the following statement in details of the HWR bit. This bit reads "0" when read by a read access. A write access (writing "0" or "1") to this bit sets it to "0".<br/>           →<br/>           A read access or a write access (writing "0" or "1") to this bit sets it to "0".</p> <p>Revised the following statement in details of the SWR bit. This bit reads "0" when read by a read access. A write access (writing "0" or "1") to this bit or a power-on reset sets it to "0".<br/>           →<br/>           A read access or a write access (writing "0" or "1") to this bit or a power-on reset sets it to "0".</p>   |
| 364  | CHAPTER 22 EXAMPLE OF SERIAL PROGRAMMING CONNECTION<br>22.2 Example of Serial Programming Connection<br>■ Example of Serial Programming Connection | Added statements related to the use of the pull-up resistor.  |
| 439  | CHAPTER 26 TOUCH SENSOR<br>26.2 Configuration<br>■ Block Diagram of TS<br>Figure 26.2-1  | Add remark *2.  |
| 440  | 26.2 Configuration<br>■ Block Diagram of TS<br>● AIC controller  | Revised the following statement.<br>The AIC controller controls AIC.<br>→<br>The AIC controller controls the automatic impedance function.  |

| Page       | Revisions (For details, see their respective pages.)                    |  |
|------------|---|--|
| 442        | 26.3 Pins<br>■ Pins of TS<br>● AREF pin                                 | New section  |
| 443        | 26.4 Interrupts<br>■ Details of GINT                                    | Revised Table 26.4-1.  |
| 444        |   | Revised the remark of "Note".<br>Minimum TINT interval = Setting of ITGTM × Setting of FLTP × SNCLK<br>→<br>Minimum TINT interval = Setting of ITGTM × Setting of FLTP × SNCLKs  |
| 449 to 453 | 26.5 Operations and Setting Procedure Example<br>■ Operations of TS     | New section  |
| 455        | 26.6 Registers  | Revised Table 26.6-1.  |
| 461        | 26.6.3 TS Prescaler Control Register (PSC)                              | Revised Figure 26.6-1.   |
| 462        | ■ Register Functions  | Added the following statement to details of the DIVSEL[3:0] bits.<br>For details of the SNCLKs frequency range, refer to the device data sheet.  |
| 464        | 26.6.5 TS Sensitivity Select Register 0 (RSEL0)                         | Revised the register name of the RSEL0 register.<br>TS Resistance Select Register 0<br>→<br>TS Sensitivity Select Register 0   |
|            |   | Revised the section summary.<br>The TS resistance select register 0 (RSEL0) selects a resistance value for the sensitivity of the touch reference channel.<br>→<br>The TS sensitivity select register 0 (RSEL0) selects a sensitivity value for the touch reference channel. |
|            | 26.6.5 TS Sensitivity Select Register 0 (RSEL0)<br>■ Register Functions | Revised details of the SREFB[2:0] bits.<br>Revised the table for the SREFB[2:0] bits.  |

| Page | Revisions (For details, see their respective pages.)                    |  |  |
|------|---|--|--|
| 465  | 26.6.6 TS Sensitivity Select Register 1 (RSEL1)                         | <p>Revised the register name of the RSEL1 register.<br/>TS Resistance Select Register 1<br/>→<br/>TS Sensitivity Select Register 1</p> <p>Revised the section summary.<br/>The TS resistance select register 1 (RSEL1) selects the individual resistance value for the sensitivity of touch ch. 0 and touch ch. 1.<br/>→<br/>The TS sensitivity select register 1 (RSEL1) selects the respective sensitivity values for touch ch. 0 and touch ch. 1.</p> |  |
|      | 26.6.6 TS Sensitivity Select Register 1 (RSEL1)<br>■ Register Functions | <p>Revised details of the S01B[2:0] bits.</p> <p>Revised the table for the S01B[2:0] bits.</p> <p>Revised details of the S00B[2:0] bits.</p> <p>Revised the table for the S00B[2:0] bits.</p>  |  |
|      | 466   | 26.6.7 TS Sensitivity Select Register 2 (RSEL2)  | <p>Revised the register name of the RSEL2 register.<br/>TS Resistance Select Register 2<br/>→<br/>TS Sensitivity Select Register 2</p> <p>Revised the section summary.<br/>The TS resistance select register 2 (RSEL2) selects the individual resistance value for the sensitivity of touch ch. 2 and touch ch. 3.<br/>→<br/>The TS sensitivity select register 2 (RSEL2) selects the respective sensitivity values for touch ch. 2 and touch ch. 3.</p> |
|      |   | 26.6.7 TS Sensitivity Select Register 2 (RSEL2)<br>■ Register Functions  | <p>Revised details of the S03B[2:0] bits.</p> <p>Revised the table for the S03B[2:0] bits.</p> <p>Revised details of the S02B[2:0] bits.</p> <p>Revised the table for the S02B[2:0] bits.</p>  |
| 467  |   | 26.6.8 TS Sensitivity Select Register 3 (RSEL3)  | <p>Revised the register name of the RSEL3 register.<br/>TS Resistance Select Register 3<br/>→<br/>TS Sensitivity Select Register 3</p> <p>Revised the section summary.<br/>The TS resistance select register 3 (RSEL3) selects the individual resistance value for the sensitivity of touch ch. 4 and touch ch. 5.<br/>→<br/>The TS sensitivity select register 3 (RSEL3) selects the respective sensitivity values for touch ch. 4 and touch ch. 5.</p> |
|      |   | 26.6.8 TS Sensitivity Select Register 3 (RSEL3)<br>■ Register Functions  | <p>Revised details of the S05B[2:0] bits.</p> <p>Revised the table for the S05B[2:0] bits.</p> <p>Revised details of the S04B[2:0] bits.</p> <p>Revised the table for the S04B[2:0] bits.</p>  |

| Page   | Revisions (For details, see their respective pages.)                    |  |  |
|--|---|--|--|
| 468  | 26.6.9 TS Sensitivity Select Register 4 (RSEL4)                         | <p>Revised the register name of the RSEL4 register.<br/>TS Resistance Select Register 4<br/>→<br/>TS Sensitivity Select Register 4</p> <p>Revised the section summary.<br/>The TS resistance select register 4 (RSEL4) selects the individual resistance value for the sensitivity of touch ch. 6 and touch ch. 7.<br/>→<br/>The TS sensitivity select register 4 (RSEL4) selects the respective sensitivity values for touch ch. 6 and touch ch. 7.</p> |  |
|  | 26.6.9 TS Sensitivity Select Register 4 (RSEL4)<br>■ Register Functions | <p>Revised details of the S07B[2:0] bits.</p> <p>Revised the table for the S07B[2:0] bits.</p> <p>Revised details of the S06B[2:0] bits.</p> <p>Revised the table for the S06B[2:0] bits.</p>  |  |
|  | 469   | 26.6.10 TS Sensitivity Select Register 5 (RSEL5)   | <p>Revised the register name of the RSEL5 register.<br/>TS Resistance Select Register 5<br/>→<br/>TS Sensitivity Select Register 5</p> <p>Revised the section summary.<br/>The TS resistance select register 5 (RSEL5) selects the individual resistance value for the sensitivity of touch ch. 8 and touch ch. 9.<br/>→<br/>The TS sensitivity select register 5 (RSEL5) selects the respective sensitivity values for touch ch. 8 and touch ch. 9.</p> |
|  |   | 26.6.10 TS Sensitivity Select Register 5 (RSEL5)<br>■ Register Functions   | <p>Revised details of the S09B[2:0] bits.</p> <p>Revised the table for the S09B[2:0] bits.</p> <p>Revised details of the S08B[2:0] bits.</p> <p>Revised the table for the S08B[2:0] bits.</p>  |
|  |   | 470  | 26.6.11 TS Sensitivity Select Register 6 (RSEL6)   |
| 26.6.11 TS Sensitivity Select Register 6 (RSEL6)<br>■ Register Functions |   |  | <p>Revised details of the S11B[2:0] bits.</p> <p>Revised the table for the S11B[2:0] bits.</p> <p>Revised details of the S10B[2:0] bits.</p> <p>Revised the table for the S10B[2:0] bits.</p>  |

| Page     | Revisions (For details, see their respective pages.)                              |   |
|----------|---|---|
| 478      | 26.6.17 TS AIC Wait Time Setting Register (AICWAT)<br>■ Register Functions        | Revised the following statement.<br>When there is no sensor input and the AIC wait time elapses, the AIC starts.<br>→<br>When no touch pad has been touched and the AIC wait time elapses, AIC starts.                                    |
| 479      | 26.6.18 TS Calibration Interval Setting Register (CALITV)<br>■ Register Functions | Revised details of the CALI[7:0] bits.  |
| 480      | 26.6.19 TS Integration Time Setting Register (ITGTM)<br>■ Register Functions      | Revised the following statement in details of the INTET[7:0] bits.<br>The integration time is the period of accumulating touch output.<br>→<br>The integration time is the period of accumulating touch output as shown in Figure 26.6-2. |
| 481      |   | Added Figure 26.6-2.  |
| 487, 488 | 26.6.24 TS Interrupt Pending Register (INTPR)                                     | Revised the position of this section from 26.6.35 to 26.6.24.   |
| 487      | 26.6.24 TS Interrupt Pending Register (INTPR)<br>■ Register Functions             | Added the following statement to details of the NEW bit.<br>This bit is cleared by writing "1" to the NEW bit in the INTCR register.  |
|          |   | Added the following statement to details of the EOC bit.<br>This bit is cleared by writing "1" to the EOC bit in the INTCR register.  |
|          |   | Added the following statement to details of the ITA bit.<br>This bit is cleared by writing "1" to the ITA bit in the INTCR register.  |
|          |   | Added the following statement to details of the ATI bit.<br>This bit is cleared by writing "1" to the ATI bit in the INTCR register.  |
| 488      |   | Added the following statement to details of the TINT bit.<br>This bit is cleared by writing "1" to the TINT bit in the INTCR register.  |
| 489      | 26.6.25 TS Filter Period Setting Register (FLTP)<br>■ Register Functions          | Added the following statement to details of the FILP[7:0] bits.<br>(The algorithm for this operation is similar to APIS.)   |
| 491      | 26.6.27 TS Reference Delay Setting Register (REFDLY)<br>■ Register Functions      | Corrected the following bit name in details of the REFD[6:0] bits.<br>SREF[2:0]<br>→<br>SREFB[2:0]  |
| 492      | 26.6.3 TS Prescaler Control Register (PSC)<br>■ Register Functions                | Corrected the resolution in details of the ALP[6:0] bits.<br>0.06 pF → 0.063 pF   |
|          |   | Corrected the capacitance in details of the ALP[6:0] bits.<br>0.3 pF (5 × 0.06 pF) → 0.315 pF (5 × 0.063 pF)  |

| Page | Revisions (For details, see their respective pages.)   |  |
|------|--|--|
| 496  | 26.6.32 TS Calibrated Impedance Register ch. n (CALIPn)<br>■ Register Functions  | Revised the following statement in details of the CIR[6:0] bits. When the touch pad is touched and the decrease in impedance resulting from the touch is larger than the alpha value, the touch can be recognized by the TS.<br>→<br>When the touch pad is touched and the decrease* in impedance (value of the IMPEn register) resulting from the touch is larger than the alpha value, the touch can be recognized by the TS.<br><hr/> Added a remark to details of the CIR[6:0] bits. |
| 497  | 26.6.33 TS Impedance Register ch. n (IMPEn)<br>■ Register Functions  | Revised the following statement in details of the IR[6:0] bits. The AIC controller uses the impedance change for calibrating touch ch. n.<br>→<br>The AIC controller uses the value of the IMPEn register for calibrating touch ch. n.   |
| 500  | 26.6.35 TS Touch Data Register (Upper) (TOUCHH)<br>■ Register Configuration<br><hr/> 26.6.35 TS Touch Data Register (Upper) (TOUCHH)<br>■ Register Functions | Corrected the configuration of the TOUCHH register.<br><hr/> Corrected the number of undefined bits. [bit7:3] → [bit7:4]<br><hr/> Corrected the bit number of the TS11 bit. [bit2] → [bit3]<br><hr/> Corrected the bit number of the TS10 bit. [bit1] → [bit2]<br><hr/> Corrected the bit number of the TS09 bit. [bit0] → [bit1]<br><hr/> Added details of the TS08 bit.  |

# **CHAPTER 1**

---

# ***MEMORY ACCESS MODE***

**This chapter describes the memory access mode.**

## 1.1 Memory Access Mode

## 1.1 Memory Access Mode

---

**The MB95850K/860K/870K Series supports only one memory access mode: single-chip mode.**

---

### ■ Single-chip Mode

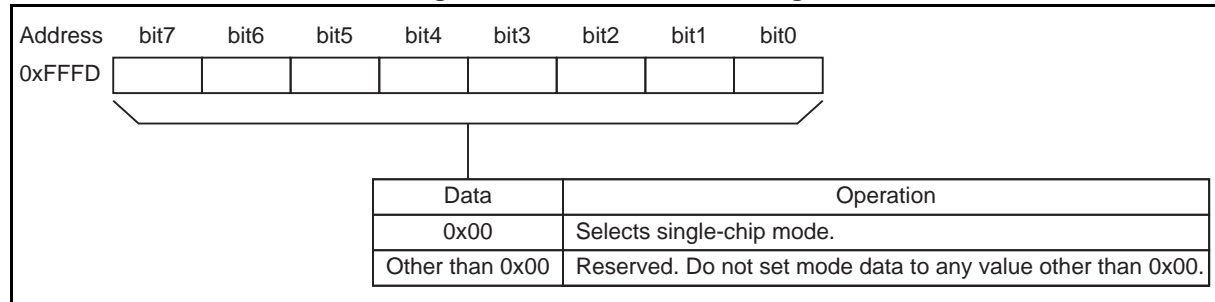
In single-chip mode, only the internal RAM and the Flash memory are used, and no external bus access is executed.

#### ● Mode data

Mode data is the data used to determine the memory access mode of the CPU.

The mode data address is fixed at "0xFFFFD". Always set the mode data of the Flash memory to "0x00" to select the single-chip mode.

**Figure 1.1-1 Mode Data Settings**



After a reset is released, the CPU fetches mode data first.

The CPU then fetches the reset vector after the mode data. It starts executing instructions from the address set in the reset vector.



# **CHAPTER 2**

---

## **CPU**

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

- 2.1 Dedicated Registers
- 2.2 General-purpose Register
- 2.3 Placement of 16-bit Data in Memory

## 2.1 Dedicated Registers

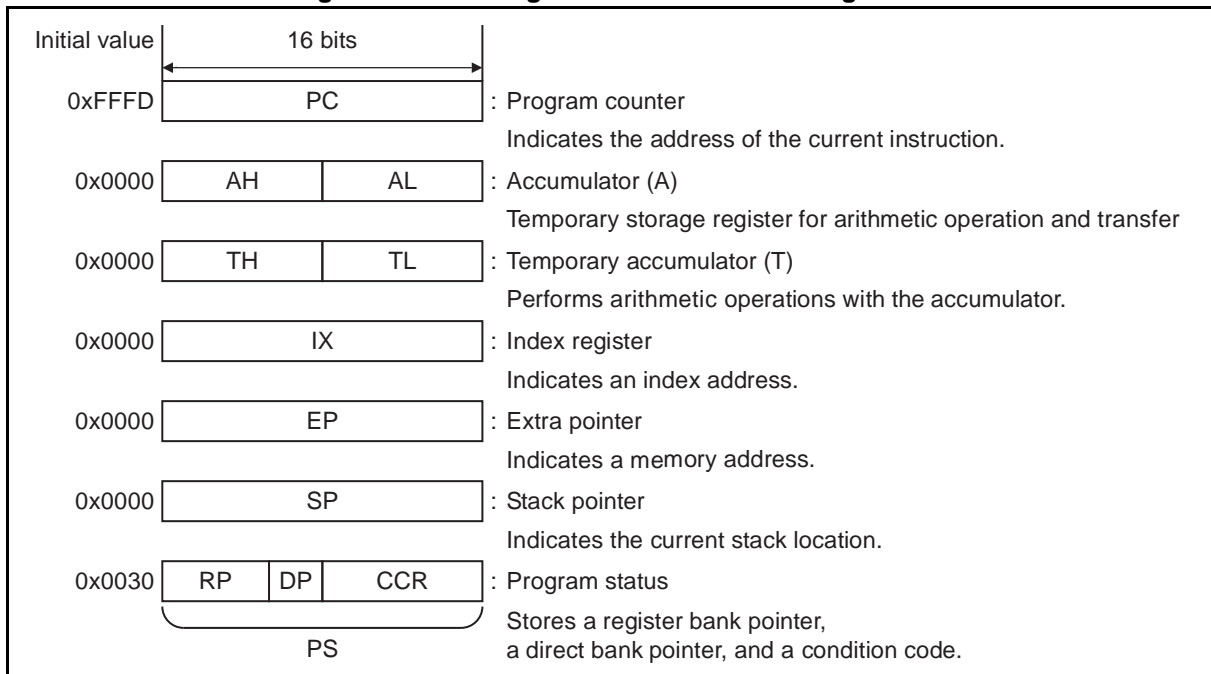
The CPU has dedicated registers: a program counter (PC), two registers for arithmetic operations (A and T), three address pointers (IX, EP, and SP), and the program status (PS) register. Each of the registers is 16 bits long. The PS register consists of the register bank pointer (RP), direct bank pointer (DP), and condition code register (CCR).

### ■ Configuration of Dedicated Registers

The dedicated registers in the CPU consist of seven 16-bit registers. As for the accumulator (A) and the temporary accumulator (T), using only the lower eight bits of the respective registers is also supported.

Figure 2.1-1 shows the configuration of the dedicated registers.

Figure 2.1-1 Configuration of Dedicated Registers



### ■ Functions of Dedicated Registers

#### ● Program counter (PC)

The program counter is a 16-bit counter which contains the memory address of the instruction currently executed by the CPU. The program counter is updated whenever an instruction is executed or an interrupt or a reset occurs. The initial value set immediately after a reset is the mode data read address (0xFFFFD).

#### ● Accumulator (A)

The accumulator is a 16-bit register for arithmetic operation. It is used for a variety of arithmetic and transfer operations of data in memory or data in other registers such as the temporary accumulator (T). The data in the accumulator can be handled either as word (16-bit) data or byte (8-bit) data. For byte-length arithmetic and transfer operations, only the lower eight bits (AL) of the accumulator are used with the upper eight bits (AH) left unchanged. The initial value set immediately after a reset is "0x0000".

## ● Temporary accumulator (T)

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

When a MOV instruction is used to transfer data to the accumulator (A), the previous contents of the accumulator are automatically transferred to the temporary accumulator. When transferring byte-length data, the upper eight bits (TH) of the temporary accumulator remain unchanged. The initial value after a reset is "0x0000".

## ● Index register (IX)

The index register is a 16-bit register used to hold the index address. The index register is used with a single-byte offset (-128 to +127). The offset value is added to the index address to generate the memory address for data access. The initial value after a reset is "0x0000".

## ● Extra pointer (EP)

The extra pointer is a 16-bit register which contains the value indicating the memory address for data access. The initial value after a reset is "0x0000".

## ● Stack pointer (SP)

The stack pointer is a 16-bit register which holds the address referenced when an interrupt or a sub-routine call occurs and by the stack push and pop instructions. During program execution, the value of the stack pointer indicates the address of the most recent data pushed onto the stack. The initial value after a reset is "0x0000".

## ● Program status (PS)

The program status is a 16-bit control register. The upper eight bits consists of the register bank pointer (RP) and direct bank pointer (DP); the lower eight bits consists of the condition code register (CCR).

In the upper eight bits, the upper five bits consists of the register bank pointer used to contain the address of the general-purpose register bank. The lower three bits consists of the direct bank pointer which locates the area to be accessed at high-speed by direct addressing.

The lower eight bits consists of the condition code register (CCR) which consists of flags that represent the state of the CPU.

The instructions that can access the program status are "MOVW A,PS" and "MOVW PS,A". The register bank pointer (RP) and direct bank pointer (DP) in the program status register can also be read from and written to by accessing the mirror address (0x0078).

Note that the condition code register (CCR) is a part of the program status register and cannot be accessed independently.

Refer to the "F<sup>2</sup>MC-8FX Programming Manual" for details on using the dedicated registers.

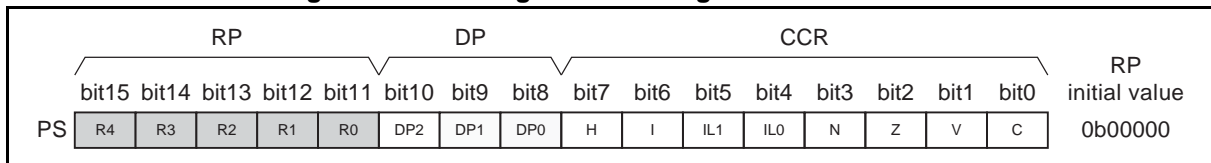
## 2.1.1 Register Bank Pointer (RP)

The register bank pointer (RP) in bit15 to bit11 of the program status (PS) register contains the address of the general-purpose register bank that is currently in use and is translated into a real address when general-purpose register addressing is used.

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

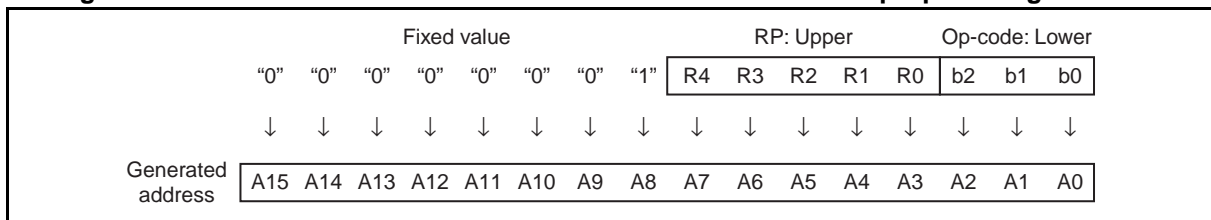
Figure 2.1-2 shows the configuration of the register bank pointer.

Figure 2.1-2 Configuration of Register Bank Pointer



The register bank pointer contains the address of the register bank currently in use. The content of the register bank pointer is translated into a real address according to the rule shown in Figure 2.1-3.

Figure 2.1-3 Rule for Translation into Real Addresses in General-purpose Register Area



The register bank pointer specifies the register bank used as general-purpose registers in the RAM area. There are a total of 32 register banks, which are specified by setting a value between 0 and 31 in the upper five bits of the register bank pointer. Each register bank has eight 8-bit general-purpose registers which are selected by the lower three bits of the op-code.

The register bank pointer allows the space from "0x0100" to "0x01FF"(max) to be used as a general-purpose register area. However, certain products have restrictions on the size of the area available for the general-purpose register area. The initial value of the register bank pointer after a reset is "0x0000".

### ■ Mirror Address for Register Bank and Direct Bank Pointer

Values can be written to the register bank pointer (RP) and the direct bank pointer (DP) by accessing the program status (PS) register with the "MOVW PS,A" instruction; the two pointers can be read by accessing PS with the "MOVW A,PS" instruction. Values can also be directly written to and read from the two pointers by accessing "0x0078", the mirror address of the register bank pointer.

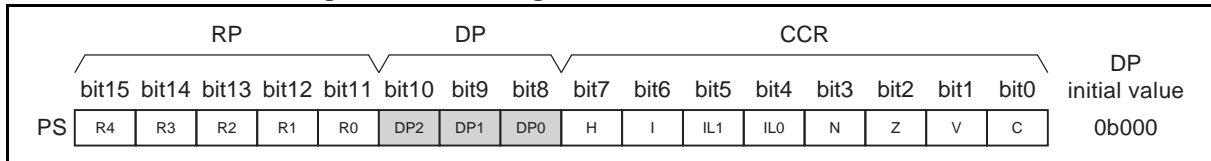
## 2.1.2 Direct Bank Pointer (DP)

The direct bank pointer (DP) in bit10 to bit8 of the program status (PS) register specifies the area to be accessed by direct addressing.

### ■ Configuration of Direct Bank Pointer (DP)

Figure 2.1-4 shows the configuration of the direct bank pointer.

Figure 2.1-4 Configuration of Direct Bank Pointer



The area of "0x0000 to 0x007F" and that of "0x0090 to 0x047F" can be accessed by direct addressing. Access to 0x0000 to 0x007F is specified by an operand regardless of the value in the direct bank pointer. Access to 0x0090 to 0x047F is specified by the value of the direct bank pointer and the operand.

Table 2.1-1 shows the relationship between the direct bank pointer (DP) and the access area; Table 2.1-2 lists the direct addressing instructions.

Table 2.1-1 Direct Bank Pointer and Access Area

| Direct bank pointer (DP[2:0])       | Operand-specified dir | Access area*     |
|-------------------------------------|-----------------------|------------------|
| 0bXXX (It does not affect mapping.) | 0x0000 to 0x007F      | 0x0000 to 0x007F |
| 0b000 (Initial value)               | 0x0090 to 0x00FF      | 0x0090 to 0x00FF |
| 0b001                               | 0x0080 to 0x00FF      | 0x0100 to 0x017F |
| 0b010                               |                       | 0x0180 to 0x01FF |
| 0b011                               |                       | 0x0200 to 0x027F |
| 0b100                               |                       | 0x0280 to 0x02FF |
| 0b101                               |                       | 0x0300 to 0x037F |
| 0b110                               |                       | 0x0380 to 0x03FF |
| 0b111                               |                       | 0x0400 to 0x047F |

\*: The available access area varies among products. For details, refer to the device data sheet.

**Table 2.1-2 Direct Address Instruction List**

| Applicable instructions |
|-------------------------|
| CLRB dir:bit            |
| SETB dir:bit            |
| BBC dir:bit,rel         |
| BBS dir:bit,rel         |
| 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,#imm            |
| CMP dir,#imm            |
| MOVW A,dir              |
| MOVW dir,A              |

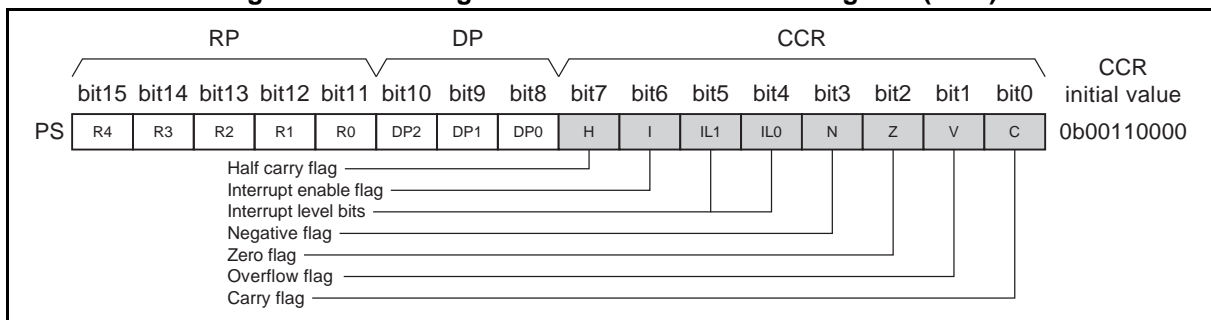
## MB95850K/860K/870K Series

### 2.1.3 Condition Code Register (CCR)

The condition code register (CCR) in the lower eight bits of the program status (PS) register consists of the bits (H, N, Z, V, and C) containing information about the arithmetic result or transfer data and the bits (I, IL1, and IL0) used to control the acceptance of interrupt requests.

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

Figure 2.1-5 Configuration of Condition Code Register (CCR)



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

#### ■ Bits Showing Operation Results

##### ● Half carry flag (H)

This flag is set to "1" when a carry from bit3 to bit4 or a borrow from bit4 to bit3 occurs due to the result of an operation. Otherwise, the flag is set to "0". Do not use this flag for any operation other than addition and subtraction as the flag is intended for decimal-adjusted instructions.

##### ● Negative flag (N)

This flag is set to "1" when the value of the most significant bit is "1" due to the result of an operation, and is set to "0" when the value of the most significant bit is "0".

##### ● Zero flag (Z)

This flag is set to "1" when the result of an operation is "0", and is set to "0" when the result of an operation is a value other than "0".

##### ● Overflow flag (V)

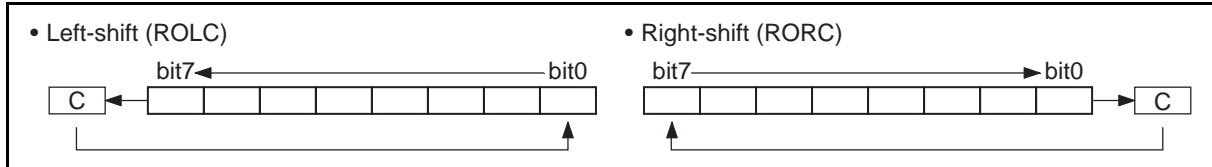
This flag indicates whether the result of an operation has caused an overflow, with the operand used in the operation being regarded as an integer expressed as a complement of two. If an overflow occurs, the overflow flag is set to "1"; otherwise, it is set to "0".

● Carry flag (C)

This flag is set to "1" when a carry from bit7 or a borrow to bit7 occurs due to the result of an operation. Otherwise, the flag is set to "0". When a shift instruction is executed, the flag is set to the shift-out value.

Figure 2.1-6 shows how the carry flag is updated by a shift instruction.

**Figure 2.1-6 Carry Flag Updated by Shift Instruction**



■ Interrupt Acceptance Control Bits

● Interrupt enable flag (I)

When this flag is set to "1", interrupts are enabled and accepted by the CPU. When this flag is set to "0", interrupts are disabled and rejected by the CPU.

The initial value after a reset is "0".

The SETI and CLRI instructions set and clear the flag to "1" and "0", respectively.

● Interrupt level bits (IL[1:0])

These bits indicate the level of the interrupt currently accepted by the CPU.

The interrupt level is compared with the value of the interrupt level setting register (ILR0 to ILR5) that corresponds to the interrupt request (IRQ00 to IRQ23) of each peripheral function.

The CPU services an interrupt request only when its interrupt level is smaller than the value of these bits with the interrupt enable flag set (CCR:I = 1). Table 2.1-3 lists interrupt level priorities. The initial value after a reset is "0b11".

**Table 2.1-3 Interrupt Levels**

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

The interrupt level bits (IL[1:0]) are usually "0b11" when the CPU does not service an interrupt (with the main program running).

For details of interrupts, see "5.1 Interrupts".



## 2.2 General-purpose Register

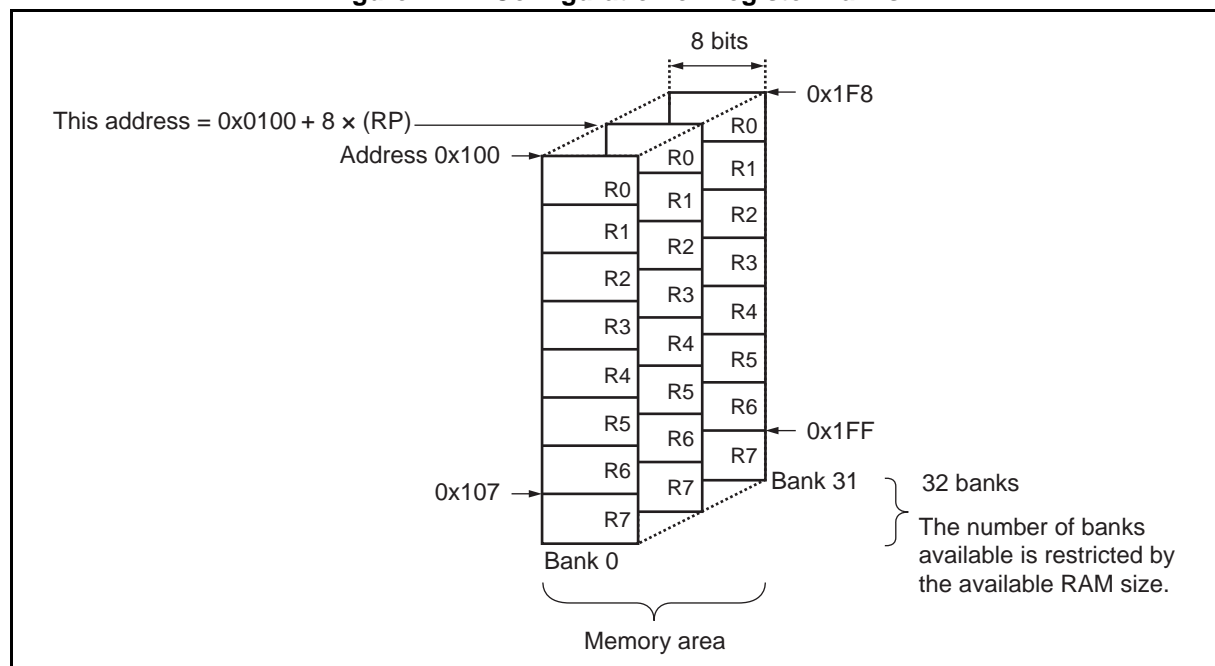
The general-purpose registers are a memory block in which each bank consists of eight 8-bit registers. Up to 32 register banks can be used in total. The register bank pointer (RP) is used to specify a register bank. Register banks are useful for interrupt handling, vector call processing, and sub-routine calls.

### ■ Configuration of General-purpose Register

- The general-purpose register is an 8-bit register and is located in a register bank in the general-purpose register area (in RAM).
- Up to 32 banks can be used, each of which consists of eight registers (R0 to R7).
- The register bank pointer (RP) specifies the register bank currently being used and the lower three bits of the op-code specify the general-purpose register 0 (R0) to the general-purpose register 7 (R7).

Figure 2.2-1 shows the configuration of the register banks.

Figure 2.2-1 Configuration of Register Banks



For information on the general-purpose register area available on each product, see "■ AREAS FOR SPECIFIC APPLICATIONS" in the device data sheet.

### ■ Features of General-purpose Registers

The general-purpose register has the following features.

- High-speed access to RAM with short instructions (general-purpose register addressing).
- Grouping registers into a block of register banks facilitates data protection and division of registers in terms of functions.

A general-purpose register bank can be allocated exclusively to an interrupt service routine or a vector call (CALLV #0 to #7) service routine. For instance, the fourth register bank is always assigned to the second interrupt.

Data of a general-purpose register before an interrupt can be saved to a dedicated register bank by just specifying that register bank at the beginning of an interrupt service routine. This therefore eliminates the need to save data of a general-purpose register in a stack, thereby enabling the CPU to receive interrupts at high speed.

---

Note:

In an interrupt service routine, include one of the following in a program to ensure that values of the interrupt level bits (CCR:IL[1:0]) of the condition code register are not modified when modifying a register bank pointer (RP) to specify a register bank.

- Read the interrupt level bits and save their values before writing a value to the RP.
  - Directly write a new value to the RP mirror address "0x0078" to update the RP.
  - As for a product whose RAM size is 256 bytes, the area available for general-purpose registers is from "0x0100" to "0x018F", which is half of that of the product whose RAM size is 512 bytes or above. Therefore, when using a program development tool such as a C compiler to set a general-purpose register area, ensure that the area used as a general-purpose register area does not exceed the size of RAM installed.
-

## 2.3 Placement of 16-bit Data in Memory

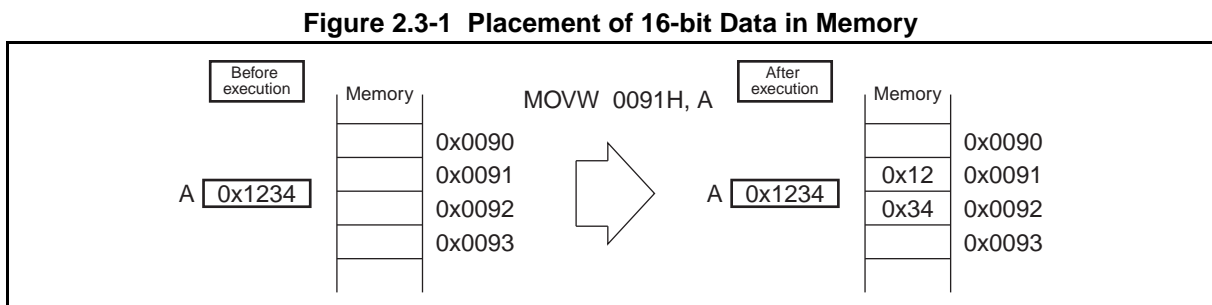
This section describes how 16-bit data is stored in memory.

### ■ Placement of 16-bit Data in Memory

- State of 16-bit data stored in RAM

When 16-bit data is written to memory, the upper byte of the data is stored at a smaller address and the lower byte is stored at the next address. When 16-bit data is read, it is handled in the same way.

Figure 2.3-1 shows how 16-bit data is placed in memory.

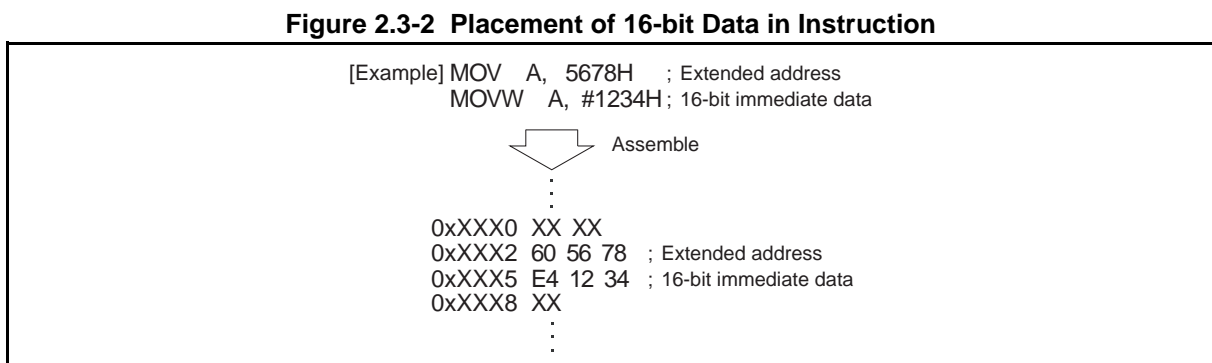


- Storage state of 16-bit data specified by an operand

Even when the operand in an instruction specifies 16-bit data, the upper byte is stored at the address closer to the op-code (instruction) and the lower byte is stored at the address next to the one at which the upper byte is stored.

That is true whether an operand is either a memory address or 16-bit immediate data.

Figure 2.3-2 shows how 16-bit data in an instruction is placed.



- Storage state of 16-bit data in the stack

When 16-bit register data is saved in a stack on an interrupt, the upper byte is stored at a lower address in the same way as 16-bit data specified by an operand.



# CHAPTER 3

---

# CLOCK CONTROLLER

**This chapter describes the functions and operations of the clock controller.**

- 3.1 Overview
- 3.2 Oscillation Stabilization Wait Time
- 3.3 Registers
- 3.4 Clock Modes
- 3.5 Operations in Low Power Consumption Mode (Standby Mode)
- 3.6 Clock Oscillator Circuit
- 3.7 Overview of Prescaler
- 3.8 Configuration of Prescaler
- 3.9 Operation of Prescaler
- 3.10 Notes on Using Prescaler

## **3.1 Overview**

---

**The New 8FX family has a built-in clock controller that optimizes its power consumption. It supports both of the external main clock and the external subclock.**

**The clock controller enables/disables clock oscillation, enables/disables the supply of clock signals to the internal circuit, selects the clock source, and controls the internal CR oscillator and frequency divider circuits.**

---

### **■ Overview of Clock Controller**

The clock controller enables/disables clock oscillation, enables/disables clock supply to the internal circuit, selects the clock source, and controls the internal CR oscillator and frequency divider circuits.

The clock controller controls the internal clock according to the clock mode, standby mode settings and the reset operation. The clock mode is used to select an internal operating clock; the standby mode is used to enable or disable clock oscillation and signal supply.

The clock controller selects the optimum power consumption and functions depending on the combination of clock mode and standby mode.

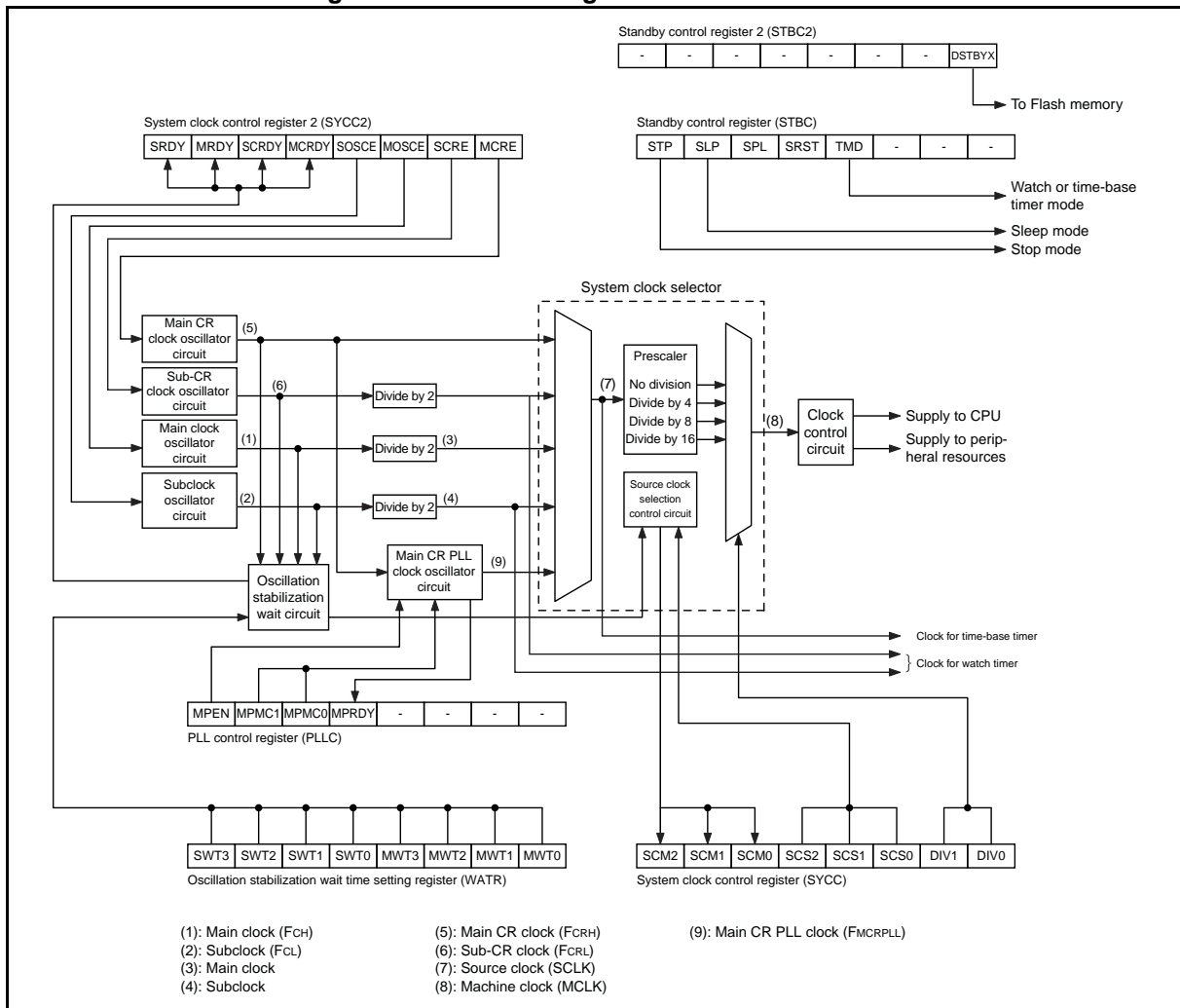
This device has five source clocks: a main clock formed by dividing the main oscillation clock by two, a subclock formed by dividing the suboscillation clock by two, a main CR clock, a main CR PLL clock formed by multiplying the main CR clock by the PLL multiplication rate, and a sub-CR clock formed by dividing the sub-CR oscillation clock by two.

# MB95850K/860K/870K Series

## ■ Block Diagram of Clock Controller

Figure 3.1-1 is the block diagram of the clock controller.

**Figure 3.1-1 Block Diagram of Clock Controller**



## ■ Configuration of Clock Controller

- Main clock oscillator circuit

This block is the oscillator circuit for the main clock.

- Subclock oscillator circuit

This block is the oscillator circuit for the subclock.

- Main CR clock oscillator circuit

This block is the oscillator circuit for the main CR clock.

- Main CR PLL clock oscillator circuit

This block is the oscillator circuit for the main CR PLL clock.

- Sub-CR clock oscillator circuit

This block is the oscillator circuit for the sub-CR clock.

- System clock selector

This block selects a clock according to the clock mode used from the following five types of source clock: main clock, subclock, main CR clock, main CR PLL clock and sub-CR clock. The source clock selected is divided by the prescaler. The divided clock is called "machine clock", which is to be supplied to the clock control circuit.

- Clock control circuit

This block controls the supply of the machine clock to the CPU and each peripheral function according to the standby mode used or oscillation stabilization wait time.

- Oscillation stabilization wait circuit

This block outputs oscillation stabilization wait time signals according to clocks that are enabled to operate.

In the case of main clock, its oscillation stabilization signal can be selected from 14 types of oscillation stabilization signals created by a dedicated timer in the oscillation stabilization wait circuit. In case of subclock, its oscillation stabilization signal can be selected from 15 types of oscillation stabilization signals created by the same dedicated timer.

- System clock control register (SYCC)

This register selects a clock mode and a machine clock divide ratio, and indicates the current clock mode.

- PLL control register (PLLC)

This register controls the main CR PLL clock multiplication rate settings.

- Standby control register (STBC)

This register controls the transition from RUN state to standby mode, the setting of pin states in stop mode, time-base timer mode, or watch mode, and the generation of software resets.



## MB95850K/860K/870K Series

- System clock control register 2 (SYCC2)

This register enables or disables the oscillations of the main clock, main CR clock, subclock, and sub-CR clock, and displays the ready signals of main clock oscillation, main CR clock oscillation, subclock oscillation and sub-CR clock oscillation.

- Oscillation stabilization wait time setting register (WATR)

This register sets the oscillation stabilization wait times for the main clock and subclock.

- Standby control register 2 (STBC2)

This register controls the deep standby mode.

■ **Clock Modes**

There are five clock modes:

- Main clock mode
- Main CR clock mode
- Main CR PLL clock mode
- Subclock mode
- Sub-CR clock mode.

Table 3.1-1 shows the relationships between the clock modes and the machine clock (operating clock for the CPU and peripheral functions).

**Table 3.1-1 Clock Modes and Machine Clock Selection**

| Clock mode             | Machine clock   |
|------------------------|---|
| Main clock mode        | The machine clock is generated by dividing the main clock by two.                             |
| Main CR clock mode     | The machine clock is generated from the main CR clock.  |
| Main CR PLL clock mode | The machine clock is generated by multiplying the main CR clock by a PLL multiplication rate. |
| Subclock mode          | The machine clock is generated by dividing the subclock by two.                               |
| Sub-CR clock mode      | The machine clock is generated by dividing the sub-CR clock by two.                           |

In any clock mode, the frequency of a selected clock can be divided.

# MB95850K/860K/870K Series

## ■ Standby Mode

The clock controller selects whether to enable or disable clock oscillation and clock supply to the internal circuitry according to the standby mode selected. With the exception of time-base timer mode and watch mode, the standby mode can be set independently of the clock mode.

Table 3.1-2 shows the relationships between standby modes and clock supply states.

**Table 3.1-2 Standby Mode and Clock Supply States**

| Standby mode         | Clock supply state  |
|----------------------|---|
| Sleep mode           | Clock supply to the CPU is stopped. As a result, the CPU stops operating, but other peripheral functions continue operating.  |
| Time-base timer mode | Clock signals are only supplied to the time-base timer and the watch prescaler, while the clock supply to other circuits is stopped. As a result, all the functions other than the time-base timer, watch prescaler, external interrupt, and low-voltage detection reset (option) are stopped.<br>The time-base timer mode can be used in main clock mode, main CR clock mode and main CR PLL clock mode. |
| Watch mode           | Main clock oscillation is stopped. Clock signals are supplied only to the watch prescaler, while clock supply to other circuits is stopped. As a result, all the functions other than the watch prescaler, external interrupt, and low-voltage detection reset (option) are stopped.<br>The watch mode is the standby mode that can be used in subclock mode and sub-CR clock mode.                       |
| Stop mode            | Main clock oscillation and subclock oscillation are stopped, and clock supply to all circuits is stopped. As a result, all the functions other than external interrupt and low-voltage detection reset (option) are stopped.  |

In every standby mode, two further operating mode options, normal standby mode and deep standby mode, can be selected by the deep standby mode control bit in the standby control register 2 (STBC2:DSTBYX).

For details, see "3.5.1 Notes on Using Standby Mode".

---

**Note:**

Clocks that are not mentioned in Table 3.1-2 are supplied under particular settings.

For example, with main clock mode being used in stop mode, when SYCC2:SOSCE or SYCC2:SCRE has been set to "1", the watch prescaler continues its operation.

In addition, with the hardware watchdog timer already started, the watchdog timer operates also in standby mode, depending on the settings of the non-volatile register (NVR) interface. For details of the non-volatile register (NVR) interface, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

---

■ **Combinations of Clock Mode and Standby Mode**

Table 3.1-3 and Table 3.1-4 list the combinations of clock mode and standby mode, and the respective operating states of different internal circuits with different combinations of clock mode and standby mode.

**Table 3.1-3 Combinations of Standby Mode and Clock Mode, and Internal Operating States (1)**

| Function                            | RUN                         |   |                         |                         | Sleep                       |   |                          |                         |
|-------------------------------------|-----------------------------|---|-------------------------|-------------------------|-----------------------------|---|--------------------------|-------------------------|
|                                     | Main clock mode             | Main CR clock mode/<br>Main CR PLL clock mode | Subclock mode           | Sub-CR clock mode       | Main clock mode             | Main CR clock mode/<br>Main CR PLL clock mode | Subclock mode            | Sub-CR clock mode       |
| Main clock                          | Operating                   | Stopped <sup>*1</sup>                         | Stopped                 |                         | Operating                   | Stopped <sup>*1</sup>                         | Stopped                  |                         |
| Main CR clock/<br>Main CR PLL clock | Stopped <sup>*2</sup>       | Operating                                     | Stopped                 |                         | Stopped <sup>*2</sup>       | Operating                                     | Stopped                  |                         |
| Subclock                            | Operating <sup>*3</sup>     |   | Operating               | Operating <sup>*3</sup> | Operating <sup>*3</sup>     |   | Operating                | Operating <sup>*3</sup> |
| Sub-CR clock                        | Operating <sup>*4</sup>     |   | Operating <sup>*4</sup> | Operating               | Operating <sup>*4</sup>     |   | Operating <sup>*4</sup>  | Operating               |
| CPU                                 | Operating                   |   | Operating               |                         | Stopped                     |   | Stopped                  |                         |
| Flash memory                        | Operating                   |   | Operating               |                         | Value held <sup>*6</sup>    |   | Value held <sup>*6</sup> |                         |
| RAM                                 | Operating                   |   | Operating               |                         | Value held                  |   | Value held               |                         |
| I/O ports                           | Operating                   |   | Operating               |                         | Output held                 |   | Output held              |                         |
| Time-base timer                     | Operating                   |   | Stopped                 |                         | Operating                   |   | Stopped                  |                         |
| Watch prescaler                     | Operating <sup>*3, *4</sup> |   | Operating               |                         | Operating <sup>*3, *4</sup> |   | Operating                |                         |
| External interrupt                  | Operating                   |   | Operating               |                         | Operating                   |   | Operating                |                         |
| Hardware watchdog timer             | Operating                   |   | Operating               |                         | Operating <sup>*5</sup>     |   | Operating <sup>*5</sup>  |                         |
| Software watchdog timer             | Operating                   |   | Operating               |                         | Stopped                     |   | Stopped                  |                         |
| Low-voltage detection reset         | Operating                   |   | Operating               |                         | Operating                   |   | Operating                |                         |
| Touch sensor                        | Operating                   |   | Operating               |                         | Operating                   |   | Operating                |                         |
| Other peripheral functions          | Operating                   |   | Operating               |                         | Operating                   |   | Operating                |                         |

\*1: The main clock runs when the main clock oscillation enable bit in the system clock control register 2 (SYCC2:MOSCE) is set to "1".

\*2: The main CR clock or the main CR PLL clock runs when main CR clock oscillation enable bit in the system clock control register 2 (SYCC2:MCRE) is set to "1".

\*3: The module runs when the subclock oscillation enable bit in the system clock control register 2 (SYCC2:SOSCE) is set to "1".

\*4: The module runs when the sub-CR clock oscillation enable bit in the system clock control register 2 (SYCC2:SCRE) is set to "1".

\*5: The hardware watchdog timer stops when the hardware watchdog timer is disabled by the non-volatile register (NVR) interface.

\*6: The state of the Flash memory in a standby mode can be selected from two options, normal state and low-power state, by the deep standby mode control bit in the standby control register 2 (STBC2:DSTBYX).

# MB95850K/860K/870K Series

**Table 3.1-4 Combinations of Standby Mode and Clock Mode and Internal Operating States (2)**

| Function                            | Time-base timer             |   | Watch                    |                         | Stop                       |   |               |                   |
|-------------------------------------|-----------------------------|---|--------------------------|-------------------------|----------------------------|---|---------------|-------------------|
|                                     | Main clock mode             | Main CR clock mode/<br>Main CR PLL clock mode | Subclock mode            | Sub-CR clock mode       | Main clock mode            | Main CR clock mode/<br>Main CR PLL clock mode | Subclock mode | Sub-CR clock mode |
| Main clock                          | Operating                   | Stopped <sup>*1</sup>                         | Stopped                  |                         | Stopped                    |   |               |                   |
| Main CR clock/<br>Main CR PLL clock | Stopped <sup>*2</sup>       | Operating                                     | Stopped                  |                         | Stopped                    |   |               |                   |
| Subclock                            | Operating <sup>*3</sup>     |   | Operating                | Operating <sup>*3</sup> | Operating <sup>*3</sup>    |   | Stopped       |                   |
| Sub-CR clock                        | Operating <sup>*4</sup>     |   | Operating <sup>*4</sup>  | Operating               | Operating <sup>*4</sup>    |   | Stopped       |                   |
| CPU                                 | Stopped                     |   | Stopped                  |                         | Stopped                    |   |               |                   |
| Flash memory                        | Value held <sup>*6</sup>    |   | Value held <sup>*6</sup> |                         | Value held <sup>*6</sup>   |   |               |                   |
| RAM                                 | Value held                  |   | Value held               |                         | Value held                 |   |               |                   |
| I/O ports                           | Output held / Hi-Z          |   | Output held/Hi-Z         |                         | Output held/Hi-Z           |   |               |                   |
| Time-base timer                     | Operating                   |   | Stopped                  |                         | Stopped                    |   |               |                   |
| Watch prescaler                     | Operating <sup>*3, *4</sup> |   | Operating                |                         | Operating <sup>*3, 4</sup> |   | Stopped       |                   |
| External interrupt                  | Operating                   |   | Operating                |                         | Operating                  |   |               |                   |
| Hardware watchdog timer             | Operating <sup>*5</sup>     |   | Operating <sup>*5</sup>  |                         | Operating <sup>*5</sup>    |   |               |                   |
| Software watchdog timer             | Stopped                     |   | Stopped                  |                         | Stopped                    |   |               |                   |
| Low-voltage detection reset         | Operating                   |   | Operating                |                         | Operating                  |   |               |                   |
| Touch sensor                        | Operating <sup>*7</sup>     |   | Operating <sup>*7</sup>  |                         | Operating <sup>*7</sup>    |   | Stopped       |                   |
| Other peripheral functions          | Stopped                     |   | Stopped                  |                         | Stopped                    |   |               |                   |

- \*1: The main clock runs when the main clock oscillation enable bit in the system clock control register 2 (SYCC2:MOSCE) is set to "1".
- \*2: The main CR clock or the main CR PLL clock runs when main CR clock oscillation enable bit in the system clock control register 2 (SYCC2:MCRE) is set to "1".
- \*3: The module runs when the subclock oscillation enable bit in the system clock control register 2 (SYCC2:SOSCE) is set to "1".
- \*4: The module runs when the sub-CR clock oscillation enable bit in the system clock control register 2 (SYCC2:SCRE) is set to "1".
- \*5: The hardware watchdog timer stops when the hardware watchdog timer is disabled by the non-volatile register (NVR) interface.
- \*6: The state of the Flash memory in a standby mode can be selected from two options, normal state and low-power state, by the deep standby mode control bit in the standby control register 2 (STBC2:DSTBYX).
- \*7: The touch sensor runs when the SCRE bit in the SYCC2 register and the stop mode enable bit in the TS prescaler control register (PSC:STPE) are both set to "1".

## 3.2 Oscillation Stabilization Wait Time

The oscillation stabilization wait time is the time after the oscillator circuit stops oscillation until the oscillator resumes its stable oscillation at its natural frequency. The clock controller obtains the oscillation stabilization wait time after the start of oscillation by counting a specific number of oscillation clock cycles. During the oscillation stabilization wait time, the clock controller stops clock supply to internal circuits.

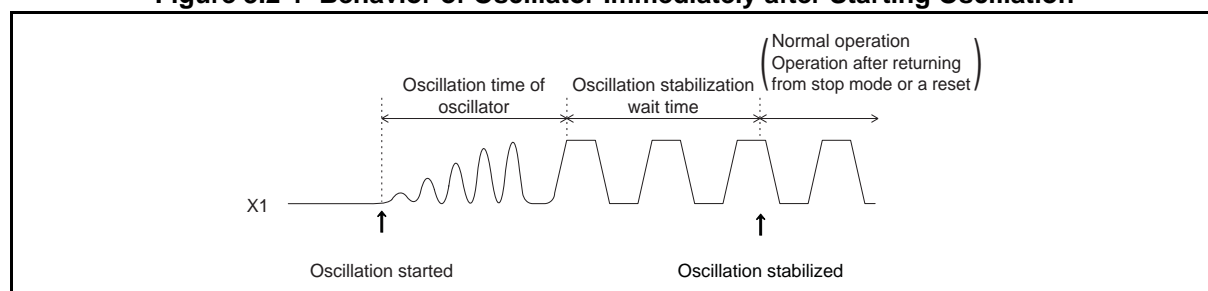
### ■ Oscillation Stabilization Wait Time

The clock controller obtains the oscillation stabilization wait time after the start of oscillation by counting a specific number of oscillation clock cycles. During the oscillation stabilization wait time, the clock controller stops clock supply to internal circuits.

When the power is switched on, or when a state transition request making the oscillator start from the oscillation stop state is generated due to a change of clock mode caused by a reset, by an interrupt in stop mode or by the software operation, before making the clock mode transit to another mode, the clock controller automatically waits for the oscillation stabilization wait time of the clock for that mode to elapse.

Figure 3.2-1 shows how the oscillator runs immediately after starting oscillating.

Figure 3.2-1 Behavior of Oscillator Immediately after Starting Oscillation



Oscillation stabilization wait time of main clock, subclock, main CR clock, main CR PLL clock or sub-CR clock is counted by using a dedicated counter. The count value can be set in the oscillation stabilization wait time setting register (WATR). Set it in keeping with the oscillator characteristics.

When a power-on reset occurs, the oscillation stabilization wait time is fixed at the initial value.

Table 3.2-1 shows the length of oscillation stabilization wait time.

Table 3.2-1 Oscillation Stabilization Wait Time

| Clock      | Reset source              | Oscillation stabilization wait time                                   |
|------------|---------------------------|---|
| Main clock | Power-on reset            | Initial value: $(2^{14}-2)/F_{CH}$ ( $F_{CH}$ : main clock frequency) |
|            | Other than power-on reset | Register settings (WATR:MWT[3:0])                                     |
| Subclock   | Power-on reset            | Initial value: $(2^{15}-2)/F_{CL}$ ( $F_{CL}$ : subclock frequency)   |
|            | Other than power-on reset | Register settings (WATR:SWT[3:0])                                     |

■ PLL Clock Oscillation Stabilization Wait Time

As with the oscillation stabilization wait time of the oscillator, when a request for state transition from PLL oscillation stopped state to PLL oscillation start is generated due to an interrupt in stop mode or a change of clock mode by software, the clock controller first waits for the main CR clock oscillation stabilization wait time to elapse, and then automatically waits for the PLL clock oscillation stabilization wait time to elapse.

Table 3.2-2 shows the PLL oscillation stabilization wait time.

**Table 3.2-2 PLL Oscillation Stabilization Wait Time**

|                   | PLL oscillation stabilization wait time |
|-------------------|---|
| Main CR PLL clock | $2^{12}/F_{MCRPLL}^*$                   |

\*:  $F_{MCRPLL} = 16 \text{ MHz}$

■ CR Clock Oscillation Stabilization Wait Time

As with the oscillation stabilization wait time of the oscillator, when a state transition request making CR oscillation start from the CR oscillation stop state is generated due to a change of clock mode caused by an interrupt in standby mode or by the software operation, the clock controller automatically waits for the CR oscillation stabilization wait time to elapse.

Table 3.2-3 shows the CR oscillation stabilization wait time.

**Table 3.2-3 CR Oscillation Stabilization Wait Time**

|               | CR oscillation stabilization wait time |
|---------------|--|
| Main CR clock | $2^{10}/F_{CRH}^{*1}$                  |
| Sub-CR clock  | $2^5/F_{CRL}^{*2}$                     |

\*1:  $F_{CRH} = 4 \text{ MHz}$

\*2:  $F_{CRL} = 150 \text{ kHz}$

■ Oscillation Stabilization Wait Time and Clock Mode/Standby Mode Transition

If state transition occurs, the clock controller automatically waits for the oscillation stabilization wait time to elapse whenever necessary. Depending on the circumstances under which state transition occurs, the clock controller does not wait for the oscillation stabilization wait time to elapse even if state transition occurs.

For details on state transition, see "3.4 Clock Modes" and "3.5 Operations in Low Power Consumption Mode (Standby Mode)".

■ **Order of Priority for Oscillation Stabilization Wait Times**

When multiple clocks are enabled simultaneously, the clock controller counts the respective oscillation stabilization wait times of clocks according to a designated order of priority. Below are the respective orders of priority for counting different oscillation stabilization wait times in different clock modes.

- Main clock mode  
Sub-CR clock > Subclock > Main CR clock > Main CR PLL clock
- Main CR clock mode  
Sub-CR clock > Subclock > Main CR PLL clock > Main clock
- Main CR PLL clock mode  
Sub-CR clock > Subclock > Main clock
- Subclock mode  
Sub-CR clock > Main CR clock or main clock > Main CR PLL clock
- Sub-CR clock mode  
Main CR clock or main clock > Subclock > Main CR PLL clock



### 3.3 Registers

---

**This section provides details of registers of the clock controller.**

---

**Table 3.3-1 List of Clock Controller Registers**

| Register abbreviation | Register name  | Reference |
|-----------------------|--|-----------|
| SYCC                  | System clock control register                        | 3.3.1     |
| PLLC                  | PLL control register                                 | 3.3.2     |
| WATR                  | Oscillation stabilization wait time setting register | 3.3.3     |
| STBC                  | Standby control register                             | 3.3.4     |
| SYCC2                 | System clock control register 2                      | 3.3.5     |
| STBC2                 | Standby control register 2                           | 3.3.6     |

### 3.3.1 System Clock Control Register (SYCC)

The system clock control register (SYCC) selects a machine clock divide ratio and a clock mode, and indicates the current clock mode.

#### Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | SCM2 | SCM1 | SCM0 | SCS2 | SCS1 | SCS0 | DIV1 | DIV0 |
| Attribute     | R    | R    | R    | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | X    | X    | X    | 1    | 1    | 0    | 1    | 1    |

#### Register Functions

[bit7:5] SCM[2:0]: Clock mode monitor bits

These bits indicate the current clock mode.

These bits are read-only bits. Writing values to these bits has no effect on operation.

| bit7:5        | Details  |
|---------------|--|
| Reading "000" | Indicates that the current clock mode is subclock mode.          |
| Reading "010" | Indicates that the current clock mode is main clock mode.        |
| Reading "100" | Indicates that the current clock mode is sub-CR clock mode.      |
| Reading "110" | Indicates that the current clock mode is main CR clock mode.     |
| Reading "111" | Indicates that the current clock mode is main CR PLL clock mode. |

[bit4:2] SCS[2:0]: Clock mode select bits

These bits select a clock mode.

| bit4:2        | Details                         |
|---------------|---------------------------------|
| Writing "000" | Selects subclock mode.          |
| Writing "010" | Selects main clock mode.        |
| Writing "100" | Selects sub-CR clock mode.      |
| Writing "110" | Selects main CR clock mode.     |
| Writing "111" | Selects main CR PLL clock mode. |

Note: Do not write to SCS[2:0] any value other than those listed in the table above.

[bit1:0] DIV[1:0]: Machine clock divide ratio select bits

These bits select the machine clock divide ratio for the source clock.

The machine clock is generated from the source clock according to the divide ratio set by these bits.

| bit1:0       | Details                    |
|--------------|----------------------------|
| Writing "00" | Source clock (no division) |
| Writing "01" | Source clock/4             |
| Writing "10" | Source clock/8             |
| Writing "11" | Source clock/16            |

## MB95850K/860K/870K Series

### 3.3.2 PLL Control Register (PLLC)

The PLL control register (PLLC) controls the main CR PLL clock multiplication rate settings.

#### Register Configuration

| bit           | 7    | 6     | 5     | 4     | 3 | 2 | 1 | 0 |
|---------------|------|-------|-------|-------|---|---|---|---|
| Field         | MPEN | MPMC1 | MPMC0 | MPRDY | — | — | — | — |
| Attribute     | R/W  | R/W   | R/W   | R     | — | — | — | — |
| Initial value | 0    | 0     | 0     | X     | 0 | 0 | 0 | 0 |

#### Register Functions

[bit7] MPEN: Main CR PLL clock enable bit

This bit enables or disables the main CR PLL clock.

When SCS[2:0] are set to "0b111", this bit is automatically set to "1".

When SCS[2:0] or SCM[2:0] are set to "0b111", writing "0" to this bit has no effect on operation.

This bit is automatically set to "0" when the clock mode transits from one mode to another mode except main CR PLL clock mode.

When the current clock mode is subclock mode or sub-CR clock mode, writing "1" to this bit has no effect on operation.

| bit7               | Details                         |
|--------------------|---------------------------------|
| <b>Writing "0"</b> | Disables the main CR PLL clock. |
| <b>Writing "1"</b> | Enables the main CR PLL clock.  |

[bit6:5] MPMC[1:0]: Main CR PLL clock multiplication rate select bits

These bits select a main CR PLL clock multiplication rate.

The settings of these bits can be modified only when the main CR PLL clock is stopped. Thus these bits can be modified in main clock mode, main CR clock mode, subclock mode or sub-CR clock mode.

| bit6:5              | Details             |
|---------------------|---------------------|
| <b>Writing "00"</b> | Main CR clock × 2   |
| <b>Writing "01"</b> | Main CR clock × 2.5 |
| <b>Writing "10"</b> | Main CR clock × 3   |
| <b>Writing "11"</b> | Main CR clock × 4   |

Note: When SCS[2:0] or SCM[2:0] are set to "0b111", writing values to MPMC[1:0] is prohibited.

[bit4] MPRDY: Main CR PLL clock oscillation stabilization bit

This bit indicates whether the main CR PLL clock oscillation is ready.

| bit4               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that main CR PLL clock is in the oscillation stabilization wait state or that the main CR PLL clock oscillation has stopped. |
| <b>Reading "1"</b> | Indicates that the main CR PLL clock oscillation wait time is over.  |

[bit3:0] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

### 3.3.3 Oscillation Stabilization Wait Time Setting Register (WATR)

The oscillation stabilization wait time setting register (WATR) selects oscillation stabilization wait times.

#### ■ Register Configuration

|               |      |      |      |      |      |      |      |      |
|---------------|------|------|------|------|------|------|------|------|
| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
| Field         | SWT3 | SWT2 | SWT1 | SWT0 | MWT3 | MWT2 | MWT1 | MWT0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 1    | 1    | 1    | 1    | 1    | 1    | 1    | 1    |

#### ■ Register Functions

[bit7:4] SWT[3:0]: Subclock oscillation stabilization wait time select bits

These bits select the subclock oscillation stabilization wait time.

| bit7:4         | Details       |                                    |                     |
|----------------|---------------|------------------------------------|---------------------|
|                | No. of cycles | Subclock ( $F_{CL}$ ) = 32.768 kHz |                     |
| Writing "1111" | $2^{15} - 2$  | $(2^{15}-2)/F_{CL}$                | About 1.0 s         |
| Writing "1110" | $2^{14} - 2$  | $(2^{14}-2)/F_{CL}$                | About 0.5 s         |
| Writing "1101" | $2^{13} - 2$  | $(2^{13}-2)/F_{CL}$                | About 0.25 s        |
| Writing "1100" | $2^{12} - 2$  | $(2^{12}-2)/F_{CL}$                | About 0.125 s       |
| Writing "1011" | $2^{11} - 2$  | $(2^{11}-2)/F_{CL}$                | About 62.44 ms      |
| Writing "1010" | $2^{10} - 2$  | $(2^{10}-2)/F_{CL}$                | About 31.19 ms      |
| Writing "1001" | $2^9 - 2$     | $(2^9-2)/F_{CL}$                   | About 15.56 ms      |
| Writing "1000" | $2^8 - 2$     | $(2^8-2)/F_{CL}$                   | About 7.75 ms       |
| Writing "0111" | $2^7 - 2$     | $(2^7-2)/F_{CL}$                   | About 3.85 ms       |
| Writing "0110" | $2^6 - 2$     | $(2^6-2)/F_{CL}$                   | About 1.89 ms       |
| Writing "0101" | $2^5 - 2$     | $(2^5-2)/F_{CL}$                   | About 915.5 $\mu$ s |
| Writing "0100" | $2^4 - 2$     | $(2^4-2)/F_{CL}$                   | About 427.2 $\mu$ s |
| Writing "0011" | $2^3 - 2$     | $(2^3-2)/F_{CL}$                   | About 183.1 $\mu$ s |
| Writing "0010" | $2^2 - 2$     | $(2^2-2)/F_{CL}$                   | About 61.0 $\mu$ s  |
| Writing "0001" | $2^1 - 2$     | $(2^1-2)/F_{CL}$                   | 0.0 $\mu$ s         |
| Writing "0000" | $2^1 - 2$     | $(2^1-2)/F_{CL}$                   | 0.0 $\mu$ s         |

The number of cycles in the above table is the minimum value. The maximum value is the number of cycles in the above table plus  $1/F_{CL}$ .

Note: Do not modify these bits during subclock oscillation stabilization wait time. Modify them when the subclock oscillation stabilization bit in the system clock control register 2 (SYCC2:SRDY) has been set to "1". These bits can be modified when the subclock is stopped with the subclock oscillation stop bit in the system clock control register 2 (SYCC2:SOSCE) set to "0" in main clock mode, main CR clock mode, main CR PLL clock mode, or sub-CR clock mode.

**MB95850K/860K/870K Series**

[bit3:0] MWT[3:0]: Main clock oscillation stabilization wait time select bits

These bits select the main clock oscillation stabilization wait time.

| bit3:0         | Details       |                                 |                     |
|----------------|---------------|---------------------------------|---------------------|
|                | No. of cycles | Main clock ( $F_{CH}$ ) = 4 MHz |                     |
| Writing "1111" | $2^{14} - 2$  | $(2^{14} - 2)/F_{CH}$           | About 4.10 ms       |
| Writing "1110" | $2^{13} - 2$  | $(2^{13} - 2)/F_{CH}$           | About 2.05 ms       |
| Writing "1101" | $2^{12} - 2$  | $(2^{12} - 2)/F_{CH}$           | About 1.02 ms       |
| Writing "1100" | $2^{11} - 2$  | $(2^{11} - 2)/F_{CH}$           | About 511.5 $\mu$ s |
| Writing "1011" | $2^{10} - 2$  | $(2^{10} - 2)/F_{CH}$           | About 255.5 $\mu$ s |
| Writing "1010" | $2^9 - 2$     | $(2^9 - 2)/F_{CH}$              | About 127.5 $\mu$ s |
| Writing "1001" | $2^8 - 2$     | $(2^8 - 2)/F_{CH}$              | About 63.5 $\mu$ s  |
| Writing "1000" | $2^7 - 2$     | $(2^7 - 2)/F_{CH}$              | About 31.5 $\mu$ s  |
| Writing "0111" | $2^6 - 2$     | $(2^6 - 2)/F_{CH}$              | About 15.5 $\mu$ s  |
| Writing "0110" | $2^5 - 2$     | $(2^5 - 2)/F_{CH}$              | About 7.5 $\mu$ s   |
| Writing "0101" | $2^4 - 2$     | $(2^4 - 2)/F_{CH}$              | About 3.5 $\mu$ s   |
| Writing "0100" | $2^3 - 2$     | $(2^3 - 2)/F_{CH}$              | About 1.5 $\mu$ s   |
| Writing "0011" | $2^2 - 2$     | $(2^2 - 2)/F_{CH}$              | About 0.5 $\mu$ s   |
| Writing "0010" | $2^1 - 2$     | $(2^1 - 2)/F_{CH}$              | 0.0 $\mu$ s         |
| Writing "0001" | $2^1 - 2$     | $(2^1 - 2)/F_{CH}$              | 0.0 $\mu$ s         |
| Writing "0000" | $2^1 - 2$     | $(2^1 - 2)/F_{CH}$              | 0.0 $\mu$ s         |

The number of cycles in the above table is the minimum value. The maximum value is the number of cycles in the above table plus  $1/F_{CH}$ .

Note: Do not modify these bits during main clock oscillation stabilization wait time. Modify them when the main clock oscillation stabilization bit in the system clock control register 2 (SYCC2:MRDY) has been set to "1". These bits can be modified when the main clock is stopped with the main clock oscillation stop bit in the system clock control register 2 (SYCC2:MOSCE) set to "0" in main CR clock mode, main CR PLL clock mode, subclock mode, or sub-CR clock mode.

### 3.3.4 Standby Control Register (STBC)

The standby control register (STBC) controls transition from the RUN state to sleep mode, stop mode, time-base timer mode, or watch mode, sets the pin state in stop mode, time-base timer mode, and watch mode, and controls the generation of software resets.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4    | 3   | 2 | 1 | 0 |
|---------------|-----|-----|-----|------|-----|---|---|---|
| Field         | STP | SLP | SPL | SRST | TMD | — | — | — |
| Attribute     | W   | W   | R/W | W    | W   | — | — | — |
| Initial value | 0   | 0   | 0   | 0    | 0   | 0 | 0 | 0 |

#### ■ Register Functions

##### [bit7] STP: Stop bit

This bit sets the transition to stop mode.  
The read value of this bit is always "0".

| bit7               | Details                                    |
|--------------------|--|
| <b>Writing "0"</b> | Has no effect on operation.                |
| <b>Writing "1"</b> | Causes the device to transit to stop mode. |

Note: After an interrupt request is generated, writing "1" to this bit is ignored. For details, see "3.5.1 Notes on Using Standby Mode".

##### [bit6] SLP: Sleep bit

This bit sets the transition to sleep mode.  
The read value of this bit is always "0".

| bit6               | Details                                     |
|--------------------|---|
| <b>Writing "0"</b> | Has no effect on operation.                 |
| <b>Writing "1"</b> | Causes the device to transit to sleep mode. |

Note: After an interrupt request is generated, writing "1" to this bit is ignored. For details, see "3.5.1 Notes on Using Standby Mode".

##### [bit5] SPL: Pin state setting bit

This bit sets the states of external pins in stop mode, time-base timer mode, and watch mode.

| bit5               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | The state (level) of an external pin in stop mode, time-base timer mode and watch mode is kept.  |
| <b>Writing "1"</b> | An external pin becomes high impedance in stop mode, time-base timer mode and watch mode. (A pin for which connection to a pull-up resistor has been selected in the pull-up register is pulled up.) |

## MB95850K/860K/870K Series

### [bit4] SRST: Software reset bit

This bit sets the software reset.

The read value of this bit is always "0".

| bit4               | Details                                   |
|--------------------|---|
| <b>Writing "0"</b> | Has no effect on operation.               |
| <b>Writing "1"</b> | Generates a 3-machine clock reset signal. |

### [bit3] TMD: Watch bit

This bit sets transition to time-base timer mode or watch mode.

Writing "1" to this bit in main clock mode, main CR clock mode, or main CR PLL clock mode causes the device to transit to time-base timer mode.

Writing "1" to this bit in subclock mode or sub-CR clock mode causes the device to transit to watch mode.

Writing "0" to this bit has no effect on operation.

The read value of this bit is always "0".

| bit3               | Details  |  |
|--------------------|--|--|
|                    | In main clock mode, main CR clock mode or main CR PLL clock mode | In subclock mode or sub-CR clock mode      |
| <b>Writing "0"</b> | Has no effect on operation.                                      | Has no effect on operation.                |
| <b>Writing "1"</b> | Causes the device to transit to time-base timer mode.            | Causes the device to transit to watch mode |

Note: After an interrupt request is generated, writing "1" to this bit is ignored. For details, see "3.5.1 Notes on Using Standby Mode".

### [bit2:0] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### Notes:

- Set a standby mode after making sure that the transition to clock mode has been completed by comparing the values of the clock mode monitor bits (SYCC:SCM[2:0]) and clock mode select bits (SYCC:SCS[2:0]) in the system clock control register.
- If two or more of the following bits, stop bit (STP), sleep bit (SLP), software reset bit (SRST) and watch bit (TMD), are set to "1" together, the order of priority for such bits is as follows:
  - (1) Software reset bit (SRST)
  - (2) Stop bit (STP)
  - (3) Watch bit (TMD)
  - (4) Sleep bit (SLP)

When released from standby mode, the device returns to the normal operating state.

### 3.3.5 System Clock Control Register 2 (SYCC2)

The system clock control register 2 (SYCC2) indicates the respective stabilization conditions of main clock oscillation, subclock oscillation, main CR clock oscillation and sub-CR clock oscillation, and controls main clock oscillation, subclock oscillation, main CR clock oscillation and sub-CR clock oscillation.

#### Register Configuration

| bit           | 7    | 6    | 5     | 4     | 3     | 2     | 1    | 0    |
|---------------|------|------|-------|-------|-------|-------|------|------|
| Field         | SRDY | MRDY | SCRDY | MCRDY | SOSCE | MOSCE | SCRE | MCRE |
| Attribute     | R    | R    | R     | R     | R/W   | R/W   | R/W  | R/W  |
| Initial value | X    | X    | X     | X     | 0     | 0     | 1    | 1    |

#### Register Functions

##### [bit7] SRDY: Subclock oscillation stabilization bit

This bit indicates whether the subclock oscillation has become stable.  
This bit is read-only. Writing a value to this bit has no effect on operation.

| bit7        | Details   |
|-------------|---|
| Reading "0" | Indicates that the clock controller is in the subclock oscillation stabilization wait state or that the subclock oscillation has stopped. |
| Reading "1" | Indicates that the subclock oscillation wait time is over.  |

##### [bit6] MRDY: Main clock oscillation stabilization bit

This bit indicates whether the main clock oscillation has become stable.  
This bit is read-only. Writing a value to this bit has no effect on operation.

| bit6        | Details   |
|-------------|---|
| Reading "0" | Indicates that the clock controller is in the main clock oscillation stabilization wait state or that the main clock oscillation has stopped. |
| Reading "1" | Indicates that the main clock oscillation wait time is over.  |

##### [bit5] SCRDY: Sub-CR clock oscillation stabilization bit

This bit indicates whether the sub-CR clock oscillation has become stable.  
This bit is read-only. Writing a value to this bit has no effect on operation.

| bit5        | Details   |
|-------------|---|
| Reading "0" | Indicates that the clock controller is in the sub-CR clock oscillation stabilization wait state or that the sub-CR clock oscillation has stopped. |
| Reading "1" | Indicates that the sub-CR clock oscillation wait time is over.  |

##### [bit4] MCRDY: Main CR clock oscillation stabilization bit

This bit indicates whether the main CR clock oscillation has become stable.  
This bit is read-only. Writing a value to this bit has no effect on operation.

| bit4        | Details   |
|-------------|---|
| Reading "0" | Indicates that the clock controller is in the main CR clock oscillation stabilization wait state or that the main CR clock oscillation has stopped. |
| Reading "1" | Indicates that the main CR clock oscillation wait time is over.   |



## MB95850K/860K/870K Series

### [bit3] SOSCE: Subclock oscillation enable bit

This bit enables or disables the subclock oscillation.

When SCS[2:0] are set to "0b000", this bit is automatically set to "1".

When SCS[2:0] or SCM[2:0] are set to "0b000", writing "0" to this bit has no effect on operation.

| bit3               | Details                            |
|--------------------|------------------------------------|
| <b>Writing "0"</b> | Disables the subclock oscillation. |
| <b>Writing "1"</b> | Enables the subclock oscillation.  |

### [bit2] MOSCE: Main clock oscillation enable bit

This bit enables or disables the main clock oscillation.

When SCS[2:0] are set to "0b010", this bit is automatically set to "1".

When SCS[2:0] or SCM[2:0] are set to "0b010", writing "0" to this bit has no effect on operation.

This bit is automatically set to "0" when the clock mode is changed from one mode to another mode other than main clock mode.

When the current clock mode is subclock mode or sub-CR clock mode, writing "1" to this bit has no effect on operation.

| bit2               | Details                              |
|--------------------|--------------------------------------|
| <b>Writing "0"</b> | Disables the main clock oscillation. |
| <b>Writing "1"</b> | Enables the main clock oscillation.  |

### [bit1] SCRE: Sub-CR clock oscillation enable bit

This bit enables or disables the sub-CR clock oscillation.

When SCS[2:0] are set to "0b100", this bit is automatically set to "1".

When SCS[2:0] or SCM[2:0] are set to "0b100", writing "0" to this bit has no effect on operation.

When SCS[2:0] and SCM[2:0] are not set to "0b100", this bit can be modified independently of other bits.

| bit1               | Details                                |
|--------------------|--|
| <b>Writing "0"</b> | Disables the sub-CR clock oscillation. |
| <b>Writing "1"</b> | Enables the sub-CR clock oscillation.  |

### [bit0] MCRE: Main CR clock oscillation enable bit

This bit enables or disables the main CR clock oscillation.

When SCS[2:0] are set to "0b110" or "0b111", this bit is automatically set to "1".

When SCS[2:0] or SCM[2:0] are set to "0b110" or "0b111", writing "0" to this bit has no effect on operation.

This bit is automatically set to "0" when the clock mode is changed from one mode to another mode except main CR clock mode or from main CR PLL clock mode.

When the current clock mode is subclock mode or sub-CR clock mode, writing "1" to this bit has no effect on operation.

| bit0               | Details                                 |
|--------------------|---|
| <b>Writing "0"</b> | Disables the main CR clock oscillation. |
| <b>Writing "1"</b> | Enables the main CR clock oscillation.  |

### 3.3.6 Standby Control Register 2 (STBC2)

The standby control register 2 (STBC2) controls the deep standby mode.

#### Register Configuration

|               |   |   |   |   |   |   |   |        |
|---------------|---|---|---|---|---|---|---|--------|
| bit           | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0      |
| Field         | — | — | — | — | — | — | — | DSTBYX |
| Attribute     | — | — | — | — | — | — | — | R/W    |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0      |

#### Register Functions

[bit7:1] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit0] DSTBYX: Deep standby mode control bit

This bit makes the device transit to deep standby mode by setting the Flash memory to the low-power state in standby mode.

| bit0        | Details   |
|-------------|---|
| Writing "0" | Sets the Flash memory to the low-power state when the device enters standby mode according to the setting of the standby control register (STBC). (deep standby mode) |
| Writing "1" | Keeps the Flash memory at the normal state when the device enters standby mode according to the setting of the standby control register (STBC). (normal standby mode) |

Notes:

- Waking up the device from deep standby mode and waking up the device from the normal standby mode have the following difference.

|   | Maximum time required to wake up the device from deep standby mode<br>(SCLK: source clock, MCLK: machine clock)       |
|---|---|
| In main clock mode, main CR clock mode, or main CR PLL clock mode | $(10 \text{ SCLK} + 150 \mu\text{s} + 6 \text{ MCLK}) +$ time required to wake up the device from normal standby mode |
| In subclock mode or sub-CR clock mode                             | $(2 \text{ SCLK} + 150 \mu\text{s} + 6 \text{ MCLK}) +$ time required to wake up the device from normal standby mode  |

- Refer to "ELECTRICAL CHARACTERISTICS" in the device data sheet for the difference between the deep standby mode and the normal standby mode in power consumption.
- Do not make the device transit to deep standby mode when a Flash command sequence (except read/reset) has been invoked.

## MB95850K/860K/870K Series

### 3.4 Clock Modes

---

**There are five clock modes: main clock mode, subclock mode, main CR clock mode, main CR PLL clock mode, and sub-CR clock mode. The clock mode switches according to the settings in the system clock control register (SYCC).**

---

#### ■ Operations in Main Clock Mode

In main clock mode, the main clock is used as the machine clock for the CPU and peripheral functions.

The time-base timer operates using the main clock.

The watch prescaler operates using the subclock or the sub-CR clock.

While the device is operating in main clock mode, it can be set to transit to one of the following standby mode: sleep mode, stop mode, or time-base timer mode.

After a reset, the device always enters main CR clock mode regardless of the clock mode used before that reset.

#### ■ Operations in Subclock Mode

In subclock mode, main clock oscillation is stopped\* and the subclock is used as the machine clock for the CPU and peripheral functions. In this mode, the time-base timer stops as it requires the main clock for operation.

While the device is operating in subclock mode, it can be set to transit to one of the following standby mode: sleep mode, stop mode, or watch mode.

#### ■ Operations in Main CR Clock Mode or Main CR PLL Clock Mode

In main CR clock mode or main CR PLL clock mode, the main CR clock or the main CR PLL clock is used as the machine clock for the CPU and peripheral functions. The time-base timer and the watchdog timer operate using the main CR clock or the main CR PLL clock.

The watch prescaler operates using the subclock or the sub-CR clock.

While the device is operating in main CR clock mode or the main CR PLL clock mode, it can be set to transit to one of the following standby mode: sleep mode, stop mode, or time-base timer mode.

#### ■ Operations in Sub-CR Clock Mode

In sub-CR clock mode, main clock oscillation is stopped\* and the sub-CR clock is used as the machine clock for the CPU and peripheral functions. In this mode, the time-base timer stops as it requires the main clock for operation. The watch prescaler operates using the sub-CR clock.

While the device is operating in sub-CR clock mode, it can be set to transit to one of the following standby mode: sleep mode, stop mode, or watch mode.

\*: The oscillation of the main clock, main CR clock, or main CR PLL clock is automatically disabled (writing "0" to SYCC2:MOSCE, SYCC2:MCRE or PLLC:MPEN respectively) when the clock mode transits from main clock mode, main CR clock mode or main CR PLL clock mode to subclock mode or sub-CR clock mode. In subclock mode or sub-CR clock mode, writing "1" to SYCC2:MOSCE, SYCC2:MCRE or PLLC:MPEN cannot enable the main clock, the main CR clock, or the main CR PLL clock respectively.

■ **Clock Mode State Transition Diagram**

There are five clock modes: main clock mode, subclock mode, main CR clock mode, main CR PLL clock mode and sub-CR clock mode. The device can switch between these modes according to the settings in the system clock control register (SYCC).

**Figure 3.4-1 Clock Mode State Transition Diagram**

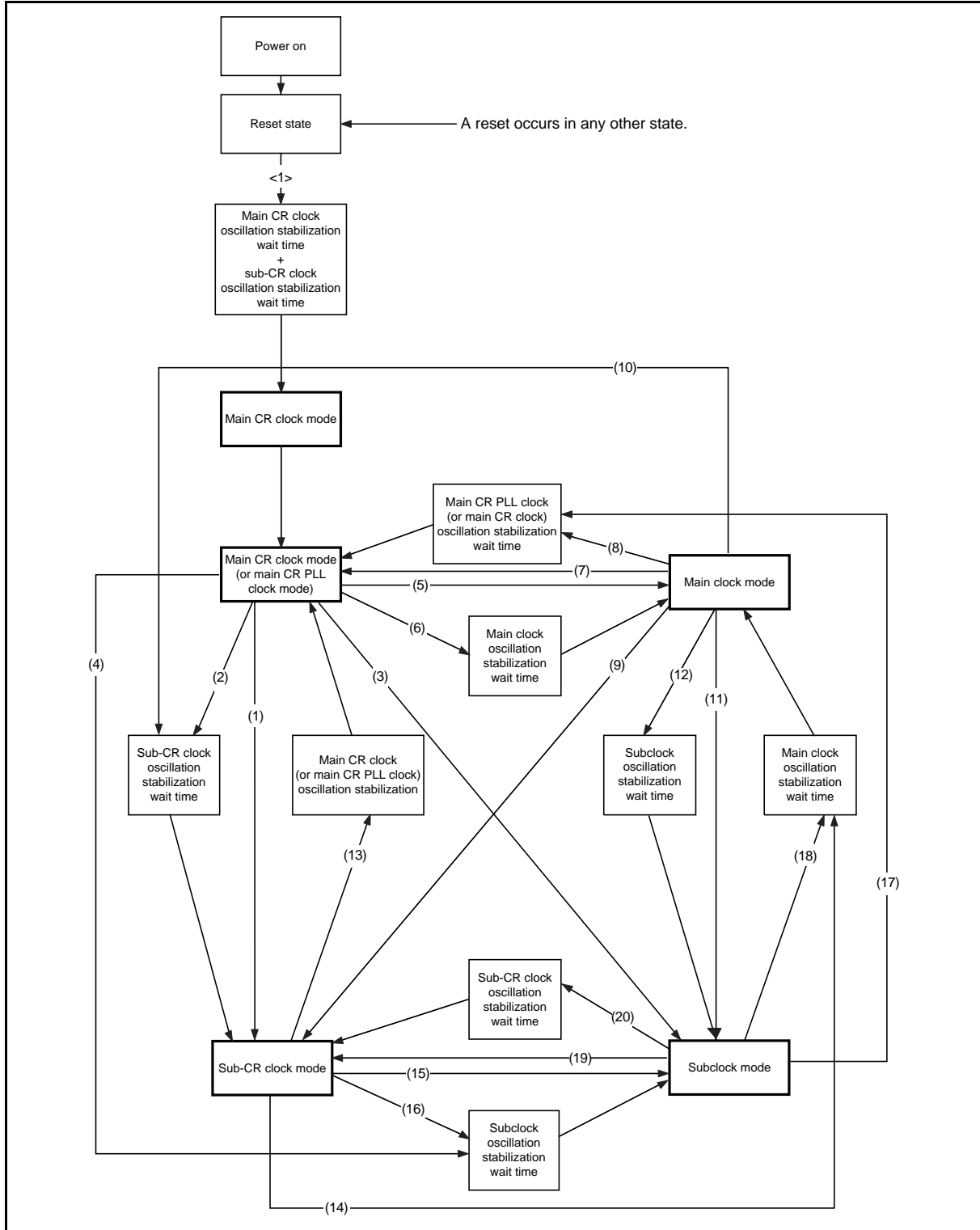


Table 3.4-1 Clock Mode State Transition Table (1 / 2)

|      | Current State                          | Next State    | Description  |
|------|--|---------------|--|
| <1>  | Reset state                            | Main CR clock | After a reset, the device waits for the main CR clock oscillation stabilization wait time and the sub-CR clock oscillation stabilization wait time to elapse and transits to main CR clock mode. Even if that reset is a watchdog reset, software reset or external reset caused in any clock mode, the device waits for the sub-CR clock oscillation stabilization wait time and the main CR clock oscillation stabilization wait time to elapse.   |
| (1)  | Main CR clock/<br>Main CR PLL<br>clock | Sub-CR clock  | The device transits to sub-CR clock mode when the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b100".<br>However, when the sub-CR has been stopped according to the setting of the sub-CR clock oscillation enable bit in the system clock control register 2 (SYCC2:SCRE), the device waits for the sub-CR clock oscillation stabilization wait time to elapse before transiting to sub-CR clock mode. In other words, when the sub-CR clock oscillation is enabled in advance, and the sub-CR clock oscillation stabilization bit in the system clock control register 2 (SYCC2:SCRDY) is "1", the device transits to sub-CR clock mode immediately after the clock mode select bits (SYCC:SCS[2:0]) are set to "0b100". |
| (2)  |  |               | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b000", the device transits to subclock mode after waiting for the subclock oscillation stabilization wait time.<br>When the subclock oscillation is enabled by the setting of the subclock oscillation enable bit in the system clock control register 2 (SYCC2:SOSCE), and the subclock oscillation stabilization bit in the system clock control register 2 (SYCC2:SRDY) is "1", the device transits to subclock mode immediately after the clock mode select bits (SYCC:SCS[2:0]) are set to "0b000".   |
| (3)  |  | Subclock      | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b010", the device transits to main clock mode after waiting for the main clock oscillation stabilization wait time.<br>When the main clock oscillation is enabled by the setting of the main clock oscillation enable bit in the system clock control register 2 (SYCC2:MOSCE), and the main clock oscillation stabilization bit in the system clock control register 2 (SYCC2:MRDY) is "1", the device transits to main clock mode immediately after the clock mode select bits (SYCC:SCS[2:0]) are set to "0b010".   |
| (4)  |  |               | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b110", the device transits to main CR clock mode after waiting for the main CR clock oscillation stabilization wait time. When the main CR clock oscillation is enabled by the setting of the main CR clock oscillation enable bit in the system clock control register 2 (SYCC2:MCRE), and the main CR clock oscillation stabilization bit in the system clock control register 2 (SYCC2:MCRDY) is "1", the device transits to main CR clock mode immediately after the clock mode select bits (SYCC:SCS[2:0]) are set to "0b110".  |
| (5)  |  | Main clock    | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b111", the device transits to main CR PLL clock mode after waiting for the main CR PLL clock oscillation stabilization wait time. When the main CR PLL clock oscillation is enabled by the setting of the main CR PLL clock enable bit in the PLL control register (PLLC:MPEN), and the main CR PLL clock oscillation stabilization bit in the PLL control register (PLLC:MPRDY) is "1", the device transits to main CR PLL clock mode immediately after the clock mode select bits (SYCC:SCS[2:0]) are set to "0b111".  |
| (6)  |  |               | Sub-CR clock   |
| (7)  | Main clock                             | Subclock      | Same as (3) and (4)  |
| (8)  |  |               |  |
| (9)  |  |               |  |
| (10) |  |               |  |
| (11) |  |               |  |
| (12) |  |               |  |

**Table 3.4-1 Clock Mode State Transition Table (2 / 2)**

|      | Current State | Next State                             | Description  |
|------|---------------|--|--|
| (13) | Sub-CR clock  | Main CR clock/<br>Main CR PLL<br>clock | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b110", the device transits to main CR clock mode after waiting for the main CR clock oscillation stabilization wait time.<br>When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b111", the device transits to main CR PLL clock mode after waiting for the main CR PLL clock oscillation stabilization wait time. |
| (14) |               | Main clock                             | When the clock mode select bits in the system clock control register (SYCC:SCS[2:0]) are set to "0b010", the device transits to main clock mode after waiting for the main clock oscillation stabilization wait time.  |
| (15) |               | Subclock                               | Same as (3) and (4)  |
| (16) |               |  |  |
| (17) | Subclock      | Main CR clock/<br>Main CR PLL<br>clock | Same as (13)   |
| (18) |               | Main clock                             | Same as (14)   |
| (19) |               | Sub-CR clock                           | Same as (1) and (2)  |
| (20) |               |  |  |

## 3.5 Operations in Low Power Consumption Mode (Standby Mode)

---

There are four standby modes: sleep mode, stop mode, time-base timer mode and watch mode.

---

### ■ Overview of Transiting to and Returning from Standby Mode

There are four standby modes: sleep mode, stop mode, time-base timer mode, and watch mode. The device transits to standby mode according to the settings in the standby control register (STBC).

The device is released from standby mode by an interrupt or a reset. Before transiting to normal operation, the device may wait for the oscillation stabilization wait time to elapse if necessary.

When the clock mode returns from standby mode due to a reset, the device returns to main CR clock mode. When the clock mode returns from standby mode due to an interrupt, the device returns to the previous clock mode before transiting to standby mode.

### ■ Pin States in Standby Mode

The pin state setting bit (STBC:SPL) of the standby control register can be used to keep the preceding state of an I/O port or a peripheral function pin before its transition to stop mode, time-base timer mode or watch mode, and to set an I/O port or a peripheral function pin to high impedance in stop mode, time-base timer mode or watch mode.

Refer to the device data sheet for the states of all pins in standby mode.

### **3.5.1 Notes on Using Standby Mode**

---

**Even if the standby control register (STBC) sets standby mode, transition to standby mode does not occur when an interrupt request has been generated from a peripheral function. When the device returns from standby mode to the normal operating state in response to an interrupt, the operation that follows varies depending on whether the interrupt request is accepted or not.**

---

■ **Insert at least three NOP instructions immediately after a standby mode setting instruction.**

The device requires four machine clock cycles before entering standby mode after it is set in the standby control register. During that period, the CPU executes the program. To avoid program execution during this transition to standby mode, insert at least three NOP instructions.

The device still runs normally even if instructions other than NOP instructions are inserted after the instruction that sets the device to transit to standby mode. On this occasion, the following two events may occur. Firstly, an instruction that should be executed after the standby mode is released may be executed before the device transits to standby mode. Secondly, the device may transit to standby mode while an instruction is being executed, and the execution of that same instruction is resumed after the device is released from standby mode (increasing the number of instruction execution cycles).

■ **Check that clock mode transition has been completed before setting the standby mode.**

Before setting the standby mode, ensure that clock-mode transition has been completed by comparing the values of the clock mode monitor bits (SYCC:SCM[2:0]) and clock mode select bits (SYCC:SCS[2:0]) in the system clock control register.

■ **An interrupt request may suppress the transition to standby mode.**

When the standby mode is set with an interrupt request whose interrupt level is higher than "0b11" having been issued, the device ignores the value written to the standby control register and continues executing instructions without transiting to the standby mode set. Even after the interrupt of that interrupt request is processed, the device does not transit to the standby mode set.

The same operations are executed when interrupts are disabled by the interrupt enable flag (CCR:I) and the interrupt level bits (CCR:IL[1:0]) of the condition code register of the CPU.

■ **The standby mode is also released when the CPU rejects interrupts.**

When an interrupt request whose interrupt level is higher than "0b11" is issued in standby mode, the device is released from standby mode, regardless of the settings of the interrupt enable flag (CCR:I) and the interrupt level bits (CCR:IL[1:0]) of the condition code register (CCR) of the CPU.

After being released from standby mode, the device processes interrupts if interrupts are to be accepted according to the settings of the condition code register (CCR) of the CPU. If interrupts are not to be accepted according to the settings of CCR, the device resumes instruction execution from the instruction following the one executed before the device transits to standby mode.



■ In every standby mode, the following two operating modes can be selected.

1. Deep standby mode (STBC2:DSTBYX = 0)

In standby mode, the power consumption in deep standby mode is lower than that in normal standby mode. However, since the device has to wait for the Flash recovery wait time to elapse before being woken up from deep standby mode by a reset or an interrupt source, it takes more time to wake up the device from deep standby mode than from normal standby mode.

2. Normal standby mode (STBC2:DSTBYX = 1)

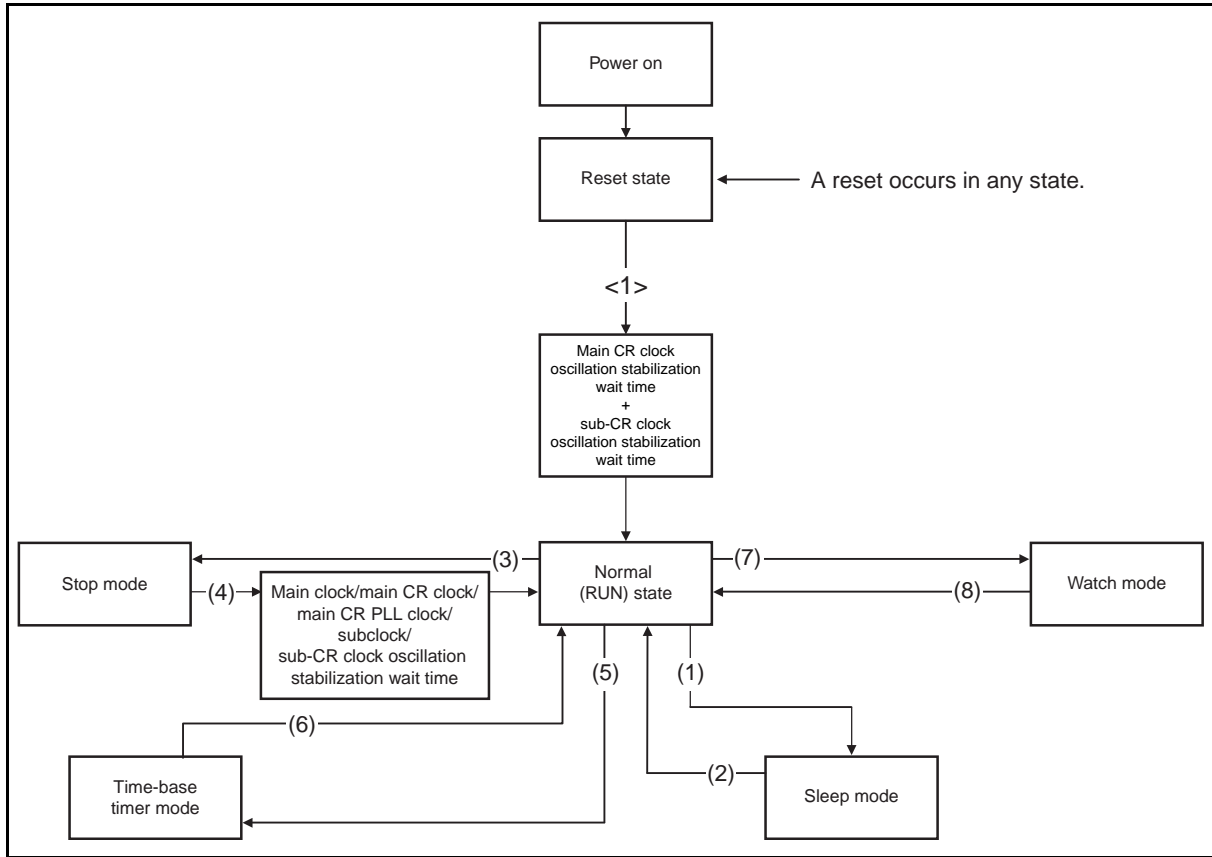
In standby mode, the power consumption in normal standby mode is higher than that in deep standby mode. However, since the device does not have to wait for the Flash recovery wait time to elapse before being woken up from normal standby mode by a reset or an interrupt source, it takes lesser time to wake up the device from deep standby mode than from normal standby mode.

For details of the Flash recovery wait time, see Table 3.5-2. For the difference between deep standby mode and normal standby mode in power consumption, refer to "■ ELECTRICAL CHARACTERISTICS" in the device data sheet.

■ **Standby Mode State Transition Diagram (with Deep Standby Mode Disabled)**

Figure 3.5-1 shows a standby mode state transition diagram (with deep standby mode disabled).

**Figure 3.5-1 Standby Mode State Transition Diagram (with Deep Standby Mode Disabled)**



**Table 3.5-1 Table of State Transition with Deep Standby Mode Disabled (Transition to and from Standby Mode) (1 / 2)**

|     | State transition                   | Description  |
|-----|------------------------------------|--|
| <1> | Normal operation after reset state | After a reset, the device transits to main CR clock mode.<br>If the reset that has occurred is a power-on reset, a watchdog reset, a software reset, or an external reset, the device always waits for the main CR clock oscillation stabilization wait time and the sub-CR clock oscillation stabilization wait time to elapse. |
| (1) | Sleep mode                         | The device transits to sleep mode when "1" is written to the sleep bit in the standby control register (STBC:SLP).   |
| (2) |                                    | The device returns to the RUN state in response to an interrupt from a peripheral function.  |
| (3) | Stop mode                          | The device transits to stop mode when "1" is written to the stop bit in the standby control register (STBC:STP).   |
| (4) |                                    | In response to an external interrupt, after waiting for the elapse of the oscillation stabilization wait time required according to the current clock mode, the device returns to the RUN state.   |
| (5) | Time-base timer mode               | The device transits to time-base timer mode when "1" is written to the watch bit in the standby control register (STBC:TMD) in main clock mode, main CR clock mode, or main CR PLL clock mode.   |
| (6) |                                    | The device returns to the RUN state in response to an interrupt from a peripheral function.  |

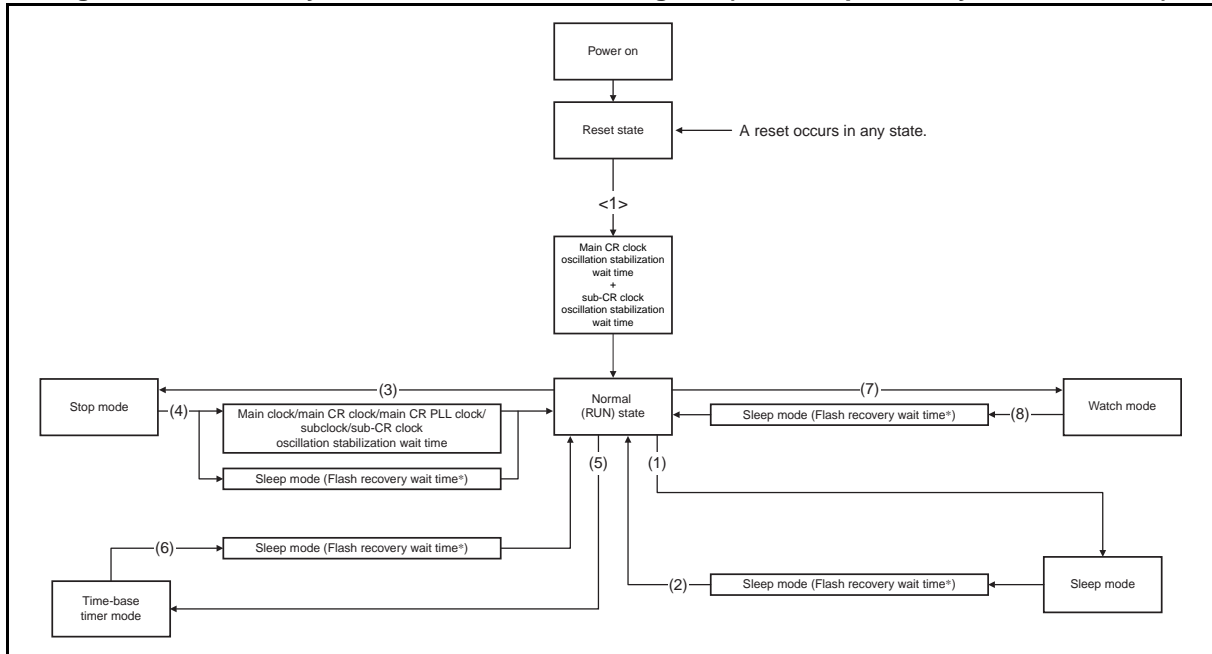
**Table 3.5-1 Table of State Transition with Deep Standby Mode Disabled (Transition to and from Standby Mode) (2 / 2)**

|     | State transition | Description  |
|-----|------------------|--|
| (7) | Watch mode       | The device transits to watch mode when "1" is written to the watch bit in the standby control register (STBC:TMD) in subclock mode or sub-CR clock mode. |
| (8) |                  | The device returns to the RUN state in response to an interrupt from a peripheral function.  |

■ **Standby Mode State Transition Diagram (with Deep Standby Mode Enabled)**

Figure 3.5-2 shows a standby mode state transition diagram (with deep standby mode enabled).

**Figure 3.5-2 Standby Mode State Transition Diagram (with Deep Standby Mode Enabled)**



\*: Flash memory recovery wait time (SCLK: source clock, MCLK: machine clock)

- In main clock mode, main CR clock mode, or main CR PLL clock mode  
 Maximum: 10 SCLK + 150 μs + 6 MCLK
- In subclock mode or sub-CR clock mode  
 Maximum: 2 SCLK + 150 μs + 6 MCLK

**Table 3.5-2 Table of State Transition with Deep Standby Mode Enabled (Transition to and from Standby Mode)**

|     | State transition                   | Description   |
|-----|------------------------------------|---|
| <1> | Normal operation after reset state | After a reset, the device transits to main CR clock mode.<br>If the reset that has occurred is a power-on reset, a watchdog reset, a software reset, or an external reset, the device always wait for the main CR clock oscillation stabilization wait time and the sub-CR clock oscillation stabilization wait time to elapse.   |
| (1) | Sleep mode                         | The device transits to sleep mode when "1" is written to the sleep bit in the standby control register (STBC:SLP).  |
| (2) |                                    | In response to an interrupt from a peripheral function, after the Flash recovery wait time elapses, the device returns to the RUN state.<br>During the Flash recovery wait time, the device transits to sleep mode. (The CPU stops its operation; the peripheral function resumes its operation.)<br>However, if a program is being executed on the RAM, no Flash recovery wait time occurs.  |
| (3) | Stop mode                          | The device transits to stop mode when "1" is written to the stop bit in the standby control register (STBC:STP).  |
| (4) |                                    | In response to an external interrupt, after the oscillation stabilization wait time required according to the current clock mode and the Flash recovery wait time elapse, the device returns to the RUN state.<br>When the oscillation stabilization wait time is shorter than the Flash recovery wait time, after the oscillation stabilization wait time elapses, the device transits to sleep mode and remains in sleep mode until the Flash recovery wait time elapses.<br>When the oscillation stabilization wait time is longer than the Flash recovery wait time, after the oscillation stabilization wait time elapses, the device returns to the RUN state.<br>However, if a program is being executed on the RAM, no Flash recovery wait time occurs. |
| (5) | Time-base timer mode               | The device transits to time-base timer mode when "1" is written to the watch bit in the standby control register (STBC:TMD) in main clock mode or main CR clock mode.   |
| (6) |                                    | In response to an interrupt from a peripheral function, after the Flash recovery wait time elapses, the device returns to the RUN state.<br>During the Flash recovery wait time, the device transits to sleep mode. (The CPU stops its operation; the peripheral function resumes its operation.)<br>However, if a program is being executed on the RAM, no Flash recovery wait time occurs.  |
| (7) | Watch mode                         | The device transits to watch mode when "1" is written to the watch bit in the standby control register (STBC:TMD) in subclock mode or sub-CR clock mode.  |
| (8) |                                    | In response to an interrupt from a peripheral function, after the Flash recovery wait time elapses, the device returns to the RUN state.<br>During the Flash recovery wait time, the device transits to sleep mode. (The CPU stops its operation; the peripheral function resumes its operation.)<br>However, if a program is being executed on the RAM, no Flash recovery wait time occurs.  |

## **3.5.2 Sleep Mode**

---

**In sleep mode, the operations of the CPU and watchdog timer are stopped.**

---

### **■ Operations in Sleep Mode**

In sleep mode, the CPU and the operating clock for the watchdog timer are stopped. The CPU retains the contents of registers and RAM existing at the point immediately before the device transits to sleep mode and stops; however, all peripheral functions except the watchdog timer continue their operations.

In the case of hardware watchdog timer, if it is enabled in standby mode by the non-volatile register function, in sleep mode, the sub-CR clock does not stop and the hardware watchdog timer continues its operation. For details, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

#### **● Transition to sleep mode**

Writing "1" to the sleep bit in the standby control register (STBC:SLP) causes the device to enter sleep mode.

#### **● Release from sleep mode**

A reset or an interrupt from a peripheral function releases the device from sleep mode.

With the deep standby mode control bit (STBC2:DSTBYX) set to "0", even after a reset occurs or an interrupt is generated by a peripheral function, the device continues operating in sleep mode until the Flash recovery wait time elapses.

However, if a program is being executed on the RAM, no Flash recovery wait time occurs.

### 3.5.3 Stop Mode

---

**In stop mode, the main clock, the main CR clock, the main CR PLL clock and the subclock are stopped.**

---

#### ■ Operations in Stop Mode

In stop mode, the main clock, the main CR clock, the main CR PLL clock and the subclock are stopped. In this mode, while retaining the contents of registers and RAM existing at the point immediately before the device transits to stop mode, the device stops all functions except external interrupt and low-voltage detection reset.

As for hardware watchdog timer, if it is enabled in standby mode by the non-volatile register function, in stop mode, the sub-CR clock does not stop and the hardware watchdog timer continues its operation. For details, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

#### ● Transition to stop mode

Writing "1" to the stop bit in the standby control register (STBC:STP) causes the device to transit to stop mode. At that point, if the pin state setting bit in the standby control register (STBC:SPL) is "0", the states of the external pins are kept; if the SPL bit is "1", the states of the external pins become high impedance (a pin is pulled up if the pull-up resistor connection for that pin is selected in the pull-up register).

#### ● Release from stop mode

The device is released from stop mode by a reset or an external interrupt. In any clock mode, if the hardware watchdog timer is enabled in standby mode by the non-volatile register function, the sub-CR clock does not stop, and the watchdog timer and the watch prescaler operate in stop mode. The device can also be released from stop mode by an interrupt from the watch prescaler. For details, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

If the touch sensor is enabled in standby mode, the sub-CR does not stop and the touch sensor operates in stop mode. The device can also be released from stop mode by an interrupt from the touch sensor.

With the deep standby mode control bit (STBC2:DSTBYX) set to "0", after a reset occurs or an interrupt is generated by a peripheral function, the device executes different operations, depending on the relation between the oscillation stabilization wait time and the Flash recovery wait time as explained below.

- When the oscillation stabilization wait time is shorter than the Flash recovery wait time  
After the oscillation stabilization wait time elapses, the device transits to sleep mode and remains in sleep mode until the Flash recovery wait time elapses.
  - When the oscillation stabilization wait time is longer than the Flash recovery wait time  
After the oscillation stabilization wait time elapses, the device returns to the RUN state.
- However, if a program is being executed on the RAM, no Flash recovery wait time occurs.

Note:

If the device is released from stop mode by an interrupt, a peripheral function having transitioned to stop mode during operation resumes operating from the point at which it transitioned to stop mode. Therefore, some settings of that peripheral function, such as the initial interval time of the interval timer, become undefined. Initialize that peripheral function if necessary after releasing the device from stop mode.

---



### 3.5.4 Time-base Timer Mode

---

**In time-base timer mode, only the main clock oscillator, the subclock oscillator, the time-base timer, and the watch prescaler operate. The CPU and the operating clock for peripheral functions are stopped in this mode.**

---

#### ■ Operations in Time-base Timer Mode

The time-base timer mode is a mode in which main clock supply is stopped except the clock supply to the time-base timer. In this mode, while retaining the contents of registers and RAM existing at the point immediately before the device transits to time-base timer mode, the device stops all functions except the time-base timer, external interrupt and low-voltage detection reset.

Subclock oscillation and sub-CR clock oscillation can be enabled or disabled by the subclock oscillation enable bit and the sub-CR clock oscillation enable bit in the system clock control register 2 (SYCC2:SOSCE, SCRE) respectively. If the subclock oscillates, the watch prescaler continues its operation.

In the case of hardware watchdog timer, if it is enabled in standby mode by the non-volatile register function, in time-base timer mode, the sub-CR clock does not stop and the hardware watchdog timer continues its operation. For details, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

#### ● Transition to time-base timer mode

If the clock mode monitor bits in the system clock control register (SYCC:SCM[2:0]) are "0b010", "0b110", or "0b111", writing "1" to the watch bit in the standby control register (STBC:TMD) causes the device to transit to time-base timer mode.

The device can transit to time-base timer mode only when the clock mode is main clock mode, main CR clock mode or main CR PLL clock mode.

After the device transits to time-base timer mode, if the pin state setting bit in the standby control register (STBC:SPL) is "0", the states of the external pins are kept; if the SPL bit is "1", the states of the external pins become high impedance (a pin is pulled up if the pull-up resistor connection for that pin is selected in the pull-up register).

#### ● Release from time-base timer mode

The device is released from time-base timer mode by a reset, a time-base timer interrupt, or an external interrupt.

Subclock oscillation and sub-CR clock oscillation can be enabled or disabled by setting the subclock oscillation enable bit (SOSCE) and the sub-CR clock oscillation enable bit (SCRE) in the system clock control register 2 (SYCC2). When the subclock oscillates, the device can be released from time-base timer mode by an interrupt from the watch prescaler.

If the touch sensor is enabled in standby mode, the sub-CR does not stop and the touch sensor operates in time-base timer mode. The device can also be released from time-base timer mode by an interrupt from the touch sensor.

With the deep standby mode control bit (STBC2:DSTBYX) set to "0", even after a reset occurs or an interrupt is generated by a peripheral function, the device continues operating in sleep mode until the Flash recovery wait time elapses.

However, if a program is being executed on the RAM, no Flash recovery wait time occurs.

---

Note:

If the device is released from time-base timer mode by an interrupt, a peripheral function having transited to time-base timer mode during operation resumes operating from the point at which it transited to time-base timer mode. Therefore, some settings of that peripheral function, such as the initial interval time of the interval timer, become undefined. Initialize that peripheral function if necessary after releasing the device from time-base timer mode.

---

### 3.5.5 Watch Mode

---

**In watch mode, only the subclock, the sub-CR clock and the watch prescaler operate. The CPU and the operating clock for peripheral functions are stopped in this mode.**

---

#### ■ Operations in Watch Mode

In watch mode, while retaining the contents of registers and RAM existing at the point immediately before the device transits to watch mode, the device stops all functions except the watch prescaler, external interrupt and low-voltage detection reset.

In the case of hardware watchdog timer, if it is enabled in standby mode by the non-volatile register function, in watch mode, the sub-CR clock does not stop and the hardware watchdog timer continues its operation. For details, see "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE".

#### ● Transition to watch mode

If the clock mode monitor bits in the system clock control register (SYCC:SCM[2:0]) are "0b000" or "0b100", writing "1" to the watch bit in the standby control register (STBC:TMD) causes the device to transit to watch mode.

The device can transit to watch mode only when the clock mode is subclock mode or sub-CR clock mode.

After the device transits to watch mode, if the pin state setting bit in the standby control register (STBC:SPL) is "0", the states of the external pins are kept; if the SPL bit is "1", the states of the external pins become high impedance (a pin is pulled up if the pull-up resistor connection for that pin is selected in the pull-up register).

#### ● Release from watch mode

The device is released from watch mode by a reset, a watch interrupt, or an external interrupt.

If the touch sensor is enabled in standby mode, the sub-CR does not stop and the touch sensor operates in watch mode. The device can also be released from watch mode by an interrupt from the touch sensor.

With the deep standby mode control bit (STBC2:DSTBYX) set to "0", even after a reset occurs or an interrupt is generated by a peripheral function, the device continues operating in sleep mode until the Flash recovery wait time elapses.

However, if a program is being executed on the RAM, no Flash recovery wait time occurs.

---

#### Note:

If the device is released from watch mode by an interrupt, a peripheral function having transited to watch mode during operation resumes operating from the point at which it transited to watch mode. Therefore, some settings of that peripheral function, such as the initial interval time of the interval timer, become undefined. Initialize that peripheral function if necessary after releasing the device from watch mode.

---

### 3.6 Clock Oscillator Circuit

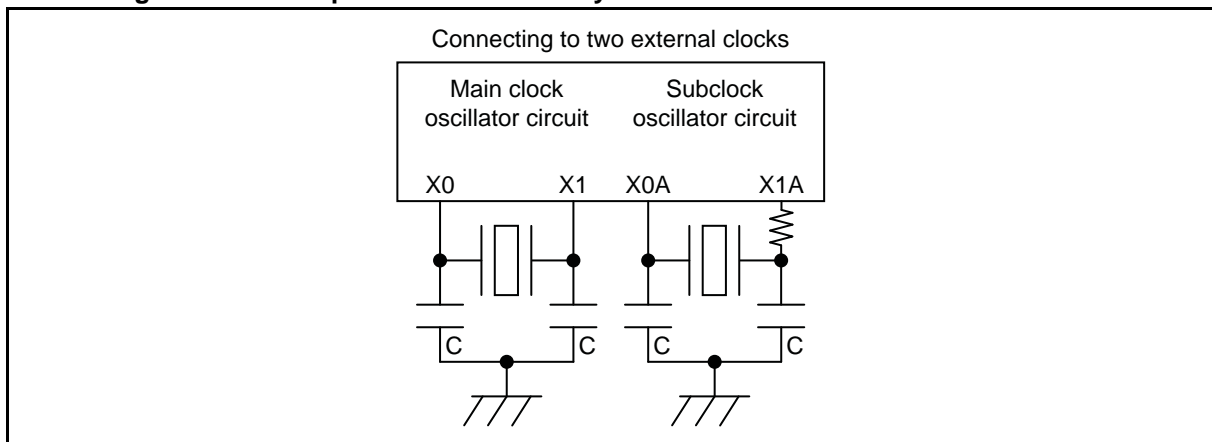
The clock oscillator circuit generates an internal clock with an oscillator connected to the clock oscillation pin or by inputting a clock signal to the clock oscillation pin.

#### ■ Clock Oscillator Circuit

- Using crystal oscillators and ceramic oscillators

Connect crystal oscillators or ceramic oscillators as shown in Figure 3.6-1.

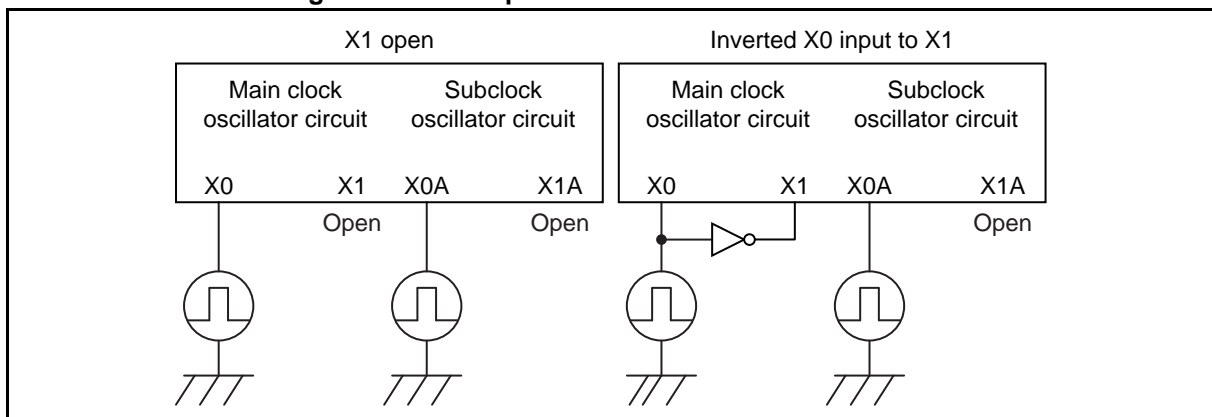
Figure 3.6-1 Sample Connection of Crystal Oscillators and Ceramic Oscillators



- Using external clock

As shown in Figure 3.6-2, connect the external clock to the X0 pin while leaving the X1 pin unconnected or supplying inverted clock of the X0 pin to the X1 pin (refer to the device data sheet). To supply clock signals to the subclock from an external clock, connect that external clock to the X0A pin while leaving the X1A pin unconnected.

Figure 3.6-2 Sample Connection of External Clocks



## MB95850K/860K/870K Series

### 3.7 Overview of Prescaler

---

The prescaler generates the count clock source to be supplied to various peripheral functions from the machine clock (MCLK) and the count clock output from the time-base timer.

---

#### ■ Prescaler

The prescaler generates the count clock source to be supplied to various peripheral functions from the machine clock (MCLK) with which the CPU operates and from the count clock ( $F_{CH}/2^7$ ,  $F_{CH}/2^8$ ,  $F_{CRH}/2^6$ ,  $F_{CRH}/2^7$ ,  $F_{MCRPLL}/2^6$ , or  $F_{MCRPLL}/2^7$ ) output from the time-base timer. The count clock source is a clock whose frequency is divided by the prescaler or a buffered clock. The peripheral functions listed below use the clock whose frequency is divided by the prescaler as the count clock source.

The prescaler has no control register and always operates with the machine clock (MCLK) and the count clock ( $F_{CH}/2^7$ ,  $F_{CH}/2^8$ ,  $F_{CRH}/2^6$ ,  $F_{CRH}/2^7$ ,  $F_{MCRPLL}/2^6$ , or  $F_{MCRPLL}/2^7$ ) of the time-base timer.

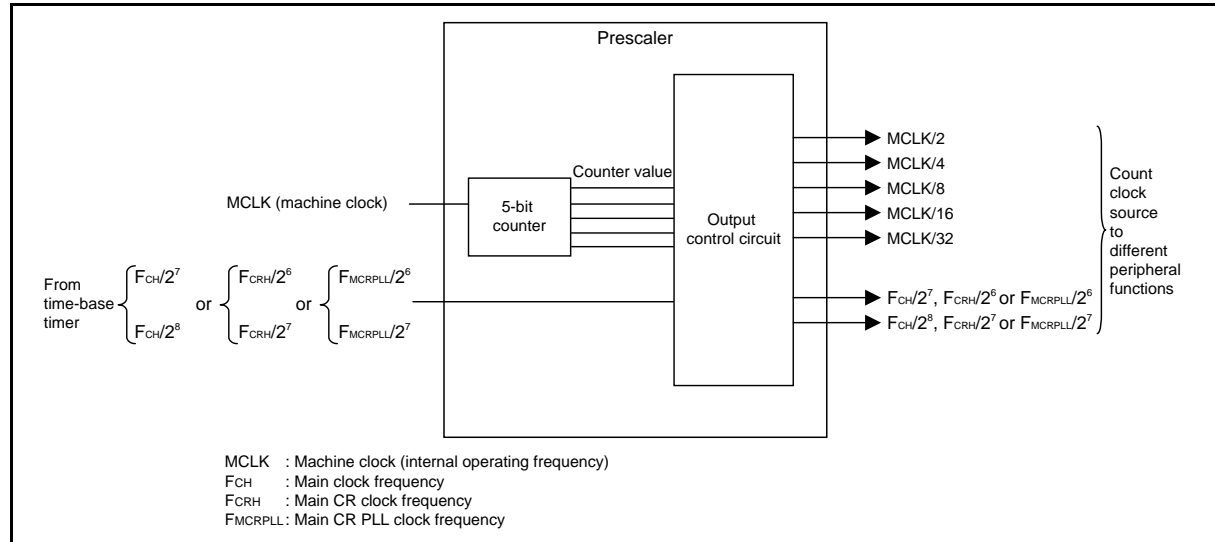
- 8/16-bit composite timer
- 8/10-bit A/D converter
- 8/16-bit PPG
- UART/SIO dedicated baud rate generator

## 3.8 Configuration of Prescaler

Figure 3.8-1 is the block diagram of the prescaler.

### ■ Block Diagram of Prescaler

Figure 3.8-1 Block Diagram of Prescaler



- 5-bit counter

This counter counts the machine clock (MCLK) and outputs the count value to the output control circuit.

- Output control circuit

Based on the 5-bit counter value, this circuit supplies clocks generated by dividing the machine clock (MCLK) by 2, 4, 8, 16, or 32 to individual peripheral functions. The circuit also buffers the clock from the time-base timer ( $F_{CH}/2^7$ ,  $F_{CH}/2^8$ ,  $F_{CRH}/2^6$ ,  $F_{CRH}/2^7$ ,  $F_{MCRPLL}/2^6$ , or  $F_{MCRPLL}/2^7$ ) and supplies it to peripheral functions.

### ■ Input Clock

The prescaler uses the machine clock, or the output clock of the time-base timer as the input clock.

### ■ Output Clock

The prescaler supplies clocks to the following peripheral functions:

- 8/16-bit composite timer
- 8/10-bit A/D converter
- 8/16-bit PPG
- UART/SIO dedicated baud rate generator

## MB95850K/860K/870K Series

### 3.9 Operation of Prescaler

The prescaler generates count clock sources to different peripheral functions.

#### ■ Operation of Prescaler

The prescaler generates count clock sources from a clock whose frequency is generated by dividing the machine clock (MCLK), or from buffered signals from the time-base timer ( $F_{CH}/2^7$ ,  $F_{CH}/2^8$ ,  $F_{CRH}/2^6$ ,  $F_{CRH}/2^7$ ,  $F_{MCRPLL}/2^6$ , or  $F_{MCRPLL}/2^7$ ), and then supplies them to different peripheral functions. The prescaler keeps operating while the machine clock and the clocks from the time-base timer are being supplied.

Table 3.9-1, Table 3.9-2 and Table 3.9-3 list the count clock sources generated by the prescaler.

**Table 3.9-1 Count Clock Sources Generated by Prescaler ( $F_{CH}$ )**

| Count clock source frequency | Frequency ( $F_{CH} = 20$ MHz, MCLK = 10 MHz) | Frequency ( $F_{CH} = 32$ MHz, MCLK = 16 MHz) | Frequency ( $F_{CH} = 32.5$ MHz, MCLK = 16.25 MHz) |
|------------------------------|---|---|--|
| MCLK/2                       | 5 MHz   | 8 MHz   | 8.125 MHz  |
| MCLK/4                       | 2.5 MHz                                       | 4 MHz   | 4.0625 MHz   |
| MCLK/8                       | 1.25 MHz                                      | 2 MHz   | 2.0313 MHz   |
| MCLK/16                      | 0.625 MHz                                     | 1 MHz   | 1.0156 MHz   |
| MCLK/32                      | 0.3125 MHz                                    | 0.5 MHz                                       | 0.5078 MHz   |
| $F_{CH}/2^7$                 | 156.25 kHz                                    | 250 kHz                                       | 253.9 kHz  |
| $F_{CH}/2^8$                 | 78.125 kHz                                    | 125 kHz                                       | 126.95 kHz   |

**Table 3.9-2 Count Clock Sources Generated by Prescaler ( $F_{CRH}$ )**

| Count clock source frequency | Frequency ( $F_{CRH} = 4$ MHz, MCLK = 4 MHz) |
|------------------------------|--|
| MCLK/2                       | 2 MHz  |
| MCLK/4                       | 1 MHz  |
| MCLK/8                       | 0.5 MHz                                      |
| MCLK/16                      | 0.25 MHz                                     |
| MCLK/32                      | 125 kHz                                      |
| $F_{CRH}/2^6$                | 62.5 kHz                                     |
| $F_{CRH}/2^7$                | 31.25 kHz                                    |

**Table 3.9-3 Count Clock Sources Generated by Prescaler ( $F_{MCRPLL}$ )**

| Count clock source frequency | Frequency<br>( $F_{MCRPLL} = 8$ MHz,<br>MCLK = 8 MHz) | Frequency<br>( $F_{MCRPLL} = 10$ MHz,<br>MCLK = 10 MHz) | Frequency<br>( $F_{MCRPLL} = 12$ MHz,<br>MCLK = 12 MHz) | Frequency<br>( $F_{MCRPLL} = 16$ MHz,<br>MCLK = 16 MHz) |
|------------------------------|---|---|---|---|
| MCLK/2                       | 4 MHz   | 5 MHz   | 6 MHz   | 8 MHz   |
| MCLK/4                       | 2 MHz   | 2.5 MHz   | 3 MHz   | 4 MHz   |
| MCLK/8                       | 1 MHz   | 1.25 MHz  | 1.5 MHz   | 2 MHz   |
| MCLK/16                      | 0.5 MHz   | 0.625 MHz   | 0.75 MHz  | 1 MHz   |
| MCLK/32                      | 0.25 MHz  | 0.3125 MHz  | 0.375 MHz   | 0.5 MHz   |
| $F_{MCRPLL}/2^6$             | 125 kHz   | 156.25 kHz  | 187.5 kHz   | 0.25 MHz  |
| $F_{MCRPLL}/2^7$             | 62.5 kHz  | 78.125 kHz  | 93.75 kHz   | 125 kHz   |



## MB95850K/860K/870K Series

### 3.10 Notes on Using Prescaler

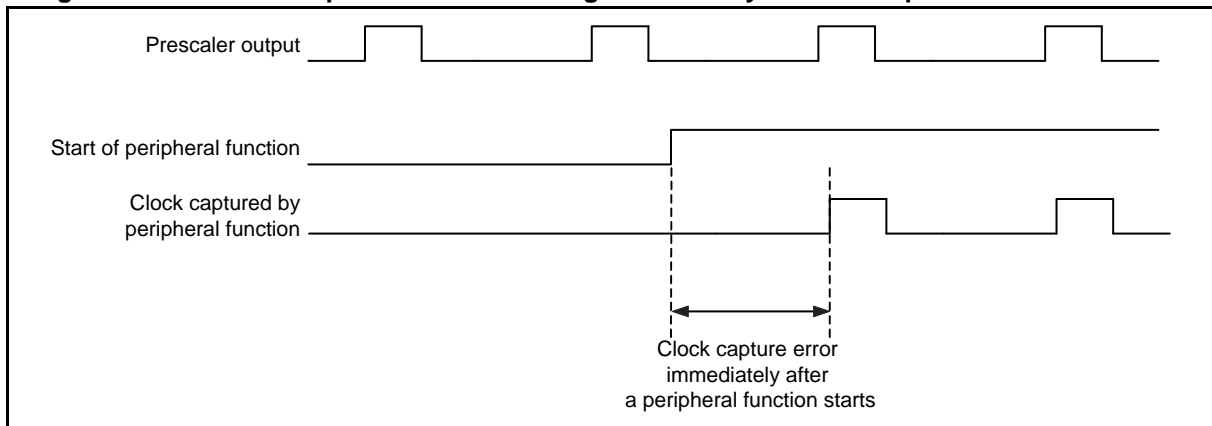
---

**This section provides notes on using the prescaler.**

---

The prescaler operates with the machine clock and the clock generated from the time-base timer, and keeps operating while those clocks are being supplied. Therefore, in the operation immediately after a peripheral function is started, an error of up to one cycle of the clock source captured by that peripheral function will occur, depending on the output value of the prescaler.

**Figure 3.10-1 Clock Capture Error Occurring Immediately after a Peripheral Function Starts**



The prescaler count value affects the following peripheral functions:

- 8/16-bit composite timer
- 8/10-bit A/D converter
- 8/16-bit PPG
- UART/SIO dedicated baud rate generator



# **CHAPTER 4**

---

# **RESET**

**This section describes the reset operation.**

- 4.1 Reset Operation
- 4.2 Register
- 4.3 Notes on Using Reset

## 4.1 Reset Operation

When a reset source occurs, the CPU immediately stops the process being executed and enters the reset release wait state. When the reset is released, the CPU reads mode data and the reset vector from the Flash memory (mode fetch). When the power is switched on or when the device is released from a reset in subclock mode, sub-CR clock mode, or stop mode, the CPU performs mode fetch after the oscillation stabilization wait time has elapsed.

### ■ Reset Sources

There are five reset sources for the reset.

Table 4.1-1 Reset Sources

| Reset source                | Reset condition   |
|-----------------------------|---|
| External reset              | "L" level is input to the external reset pin.   |
| Software reset              | "1" is written to the software reset bit in the standby control register (STBC:SRST). |
| Watchdog reset              | The watchdog timer overflows.   |
| Power-on reset              | The power is switched on.   |
| Low-voltage detection reset | The supply voltage falls below the detection voltage.                                 |

#### ● External reset

An external reset is generated if "L" level is input to the external reset pin ( $\overline{\text{RST}}$ ).

An external input reset signal is received asynchronously with the operating clock of the microcontroller via the internal noise filter and then generates an internal reset signal that is synchronized with the machine clock to initialize the internal circuit. Therefore, the operating clock of the microcontroller is necessary for initializing the internal circuit. In order to operate with the external clock, external clock signals must be input. However, the external pins (including I/O ports and peripheral functions) are reset asynchronously. In addition, there is a standard value of the pulse width for external reset input. If the value is below the standard value, a reset signal may not be accepted.

The standard value is shown in the device data sheet. Design an external reset circuit that satisfies the standard value.

#### ● Software reset

Writing "1" to the software reset bit of the standby control register (STBC:SRST) generates a software reset.

#### ● Watchdog reset

After the watchdog timer starts, a watchdog reset is generated if the watchdog timer is not cleared within a predetermined period of time.

#### ● Power-on reset

A power-on reset is generated when the power is switched on.

- Low-voltage detection reset

The low-voltage detection reset circuit generates a reset if the power supply voltage falls below a predetermined level.

The logical function of the low-voltage detection reset is equivalent to that of the power-on reset. All information relating to the power-on reset of this hardware manual also applies to the low-voltage detection reset.

For details of the low-voltage detection reset, see "CHAPTER 16 LOW-VOLTAGE DETECTION RESET CIRCUIT".

## ■ Reset Time

The reset time varies according to the reset source.

- In the case of software reset, watchdog reset and external reset:

The reset time is affected by the number of machine cycles selected before a reset, the RAM access protection function inhibiting resets during the RAM access, and the sub-CR clock oscillation stabilization wait time. The effective time of the RAM access protection function lengthens according to the number of machine clock cycles selected before a reset.

When a reset occurs with the sub-CR clock oscillation stabilization bit in the system clock control register 2 (SYCC2:SCRDY) set to "1", the device is released from the reset state after the main CR clock oscillation stabilization wait time elapses.

When a reset occurs with the sub-CR clock oscillation stabilization bit in the system clock control register 2 (SYCC2:SCRDY) set to "0", the device is released from the reset state after both sub-CR clock oscillation stabilization wait time and main CR clock oscillation stabilization wait time elapse.

- In the case of power-on reset and low-voltage detection reset:

The device is released from the reset state after both sub-CR clock oscillation stabilization wait time and main CR clock oscillation stabilization wait time elapse.

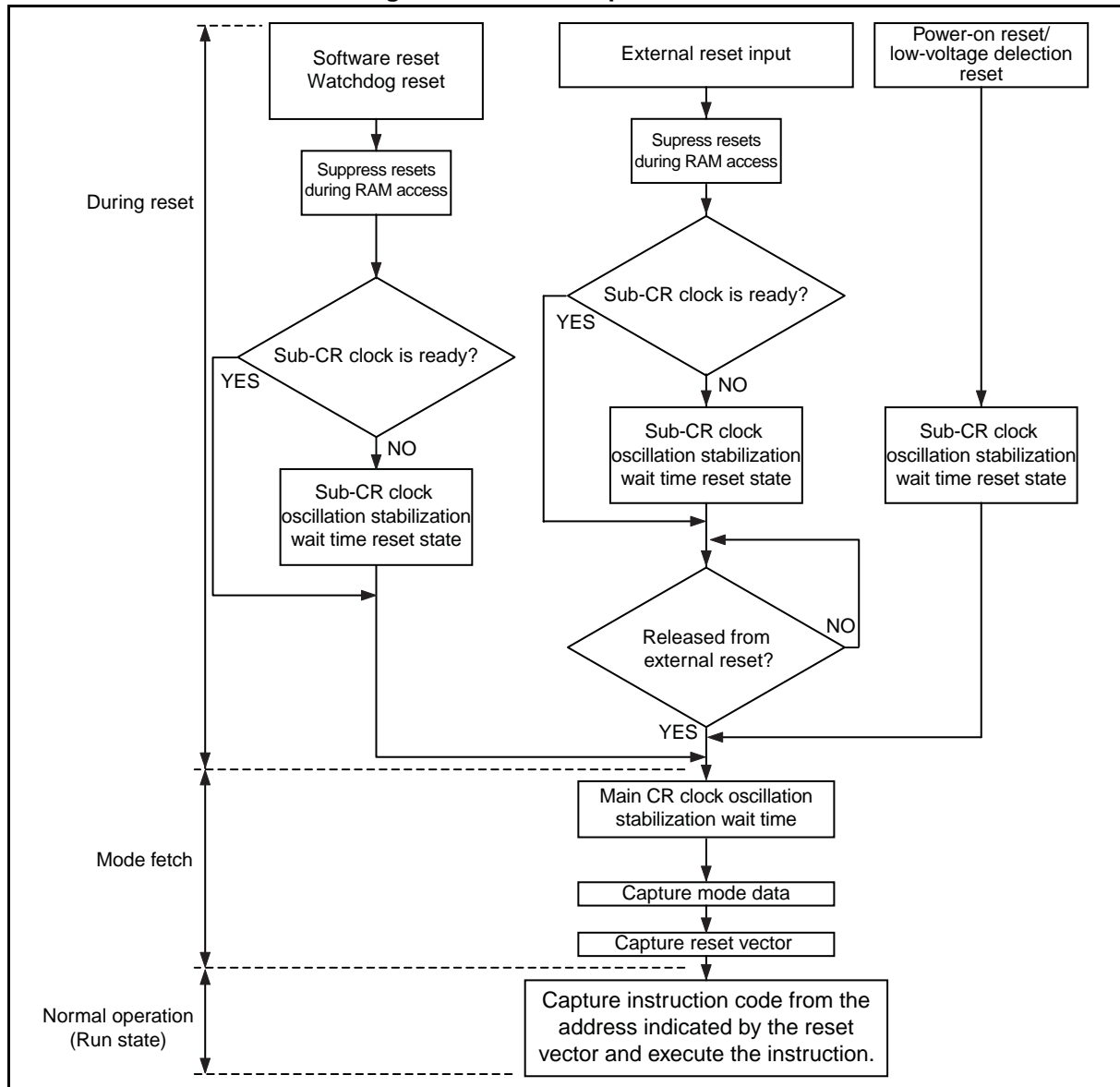
## ■ Reset Output

When the reset input function is effective and the reset output function is effective, the  $\overline{\text{RST}}$  pin outputs "L" level while resetting it. However, the function to output "L" level is not provided for external reset in the reset pin.

For details of the reset input function and the reset output function setting, see "CHAPTER 28 SYSTEM CONFIGURATION CONTROLLER".

■ Overview of Reset Operation

Figure 4.1-1 Reset Operation Flow



In any reset, the CPU performs mode fetch after the main CR clock oscillation stabilization wait time elapses.

■ Effect of Reset on RAM Contents

When a reset occurs, the CPU halts the operation of the command currently being executed, and enters the reset state. However, during RAM access execution, in order to protect the RAM access, an internal reset signal synchronized with the machine clock is generated after an RAM access ends. This function prevents a word-data write operation from being interrupted by a reset while data of two bytes is being written.

## ■ Pin State During a Reset

When a reset occurs, an I/O port or a peripheral function pin remains high impedance until the setting of that I/O port or that peripheral function pin by software is executed after the reset is released.

---

### Note:

Connect a pull-up resistor to a pin that becomes high impedance during a reset to prevent the device from malfunctioning.

---

For details of the states of all pins during a reset, refer to the device data sheet.

## 4.2 Register

---

**This section provides details of the register for reset.**

---

**Table 4.2-1 List of Register for Reset**

| Register abbreviation | Register name         | Reference |
|-----------------------|-----------------------|-----------|
| RSRR                  | Reset source register | 4.2.1     |



# MB95850K/860K/870K Series

## 4.2.1 Reset Source Register (RSRR)

The reset source register indicates the source of a reset generated.

### Register Configuration

| bit           | 7 | 6 | 5 | 4    | 3    | 2    | 1   | 0   |
|---------------|---|---|---|------|------|------|-----|-----|
| Field         | — | — | — | EXTS | WDTR | PONR | HWR | SWR |
| Attribute     | — | — | — | R/W  | R/W  | R/W  | R/W | R/W |
| Initial value | 0 | 0 | 0 | X    | X    | X    | X   | X   |

### Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4] EXTS: External reset flag bit

When this bit is set to "1", that indicates an external reset has occurred.

When any other reset occurs, this bit retains the value that has existed before such reset occurs.

A read access or a write access (writing "0" or "1") to this bit sets it to "0".

| bit4                    | Details  |
|-------------------------|--|
| <b>Read access</b>      | The read value is always "0".                      |
| <b>Being set to "1"</b> | Indicates that the an external reset has occurred. |
| <b>Write access</b>     | Sets this bit to "0".                              |

#### [bit3] WDTR: Watchdog reset flag bit

When this bit is set to "1", that indicates a watchdog reset has occurred.

When any other reset occurs, this bit retains the value that has existed before such reset occurs.

A read access or a write access (writing "0" or "1") to this bit sets it to "0".

| bit3                    | Details   |
|-------------------------|---|
| <b>Read access</b>      | The read value is always "0".                     |
| <b>Being set to "1"</b> | Indicates that the a watchdog reset has occurred. |
| <b>Write access</b>     | Sets this bit to "0".                             |

#### [bit2] PONR: Power-on reset flag bit

When this bit is set to "1", that indicates a power-on reset or a low-voltage detection reset has occurred.

When any other reset occurs, this bit retains the value that has existed before such reset occurs.

A read access or a write access (writing "0" or "1") to this bit sets it to "0".

| bit2                    | Details  |
|-------------------------|--|
| <b>Read access</b>      | The read value is always "0".  |
| <b>Being set to "1"</b> | Indicates that the a power-on reset or a low-voltage detection reset has occurred. |
| <b>Write access</b>     | Sets this bit to "0".  |

### [bit1] HWR: Hardware reset flag bit

When this bit is set to "1", that indicates a hardware reset (power-on reset, low-voltage detection reset, external reset or watchdog reset) has occurred. Therefore, when any of bit2 to bit4 is set to "1", this bit is set to "1" as well.

When a software reset occurs, the bit retains the value that has existed before the software reset occurs.

A read access or a write access (writing "0" or "1") to this bit sets it to "0".

| bit1                    | Details   |
|-------------------------|---|
| <b>Read access</b>      | The read value is always "0".                     |
| <b>Being set to "1"</b> | Indicates that the a hardware reset has occurred. |
| <b>Write access</b>     | Sets this bit to "0".                             |

### [bit0] SWR: Software reset flag bit

When this bit is set to "1", that indicates a software reset has occurred.

When a hardware reset occurs, the bit retains the value that has existed before the hardware reset occurs.

A read access or a write access (writing "0" or "1") to this bit or a power-on reset sets it to "0".

| bit0                    | Details   |
|-------------------------|---|
| <b>Read access</b>      | The read value is always "0".                     |
| <b>Being set to "1"</b> | Indicates that the a software reset has occurred. |
| <b>Write access</b>     | Sets this bit to "0".                             |

---

Note:

Since reading the reset source register clears its contents, save the contents of this register to the RAM before using those contents for calculation.

---

## MB95850K/860K/870K Series

### ■ State of Reset Source Register (RSRR)

Table 4.2-2 State of Reset Source Register

| Reset source                | EXTS | WDTR | PONR | HWR | SWR |
|-----------------------------|------|------|------|-----|-----|
| Power-on reset              | ×    | ×    | 1    | 1   | 0   |
| Low-voltage detection reset | ×    | ×    | 1    | 1   | 0   |
| Software reset              | △    | △    | △    | △   | 1   |
| Watchdog reset              | △    | 1    | △    | 1   | △   |
| External reset              | 1    | △    | △    | 1   | △   |

1: Flag set

△: Previous state kept

×: Indeterminate

EXTS: When this bit is set to "1", that indicates an external reset has occurred.

WDTR: When this bit is set to "1", that indicates a watchdog reset has occurred.

PONR: When this bit is set to "1", that indicates a power-on reset or low-voltage detection reset has occurred.

HWR: When this bit is set to "1", that indicates one of the following reset has occurred: an external reset, a watchdog reset, a power-on reset or a low-voltage detection reset.

SWR: When this bit is set to "1", that indicates that a software reset has occurred.

## **4.3 Notes on Using Reset**

---

**This section provides notes on using the reset.**

---

### **■ Notes on Using Reset**

- Initialization of registers and bits by reset source

Some registers and bits are initialized only by a certain reset source.

- The type of reset source determines which bit in the reset source register (RSRR) is to be initialized.
- The oscillation stabilization wait time setting register (WATR) of the clock controller can only be initialized by a power-on reset.

# **CHAPTER 5**

---

# **INTERRUPTS**

**This chapter describes the interrupts.**

## 5.1 Interrupts

## **5.1 Interrupts**

---

**This section describes the interrupts.**

---

### **■ Overview of Interrupts**

The New 8FX family has 24 interrupt request inputs for respective peripheral functions, for each of which an interrupt level can be set independently to each other.

When a peripheral function generates an interrupt request, the interrupt request is output to the interrupt controller. The interrupt controller checks the interrupt level of that interrupt request and then notifies the CPU of the generation of the interrupt. The CPU processes that interrupt according to the interrupt acceptance status. The device wakes up from standby mode by an interrupt request generated in standby mode and resumes executing instructions.

### **■ Interrupt Requests from Peripheral Functions**

When the CPU receives an interrupt request, it branches to the interrupt service routine with the interrupt vector table address corresponding to the interrupt request as the address of the branch destination.

The priority of each interrupt request in interrupt processing can be set to one of the four levels by the interrupt level setting registers (ILR0 to ILR5).

While an interrupt is being processed in the interrupt service routine, if another interrupt whose interrupt request is of the same level or below the one of the interrupt being processed is generated, it is processed after the current interrupt service routine is completed. In addition, if multiple interrupt requests that are set to the same interrupt level are made, IRQ00 is at the top of the priority order.

For interrupt sources, refer to "■ INTERRUPT SOURCE TABLE" in the device data sheet.

## MB95850K/860K/870K Series

### 5.1.1 Interrupt Level Setting Registers (ILR0 to ILR5)

The interrupt level setting registers (ILR0 to ILR5) contain 24 pairs of 2-bit data assigned to the interrupt requests of different peripheral functions. Each pair of bits (interrupt level setting bits) is used to set the interrupt level of an interrupt request.

#### ■ Register Configuration

ILR0

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L03[1:0] |     | L02[1:0] |     | L01[1:0] |     | L00[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

ILR1

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L07[1:0] |     | L06[1:0] |     | L05[1:0] |     | L04[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

ILR2

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L11[1:0] |     | L10[1:0] |     | L09[1:0] |     | L08[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

ILR3

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L15[1:0] |     | L14[1:0] |     | L13[1:0] |     | L12[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

ILR4

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L19[1:0] |     | L18[1:0] |     | L17[1:0] |     | L16[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

ILR5

|               |          |     |          |     |          |     |          |     |
|---------------|----------|-----|----------|-----|----------|-----|----------|-----|
| bit           | 7        | 6   | 5        | 4   | 3        | 2   | 1        | 0   |
| Field         | L23[1:0] |     | L22[1:0] |     | L21[1:0] |     | L20[1:0] |     |
| Attribute     | R/W      | R/W | R/W      | R/W | R/W      | R/W | R/W      | R/W |
| Initial value | 1        | 1   | 1        | 1   | 1        | 1   | 1        | 1   |

The interrupt level setting registers assign a pair of bits to every interrupt request. The values of interrupt level setting bits in these registers represent the priority of an interrupt request (interrupt level: 0 to 3) in interrupt processing.

The interrupt level setting bits are compared with the interrupt level bits in the condition code register (CCR:IL[1:0]).

If the interrupt level of an interrupt request is 3, the CPU ignores that interrupt request.

Table 5.1-1 shows the relationships between interrupt level setting bits and interrupt levels.

**Table 5.1-1 Relationships Between Interrupt Level Setting Bits and Interrupt Levels**

| LXX[1:0] | Interrupt level | Priority              |
|----------|-----------------|-----------------------|
| 00       | 0               | Highest               |
| 01       | 1               | ↕                     |
| 10       | 2               |                       |
| 11       | 3               | Lowest (No interrupt) |

XX:00 to 23 Number of an interrupt request

While the main program is being executed, the interrupt level bits in the condition code register (CCR:IL[1:0]) are "0b11".



# MB95850K/860K/870K Series

## 5.1.2 Interrupt Processing

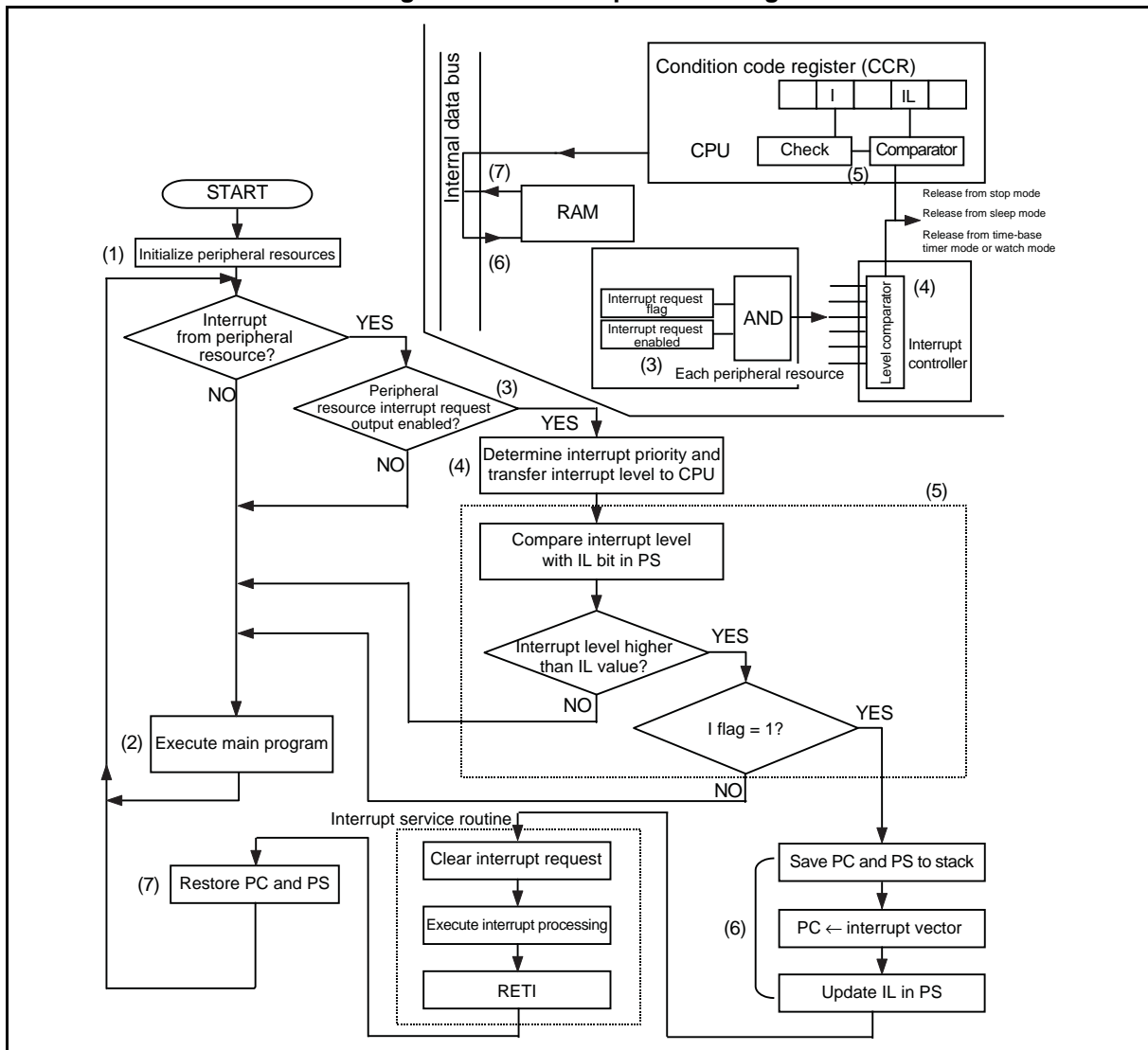
When an interrupt request is made by a peripheral function, the interrupt controller notifies the CPU of the interrupt level of that interrupt request. When the CPU is ready to accept interrupts, it halts the program it is executing and executes an interrupt service routine.

### ■ Interrupt Processing

The procedure for processing an interrupt is as follows: the generation of an interrupt source in a peripheral function, the execution of the main program, the setting of the interrupt request flag bit, the checking of the interrupt request enable bit, the determination of the interrupt level (ILR0 to ILR5 and CCR:IL[1:0]), the checking for interrupt requests of the same interrupt level made simultaneously, and the checking of the interrupt enable flag (CCR:I).

Figure 5.1-1 shows the interrupt processing.

Figure 5.1-1 Interrupt Processing



- (1) All interrupt requests are disabled immediately after a reset. In the peripheral function initialization program, initialize those peripheral functions that generate interrupts and set their interrupt levels in their respective interrupt level setting registers (ILR0 to ILR5) before starting operating such peripheral functions. The interrupt level can be set to 0, 1, 2, or 3. Level 0 is given the highest priority, and level 1 the second highest. Assigning level 3 to a peripheral function disables interrupts from that peripheral function.
- (2) Execute the main program (or the interrupt service routine in the case of nested interrupts).
- (3) When an interrupt source is generated in a peripheral function, the interrupt request flag bit for that peripheral function is set to "1". Provided that the interrupt request enable bit for that peripheral function has been set to the value that enables interrupts, an interrupt request of that peripheral function is output to the interrupt controller.
- (4) The interrupt controller keeps monitoring interrupt requests from individual peripheral functions and notifies the CPU of the interrupt level having priority over the others among interrupt levels already made. If there are interrupt requests having the same interrupt level, their positions in the priority order are also compared in the interrupt controller.
- (5) If the interrupt level received has priority over (smaller interrupt level number) the level set in the interrupt level bits in the condition code register (CCR:IL[1:0]), the CPU checks the content of the interrupt enable flag (CCR:I), and accepts the interrupt provided that interrupts have been enabled (CCR:I = 1).
- (6) The CPU saves the contents of the program counter (PC) and the program status (PS) to the stack, captures the start address of the interrupt service routine from the corresponding interrupt vector table address, modifies the values of the interrupt level bits in the condition code register (CCR:IL[1:0]) to the values of the interrupt level received, then starts executing the interrupt service routine.
- (7) Finally, the CPU uses the RETI instruction to restore the values of the program counter (PC) and the program status (PS) from the stack and resumes executing the instruction following the one executed just before the interrupt.

---

**Note:**

The interrupt request flag bit for a peripheral function is not automatically cleared to "0" after an interrupt request is accepted. Therefore, clear such bit to "0" by using a program (writing "0" to the interrupt request flag bit) in the interrupt service routine.

---

The low power consumption mode (standby mode) is released by an interrupt. For details, see "3.5 Operations in Low Power Consumption Mode (Standby Mode)".

## MB95850K/860K/870K Series

### 5.1.3 Nested Interrupts

Different interrupt levels can be assigned to multiple interrupt requests from peripheral functions in the interrupt level setting registers (ILR0 to ILR5) to process nested interrupts.

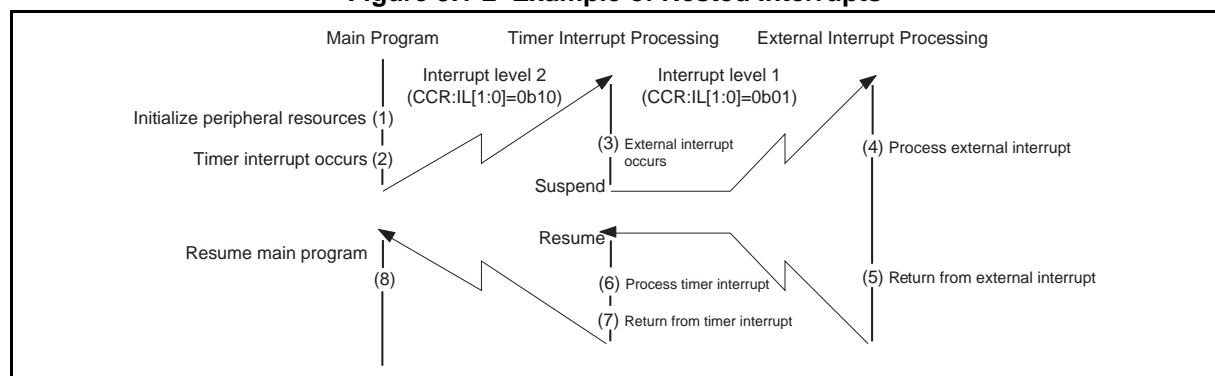
#### ■ Nested Interrupts

During the execution of an interrupt service routine, if another interrupt request whose interrupt level has priority over the interrupt level of the interrupt being processed is made, the CPU suspends the current interrupt processing and accepts the interrupt request given priority. The interrupt level of an interrupt request can be set to 0 to 3. If it is set to 3, the CPU does not accept that interrupt request.

#### [Example: Nested interrupts]

In the following example of nested interrupts, assuming that the external interrupt is to be given priority over the timer interrupt, the interrupt level of the timer interrupt is set to 2 and that of the external interrupt to 1. If the external interrupt is generated while the timer interrupt is being processed, they are processed as shown in Figure 5.1-2.

Figure 5.1-2 Example of Nested Interrupts



- While the timer interrupt is being processed, the interrupt level bits in the condition code register (CCR:IL[1:0]) hold the same value as that of the interrupt level setting registers (ILR0 to ILR5) corresponding to the timer interrupt (level 2 in the above example). If an interrupt request whose interrupt level has priority over the interrupt level of the timer interrupt (level 1 in the above example) is made, that interrupt is processed first.
- To temporarily disable nested interrupts processing while the timer interrupt is being processed, disable interrupts by setting the interrupt enable flag in the condition code register (CCR:I) to "0", or set the interrupt level bits (CCR:IL[1:0]) to "0b00".
- After the interrupt processing is completed, if the interrupt return instruction (RETI) is executed, the value of the program counter (PC) and that of the program status (PS) are restored, and the CPU resumes executing the program interrupted. In addition, the values of the condition code register (CCR) return to the ones existing before the interrupt due to the restoration of the value of the program status (PS).

## 5.1.4 Interrupt Processing Time

Before the CPU enters the interrupt service routine after an interrupt request is made, it needs to wait for the interrupt processing time, which consists of the time between the occurrence of an interrupt request and the end of the execution of the instruction being executed, and the interrupt handling time (the time required to initiate interrupt processing) to elapse. The maximum interrupt processing time is 26 machine clock cycles.

### ■ Interrupt Processing Time

Before executing the interrupt service routine after an interrupt request is made, the CPU needs to wait for the interrupt request sampling wait time and the interrupt handling time to elapse.

#### ● Interrupt request sampling wait time

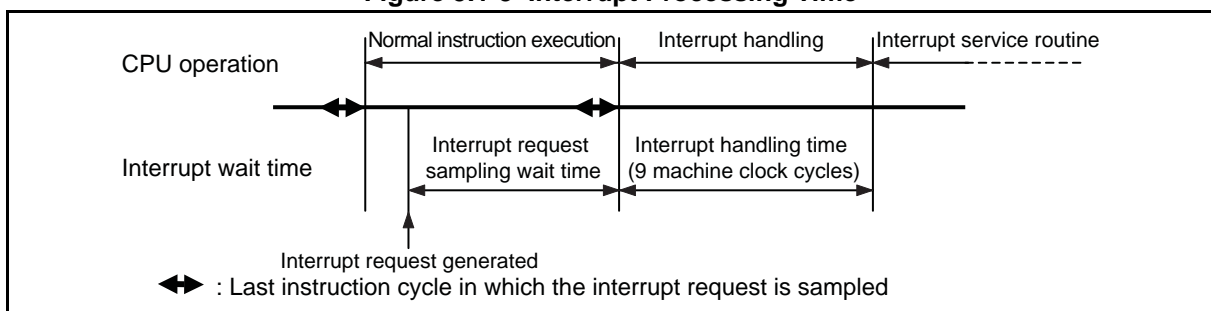
The CPU decides whether an interrupt request has occurred by sampling the interrupt request during the last cycle of each instruction. Therefore, the CPU cannot recognize interrupt requests while executing an instruction. This sampling wait time reaches its maximum when an interrupt request occurs immediately after the CPU starts executing the DIVU instruction, whose execution cycle is the longest (17 machine clock cycles).

#### ● Interrupt handling time

After accepting an interrupt, the CPU requires nine machine clock cycles to perform the following interrupt processing setup:

- Saves the value of the program counter (PC) and that of the program status (PS) to the stack.
- Sets the PC to the start address (interrupt vector) of interrupt service routine.
- Updates the interrupt level bits (CCR:IL[1:0]) in the program status (PS).

Figure 5.1-3 Interrupt Processing Time



When an interrupt request occurs immediately after the CPU starts executing the DIVU instruction, whose execution cycle is the longest (17 machine clock cycles), the interrupt processing time spans 26 machine clock cycles.

The span of a machine clock cycle varies depending on the clock mode and main clock speed change (gear function). For details, see "CHAPTER 3 CLOCK CONTROLLER".

## MB95850K/860K/870K Series

### 5.1.5 Stack Operation During Interrupt Processing

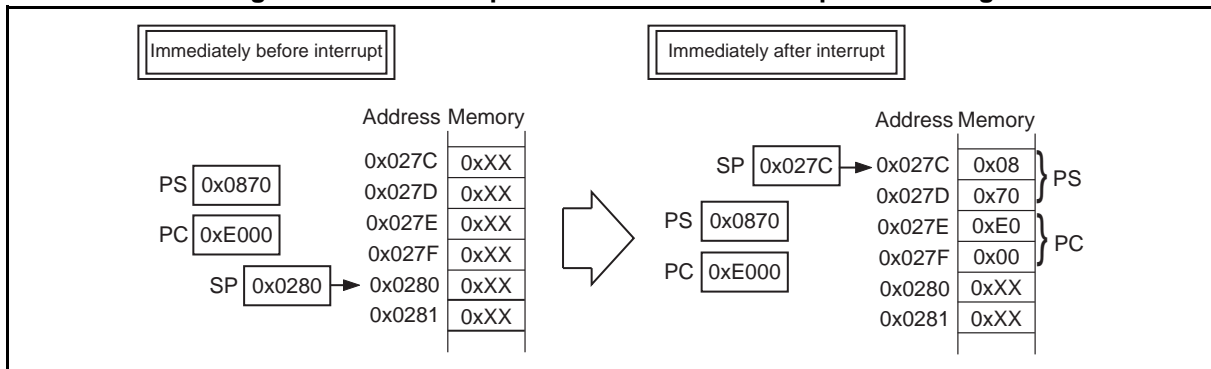
This section describes how the contents of a register are saved and restored during interrupt processing.

#### ■ Stack Operation at the Start of Interrupt Processing

Once the CPU accepts an interrupt, it automatically saves the current value of the program counter (PC) and that of the program status (PS) values to the stack.

Figure 5.1-4 shows the stack operation at the start of interrupt processing.

Figure 5.1-4 Stack Operation at Start of Interrupt Processing



#### ■ Stack Operation after Returning from Interrupt

When the CPU executes the interrupt return instruction (RETI) at the end of interrupt processing, it restores from the stack the value of the program status (PS) first and that of the program counter (PC), which is opposite to the sequence of saving the two values to the stack. After the restoration, both PS and PC return to their states before the start of interrupt processing.

**Note:**

Since the value of the accumulator (A) and that of the temporary accumulator (T) are not automatically saved to the stack, use the PUSHW and POPW instructions to save and restore the values of A and T.

## 5.1.6 Interrupt Processing Stack Area

The stack area in RAM is used for interrupt processing. The stack pointer (SP) contains the start address of the stack area.

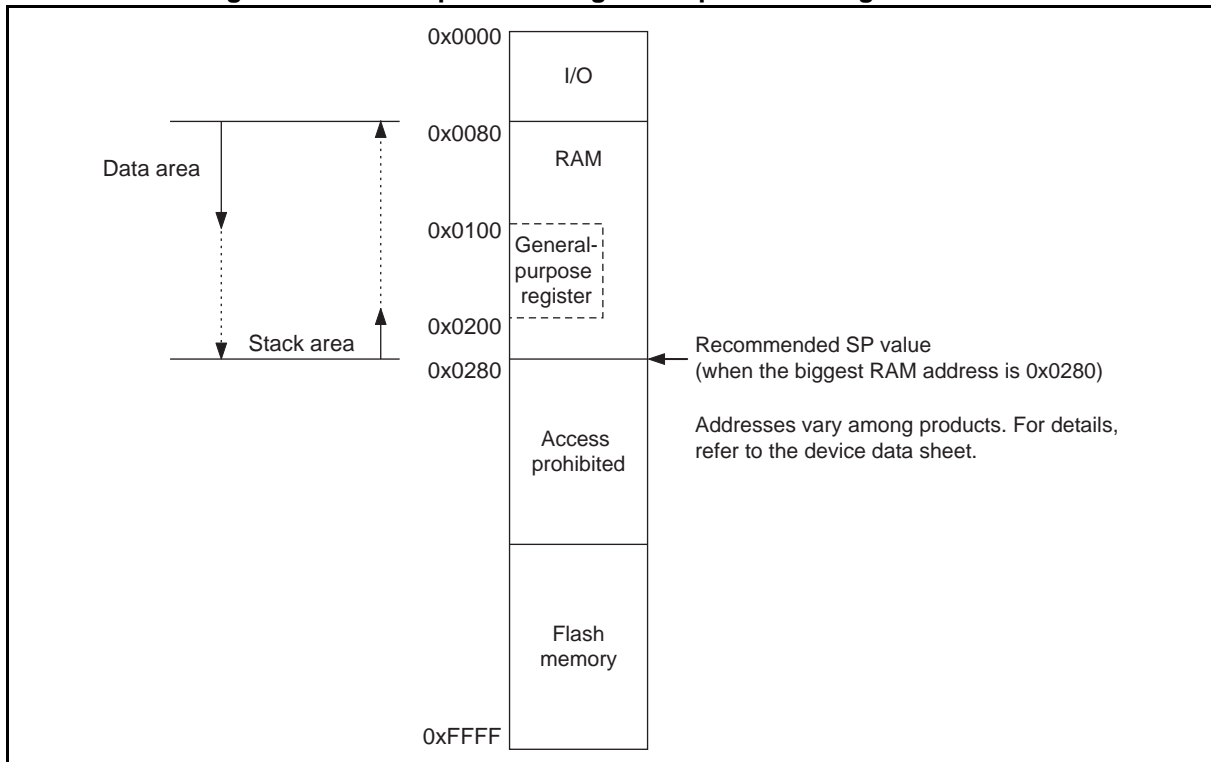
### ■ Interrupt Processing Stack Area

The stack area is also used for saving and restoring the program counter (PC) when the subroutine call instruction (CALL) or the vector call instruction (CALLV) is executed, and for saving temporarily and restoring register contents by the PUSHW and POPW instructions.

- The stack area is secured on the RAM together with the data area.
- Initialize the stack pointer (SP) so that it indicates the biggest RAM address and make the data area start from the smallest RAM address.

Figure 5.1-5 shows an example of setting the interrupt processing stack area.

Figure 5.1-5 Example of Setting Interrupt Processing Stack Area



Note:

The stack area is utilized by interrupts, sub-routine calls, the PUSHW instruction, etc. in descending order of addresses. It is released by return instructions (RETI, RET), the POPW instruction, etc. in ascending order of addresses. If the address value of the stack area used decreases due to nested interrupts or subroutine calls, do not let the stack area overlap the data area and the general-purpose register area, both of which retain other data.

# **CHAPTER 6**

---

# ***I/O PORT***

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

6.1 Overview

6.2 Configuration and Operations of I/O Port

## 6.1 Overview

---

**The I/O port is used to control general-purpose I/O pins.**

---

### ■ Overview of I/O Port

The I/O port has functions to output data from the CPU and capture input signals into the CPU with the port data register (PDR). The I/O direction of an individual I/O pin can be set as desired by using the corresponding to that I/O pin in the port direction register (DDR).

The number of I/O ports varies among products. For the exact number of I/O ports on a product, refer to the device data sheet.

In this chapter, "x" represents the port number in a register name. For details of register names and their respective abbreviations of a product, refer to the device data sheet.

Table 6.1-1 lists the registers for each port.

**Table 6.1-1 List of Port Registers**

| Register name                       | Register abbreviation |
|-------------------------------------|-----------------------|
| Port x data register                | PDRx                  |
| Port x direction register           | DDRx                  |
| Port x pull-up register             | PULx                  |
| A/D input disable register (lower)* | AIDRL                 |
| Touch input disable register 0*     | TIDR0                 |
| Touch input disable register 1*     | TIDR1                 |

\*: Refer to "■ I/O MAP" in the device data sheet for the availability of the A/D input disable register (lower), touch input disable register 0 and touch input disable register 1.



## 6.2 Configuration and Operations of I/O Port

---

This section focuses on its configuration and operations as a general-purpose I/O port.

For details of peripheral functions, see their respective chapters.

---

### ■ Configuration of I/O Port

An I/O port is made up of the following elements.

- General-purpose I/O pins/peripheral function I/O pins
- Port x data register (PDRx)
- Port x direction register (DDRx)
- Port x pull-up register (PULx)
- A/D input disable register (lower) (AIDRL)
- Touch input disable register 0 (TIDR0)
- Touch input disable register 1 (TIDR1)

## ■ Operations of I/O Port

### ● Operation as an output port

- A pin becomes an output port if the bit in the DDRx register corresponding to that pin is set to "1".
- For a pin shared with other peripheral functions, disable the output of such peripheral functions.
- When a pin is used as an output port, it outputs the value of the PDRx register to external pins.
- If data is written to the PDRx register, the value is stored in the output latch and is output to the pin set as an output port as it is.
- Reading the PDRx register returns the PDRx register value.

### ● Operation as an input port

- A pin becomes an input port if the bit in the DDRx register corresponding to that pin is set to "0".
- For a pin shared with other peripheral functions, disable the output of such peripheral functions.
- When using a pin shared with the analog input function as an input port, set the corresponding bit in the A/D input disable register (lower) (AIDRL) to "1".
- When using a pin shared with the touch input function as an input port, set the bit in the touch input disable register 0/1 (TIDR0/TIDR1) corresponding to that pin to "1".
- If data is written to the PDRx register, the value is stored in the output latch but is not output to the pin set as an input port.
- Reading the PDRx register returns the pin value. However, if the read-modify-write (RMW) type of instruction is used to read the PDRx register, the PDRx register value is returned.

### ● Operation as a peripheral function output pin

- A pin becomes a peripheral function output pin if the peripheral output function is enabled by setting the output enable bit of a peripheral function corresponding to that pin.
- The pin value can be read from the PDRx register even if the peripheral function output is enabled. Therefore, the output value of a peripheral function can be read by the read operation on the PDRx register. However, if the read-modify-write (RMW) type of instruction is used to read the PDRx register, the PDRx register value is returned.

### ● Operation as a peripheral function input pin

- To set a pin as an input port, set the bit in the DDRx register bit corresponding to the input pin of a peripheral function to "0".
- When using a pin shared with the analog input function as another peripheral function input pin, configure it as an input port by setting the bit in the AIDRL register corresponding to that pin to "1".
- Reading the PDRx register returns the pin value, regardless of whether the peripheral function uses that pin as its input pin. However, if the read-modify-write (RMW) type of instruction is used to read the PDRx register, the PDRx register value is returned.

## ● Operation at reset

If the CPU is reset, all bits in the DDRx register are initialized to "0" and port input is enabled. As for a pin shared with the analog input function, its port input is disabled because the AIDRL register is initialized to "0". As for a pin shared with the touch input function, its port input is disabled because the TIDR0/TIDR1 register is initialized to "0".

## ● Operation in stop mode and watch mode

- If the pin state setting bit in the standby control register (STBC:SPL) is set to "1" and the device transits to stop mode or watch mode, the pin is compulsorily made to enter the high impedance state regardless of the DDRx register value. The input of that pin is locked at "L" level and blocked in order to prevent leaks due to input open. However, if the interrupt input is enabled for the external interrupt, the input is enabled and not blocked.
- If the pin state setting bit is "0", the state of the port I/O or that of the peripheral function I/O remains unchanged and the output level is maintained.

## ● Operation as an analog input pin

- Set the bit in the DDRx register bit corresponding to the analog input pin to "0" and the bit corresponding to that pin in the AIDRL register to "0".
- For a pin shared with other peripheral functions, disable the output of such peripheral functions. In addition, set the corresponding bit in the PULx register to "0".

## ● Operation as a touch input pin

- Set the bit in the DDRx register bit corresponding to the touch input pin to "0" and the bit corresponding to that pin in the TIDR0/TIDR1 register to "0".
- For a pin shared with other peripheral functions, disable the output of such peripheral functions. In addition, set the corresponding bit in the PULx register to "0".

## ● Operation as an external interrupt input pin

- Set the bit in the DDRx register corresponding to the external interrupt input pin to "0".
- For a pin shared with other peripheral functions, disable the output of such peripheral functions.
- The pin value is always input to the external interrupt circuit. When using a pin for a function other than the interrupt, disable the external interrupt function corresponding to that pin.

## ● Operation of the pull-up register

Setting the bit in the PULx register to "1" makes the pull-up resistor be internally connected to the pin. When the pin output is "L" level, the pull-up resistor is disconnected regardless of the value of the PULx register.



# **CHAPTER 7**

---

# ***TIME-BASE TIMER***

**This chapter describes the functions and operations of the time-base timer.**

- 7.1 Overview
- 7.2 Configuration
- 7.3 Interrupt
- 7.4 Operations and Setting Procedure Example
- 7.5 Register
- 7.6 Notes on Using Time-base Timer

## 7.1 Overview

The time-base timer is a 24-bit free-run down-counting counter. It is synchronized with the main clock divided by two, or with the main CR clock or with the main CR PLL clock. The clock can be selected by the SCS[2:0] bits in the SYCC register. The time-base timer has an interval timer function that can repeatedly generate interrupt requests at regular intervals.

### ■ Interval Timer Function

The interval timer function repeatedly generates interrupt requests at regular intervals by using the main clock divided by two, or using the main CR clock or using the main CR PLL clock as the count clock.

- The counter of the time-base timer counts down so that an interrupt request is generated every time a selected interval time elapses.
- The length of an interval time can be selected from the following 16 types.

Table 7.1-1 shows the interval times available for the time-base timer.

**Table 7.1-1 Interval Times of Time-base Timer**

|      | Interval time if the main clock is used<br>$(2^n \times 2/F_{CH}^{*1})$ | Interval time if the main CR clock is used<br>$(2^n \times 1/F_{CRH}^{*2})$ | Interval time if the main CR clock is multiplied by a PLL multiplication rate of 2<br>$(2^n \times 1/F_{MCRPLL}^{*3})$ |
|------|---|---|--|
| n=9  | 256 $\mu$ s   | 128 $\mu$ s   | 64 $\mu$ s   |
| n=10 | 512 $\mu$ s   | 256 $\mu$ s   | 128 $\mu$ s  |
| n=11 | 1.024 ms  | 512 $\mu$ s   | 256 $\mu$ s  |
| n=12 | 2.048 ms  | 1.024 ms  | 512 $\mu$ s  |
| n=13 | 4.096 ms  | 2.048 ms  | 1.024 ms   |
| n=14 | 8.192 ms  | 4.096 ms  | 2.048 ms   |
| n=15 | 16.384 ms   | 8.192 ms  | 4.096 ms   |
| n=16 | 32.768 ms   | 16.384 ms   | 8.192 ms   |
| n=17 | 65.536 ms   | 32.768 ms   | 16.384 ms  |
| n=18 | 131.072 ms  | 65.536 ms   | 32.768 ms  |
| n=19 | 262.144 ms  | 131.072 ms  | 65.536 ms  |
| n=20 | 524.288 ms  | 262.144 ms  | 131.072 ms   |
| n=21 | 1.049 s   | 524.288 ms  | 262.144 ms   |
| n=22 | 2.097 s   | 1.049 s   | 524.288 ms   |
| n=23 | 4.194 s   | 2.097 s   | 1.049 s  |
| n=24 | 8.389 s   | 4.194 s   | 2.097 s  |

\*1:  $F_{CH} = 4 \text{ MHz}$

$\therefore 2/F_{CH} = 0.5 \text{ } \mu\text{s}$

\*2:  $F_{CRH} = 4 \text{ MHz}$

$\therefore 1/F_{CRH} = 0.25 \text{ } \mu\text{s}$

\*3:  $F_{MCRPLL} = 8 \text{ MHz}$

PLL multiplication rate = 2

$F_{CRH} \times \text{PLL multiplication rate} = 4 \text{ MHz} \times 2 = 8 \text{ MHz}$

$\therefore 1/F_{MCRPLL} = 0.125 \text{ } \mu\text{s}$

# MB95850K/860K/870K Series

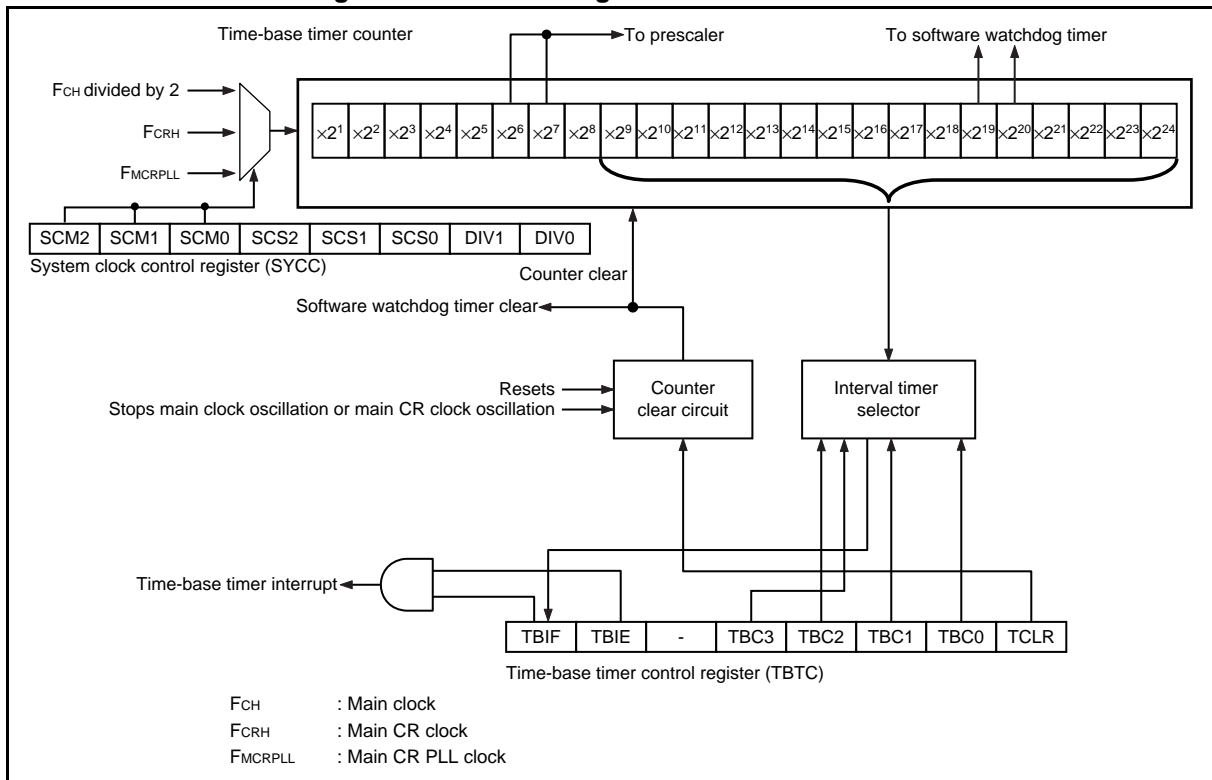
## 7.2 Configuration

The time-base timer consists of the following blocks:

- Time-base timer counter
- Counter clear circuit
- Interval timer selector
- Time-base timer control register (TBTC)

### ■ Block Diagram of Time-base Timer

Figure 7.2-1 Block Diagram of Time-base Timer



- Time-base timer counter

This is a 24-bit downcounter using the main clock divided by two, the main CR clock or the main CR PLL clock as its count clock.

- Counter clear circuit

This circuit controls the clearing of the time-base timer counter.

- Interval timer selector

This circuit selects one bit out of 16 bits in the 24 bits of the time-base timer counter as the interval timer.

- Time-base timer control register (TBTC)

This register selects the interval time, clears the counter, controls interrupts and checks the state of the time-base timer.

### ■ Input Clock

The time-base timer uses the main clock divided by two, or the main CR clock or the main CR PLL clock as its input clock (count clock).

### ■ Output Clock

The time-base timer supplies clocks to the clock supervisor counter, the software watchdog timer and the prescaler.



## MB95850K/860K/870K Series

### 7.3 Interrupt

---

**An interrupt request is generated when the interval time selected by the time-base timer elapses (interval timer function).**

---

#### ■ Interrupt When Interval Function Is in Operation

When the time-base timer counter counts down by using the internal count clock and the time-base timer counter underflows due to the passage of the selected interval time, the time-base timer interrupt request flag bit (TBTC:TBIF) is set to "1". If the time-base timer interrupt request enable bit is enabled (TBTC:TBIE = 1), an interrupt request will be generated to the interrupt controller.

- Regardless of the value of the TBIE bit, the TBIF bit is set to "1" when the selected bit underflows.
- With the TBIF bit having been set to "1", if the TBIE bit is changed from the disable state to the enable state (0 → 1), an interrupt request is generated immediately.
- The TBIF bit will not be set to "1" if the counter is cleared (TBTC:TCLR = 1) at the same time as the time-base timer counter underflows.
- In the interrupt service routine, write "0" to the TBIF bit to clear an interrupt request.

---

**Note:**

When enabling the output of interrupt requests after cancelling a reset (TBTC:TBIE = 1), always clear the TBIF bit at the same time (TBTC:TBIF = 0).

---

**Table 7.3-1 Interrupt of Time-base Timer**

| Item                | Description   |
|---------------------|---|
| Interrupt condition | The interval time set by "TBTC:TBC[3:0]" has elapsed. |
| Interrupt flag      | TBTC:TBIF   |
| Interrupt enable    | TBTC:TBIE   |

## 7.4 Operations and Setting Procedure Example

This section describes the operations of the interval timer function of the time-base timer.

### ■ Operations of Time-base Timer

The counter of the time-base timer is initialized to "0xFFFFFFFF" after a reset, and starts counting while being synchronized with the main clock divided by two, or with the main CR clock or with the main CR PLL clock.

The time-base timer continues to count down as long as the main clock, the main CR clock or the main CR PLL clock is oscillating. Once the main clock, the main CR clock or the main CR PLL clock stops, the counter stops counting and is initialized to "0xFFFFFFFF".

To use the interval timer function, do the settings shown in Figure 7.4-1.

Figure 7.4-1 Settings of Interval Timer Function

|      | bit7  | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|------|---|------|------|------|------|------|------|------|
| TBTC | TBIF  | TBIE | -    | TBC3 | TBC2 | TBC1 | TBC0 | TCLR |
|      | 0   | 1    |      | ⊙    | ⊙    | ⊙    | ⊙    | 0    |
|      | ⊙: Bit to be used<br>1: Set to "1".<br>0: Set to "0". |      |      |      |      |      |      |      |

When the time-base timer clear bit in the time-base timer control register (TBTC:TCLR) is set to "1", the counter of the time-base timer is initialized to "0xFFFFFFFF" and continues to count down. When the selected interval time has elapsed, the time-base timer interrupt request flag bit in the time-base timer control register (TBTC:TBIF) becomes "1". In other words, an interrupt request is generated at each interval time selected, based on the time when the counter was last cleared.

### ■ Clearing Time-base Timer

With the output of the time-base timer being used in other peripheral functions, clearing the time-base timer affects their operations in various ways such as changing the count time of a peripheral function.

When clearing the counter by using the time-base timer clear bit (TBTC:TCLR), modify the settings of other peripheral functions whenever necessary so that clearing the counter does not have any unexpected effect on them.

When the output of the time-base timer is selected as the count clock for the watchdog timer, clearing the time-base timer also clears the watchdog timer.

The time-base timer is cleared not only by the TCLR bit, but also when the main clock, the main CR clock, or the main CR PLL clock is stopped and the oscillation stabilization wait time is necessary. The time-base timer is cleared in the following situations:

- The device transits from the main clock mode, the main CR clock mode or the main CR PLL clock mode to the stop mode.
- The device transits from the main clock mode, the main CR clock mode or the main CR PLL clock mode to the subclock mode or the sub-CR clock mode.
- At power-on
- At low-voltage detection reset

■ Operation Examples of Time-base Timer

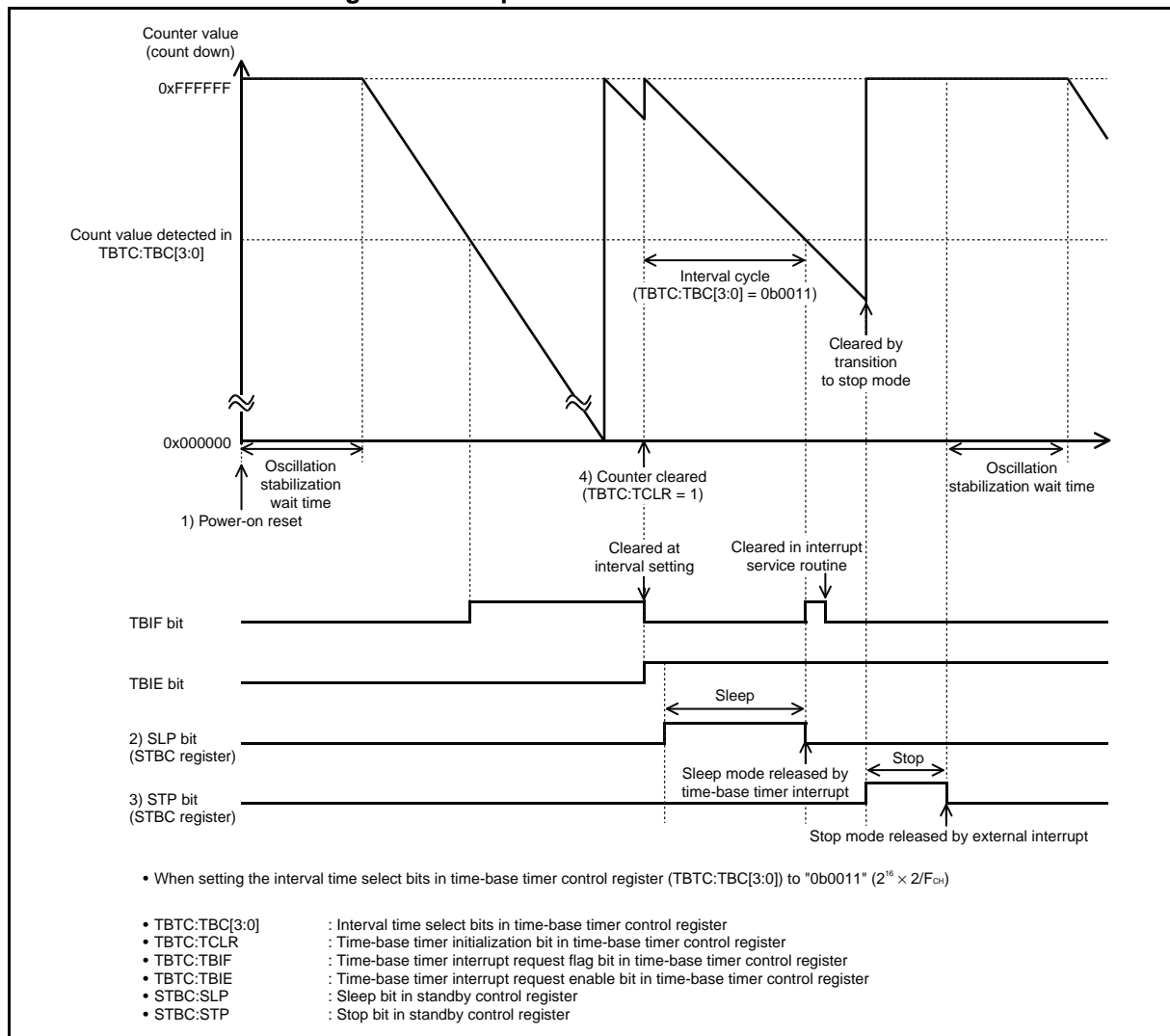
Figure 7.4-2 shows examples of operations under the following conditions:

1. A power-on reset is generated.
2. The device transits to the sleep mode during the operation of the interval timer function in main clock mode, main CR clock mode or main CR PLL clock mode.
3. The device transits to the stop mode in main clock mode, main CR clock mode or main CR PLL clock mode.
4. A request is generated to clear the counter.

If the device transits to the time-base timer mode, the same operations are executed as those executed when the device transits to the sleep mode.

In stop mode in which the clock mode is subclock mode, sub-CR clock mode, main clock mode, main CR clock mode or main CR PLL clock mode, the timer operation stops because it is cleared and the main clock stops.

Figure 7.4-2 Operations of Time-base Timer



**■ Setting Procedure Example**

Below is an example of procedure for setting the time-base timer.

● Initial settings

1. Set the interrupt level. (ILR\*)
2. Set the interval time. (TBTC:TBC[3:0])
3. Enable interrupts and clear the interrupt request flag. (TBTC:TBIE = 1, TBTC:TBIF = 0)
4. Clear the counter. (TBTC:TCLR = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

● Processing interrupts

1. Clear the interrupt request flag. (TBTC:TBIF = 0)
2. Clear the counter. (TBTC:TCLR = 1)

**MB95850K/860K/870K Series****7.5 Register**

---

**This section describes the register of the time-base timer.**

---

**Table 7.5-1 List of Time-base Timer Register**

| Register abbreviation | Register name                    | Reference |
|-----------------------|----------------------------------|-----------|
| TBTC                  | Time-base timer control register | 7.5.1     |

## 7.5.1 Time-base Timer Control Register (TBTC)

The time-base timer control register (TBTC) selects the interval time, clears the counter, controls interrupts and checks the status of the time-base timer.

### Register Configuration

| bit           | 7    | 6    | 5 | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|---|------|------|------|------|------|
| Field         | TBIF | TBIE | — | TBC3 | TBC2 | TBC1 | TBC0 | TCLR |
| Attribute     | R/W  | R/W  | — | R/W  | R/W  | R/W  | R/W  | W    |
| Initial value | 0    | 0    | 0 | 0    | 0    | 0    | 0    | 0    |

### Register Functions

#### [bit7] TBIF: Time-base timer interrupt request flag bit

This bit is set to "1" when the interval time selected by the time-base timer has elapsed.

When this bit and the time-base timer interrupt request enable bit (TBIE) are set to "1", an interrupt request is output.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit7               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the interval time has not elapsed. |
| <b>Reading "1"</b> | Indicates that the interval time has elapsed.     |
| <b>Writing "0"</b> | Clears this bit.                                  |
| <b>Writing "1"</b> | Has no effect on operation.                       |

#### [bit6] TBIE: Time-base timer interrupt request enable bit

This bit enables or disables output of interrupt requests to interrupt controller.

When this bit and the time-base timer interrupt request flag bit (TBIF) are set to "1", a time-base timer interrupt request is output.

| bit6               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the time-base timer interrupt request. |
| <b>Writing "1"</b> | Enables the time-base timer interrupt request.  |

#### [bit5] Undefined bit

The read value is always "0". Writing a value to this bit has no effect on operation.

[bit4:1] TBC[3:0]: Interval time select bits

These bits select interval time.

| bit4:1         | Details   |   |  |
|----------------|---|---|--|
|                | Interval time<br>(Main clock,<br>$F_{CH} = 4 \text{ MHz}$ ) | Interval time<br>(Main CR clock,<br>$F_{CRH} = 4 \text{ MHz}$ ) | Interval time<br>(Main CR clock multiplied by<br>a PLL multiplication rate of<br>2, $F_{MCRPLL} = 8 \text{ MHz}$ ) |
| Writing "0100" | $2^9 \times 2/F_{CH}$ (256 $\mu\text{s}$ )                  | $2^9 \times 1/F_{CRH}$ (128 $\mu\text{s}$ )                     | $2^9 \times 1/F_{MCRPLL}$ (64 $\mu\text{s}$ )  |
| Writing "0000" | $2^{10} \times 2/F_{CH}$ (512 $\mu\text{s}$ )               | $2^{10} \times 1/F_{CRH}$ (256 $\mu\text{s}$ )                  | $2^{10} \times 1/F_{MCRPLL}$ (128 $\mu\text{s}$ )  |
| Writing "0101" | $2^{11} \times 2/F_{CH}$ (1.024 ms)                         | $2^{11} \times 1/F_{CRH}$ (512 $\mu\text{s}$ )                  | $2^{11} \times 1/F_{MCRPLL}$ (256 $\mu\text{s}$ )  |
| Writing "0001" | $2^{12} \times 2/F_{CH}$ (2.048 ms)                         | $2^{12} \times 1/F_{CRH}$ (1.024 ms)                            | $2^{12} \times 1/F_{MCRPLL}$ (512 $\mu\text{s}$ )  |
| Writing "0110" | $2^{13} \times 2/F_{CH}$ (4.096 ms)                         | $2^{13} \times 1/F_{CRH}$ (2.048 ms)                            | $2^{13} \times 1/F_{MCRPLL}$ (1.024ms)   |
| Writing "0010" | $2^{14} \times 2/F_{CH}$ (8.192 ms)                         | $2^{14} \times 1/F_{CRH}$ (4.096 ms)                            | $2^{14} \times 1/F_{MCRPLL}$ (2.048 ms)  |
| Writing "0111" | $2^{15} \times 2/F_{CH}$ (16.384 ms)                        | $2^{15} \times 1/F_{CRH}$ (8.192 ms)                            | $2^{15} \times 1/F_{MCRPLL}$ (4.096 ms)  |
| Writing "0011" | $2^{16} \times 2/F_{CH}$ (32.768 ms)                        | $2^{16} \times 1/F_{CRH}$ (16.384 ms)                           | $2^{16} \times 1/F_{MCRPLL}$ (8.192 ms)  |
| Writing "1000" | $2^{17} \times 2/F_{CH}$ (65.536 ms)                        | $2^{17} \times 1/F_{CRH}$ (32.768 ms)                           | $2^{17} \times 1/F_{MCRPLL}$ (16.384 ms)   |
| Writing "1001" | $2^{18} \times 2/F_{CH}$ (131.072 ms)                       | $2^{18} \times 1/F_{CRH}$ (65.536 ms)                           | $2^{18} \times 1/F_{MCRPLL}$ (32.768 ms)   |
| Writing "1010" | $2^{19} \times 2/F_{CH}$ (262.144 ms)                       | $2^{19} \times 1/F_{CRH}$ (131.072 ms)                          | $2^{19} \times 1/F_{MCRPLL}$ (65.536 ms)   |
| Writing "1011" | $2^{20} \times 2/F_{CH}$ (524.288 ms)                       | $2^{20} \times 1/F_{CRH}$ (262.144 ms)                          | $2^{20} \times 1/F_{MCRPLL}$ (131.072 ms)  |
| Writing "1100" | $2^{21} \times 2/F_{CH}$ (1.049 s)                          | $2^{21} \times 1/F_{CRH}$ (524.288 ms)                          | $2^{21} \times 1/F_{MCRPLL}$ (262.144 ms)  |
| Writing "1101" | $2^{22} \times 2/F_{CH}$ (2.097 s)                          | $2^{22} \times 1/F_{CRH}$ (1.049 s)                             | $2^{22} \times 1/F_{MCRPLL}$ (524.288 ms)  |
| Writing "1110" | $2^{23} \times 2/F_{CH}$ (4.194 s)                          | $2^{23} \times 1/F_{CRH}$ (2.097 s)                             | $2^{23} \times 1/F_{MCRPLL}$ (1.049 s)   |
| Writing "1111" | $2^{24} \times 2/F_{CH}$ (8.389 s)                          | $2^{24} \times 1/F_{CRH}$ (4.194 s)                             | $2^{24} \times 1/F_{MCRPLL}$ (2.097 s)   |

[bit0] TCLR: Time-base timer clear bit

This bit clears all bits in the counter of the time-base timer to "1".

| bit0        | Details   |
|-------------|---|
| Read access | The read value is always "0".                                 |
| Writing "0" | Has no effect on operation.                                   |
| Writing "1" | Clears all bits in the counter of the time-base timer to "1". |

Note: When the output of the time-base timer is selected as the count clock for the software watchdog timer, clear the time-base timer with this bit also clears the software watchdog timer.

## 7.6 Notes on Using Time-base Timer

---

This section provides notes on using the time-base timer.

---

### ■ Notes on Using Time-base Timer

- When setting the timer by program

The timer cannot return from interrupt processing when the time-base timer interrupt request flag bit (TBTC:TBIF) is set to "1" and the interrupt request enable bit is enabled (TBTC:TBIE = 1). Always clear the TBIF bit in the interrupt service routine.

- Clearing Time-base Timer

The time-base timer is cleared not only by the time-base timer clear bit (TBTC:TCLR = 1) but also when the oscillation stabilization wait time of the main clock, the main CR clock, or the main CR PLL clock is required. When the time-base timer is selected as the count clock of the software watchdog timer (WDTC:CS[1:0] = 0b00 or 0b01), clearing the time-base timer also clears the software watchdog timer.

- Peripheral functions receiving clock from time-base timer

In the mode where the source oscillation of the main clock is stopped, the counter is cleared and the time-base timer stops operating. In addition, if the counter of the time-base timer is cleared with the output of the time-base timer being used in other peripheral functions, that will affect the operations of such peripheral operations such as the changing of their operating cycles.

After the counter of the time-base timer is cleared, the clock that is output from the time-base timer for the software watchdog timer returns to the initial state. However, since the software watchdog timer counter is also cleared at the same time as the clock for the software watchdog timer returns to the initial state, the software watchdog timer operates in its normal cycle.



# **CHAPTER 8**

---

# ***HARDWARE/SOFTWARE WATCHDOG TIMER***

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

- 8.1 Overview
- 8.2 Configuration
- 8.3 Operations and Setting Procedure Example
- 8.4 Register
- 8.5 Notes on Using Watchdog Timer

## 8.1 Overview

**The watchdog timer serves as a counter used to prevent programs from running out of control.**

### ■ Watchdog Timer Function

The watchdog timer functions as a counter used to prevent programs from running out of control. Once the watchdog timer is activated, its counter needs to be cleared at specified intervals regularly. A watchdog reset is generated if the timer is not cleared within a certain amount of time due to a problem such as a program entering an infinite loop.

#### ● Count clock for the software/hardware watchdog timer

- For the software watchdog timer, the output of the time-base timer or of the watch prescaler or of the sub-CR timer can be used as the count clock.
- For the hardware watchdog timer, only the output of the sub-CR timer can be used as the count clock.

#### ● Activation of the software/hardware watchdog timer

- The software/hardware watchdog timer is to be activated according to the values at the addresses 0xFFBE and 0xFFBF on the Flash memory, which are copied to the watchdog timer selection ID register (upper/lower) (WDTH/WDTL) (0x0FEB/0x0FEC).
- In the case of software activation (software watchdog), the watchdog timer register (WDTC) must be set to start the watchdog timer function.
- In the case of hardware activation (hardware watchdog), the watchdog timer starts automatically after a reset. It can also stop or run in stop mode according to the values at the addresses 0xFFBE and 0xFFBF on the Flash memory, which are copied to the watchdog timer selection ID register (upper/lower) (WDTH/WDTL) (0x0FEB/0x0FEC). See "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE" for details of the watchdog timer selection ID.
- The intervals of the watchdog timer are shown in Table 8.1-1. If the counter of the watchdog timer is not cleared, a watchdog reset is generated between the minimum time and the maximum time. Clear the counter of the watchdog timer within the minimum time.

**Table 8.1-1 Interval Times of Watchdog Timer**

| Count clock type                                   | Count clock switch bit<br>CS[1:0], CSP                            | Interval time |              |
|--|---|---------------|--------------|
|  |   | Minimum time  | Maximum time |
| Time-base timer output<br>(main clock = 4 MHz)     | 0b000 (software watchdog timer)                                   | 524 ms        | 1.05 s       |
|  | 0b010 (software watchdog timer)                                   | 262 ms        | 524 ms       |
| Watch prescaler output<br>(subclock = 32.768 kHz)  | 0b100 (software watchdog timer)                                   | 500 ms        | 1.00 s       |
|  | 0b110 (software watchdog timer)                                   | 250 ms        | 500 ms       |
| Sub-CR timer<br>(sub-CR clock = 50 kHz to 150 kHz) | 0bXX1*1 (software watchdog timer) or<br>hardware watchdog timer*2 | 437 ms        | 2.62 s       |

\*1: X = 0 or 1

\*2: CS[1:0] = 0b00, CSP = 1 (read-only)

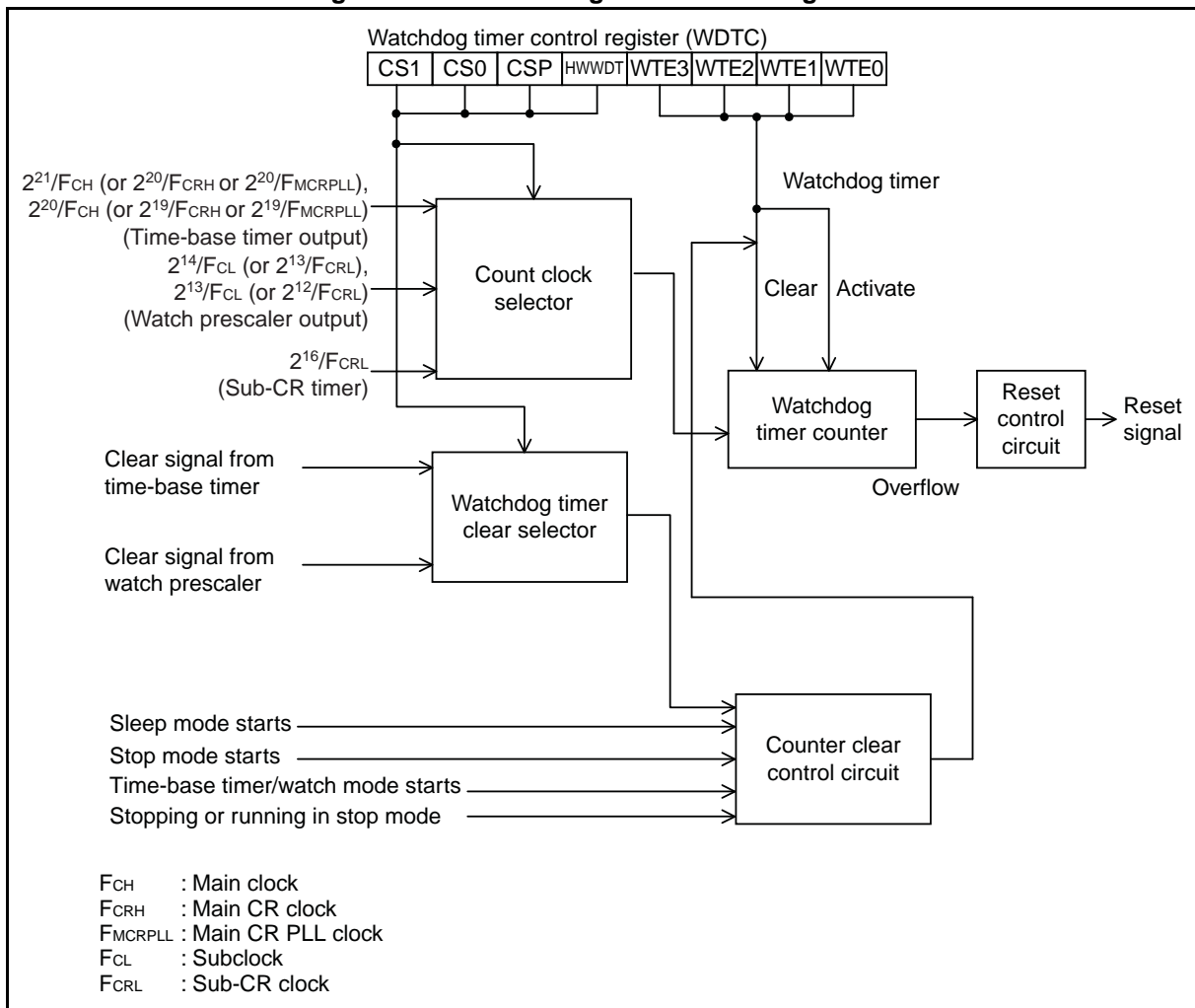
## 8.2 Configuration

The watchdog timer consists of the following 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 Watchdog Timer

Figure 8.2-1 Block Diagram of Watchdog Timer



- Count clock selector

This selector selects the count clock of the watchdog timer counter.

- Watchdog timer counter

This is a 1-bit counter that uses the output of the time-base timer, of the watch prescaler or of the sub-CR timer as the count clock.

- Reset control circuit

This circuit generates a reset signal when the watchdog timer counter overflows.

- Watchdog timer clear selector

This selector selects the watchdog timer clear signal.

- Counter clear control circuit

This circuit controls the clearing and stopping of the watchdog timer counter.

- Watchdog timer control register (WDTC)

This register performs setup for activating/clearing the watchdog timer counter as well as for selecting the count clock.

## ■ Input Clock

The watchdog timer uses the output clock of the time-base timer, of the watch prescaler or of the sub-CR timer as the input clock (count clock).

## 8.3 Operations and Setting Procedure Example

---

**The watchdog timer generates a watchdog reset when the watchdog timer counter overflows.**

---

### ■ Operations of Watchdog Timer

#### ● How to activate the watchdog timer

##### Software watchdog

- The watchdog timer is activated when "0b0101" is written to the watchdog control bits of the watchdog timer control register (WDTC:WTE[3:0]) for the first time after a reset. The count clock switch bits of the watchdog timer control register (WDTC:CS[1:0], CSP) should also be set at the same time.
- Once the watchdog timer is activated, a reset is the only way to stop its operation.

##### Hardware watchdog

- To activate the hardware watchdog timer, write any value except "0xA596" to the addresses 0xFFBE and 0xFFBF on the Flash memory. After a reset, the data in 0xFFBE and 0xFFBF on the Flash memory are copied to the watchdog timer selection ID register (upper/lower) (WDTH/WDTL) (0xFEBC/0xFECD). Writing "0xA597" to the addresses 0xFFBE and 0xFFBF on the Flash memory enables the hardware watchdog timer except in standby modes; writing any value other than "0xA596" and "0xA597" enables the hardware watchdog timer in all modes. See "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE" for details of the watchdog timer selection ID.
- Start operation after a reset is released.
- CS[1:0] and CSP bits are read-only bits fixed at "0b001".
- The counter of the watchdog timer is cleared by a reset, and the watchdog timer resumes its operation after the reset is released.

#### ● Clearing the watchdog timer

- When the counter of the watchdog timer is not cleared within the interval time, it overflows, allowing the watchdog timer to generate a watchdog reset.
- The counter of the hardware watchdog timer is cleared when "0b0101" is written to the watchdog control bits of the watchdog timer control register (WDTC:WTE[3:0]). The counter of the software watchdog timer is cleared when "0b0101" is written to the watchdog control bits of the watchdog timer control register (WDTC:WTE[3:0]) for the second time and from the second time onward.
- The watchdog timer is cleared at the same time as the timer selected as the count clock (time-base timer or watch prescaler) is cleared.

#### ● Operation in standby mode

- In the case of activating the software watchdog timer, or starting the hardware watchdog timer with its operation in standby mode disabled, regardless of the clock mode selected, once the device transits to standby mode, the counter of the watchdog timer is cleared and the watchdog timer stops its operation. When the device wakes up from standby mode, the watchdog timer resumes its operation.
- In the case of activating the hardware watchdog timer with its operation in standby mode enabled, whether the device transits to standby mode or wakes up from standby mode, the counter of the watchdog timer is not cleared and the watchdog timer continues its operation.

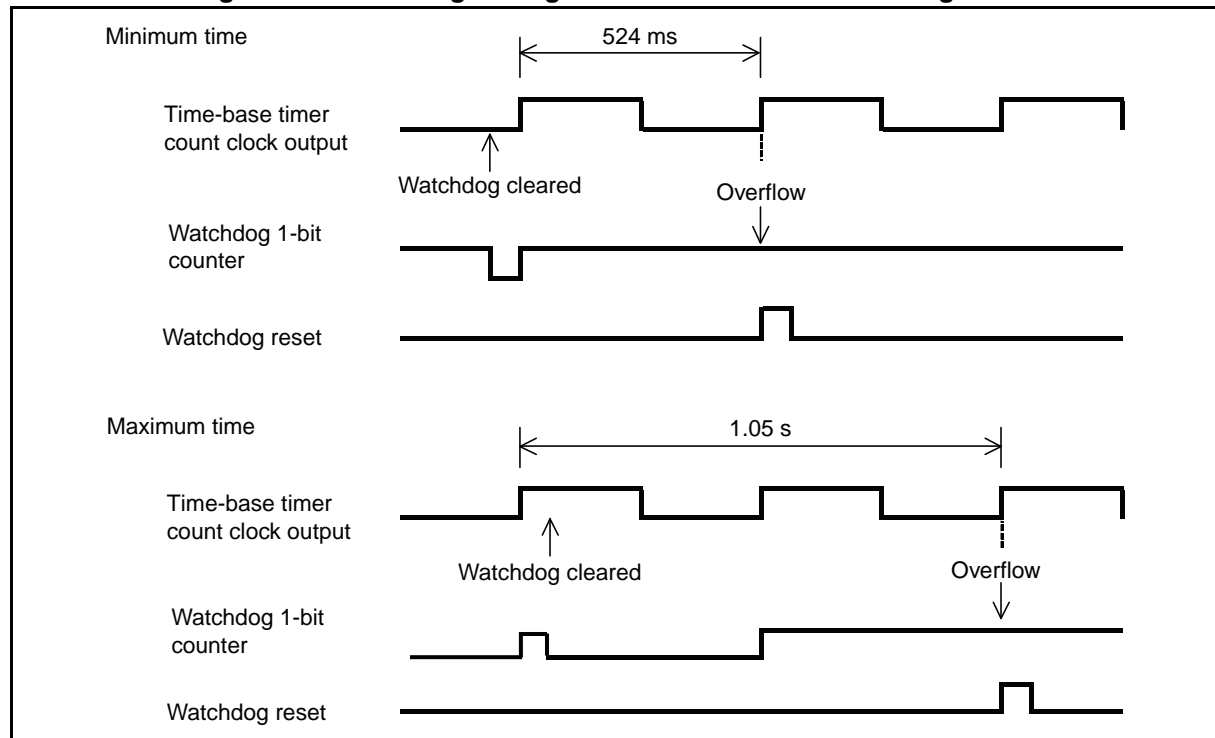
**Note:**

The watchdog timer is also cleared when the timer selected as the count clock (time-base timer or watch prescaler) is cleared. For this reason, the watchdog timer cannot function if the software is set to repeatedly clear the timer selected as the count clock of the watchdog timer at the interval time selected for the watchdog timer.

- Interval time

The interval time varies depending on the timing of clearing the watchdog timer. Figure 8.3-1 shows the correlation between the timing of clearing the watchdog timer and the interval time when the time-base timer output  $F_{CH}/2^{21}$  ( $F_{CH}$ : main clock) is selected as the count clock (main clock = 4 MHz).

**Figure 8.3-1 Clearing Timing and Interval Time of Watchdog Timer**



- Operation in subclock mode

When a watchdog reset is generated in subclock mode, the timer starts operating in main clock mode after the oscillation stabilization wait time has elapsed. The reset signal is output during this oscillation stabilization wait time.

■ **Setting Procedure Example**

Below is the procedure for setting the software watchdog timer.

1. Select the count clock. (WDTC:CS[1:0], CSP)
2. Activate the watchdog timer. (WDTC:WTE[3:0] = 0b0101)
3. Clear the watchdog timer. (WDTC:WTE[3:0] = 0b0101)

Below is the procedure for setting the hardware watchdog timer.

1. Write any value except "0xA596" to the addresses 0xFFBE and 0xFFBF on the Flash memory. After a reset, the data in 0xFFBE and 0xFFBF on the Flash memory are copied to the watchdog timer selection ID register (upper/lower) (WDTH/WDTL) (0x0FEB/0x0FEC). Writing "0xA597" to the addresses 0xFFBE and 0xFFBF on the Flash memory enables the hardware watchdog timer except in standby modes; writing any value other than "0xA596" and "0xA597" enables the hardware watchdog timer in all modes. See "CHAPTER 24 NON-VOLATILE REGISTER (NVR) INTERFACE" for details of the watchdog timer selection ID.
2. Clear the watchdog timer. (WDTC:WTE[3:0] = 0b0101)

## 8.4 Register

---

**This section describes the register of the watchdog timer.**

---

**Table 8.4-1 List of Watchdog Timer Register**

| Register abbreviation | Register name                   | Reference |
|-----------------------|---------------------------------|-----------|
| WDTC                  | Watchdog timer control register | 8.4.1     |



## 8.4.1 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) activates or clears the watchdog timer.

### Register Configuration

| bit  | 7   | 6   | 5   | 4     | 3    | 2    | 1    | 0    |
|--|-----|-----|-----|-------|------|------|------|------|
| Field  | CS1 | CS0 | CSP | HWWDT | WTE3 | WTE2 | WTE1 | WTE0 |
| Attribute and initial values for software watchdog timer |     |     |     |       |      |      |      |      |
| Attribute  | R/W | R/W | R/W | R     | W    | W    | W    | W    |
| Initial value  | 0   | 0   | 0   | 0     | 0    | 0    | 0    | 0    |
| Attribute and initial values for hardware watchdog timer |     |     |     |       |      |      |      |      |
| Attribute  | R   | R   | R   | R     | W    | W    | W    | W    |
| Initial value  | 0   | 0   | 1   | 1     | 0    | 0    | 0    | 0    |

### Register Functions

[bit7:6] CS[1:0]: Count clock switch bits

[bit5] CSP: Count clock select sub-CR selector bit

These bits select the count clock of the watchdog timer.

Write to these bits at the same time as activating the watchdog timer by the watchdog control bits.

No change can be made once the watchdog timer is activated.

|                | bit7 | bit6 | bit5 | Details<br>(F <sub>CH</sub> : main clock, F <sub>CRH</sub> : main CR clock, F <sub>MCRPLL</sub> : main CR PLL clock,<br>F <sub>CL</sub> : subclock, F <sub>CRL</sub> : sub-CR clock) |
|----------------|------|------|------|--|
| <b>Writing</b> | 0    | 0    | 0    | Output cycle of time-base timer ( $2^{21}/F_{CH}$ , $2^{20}/F_{CRH}$ or $2^{20}/F_{MCRPLL}$ )  |
| <b>Writing</b> | 0    | 1    | 0    | Output cycle of time-base timer ( $2^{20}/F_{CH}$ , $2^{19}/F_{CRH}$ or $2^{19}/F_{MCRPLL}$ )  |
| <b>Writing</b> | 1    | 0    | 0    | Output cycle of watch prescaler ( $2^{14}/F_{CL}$ or $2^{13}/F_{CRL}$ )  |
| <b>Writing</b> | 1    | 1    | 0    | Output cycle of watch prescaler ( $2^{13}/F_{CL}$ or $2^{12}/F_{CRL}$ )  |
| <b>Writing</b> | 0/1  | 0/1  | 1    | Output cycle of sub-CR timer ( $2^{16}/F_{CRL}$ )  |

Note: Since the time-base timer is stopped in subclock mode or sub-CR clock mode, always select the output of the watch prescaler in subclock mode.

[bit4] HWWDT: Hardware watchdog timer start bit

This is a read-only bit used to confirm the start/stop of the hardware watchdog timer.

| bit4               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the hardware watchdog timer has stopped (The software watchdog timer can be activated). |
| <b>Reading "1"</b> | Indicates that the hardware watchdog timer has been activated.   |

[bit3:0] WTE[3:0]: Watchdog control bits

These bits controls the watchdog timer.

The read value of these bits is always "0b0000".

| bit3:0                                   | Details  |
|--|--|
| <b>Writing "0101"</b>                    | Activates the watchdog timer (in the first write access after a reset) or clears it (from the second write access after a reset). <ul style="list-style-type: none"> <li>• In the case of activating the watchdog timer<br/>Writing "0101" to these bits in the first write access after a reset starts the software watchdog timer.</li> <li>• In the case of clearing the watchdog timer<br/>Writing "0101" to these bits in the first write access or later after a reset clears the hardware watchdog timer.<br/>Writing "0101" to these bits in the second write access or later after a reset clears the software watchdog timer.</li> </ul> |
| <b>Writing a value other than "0101"</b> | Has no effect on operation.  |

---

**Note:**

Using the read-modify-write (RMW) type of instruction to access the WDTC register is prohibited.

---

## 8.5 Notes on Using Watchdog Timer

---

This section provides notes on using the watchdog timer.

---

### ■ Notes on Using Watchdog Timer

- Stopping the watchdog timer

Software watchdog timer

Once activated, the watchdog timer cannot be stopped until a reset is generated.

- Selecting the count clock

Software watchdog timer

The count clock switch bits (WDTC:CS[1:0], CSP) can be modified only when the watchdog control bits (WDTC:WTE[3:0]) are set to "0b0101" after the activation of the watchdog timer. The count clock switch bits cannot be set by a bit manipulation instruction. Moreover, the bit settings should not be changed once the timer is activated.

In subclock mode or sub-CR clock mode, the time-base timer does not operate because the main clock, the main CR clock, or the main CR PLL clock stops oscillating.

In order to make the watchdog timer operate in subclock mode or sub-CR clock mode, it is necessary to select the watch prescaler as the count clock beforehand and set WDTC:CS[1:0], CSP to "0b100" or "0b110" or "0bXX1" (X = 0 or 1).

- Clearing the watchdog timer

Clearing the timer (time-base timer, watch prescaler or sub-CR timer) used as the count clock of the watchdog timer also clears the counter of the watchdog timer.

The counter of the watchdog timer is cleared when the watchdog timer transits to sleep mode, stop mode, or watch mode, except in the case of activating hardware watchdog timer whose operation in standby mode has been enabled.

- Programming precaution

When creating a program in which the watchdog timer is cleared repeatedly in the main loop, set the processing time of the main loop including the interrupt processing time to the minimum watchdog timer interval time or shorter.

- Hardware watchdog timer (operation in standby mode has been enabled)

The hardware watchdog timer does not stop in stop mode, sleep mode, time-base timer mode or watch mode. Therefore, the hardware watchdog timer is not cleared by the CPU even if the internal clock stops. (in stop mode, sleep mode, time-base timer mode or watch mode).

Regularly release the device from standby mode and clear the watchdog timer. However, depending on the setting of the oscillation stabilization wait time setting register, a watchdog reset may be generated after the CPU wakes up from stop mode in subclock mode or sub-CR clock mode.

Take account of the setting of the subclock stabilization wait time when selecting the subclock.



# **CHAPTER 9**

---

# **WATCH PRESCALER**

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

- 9.1 Overview
- 9.2 Configuration
- 9.3 Interrupt
- 9.4 Operations and Setting Procedure Example
- 9.5 Register
- 9.6 Notes on Using Watch Prescaler

## 9.1 Overview

The watch prescaler is a 16-bit down-counting, free-run counter, which is synchronized with the subclock divided by two or the sub-CR clock divided by two. It has an interval timer function that continuously generates interrupt requests at regular intervals.

### ■ Interval Timer Function

The interval timer function continuously generates interrupt requests at regular intervals, using the subclock divided by two or the sub-CR clock divided by two as its count clock.

- The counter of the watch prescaler counts down and an interrupt request is generated whenever the selected interval time has elapsed.
- The interval time can be selected from the following eight types:

Table 9.1-1 shows the interval times of the watch prescaler.

**Table 9.1-1 Interval Times of Watch Prescaler**

|      | Interval time<br>(Sub-CR clock)<br>( $2^n \times 2/F_{CRL}^{*1}$ ) | Interval time<br>(Subclock)<br>( $2^n \times 2/F_{CL}^{*2}$ ) |
|------|--|---|
| n=10 | 20.48 ms   | 62.5 ms   |
| n=11 | 40.96 ms   | 125 ms  |
| n=12 | 81.92 ms   | 250 ms  |
| n=13 | 163.84 ms  | 500 ms  |
| n=14 | 327.68 ms  | 1 s   |
| n=15 | 655.36 ms  | 2 s   |
| n=16 | 1.311 s  | 4 s   |
| n=17 | 2.621 s  | 8 s   |

\*1:  $2/F_{CRL}=20 \mu\text{s}$  when  $F_{CRL}=100 \text{ kHz}$

\*2:  $2/F_{CL}=61.035 \mu\text{s}$  when  $F_{CL}=32.768 \text{ kHz}$

Note:

Refer to the device data sheet for the accuracy of the sub-CR clock frequency.

# MB95850K/860K/870K Series

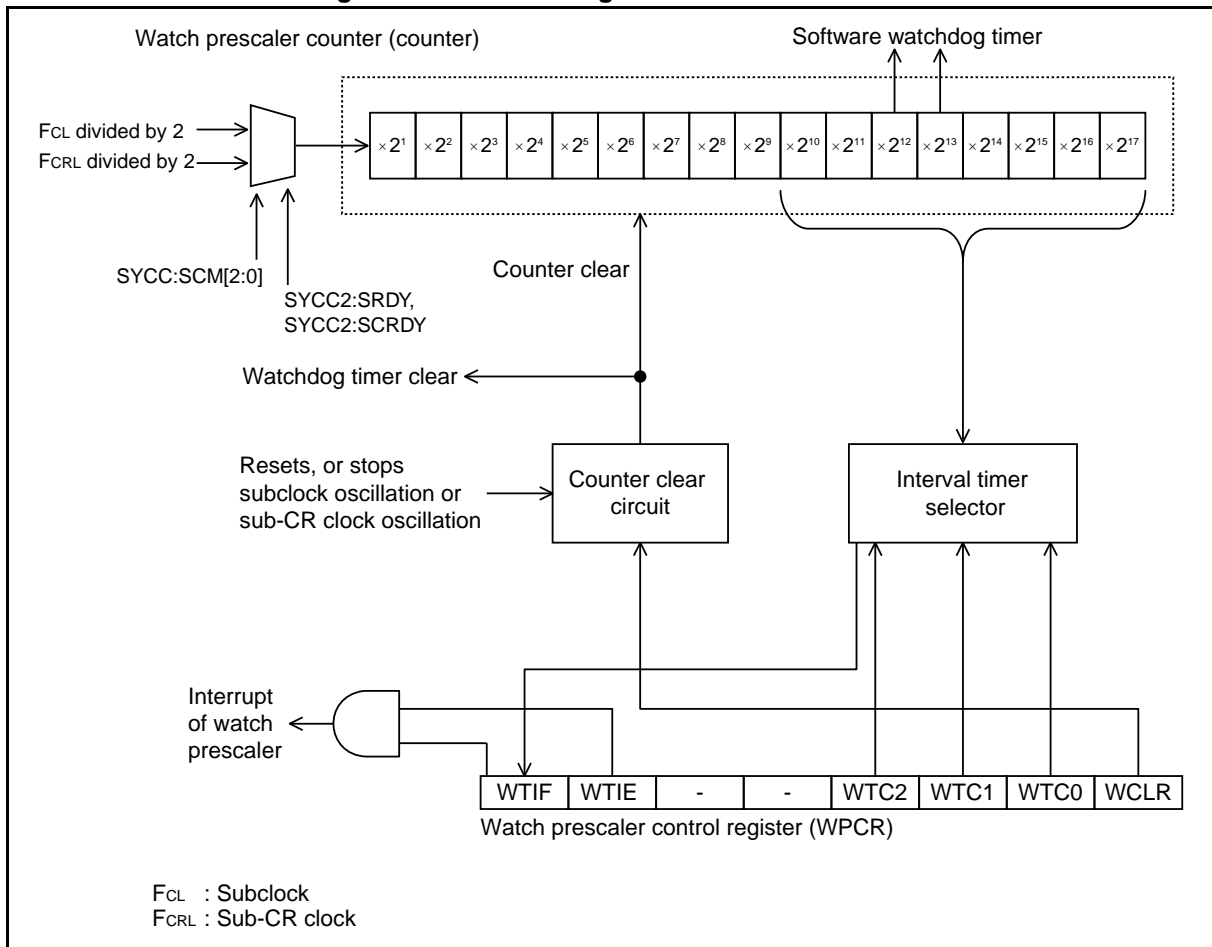
## 9.2 Configuration

The watch prescaler consists of the following blocks:

- Watch prescaler counter
- Counter clear circuit
- Interval timer selector
- Watch prescaler control register (WPCR)

### ■ Block Diagram of Watch Prescaler

Figure 9.2-1 Block Diagram of Watch Prescaler



- Watch prescaler counter (counter)

This is a 16-bit downcounter that uses the subclock divided by two or the sub-CR clock divided by two as its count clock.

- Counter clear circuit

This circuit controls the clearing of the watch prescaler.

- Interval timer selector

This circuit selects one out of the eight bits used for the interval timer among 17 bits available in the watch prescaler counter.

- Watch prescaler control register (WPCR)

This register selects the interval time, clears the counter, controls interrupts and checks the status.

### ■ Input Clock

The watch prescaler uses the subclock divided by two or the sub-CR clock divided by two as its input clock (count clock).

### ■ Output Clock

The watch prescaler supplies its clock to the software watchdog timer.



## MB95850K/860K/870K Series

### 9.3 Interrupt

**An interrupt request is generated when the selected interval time of the watch prescaler has elapsed (interval timer function).**

#### ■ Interrupts in Operation of Interval Timer Function (Watch Prescaler Interrupts)

In any mode except the stop mode in which the subclock mode or the sub-CR clock mode is used, if the watch prescaler counter counts down using the subclock divided by two or the sub-CR clock divided by two and the selected interval time elapses, the watch prescaler interrupt request flag bit is set to "1" (WPCR:WTIF = 1). At that time, if the watch prescaler interrupt request enable bit has been enabled (WPCR:WTIE = 1), an interrupt request is output from the watch prescaler to the interrupt controller.

- Regardless of the value in the WTIE bit, the WTIF bit is set to "1" as soon as the time set by the watch prescaler interrupt interval time select bits has elapsed.
- When the WTIF bit is set to "1", changing the WTIE bit from the disable state to the enable state (WPCR:WTIE = 0 → 1) immediately generates an interrupt request.
- The WTIF bit will not be set to "1" if the counter is cleared (WPCR:WCLR = 1) at the same time as the selected bit overflows.
- Write "0" to the WTIF bit in the interrupt service routine to clear an interrupt request.

**Note:**

To enable the output of interrupt requests after releasing a reset, set the WTIE bit in the WPCR register to "1" and clear the WTIF bit in the same register simultaneously.

**Table 9.3-1 Interrupt of Watch Prescaler**

| Item                | Description                                       |
|---------------------|---|
| Interrupt condition | Interval time set by "WPCR:WTC[2:0]" has elapsed. |
| Interrupt flag      | WPCR:WTIF   |
| Interrupt enable    | WPCR:WTIE   |

## 9.4 Operations and Setting Procedure Example

---

The watch prescaler operates as an interval timer.

---

### ■ Operations of Interval Timer Function (Watch Prescaler)

The counter of the watch prescaler continues to count down using the subclock divided by two or the sub-CR clock divided by two as its count clock as long as the subclock or the sub-CR clock oscillates.

When cleared (WPCR:WCLR = 1), the counter starts counting down from "0xFFFF". Once it reaches "0x0000", it returns to "0xFFFF" to continue counting. As soon as the time set by the interrupt interval time select bits has elapsed during the counting down, the watch prescaler interrupt request flag bit (WPCR:WTIF) is set to "1" in any mode except the stop mode in which the subclock mode or the sub-CR clock mode is used. In other words, a watch interrupt request is generated at every selected interval time, based on the time when the counter was last cleared.

### ■ Clearing Watch Prescaler

If the watch prescaler is cleared, other peripheral functions that are using the watch prescaler output are affected by changes in count time and by other factors.

When clearing the counter using the watch prescaler clear bit (WPCR:WCLR), modify the settings of other peripheral functions so that clearing the counter does not have any unexpected effect on them.

When the output of the watch prescaler is selected as the count clock, clearing the watch prescaler also clears the watchdog timer.

The watch prescaler is cleared not only by the watch prescaler clear bit (WPCR:WCLR) but also when the subclock or the sub-CR clock is stopped and the oscillation stabilization wait time is necessary. The watch prescaler is cleared in the following situations:

- The device transits from the subclock mode or sub-CR clock mode to the stop mode.
- The subclock oscillation enable bit or the sub-CR clock oscillation enable bit in the system clock control register 2 (SYCC2:SOSCE or SCRE) is set to "0" in main clock mode, main CR clock mode, or main CR PLL clock mode.

In addition, the counter of the watch prescaler is cleared and stops operating when a reset is generated.

### ■ Input Clock Selection for Watch Prescaler

Below are the clocks selected as input clocks of the watch prescaler in different clock modes.

- In main clock mode, main CR clock mode, and main CR PLL clock mode

When the subclock oscillation is enabled and the subclock oscillation stabilization wait time elapses, the subclock is selected as the input clock of the watch prescaler.

When the sub-CR clock oscillation is enabled and the sub-CR clock oscillation stabilization wait time elapses, the sub-CR clock is selected as the input clock of the watch prescaler.

When the subclock oscillation and the sub-CR clock oscillation are enabled, and the oscillation stabilization wait time elapses, the subclock is selected as the input clock of the watch prescaler.

- In subclock mode

Only the subclock is used as the input clock of the watch prescaler.

- In sub-CR clock mode  
Only the sub-CR clock is used as the input clock of the watch prescaler.

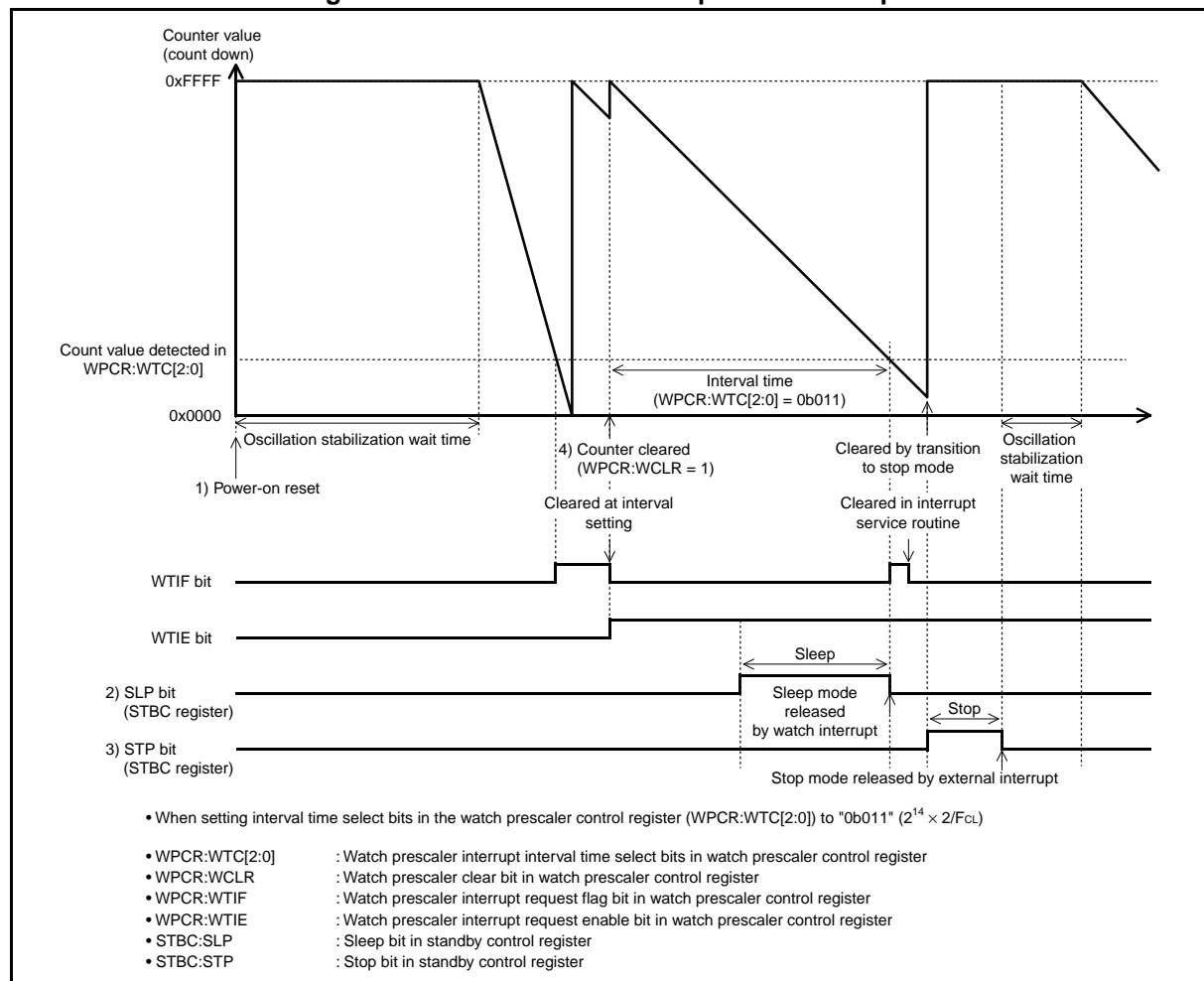
■ Operation Example of Watch Prescaler

Figure 9.4-1 shows an operation example under the following conditions:

1. When a power-on reset occurs
2. When the device transits to the sleep mode during the operation of the interval timer function in subclock mode or sub-CR clock mode
3. When the device transits to the stop mode during the operation of the interval timer function in subclock mode or sub-CR clock mode
4. When a request for clearing the counter is issued

The same operation is performed when changing to the watch mode as for when changing to the sleep mode.

Figure 9.4-1 Watch Prescaler Operation Example



**■ Setting Procedure Example**

Below is an example of procedure for setting the watch prescaler.

● Initial settings

1. Set the interrupt level. (ILR\*)
2. Set the interval time. (WPCR:WTC[2:0])
3. Enable interrupts and clear the interrupt request flag. (WPCR:WTIE = 1, WPCR:WTIF = 0)
4. Clear the counter. (WPCR:WCLR = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

● Processing interrupts

1. Clear the interrupt request flag. (WPCR:WTIF = 0)
2. Clear the counter. (WPCR:WCLR = 1)

**MB95850K/860K/870K Series****9.5 Register**

---

**This section describes the register of the watch prescaler.**

---

**Table 9.5-1 List of Watch Prescaler Register**

| Register abbreviation | Register name                    | Reference |
|-----------------------|----------------------------------|-----------|
| WPCR                  | Watch prescaler control register | 9.5.1     |

## 9.5.1 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 the status of the watch prescaler.

### ■ Register Configuration

| bit           | 7    | 6    | 5 | 4 | 3    | 2    | 1    | 0    |
|---------------|------|------|---|---|------|------|------|------|
| Field         | WTIF | WTIE | — | — | WTC2 | WTC1 | WTC0 | WCLR |
| Attribute     | R/W  | R/W  | — | — | R/W  | R/W  | R/W  | W    |
| Initial value | 0    | 0    | 0 | 0 | 0    | 0    | 0    | 0    |

### ■ Register Functions

[bit7] WTIF: Watch prescaler interrupt request flag bit

This bit is set to "1" when the interval time selected by the watch prescaler has elapsed.

When this bit and the watch prescaler interrupt request enable bit (WTIE) are set to "1", a watch prescaler interrupt request is output.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit7               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the interval time has not elapsed. |
| <b>Reading "1"</b> | Indicates that the interval time has elapsed.     |
| <b>Writing "0"</b> | Clears this bit.                                  |
| <b>Writing "1"</b> | Has no effect on operation.                       |

[bit6] WTIE: Watch prescaler interrupt request enable bit

This bit enables or disables output of interrupt requests to interrupt controller.

When this bit and the watch prescaler interrupt request flag bit (WTIF) are set to "1", a watch prescaler interrupt request is output.

| bit6               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the watch prescaler interrupt request. |
| <b>Writing "1"</b> | Enables the watch prescaler interrupt request.  |

[bit5:4] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit3:1] WTC[2:0]: Watch prescaler interrupt interval time select bits

These bits select the interval time.

| bit3:1        | Details   |   |
|---------------|---|---|
|               | Interval time<br>(Subclock, $F_{CL} = 32.768 \text{ kHz}$ ) | Interval time<br>(Sub-CR clock, $F_{CRL} = 100 \text{ kHz}$ ) |
| Writing "100" | $2^{10} \times 2/F_{CL}$ (62.5 ms)                          | $2^{10} \times 2/F_{CRL}$ (20.48 ms)                          |
| Writing "000" | $2^{11} \times 2/F_{CL}$ (125 ms)                           | $2^{11} \times 2/F_{CRL}$ (40.96 ms)                          |
| Writing "001" | $2^{12} \times 2/F_{CL}$ (250 ms)                           | $2^{12} \times 2/F_{CRL}$ (81.92 ms)                          |
| Writing "010" | $2^{13} \times 2/F_{CL}$ (500 ms)                           | $2^{13} \times 2/F_{CRL}$ (163.84 ms)                         |
| Writing "011" | $2^{14} \times 2/F_{CL}$ (1 s)                              | $2^{14} \times 2/F_{CRL}$ (327.68 ms)                         |
| Writing "101" | $2^{15} \times 2/F_{CL}$ (2 s)                              | $2^{15} \times 2/F_{CRL}$ (655.36 ms)                         |
| Writing "110" | $2^{16} \times 2/F_{CL}$ (4 s)                              | $2^{16} \times 2/F_{CRL}$ (1.311 s)                           |
| Writing "111" | $2^{17} \times 2/F_{CL}$ (8 s)                              | $2^{17} \times 2/F_{CRL}$ (2.621 s)                           |

[bit0] WCLR: Watch prescaler clear bit

This bit clears all bits in the counter of the watch prescaler to "1".

| bit0        | Details   |
|-------------|---|
| Read access | The read value is always "0".                                 |
| Writing "0" | Has no effect on operation.                                   |
| Writing "1" | Clears all bits in the counter of the watch prescaler to "1". |

Note: When the output of the watch prescaler is selected as the count clock of the software watchdog timer, clearing the watch prescaler with this bit also clears the software watchdog timer.

## 9.6 Notes on Using Watch Prescaler

---

This section provides notes on using the watch prescaler.

---

### ■ Notes on Using Watch Prescaler

- When setting interrupt processing in a program

The watch prescaler cannot be waken up from interrupt processing if the watch prescaler interrupt request flag bit (WPCR:WTIF) is set to "1" and the interrupt request is enabled (WPCR:WTIE = 1). Always clear the WTIF bit in the interrupt routine.

- Clearing the watch prescaler

When the watch prescaler is selected as the count clock of the software watchdog timer (WDTC:CS[1:0], CSP = 0b100 or 0b110), clearing the watch prescaler also clears the software watchdog timer.

- Watch prescaler interrupts

In stop mode in which the main clock, the main CR clock, or the main CR PLL clock is used, the watch prescaler performs counting, and can generate the watch prescaler interrupt.

- Peripheral functions receiving clock from the watch prescaler

If the counter of the watch prescaler is cleared when the output of the watch prescaler is used in other peripheral functions, the operations of such peripheral functions may be affected such as the changing of their operating cycles.

After the counter of the watch prescaler is cleared, the clock for the software watchdog timer output from the watch prescaler returns to the initial state. However, since the software watchdog timer counter is also cleared at the same time as the clock for the software watchdog timer returns to the initial state, the software watchdog timer operates in its normal cycle.



# **CHAPTER 10**

---

# **WATCH COUNTER**

**This chapter describes the functions and operations of the watch counter.**

- 10.1 Overview
- 10.2 Configuration
- 10.3 Interrupt
- 10.4 Operations and Setting Procedure Example
- 10.5 Registers
- 10.6 Notes on Using Watch Counter

## 10.1 Overview

---

**The watch counter generates interrupt requests at intervals from 40.96 ms (min.) to 63 s (max.).**

---

### ■ Function of Watch Counter

The watch counter performs counting for the number of times specified in the watch counter data register (WCDR) by using a selected count clock and generates an interrupt request. The count clock can be selected from eight types shown in Table 10.1-1. The count value can be set to any number from "0" to "63". No interrupt request is generated when "0" is selected as the count value.

When the count clock is set to 1 s and the count value "60", an interrupt request is generated every one minute.

**Table 10.1-1 Count Clock Types**

|        | Count clock (Sub-CR clock)<br>( $2^n \times 2/F_{CRL}^{*1}$ ) | Count clock (Subclock)<br>( $2^n \times 2/F_{CL}^{*2}$ ) |
|--------|---|--|
| n = 11 | 40.96 ms  | 125 ms   |
| n = 12 | 81.92 ms  | 250 ms   |
| n = 13 | 163.84 ms   | 500 ms   |
| n = 14 | 327.68 ms   | 1 s  |

\*1:  $2/F_{CRL} = 20 \mu\text{s}$  when  $F_{CRL} = 100 \text{ kHz}$

\*2:  $2/F_{CL} = 61.035 \mu\text{s}$  when  $F_{CL} = 32.768 \text{ kHz}$

---

Note:

Refer to the device data sheet for the accuracy of the sub-CR clock frequency.

---

# MB95850K/860K/870K Series

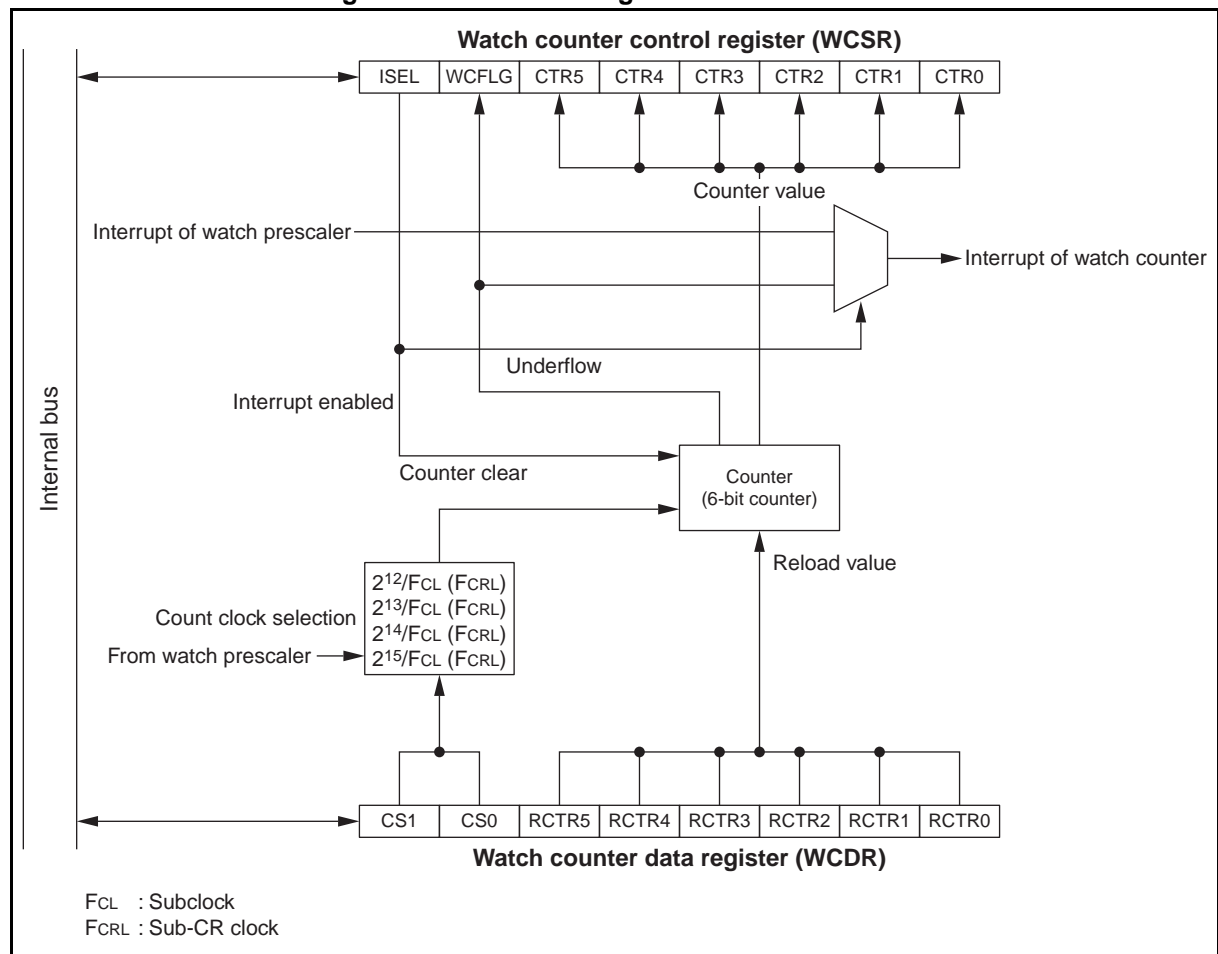
## 10.2 Configuration

The watch counter consists of the following blocks:

- Counter
- Watch counter control register (WCSR)
- Watch counter data register (WCDR)

### ■ Block Diagram of Watch Counter

Figure 10.2-1 Block Diagram of Watch Counter



- Counter

This is a 6-bit downcounter using the output clock of the watch prescaler as its count clock.

- Watch counter control register (WCSR)

This register controls interrupts, checks interrupt status and reads the count value.

- Watch counter data register (WCDR)

This register selects the count clock and sets the counter reload value.

- **Input Clock**

The watch counter uses the output clock of the watch prescaler as its input clock (count clock).

# MB95850K/860K/870K Series

## 10.3 Interrupt

The watch counter outputs an interrupt request when the counter underflows (counter value = 0b000001).

### ■ Watch Counter Interrupt

When the counter of the watch counter underflows, the watch counter interrupt request flag bit (WCFLG) in the watch counter control register (WCSR) is set to "1". Provided that the watch counter start and interrupt request enable bit (ISEL) in the WCSR register has also been set to "1", the watch counter outputs an interrupt request to the interrupt controller.

Table 10.3-1 shows the interrupt control bits and interrupt sources of the watch counter.

**Table 10.3-1 Interrupt Control Bits and Interrupt Sources of Watch Counter**

| Item                         | Details           |
|------------------------------|-------------------|
| Interrupt request flag bit   | WCSR:WCFLG        |
| Interrupt request enable bit | WCSR:ISEL         |
| Interrupt source             | Counter underflow |

## 10.4 Operations and Setting Procedure Example

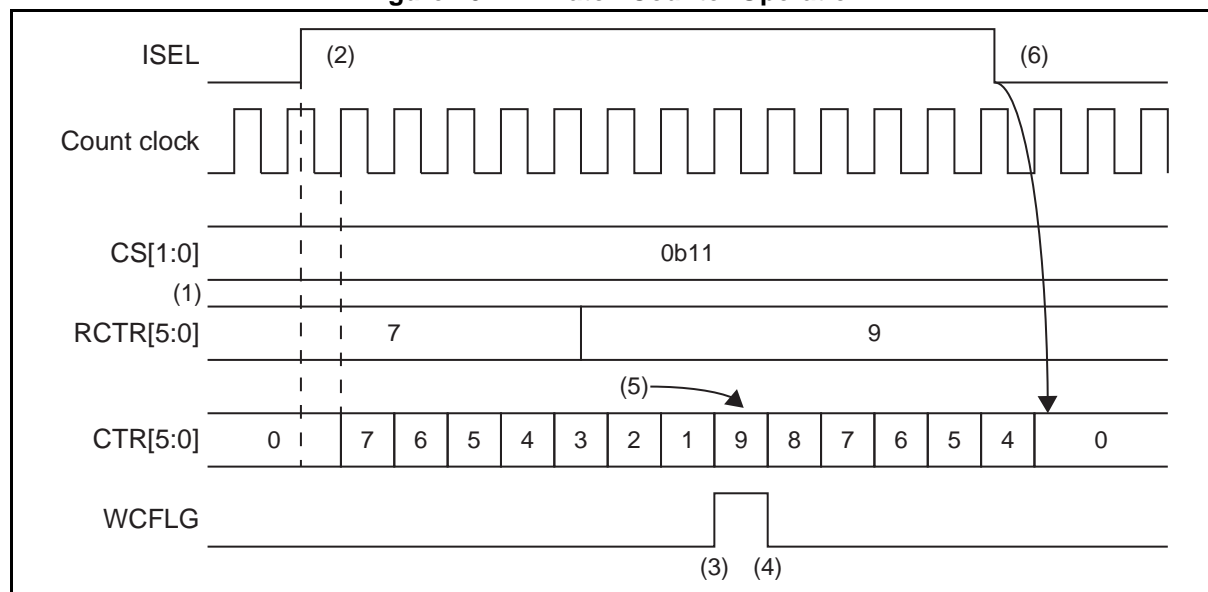
With the ISEL bit in the WCSR register set to "1", the watch counter counts down for the number of times specified as a count value in the RCTR[5:0] bits in the WCDR register with the count clock selected in the CS[1:0] bits in the WCDR register. When the counter underflows, the WCFLG bit in the WCSR register is set to "1", and the watch counter generates an interrupt request.

### ■ Setup Procedure of Watch Counter

Below is the procedure for setting up the watch counter.

1. Select the count clock (CS[1:0]) and set the counter reload value (RCTR[5:0]).
2. Set the ISEL bit to "1" to start a down count and enable interrupts. In addition, disable interrupts of the watch prescaler.  
The watch counter performs counting by using a divided clock (asynchronous) from the watch prescaler. An error of up to one count clock may occur at the beginning of a count cycle, depending on the time at which the ISEL bit is set to "1".
3. When the counter underflows, the WCFLG bit is set to "1", causing the watch counter to generate an interrupt request.
4. Write "0" to the WCFLG bit to clear it.
5. When the RCTR[5:0] bits are modified during counting, the reload value is updated at a reload occurring after the counter is set to "1".
6. When "0" is written to the ISEL bit, the counter becomes "0" and stops operating.

**Figure 10.4-1 Watch Counter Operation**



**Note:**

Before restarting the counter by setting the ISEL bit to "0" stop the counter, read the CTR[5:0] bits in the WCSR register twice and ensure that the CTR[5:0] bits have been cleared to "0b000000".

**■ Operation in Substop Mode and Sub-CR Clock Stop Mode**

When the device enters substop mode or sub-CR clock stop mode, the watch counter stops the count operation and the watch prescaler is also cleared. Therefore, the watch counter cannot count the correct value after the device exits substop mode or sub-CR clock stop mode. After the device exits substop mode or sub-CR clock stop mode, always write "0" to the ISEL bit in the WCSR register to clear the counter. In any other standby mode except the substop mode and sub-CR clock stop mode, the watch counter keeps operating.

**■ Operation in Main Stop Mode and Main CR Clock Stop Mode**

In main stop mode or main CR clock stop mode, though the watch counter continues the count operation, no interrupt is generated. The watch counter stops when the subclock oscillation enable bit (SOSCE) and sub-CR clock oscillation enable bit (SCRE) in the system clock control register 2 (SYCC2) are both set to "0".

**■ Setting Procedure Example**

Below is an example of procedure for setting the watch counter.

## ● Initial settings

1. Set the interrupt level. (ILR\*)
2. Select the count clock. (WCDR:CS[1:0])
3. Set the counter reload value. (WCDR:RCTR[5:0])
4. Activate the watch counter and enable interrupts. (WCSR:ISEL = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

## ● Interrupt processing

1. Clear the interrupt request flag. (WCSR:WCFLG = 0)
2. Process any interrupt.

## 10.5 Registers

---

**This section describes the registers of the watch counter.**

---

**Table 10.5-1 List of Watch Counter Registers**

| Register abbreviation | Register name                  | Reference |
|-----------------------|--------------------------------|-----------|
| WCDR                  | Watch counter data register    | 10.5.1    |
| WCSR                  | Watch counter control register | 10.5.2    |



## MB95850K/860K/870K Series

### 10.5.1 Watch Counter Data Register (WCDR)

The watch counter data register (WCDR) selects the count clock and sets the counter reload value.

#### ■ Register Configuration

| bit           | 7   | 6   | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-----|-----|-------|-------|-------|-------|-------|-------|
| Field         | CS1 | CS0 | RCTR5 | RCTR4 | RCTR3 | RCTR2 | RCTR1 | RCTR0 |
| Attribute     | R/W | R/W | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0   | 0   | 1     | 1     | 1     | 1     | 1     | 1     |

#### ■ Register Functions

[bit7:6] CS[1:0]: Watch counter count clock select bits

These bits select the count clock of the watch counter.

Ensure that the ISEL bit in the WCSR register has been set to "0" before modifying these bits.

| bit7:6       | Details   |   |
|--------------|---|---|
|              | Count clock<br>(Subclock, $F_{CL} = 32.768 \text{ kHz}$ ) | Count clock<br>(Sub-CR clock, $F_{CRL} = 100 \text{ kHz}$ ) |
| Writing "00" | $2^{12}/F_{CL}$ (125 ms)                                  | $2^{12}/F_{CRL}$ (40.96 ms)                                 |
| Writing "01" | $2^{13}/F_{CL}$ (250 ms)                                  | $2^{13}/F_{CRL}$ (81.92 ms)                                 |
| Writing "10" | $2^{14}/F_{CL}$ (500 ms)                                  | $2^{14}/F_{CRL}$ (163.84 ms)                                |
| Writing "11" | $2^{15}/F_{CL}$ (1 s)                                     | $2^{15}/F_{CRL}$ (327.68 ms)                                |

[bit5:0] RCTR[5:0]: Watch counter counter reload value setting bits

These bits set the counter reload value.

When the counter reload value is modified during the count operation, the new value becomes effective at a reload after the counter underflows.

When the RCTR[5:0] bits are set to "0", no interrupt request is to be generated.

When the counter reload value is modified at the same time as an interrupt request is generated (WCSR:WCFLR = 1), the value to be reloaded is not correct. Therefore, modifying the counter reload value should be done before an interrupt request is generated, i.e. during the interrupt service routine or after the stopping of the watch counter (WCSR:ISEL = 0).

## 10.5.2 Watch Counter Control Register (WCSR)

The watch counter control register (WCSR) controls the watch counter operation and the interrupt of the watch counter, and reads the counter value.

### Register Configuration

| bit           | 7    | 6     | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|-------|------|------|------|------|------|------|
| Field         | ISEL | WCFLG | CTR5 | CTR4 | CTR3 | CTR2 | CTR1 | CTR0 |
| Attribute     | R/W  | R/W   | R    | R    | R    | R    | R    | R    |
| Initial value | 0    | 0     | 0    | 0    | 0    | 0    | 0    | 0    |

### Register Functions

#### [bit7] ISEL: Watch counter start and interrupt request enable bit

This bit starts the watch counter, and enables or disables the interrupt request of the watch counter and that of the watch prescaler.

Always disable the interrupt request of the watch prescaler before setting this bit to "1" to enable the interrupt request of the watch counter.

The watch counter counts with the asynchronous clock from the watch prescaler. Therefore, an error of up to one count clock may occur at the beginning of counting, depending on the time of setting the ISEL bit to "1".

| bit7               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the watch counter and disables the interrupt request of the watch counter (enables the interrupt request of the watch prescaler).  |
| <b>Writing "1"</b> | Starts the watch counter and enables the interrupt request of the watch counter (disables the interrupt request of the watch prescaler). |

#### [bit6] WCFLG: Watch counter interrupt request flag bit

This bit is set to "1" when the counter underflows.

When this bit and the ISEL bit are both set to "1", a watch counter interrupt request is generated.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit6               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no watch counter interrupt request has been generated. |
| <b>Reading "1"</b> | Indicates that a watch counter interrupt request has been generated.  |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.   |

#### [bit5:0] CTR[5:0]: Watch counter counter read bits

These bits read the value of the counter that is counting.

While the counter value is being changed, the counter value that these bits read may not be correct. Therefore, before using the counter value these bits read, ensure that the counter reads the same value in times of reading by reading these bits twice.

Writing values to these bits has no effect on operation.

## 10.6 Notes on Using Watch Counter

---

**This section provides notes on using the watch counter**

---

- When the watch prescaler is cleared while the watch counter is running, the watch counter may not be able to run normally. Before clearing the watch prescaler, write "0" to the ISEL bit in the WCSR register to stop the watch counter.
- Before writing "1" to the ISEL bit to restart the watch counter after stopping it, read the CTR[5:0] bits in the WCSR register twice to ensure that the CTR[5:0] bits have been cleared to "0b000000".



# ***CHAPTER 11***

---

# ***WILD REGISTER***

# ***FUNCTION***

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

- 11.1 Overview
- 11.2 Configuration
- 11.3 Operations
- 11.4 Registers
- 11.5 Typical Hardware Connection Example

## **11.1 Overview**

---

**The wild register function can be used to patch bugs in a program with addresses and modification data, both of which are to be set in built-in registers. This section describes the wild register function.**

---

### **■ Wild Register Function**

The wild register consists of three wild register data setting registers, three wild register address setting registers, a 1-byte address compare enable register and a 1-byte wild register data test setting register. If addresses and data that are to be modified are set to these registers, ROM data can be replaced with modification data set in the registers. Data of up to three different addresses can be modified.

The wild register function can be used to debug a program after creating the mask and to patch bugs in the program.

# MB95850K/860K/870K Series

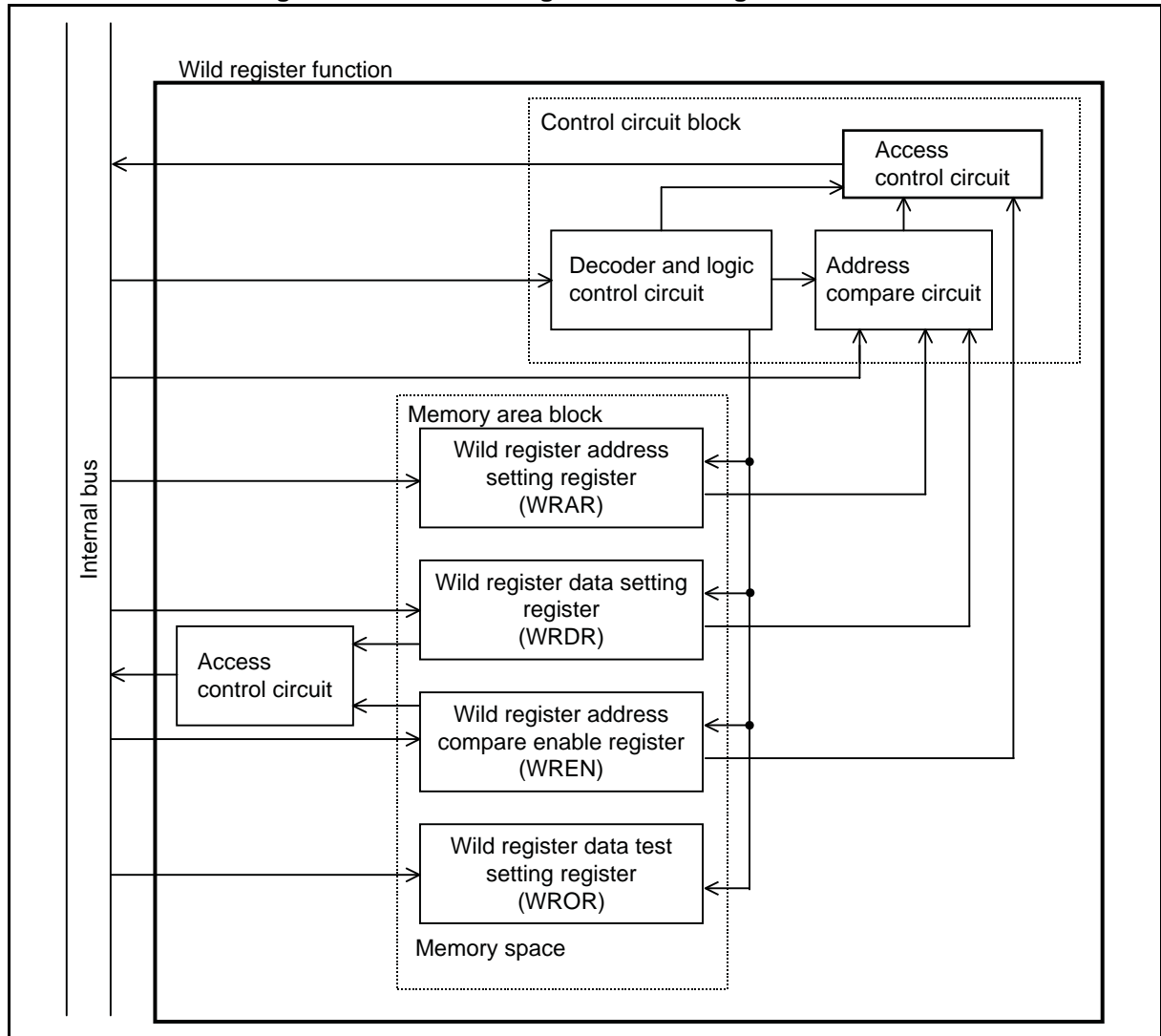
## 11.2 Configuration

The block diagram of the wild register is shown below. The wild register consists of the following blocks:

- Memory area block
  - Wild register data setting register (WRDR)
  - Wild register address setting register (WRAR)
  - Wild register address compare enable register (WREN)
  - Wild register data test setting register (WROR)
- Control circuit block

### ■ Block Diagram of Wild Register Function

Figure 11.2-1 Block Diagram of Wild Register Function



- Memory area block

The memory area block consists of the wild register data setting registers (WRDR), wild register address setting registers (WRAR), wild register address compare enable register (WREN) and wild register data test setting register (WROR). The wild register function is used to specify the addresses and data that need to be replaced. The wild register address compare enable register (WREN) enables the wild register function for each wild register data setting register (WRDR). In addition, the wild register data test setting register (WROR) enables the normal read function for each wild register data setting register (WRDR).

- Control circuit block

This circuit compares the actual address data with addresses set in the wild register address setting registers (WRAR). If they match, the circuit outputs the data from the wild register data setting register (WRDR) to the data bus. The operation of the control circuit block is controlled by the wild register address compare enable register (WREN).



# MB95850K/860K/870K Series

## 11.3 Operations

This section describes the procedure for setting the wild register function.

### ■ Procedure for Setting Wild Register Function

Prepare a program that can read the value to be set in the wild register from external memory (e.g. EEPROM or FRAM) in the user program before using the wild register function. The setting method for the wild register is shown below.

This section does not include information on the method of communications between the external memory and the device.

- Write the address of the built-in ROM code that will be modified to the wild register address setting register (WRAR0 to WRAR2).
- Write a new code to the wild register data setting register (WRDR0 to WRDR2) corresponding to the wild register address setting register to which the address has been written.
- Write "1" to the EN bit in the wild register address compare enable register (WREN) corresponding to the wild register number to enable the wild register function represented by that wild register number.

Table 11.3-1 shows the procedure for setting the registers of the wild register function.

**Table 11.3-1 Procedure for Setting Registers of Wild Register Function**

| Step | Operation   | Operation example  |
|------|---|--|
| 1    | Read replacement data from a peripheral function outside through a certain communication method.  | Suppose the built-in ROM code to be modified is at the address 0xF011 and the data to be modified is "0xB5", and there are three built-in ROM codes to be modified.  |
| 2    | Write the replacement address to a wild register address setting register (WRAR0 to WRAR2).   | Set wild register address setting registers (WRAR0 = 0xF011, WRAR1 = ..., WRAR2 = ...).  |
| 3    | Write a new ROM code (replacement for the built-in ROM code) to a wild register data setting register (WRDR0 to WRDR2).                                     | Set the wild register data setting registers (WRDR0 = 0xB5, WRDR1 = ..., WRDR2 = ...).   |
| 4    | Enable the EN bit in the wild register address compare enable register (WREN) corresponding to the wild register number of the wild register function used. | Setting bit0 of the address compare enable register (WREN) to "1" enables the wild register function of the wild register number 0. If the address matches the value set in the wild register address setting register (WRAR), the value of the wild register data setting register (WRDR) will be replaced with the built-in ROM code. When replacing more than one built-in ROM code, enable the related EN bits in the wild register address compare enable register (WREN) corresponding to respective built-in ROM codes. |

### ■ Wild Register Function Applicable Addresses

The wild register function can be applied to all address space except the address "0x0078".

Since the address "0x0078" is used as a mirror address for the register bank pointer and the direct bank pointer, this address cannot be patched.

## 11.4 Registers

This section describes the registers of the wild register function.

**Table 11.4-1 List of Hardware/software Watchdog Timer Registers**

| Register abbreviation | Register name                                 | Reference |
|-----------------------|---|-----------|
| WRDR0                 | Wild register data setting register 0         | 11.4.1    |
| WRDR1                 | Wild register data setting register 1         | 11.4.1    |
| WRDR2                 | Wild register data setting register 2         | 11.4.1    |
| WRAR0                 | Wild register address setting register 0      | 11.4.2    |
| WRAR1                 | Wild register address setting register 1      | 11.4.2    |
| WRAR2                 | Wild register address setting register 2      | 11.4.2    |
| WREN                  | Wild register address compare enable register | 11.4.3    |
| WROR                  | Wild register data test setting register      | 11.4.4    |

### ■ Wild Register Number

A wild register number is assigned to each wild register address setting register (WRAR) and each wild register data setting register (WRDR).

**Table 11.4-2 Wild Register Numbers Corresponding to Wild Register Address Setting Registers and Wild Register Data Setting Registers**

| Wild register number | Wild register address setting register (WRAR) | Wild register data setting register (WRDR) |
|----------------------|---|--|
| 0                    | WRAR0   | WRDR0                                      |
| 1                    | WRAR1   | WRDR1                                      |
| 2                    | WRAR2   | WRDR2                                      |

## MB95850K/860K/870K Series

### 11.4.1 Wild Register Data Setting Registers (WRDR0 to WRDR2)

The wild register data setting registers (WRDR0 to WRDR2) use the wild register function to specify the data to be modified.

#### ■ Register Configuration

WRDR0

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

WRDR1

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

WRDR2

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### ■ Register Functions

[bit7:0] RD[7:0]: Wild register data setting bits

These bits specify the data to be modified by the wild register function.

These bits are used to set the modification data at the address assigned by the wild register address setting register (WRAR). Data is valid at an address corresponding to one of the wild register numbers.

The read access to one of these bits is enabled only when the data test setting bit in the wild register data test setting register (WROR) corresponding to the bit to be read is set to "1".

## 11.4.2 Wild Register Address Setting Registers (WRAR0 to WRAR2)

The wild register address setting registers (WRAR0 to WRAR2) set the address to be modified by the wild register function.

### ■ Register Configuration

#### WRAR0

|               |      |      |      |      |      |      |     |     |
|---------------|------|------|------|------|------|------|-----|-----|
| bit           | 15   | 14   | 13   | 12   | 11   | 10   | 9   | 8   |
| Field         | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA9 | RA8 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W | R/W |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0   | 0   |

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### WRAR1

|               |      |      |      |      |      |      |     |     |
|---------------|------|------|------|------|------|------|-----|-----|
| bit           | 15   | 14   | 13   | 12   | 11   | 10   | 9   | 8   |
| Field         | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA9 | RA8 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W | R/W |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0   | 0   |

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### WRAR2

|               |      |      |      |      |      |      |     |     |
|---------------|------|------|------|------|------|------|-----|-----|
| bit           | 15   | 14   | 13   | 12   | 11   | 10   | 9   | 8   |
| Field         | RA15 | RA14 | RA13 | RA12 | RA11 | RA10 | RA9 | RA8 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W | R/W |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0   | 0   |

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

### ■ Register Functions

[bit15:0] RA[15:0]: Wild register address setting bits

These bits set the address to be modified by the wild register function.

The address to be assigned to modification data is set to these bits. The address is to be specified according to the wild register number corresponding to a wild register address setting register.

**MB95850K/860K/870K Series****11.4.3 Wild Register Address Compare Enable Register (WREN)**

The wild register address compare enable register (WREN) enables or disables the operations of wild register functions using their respective wild register numbers.

**Register Configuration**

| bit           | 7 | 6 | 5        | 4        | 3        | 2   | 1   | 0   |
|---------------|---|---|----------|----------|----------|-----|-----|-----|
| Field         | — | — | Reserved | Reserved | Reserved | EN2 | EN1 | EN0 |
| Attribute     | — | — | W        | W        | W        | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0        | 0        | 0        | 0   | 0   | 0   |

**Register Functions**[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] Reserved bits

Always set these bits to "0".

[bit2:0] EN[2:0]: Wild register address compare enable bits

These bits enable or disable the operation of the wild register.

- EN0 corresponds to wild register number 0.
- EN1 corresponds to wild register number 1.
- EN2 corresponds to wild register number 2.

| bit2/bit1/bit0     | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the operation of the wild register function. |
| <b>Writing "1"</b> | Enables the operation of the wild register function.  |

## 11.4.4 Wild Register Data Test Setting Register (WROR)

The wild register data test setting register (WROR) enables or disables data reading from the corresponding wild register data setting register (WRDR0 to WRDR2).

### ■ Register Configuration

| bit           | 7 | 6 | 5        | 4        | 3        | 2    | 1    | 0    |
|---------------|---|---|----------|----------|----------|------|------|------|
| Field         | — | — | Reserved | Reserved | Reserved | DRR2 | DRR1 | DRR0 |
| Attribute     | — | — | W        | W        | W        | R/W  | R/W  | R/W  |
| Initial value | 0 | 0 | 0        | 0        | 0        | 0    | 0    | 0    |

### ■ Register Functions

#### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit5:3] Reserved bits

Always set these bits to "0".

#### [bit2:0] DRR[2:0]: Wild register data test setting bits

These bits enable or disable the normal reading from the corresponding data setting register of the wild register.

- DRR0 corresponds to wild register number 0.
- DRR1 corresponds to wild register number 1.
- DRR2 corresponds to wild register number 2.

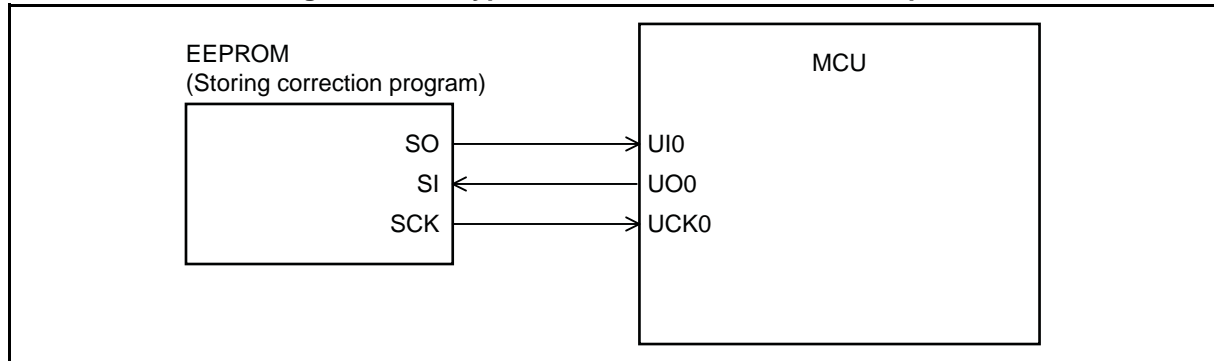
| bit2/bit1/bit0     | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables reading from the wild register data setting register. |
| <b>Writing "1"</b> | Enables reading from the wild register data setting register.  |

## 11.5 Typical Hardware Connection Example

Below is an example of typical hardware connection for the application of the wild register function.

### ■ Hardware Connection Example

Figure 11.5-1 Typical Hardware Connection Example







# CHAPTER 12

---

## 8/16-BIT COMPOSITE TIMER

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

- 12.1 Overview
- 12.2 Configuration
- 12.3 Channel
- 12.4 Pins
- 12.5 Interrupts
- 12.6 Operation of Interval Timer Function (One-shot Mode)
- 12.7 Operation of Interval Timer Function (Continuous Mode)
- 12.8 Operation of Interval Timer Function (Free-run Mode)
- 12.9 Operation of PWM Timer Function (Fixed-cycle mode)
- 12.10 Operation of PWM Timer Function (Variable-cycle Mode)
- 12.11 Operation of PWC Timer Function
- 12.12 Operation of Input Capture Function
- 12.13 Operation of Noise Filter
- 12.14 Registers
- 12.15 Notes on Using 8/16-bit Composite Timer

## 12.1 Overview

---

The 8/16-bit composite timer consists of two 8-bit counters. It can be used as two 8-bit timers, or as a 16-bit timer if the two counters are connected in cascade.

The 8/16-bit composite timer has the following functions:

- Interval timer function
  - PWM timer function
  - PWC timer function (pulse width measurement)
  - Input capture function
- 

### ■ Interval Timer Function (One-shot Mode)

When the interval timer function (one-shot mode) is selected, the counter starts counting from "0x00" as the timer is started. When the counter value matches the value of the 8/16-bit composite timer data register, the timer output is inverted, an interrupt request occurs, and the counter stops counting.

### ■ Interval Timer Function (Continuous Mode)

When the interval timer function (continuous mode) is selected, the counter starts counting from "0x00" as the timer is started. When the counter value matches the value of the 8/16-bit composite timer data register, the timer output is inverted, an interrupt request occurs, and the counter counts from "0x00" again. The timer outputs square wave as a result of this repeated operation.

### ■ Interval Timer Function (Free-run Mode)

When the interval timer function (free-run mode) is selected, the counter starts counting from "0x00". When the counter value matches the value of the 8/16-bit composite timer data register, the timer output is inverted and an interrupt request occurs. Under these conditions, if the counter continues to count and reaches "0xFF", it restarts counting from "0x00". The timer outputs square wave as a result of this repeated operation.

### ■ PWM Timer Function (Fixed-cycle Mode)

When the PWM timer function (fixed-cycle mode) is selected, a PWM signal with a variable "H" pulse width is generated in fixed cycles. The cycle is fixed at "0xFF" in 8-bit operation or at "0xFFFF" in 16-bit operation. The time is determined by the count clock selected. The "H" pulse width is specified by setting a specific register.

### ■ PWM Timer Function (Variable-cycle Mode)

When the PWM timer function (variable-cycle mode) is selected, two 8-bit counters are used to generate an 8-bit PWM signal of variable cycle and duty depending on the cycle and "L" pulse width specified by registers.

In this operating mode, since the two 8-bit counters have to be used separately, the composite timer cannot operate as a 16-bit counter.

## MB95850K/860K/870K Series

### ■ PWC Timer Function

When the PWC timer function is selected, the width and cycle of an external input pulse can be measured.

In this operating mode, the counter starts counting from "0x00" immediately after a count start edge of an external input signal is detected. Afterward, when a count end edge is detected, the counter transfers its value to a register to generate an interrupt.

### ■ Input Capture Function

When the input capture function is selected, the counter value is stored in a register immediately after the detection of an edge of an external input signal.

This function is available in either free-run mode or clear mode for count operation.

In clear mode, the counter starts counting from "0x00", and transfers its value to a register to generate an interrupt after an edge is detected. Afterward, the counter restarts counting from "0x00".

In free-run mode, the counter transfers its value to a register to generate an interrupt immediately after the detection of an edge. Afterward, unlike in clear mode, the counter continues to count without being cleared to "0x00".

## **12.2 Configuration**

---

The 8/16-bit composite timer consists of the following blocks:

- 8-bit counter
  - 8-bit comparator (including a temporary latch)
  - 8/16-bit composite timer data register (Tn0DR/Tn1DR)
  - 8/16-bit composite timer status control register 0 (Tn0CR0/Tn1CR0)
  - 8/16-bit composite timer status control register 1 (Tn0CR1/Tn1CR1)
  - 8/16-bit composite timer timer mode control register (TMCRn)
  - Output controller
  - Control logic
  - Count clock selector
  - Edge detector
  - Noise filter
- 

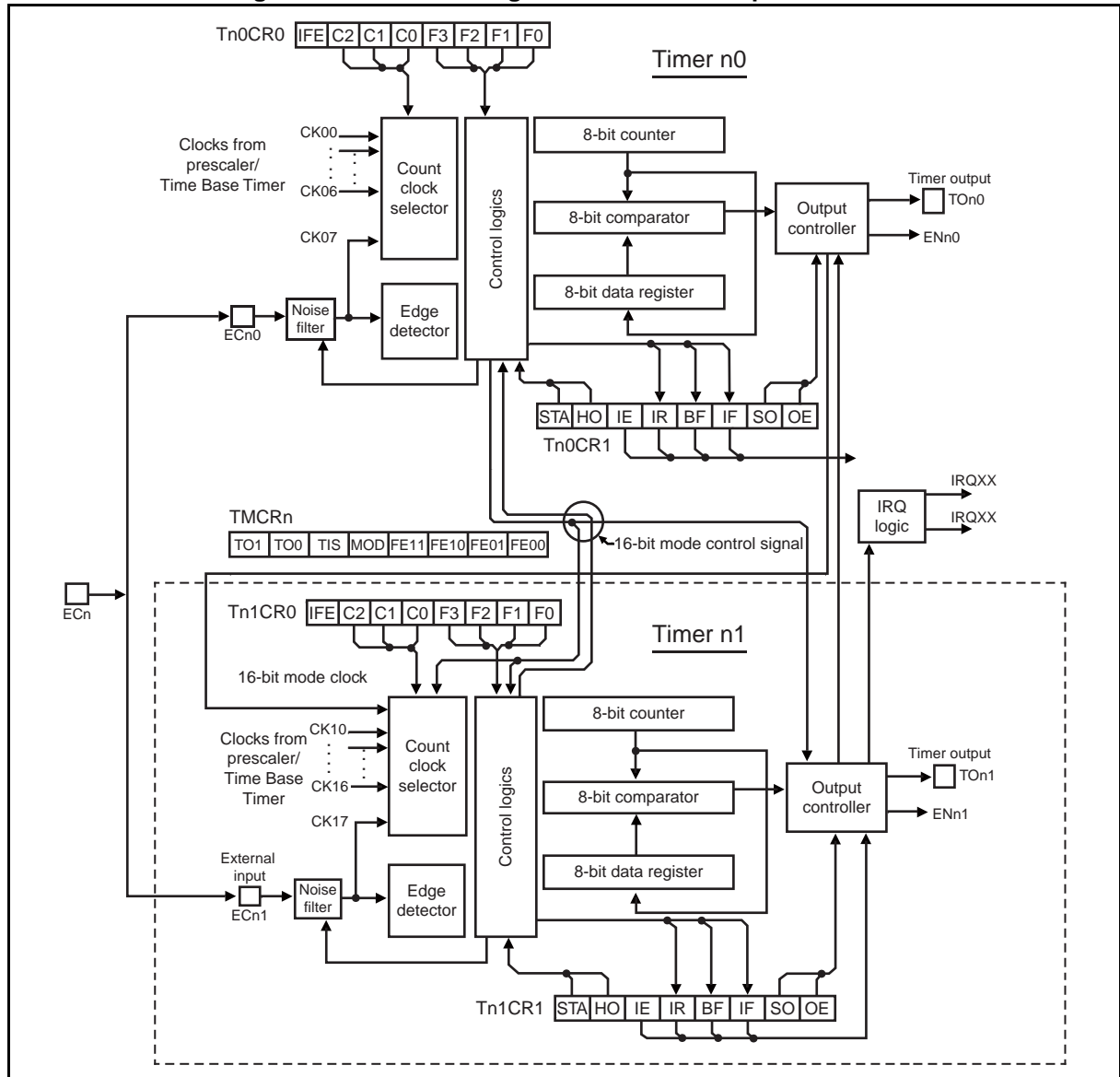
The number of pins and that of channels of the 8/16-bit composite timer vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

# MB95850K/860K/870K Series

## ■ Block Diagram of 8/16-bit Composite Timer

Figure 12.2-1 Block Diagram of 8/16-bit Composite Timer



- 8-bit counter

This counter serves as the basis for various timer operations. It can be used either as two 8-bit counters or as a 16-bit counter.

- 8-bit comparator

The comparator compares the value in the 8/16-bit composite timer data register and that in the counter. It incorporates a latch that temporarily stores the 8/16-bit composite timer data register value.

- 8/16-bit composite timer data register (Tn0DR/Tn1DR)

These registers are used to write the maximum value counted during interval timer operation or PWM timer operation and to read the count value during PWC timer operation or input capture operation.

- 8/16-bit composite timer status control register 0 (Tn0CR0/Tn1CR0)

These registers are used to select the timer operating mode and the count clock, and to enable or disable IF flag interrupts.

- 8/16-bit composite timer status control register 1 (Tn0CR1/Tn1CR1)

These registers are used to control interrupt flags, timer output, and timer operation.

- 8/16-bit composite timer timer mode control register (TMCRn)

This register is used to select the noise filter function, 8-bit or 16-bit operating mode, and signal input to timer n0/n1 and to indicate the timer output value.

- Output controller

The output controller controls timer output. The timer output is supplied to the external pin when the pin output has been enabled.

- Control logic

The control logic controls timer operation.

- Count clock selector

The selector selects the counter operating clock signal from different prescaler output signals (divided machine clock signal and time-base timer output signal).

- Edge detector

The edge detector selects the edge of an external input signal to be used as an event for PWC timer operation or input capture operation.

- Noise filter

This filter serves as a noise filter for external input signals. The filter function can be selected from "H" pulse noise elimination, "L" pulse noise elimination, and "H"/"L"-pulse noise elimination.

## ■ Input Clock

The 8/16-bit composite timer uses the output clock from the prescaler as its input clock (count clock).

## MB95850K/860K/870K Series

### 12.3 Channel

---

This section describes the channels of the 8/16-bit composite timer.

---

#### ■ Channel of 8/16-bit Composite Timer

On a channel, there are two 8-bit counters. They can be used as two 8-bit timers or one 16-bit timer. The following table lists the external pins on a channel.

**Table 12.3-1 External Pins of 8/16-bit Composite Timer**

| Pin name | Pin function                      |
|----------|-----------------------------------|
| TOn0     | Timer n0 output                   |
| TOn1     | Timer n1 output                   |
| ECn      | Timer n0 input and timer n1 input |

## 12.4 Pins

---

**This section describes the pins of the 8/16-bit composite timer.**

---

### ■ Pins of 8/16-bit Composite Timer

The external pins of the 8/16-bit composite timer are TOn0, TOn1 and ECn.

- TOn0 pin

**TOn0:**

This pin serves as the timer output pin for timer n0 in 8-bit operation or for timers n0 and n1 in 16-bit operation. When the output is enabled ( $Tn0CR1:OE = 1$ ) in the interval timer function, PWM timer function, or PWC timer function, this pin becomes an output pin automatically regardless of the setting of the port direction register (DDR) and functions as the timer output TOn0 pin.

The output becomes undetermined if output is enabled with the input capture function in use.

- TOn1 pin

**TOn1:**

This pin serves as the timer output pin for timer n1 in 8-bit operation. When the output is enabled ( $Tn1CR1:OE = 1$ ) in interval timer function, PWM timer function (fixed-cycle mode), or PWC timer function, the pin becomes an output pin automatically regardless of the port direction register (DDR) and functions as the timer output TOn1 pin.

In 16-bit operation, if output is enabled with the PWM timer function (variable-cycle mode) or input capture function in use, the output becomes undetermined.

- ECn pin

The ECn pin is connected to the ECn0 and ECn1 internal pins.

**ECn0 internal pin:**

This pin serves as the external count clock input pin for timer n0 when the interval timer function or PWM timer function is selected, or as the signal input pin for timer n0 when the PWC timer function or input capture function is selected. The pin cannot be set to serve as the external count clock input pin when the PWC timer function or input capture function is selected.

To use the input function mentioned above, set the bit in the port direction register corresponding to ECn pin to "0" to make the pin as an input port.

**ECn1 internal pin:**

This pin serves as the external count clock input pin for timer n1 when the interval timer function or PWM timer function is selected, or as the signal input pin for timer n1 when the PWC timer function or input capture function is selected. The pin cannot be set to serve as the external count clock input pin when the PWC timer function or input capture function is selected.

In 16-bit operation, the input function of this pin is not used. If the PWM timer function (variable-cycle mode) is selected, the input function of this pin can also be used.

To use the input function mentioned above, set the bit in the port direction register corresponding to ECn pin to "0" to make the pin as an input port.



## MB95850K/860K/870K Series

### 12.5 Interrupts

The 8/16-bit composite timer generates the following types of interrupts. An interrupt number and an interrupt vector are assigned to each type of interrupts.

- Timer n0 interrupt
- Timer n1 interrupt

#### ■ Timer n0 Interrupt

Table 12.5-1 shows the timer n0 interrupt and its sources.

**Table 12.5-1 Timer n0 Interrupt**

| Item                        | Description   |  |   |
|-----------------------------|---|--|---|
| Interrupt generating source | Comparison match in the interval timer operation or the PWM timer operation (variable-cycle mode) | Overflow in the PWC timer operation or the input capture operation | Completion of measurement in the PWC timer operation or edge detection in the input capture operation |
| Interrupt flag              | Tn0CR1:IF   | Tn0CR1:IF  | Tn0CR1:IR   |
| Interrupt enable            | Tn0CR1:IE and Tn0CR0:IFE  | Tn0CR1:IE and Tn0CR0:IFE   | Tn0CR1:IE   |

#### ■ Timer n1 Interrupt

Table 12.5-2 shows the timer n1 interrupt and its sources.

**Table 12.5-2 Timer n1 Interrupt**

| Item                        | Description   |  |   |
|-----------------------------|---|--|---|
| Interrupt generating source | Comparison match in the interval timer operation or the PWM timer operation (variable-cycle mode), except in 16-bit operation | Overflow in the PWC timer operation or the input capture operation, except in 16-bit operation | Completion of measurement in the PWC timer operation or edge detection in the input capture operation, except in 16-bit operation |
| Interrupt flag              | Tn1CR1:IF   | Tn1CR1:IF  | Tn1CR1:IR   |
| Interrupt enable            | Tn1CR1:IE and Tn1CR0:IFE  | Tn1CR1:IE and Tn1CR0:IFE   | Tn1CR1:IE   |

## 12.6 Operation of Interval Timer Function (One-shot Mode)

This section describes the operation of the interval timer function (one-shot mode) of the 8/16-bit composite timer.

### ■ Operation of Interval Timer Function (One-shot Mode)

To use the interval timer function (one-shot mode), do the settings shown in Figure 12.6-1.

Figure 12.6-1 Settings of Interval Timer Function (One-shot Mode)

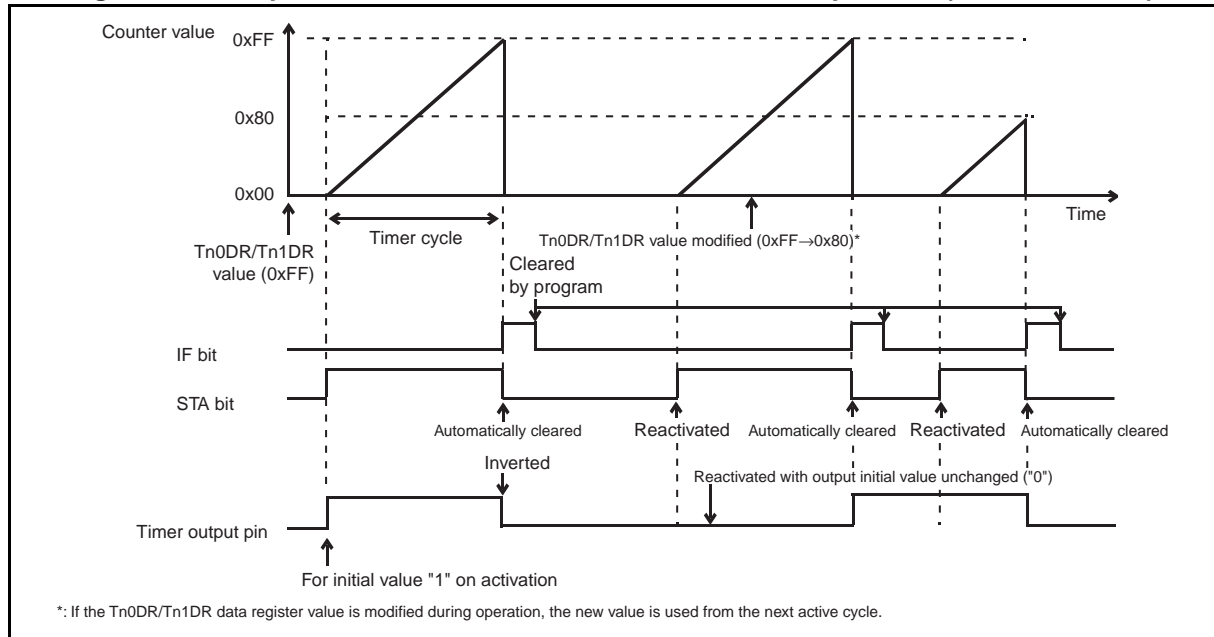
|               | bit7                                       | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|--|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE  | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○  | ○    | ○    | ○    | 0    | 0    | 0    | 0    |
| Tn0CR1/Tn1CR1 | STA  | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1  | ○    | ○    | x    | x    | ○    | ○    | ○    |
| TMCRn         | TO1  | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○  | ○    | x    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Sets interval time (counter compare value) |      |      |      |      |      |      |      |
|               | ○: Used bit                                |      |      |      |      |      |      |      |
|               | x: Unused bit                              |      |      |      |      |      |      |      |
|               | 1: Set to "1"                              |      |      |      |      |      |      |      |
|               | 0: Set to "0"                              |      |      |      |      |      |      |      |

As for the interval timer function (one-shot mode), enabling timer operation (Tn0CR1/Tn1CR1:STA = 1) causes the counter to start counting from "0x00" at the rising edge of a selected count clock signal. When the counter value matches the value of the 8/16-bit composite timer data register (Tn0DR/Tn1DR), the timer output (TMCRn:TO0/TO1) is inverted, the interrupt flag (Tn0CR1/Tn1CR1:IF) is set to "1", the timer operation enable bit (Tn0CR1/Tn1CR1:STA) is set to "0", and the counter stops counting.

The value of the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is transferred to the temporary storage latch (comparison data storage latch) in the comparator when the counter starts counting. Do not write "0x00" to the 8/16-bit composite timer data register.

Figure 12.6-2 shows the operation of the interval timer function in 8-bit operation.

Figure 12.6-2 Operation of Interval Timer Function in 8-bit Operation (One-shot Mode)



## 12.7 Operation of Interval Timer Function (Continuous Mode)

This section describes the interval timer function (continuous mode operation) of the 8/16-bit composite timer.

### ■ Operation of Interval Timer Function (Continuous Mode)

To use interval timer function (continuous mode), do the settings shown in Figure 12.7-1.

Figure 12.7-1 Settings for Interval Timer Function (Continuous Mode)

|               | bit7                                       | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|--|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE  | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○  | ○    | ○    | ○    | 0    | 0    | 0    | 1    |
| Tn0CR1/Tn1CR1 | STA  | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1  | ○    | ○    | x    | x    | ○    | ○    | ○    |
| TMCRn         | TO1  | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○  | ○    | x    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Sets interval time (counter compare value) |      |      |      |      |      |      |      |

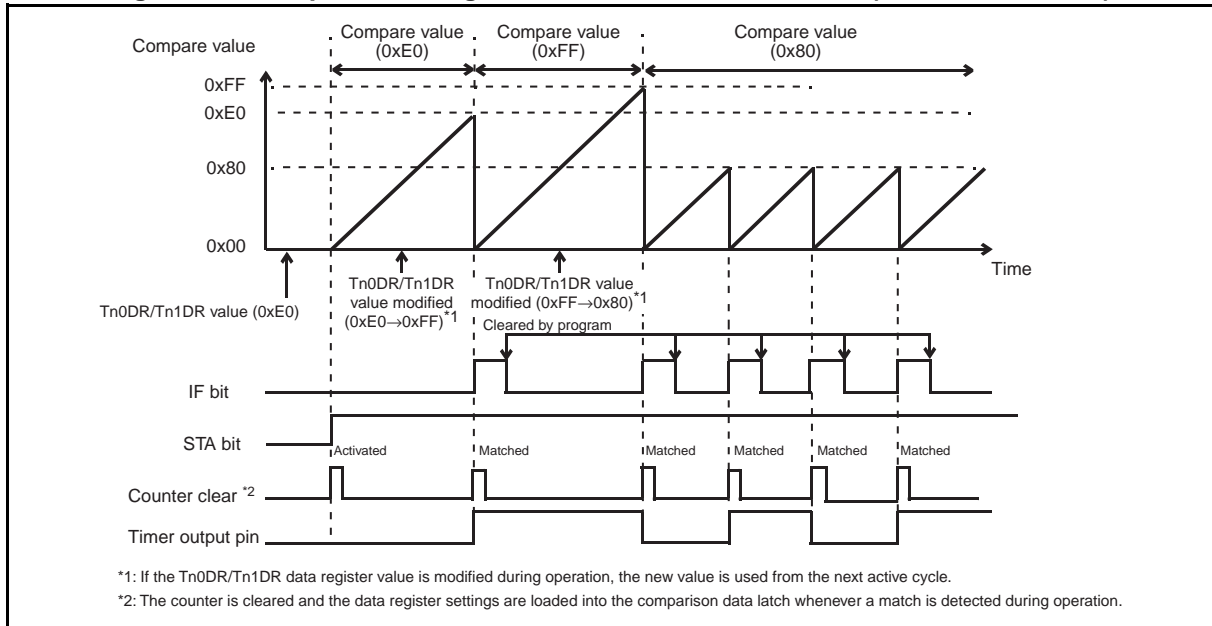
○: Bit to be used  
x: Unused bit  
1: Set to "1"  
0: Set to "0"

As for the interval timer function (continuous mode), enabling timer operation (Tn0CR1/Tn1CR1:STA = 1) causes the counter to start counting from "0x00" at the rising edge of a selected count clock signal. When the counter value matches the value in the 8/16-bit composite timer data register (Tn0DR/Tn1DR), the timer output bit (TMCRn:TO0/TO1) is inverted, the interrupt flag (Tn0CR1/Tn1CR1:IF) is set to "1", and the counter returns to "0x00" and restarts counting. The timer outputs square wave as a result of this continuous operation.

The value of the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is transferred to the temporary storage latch (comparison data storage latch) in the comparator either when the counter starts counting or when a counter value comparison match is detected. Do not write "0x00" to the 8/16-bit composite timer data register while the counter is counting.

When the timer stops operating, the timer output bit (TMCRn:TO0/TO1) holds the last value.

**Figure 12.7-2 Operation Diagram of Interval Timer Function (Continuous Mode)**



## 12.8 Operation of Interval Timer Function (Free-run Mode)

This section describes the operation of the interval timer function (free-run mode) of the 8/16-bit composite timer.

### ■ Operation of Interval Timer Function (Free-run Mode)

To use the interval timer function (free-run mode), do the settings shown in Figure 12.8-1.

**Figure 12.8-1 Settings for Interval Timer Function (Free-run Mode)**

|               | bit7                                       | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|--|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE  | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○  | ○    | ○    | ○    | 0    | 0    | 1    | 0    |
| Tn0CR1/Tn1CR1 | STA  | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1  | ○    | ○    | x    | x    | ○    | ○    | ○    |
| TMCRn         | TO1  | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○  | ○    | x    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Sets interval time (counter compare value) |      |      |      |      |      |      |      |

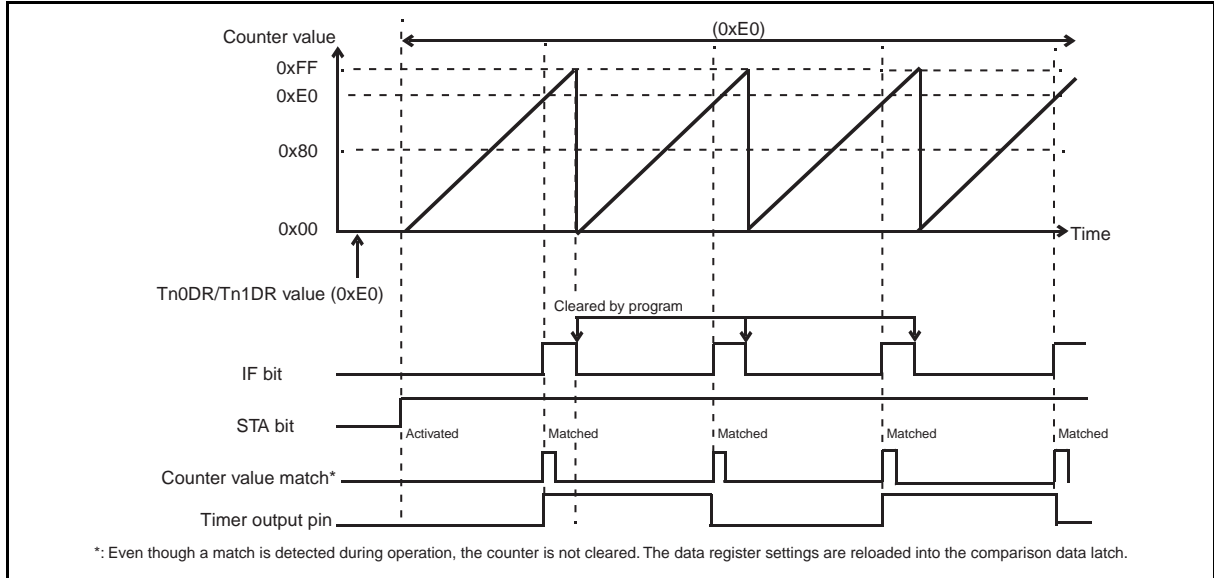
○: Bit to be used  
 x: Unused bit  
 1: Set to "1"  
 0: Set to "0"

As for the interval timer function (free-run mode), enabling timer operation (Tn0CR1/Tn1CR1:STA = 1) causes the counter to start counting from "0x00" at the rising edge of a selected count clock signal. When the counter value matches the value in the 8/16-bit composite timer data register (Tn0DR/Tn1DR), the timer output bit (TMCRn:TO0/TO1) is inverted and the interrupt flag (Tn0CR1/Tn1CR1:IF) is set to "1". If the counter continues to count with the above settings and then reaches "0xFF", it returns to "0x00" and restarts counting. The timer outputs square wave as a result of this continuous operation.

The value of the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is transferred to the temporary storage latch (comparison data storage latch) in the comparator either when the counter starts counting or when a counter value comparison match is detected. Do not write "0x00" to the 8/16-bit composite timer data register.

When the timer stops operation, the timer output bit (TMCRn:TO0/TO1) holds the last value.

**Figure 12.8-2 Operation Diagram of Interval Timer Function (Free-run Mode)**



## 12.9 Operation of PWM Timer Function (Fixed-cycle mode)

This section describes the operation of the PWM timer function (fixed-cycle mode) of the 8/16-bit composite timer.

### ■ Operation of PWM Timer Function (Fixed-cycle Mode)

To use the PWM timer function (fixed-cycle mode), do the settings shown in Figure 12.9-1.

Figure 12.9-1 Settings for PWM Timer Function (Fixed-cycle Mode)

|               | bit7                                 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|--------------------------------------|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE                                  | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | x                                    | ○    | ○    | ○    | 0    | 0    | 1    | 1    |
| Tn0CR1/Tn1CR1 | STA                                  | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | ○                                    | ○    | x    | x    | x    | x    | x    | ○    |
| TMCRn         | TO1                                  | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○                                    | ○    | x    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Sets "H" pulse width (compare value) |      |      |      |      |      |      |      |

○: Bit to be used  
x: Unused bit  
1: Set to "1"  
0: Set to "0"

As for the PWM timer function (fixed-cycle mode), PWM signal that has a fixed cycle and variable "H" pulse width is output from the timer output pin (TON0/TOn1). The cycle is fixed at "0xFF" in 8-bit operation or "0xFFFF" in 16-bit operation. The time is determined by the count clock selected. The "H" pulse width is specified by the value in the 8/16-bit composite timer data register (Tn0DR/Tn1DR).

This function has no effect on the interrupt flag (Tn0CR1/Tn1CR1:IF). Since each cycle always starts with "H" pulse output, the timer output initial value setting bit (Tn0CR1/Tn1CR1:SO) has no effect on operation.

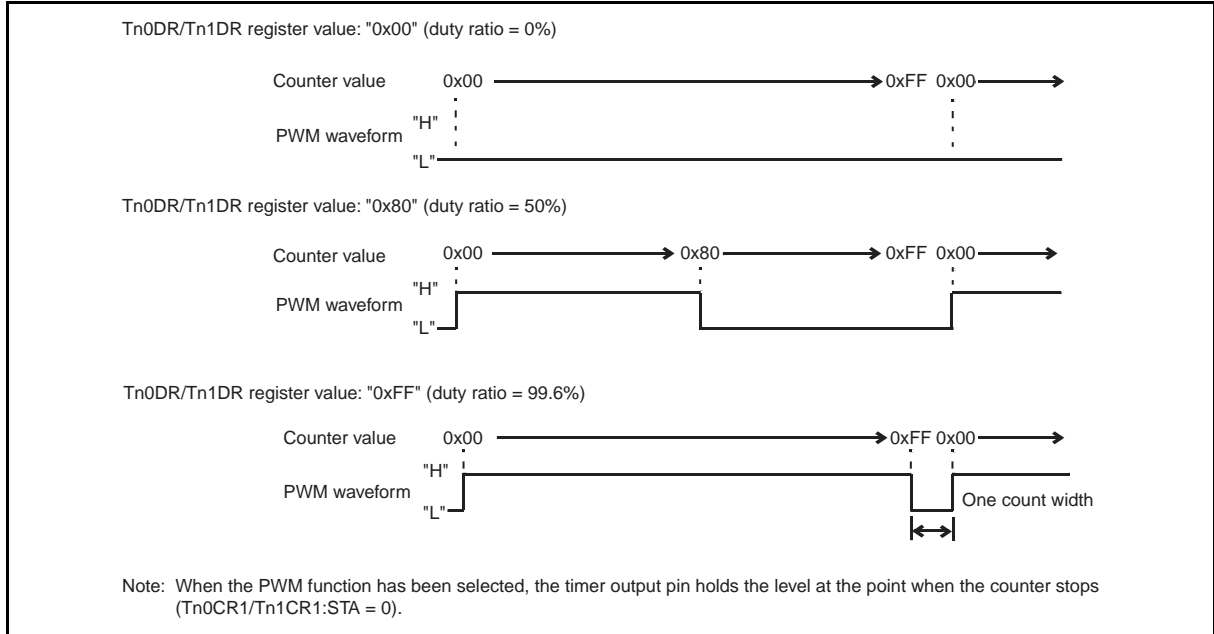
The value of the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is transferred to the temporary storage latch (comparison data storage latch) in the comparator either when the counter starts counting or when a counter value comparison match is detected.

When the timer stops operation, the timer output bit (TMCRn:TO0/TO1) holds the last value.

In the output waveform immediately after activation of the timer (write "1" to the STA bit), the "H" pulse is one count clock shorter than the value set in the Tn0DR/Tn1DR register.



**Figure 12.9-2 Operation Diagram of PWM Timer Function (Fixed-cycle Mode)**



## 12.10 Operation of PWM Timer Function (Variable-cycle Mode)

This section describes the operation of the PWM timer function (variable-cycle mode) of the 8/16-bit composite timer.

### ■ Operation of PWM Timer Function (Variable-cycle Mode)

To use the PWM timer function (variable-cycle mode), do the settings shown in Figure 12.10-1.

Figure 12.10-1 Settings for PWM Timer Function (Variable-cycle Mode)

|               | bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|--|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE  | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○  | ○    | ○    | ○    | 0    | 1    | 0    | 0    |
| Tn0CR1/Tn1CR1 | STA  | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1  | ○    | ○    | x    | x    | ○    | x    | x    |
| TMCRn         | TO1  | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○  | ○    | x    | x    | ○    | ○    | ○    | ○    |
| Tn0DR         | Sets "L" pulse width (compare value)           |      |      |      |      |      |      |      |
| Tn1DR         | Sets the cycle of PWM waveform (compare value) |      |      |      |      |      |      |      |

○: Bit to be used  
x: Unused bit  
1: Set to "1"  
0: Set to "0"

As for the PWM timer function (variable-cycle mode), both timers n0 and n1 are used. PWM signal of any cycle and of any duty is output from the timer output pin (TON0). The cycle is specified by the 8/16-bit composite timer n1 data register (Tn1DR), and the "L" pulse width is specified by the 8/16-bit composite timer n0 data register (Tn0DR).

Since both the 8-bit counters are used for this function, the composite timer cannot form a 16-bit counter.

Enabling timer operation (Tn0CR1/Tn1CR1:STA = 1) sets the mode bit (TMCRn:MOD) to "0". As the first cycle always begins with "L" pulse output, the timer initial value setting bit (Tn0CR1/Tn1CR1:SO) has no effect on operation.

An interrupt flag (Tn0CR1/Tn1CR1:IF) is set when the 8-bit counter corresponding to that interrupt flag matches the value in its corresponding 8/16-bit composite timer data register (Tn0DR/Tn1DR).

The 8/16-bit composite timer data register value is transferred to the temporary storage latch (comparison data storage latch) in the comparator either when the counter starts counting or when a comparison match with each counter value is detected.

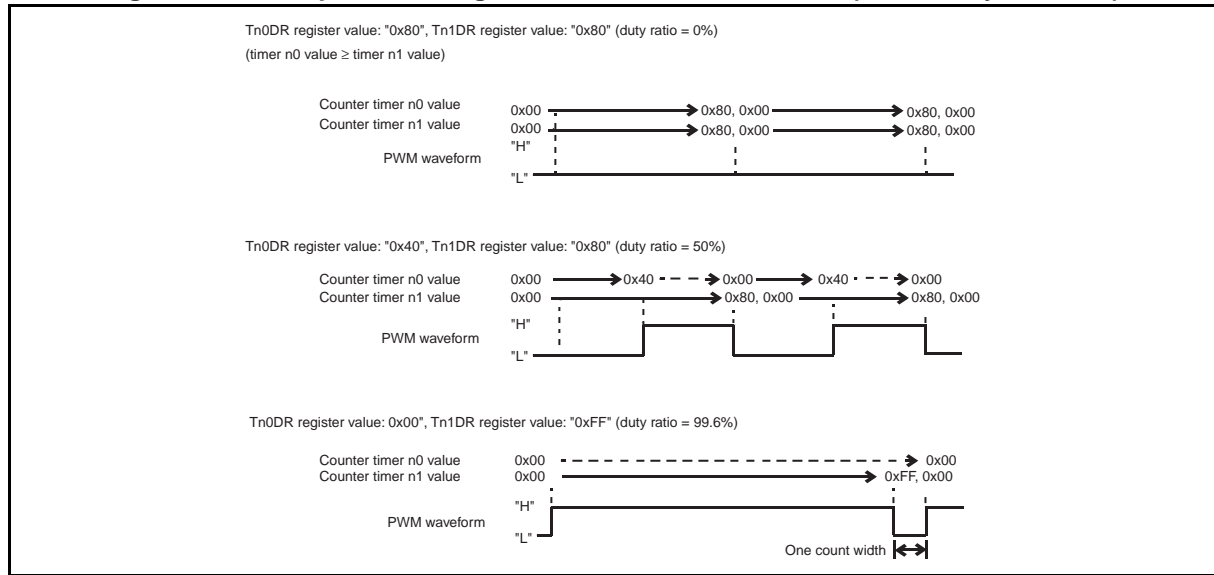
"H" is not output when the "L" pulse width setting value is greater than the cycle setting value.

The count clock must be selected for both timers n0 and n1. Selecting different count clocks for the two timers is prohibited.

When the timer stops operating, the timer output bit (TMCRn:TO0) holds the last output value.

If the 8/16-bit composite timer data register is modified during operation, the data written will become valid from the cycle immediately after the detection of a synchronous match.

**Figure 12.10-2 Operation Diagram of PWM Timer Function (Variable-cycle Mode)**



## 12.11 Operation of PWC Timer Function

This section describes the operation of the PWC timer function of the 8/16-bit composite timer.

### ■ Operation of PWC Timer Function

To use the PWC timer function, do the settings shown in Figure 12.11-1.

**Figure 12.11-1 Settings for PWC Timer Function**

|               | bit7                                | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|-------------------------------------|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE                                 | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○                                   | ○    | ○    | ○    | ○    | ○    | ○    | ○    |
| Tn0CR1/Tn1CR1 | STA                                 | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1                                   | ○    | ○    | ○    | ○    | ○    | ○    | x    |
| TMCRn         | TO1                                 | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | ○                                   | ○    | ○    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Holds pulse width measurement value |      |      |      |      |      |      |      |

○: Bit to be used  
 x: Unused bit  
 1: Set to "1"

When the PWC timer function is selected, the width and cycle of an external input pulse can be measured. The edges at which counting starts and ends are selected by the timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]).

In the operation of this function, the counter starts counting from "0x00" immediately after a specified count start edge of an external input signal is detected. Upon the detection of a specified count end edge, the count value is transferred to the 8/16-bit composite timer data register (Tn0DR/Tn1DR), and the interrupt flag (Tn0CR1/Tn1CR1:IR) and the buffer full flag (Tn0CR1/Tn1CR1:BF) are set to "1". The buffer full flag is set to "0" when the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is read.

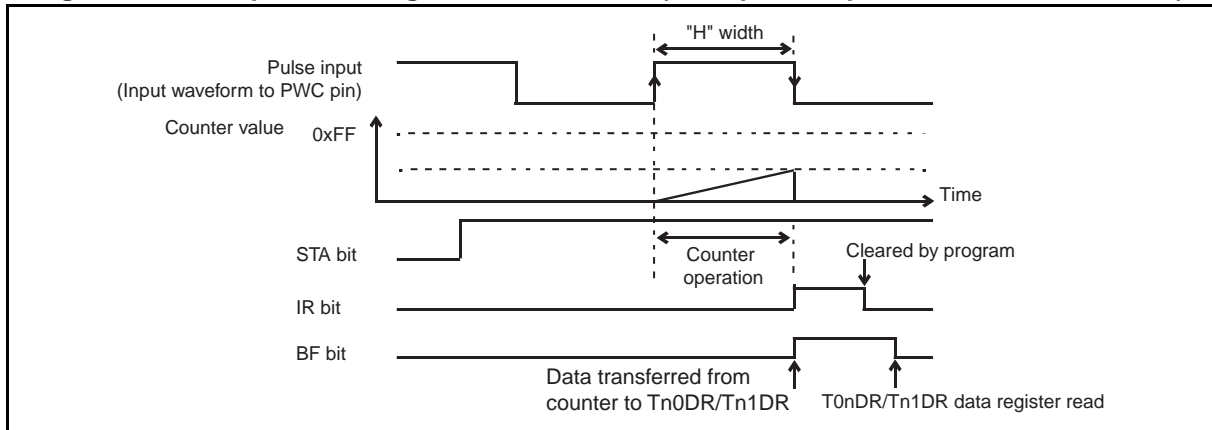
If the buffer full flag is set to "1", the 8/16-bit composite timer data register holds data. Even if the next edge is detected during that time, the next measurement result is lost since the count value has not been transferred to the 8/16-bit composite timer data register.

There is an exception. With the F[3:0] bits in the Tn0CR0/Tn1CR0 register having been set to "0b1001", even though the BF bit is set to "1", the "H" pulse measurement result is transferred to the 8/16-bit composite timer data register, while the cycle measurement result is not transferred to the 8/16-bit composite timer data register. Therefore, in order to perform cycle measurement, the "H" pulse measurement result must be read before a cycle is completed. In addition, the result of "H" pulse measurement and that of cycle measurement are lost if they are not read before the completion of the next "H" pulse.

The time exceeding the range of the counter can be measured by counting the number of counter overflows using the software. When the counter overflows, the interrupt flag (Tn0CR1/Tn1CR1:IF) is set to "1". The interrupt service routine can therefore be used to count the number of overflows. In addition, the timer output is inverted due to the overflow. The timer output initial value can be set by the timer output initial value bit (Tn0CR1/Tn1CR1:SO).

When the timer stops operating, the timer output bit (TMCRn:TO1/TO0) holds the last value.

Figure 12.11-2 Operation Diagram of PWC Timer (Example of H-pulse Width Measurement)



## 12.12 Operation of Input Capture Function

This section describes the operation of the input capture function of the 8/16-bit composite timer.

### ■ Operation of Input Capture Function

To use the input capture function, do the settings shown in Figure 12.12-1.

Figure 12.12-1 Settings for Input Capture Function

|               | bit7                                | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---------------|-------------------------------------|------|------|------|------|------|------|------|
| Tn0CR0/Tn1CR0 | IFE                                 | C2   | C1   | C0   | F3   | F2   | F1   | F0   |
|               | ○                                   | ○    | ○    | ○    | ○    | ○    | ○    | ○    |
| Tn0CR1/Tn1CR1 | STA                                 | HO   | IE   | IR   | BF   | IF   | SO   | OE   |
|               | 1                                   | ○    | ○    | ○    | x    | ○    | x    | x    |
| TMCRn         | TO1                                 | TO0  | TIS  | MOD  | FE11 | FE10 | FE01 | FE00 |
|               | x                                   | x    | ○    | ○    | ○    | ○    | ○    | ○    |
| Tn0DR/Tn1DR   | Holds pulse width measurement value |      |      |      |      |      |      |      |

○: Bit to be used  
 x: Unused bit  
 1: Set to "1"

When the input capture function is selected, the counter value is stored to the 8/16-bit composite timer data register (Tn0DR/Tn1DR) immediately after an edge of the external signal input is detected. The target edge to be detected is selected by the timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]).

This function is available in free-run mode and clear mode, which can be selected by the timer operating mode select bits.

In clear mode, the counter starts counting from "0x00". When an edge is detected, the counter value is transferred to the 8/16-bit composite timer data register (Tn0DR/Tn1DR), the interrupt flag (Tn0CR1/Tn1CR1:IR) is set to "1", and the counter returns to "0x00" and restarts counting.

In free-run mode, when an edge is detected, the counter value is transferred to the 8/16-bit composite timer data register (Tn0DR/Tn1DR) and the interrupt flag (Tn0CR1/Tn1CR1:IR) is set to "1". In this case, the counter continues to count without being cleared.

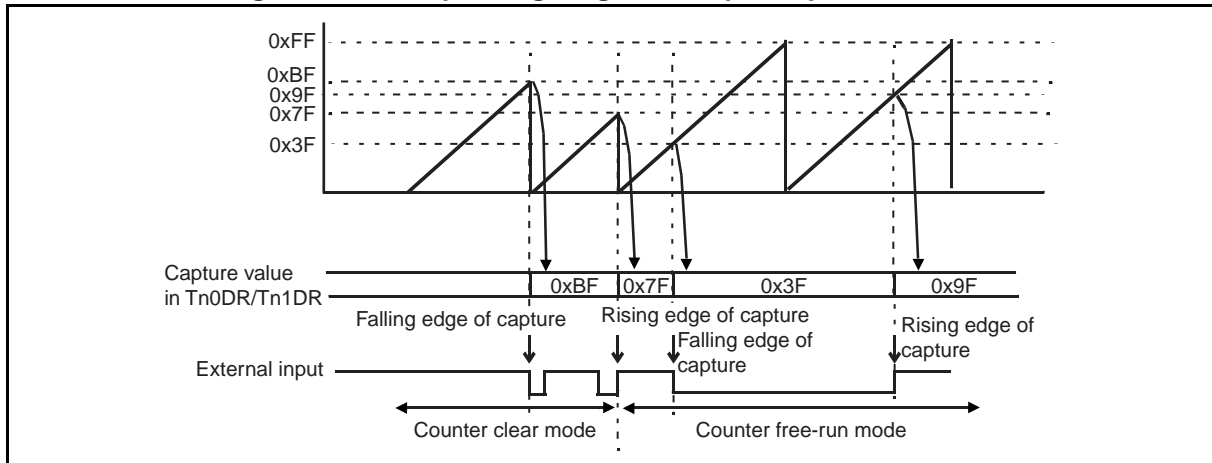
This function has no effect on the buffer full flag (Tn0CR1/Tn1CR1:BF).

The time exceeding the range of the counter can be measured by counting the number of counter overflows using the software. When the counter overflows, the interrupt flag (Tn0CR1/Tn1CR1:IF) is set to "1". The interrupt service routine can therefore be used to count the number of overflows. In addition, the timer output is inverted due to the overflow. The timer output initial value can be set by the timer output initial value bit (Tn0CR1/Tn1CR1:SO).

Note:

See "12.15 Notes on Using 8/16-bit Composite Timer" for notes on using the input capture function.

**Figure 12.12-2 Operating Diagram of Input Capture Function**

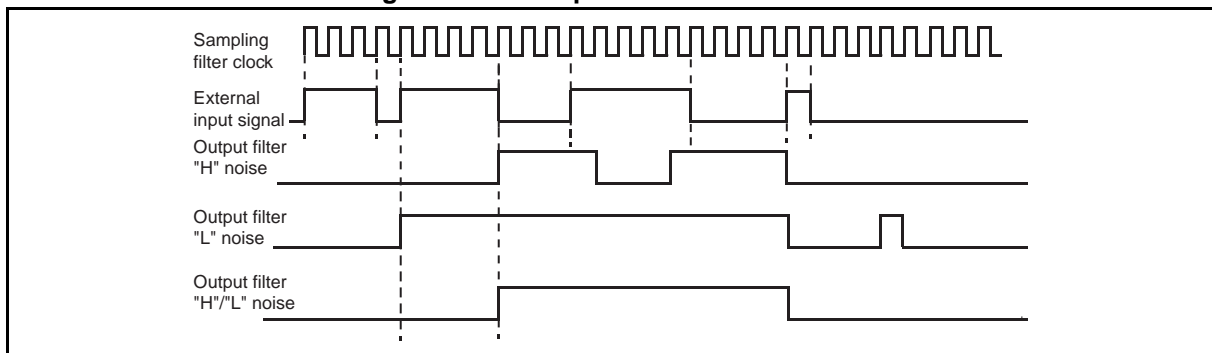


## 12.13 Operation of Noise Filter

This section describes the operation of the noise filter of the 8/16-bit composite timer.

When the input capture function or PWC timer function is selected, a noise filter can be used to eliminate the pulse noise of the signal from the external input pin (ECn). H-pulse noise, L-pulse noise, or H/L-pulse noise elimination can be selected by the FE11, FE10, FE01 and FE00 bits in the TMCRn register. The maximum pulse width that can be eliminated is three machine clock cycles. If the noise filter function is activated, the signal input will be delayed for four machine clock cycles.

Figure 12.13-1 Operation of Noise Filter





## MB95850K/860K/870K Series

### 12.14 Registers

---

This section describes the registers of the 8/16-bit composite timer.

---

**Table 12.14-1 List of 8/16-bit Composite Timer Registers**

| Register abbreviation | Register name   | Reference |
|-----------------------|---|-----------|
| Tn0CR0                | 8/16-bit composite timer n0 status control register 0 | 12.14.1   |
| Tn1CR0                | 8/16-bit composite timer n1 status control register 0 | 12.14.1   |
| Tn0CR1                | 8/16-bit composite timer n0 status control register 1 | 12.14.2   |
| Tn1CR1                | 8/16-bit composite timer n1 status control register 1 | 12.14.2   |
| TMCRn                 | 8/16-bit composite timer timer mode control register  | 12.14.3   |
| Tn0DR                 | 8/16-bit composite timer n0 data register             | 12.14.4   |
| Tn1DR                 | 8/16-bit composite timer n1 data register             | 12.14.4   |

## 12.14.1 8/16-bit Composite Timer Status Control Register 0 (Tn0CR0/Tn1CR0)

The 8/16-bit composite timer status control register 0 (Tn0CR0/Tn1CR0) selects the timer operation mode, selects the count clock, and enables or disables IF flag interrupts. The Tn0CR0 and Tn1CR0 registers correspond to timers n0 and n1 respectively.

### ■ Register Configuration

|               |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | IFE | C2  | C1  | C0  | F3  | F2  | F1  | F0  |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

### ■ Register Functions

[bit7] IFE: IF flag interrupt enable bit

This bit enables or disables IF flag interrupts.

During timer operation (Tn0CR1/Tn1CR1:STA = 1), the write access to this bit has no effect on operation. Ensure that the timer has stopped before modifying this bit.

With this bit set to "1", an IF flag interrupt request is output when both the IE bit (Tn0CR1/Tn1CR1:IE) and the IF flag (Tn0CR1/Tn1CR1:IF) are set to "1".

| bit7               | Details                         |
|--------------------|---------------------------------|
| <b>Writing "0"</b> | Disables the IF flag interrupt. |
| <b>Writing "1"</b> | Enables the IF flag interrupt.  |

## MB95850K/860K/870K Series

### [bit6:4] C[2:0]: Count clock select bits

These bits select the count clock.

The count clock is generated by the prescaler. See "3.9 Operation of Prescaler".

During timer operation (Tn0CR1/Tn1CR1:STA = 1), the write access to these bits has no effect on operation in timer operation.

The clock selection of Tn1CR0 (timer n1) is nullified in 16-bit operation.

These bits cannot be set to "0b111" when the PWC function or input capture function is used. An attempt to write "0b111" with the PWC function or input capture function in use resets the bits to "0b000". The bits are also reset to "0b000" if the timer enters the input capture operation mode with the bits set to "0b111".

When these bits are set to "0b110", the count clock from the time-base timer will be used as the count clock. Depending on the settings of the SYCC register, the count clock from the time-base timer can be generated from the main clock, the main CR clock, or the main CR PLL clock. In the case of using the count clock from the time-base timer as the count clock, resetting the time-base timer by writing "1" to the time-base timer clear bit in the time-base timer control register (TBTC:TCLR) will affect the count time.

| bit6:4        | Details<br>(MCLK: machine clock, F <sub>CH</sub> : main clock, F <sub>CRH</sub> : main CR clock,<br>F <sub>MCRPLL</sub> : main CR PLL clock) |
|---------------|--|
| Writing "000" | 1 MCLK   |
| Writing "001" | MCLK/2   |
| Writing "010" | MCLK/4   |
| Writing "011" | MCLK/8   |
| Writing "100" | MCLK/16  |
| Writing "101" | MCLK/32  |
| Writing "110" | F <sub>CH</sub> /2 <sup>7</sup> , F <sub>CRH</sub> /2 <sup>6</sup> or F <sub>MCRPLL</sub> /2 <sup>6</sup> *                                  |
| Writing "111" | External clock   |

\*: The value to be used as the count clock depends on the settings of the SCS[2:0] bits in the SYCC register.

[bit3:0] F[3:0]: Timer operating mode select bits

These bits select the timer operating mode.

The PWM timer function (variable-cycle mode; F[3:0] = 0b0100) is set by either the Tn0CR0 (timer n0) register or Tn1CR0 (timer n1) register. If one of the timers starts operating (Tn0CR1/Tn1CR1: STA = 1), the F[3:0] bits of the other timer are automatically set to "0b0100".

With the 16-bit operating mode having been selected (TMCRn:MOD = 1), if the composite timer starts operating using the PWM timer function (variable-cycle mode) (Tn0CR1/Tn1CR1:STA = 1), the MOD bit is set to "0" automatically.

Write access to these bits is nullified in timer operation (Tn0CR1/Tn1CR1:STA = 1).

| bit3:0         | Details   |
|----------------|---|
| Writing "0000" | Interval timer (one-shot mode)  |
| Writing "0001" | Interval timer (continuous mode)  |
| Writing "0010" | Interval timer (free-run mode)  |
| Writing "0011" | PWM timer (fixed-cycle mode)  |
| Writing "0100" | PWM timer (variable-cycle mode)   |
| Writing "0101" | PWC timer (H pulse = rising edge to falling edge)                                     |
| Writing "0110" | PWC timer (L pulse = falling edge to rising edge)                                     |
| Writing "0111" | PWC timer (cycle = rising edge to rising edge)  |
| Writing "1000" | PWC timer (cycle = falling edge to falling edge)                                      |
| Writing "1001" | PWC timer (H pulse = rising edge to falling edge; cycle = rising edge to rising edge) |
| Writing "1010" | Input capture (rising edge, free-run counter)   |
| Writing "1011" | Input capture (falling edge, free-run counter)  |
| Writing "1100" | Input capture (both edges, free-run counter)  |
| Writing "1101" | Input capture (rising edge, counter clear)  |
| Writing "1110" | Input capture (falling edge, counter clear)   |
| Writing "1111" | Input capture (both edges, counter clear)   |

## MB95850K/860K/870K Series

### 12.14.2 8/16-bit Composite Timer Status Control Register 1 (Tn0CR1/Tn1CR1)

The 8/16-bit composite timer status control register 1 (Tn0CR1/Tn1CR1) controls the interrupt flag, timer output, and timer operations. Tn0CR1 and Tn1CR1 registers correspond to timers n0 and n1 respectively.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Field         | STA | HO  | IE  | IR  | BF  | IF  | SO  | OE  |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### ■ Register Functions

[bit7] STA: Timer operation enable bit

This bit enables or stops the timer operation.

**Writing "0"**: stops the timer operation and sets the count value to "0x00".

- With the PWM timer function (variable-cycle mode) in use (Tn0CR0/Tn1CR0:F[3:0] = 0b0100), the STA bit in either the Tn0CR1 (timer n0) or the Tn1CR1 (timer n1) register can be used to enable or disable the timer operation. If the STA bit in one of the registers is set to "0", the STA bit in the other one is automatically set to the same value.
- In 16-bit operating mode (TMCRn:MOD = 1), use the STA bit in the Tn0CR1 (timer n0) register to enable or disable timer operation. If the STA bit of one of the timers is set to "0", the STA bit in the other one is automatically set to the same value.

**Writing "1"**: starts the timer operation from the count value "0x00".

- Before setting this bit to "1", set the count clock select bits (Tn0CR0/Tn1CR0:C[2:0]), timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]), timer output initial value bit (Tn0CR1/Tn1CR1:SO), 8-bit/16-bit operating mode select bit (TMCRn:MOD), and filter function select bits (TMCRn:FEn1, FEn0).

| bit7               | Details                      |
|--------------------|------------------------------|
| <b>Writing "0"</b> | Stops the timer operation.   |
| <b>Writing "1"</b> | Enables the timer operation. |

[bit6] HO: Timer suspend bit

This bit suspends or resumes the timer operation.

Writing "1" to this bit during timer operation suspends the timer operation.

When the timer operation has been enabled (Tn0CR1/Tn1CR1:STA = 1), writing "0" to the bit resumes the timer operation.

With the PWM timer function (variable-cycle mode) in use (Tn0CR0/Tn1CR0:F[3:0] = 0b0100), the HO bit in either Tn0CR1 (timer n0) or Tn1CR1 (timer n1) can be used to suspend or resume timer operation. If the HO bit in one of the registers is set to "0" or "1", the HO bit in the other one is automatically set to the same value.

In 16-bit operating mode (TMCRn:MOD = 1), use the HO bit in the Tn0CR1 (timer n0) register to suspend or resume timer operation. If the HO bit in one of the registers is set to "0" or "1", the HO bit in the other one is automatically set to the same value.

| bit6               | Details                       |
|--------------------|-------------------------------|
| <b>Writing "0"</b> | Resumes the timer operation.  |
| <b>Writing "1"</b> | Suspends the timer operation. |

[bit5] IE: Interrupt request enable bit

This bit enables or disables the output of interrupt requests.

Writing "0" to this bit disables the interrupt request.

Writing "1" to this bit outputs an interrupt request when the pulse width measurement completion/edge detection flag (Tn0CR1/Tn1CR1:IR) or timer reload/overflow flag (Tn0CR1/Tn1CR1:IF) is set to "1".

However, an interrupt request from the timer reload/overflow flag (Tn0CR1/Tn1CR1:IF) is not output unless the IF flag interrupt enable bit (Tn0CR0/Tn1CR0:IFE) is also set to "1".

| bit5               | Details                         |
|--------------------|---------------------------------|
| <b>Writing "0"</b> | Disables the interrupt request. |
| <b>Writing "1"</b> | Enables the interrupt request.  |

[bit4] IR: Pulse width measurement completion/edge detection flag

This bit indicates the completion of pulse width measurement or the detection of an edge.

Writing "0" to this bit sets it to "0".

Writing "1" to this bit has no effect on operation.

With the PWC timer function in use, this bit is set to "1" immediately after pulse width measurement is complete.

With the input capture function in use, this bit is set to "1" immediately after an edge is detected.

The bit is set to "0" when the function of the composite timer selected is neither the PWC timer function nor the input capture function.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

The IR bit in the Tn1CR1 (timer n1) register is set to "0" in 16-bit operation.

| bit4               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the pulse width measurement has been completed or no edge has been detected. |
| <b>Reading "1"</b> | Indicates that the pulse width measurement has been completed or an edge has been detected. |
| <b>Writing "0"</b> | Clears this flag.   |
| <b>Writing "1"</b> | Has no effect on operation.   |

**MB95850K/860K/870K Series**[bit3] BF: Data register full flag

With the PWC timer function in use, this bit is set to "1" when a count value is stored in the 8/16-bit composite timer data register (Tn0DR/Tn1DR) immediately after pulse width measurement is complete.

In 8-bit operation, this bit is set to "0" when the 8/16-bit composite timer data register (Tn0DR/Tn1DR) is read.

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) holds data if this bit is set to "1". With this bit being "1", even when the next edge is detected, the count value is not transferred to the Tn0DR/Tn1DR register, and the next measurement result is thus lost. Nonetheless, there is an exception. With the F[3:0] bits in the Tn0CR0/Tn1CR0 register having been set to "0b1001", even though the BF bit is set to "1", the "H" pulse measurement result is transferred to the Tn0DR/Tn1DR register, while the cycle measurement result is not transferred to the Tn0DR/Tn1DR register. Therefore, in order to perform cycle measurement, the "H" pulse measurement result must be read before a cycle is completed. In addition, the result of "H" pulse measurement and that of cycle measurement are lost if they are not read before the completion of the next "H" pulse.

The BF bit in the Tn0CR1 (timer n0) register is set to "0" when the Tn1DR (timer n1) register is read during 16-bit operation.

The BF bit in the Tn1CR1 (timer n1) register is set to "0" during 16-bit operation.

This bit is "0" when any timer function other than the PWC timer function is selected.

Writing a value to this bit has no effect on operation.

| bit3               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that there is no measurement data in the 8/16-bit composite timer data register (Tn0DR/Tn1DR). |
| <b>Reading "1"</b> | Indicates that there is measurement data in the 8/16-bit composite timer data register (Tn0DR/Tn1DR).    |

[bit2] IF: Timer reload/overflow flag

This bit detects the count value match and the counter overflow.

With the interval timer function (one-shot or continuous) or the PWM timer function (variable-cycle mode) in use, this bit is set to "1" if the 8/16-bit composite timer data register (Tn0DR/Tn1DR) value matches the count value.

With the PWC timer function or the input capture function in use, this bit is set to "1" if a counter overflow occurs.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1". Writing "0" to this bit sets it to "0".

Writing "1" to this bit has no effect on operation.

This bit becomes "0" if the PWM function (variable-cycle mode) is selected.

The IF bit in the Tn1CR1 (timer n1) register is "0" in 16-bit operation.

| bit2               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that neither timer reload nor overflow has occurred. |
| <b>Reading "1"</b> | Indicates that a timer reload or an overflow has occurred.     |
| <b>Writing "0"</b> | Clears this flag.  |
| <b>Writing "1"</b> | Has no effect on operation.                                    |

[bit1] SO: Timer output initial value bit

The timer output (TMCRn:TO1/TO0) initial value is set by writing a value to this bit. The value in this bit is reflected in the timer output when the timer operation enable bit (Tn0CR1/Tn1CR1:STA) changes from "0" to "1".

In 16-bit operating mode (TMCRn:MOD = 1), use the SO bit in the Tn0CR1 (timer n0) register to set the timer output initial value. In this case, the value of the SO bit in the other one has no effect on operation.

During timer operation (Tn0CR1/Tn1CR1:STA = 1), the write access to this bit is invalid. However, in 16-bit operating mode, although a value can be written to the SO bit in the Tn1CR1 (timer n1) register even during timer operation, the value written has no direct effect on the timer output.

When the PWM timer function (fixed cycle mode or variable cycle mode) or the input capture function is in use, the value of this bit has no effect on operation.

| bit1               | Details                                     |
|--------------------|---|
| <b>Writing "0"</b> | Sets "0" as the timer output initial value. |
| <b>Writing "1"</b> | Sets "1" as the timer output initial value. |

[bit0] OE: Timer output enable bit

This bit enables or disables timer output.

Writing "0" to this bit disables outputting the timer value (TMCRn:TO1/TO0) to the external pin. In this case, the external pin serves as a general-purpose port.

Writing "1" to this bit enables outputting the timer value to the external pin.

| bit0               | Details                |
|--------------------|------------------------|
| <b>Writing "0"</b> | Disables timer output. |
| <b>Writing "1"</b> | Enables timer output.  |



## MB95850K/860K/870K Series

### 12.14.3 8/16-bit Composite Timer Timer Mode Control Register (TMCRn)

The 8/16-bit composite timer timer mode control register (TMCRn) selects the filter function, 8-bit or 16-bit operating mode, and signal input to timer n0 and indicates the timer output value. This register serves both timer n0 and timer n1.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3    | 2    | 1    | 0    |
|---------------|-----|-----|-----|-----|------|------|------|------|
| Field         | TO1 | TO0 | TIS | MOD | FE11 | FE10 | FE01 | FE00 |
| Attribute     | R   | R   | R/W | R/W | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0   | 0   | 0   | 0   | 0    | 0    | 0    | 0    |

#### ■ Register Functions

##### [bit7] TO1: Timer n1 output bit

This bit indicates the output value of timer n1. When the timer starts operation (Tn0CR1/Tn1CR1:STA = 1), the value in the bit changes depending on the timer function selected.

Writing a value to this bit has no effect on operation.

In 16-bit operating mode, if the PWM timer function (variable-cycle mode) or the input capture function is selected, the value in the bit becomes undefined.

With the interval timer function or the PWC timer function having been selected, if the timer stops operating (Tn0CR1/Tn1CR1:STA = 0), this bit holds the last value.

With the PWM timer function (variable-cycle mode) having been selected, if the timer stops operating (Tn0CR1/Tn1CR1:STA = 0), this bit holds the last value.

When the timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]) are modified with the timer stopping operating, this bit indicates the last value of timer operation if the same timer operation has been performed; otherwise it indicates its initial value "0".

##### [bit6] TO0: Timer n0 output bit

This bit indicates the output value of timer n0. When the timer starts operation (Tn0CR1/Tn1CR1:STA = 1), the value in the bit changes depending on the selected timer function.

Writing a value to this bit has no effect on operation.

If the input capture function is selected, the value in the bit becomes undefined.

With the interval timer function or the PWC timer function having been selected, if the timer stops operating (Tn0CR1/Tn1CR1:STA = 0), this bit holds the last value.

With the PWM timer function (variable-cycle mode) having been selected, if the timer stops operating (Tn0CR1/Tn1CR1:STA = 0), this bit holds the last value.

When the timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]) are modified with the timer stopping operating, this bit indicates the last value of timer operation if the same timer operation has been performed; otherwise it indicates its initial value "0".

[bit5] TIS: Timer n0 internal signal select bit

This bit selects the signal input to timer n0 when the PWC timer function or input capture function is selected.

| bit5               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Selects the external signal (ECn) as the signal input for timer n0. |
| <b>Writing "1"</b> | Setting prohibited.   |

[bit4] MOD: 8-bit/16-bit operating mode select bit

This bit selects 8-bit or 16-bit operating mode.

Writing "0" to this bit allows timers n0 and n1 to operate as separate 8-bit timers.

Writing "1" to this bit allows timers n0 and n1 to operate as a 16-bit timer.

While this bit is "1", if the timer starts operating (Tn0CR1/Tn1CR1:STA = 1) with the PWM timer function (variable-cycle mode), this bit is automatically set to "0".

During timer operation (Tn0CR1/Tn1CR1:STA = 1), the write access to this bit is invalid.

| bit4               | Details                            |
|--------------------|------------------------------------|
| <b>Writing "0"</b> | Selects the 8-bit operating mode.  |
| <b>Writing "1"</b> | Selects the 16-bit operating mode. |

[bit3:2] FE1[1:0]: Timer n1 filter function select bits

These bits select the filter function for the external signal (ECn) to timer n1 when the PWC timer function or the input capture function is selected.

During timer operation (Tn1CR1:STA = 1), the write access to these bits is invalid.

The settings of the bits have no effect on operation when the interval timer function or the PWM timer function is selected (the filter function does not operate.).

| bit3:2              | Details   |
|---------------------|---|
| <b>Writing "00"</b> | Disables the filter function.                         |
| <b>Writing "01"</b> | Filters out "H" pulse noise.                          |
| <b>Writing "10"</b> | Filters out "L" pulse noise.                          |
| <b>Writing "11"</b> | Filters out both "H" pulse noise and "L" pulse noise. |

[bit1:0] FE0[1:0]: Timer n0 filter function select bits

These bits select the filter function for the external signal (ECn) to timer n0 when the PWC timer function or the input capture function is selected.

During timer operation (Tn0CR1:STA = 1), the write access to these bits is invalid.

The settings of the bits have no effect on operation when the interval timer function or the PWM timer function is selected (the filter function does not operate.).

| bit1:0              | Details   |
|---------------------|---|
| <b>Writing "00"</b> | Disables the filter function.                         |
| <b>Writing "01"</b> | Filters out "H" pulse noise.                          |
| <b>Writing "10"</b> | Filters out "L" pulse noise.                          |
| <b>Writing "11"</b> | Filters out both "H" pulse noise and "L" pulse noise. |

## MB95850K/860K/870K Series

### 12.14.4 8/16-bit Composite Timer Data Register (Tn0DR/Tn1DR)

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) is used to set the maximum count value during the interval timer operation or the PWM timer operation and to read the count value during the PWC timer operation or the input capture operation. The Tn0DR and Tn1DR registers correspond to timers n0 and n1 respectively.

#### ■ Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | TDR7 | TDR6 | TDR5 | TDR4 | TDR3 | TDR2 | TDR1 | TDR0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

#### ● Interval timer function

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) is used to set the interval time. When the timer starts operating (Tn0CR1/Tn1CR1:STA = 1), the value of this register is transferred to the latch in the 8-bit comparator and the counter starts counting. When the count value matches the value held in the latch in the 8-bit comparator, the value of this register is transferred again to the latch, and the counter returns to "0x00" and continues to count.

The current count value can be read from this register.

An attempt to write "0x00" to this register is disabled in interval timer function.

In 16-bit operation, write the upper timer data to Tn1DR and lower timer data to Tn0DR, and write or read Tn1DR first and then Tn0DR.

#### ● PWM timer function (fixed-cycle)

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) is used to set "H" pulse width time. When the timer starts operating (Tn0CR1/Tn1CR1:STA = 1), the value of this register is transferred to the latch in the 8-bit comparator and the counter starts counting from timer output "H". When the count value matches the value transferred to the latch, the timer output becomes "L" and the counter continues to count until the count value reaches "0xFF". When an overflow occurs, the value of this register is transferred again to the latch in the 8-bit comparator and the counter performs the next cycle of counting.

The current value can be read from this register. In 16-bit operation, write the upper timer data to Tn1DR and lower timer data to Tn0DR, and write or read Tn1DR first and then Tn0DR.

● PWM timer function (variable-cycle)

The 8/16-bit composite timer n0 data register (Tn0DR) and 8/16-bit composite timer n1 data register (Tn1DR) are used to set "L" pulse width time and cycle respectively. When the timer starts operating (Tn0CR1/Tn1CR1:STA = 1), the value of each register is transferred to the latch in the 8-bit comparator and the two counters start counting from timer output "L". When the Tn0DR value transferred to the latch matches the timer n0 counter value, the timer output becomes "H" and the counting continues until the Tn1DR value transferred to the latch matches the timer n1 counter value. When the Tn1DR value transferred to the latch of the 8-bit comparator matches the timer n1 counter value, the values of the Tn0DR register and the T01DR register are transferred again to the latch and the counter performs the next PWM cycle of counting.

The current count value can be read from this register. In 16-bit operating mode, write the upper timer data to Tn1DR and lower timer data to Tn0DR, and read Tn1DR first and then Tn0DR.

● PWC timer function

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) is used to read PWC measurement results. When PWC measurement is completed, the counter value is transferred to this register and the BF bit is set to "1".

When the 8/16-bit composite timer data register is read, the BF bit is set to "0". While the BF bit is "1", no data is transferred to the 8/16-bit composite timer data register.

There is an exception. With the F[3:0] bits in the Tn0CR0/Tn1CR0 register having been set to "0b1001", even though the BF bit is set to "1", the "H" pulse measurement result is transferred to the 8/16-bit composite timer data register, while the cycle measurement result is not transferred to the 8/16-bit composite timer data register. Therefore, in order to perform cycle measurement, the "H" pulse measurement result must be read before a cycle is completed. In addition, the result of "H" pulse measurement and that of cycle measurement are lost if they are not read before the completion of the next "H" pulse.

When reading the 8/16-bit composite timer data register, ensure that the BF bit is not cleared accidentally.

If new data is written to the 8/16-bit composite timer data register, the stored measurement data is replaced with the new data. Therefore, do not write data to the register. In 16-bit operating mode, write the upper timer data to Tn1DR and lower timer data to Tn0DR, and read Tn1DR first and then Tn0DR.

● Input capture function

The 8/16-bit composite timer data register (Tn0DR/Tn1DR) is used to read input capture results. When an edge specified is detected, the counter value is transferred to the 8/16-bit composite timer data register.

If new data is written to the 8/16-bit composite timer data register, the stored measurement data is replaced with the new data. Therefore, do not write data to the register. In 16-bit operating mode, write the upper timer data to Tn1DR and lower timer data to Tn0DR, and read Tn1DR first and then Tn0DR.

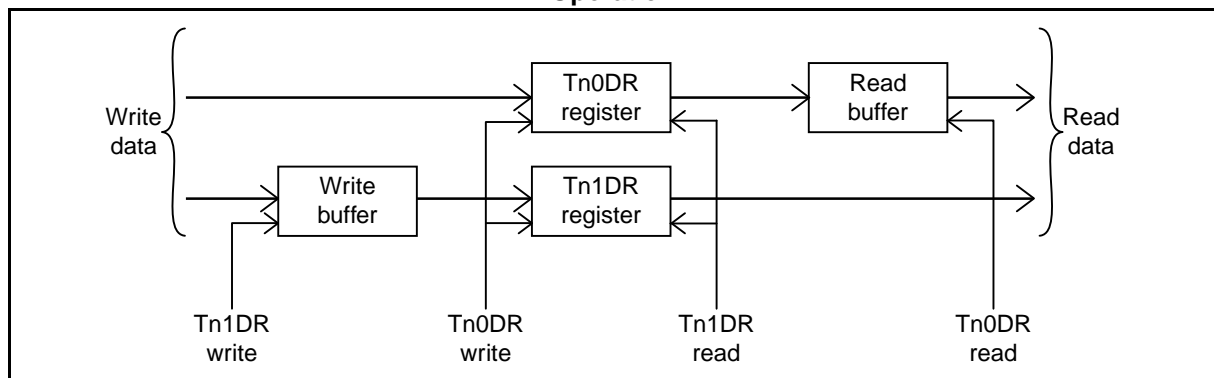
● Read and write operations

Read and write operations of Tn0DR and Tn1DR are performed in the following manner in 16-bit operating mode or when the PWM timer function (variable-cycle) is selected.

- Read from Tn1DR: In addition to the read access to Tn1DR, the value of Tn0DR is also stored in the internal read buffer at the same time.
- Read from Tn0DR: The internal read buffer is read.
- Write to Tn1DR: Data is written to the internal write buffer.
- Write to Tn0DR: In addition to the write access to Tn0DR, the value of the internal write buffer is stored in Tn1DR at the same time.

Figure 12.14-1 shows the Tn0DR and Tn1DR registers read from and written to during 16-bit operation.

**Figure 12.14-1 Read and Write Operations of Tn0DR and Tn1DR Registers during 16-bit Operation**



## 12.15 Notes on Using 8/16-bit Composite Timer

---

This section provides notes on using the 8/16-bit composite timer.

---

### ■ Notes on Using 8/16-bit Composite Timer

- To switch the timer function with the timer operating mode select bits (Tn0CR0/Tn1CR0:F[3:0]), stop the timer operation first (Tn0CR1/Tn1CR1:STA = 0), then clear the interrupt flag (Tn0CR1/Tn1CR1:IF, IR), the interrupt enable bits (Tn0CR1/Tn1CR1:IE, Tn0CR0/Tn1CR0:IFE) and the buffer full flag (Tn0CR1/Tn1CR1:BF).
- In the case of using the input capture function, when both edges of the external input signal is selected as the timing at which the 8/16-bit composite timer captures a counter value (Tn0CR0/Tn1CR0:F[3:0] = 0b1100 or 0b1111) while "H" level external input signal is being input, the first falling edge will be ignored, no counter value will be transferred to the data register (Tn0DR/Tn1DR), and pulse width measurement completion/edge detection flag (Tn0CR1/Tn1CR1:IR) will not be set either.
  - In counter clear mode, the counter will not be cleared at the first falling edge and no data will be transferred to the data register either. The 8/16-bit composite timer will start the input capture operation from the next rising edge.
  - In counter free-run mode, no data will be transferred to the data register at the first falling edge. The 8/16-bit composite timer will start the input capture operation from the next rising edge.
- In 8-bit operating mode (TMCRn:MOD = 0) of the PWM timer function (variable-cycle mode), when modifying the 8/16-bit composite timer data registers (Tn0DR and Tn1DR) during counter operation, modify the Tn1DR register first and then the Tn0DR register.
- The counter stops operating while holding the value when the microcontroller transits to stop mode or watch mode. When the stop mode or watch mode is released by an interrupt, the counter resumes operating with the last value that it holds (see Figure 12.15-1 and Figure 12.15-2). Therefore, the first interval time or the initial external clock count value is incorrect. Always initialize the counter value after the microcontroller is released from stop mode or watch mode.

Figure 12.15-1 Operations of Counter in Standby Mode or in Pause (Not Serving as PWM Timer)

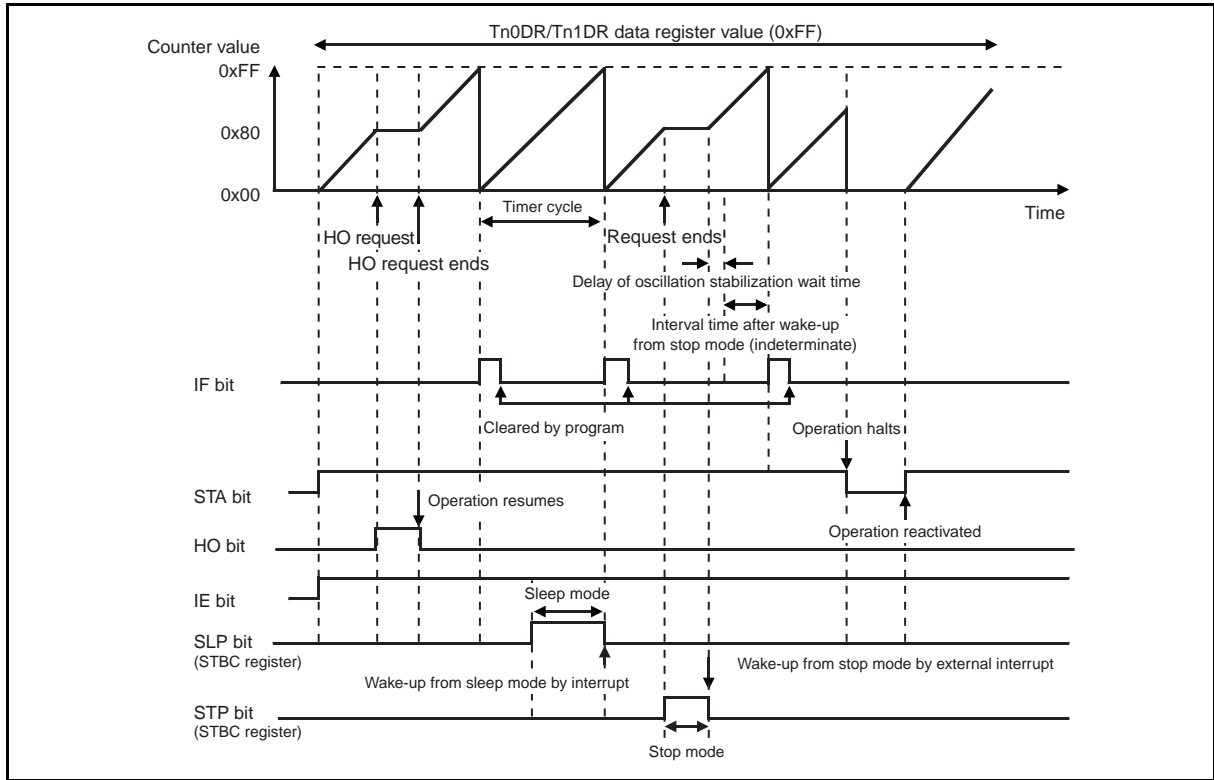
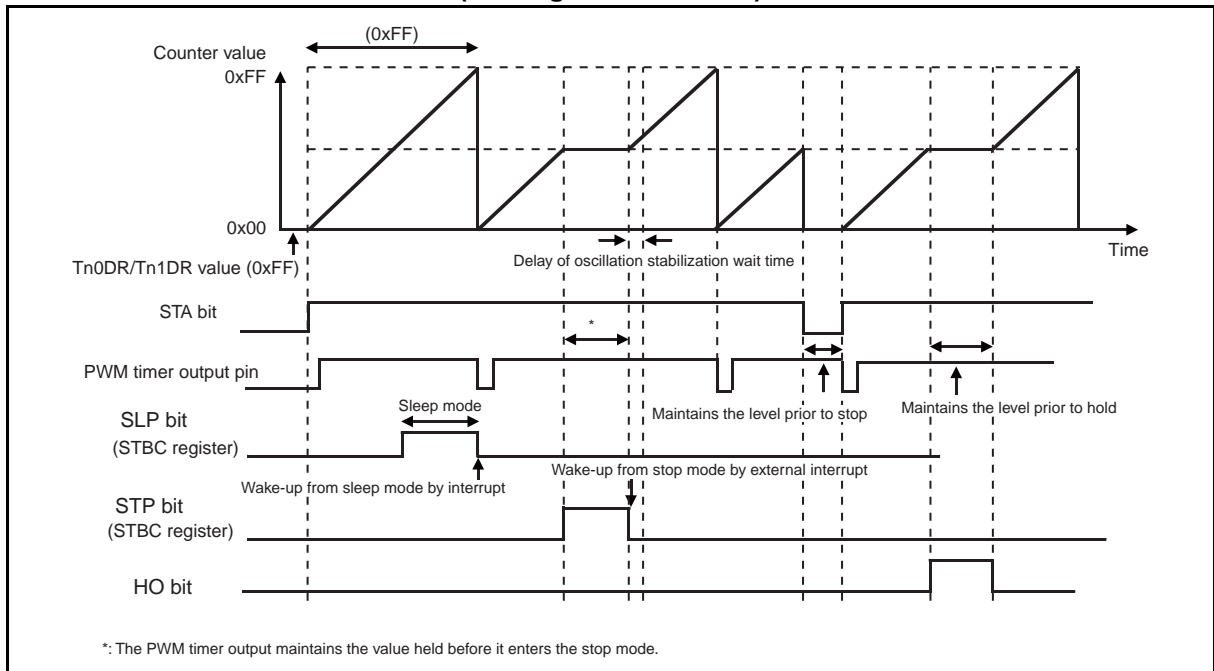


Figure 12.15-2 Operations of Counter in Standby Mode or in Pause (Serving as PWM Timer)







# **CHAPTER 13**

---

# ***EXTERNAL INTERRUPT CIRCUIT***

**This chapter describes the functions and operations of the external interrupt circuit.**

- 13.1 Overview
- 13.2 Configuration
- 13.3 Channels
- 13.4 Pin
- 13.5 Interrupt
- 13.6 Operations and Setting Procedure Example
- 13.7 Register
- 13.8 Notes on Using External Interrupt Circuit

## **13.1 Overview**

---

**The external interrupt circuit detects edges on the signal that is input to the external interrupt pin, and outputs interrupt requests to the interrupt controller.**

---

### **■ Function of External Interrupt Circuit**

The external interrupt circuit detects any edge of a signal that is input to an external interrupt pin and generates interrupt requests to the interrupt controller. The interrupt generated according to this interrupt request can cause the device to wake up from standby mode and return to its normal operating state. Therefore, the operating mode of the device can be changed when a signal is input to the external interrupt pin.

# MB95850K/860K/870K Series

## 13.2 Configuration

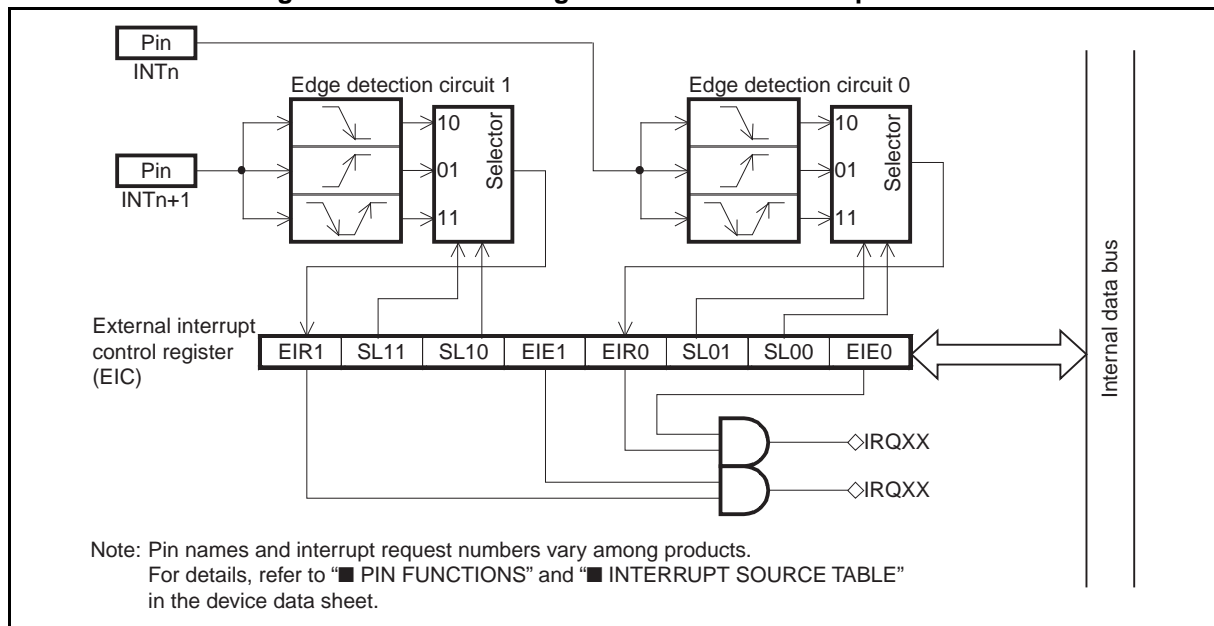
The external interrupt circuit consists of the following blocks:

- Edge detection circuit
- External interrupt control register

### ■ Block Diagram of External Interrupt Circuit

Figure 13.2-1 is the block diagram of the external interrupt circuit.

Figure 13.2-1 Block Diagram of External Interrupt Circuit



#### ● Edge detection circuit

When the polarity of the edge detected on a signal input to an external interrupt circuit pin (INT) matches the polarity of the edge selected in the interrupt control register (EIC), a corresponding external interrupt request flag bit (EIR) is set to "1".

#### ● External interrupt control register (EIC)

This register is used to select an edge, enable or disable interrupt requests, check for interrupt requests, etc.

## 13.3 Channels

---

**This section describes the channels of the external interrupt circuit.**

---

### ■ Channels of External Interrupt Circuit

Table 13.3-1 shows the pins and their corresponding registers of the external interrupt.

**Table 13.3-1 Pins and Register of External Interrupt Circuit**

| Pin name | Pin function                     | Corresponding register                    |
|----------|----------------------------------|---|
| INTn     | External interrupt input ch. n   | External interrupt control register (EIC) |
| INTn+1   | External interrupt input ch. n+1 |   |
| INTn+2   | External interrupt input ch. n+2 |   |
| INTn+3   | External interrupt input ch. n+3 |   |

The number of external interrupt circuit units varies among products. For the number of external interrupt circuit units in an individual product, refer to the device data sheet.

## MB95850K/860K/870K Series

### 13.4 Pin

---

**This section provides details of the pin of the external interrupt circuit.**

---

#### ■ Pin of External Interrupt Circuit

- INT pin

This pin serves both as an external interrupt input pin and as a general-purpose I/O port.

If the INT pin is set as an input port by the port direction register (DDR) and the corresponding external interrupt input is enabled by the external interrupt control register (EIC), that pin functions as an external interrupt input pin (INT).

The state of a pin can always be read from the port data register (PDR) when that pin is set as an input port. The value of PDR can be read by using the read-modify-write (RMW) type of instruction.

## **13.5 Interrupt**

---

**The interrupt source for the external interrupt circuit is the detection of the specified edge of the signal input to an external interrupt pin.**

---

### **■ Interrupt during Operation of External Interrupt Circuit**

When the specified edge of external interrupt input is detected, the corresponding external interrupt request flag bit (EIC:EIR0 or EIR1) is set to "1". In this case, if the interrupt request enable bit (EIC:EIE0 or EIE1 = 1) corresponding to that external interrupt request flag bit is enabled, an interrupt request is generated to the interrupt controller. In an interrupt service routine, write "0" to the external interrupt request flag bit corresponding to that interrupt request generated to clear the interrupt request.

## 13.6 Operations and Setting Procedure Example

This section describes the operations of the external interrupt circuit.

### ■ Operations of External Interrupt Circuit

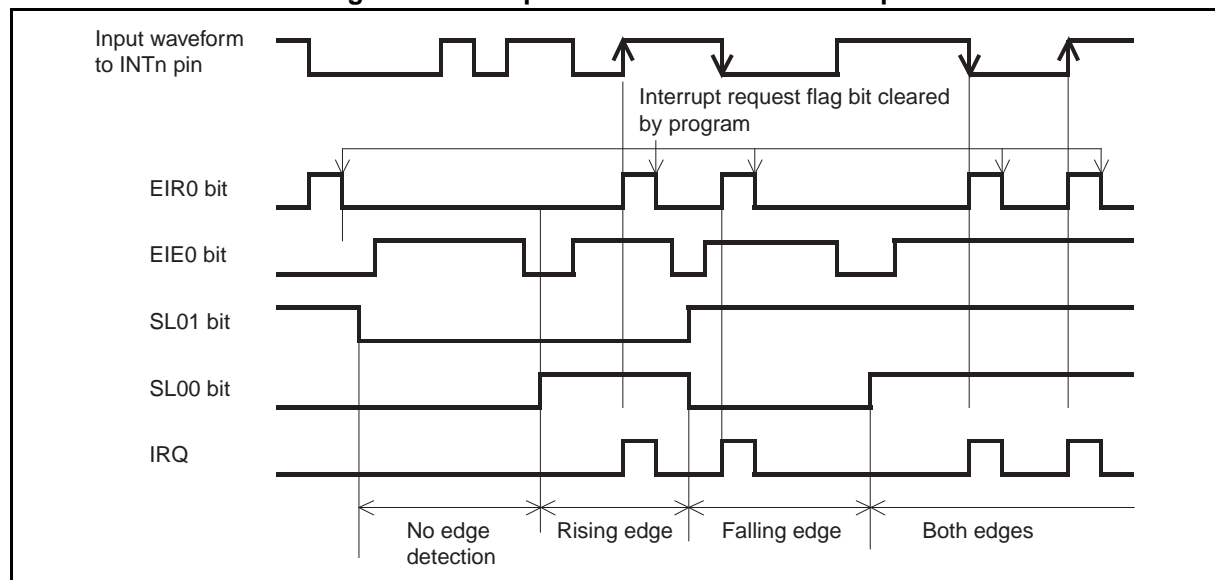
When the polarity of an edge of a signal input from one of the external interrupt pins (INTn, INTn+1) matches the polarity of the edge selected by the external interrupt control register (EIC:SL0[1:0] or SL1[1:0]), the corresponding external interrupt request flag bit (EIC:EIR0 or EIR1) is set to "1" and the interrupt request is generated.

Always set the interrupt request enable bit to "0" when not using an external interrupt to wake up the device from standby mode.

When setting the edge polarity select bit (SL00, SL01 or SL10, SL11), set the interrupt request enable bit (EIE0 or EIE1) to "0" to prevent the interrupt request from being generated accidentally. Also clear the interrupt request flag bit (EIR0 or EIR1) to "0" after changing the edge polarity.

Figure 13.6-1 shows the operations for setting the INTn pin as an external interrupt input.

Figure 13.6-1 Operations of External Interrupt



Example

**■ Setting Procedure Example**

Below is an example of procedure for setting the external interrupt circuit.

**● Initial settings**

1. Set the interrupt level. (ILR\*)
2. Select the edge polarity. (EIC:SL0[1:0])
3. Enable interrupt requests. (EIC:EIE0 = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

**● Interrupt processing**

1. Clear the interrupt request flag. (EIC:EIR0 = 0)
2. Process any interrupt.

---

**Note:**

An external interrupt input port shares the same pin with a general-purpose I/O port. Therefore, when using the pin as an external interrupt input port, set the bit in the port direction register (DDR) corresponding to that pin to "0" (input).

---



# MB95850K/860K/870K Series

## 13.7 Register

This section describes the register of the external interrupt circuit.

**Table 13.7-1 List of External Interrupt Circuit Register**

| Register abbreviation | Register name                       | Reference |
|-----------------------|-------------------------------------|-----------|
| EIC                   | External interrupt control register | 13.7.1    |

## 13.7.1 External Interrupt Control Register (EIC)

The external interrupt control register (EIC) is used to select the edge polarity for the external interrupt input and control interrupts.

### Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | EIR1 | SL11 | SL10 | EIE1 | EIR0 | SL01 | SL00 | EIE0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

### Register Functions

#### [bit7] EIR1: External interrupt request flag bit 1

This flag is set to "1" when the edge selected by the edge polarity select bits 1 (SL1[1:0]) is input to the external interrupt pin INTn+1.

When this bit and the interrupt request enable bit 1 (EIE1) are set to "1", an interrupt request is output.

Writing "0" clears this bit. Writing "1" has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit7        | Details   |
|-------------|---|
| Reading "0" | Indicates that the specified edge has not been input. |
| Reading "1" | Indicates that the specified edge has been input.     |
| Writing "0" | Clears this bit.                                      |
| Writing "1" | Has no effect on operation.                           |

#### [bit6:5] SL1[1:0]: Edge polarity select bits 1

These bits select the polarity of an edge of the pulse input to the external interrupt pin INTn+1. The edge selected is to be the interrupt source.

If these bits are set to "0b00", no edge detection is performed and no interrupt request is made.

If these bits are set to "0b01", rising edges are to be detected; if "0b10", falling edges are to be detected; if "0b11", both edges are to be detected.

| bit6:5       | Details           |
|--------------|-------------------|
| Writing "00" | No edge detection |
| Writing "01" | Rising edge       |
| Writing "10" | Falling edge      |
| Writing "11" | Both edges        |

#### [bit4] EIE1: Interrupt request enable bit 1

This bit enables or disables outputting the interrupt request to the interrupt controller. When this bit and the external interrupt request flag bit 1 (EIR1) are "1", an interrupt request is output.

When using an external interrupt pin, write "0" to the corresponding bit in the port direction register (DDR) to set the pin as an input port.

The status of the external interrupt pin can be read directly from the port data register, regardless of the status of the interrupt request enable bit.

| bit4        | Details                                    |
|-------------|--|
| Writing "0" | Disables outputting the interrupt request. |
| Writing "1" | Enables outputting the interrupt request.  |

## MB95850K/860K/870K Series

### [bit3] EIR0: External interrupt request flag bit 0

This flag is set to "1" when the edge selected by the edge polarity select bits 0 (SL0[1:0]) is input to the external interrupt pin INTn.

When this bit and the interrupt request enable bit 0 (EIE0) are set to "1", an interrupt request is output.

Writing "0" clears this bit. Writing "1" has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit3               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the specified edge has not been input. |
| <b>Reading "1"</b> | Indicates that the specified edge has been input.     |
| <b>Writing "0"</b> | Clears this bit.                                      |
| <b>Writing "1"</b> | Has no effect on operation.                           |

### [bit2:1] SL0[1:0]: Edge polarity select bits 0

These bits select the polarity of an edge of the pulse input to the external interrupt pin INTn. The edge selected is to be the interrupt source.

If these bits are set to "0b00", no edge detection is performed and no interrupt request is made.

If these bits are set to "0b01", rising edges are to be detected; if "0b10", falling edges are to be detected; if "0b11", both edges are to be detected.

| bit2:1              | Details           |
|---------------------|-------------------|
| <b>Writing "00"</b> | No edge detection |
| <b>Writing "01"</b> | Rising edge       |
| <b>Writing "10"</b> | Falling edge      |
| <b>Writing "11"</b> | Both edges        |

### [bit0] EIE0: Interrupt request enable bit 0

This bit enables or disables outputting the interrupt request to the interrupt controller. When this bit and the external interrupt request flag bit 0 (EIR0) are "1", an interrupt request is output.

When using an external interrupt pin, write "0" to the corresponding bit in the port direction register (DDR) to set the pin as an input port.

The status of the external interrupt pin can be read directly from the port data register, regardless of the status of the interrupt request enable bit.

| bit0               | Details                                    |
|--------------------|--|
| <b>Writing "0"</b> | Disables outputting the interrupt request. |
| <b>Writing "1"</b> | Enables outputting the interrupt request.  |

## 13.8 Notes on Using External Interrupt Circuit

---

This section provides notes on using the external interrupt circuit.

---

### ■ Notes on Using External Interrupt Circuit

- Before setting the edge polarity select bits (SL0[1:0] or SL1[1:0]), set the interrupt request enable bit (EIE0 or EIE1) to "0" (disabling interrupt requests). In addition, clear the external interrupt request flag bit (EIR0 or EIR1) to "0" after setting the edge polarity.
- The device cannot wake up from the interrupt service routine if the external interrupt request flag bit is "1" and the interrupt request enable bit is enabled. In the interrupt service routine, always clear the external interrupt request flag bit.

# **CHAPTER 14**

---

# **INTERRUPT PIN SELECTION CIRCUIT**

**This chapter describes the functions and operations of the interrupt pin selection circuit.**

- 14.1 Overview
- 14.2 Configuration
- 14.3 Pins
- 14.4 Operation
- 14.5 Register
- 14.6 Notes on Using Interrupt Pin Selection Circuit

## 14.1 Overview

---

**The interrupt pin selection circuit selects pins to be used as interrupt input pins from among various peripheral input pins.**

---

### ■ Interrupt Pin Selection Circuit

The interrupt pin selection circuit is used to select interrupt input pins from among various peripheral inputs (EC0, EC1, INT00, UCK0 and UI0)\*. The input signal from each peripheral function pin is selected by this circuit and the signal is used as the INT00 (ch. 0) input of external interrupt. This enables the input signal from each peripheral function pin to also serve as an external interrupt pin.

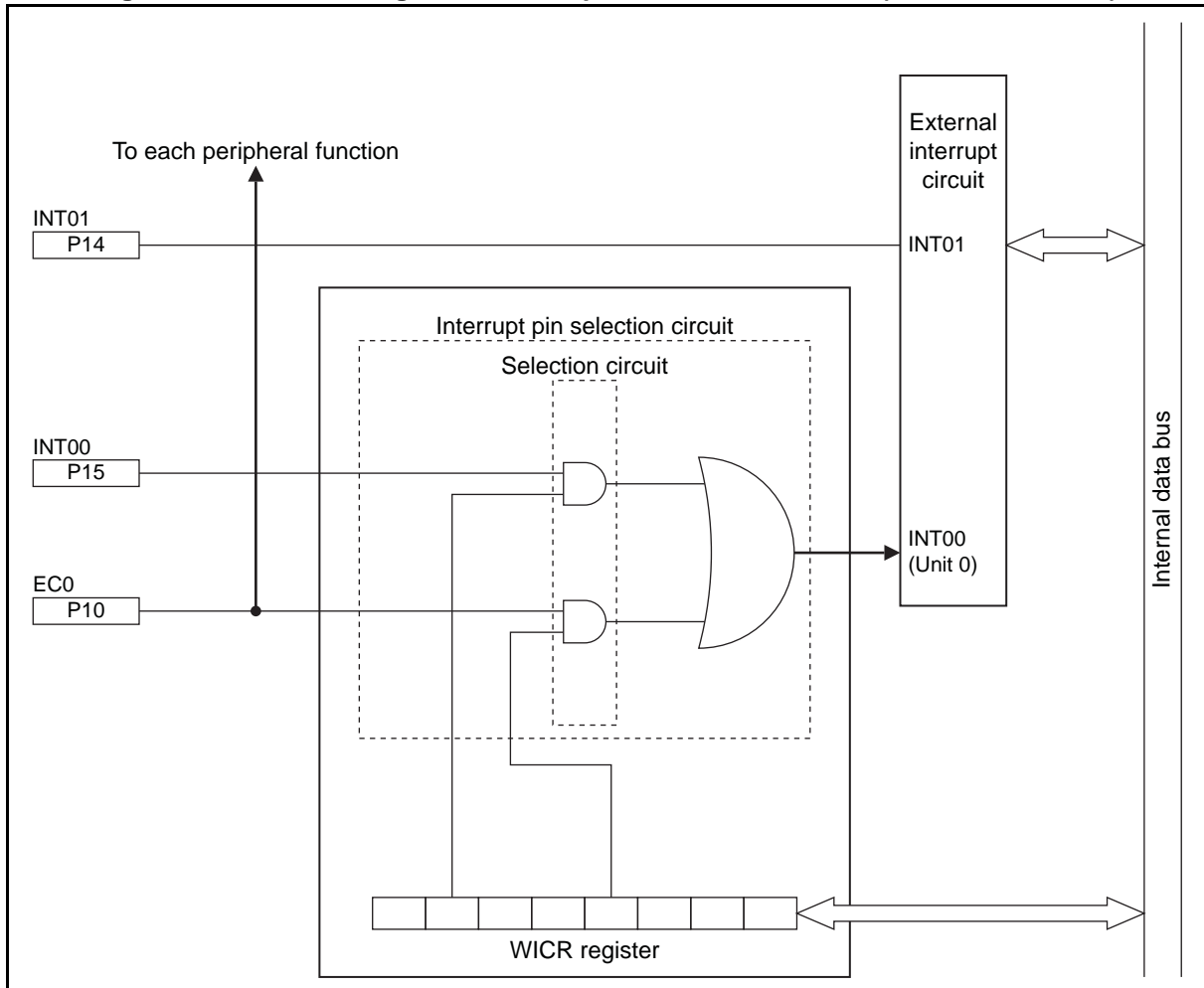
- \*: The available peripheral inputs of the MB95850K Series are EC0 and INT00.
- The available peripheral inputs of the MB95860K Series are EC0, EC1 and INT00.
- The available peripheral inputs of the MB95870K Series are EC0, EC1, INT00, UCK0 and UI0.

## 14.2 Configuration

Figure 14.2-1 to Figure 14.2-3 show the block diagrams of the interrupt pin selection circuit.

### ■ Block Diagrams of Interrupt Pin Selection Circuit

Figure 14.2-1 Block Diagram of Interrupt Pin Selection Circuit (MB95850K Series)



**Figure 14.2-2 Block Diagram of Interrupt Pin Selection Circuit (MB95860K Series)**

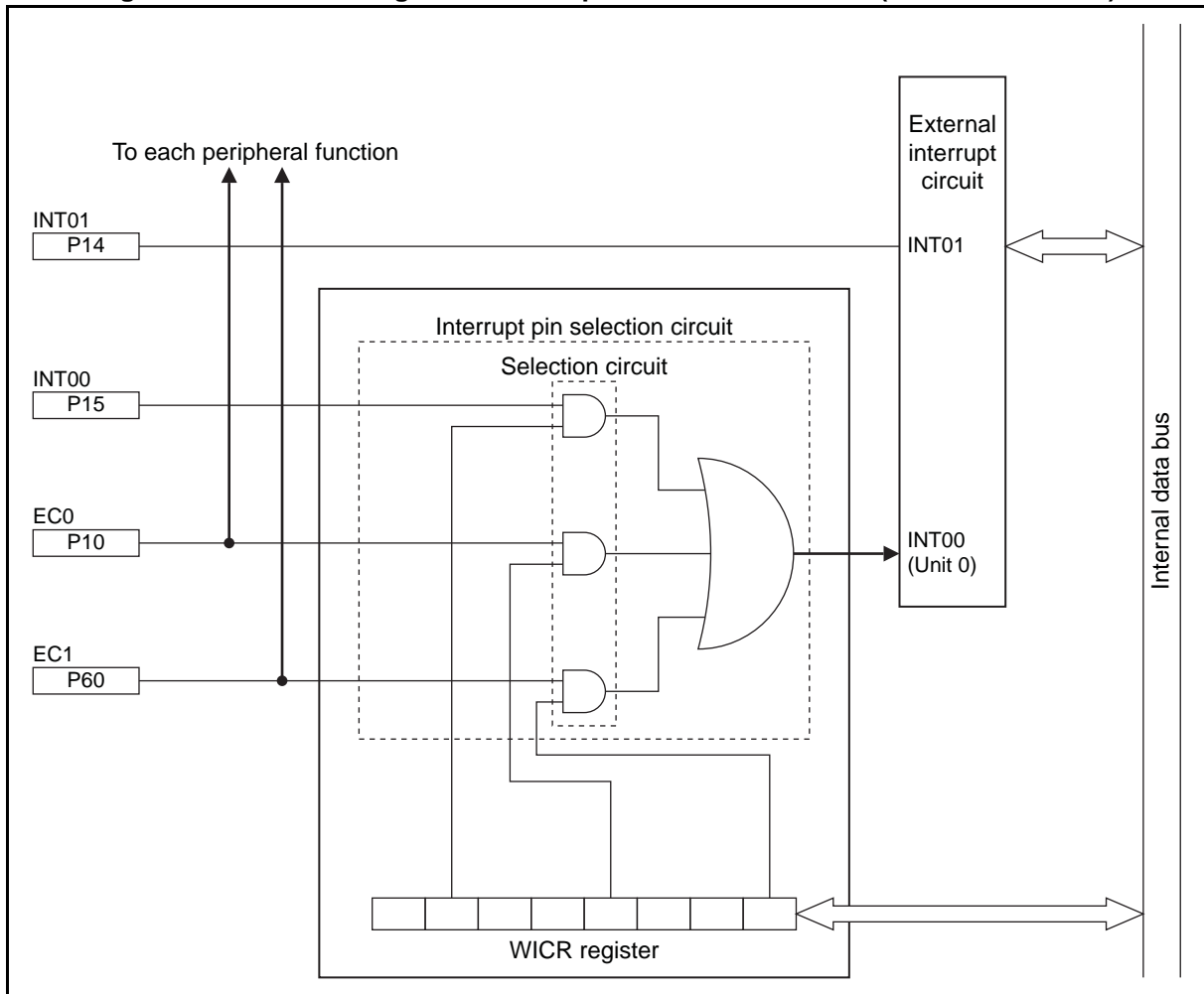
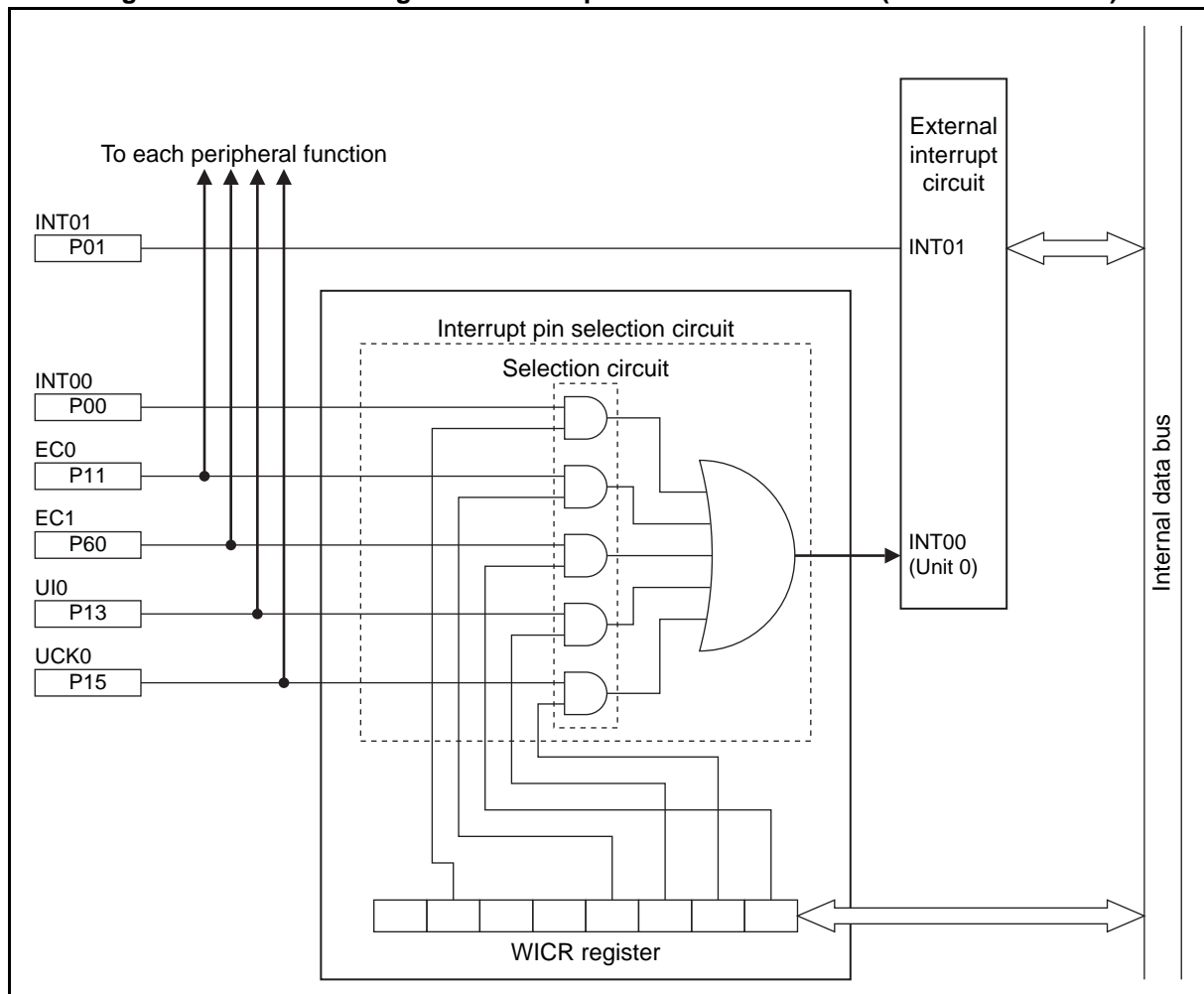




Figure 14.2-3 Block Diagram of Interrupt Pin Selection Circuit (MB95870K Series)



- **WICR register (interrupt pin selection circuit control register)**  
 This register is used to determine which of the available peripheral input pins should be output to the interrupt circuit and which interrupt pins they should serve as.
- **Selection circuit**  
 This circuit outputs the input from the pin selected by the WICR register to the INT00 input of the external interrupt circuit (ch. 0).

## 14.3 Pins

This section describes the pins of the interrupt pin selection circuit.

### ■ Pins of Interrupt Pin Selection Circuit

The peripheral function pins of the interrupt pin selection circuit are the EC0, EC1, INT00, UCK0 and UI0 pins\*. These input pins (except INT00) are also connected to their respective peripheral units in parallel and can be used for both functions simultaneously. Table 14.3-1 to Table 14.3-3 show the correspondence between the peripheral functions and peripheral input pins.

\*: The available peripheral function pins of the MB95850K Series are EC0 and INT00.

The available peripheral function pins of the MB95860K Series are EC0, EC1 and INT00.

The available peripheral function pins of the MB95870K Series are EC0, EC1, INT00, UCK0 and UI0.

**Table 14.3-1 Correspondence between Peripheral Functions and Peripheral Input Pins (MB95850K Series)**

| Peripheral input pin name | Peripheral function name               |
|---------------------------|--|
| EC0                       | 8/16-bit composite timer (event input) |
| INT00                     | Interrupt pin selection circuit        |

**Table 14.3-2 Correspondence between Peripheral Functions and Peripheral Input Pins (MB95860K Series)**

| Peripheral input pin name | Peripheral function name               |
|---------------------------|--|
| EC0                       | 8/16-bit composite timer (event input) |
| EC1                       | 8/16-bit composite timer (event input) |
| INT00                     | Interrupt pin selection circuit        |

**Table 14.3-3 Correspondence between Peripheral Functions and Peripheral Input Pins (MB95870K Series)**

| Peripheral input pin name | Peripheral function name               |
|---------------------------|--|
| EC0                       | 8/16-bit composite timer (event input) |
| EC1                       | 8/16-bit composite timer (event input) |
| INT00                     | Interrupt pin selection circuit        |
| UCK0                      | UART/SIO (clock I/O)                   |
| UI0                       | UART/SIO (data input)                  |

## 14.4 Operation

---

**The interrupt pins are selected by setting the WICR register.**

---

### ■ Operation of Interrupt Pin Selection Circuit

The WICR register selects the input pins to be input to INT00 of the external interrupt circuit (ch. 0). Below is a setup procedure for the interrupt pin selection circuit and external interrupt circuit (ch. 0), which must be followed when selecting the EC0 pin as an interrupt pin.

1. Write "0" to the corresponding bit in the port direction register (DDR) to set the pin as an input.
2. Select the EC0 pin as an interrupt input pin in the WICR register.
  - Write "0x08" to the WICR register. At this point, write "0" to the EIE0 bit in the EIC00 register of the external interrupt circuit to disable the operation of the external interrupt circuit.
3. Enable the operation of INT00 of the external interrupt circuit (ch. 0).
  - Set the SL0[1:0] bits in the EIC00 register to any value other than "0b00" in the external interrupt circuit to select the valid edge. Also write "1" to the EIE0 bit to enable interrupts.
4. The subsequent interrupt operation is the same as that of the external interrupt circuit.
  - When a reset is released, the WICR register is initialized to "0x40" and the INT00 bit is selected as the only available interrupt pin. To use any pins other than the INT00 pin as external interrupt pins, modify the settings of this register before enabling the operation of the external interrupt circuit.

## 14.5 Register

---

**This section describes the register of the interrupt pin selection circuit.**

---

**Table 14.5-1 List of External Interrupt Circuit Register**

| Register abbreviation | Register name                                    | Reference |
|-----------------------|--|-----------|
| WICR                  | Interrupt pin selection circuit control register | 14.5.1    |

## 14.5.1 Interrupt Pin Selection Circuit Control Register (WICR)

This register is used to determine which of the available peripheral input pins should be output to the interrupt circuit and which interrupt pins they should serve as.

### ■ Register Configuration (MB95850K Series)

| bit           | 7 | 6     | 5        | 4        | 3   | 2        | 1        | 0        |
|---------------|---|-------|----------|----------|-----|----------|----------|----------|
| Field         | — | INT00 | Reserved | Reserved | EC0 | Reserved | Reserved | Reserved |
| Attribute     | — | R/W   | W        | W        | R/W | W        | W        | W        |
| Initial value | 0 | 1     | 0        | 0        | 0   | 0        | 0        | 0        |

### ■ Register Functions (MB95850K Series)

#### [bit7] Undefined bit

The read value is always "0". Writing a value to this bit has no effect on operation.

#### [bit6] INT00: INT00 interrupt pin select bit

This bit determines whether to select the INT00 pin as an interrupt input pin.

Writing "0" to this bit deselects the INT00 pin as an interrupt input pin, and the interrupt pin selection circuit treats the INT00 pin input as being fixed at "0".

Writing "1" to this bit selects the INT00 pin as an interrupt input pin and the circuit passes the INT00 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the INT00 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit6               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the INT00 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the INT00 pin as an interrupt input pin.   |

#### [bit5:4] Reserved bits

Always set these bits to "0".

#### [bit3] EC0: EC0 interrupt pin select bit

This bit determines whether to select the EC0 pin as an interrupt input pin.

Writing "0" to this bit deselects the EC0 pin as an interrupt input pin, and the interrupt pin selection circuit treats the EC0 pin input as being fixed at "0".

Writing "1" to this bit selects the EC0 pin as an interrupt input pin and the circuit passes the EC0 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the EC0 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit3               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the EC0 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the EC0 pin as an interrupt input pin.   |

#### [bit2:0] Reserved bits

Always set these bits to "0".

When any of these bits (except undefined bits) is set to "1" and the operation of INT00 (ch. 0) of the external interrupt circuit is enabled in standby mode of this device, the selected pin is enabled to perform input operation. The device wakes up from the standby mode when a valid

edge pulse is input to the selected pin. For detail of the standby mode, see "3.5 Operations in Low Power Consumption Mode (Standby Mode)".

### ■ Register Configuration (MB95860K Series)

| bit           | 7 | 6     | 5        | 4        | 3   | 2        | 1        | 0   |
|---------------|---|-------|----------|----------|-----|----------|----------|-----|
| Field         | — | INT00 | Reserved | Reserved | EC0 | Reserved | Reserved | EC1 |
| Attribute     | — | R/W   | W        | W        | R/W | W        | W        | R/W |
| Initial value | 0 | 1     | 0        | 0        | 0   | 0        | 0        | 0   |

### ■ Register Functions (MB95860K Series)

#### [bit7] Undefined bit

The read value is always "0". Writing a value to this bit has no effect on operation.

#### [bit6] INT00: INT00 interrupt pin select bit

This bit determines whether to select the INT00 pin as an interrupt input pin.

Writing "0" to this bit deselects the INT00 pin as an interrupt input pin, and the interrupt pin selection circuit treats the INT00 pin input as being fixed at "0".

Writing "1" to this bit selects the INT00 pin as an interrupt input pin and the circuit passes the INT00 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the INT00 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit6               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the INT00 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the INT00 pin as an interrupt input pin.   |

#### [bit5:4] Reserved bits

Always set these bits to "0".

#### [bit3] EC0: EC0 interrupt pin select bit

This bit determines whether to select the EC0 pin as an interrupt input pin.

Writing "0" to this bit deselects the EC0 pin as an interrupt input pin, and the interrupt pin selection circuit treats the EC0 pin input as being fixed at "0".

Writing "1" to this bit selects the EC0 pin as an interrupt input pin and the circuit passes the EC0 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the EC0 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit3               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the EC0 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the EC0 pin as an interrupt input pin.   |

#### [bit2:1] Reserved bits

Always set these bits to "0".

#### [bit0] EC1: EC1 interrupt pin select bit

This bit determines whether to select the EC1 pin as an interrupt input pin.

Writing "0" to this bit deselects the EC1 pin as an interrupt input pin, and the interrupt pin selection circuit treats the EC1 pin input as being fixed at "0".

Writing "1" to this bit selects the EC1 pin as an interrupt input pin and the circuit passes the EC1 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the EC1 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the EC1 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the EC1 pin as an interrupt input pin.   |

When any of these bits (except undefined bits) is set to "1" and the operation of INT00 (ch. 0) of the external interrupt circuit is enabled in standby mode of this device, the selected pin is

enabled to perform input operation. The device wakes up from the standby mode when a valid edge pulse is input to the selected pin. For detail of the standby mode, see "3.5 Operations in Low Power Consumption Mode (Standby Mode)".



### ■ Register Configuration (MB95870K Series)

| bit           | 7 | 6     | 5        | 4        | 3   | 2   | 1    | 0   |
|---------------|---|-------|----------|----------|-----|-----|------|-----|
| Field         | — | INT00 | Reserved | Reserved | EC0 | UI0 | UCK0 | EC1 |
| Attribute     | — | R/W   | W        | W        | R/W | R/W | R/W  | R/W |
| Initial value | 0 | 1     | 0        | 0        | 0   | 0   | 0    | 0   |

### ■ Register Functions (MB95870K Series)

#### [bit7] Undefined bit

The read value is always "0". Writing a value to this bit has no effect on operation.

#### [bit6] INT00: INT00 interrupt pin select bit

This bit determines whether to select the INT00 pin as an interrupt input pin.

Writing "0" to this bit deselects the INT00 pin as an interrupt input pin, and the interrupt pin selection circuit treats the INT00 pin input as being fixed at "0".

Writing "1" to this bit selects the INT00 pin as an interrupt input pin and the circuit passes the INT00 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the INT00 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit6               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the INT00 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the INT00 pin as an interrupt input pin.   |

#### [bit5:4] Reserved bits

Always set these bits to "0".

#### [bit3] EC0: EC0 interrupt pin select bit

This bit determines whether to select the EC0 pin as an interrupt input pin.

Writing "0" to this bit deselects the EC0 pin as an interrupt input pin, and the interrupt pin selection circuit treats the EC0 pin input as being fixed at "0".

Writing "1" to this bit selects the EC0 pin as an interrupt input pin and the circuit passes the EC0 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the EC0 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit3               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the EC0 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the EC0 pin as an interrupt input pin.   |

#### [bit2] UI0: UI0 interrupt pin select bit

This bit determines whether to select the UI0 pin as an interrupt input pin.

Writing "0" to this bit deselects the UI0 pin as an interrupt input pin, and the interrupt pin selection circuit treats the UI0 pin input as being fixed at "0".

Writing "1" to this bit selects the UI0 pin as an interrupt input pin and the circuit passes the UI0 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the UI0 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit2               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Deselects the UI0 pin as the interrupt input pin. |
| <b>Writing "1"</b> | Selects the UI0 pin as the interrupt input pin.   |

[bit1] UCK0: UCK0 interrupt pin select bit

This bit determines whether to select the UCK0 pin as an interrupt input pin.

Writing "0" to this bit deselects the UCK0 pin as an interrupt input pin, and the interrupt pin selection circuit treats the UCK0 pin input as "0".

Writing "1" to this bit selects the UCK0 pin as an interrupt input pin and the circuit passes the UCK0 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the UCK0 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit1               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the UCK0 pin as the interrupt input pin. |
| <b>Writing "1"</b> | Selects the UCK0 pin as the interrupt input pin.   |

[bit0] EC1: EC1 interrupt pin select bit

This bit determines whether to select the EC1 pin as an interrupt input pin.

Writing "0" to this bit deselects the EC1 pin as an interrupt input pin, and the interrupt pin selection circuit treats the EC1 pin input as being fixed at "0".

Writing "1" to this bit selects the EC1 pin as an interrupt input pin and the circuit passes the EC1 pin input to INT00 (ch. 0) of the external interrupt circuit. In this case, the input signal to the EC1 pin can generate an external interrupt if INT00 (ch. 0) operation is enabled in the external interrupt circuit.

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Deselects the EC1 pin as an interrupt input pin. |
| <b>Writing "1"</b> | Selects the EC1 pin as an interrupt input pin.   |

When any of these bits (except undefined bits) is set to "1" and the operation of INT00 (ch. 0) of the external interrupt circuit is enabled in standby mode of this device, the selected pin is enabled to perform input operation. The device wakes up from the standby mode when a valid edge pulse is input to the selected pin. For detail of the standby mode, see "3.5 Operations in Low Power Consumption Mode (Standby Mode)".

**Note:**

The input signals to the peripheral pins do not generate an external interrupt even when "1" is written to these bits if the INT00 (ch. 0) of the external interrupt circuit is disabled.

Do not modify the values of these bits while the INT00 (ch. 0) of the external interrupt circuit is enabled. If modified, the external interrupt circuit may detect a valid edge, depending on the pin input level.

If multiple interrupt pins are selected in the WICR register simultaneously and the operation of INT00 (ch. 0) of the external interrupt circuit is enabled (the values other than "0b00" are written to the SL0[1:0] bits in the EIC register of the external interrupt circuit.), the selected pins will remain enabled to perform input so as to accept interrupts even in standby mode.

## 14.6 Notes on Using Interrupt Pin Selection Circuit

---

**This section provides notes on using the interrupt pin selection circuit.**

---

- If multiple interrupt pins are selected in the WICR register simultaneously and the operation of INT00 (ch. 0) of the external interrupt circuit is enabled (a value other than "0b00" is written to the SLO[1:0] bits in the EIC register of the external interrupt circuit to select a valid edge, and "1" is written to the EIE0 bit to enable the interrupt request.), the input to the selected pins will remain enabled so as to accept interrupts even in standby mode.
- If multiple interrupt pins are selected in the WICR register simultaneously, an input to INT00 (ch. 0) of the external interrupt circuit is treated as "H" if any of the selected input signals is "H" (It becomes "OR" of the signals input to the selected pins).



# **CHAPTER 15**

---

## **8/10-BIT A/D CONVERTER**

**This chapter describes the functions and operations of the 8/10-bit A/D converter.**

- 15.1 Overview
- 15.2 Configuration
- 15.3 Pin
- 15.4 Interrupt
- 15.5 Operations and Setting Procedure Example
- 15.6 Registers
- 15.7 Notes on Using 8/10-bit A/D Converter

## 15.1 Overview

---

**The 8/10-bit A/D converter is a 10-bit successive approximation type of 8/10-bit A/D converter. It can be started by the software and internal clock, with one input signal selected from multiple analog input pins.**

---

### ■ A/D Conversion Function

The A/D converter converts analog voltage (input voltage) input through an analog input pin to an 8-bit or 10-bit digital value.

- The input signal can be selected from multiple analog input pins.
- The conversion speed can be set in a program. (can be selected according to operating voltage and frequency).
- An interrupt is generated when A/D conversion is completed.
- The completion of conversion can be determined according to the ADI bit in the ADC1 register.

To activate the A/D conversion function, use one of the following methods.

- Activation using the AD bit in the ADC1 register
- Continuous activation using the 8/16-bit composite timer output TO00

# MB95850K/860K/870K Series

## 15.2 Configuration

The 8/10-bit A/D converter consists of the following blocks:

- Clock selector (input clock selector for starting A/D conversion)
- Analog channel selector
- Sample-and-hold circuit
- Control circuit
- 8/10-bit A/D converter data register (upper/lower) (ADDH/ADDL)
- 8/10-bit A/D converter control register 1 (ADC1)
- 8/10-bit A/D converter control register 2 (ADC2)

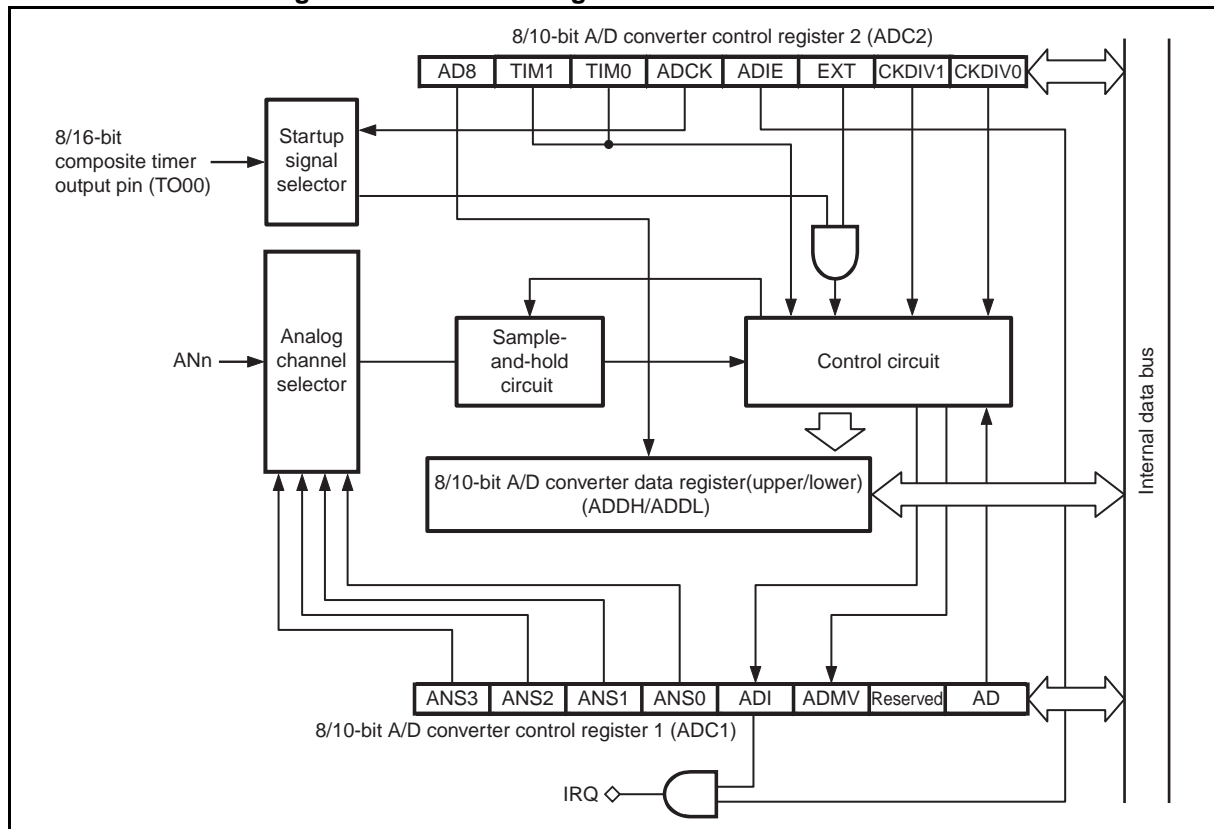
The number of analog input pins and that of analog channels of the 8/10-bit A/D converter vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name represents the analog input pin number. For details of pin names of a product, refer to the device data sheet.

### ■ Block Diagram of 8/10-bit A/D Converter

Figure 15.2-1 is the block diagram of the 8/10-bit A/D converter.

Figure 15.2-1 Block Diagram of 8/10-bit A/D Converter



- Clock selector

This selects the A/D conversion clock with continuous activation having been enabled (ADC2:EXT = 1).

- Analog channel selector

This is the circuit selecting an input channel from several analog input pins.

- Sample-and-hold circuit

This circuit holds input voltage selected by the analog channel selector. By sampling the input voltage and holding it immediately after A/D conversion starts, this circuit prevents A/D conversion from being affected by the fluctuation in input voltage during the conversion (comparison).

- Control circuit

The A/D conversion function determines the values in the 10-bit A/D data register sequentially from MSB to LSB based on the voltage compare signal from the comparator. When A/D conversion is completed, the A/D conversion function sets the interrupt request flag bit (ADC1:ADI) to "1".

- 8/10-bit A/D converter data register (upper/lower) (ADDH/ADDL)

The upper two bits of 10-bit A/D data are stored in the ADDH register; the lower eight bits in the ADDL register.

If the A/D conversion precision bit (ADC2:AD8) is set to "1", the A/D conversion precision becomes 8-bit precision, and the upper eight bits of 10-bit A/D data are to be stored in the ADDL register.

- 8/10-bit A/D converter control register 1 (ADC1)

This register is used to enable and disable different functions, select an analog input pin, and check the status of the A/D converter.

- 8/10-bit A/D converter control register 2 (ADC2)

This register is used to select an input clock, enable and disable interrupts and control different functions of the A/D converter.

## ■ Input Clock

The 8/10-bit A/D converter uses an output clock from the prescaler as the input clock (operating clock).



## MB95850K/860K/870K Series

### 15.3 Pin

---

**This section describes the pin of the 8/10-bit A/D converter.**

---

#### ■ Pin of 8/10-bit A/D Converter

The analog input pin is also used as a general-purpose I/O port.

- ANn pin

When using the A/D conversion function, input to the ANn pin the analog voltage to be converted. To use an ANn pin as an analog input pin, write "0" to the bit in the port direction register (DDR) corresponding to that pin, and the value corresponding to that pin to the analog input pin select bits (ADC1:ANS[3:0]). A pin not used as an analog input pin can be used as a general-purpose I/O port even when the 8/10-bit A/D converter is in use.

## **15.4 Interrupt**

---

**The completion of conversion during the operation of the A/D converter is an interrupt source of the 8/10-bit A/D converter.**

---

### **■ Interrupt During 8/10-bit A/D Converter Operation**

When A/D conversion is completed, the interrupt request flag bit (ADC1:ADI) is set to "1". Then if the interrupt request enable bit has been enabled (ADC2:ADIE = 1), an interrupt request is made to the interrupt controller. Write "0" to the ADI bit using the interrupt service routine to clear the interrupt request.

The ADI bit is set to "1" when A/D conversion is completed, irrespective of the value of the ADIE bit.

The CPU cannot return from interrupt processing if the interrupt request flag bit (ADC1:ADI) is "1" with interrupt requests having been enabled (ADC2:ADIE = 1). Always clear the ADI bit in the interrupt service routine.

# MB95850K/860K/870K Series

## 15.5 Operations and Setting Procedure Example

The 8/10-bit A/D converter can activate A/D conversion with the software or activate A/D conversion continuously according to the setting of the EXT bit in the ADC2 register.

### ■ Operations of 8/10-bit A/D Converter Conversion Function

- Software activation

To activate the A/D conversion function with the software, do the settings shown in Figure 15.5-1.

**Figure 15.5-1 Settings for A/D Conversion Function (Software Activation)**

|      | bit7                         | bit6      | bit5      | bit4      | bit3      | bit2      | bit1                         | bit0        |
|------|------------------------------|-----------|-----------|-----------|-----------|-----------|------------------------------|-------------|
| ADC1 | ANS3<br>⊙                    | ANS2<br>⊙ | ANS1<br>⊙ | ANS0<br>⊙ | ADI<br>⊙  | ADMV<br>⊙ | Reserved<br>0                | AD<br>1     |
| ADC2 | AD8<br>⊙                     | TIM1<br>⊙ | TIM0<br>⊙ | ADCK<br>x | ADIE<br>⊙ | EXT<br>0  | CKDIV1<br>⊙                  | CKDIV0<br>⊙ |
| ADDH | -                            | -         | -         | -         | -         | -         | A/D converted value retained |             |
| ADDL | A/D converted value retained |           |           |           |           |           |                              |             |

⊙: Bit to be used  
 x : Unused bit  
 0 : Set to "0"  
 1 : Set to "1"

When the A/D conversion function is activated, A/D conversion starts. In addition, the A/D conversion function can be re-activated even during conversion.

● Continuous activation

To execute continuous activation of the A/D conversion function, do the settings shown in Figure 15.5-2.

**Figure 15.5-2 Settings for A/D Conversion Function (Continuous Activation)**

|      | bit7                         | bit6 | bit5 | bit4 | bit3 | bit2 | bit1                         | bit0   |
|------|------------------------------|------|------|------|------|------|------------------------------|--------|
| ADC1 | ANS3                         | ANS2 | ANS1 | ANS0 | ADI  | ADMV | Reserved                     | AD     |
|      | ⊙                            | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    | 0                            | x      |
| ADC2 | AD8                          | TIM1 | TIM0 | ADCK | ADIE | EXT  | CKDIV1                       | CKDIV0 |
|      | ⊙                            | ⊙    | ⊙    | ⊙    | ⊙    | 1    | ⊙                            | ⊙      |
| ADDH | -                            | -    | -    | -    | -    | -    | A/D converted value retained |        |
| ADDL | A/D converted value retained |      |      |      |      |      |                              |        |

⊙ : Bit to be used  
 x : Unused bit  
 0 : Set to "0"  
 1 : Set to "1"

When continuous activation is enabled, the A/D conversion function is activated at the rising edge of the input clock selected to start A/D conversion. Continuous activation is stopped when disabled (ADC2:EXT = 0).

■ Operations of A/D Conversion Function

This section explains the operations of 8/10-bit A/D converter.

1. When A/D conversion is started, the conversion flag bit is set (ADC1:ADMV = 1) and the selected analog input pin is connected to the sample-and-hold circuit.
2. The voltage in the analog input pin is loaded into a sample-and-hold capacitor in the sample-and-hold circuit during the sampling cycle. This voltage is held until A/D conversion is completed.
3. The comparator in the control circuit compares the voltage loaded into sample-and-hold capacitor with the A/D conversion reference voltage, from the most significant bit (MSB) to the least significant bit (LSB), and then transfers the results to the ADDH and ADDL registers.

After the results have been transferred to the two registers, the conversion flag bit is cleared (ADC1:ADMV = 0) and the interrupt request flag bit is set to "1" (ADC1:ADI = 1).

Notes:

- The contents of the ADDH and ADDL registers are saved at the end of A/D conversion. Therefore, during A/D conversion, the values resulting from last conversion will be returned if the two registers are read.
- Do not change the analog input pin select bits (ADC1:ANS[3:0]) while AD conversion function is being used. During continuous activation in particular, disable continuous activation (ADC2:EXT = 0) before changing the analog input pin.
- A reset, or the start of the stop mode or watch mode causes the A/D converter to stop and the ADMV bit to be cleared to "0".

## ■ Setting Procedure Example

Below is an example of procedure for setting the 8/10-bit A/D converter:

### ● Initial settings

1. Set the input port. (DDR)
2. Set the interrupt level. (ILR\*)
3. Select an A/D input pin. (ADC1:ANS[3:0])
4. Set the sampling time. (ADC2:TIM[1:0])
5. Select the clock. (ADC2:CKDIV[1:0])
6. Set A/D conversion precision. (ADC2:AD8)
7. Select the operating mode. (ADC2:EXT)
8. Select the start trigger. (ADC2:ADCK)
9. Enable interrupts. (ADC2:ADIE = 1)
10. Activate the A/D conversion function. (ADC1:AD = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

### ● Interrupt processing

1. Clear the interrupt request flag to "0". (ADC1:ADI = 0)
2. Read converted values. (ADDH, ADDL)
3. Activate the A/D conversion function. (ADC1:AD = 1)

## 15.6 Registers

---

**This section describes the registers of the 8/10-bit A/D converter.**

---

**Table 15.6-1 List of 8/10-bit A/D Converter Registers**

| Register abbreviation | Register name                                | Reference |
|-----------------------|--|-----------|
| ADC1                  | 8/10-bit A/D converter control register 1    | 15.6.1    |
| ADC2                  | 8/10-bit A/D converter control register 2    | 15.6.2    |
| ADDH                  | 8/10-bit A/D converter data register (upper) | 15.6.3    |
| ADDL                  | 8/10-bit A/D converter data register (lower) | 15.6.3    |

## MB95850K/860K/870K Series

### 15.6.1 8/10-bit A/D Converter Control Register 1 (ADC1)

The 8/10-bit A/D converter control register 1 (ADC1) is used to enable and disable individual functions of the 8/10-bit A/D converter, select an analog input pin and check the status of the converter.

#### ■ Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3   | 2    | 1        | 0  |
|---------------|------|------|------|------|-----|------|----------|----|
| Field         | ANS3 | ANS2 | ANS1 | ANS0 | ADI | ADMV | Reserved | AD |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W | R    | W        | W  |
| Initial value | 0    | 0    | 0    | 0    | 0   | 0    | 0        | 0  |

#### ■ Register Functions

[bit7:4] ANS[3:0]: Analog input pin select bits

These bits select an analog input pin.

When A/D conversion is started (AD = 1) by the software (ADC2: EXT = 0), these bits can be modified simultaneously.

| bit7:4         | Details* |
|----------------|----------|
| Writing "0000" | AN00 pin |
| Writing "0001" | AN01 pin |
| Writing "0010" | AN02 pin |
| Writing "0011" | AN03 pin |
| Writing "0100" | AN04 pin |
| Writing "0101" | AN05 pin |
| Writing "0110" | AN06 pin |
| Writing "0111" | AN07 pin |
| Writing "1000" | AN08 pin |
| Writing "1001" | AN09 pin |
| Writing "1010" | AN10 pin |
| Writing "1011" | AN11 pin |

\*: The number of analog input pins vary among products. For the number of analog input pins of a product, refer to its data sheet.

Notes:

- Do not write to ANS[3:0] any value other than those listed in the table above.
- When the ADMV bit is "1", do not modify these bits. Pins not used as analog input pins can be used as general-purpose I/O ports.

[bit3] ADI: Interrupt request flag bit

This bit detects the completion of A/D conversion.

When the A/D conversion function is used, the bit is set to "1" immediately after A/D conversion is complete.

Interrupt requests are output when this bit and the interrupt request enable bit (ADC2: ADIE) are both set to "1".

When "0" is written to this bit, it is cleared. Writing "1" to this bit does not change it or affect other bits.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit3               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that A/D conversion has not been completed. |
| <b>Reading "1"</b> | Indicates that A/D conversion has been completed.     |
| <b>Writing "0"</b> | Clears this bit.                                      |
| <b>Writing "1"</b> | Has no effect on operation.                           |

[bit2] ADMV: Conversion flag bit

This bit indicates that A/D conversion is in progress.

The bit is set to "1" during A/D conversion.

This bit is read-only. Writing a value to this bit has no effect on operation.

| bit2               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that A/D conversion is not executed. |
| <b>Reading "1"</b> | Indicates that A/D conversion is in progress.  |

[bit1] Reserved bit

Always set this bit to "0".

[bit0] AD: A/D conversion start bit

This bit starts the A/D conversion function with the software.

Writing "1" to the bit starts the A/D conversion function.

When the continuous start enable bit in the ADC2 register (ADC2:EXT) is "1", starting the A/D conversion with this bit is disabled.

With the EXT bit set to "0", when "1" is written to this bit while A/D conversion is in progress, A/D conversion restarts.

| bit0               | Details                             |
|--------------------|-------------------------------------|
| <b>Writing "0"</b> | Has no effect on operation.         |
| <b>Writing "1"</b> | Starts the A/D conversion function. |

Note: Writing "0" to this bit cannot stop the operation of the A/D conversion function. The read value of this bit is always "0".



## MB95850K/860K/870K Series

### 15.6.2 8/10-bit A/D Converter Control Register 2 (ADC2)

The 8/10-bit A/D converter control register 2 (ADC2) is used to control different functions of the 8/10-bit A/D converter, select the input clock, and enable and disable interrupts.

#### ■ Register Configuration

| bit           | 7   | 6    | 5    | 4    | 3    | 2   | 1      | 0      |
|---------------|-----|------|------|------|------|-----|--------|--------|
| Field         | AD8 | TIM1 | TIM0 | ADCK | ADIE | EXT | CKDIV1 | CKDIV0 |
| Attribute     | R/W | R/W  | R/W  | R/W  | R/W  | R/W | R/W    | R/W    |
| Initial value | 0   | 0    | 0    | 0    | 0    | 0   | 0      | 0      |

#### ■ Register Functions

##### [bit7] AD8: Precision select bit

This bit selects the resolution of A/D conversion.

Writing "0" to this bit selects 10-bit precision.

Writing "1" to this bit selects 8-bit precision. Reading the ADDL register can obtain 8-bit data.

| bit7               | Details          |
|--------------------|------------------|
| <b>Writing "0"</b> | 10-bit precision |
| <b>Writing "1"</b> | 8-bit precision  |

Note: The data bits to be used are different depending on the resolution selected. Modify this bit only when the A/D converter has stopped operating.

##### [bit6:5] TIM[1:0]: Sampling time select bits

These bits select the sampling time.

Modify the sampling time according to operating conditions (voltage and frequency).

The CKIN value is determined by the clock select bits (ADC2:CKDIV[1:0]).

| bit6:5              | Details   |
|---------------------|-----------|
| <b>Writing "00"</b> | CKIN × 4  |
| <b>Writing "01"</b> | CKIN × 7  |
| <b>Writing "10"</b> | CKIN × 10 |
| <b>Writing "11"</b> | CKIN × 16 |

Note: Modify these bits only when the A/D converter has stopped operating.

##### [bit4] ADCK: External start signal select bit

This bit selects the start signal for external start (ADC2:EXT = 1).

| bit4               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Selects not using any external start signal to start the A/D conversion function.                                  |
| <b>Writing "1"</b> | Selects the 8/16-bit composite timer ch. 0 output pin (TO00) as the pin used to start the A/D conversion function. |

[bit3] ADIE: Interrupt request enable bit

This bit enables or disables outputting the interrupt request to the interrupt controller.

When both this bit and the interrupt request flag bit (ADC1: ADI) have been set to "1", an interrupt request is output.

| bit3               | Details                                    |
|--------------------|--|
| <b>Writing "0"</b> | Disables outputting the interrupt request. |
| <b>Writing "1"</b> | Enables outputting the interrupt request.  |

[bit2] EXT: Continuous start enable bit

This bit selects whether to start the A/D conversion function with the software, or to continuously start the A/D conversion function whenever a rising edge of the input clock is detected.

| bit2               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Starts the A/D conversion function with the AD bit in the ADC1 register.   |
| <b>Writing "1"</b> | Continuously start the A/D conversion function according to the clock selected by the ADCK bit in the ADC2 register. |

[bit1:0] CKDIV[1:0]: Clock select bits

These bits select the clock (CKIN) to be used for A/D conversion. The input clock is generated by the prescaler. See "3.9 Operation of Prescaler" for details.

The sampling time varies according to the clock selected by these bits.

Modify these bits according to operating conditions (voltage and frequency).

| bit1:0              | Details<br>(MCLK: machine clock) |
|---------------------|----------------------------------|
| <b>Writing "00"</b> | 1 MCLK                           |
| <b>Writing "01"</b> | MCLK/2                           |
| <b>Writing "10"</b> | MCLK/4                           |
| <b>Writing "11"</b> | MCLK/8                           |

Note: Modify these bits only when the A/D converter has stopped operating.

## MB95850K/860K/870K Series

### 15.6.3 8/10-bit A/D Converter Data Register (Upper/Lower) (ADDH/ADDL)

The 8/10-bit A/D converter data register (upper/lower) (ADDH/ADDL) store the results of 10-bit A/D conversion during 10-bit A/D conversion.

The upper two bits of 10-bit data are stored in the ADDH register and the lower eight bits the ADDL register.

#### ■ Register Configuration

| ADDH          |   |   |   |   |   |   |           |
|---------------|---|---|---|---|---|---|-----------|
| bit           | 7 | 6 | 5 | 4 | 3 | 2 | 1 0       |
| Field         | — | — | — | — | — | — | SAR9 SAR8 |
| Attribute     | — | — | — | — | — | — | R R       |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0         |

| ADDL          |      |      |      |      |      |      |           |
|---------------|------|------|------|------|------|------|-----------|
| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1 0       |
| Field         | SAR7 | SAR6 | SAR5 | SAR4 | SAR3 | SAR2 | SAR1 SAR0 |
| Attribute     | R    | R    | R    | R    | R    | R    | R R       |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0         |

#### ■ Register Functions

The upper two bits of 10-bit A/D data correspond to bit1 and bit0 in the ADDH register and the lower eight bits bit7 to bit0 in the ADDL register.

If the AD8 bit in ADC2 register is set to "1", 8-bit precision is selected. Reading the ADDL register can obtain 8-bit data.

These two registers are read-only registers. Writing data to them has no effect on operation.

In A/D conversion in which 8-bit precision is selected, SAR8 and SAR9 in the ADDH register become "0".

#### ● A/D conversion function

When A/D conversion is started, the results of conversion are finalized and stored in the ADDH and ADDL registers after the conversion time according to the register settings elapses. After A/D conversion is completed and before the next A/D conversion is completed, read A/D data registers (conversion results), and clear the interrupt request flag bit (ADI) in the ADC1 register. During A/D conversion, the values of the ADDH and ADDL registers are results of the last A/D conversion.



- The conversion time may have an error of up to  $(1 \text{ CKIN} - 1 \text{ MCLK})$ , depending on the time at which A/D conversion starts.
- When setting the A/D converter in software, ensure that the settings satisfy the specifications of "sampling time" and "compare time" of the A/D converter mentioned in the device data sheet.



# **CHAPTER 16**

---

# **LOW-VOLTAGE DETECTION RESET CIRCUIT**

**This chapter describes the function and operation of the low-voltage detection reset circuit.**

- 16.1 Overview
- 16.2 Configuration
- 16.3 Pins
- 16.4 Operation
- 16.5 Registers

## **16.1 Overview**

---

**The low-voltage detection reset circuit monitors power supply voltage and generates a reset signal if the power supply voltage drops below the low-voltage detection voltage level.**

---

### **■ Low-voltage Detection Reset Circuit**

The low-voltage detection reset circuit monitors power supply voltage and generates a reset signal if the power supply voltage drops below the low-voltage detection voltage level.

The LVD reset voltage selection ID register (LVDR) selects the reset threshold voltage. The LVD reset circuit password register (LVDPW) sets a password to prevent the reset threshold voltage from being changed and the low-voltage detection reset circuit from being disabled accidentally.

At power-on, the lowest reset threshold voltage is selected in the LVDR register.

Upon power-on, the low-voltage detection reset circuit is automatically enabled, and can be disabled by the LVD reset circuit control register (LVDCC).

Refer to the device data sheet for details of the electrical characteristics.

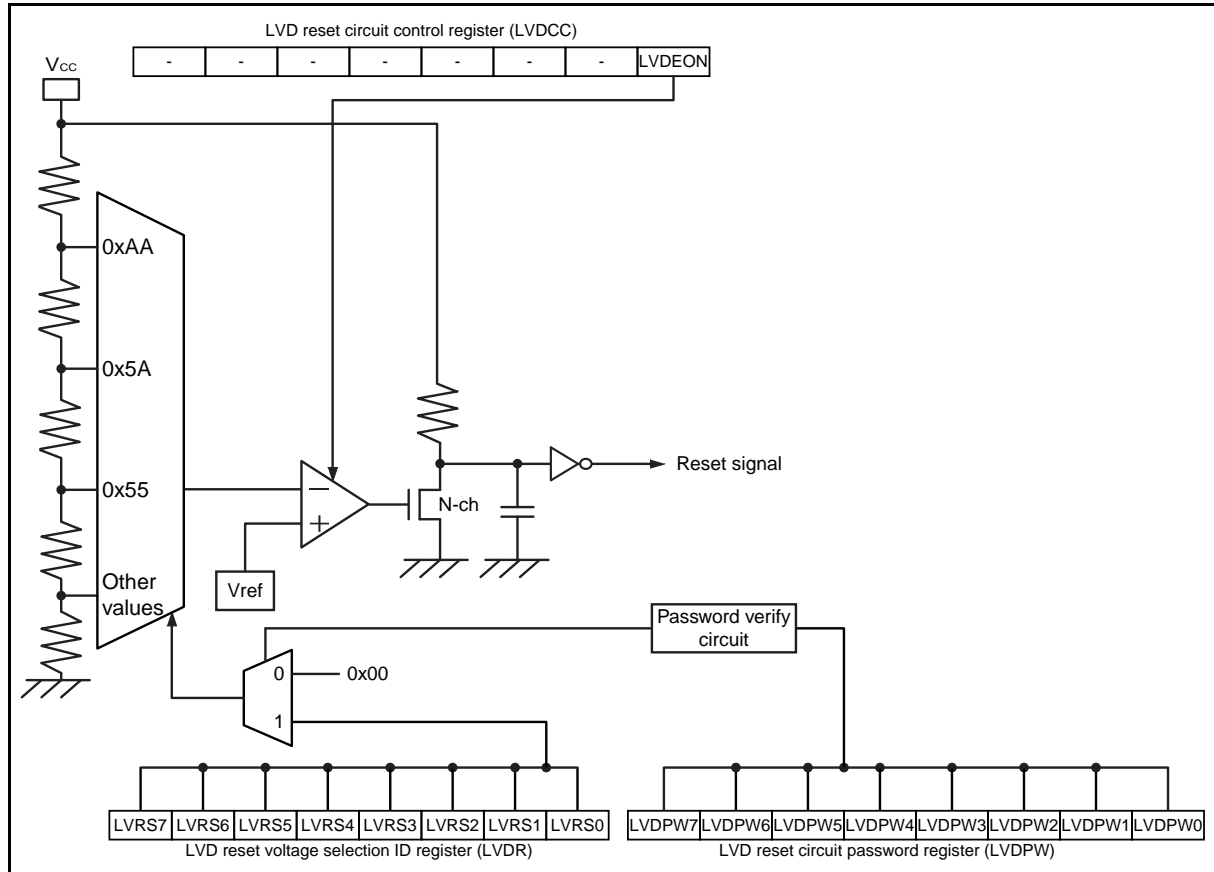


## 16.2 Configuration

Figure 16.2-1 is the block diagram of the low-voltage detection reset circuit.

### ■ Block Diagram of Low-voltage Detection Reset Circuit

Figure 16.2-1 Block Diagram of Low-voltage Detection Reset Circuit



## 16.3 Pins

---

**This section describes the pins of the low-voltage detection reset circuit.**

---

### ■ Pins of Low-voltage Detection Reset Circuit

- $V_{CC}$  pin

The low-voltage detection reset circuit monitors the voltage of this pin.

- $V_{SS}$  pin

This is the GND pin serving as the reference for voltage detection.

- $\overline{RST}$  pin

The low-voltage detection reset signal is output inside the microcontroller and to this pin.

## 16.4 Operation

The low-voltage detection reset circuit generates a reset signal if the power supply voltage falls below the detection voltage.

### ■ Reset Threshold Voltage

To change the reset threshold voltage secured with a password, input the correct password to the LVDPW register in advance.

Keep the value of the LVDPW register unchanged after selecting a reset threshold voltage, otherwise the reset threshold voltage is initialized.

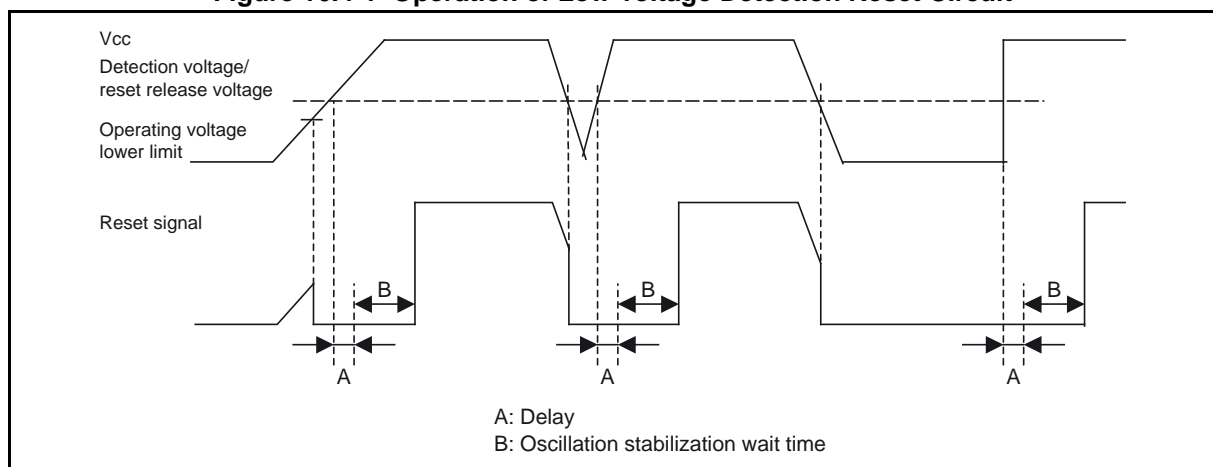
In the case of changing the reset threshold voltage in the LVDR register, the new threshold voltage does not start to take effect until the LVD reset threshold voltage transition stabilization time ( $t_{stb}$ ) elapses. For details of  $t_{stb}$ , refer to the device data sheet.

### ■ Operation of Low-voltage Detection Reset Circuit

The low-voltage detection reset circuit generates a reset signal if the power supply voltage falls below the low-voltage detection voltage. Afterward, if the low-voltage detection reset circuit detects the low-voltage detection reset release voltage, it outputs a reset signal lasting for the oscillation stabilization wait time and then releases the reset.

For details of the electrical characteristics, refer to the device data sheet.

Figure 16.4-1 Operation of Low-voltage Detection Reset Circuit



### ■ Operation in Standby Mode

The low-voltage detection reset circuit keeps operating even in standby mode (stop mode, sleep mode, subclock mode and watch mode).

## 16.5 Registers

---

**This section describes the registers of low-voltage detection reset circuit.**

---

**Table 16.5-1 List of Low-voltage Detection Reset Circuit Registers**

| Register abbreviation | Register name                           | Reference |
|-----------------------|---|-----------|
| LVDR                  | LVD reset voltage selection ID register | 16.5.1    |
| LVDPW                 | LVD reset circuit password register     | 16.5.2    |
| LVDC                  | LVD reset circuit control register      | 16.5.3    |

## 16.5.1 LVD Reset Voltage Selection ID Register (LVDR)

The LVD reset voltage selection ID register (LVDR) selects the reset threshold voltage.

### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | LVRS7 | LVRS6 | LVRS5 | LVRS4 | LVRS3 | LVRS2 | LVRS1 | LVRS0 |
| Attribute     | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

### ■ Register Functions

[bit7:0] LVRS[7:0]: LVD reset voltage select bits

These bits select the reset threshold voltage with an 8-bit code. They are cleared upon a power-on reset.

| bit7:0                               | Reset threshold voltage           |                                 |
|--------------------------------------|-----------------------------------|---------------------------------|
|                                      | Detection voltage (typical value) | Release voltage (typical value) |
| Writing "01010101"                   | 2.7 V                             | 2.8 V                           |
| Writing "01011010"                   | 3 V                               | 3.1 V                           |
| Writing "10101010"                   | 3.2 V                             | 3.3 V                           |
| Writing a value other than the above | 2.6 V                             | 2.7 V                           |

#### Note:

The reset of the low-voltage detection reset circuit has no effect on the reset threshold voltage settings as the reset cannot clear this register.

## 16.5.2 LVD Reset Circuit Password Register (LVDPW)

The LVD reset circuit password register (LVDPW) sets a password required to change the reset threshold voltage and disable the low-voltage detection reset circuit.

### ■ Register Configuration

| bit           | 7      | 6      | 5      | 4      | 3      | 2      | 1      | 0      |
|---------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Field         | LVDPW7 | LVDPW6 | LVDPW5 | LVDPW4 | LVDPW3 | LVDPW2 | LVDPW1 | LVPDW0 |
| Attribute     | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    |
| Initial value | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      |

### ■ Register Functions

[bit7:0] LVDPW[7:0]: LVD reset circuit password bits

These bits set a password required to change the reset threshold voltage and to disable the low-voltage detection reset circuit.

Only when "0b11001000" is written to the LVDPW[7:0] bits can the reset threshold voltage be changed and the low-voltage detection reset circuit be disabled.

| bit7:0                                       | Details  |
|--|--|
| <b>Writing "11001000"</b>                    | Enables changing the reset threshold voltage and disabling the low-voltage detection reset circuit.              |
| <b>Writing a value other than "11001000"</b> | Initializes the reset threshold voltage to the lowest value and enables the low-voltage detection reset circuit. |

#### Notes:

- Resetting the low-voltage detection reset circuit cannot initialize the LVDPW register.
- After changing the reset threshold voltage or the value of the LVDCC register, keep "0b11001000" as the value of the LVDPW register to prevent the reset threshold voltage and the LVDCC register from being initialized.

### 16.5.3 LVD Reset Circuit Control Register (LVDCC)

The LVD reset circuit control register (LVDCC) disables the low-voltage detection reset circuit.

#### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0      |
|---------------|---|---|---|---|---|---|---|--------|
| Field         | — | — | — | — | — | — | — | LVDEON |
| Attribute     | — | — | — | — | — | — | — | R/W    |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1      |

#### ■ Register Functions

##### [bit7:1] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit0] LVDEON: LVD reset circuit disable bits

This bit disables the low-voltage detection reset circuit. The low-voltage detection reset circuit is automatically enabled upon power-on.

The low-voltage detection reset circuit can be disabled only when "0b11001000" is written to the LVDPW register.

After "0" is written to the LVDEON bit, writing "1" to the bit has no effect on operation.

| bit0               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the low-voltage detection reset circuit. |
| <b>Writing "1"</b> | Has no effect on operation.                       |

#### Notes:

- Resetting the low-voltage detection reset circuit cannot initialize the LVDCC register.
- After disabled by writing "0" to the LVDCC register, the low-voltage detection reset circuit can be enabled again only when the LVDCC register is initialized by a power-on reset.





# **CHAPTER 17**

---

# **CLOCK SUPERVISOR COUNTER**

**This chapter describes the functions and operations of the clock supervisor counter.**

- 17.1 Overview
- 17.2 Configuration
- 17.3 Operations
- 17.4 Registers
- 17.5 Notes on Using Clock Supervisor Counter

## 17.1 Overview

---

**The clock supervisor counter checks the external clock frequency to detect the abnormal state of the external clock.**

---

### ■ Overview of Clock Supervisor Counter

The clock supervisor counter checks the external clock frequency to detect the abnormal state of the external clock.

The clock supervisor counter automatically counts up the counter based on the external clock input within the time-base timer interval time selected from eight options.

The main oscillation clock or the suboscillation clock can be selected as the count clock of this module.

---

#### Note:

The clock supervisor counter must operate in main CR clock mode with the hardware watchdog timer (running in standby mode).

Otherwise, it cannot detect the abnormal state of the external clock correctly and will hang up if the external clock stops.

See "CHAPTER 8 HARDWARE/SOFTWARE WATCHDOG TIMER" for the hardware watchdog timer (running in standby mode).

---

# MB95850K/860K/870K Series

## 17.2 Configuration

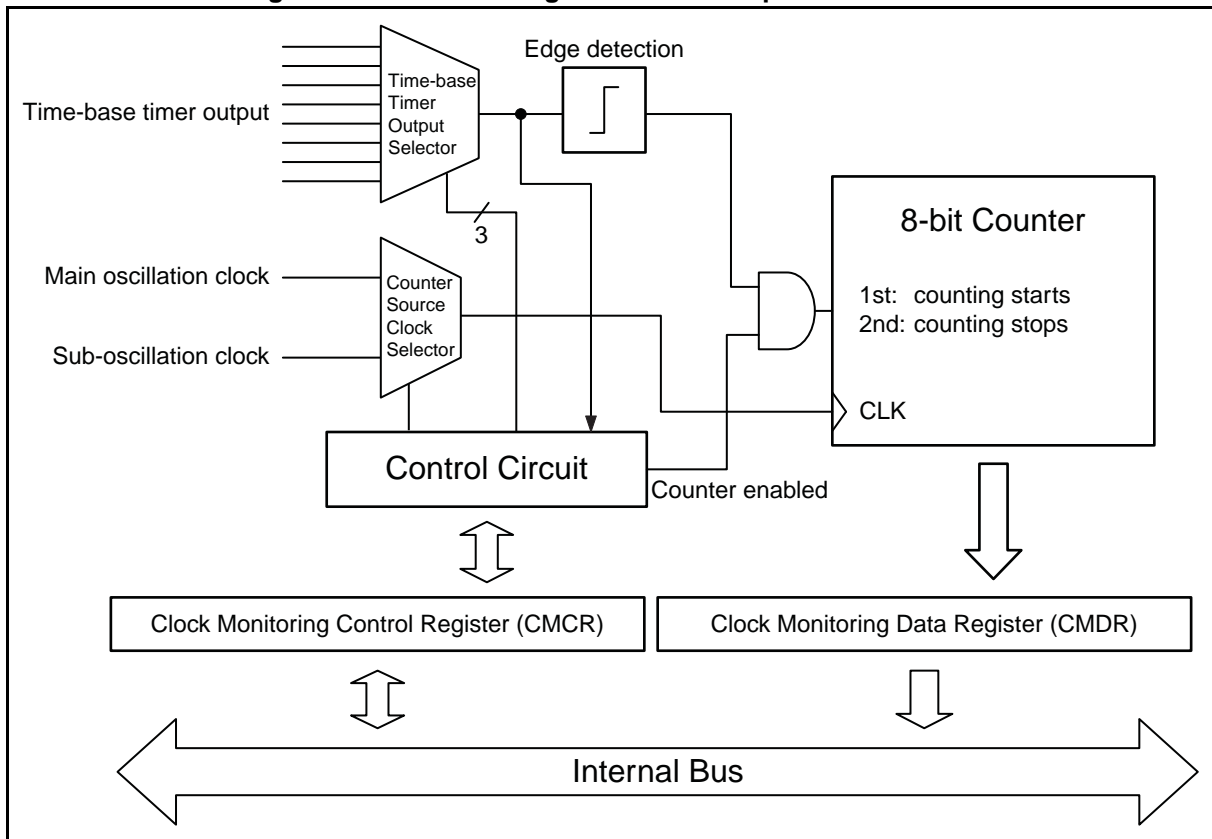
The clock supervisor counter consists of the following blocks:

- Control circuit
- Clock Monitoring Control Register (CMCR)
- Clock Monitoring Data Register (CMDR)
- Time-base timer output selector
- Counter source clock selector

### ■ Block Diagram of Clock Supervisor Counter

Figure 17.2-1 is the block diagram of the clock supervisor counter.

Figure 17.2-1 Block Diagram of Clock Supervisor Counter



- Control circuit

This block controls the start and stop of the counter, the counter clock source, and the counter enable period based on the settings of the clock monitoring control register (CMCR).

- Clock Monitoring Control Register (CMCR)

This register is used to select a counter source clock, select a counter enable period from eight different time-base timer intervals, start the counter and check whether the counter is operating or not.

- Clock Monitoring Data Register (CMDR)

This register block is used to read the counter value after the counter stops. The software determines whether the external clock frequency is correct or not according to the contents of this register.

- Time-base timer interval selector

This block is used to select the counter enable period from eight different time-base timer intervals.

- Counter source clock selector

This block is used to select the counter source clock from the main oscillation clock and the suboscillation clock.

# MB95850K/860K/870K Series

## 17.3 Operations

This section describes the operations of the clock supervisor counter.

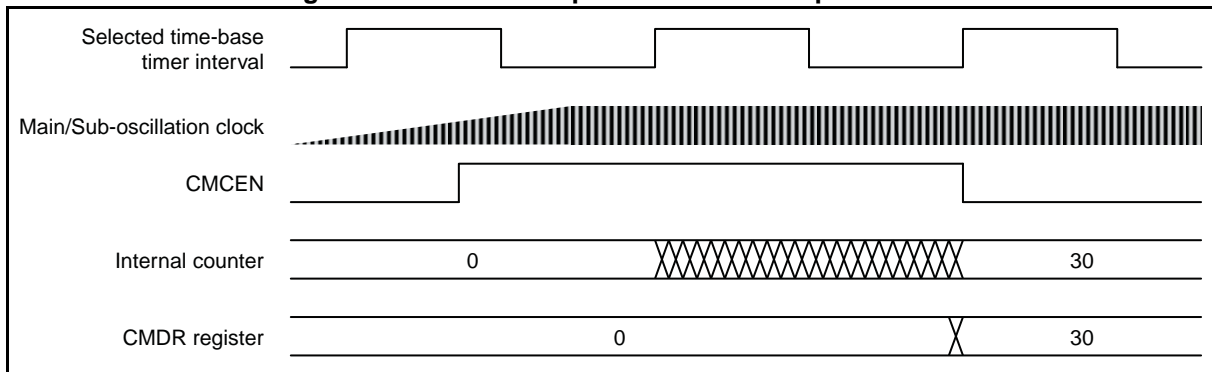
### ■ Clock Supervisor Counter

#### ● Clock Supervisor Counter Operation 1

The clock supervisor counter is first enabled by the software (CMCEN = 1), and then the clock supervisor counter operates with the time-base timer interval selected from eight options by the TBTSEL[2:0] bits. Between two rising edges of the time-base timer interval selected, the internal counter is clocked by the external clock.

The count clock of this module can be selected from the main oscillation clock and the suboscillation clock.

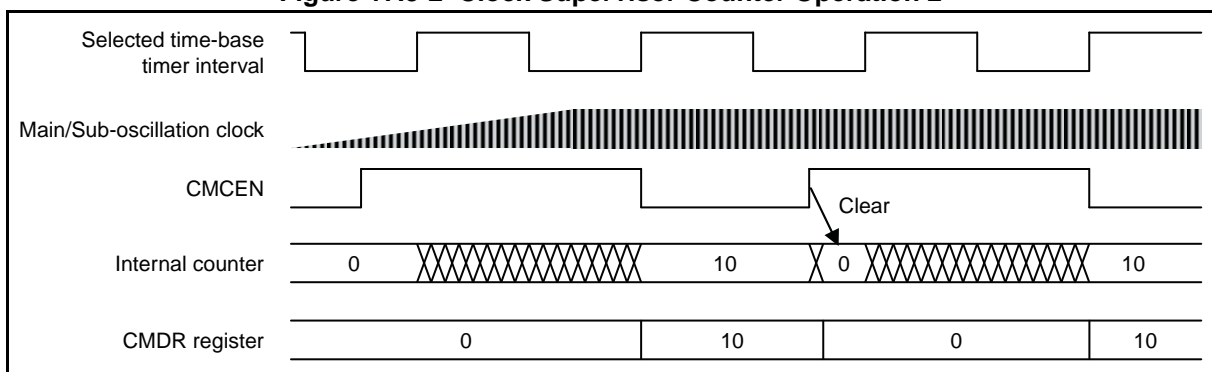
**Figure 17.3-1 Clock Supervisor Counter Operation 1**



#### ● Clock Supervisor Counter Operation 2

The CMDR register is cleared when the CMCEN bit changes from "0" to "1".

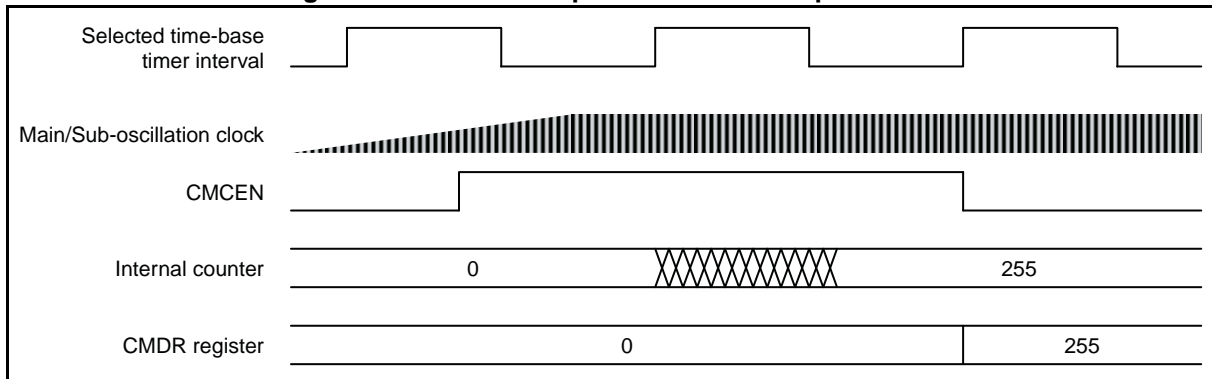
**Figure 17.3-2 Clock Supervisor Counter Operation 2**



● Clock Supervisor Counter Operation 3

The counter stops counting if it reaches "255". It cannot count further than "255".

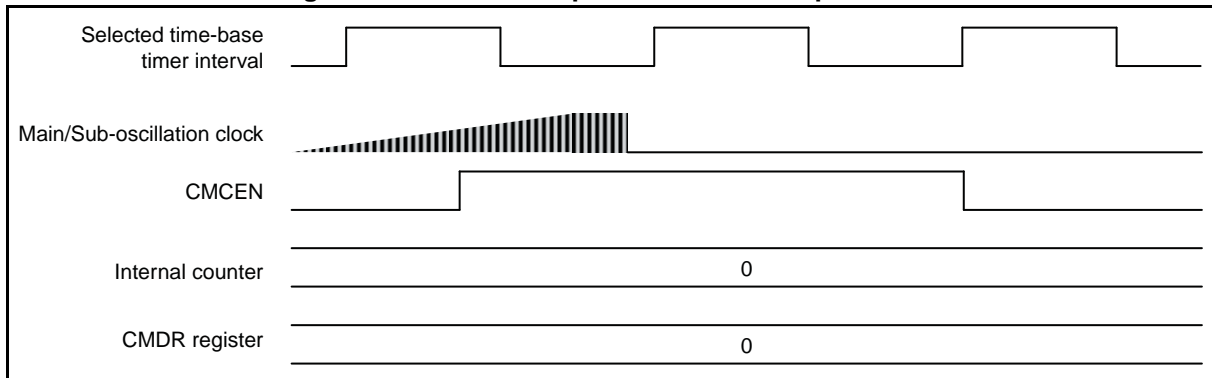
**Figure 17.3-3 Clock Supervisor Counter Operation 3**



● Clock Supervisor Counter Operation 4

If the external clock selected stops, the counter stops counting. According to this counter stop, the software identifies that the external clock selected is in the abnormal state.

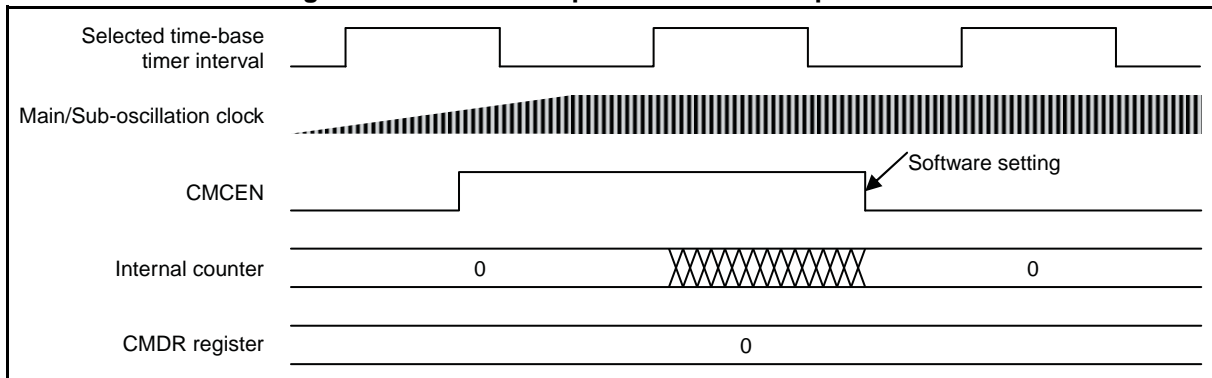
**Figure 17.3-4 Clock Supervisor Counter Operation 4**



● Clock Supervisor Counter Operation 5

The counter is cleared to "0" by the software if the CMCEN is set to "0" while the counter is operating.

**Figure 17.3-5 Clock Supervisor Counter Operation 5**



**MB95850K/860K/870K Series****■ Table of Time-base Timer Intervals & Clock Supervisor Counter Values**

Table 17.3-1 shows time-base timer intervals suitable for using different main CR clock frequencies to measure different external clocks.

**Table 17.3-1 Table of Counter Values in Relation to TBTSEL Settings**

| Main CR (FCRH) [MHz] | Main/Sub-crystal oscillation [MHz] | Main CR error | Measurement error | TBTSEL2 to TBTSEL0             |                                |                                |                                |                                   |                                   |                                   |                                   |
|----------------------|------------------------------------|---------------|-------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|
|                      |                                    |               |                   | "000"                          | "001"                          | "010"                          | "011"                          | "100"                             | "101"                             | "110"                             | "111"                             |
|                      |                                    |               |                   | $(2^3 \times 1 / \text{FCRH})$ | $(2^5 \times 1 / \text{FCRH})$ | $(2^7 \times 1 / \text{FCRH})$ | $(2^9 \times 1 / \text{FCRH})$ | $(2^{11} \times 1 / \text{FCRH})$ | $(2^{13} \times 1 / \text{FCRH})$ | $(2^{15} \times 1 / \text{FCRH})$ | $(2^{17} \times 1 / \text{FCRH})$ |
| 4                    | 0.03277                            | +2%           | -1                | 0                              | 0                              | 0                              | 1                              | 7                                 | 31                                | 130                               | 525                               |
|                      |                                    | -2%           | +1                | 1                              | 1                              | 1                              | 3                              | 9                                 | 35                                | 137                               | 548                               |
|                      | 0.5                                | +2%           | -1                | 0                              | 0                              | 6                              | 30                             | 124                               | 500                               | 2006                              | 8030                              |
|                      |                                    | -2%           | +1                | 1                              | 3                              | 9                              | 33                             | 131                               | 523                               | 2090                              | 8360                              |
|                      | 1                                  | +2%           | -1                | 0                              | 2                              | 14                             | 61                             | 249                               | 1002                              | 4014                              | 16061                             |
|                      |                                    | -2%           | +1                | 2                              | 5                              | 17                             | 66                             | 262                               | 1045                              | 4180                              | 16719                             |
|                      | 4                                  | +2%           | -1                | 2                              | 14                             | 61                             | 249                            | 1002                              | 4014                              | 16061                             | 64249                             |
|                      |                                    | -2%           | +1                | 5                              | 17                             | 66                             | 262                            | 1045                              | 4180                              | 16719                             | 66874                             |
|                      | 6                                  | +2%           | -1                | 4                              | 22                             | 93                             | 375                            | 1504                              | 6022                              | 24093                             | 96375                             |
|                      |                                    | -2%           | +1                | 7                              | 25                             | 98                             | 392                            | 1568                              | 6270                              | 25078                             | 100311                            |
|                      | 10                                 | +2%           | -1                | 8                              | 38                             | 155                            | 626                            | 2508                              | 10038                             | 40155                             | 160626                            |
|                      |                                    | -2%           | +1                | 11                             | 41                             | 164                            | 654                            | 2613                              | 10449                             | 41796                             | 167184                            |
|                      | 20                                 | +2%           | -1                | 18                             | 77                             | 312                            | 1253                           | 5018                              | 20077                             | 80312                             | 321253                            |
|                      |                                    | -2%           | +1                | 21                             | 82                             | 327                            | 1307                           | 5225                              | 20898                             | 83592                             | 334368                            |
|                      | 32.5                               | +2%           | -1                | 30                             | 126                            | 508                            | 2038                           | 8155                              | 32626                             | 130508                            | 522038                            |
|                      |                                    | -2%           | +1                | 34                             | 133                            | 531                            | 2123                           | 8490                              | 33960                             | 135837                            | 543347                            |

 : Recommended setting


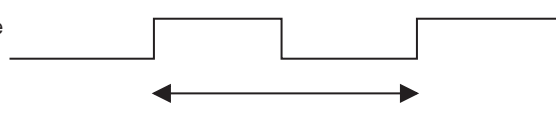
 : The counter value becomes "0" or "255".

Table 17.3-1 is calculated by the following equation:

$$\text{Counter value} = \frac{\left\{ \begin{array}{l} 2^2 \times 1/F_{\text{CRH}}(\text{TBTSEL}=000) \\ 2^2 \times 1/F_{\text{CRH}}(\text{TBTSEL}=001) \\ 2^2 \times 1/F_{\text{CRH}}(\text{TBTSEL}=010) \\ 2^2 \times 1/F_{\text{CRH}}(\text{TBTSEL}=011) \\ 2^3 \times 1/F_{\text{CRH}}(\text{TBTSEL}=100) \\ 2^3 \times 1/F_{\text{CRH}}(\text{TBTSEL}=101) \\ 2^3 \times 1/F_{\text{CRH}}(\text{TBTSEL}=110) \\ 2^3 \times 1/F_{\text{CRH}}(\text{TBTSEL}=111) \end{array} \right\} \times \text{Main/Sub-Oscillation Clock Frequency}}{2} \pm 1 \text{ (Measurement error)}$$

\*Omit the decimal places of the "counter value".



Within this period, the "counter value" in the above equation is counted by the main/sub oscillation clock.

If the time-base timer interrupt is used to make the clock supervisor counter wait for the oscillation stabilization time, please satisfy the following condition:

$$\text{Time-base Timer Interval} > \text{Main Oscillation / Suboscillation Stabilization Time} \times 1.05$$

e.g.  $F_{\text{CH}} = 4 \text{ MHz}$ ,  $F_{\text{CRH}} = 1 \text{ MHz}$ ,  $\text{MWT}[3:0] = 0b1111$  (in WATR register)

$$\text{Time-base Timer Interval} > \frac{(2^{14} - 2)}{4 \times 10^6} \times 1.05 \approx 4.3 \text{ ms}$$



$$\text{TBC}[3:0] = 0b0110 (2^{13} \times 1/F_{\text{CRH}})$$

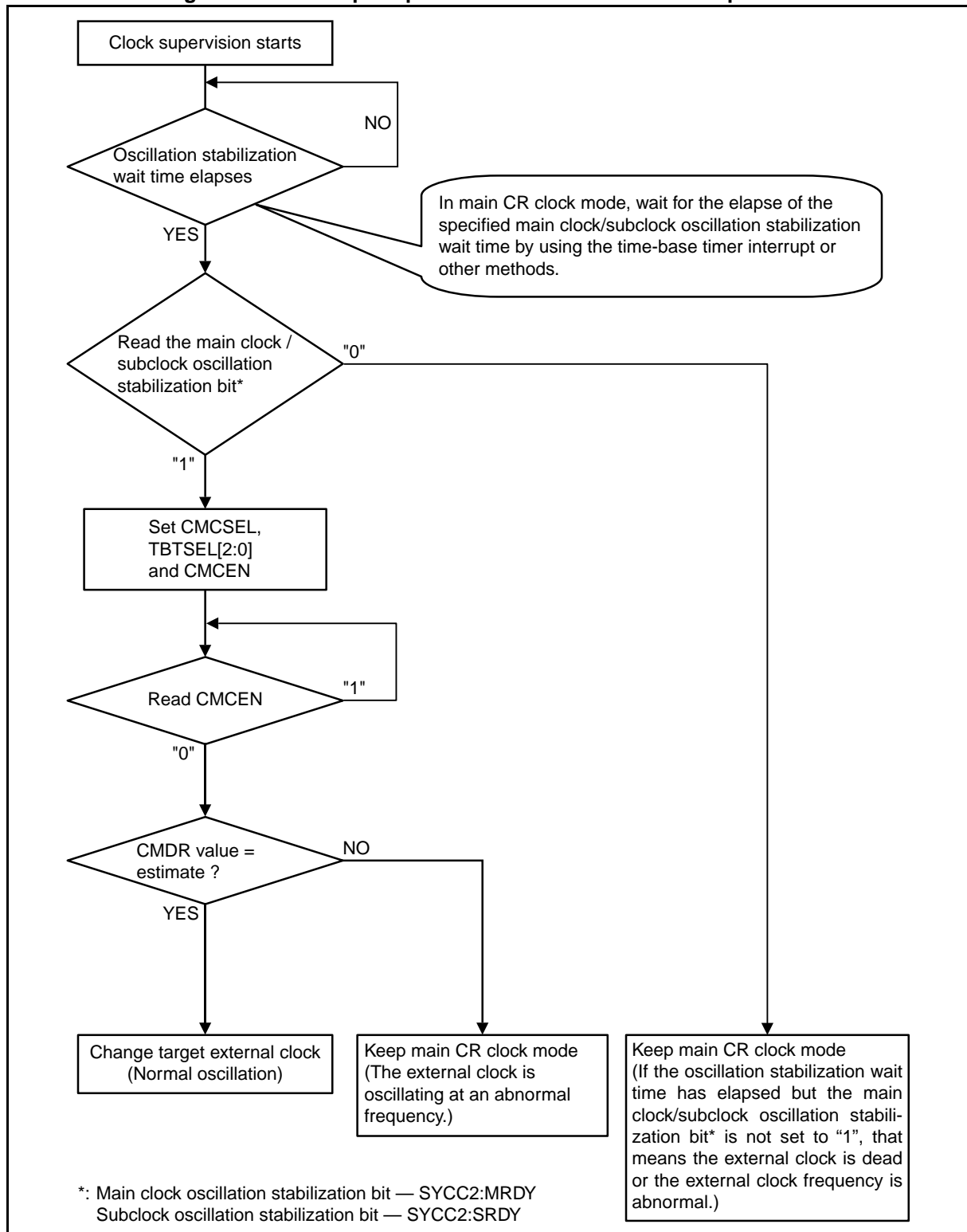
Notes:

- See "7.1 Overview" for time-base timer interval settings.
- See "3.3.3 Oscillation Stabilization Wait Time Setting Register (WATR)" for main/sub-oscillation stabilization time settings.



■ Sample Operation Flow Chart of Clock Supervisor

Figure 17.3-6 Sample Operation Flow Chart of Clock Supervisor



## 17.4 Registers

---

**This section describes the registers of the clock supervisor counter.**

---

**Table 17.4-1 List of Clock Supervisor Counter Registers**

| Register abbreviation | Register name                     | Reference |
|-----------------------|-----------------------------------|-----------|
| CMDR                  | Clock monitoring data register    | 17.4.1    |
| CMCR                  | Clock monitoring control register | 17.4.2    |

**MB95850K/860K/870K Series****17.4.1 Clock Monitoring Data Register (CMDR)**

The clock monitoring data register (CMDR) is used to read the count value after the clock supervisor counter stops. The software can check whether the external clock frequency is correct or not according to the content of this register.

**Register Configuration**

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | CMDR7 | CMDR6 | CMDR5 | CMDR4 | CMDR3 | CMDR2 | CMDR1 | CMDR0 |
| Attribute     | R     | R     | R     | R     | R     | R     | R     | R     |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

**Register Functions**

The clock monitoring data register (CMDR) is used to read the counter value after the clock supervisor counter stops.

- The counter value can be read from the clock monitoring data register (CMDR). The software can check whether the external clock frequency is correct or not according to the counter value read and the time-base timer interval selected.

[bit7:0] CMDR[7:0]: Clock monitoring data bits

These bits indicate the clock supervisor counter value after the counter stops.

These bits are cleared if one of the following events occurs:

- Reset
- The CMCEN bit in the CMCR register (CMCR:CMCEN) is modified from "0" to "1" by the software.
- The CMCEN bit is modified from "1" to "0" by the software while the counter is running.
- After the external clock stops, the falling edge of the selected time-base timer clock is detected twice. (See Figure 17.5-2.)

**Note:**

The value of this register is "0b00000000" as long as the counter is operating (CMCR:CMCEN = 1).

## 17.4.2 Clock Monitoring Control Register (CMCR)

The clock monitoring control register (CMCR) is used to select the counter source clock, select a time-base timer interval as the counter enable period, start the counter and check whether the counter is running or not.

### ■ Register Configuration

| bit           | 7 | 6 | 5        | 4      | 3       | 2       | 1       | 0     |
|---------------|---|---|----------|--------|---------|---------|---------|-------|
| Field         | — | — | Reserved | CMCSEL | TBTSEL2 | TBTSEL1 | TBTSEL0 | CMCEN |
| Attribute     | — | — | W        | R/W    | R/W     | R/W     | R/W     | R/W   |
| Initial value | 0 | 0 | 0        | 0      | 0       | 0       | 0       | 0     |

### ■ Register Functions

#### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit5] Reserved bit

Always set this bit to "0".

#### [bit4] CMCSEL: Counter clock select bit

This bit selects the counter source clock.

| bit4               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Selects the external main oscillation clock as the counter source clock. |
| <b>Writing "1"</b> | Selects the external suboscillation clock as the counter source clock.   |

#### [bit3:1] TBTSEL[2:0]: Time-base timer counter output select bits

These bits select the time-base timer interval.

The operation of the clock supervisor counter is enabled and disabled at specific times according to the time-base timer counter output selected by these bits.

The first rising edge of the interval selected enables the counter operation and the second rising edge of the same output disables the counter operation.

| bit3:1               | Details<br>( $F_{CRH}$ : main CR clock) |
|----------------------|---|
| <b>Writing "000"</b> | $2^3 \times 1/F_{CRH}$                  |
| <b>Writing "001"</b> | $2^5 \times 1/F_{CRH}$                  |
| <b>Writing "010"</b> | $2^7 \times 1/F_{CRH}$                  |
| <b>Writing "011"</b> | $2^9 \times 1/F_{CRH}$                  |
| <b>Writing "100"</b> | $2^{11} \times 1/F_{CRH}$               |
| <b>Writing "101"</b> | $2^{13} \times 1/F_{CRH}$               |
| <b>Writing "110"</b> | $2^{15} \times 1/F_{CRH}$               |
| <b>Writing "111"</b> | $2^{17} \times 1/F_{CRH}$               |

**MB95850K/860K/870K Series**[bit0] CMCEN: Counter enable bit

This bit enables or disables the clock supervisor counter.

Writing "0" to this bit stops the counter and clears the CMDR register to "0b00000000".

Writing "1" to this bit enables the counter. The counter starts counting when detecting the rising edge of the time-base timer interval. It stops counting when detecting the second rising edge of the same interval.

This bit is automatically set to "0" when the counter stops.

| bit0               | Details                         |
|--------------------|---------------------------------|
| <b>Writing "0"</b> | Disables the counter operation. |
| <b>Writing "1"</b> | Enables the counter operation.  |

## Notes:

- Do not modify the CMSEL bit when the CMCEN bit is "1".
- Do not modify the TBTSEL[2:0] bits when the CMCEN bit is "1".

## 17.5 Notes on Using Clock Supervisor Counter

This section provides notes on using the clock supervisor counter.

### ■ Notes on Using Clock Supervisor Counter

#### ● Restrictions

- The clock supervisor counter must operate in main CR clock mode with the hardware watchdog timer (running in standby mode). Otherwise, it cannot detect the abnormal state of the external clock correctly and will hang up if the external clock stops. See "CHAPTER 8 HARDWARE/SOFTWARE WATCHDOG TIMER" for the hardware watchdog timer (running in standby mode).
- Use main CR clock mode only. Do not use any other clock mode.
- If the time-base timer stops, the internal counter stops working. Do not clear the time-base timer while the clock supervisor counter is counting with the external clock.
- Select a time-base timer interval that is sufficiently long for the clock supervisor counter to operate. See Table 17.3-1 for time-base timer intervals.
- Read the CMDR register when CMCEN = 0. (The value of CMDR remains "0b00000000" while the clock supervisor counter is operating (CMCEN = 1).)
- When using the clock supervisor counter, ensure that the machine clock cycle is shorter than half the time-base timer interval selected. If the machine clock cycle is longer than half the time-base timer interval selected, CMCEN may remain "1" even after the clock supervisor counter stops.

Table 17.5-1 below shows the appropriate clock gear setting for each TBTSEL setting.

**Table 17.5-1 Appropriate Clock Gear Setting for Each TBTSEL Setting**

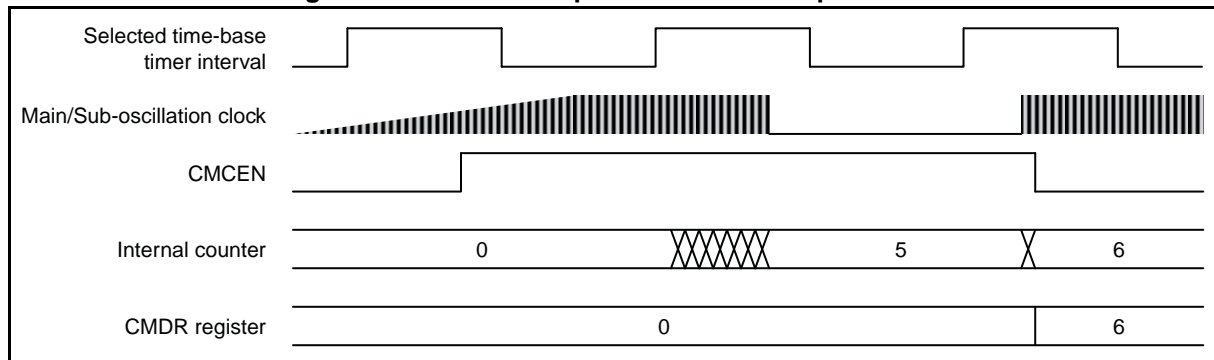
| DIV[1:0]<br>(clock gear setting) | TBTSEL[2:0]            |                        |   |
|----------------------------------|------------------------|------------------------|---|
|                                  | 000                    | 001                    | 010 to 111  |
|                                  | $2^3 \times 1/F_{CRH}$ | $2^5 \times 1/F_{CRH}$ | $2^7 \times 1/F_{CRH}$ to $2^{17} \times 1/F_{CRH}$ |
| 00 ( $1 \times 1/F_{CRH}$ )      | ○                      | ○                      | ○   |
| 01 ( $4 \times 1/F_{CRH}$ )      | x                      | ○                      | ○   |
| 10 ( $8 \times 1/F_{CRH}$ )      | x                      | ○                      | ○   |
| 11 ( $16 \times 1/F_{CRH}$ )     | x                      | x                      | ○   |

○ : Recommended

x : Prohibited

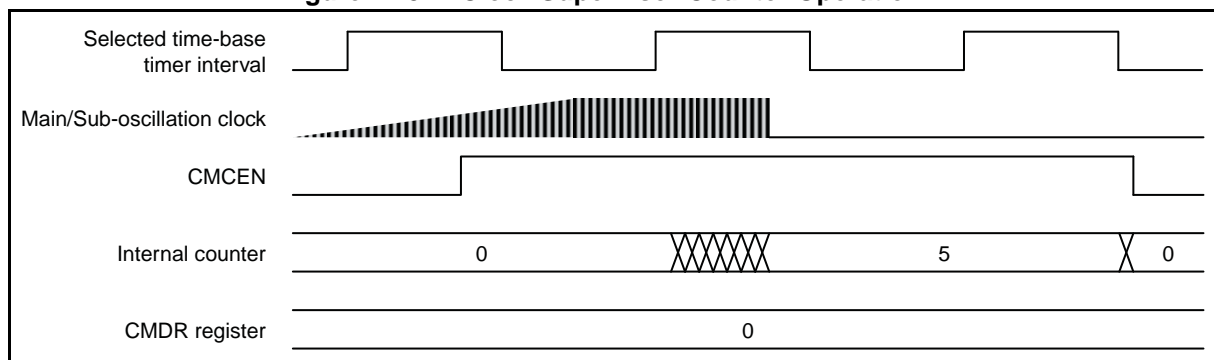
- If the external clock stops while the clock supervisor counter is operating, and it restarts after the second rising edge of the time-base timer interval selected, CMCEN is set to "0" after the external clock restarts.

**Figure 17.5-1 Clock Supervisor Counter Operation 1**



- With the clock supervisor counter running, if the external clock stops, CMCEN is set to "0" when a falling edge of the time-base timer interval selected is detected after the second rising edge of the same interval. The counter is cleared at the same falling edge.

**Figure 17.5-2 Clock Supervisor Counter Operation 2**







# **CHAPTER 18**

---

## **8/16-BIT PPG**

**This chapter describes the functions and operations of the 8/16-bit PPG.**

- 18.1 Overview
- 18.2 Configuration
- 18.3 Channel
- 18.4 Pins
- 18.5 Interrupt
- 18.6 Operations and Setting Procedure Example
- 18.7 Registers
- 18.8 Notes on Using 8/16-bit PPG

## 18.1 Overview

---

**The 8/16-bit PPG is an 8-bit reload timer module that uses pulse output control based on timer operation to perform PPG output. The 8/16-bit PPG also operates in cascade (8 bits + 8 bits) as 16-bit PPG.**

---

### ■ Overview of 8/16-bit PPG

The number of pins and that of channels of the 8/16-bit PPG vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

The 8/16-bit PPG functions are summarized as follows.

- 8-bit PPG output independent operation mode

In this mode, the unit can operate as two 8-bit PPG (PPG timer n0 and PPG timer n1).

- 8-bit prescaler + 8-bit PPG output operation mode

The rising and falling edge detection pulses from the PPG timer n1 output can be input to the downcounter of the PPG timer n0 to enable variable-cycle 8-bit PPG output.

- 16-bit PPG output operation mode

The unit can also operate in cascade (PPG timer n1 (upper 8 bits) + PPG timer n0 (lower 8 bits)) as 16-bit PPG output.

- PPG output operation

In this operation, a variable-cycle pulse waveform is output in any duty ratio.

The unit can also be used as a D/A converter in conjunction with an external circuit.

- Output inversion mode

This mode can invert the PPG output value.

# MB95850K/860K/870K Series

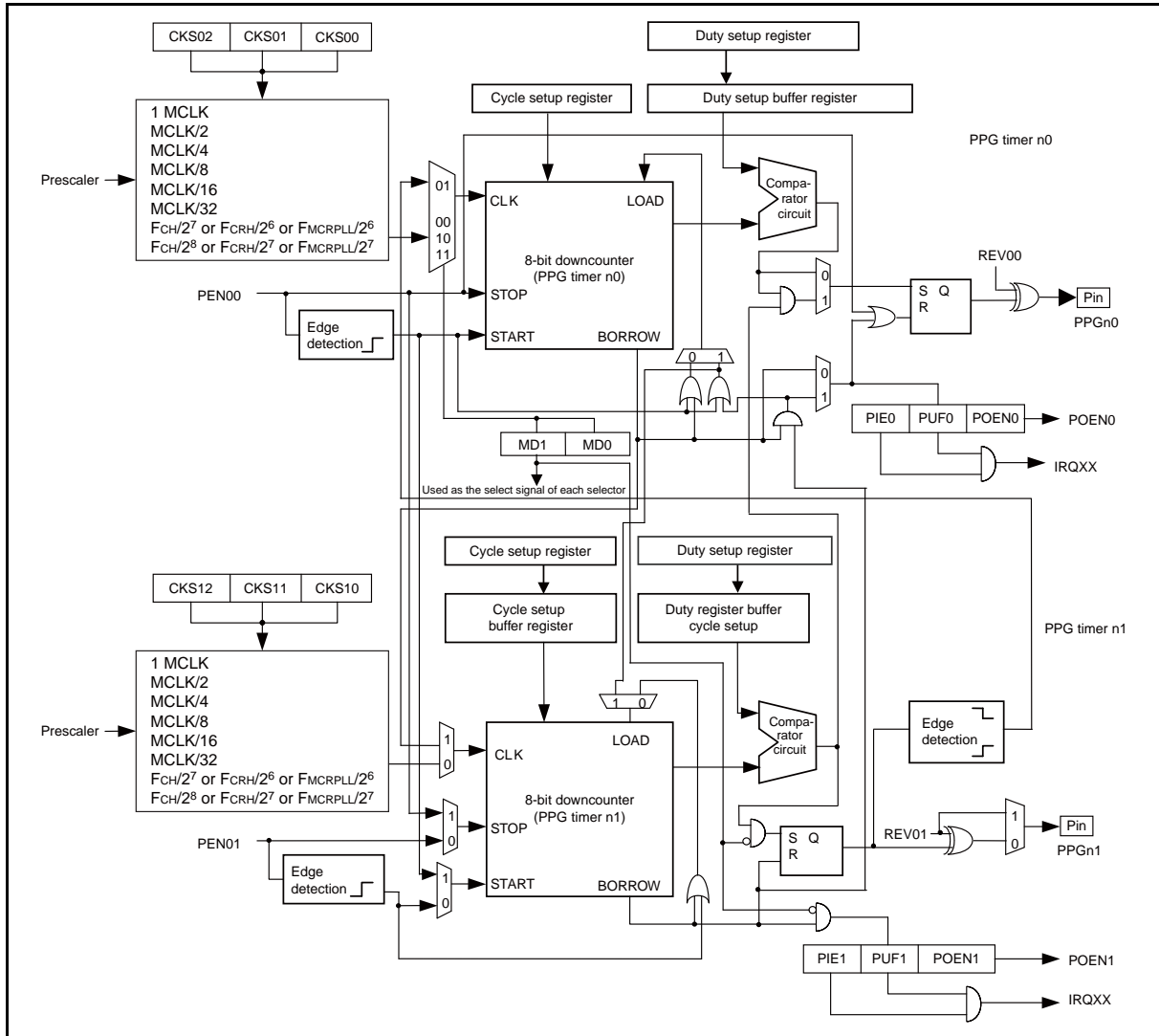
## 18.2 Configuration

This section shows the block diagram of the 8/16-bit PPG.

### ■ Block Diagram of 8/16-bit PPG

Figure 18.2-1 shows the block diagram of the 8/16-bit PPG.

Figure 18.2-1 Block Diagram of 8/16-bit PPG



- Counter clock selector

The clock for the countdown of 8-bit downcounter is selected from eight types of internal count clocks.

- 8-bit downcounter

It counts down with the count clock selected with the count clock selector.

- Comparator circuit

The output is kept "H" level until the value of 8-bit downcounter is corresponding to the value of 8/16-bit PPG duty setup buffer register from the value of 8/16-bit set buffer register of PPG cycle.

Afterwards, after keep "L" level the output until the counter value is corresponding to "1", it keeps counting 8-bit downcounter from the value of 8/16-bit PPG cycle setup buffer register.

- 8/16-bit PPG timer n1 control register (PCn1)

The operation condition on the PPG timer n1 side of 8/16-bit PPG timer is set.

- 8/16-bit PPG timer n0 control register (PCn0)

The operation mode of 8/16-bit PPG timer and the operation condition on the PPG timer n0 side are set.

- 8/16-bit PPG timer n1/n0 cycle setup buffer register (PPSn1/PPSn0)

The compare value for the cycle of 8/16-bit PPG timer is set.

- 8/16-bit PPG timer n1/n0 duty setup buffer register (PDSn1/PDSn0)

The compare value for "H" width of 8/16-bit PPG timer is set.

- 8/16-bit PPG start register

The start or the stop of 8/16-bit PPG timer is set.

- 8/16-bit PPG output inversion register

An initial level also includes the output of 8/16-bit PPG timer and it is reversed.

## ■ Input Clock

The 8/16-bit PPG uses the output clock from the prescaler as its input clock (count clock).

## MB95850K/860K/870K Series

### 18.3 Channel

---

This section describes the channel of the 8/16-bit PPG.

---

#### ■ Channel of 8/16-bit PPG

The 8/16-bit PPG consists of 8-bit PPG timer n0 and 8-bit PPG timer n1. They can be used respectively as two 8-bit PPGs or as a single 16-bit PPG.

Table 18.3-1 shows the pins and Table 18.3-2 the registers.

**Table 18.3-1 Pins of 8/16-bit PPG**

| Pin name | Pin function                                   |
|----------|--|
| PPGn0    | PPG timer n0 (8-bit PPG (n0), 16-bit PPG)      |
| PPGn1    | PPG timer n1 (8-bit PPG (n1), 8-bit prescaler) |

**Table 18.3-2 Registers of 8/16-bit PPG**

| Register abbreviation | Corresponding register (Name in this manual)      |
|-----------------------|---|
| PCn1                  | 8/16-bit PPG timer n1 control register            |
| PCn0                  | 8/16-bit PPG timer n0 control register            |
| PPSn1                 | 8/16-bit PPG timer n1 cycle setup buffer register |
| PPSn0                 | 8/16-bit PPG timer n0 cycle setup buffer register |
| PDSn1                 | 8/16-bit PPG timer n1 duty setup buffer register  |
| PDSn0                 | 8/16-bit PPG timer n0 duty setup buffer register  |
| PPGS                  | 8/16-bit PPG start register                       |
| REVC                  | 8/16-bit PPG output inversion register            |

## **18.4 Pins**

---

**This section describes the pins of the 8/16-bit PPG.**

---

### **■ Pins of 8/16-bit PPG**

- PPGn0 pin and PPGn1 pin

These pins function both as general-purpose I/O ports and 8/16-bit PPG outputs.

PPGn0, PPGn1: A PPG waveform is output to these pins. The PPG waveform can be output by enabling the output by the 8/16-bit PPG timer n1/n0 control registers (PCn0: POEN0 = 1, PCn1: POEN1 = 1).

## MB95850K/860K/870K Series

### 18.5 Interrupt

---

The 8/16-bit PPG outputs an interrupt request when a counter borrow is detected.

---

#### ■ Interrupt of 8/16-bit PPG

Table 18.5-1 shows the interrupt control bits and interrupt sources of the 8/16-bit PPG.

**Table 18.5-1 Interrupt Control Bits and Interrupt Sources of 8/16-bit PPG**

| Item                         | Description                                  |   |
|------------------------------|--|---|
|                              | PPG timer n1<br>(8-bit PPG, 8-bit prescaler) | PPG timer n0<br>(8-bit PPG, 16-bit PPG) |
| Interrupt request flag bit   | PUF1 bit in PCn1                             | PUF0 bit in PCn0                        |
| Interrupt request enable bit | PIE1 bit in PCn1                             | PIE0 bit in PCn0                        |
| Interrupt source             | Counter borrow of PPG cycle downcounter      |   |

When a counter borrow occurs on the downcounter, the 8/16-bit PPG sets the counter borrow detection flag bit (PUF) in the 8/16-bit PPG timer n0/n1 control register (PC) to "1". When the interrupt request enable bit is enabled (PIE = 1), an interrupt request is output to the interrupt controller.

In 16-bit PPG mode, the 8/16-bit PPG timer n0 control register (PCn0) is available.

## **18.6 Operations and Setting Procedure Example**

---

**This section describes the operations of the 8/16-bit PPG.**

---

The 8/16-bit PPG has the following three operating modes:

- 8-bit PPG independent mode
- 8-bit prescaler + 8-bit PPG mode
- 16-bit PPG mode



## 18.6.1 8-bit PPG Independent Mode

In this mode, the 8/16-bit PPG operates as two channels (PPG timer n0 and PPG timer n1) of the 8-bit PPG.

### ■ Setting 8-bit PPG Independent Mode

The 8/16-bit PPG requires the register settings shown in Figure 18.6-1 to operate in 8-bit PPG independent mode.

Figure 18.6-1 8-bit PPG Independent Mode

|       | bit7                                  | bit6 | bit5  | bit4  | bit3  | bit2  | bit1  | bit0  |
|-------|---------------------------------------|------|-------|-------|-------|-------|-------|-------|
| PCn1  | -                                     | -    | PIE1  | PUF1  | POEN1 | CKS12 | CKS11 | CKS10 |
|       |                                       |      | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     |
| PCn0  | MD1                                   | MD0  | PIE0  | PUF0  | POEN0 | CKS02 | CKS01 | CKS00 |
|       | 0                                     | 0    | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     |
| PPSn1 | PH7                                   | PH6  | PH5   | PH4   | PH3   | PH2   | PH1   | PH0   |
|       | Set PPG output cycle for PPG timer n1 |      |       |       |       |       |       |       |
| PPSn0 | PL7                                   | PL6  | PL5   | PL4   | PL3   | PL2   | PL1   | PL0   |
|       | Set PPG output cycle for PPG timer n0 |      |       |       |       |       |       |       |
| PDSn1 | DH7                                   | DH6  | DH5   | DH4   | DH3   | DH2   | DH1   | DH0   |
|       | Set PPG output duty for PPG timer n1  |      |       |       |       |       |       |       |
| PDSn0 | DL7                                   | DL6  | DL5   | DL4   | DL3   | DL2   | DL1   | DL0   |
|       | Set PPG output duty for PPG timer n0  |      |       |       |       |       |       |       |
| PPGS  | -                                     | -    | PEN21 | PEN20 | PEN11 | PEN10 | PEN01 | PEN00 |
|       | *                                     | *    | *     | *     | *     | *     | ⊙     | ⊙     |
| REVC  | -                                     | -    | REV21 | REV20 | REV11 | REV10 | REV01 | REV00 |
|       | *                                     | *    | *     | *     | *     | *     | ⊙     | ⊙     |

⊙ : Used bit  
 0 : Set to "0"  
 \* : The bit status depends on the number of channels provided.

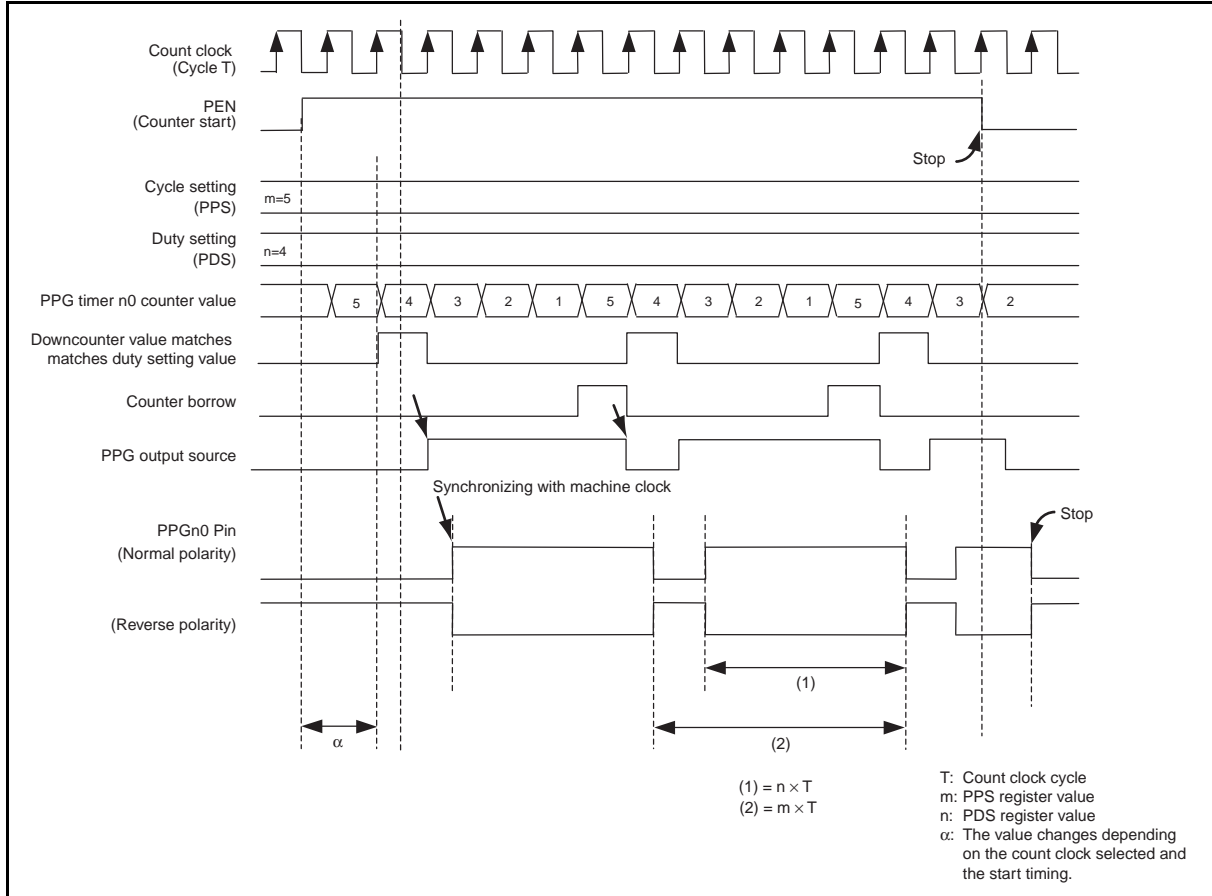
### ■ Operation of 8-bit PPG Independent Mode

- This mode is selected when the operation mode select bits (MD[1:0]) in the 8/16-bit PPG timer n0 control register (PCn0) are set to "0b00".
- When the corresponding bit (PEN) in the 8/16-bit PPG start register (PPGS) is set to "1", the value in the 8/16-bit PPG cycle setup buffer register (PPS) is loaded to start down-count operation. When the count value reaches "1", the value in the cycle setup register is reloaded to repeat the counting.
- "H" is output to the PPG output synchronizing with the count clock. When the downcounter value matches the value in the 8/16-bit PPG timer n1/n0 duty setup buffer register (PDS). After "H" which is the value of duty setting is output, "L" is output to the PPG output.

If, however, the PPG output level reverse bit is set to "1", the PPG output is set and reset inversely from the above process.

Figure 18.6-2 shows the operation of the 8-bit PPG independent mode.

**Figure 18.6-2 Operation of 8-bit PPG Independent Mode**



Example for setting the duty to 50%

When the PDS register is set to "0x02" with the PPS register set to "0x04", the PPG output is set at a duty ratio of 50% (half the value of the PPS register is set to the PDS register).

## 18.6.2 8-bit Prescaler + 8-bit PPG Mode

In this mode, the rising and falling edge detection pulses from the PPG timer n1 output can be used as the count clock of the PPG timer n0 downcounter to allow variable-cycle 8-bit PPG output from PPG timer n0.

### ■ Setting 8-bit Prescaler + 8-bit PPG Mode

The 8/16-bit PPG requires the register settings shown in Figure 18.6-3 to operate in 8-bit prescaler + 8-bit PPG mode.

Figure 18.6-3 Setting 8-bit Prescaler + 8-bit PPG Mode

|       | bit7                                  | bit6 | bit5  | bit4  | bit3  | bit2  | bit1  | bit0  |
|-------|---------------------------------------|------|-------|-------|-------|-------|-------|-------|
| PCn1  | -                                     | -    | PIE1  | PUF1  | POEN1 | CKS12 | CKS11 | CKS10 |
|       |                                       |      | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     |
| PCn0  | MD1                                   | MD0  | PIE0  | PUF0  | POEN0 | CKS02 | CKS01 | CKS00 |
|       | 0                                     | 1    | ⊙     | ⊙     | ⊙     | x     | x     | x     |
| PPSn1 | PH7                                   | PH6  | PH5   | PH4   | PH3   | PH2   | PH1   | PH0   |
|       | Set PPG output cycle for PPG timer n1 |      |       |       |       |       |       |       |
| PPSn0 | PL7                                   | PL6  | PL5   | PL4   | PL3   | PL2   | PL1   | PL0   |
|       | Set PPG output cycle for PPG timer n0 |      |       |       |       |       |       |       |
| PDSn1 | DH7                                   | DH6  | DH5   | DH4   | DH3   | DH2   | DH1   | DH0   |
|       | Set PPG output duty for PPG timer n1  |      |       |       |       |       |       |       |
| PDSn0 | DL7                                   | DL6  | DL5   | DL4   | DL3   | DL2   | DL1   | DL0   |
|       | Set PPG output duty for PPG timer n0  |      |       |       |       |       |       |       |
| PPGS  | -                                     | -    | PEN21 | PEN20 | PEN11 | PEN10 | PEN01 | PEN00 |
|       | *                                     | *    | *     | *     | *     | *     | ⊙     | ⊙     |
| REVC  | -                                     | -    | REV21 | REV20 | REV11 | REV10 | REV01 | REV00 |
|       | *                                     | *    | *     | *     | *     | *     | ⊙     | ⊙     |

⊙ : Used bit  
 0 : Set to "0"  
 1 : Set to "1"  
 x : Setting nullified  
 \* : The bit status varies depending of the number of channels implemented

### ■ Operation of 8-bit Prescaler + 8-bit PPG Mode

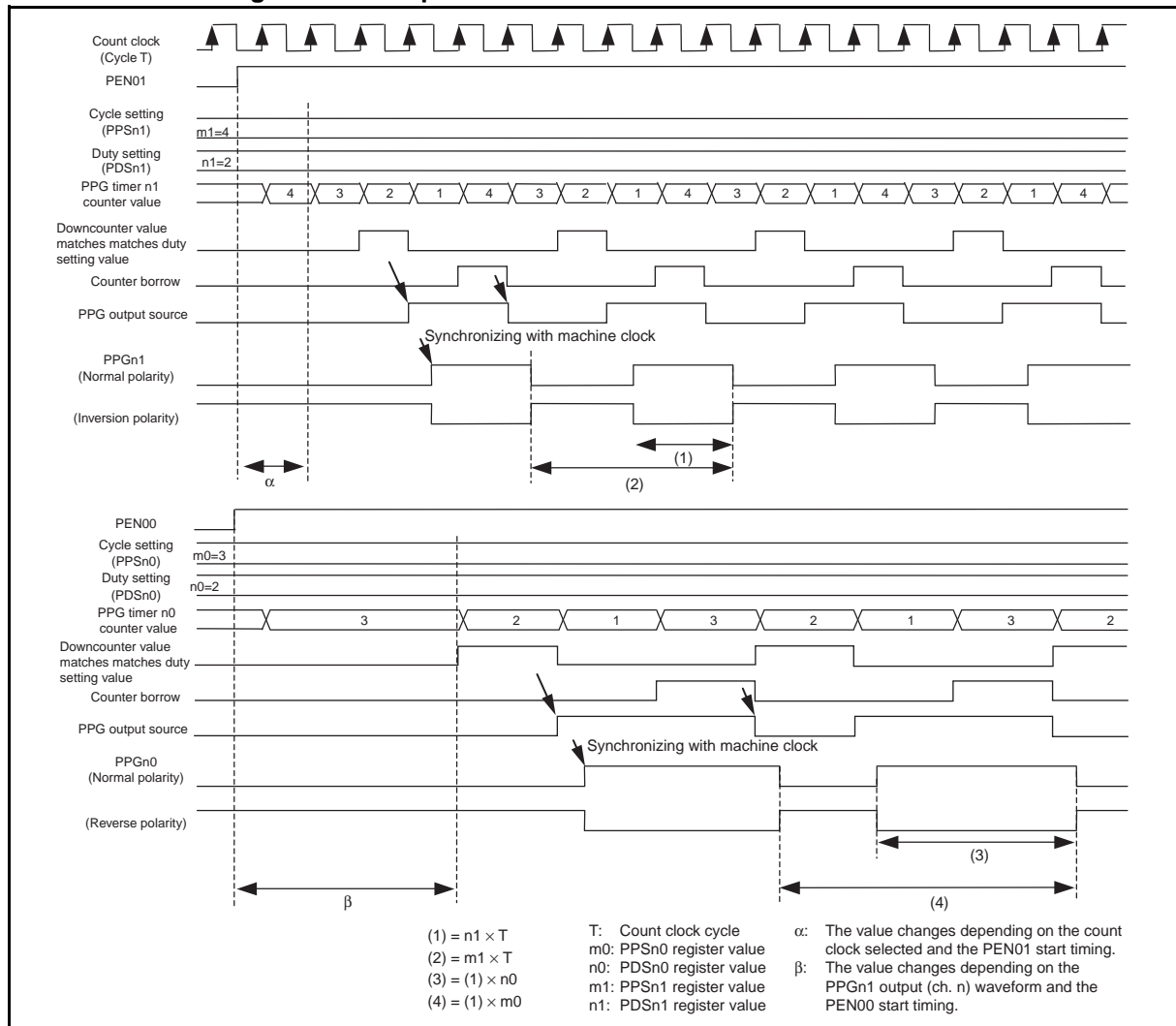
- This mode is selected by setting the operation mode select bits (MD[1:0]) in the 8/16-bit PPG timer n0 control register (PCn0) to "0b01". This allows PPG timer n1 to be used as an 8-bit prescaler and PPG timer n0 to be used as an 8-bit PPG.
- When the PPG timer n1 (ch. n) downcounter operation enable bit (PEN01) is set to "1", the 8-bit prescaler (PPG timer n1) loads the value in the 8/16-bit PPG timer n1 cycle setup buffer register (PPSn1) and starts down-count operation. When the value of the downcounter matches the value in the 8/16-bit PPG timer n1 duty setup buffer register (PDSn1), the PPGn1 output is set to "H" synchronizing with the count clock. After "H" which is the value of duty setting is output, the PPGn1 output is set to "L". If the output level reverse signal (REV01) is "0", the polarity remains the same. If it is "1", the polarity is

reversed and the signal is output to the PPGn1 pin.

- When the PPG timer n0 (ch. n) downcounter operation enable bit (PEN00) is set to "1", the 8-bit PPG (PPG timer n0) loads the value in the 8/16-bit PPG timer n0 cycle setup buffer register (PPSn0) and starts down-count operation (count clock = rising and falling edge detection pulses of PPGn1 output after PPG timer n1 operation is enabled). When the count value reaches "1", the value in the PPSn0 register is reloaded to repeat the counting. When the value of the downcounter matches the value in the 8/16-bit PPG timer n0 duty setup buffer register (PDSn0), the PPGn0 output is set to "H" synchronizing with the count clock. After "H" which is the value of duty setting is output, the PPGn0 output is reset to "L". If the output level reverse bit (REV00) is "0", the polarity remains the same. If it is "1", the polarity is reversed and the signal is output to the PPGn0 pin.
- Set that the duty of the 8-bit prescaler (PPG timer n1) output to 50%.
- When PPG timer n0 is started with the 8-bit prescaler (PPG timer n1) being stopped, PPG timer n0 does not count.
- When the duty of the 8-bit prescaler (PPG timer n1) is set to 0% or 100%, PPG timer n0 does not perform counting as the 8-bit prescaler (PPG timer n1) output does not toggle.

Figure 18.6-4 shows the operation of 8-bit prescaler + 8-bit PPG mode.

**Figure 18.6-4 Operation of 8-bit Prescaler + 8-bit PPG Mode**



### 18.6.3 16-bit PPG Mode

In this mode, the 8/16-bit PPG can operate as a 16-bit PPG when PPG timer n1 and PPG timer n0 are assigned to the upper and lower bits respectively.

#### ■ Setting 16-bit PPG Mode

The 8/16-bit PPG requires the register settings shown in Figure 18.6-5 to operate in 16-bit PPG mode.

Figure 18.6-5 Setting 16-bit PPG Mode

|       | bit7   | bit6 | bit5  | bit4  | bit3  | bit2  | bit1  | bit0  |
|-------|--|------|-------|-------|-------|-------|-------|-------|
| PCn1  | -  | -    | PIE1  | PUF1  | POEN1 | CKS12 | CKS11 | CKS10 |
|       |  |      | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     |
| PCn0  | MD1  | MD0  | PIE0  | PUF0  | POEN0 | CKS02 | CKS01 | CKS00 |
|       | 0  | 0/1  | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     | ⊙     |
| PPSn1 | PH7  | PH6  | PH5   | PH4   | PH3   | PH2   | PH1   | PH0   |
|       | Set PPG output cycle (Upper 8 bits) for PPG timer n1 |      |       |       |       |       |       |       |
| PPSn0 | PL7  | PL6  | PL5   | PL4   | PL3   | PL2   | PL1   | PL0   |
|       | Set PPG output cycle (Lower 8 bits) for PPG timer n0 |      |       |       |       |       |       |       |
| PDSn1 | DH7  | DH6  | DH5   | DH4   | DH3   | DH2   | DH1   | DH0   |
|       | Set PPG output duty (Upper 8 bits) for PPG timer n1  |      |       |       |       |       |       |       |
| PDSn0 | DL7  | DL6  | DL5   | DL4   | DL3   | DL2   | DL1   | DL0   |
|       | Set PPG output duty (Lower 8 bits) for PPG timer n0  |      |       |       |       |       |       |       |
| PPGS  | -  | -    | PEN21 | PEN20 | PEN11 | PEN10 | PEN01 | PEN00 |
|       | *  | *    | *     | *     | *     | *     | x     | ⊙     |
| REVC  | -  | -    | REV21 | REV20 | REV11 | REV10 | REV01 | REV00 |
|       | *  | *    | *     | *     | *     | *     | x     | ⊙     |

⊙ : Used bit  
 0 : Set to "0"  
 1 : Set to "1"  
 x : Setting nullified  
 \* : The bit status changes depending on the number of channels implemented.

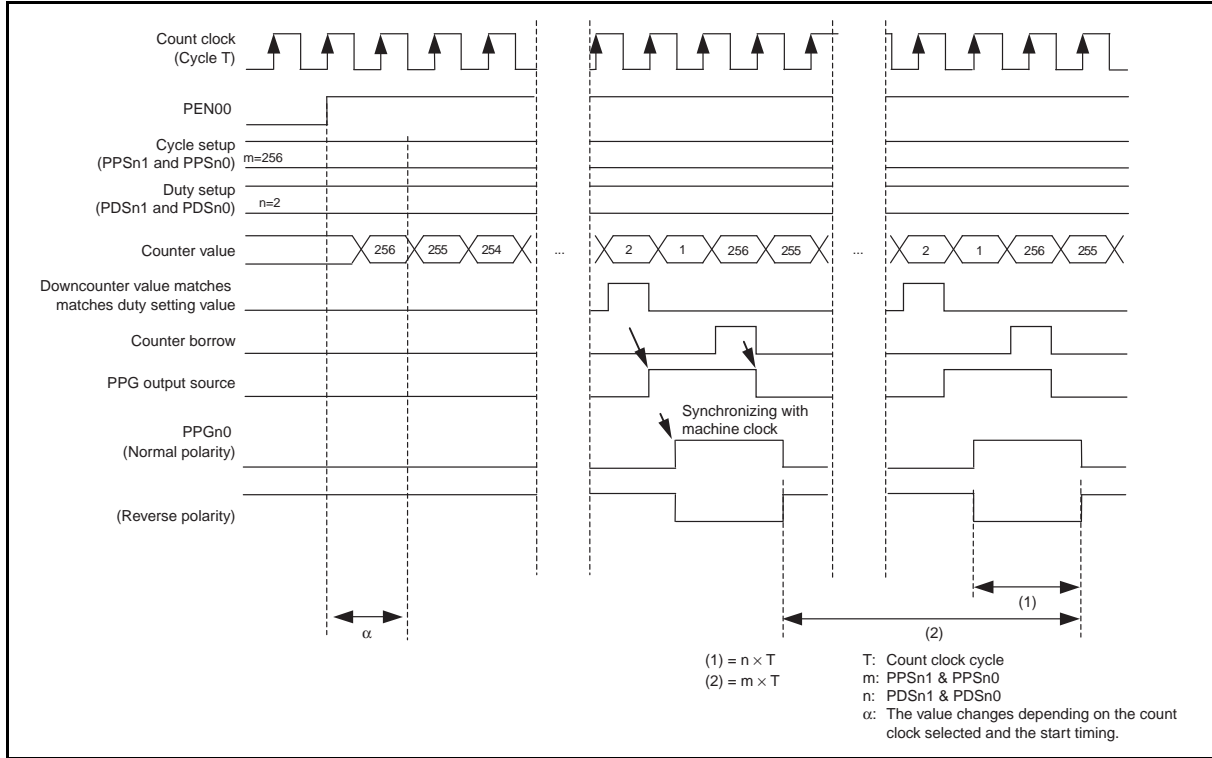
#### ■ Operation of 16-bit PPG Mode

- This mode is selected by setting the operation mode select bits (MD[1:0]) in the PPG timer n0 control register (PCn0) to "0b10" or "0b11".
- When the PPG timer n0 (ch. n) downcounter operation enable bit (PEN00) is set to "1" in 16-bit PPG mode, the 8-bit downcounters (PPG timer n0) and 8-bit downcounter (PPG timer n1) load the values in the 8/16-bit PPG timer n1/n0 cycle setup buffer registers (PPSn1 for PPG timer n1 and PPSn0 for PPG timer n0) and start down-count operation. When the count value reaches "1", the values in the cycle setup register are reloaded and the counters repeat the counting.
- When the values of the downcounters match the values in the 8/16-bit PPG timer duty setup buffer registers (both the value in PDSn1 for PPG timer n1 and the value in PDSn0 for PPG timer n0), the PPGn0 pin is set to "H" synchronizing with the count clock. After "H" which

is the value of duty setting is output, the PPGn0 pin is set to "L". If the output level reverse bit (REV00) is "0", the signal is output to the PPGn0 pin with the polarity unchanged. If it is set to "1", the polarity is reversed and the signal is output to the PPGn0 pin.

Figure 18.6-6 shows the operation of 16-bit PPG mode.

**Figure 18.6-6 Operation of 16-bit PPG Mode**



**■ Setting Procedure Example**

Below is an example of procedure for setting the 8/16-bit PPG ch. n.

**● Initial setup**

1. Set the port output. (DDR)
2. Set the interrupt level. (ILR\*)
3. Select the operating clock, enable the output and interrupt. (PCn1)
4. Select the operating clock, enable the output and interrupt, select the operation mode. (PCn0)
5. Set the cycle. (PPS)
6. Set the duty. (PDS)
7. Set the 8/16-bit PPG output reverse register. (REVC)
8. Start the 8/16-bit PPG. (PPGS)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

**● Interrupt processing**

1. Process any interrupt.
2. Clear the interrupt request flag. (PCn1:PUF1, PCn0:PUF0)
3. Start the 8/16-bit PPG. (PPGS)

## 18.7 Registers

---

**This section describes the registers of the 8/16-bit PPG.**

---

**Table 18.7-1 List of 8/16-bit PPG Registers**

| Register abbreviation | Register name                                       | Reference |
|-----------------------|---|-----------|
| PCn1                  | 8/16-bit PPG timer n1 control register              | 18.7.1    |
| PCn0                  | 8/16-bit PPG timer n0 control register              | 18.7.2    |
| PPSn1                 | 8/16-bit PPG timer n1 cycle setting buffer register | 18.7.3    |
| PPSn0                 | 8/16-bit PPG timer n0 cycle setting buffer register | 18.7.3    |
| PDSn1                 | 8/16-bit PPG timer n1 duty setting buffer register  | 18.7.4    |
| PDSn0                 | 8/16-bit PPG timer n0 duty setting buffer register  | 18.7.4    |
| PPGS                  | 8/16-bit PPG start register                         | 18.7.5    |
| REVC                  | 8/16-bit PPG output reverse register                | 18.7.6    |



## MB95850K/860K/870K Series

### 18.7.1 8/16-bit PPG timer n1 Control Register (PCn1)

The 8/16-bit PPG timer n1 control register (PCn1) sets the operating conditions for PPG timer n1.

#### ■ Register Configuration

| bit           | 7 | 6 | 5    | 4    | 3     | 2     | 1     | 0     |
|---------------|---|---|------|------|-------|-------|-------|-------|
| Field         | — | — | PIE1 | PUF1 | POEN1 | CKS12 | CKS11 | CKS10 |
| Attribute     | — | — | R/W  | R/W  | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0    | 0    | 0     | 0     | 0     | 0     |

#### ■ Register Functions

##### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation,

##### [bit5] PIE1: Interrupt request enable bit

This bit controls interrupts of PPG timer n1.

The bit outputs an interrupt request (IRQ) when the counter borrow detection bit (PUF1) and the PIE1 bit are both set to "1".

| bit5        | Details                              |
|-------------|--------------------------------------|
| Writing "0" | Disables the PPG timer n1 interrupt. |
| Writing "1" | Enables the PPG timer n1 interrupt.  |

##### [bit4] PUF1: Counter borrow detection flag bit for PPG cycle downcounter

This bit serves as the counter borrow detection flag for the PPG cycle downcounter of the PPG timer n1.

This bit is set to "1" when a counter borrow occurs in 8-bit PPG independent mode or 8-bit prescaler + 8-bit PPG mode.

In 16-bit PPG mode, this bit is not set to "1" even when a counter borrow occurs.

Writing "1" to this bit has no effect on operation. Writing "0" to this bit clears it.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit4        | Details   |
|-------------|---|
| Reading "0" | Indicates that no counter borrow of PPG timer n1 has been detected. |
| Reading "1" | Indicates that a counter borrow of PPG timer n1 has been detected.  |
| Writing "0" | Clears this bit.  |
| Writing "1" | Has no effect on operation.   |

##### [bit3] POEN1: Output enable bit

This bit enables or disables the PPG timer n1 pin output.

Setting this bit to "1" in 16-bit PPG mode sets the PPG timer n1 pin as an output pin. (The setting value of REV01 is output. "L" output is supplied when REV01 is "0".)

| bit3        | Details   |
|-------------|---|
| Writing "0" | The PPG timer n1 pin functions as a general-purpose I/O port. |
| Writing "1" | The PPG timer n1 pin functions as a PPG output pin.           |

[bit2:0] CKS1[2:0]: Operating clock select bits

These bits select the operating clock for 8-bit downcounter of the PPG timer n1.

The operating clock is generated from the prescaler. For details, see "3.9 Operation of Prescaler".

In 16-bit PPG mode, the settings of these bits have no effect on the operation.

| bit2:0        | Details<br>(MCLK: machine clock, F <sub>CH</sub> : main clock,<br>F <sub>CRH</sub> : main CR clock, F <sub>MCRPLL</sub> : main CR PLL clock) |
|---------------|--|
| Writing "000" | 1 MCLK   |
| Writing "001" | MCLK/2   |
| Writing "010" | MCLK/4   |
| Writing "011" | MCLK/8   |
| Writing "100" | MCLK/16  |
| Writing "101" | MCLK/32  |
| Writing "110" | F <sub>CH</sub> /2 <sup>7</sup> or F <sub>CRH</sub> /2 <sup>6</sup> or F <sub>MCRPLL</sub> /2 <sup>6</sup>                                   |
| Writing "111" | F <sub>CH</sub> /2 <sup>8</sup> or F <sub>CRH</sub> /2 <sup>7</sup> or F <sub>MCRPLL</sub> /2 <sup>7</sup>                                   |

Note: In subclock mode or sub-CR clock mode, since the time-base timer stops operating, setting CKS1[2:0] to "0b110" or "0b111" is prohibited.

## MB95850K/860K/870K Series

### 18.7.2 8/16-bit PPG timer n0 Control Register (PCn0)

The 8/16-bit PPG timer n0 control register (PCn0) sets the operating conditions and the operation mode for PPG timer n0.

#### Register Configuration

| bit           | 7   | 6   | 5    | 4    | 3     | 2     | 1     | 0     |
|---------------|-----|-----|------|------|-------|-------|-------|-------|
| Field         | MD1 | MD0 | PIE0 | PUF0 | POEN0 | CKS02 | CKS01 | CKS00 |
| Attribute     | R/W | R/W | R/W  | R/W  | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0   | 0   | 0    | 0    | 0     | 0     | 0     | 0     |

#### Register Functions

[bit7:6] MD[1:0]: Operating mode bits

These bits select the PPG operation mode.

Do not modify the settings of these bits during counting.

| bit7:6       | Details                          |
|--------------|----------------------------------|
| Writing "00" | 8-bit PPG independent mode       |
| Writing "01" | 8-bit prescaler + 8-bit PPG mode |
| Writing "10" | 16-bit PPG mode                  |
| Writing "11" |                                  |

[bit5] PIE0: Interrupt request enable bit

This bit controls interrupts of PPG timer n0.

In 16-bit PPG mode, use this bit to control the interrupt request of the 8/16-bit PPG.

The bit outputs an interrupt request (IRQ) when the counter borrow detection bit (PUF0) and the PIE0 bit are both set to "1".

| bit5        | Details                              |
|-------------|--------------------------------------|
| Writing "0" | Disables the PPG timer n0 interrupt. |
| Writing "1" | Enables the PPG timer n0 interrupt.  |

[bit4] PUF0: Counter borrow detection flag bit for PPG cycle downcounter

This bit serves as the counter borrow detection flag for the PPG cycle downcounter of the PPG timer n0.

In 16-bit PPG mode, only this bit is effective and the PUF1 bit in the PCn1 register has no effect on operation.

Note: In 8-bit PPG independent mode or 8-bit prescaler + 8-bit PPG mode, counter borrow detection is always enabled.

Writing "1" to this bit has no effect on operation. Writing "0" to this bit clears it.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit4        | Details   |
|-------------|---|
| Reading "0" | Indicates that no counter borrow of PPG timer n0 has been detected. |
| Reading "1" | Indicates that a counter borrow of PPG timer n0 has been detected.  |
| Writing "0" | Clears this bit.  |
| Writing "1" | Has no effect on operation.   |

[bit3] POEN0: Output enable bit

This bit enables or disables the PPG timer n0 pin output.

In 16-bit PPG mode, since the 8/16-bit PPG outputs pulse wave through the PPG timer n0 pin, this bit controls the 8/16-bit PPG output.

| bit3               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | The PPG timer n0 pin functions as a general-purpose I/O port. |
| <b>Writing "1"</b> | The PPG timer n0 pin functions as a PPG output pin.           |

[bit2:0] CKS0[2:0]: Operating clock select bits

These bits select the operating clock for 8-bit downcounter of the PPG timer n0.

The operating clock is generated from the prescaler. For details, see "3.9 Operation of Prescaler".

In 8-bit prescaler + 8-bit PPG mode, the rising and falling edge detection pulses from the PPG timer n1 output are used as the count clock for PPG timer n0. Therefore, the settings of these bits have no effect on operation.

In 16-bit PPG mode, use these bits to select the operating clock.

| bit2:0               | Details<br>(MCLK: machine clock, F <sub>CH</sub> : main clock,<br>F <sub>CRH</sub> : main CR clock, F <sub>MCRPLL</sub> : main CR PLL clock) |
|----------------------|--|
| <b>Writing "000"</b> | 1 MCLK   |
| <b>Writing "001"</b> | MCLK/2   |
| <b>Writing "010"</b> | MCLK/4   |
| <b>Writing "011"</b> | MCLK/8   |
| <b>Writing "100"</b> | MCLK/16  |
| <b>Writing "101"</b> | MCLK/32  |
| <b>Writing "110"</b> | F <sub>CH</sub> /2 <sup>7</sup> or F <sub>CRH</sub> /2 <sup>6</sup> or F <sub>MCRPLL</sub> /2 <sup>6</sup>                                   |
| <b>Writing "111"</b> | F <sub>CH</sub> /2 <sup>8</sup> or F <sub>CRH</sub> /2 <sup>7</sup> or F <sub>MCRPLL</sub> /2 <sup>7</sup>                                   |

Note: In subclock mode or sub-CR clock mode, since the time-base timer stops operating, setting CKS0[2:0] to "0b110" or "0b111" is prohibited.

## MB95850K/860K/870K Series

### 18.7.3 8/16-bit PPG timer n1/n0 Cycle Setup Buffer Register (PPSn1/PPSn0)

The 8/16-bit PPG timer n1/n0 cycle setup buffer register (PPSn1/PPSn0) sets the PPG output cycle.

#### ■ Register Configuration

| PPSn1         |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | PH7 | PH6 | PH5 | PH4 | PH3 | PH2 | PH1 | PH0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 1   |

| PPSn0         |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | PL7 | PL6 | PL5 | PL4 | PL3 | PL2 | PL1 | PL0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 1   |

#### ■ Register Functions

The PPSn1 and PPSn0 registers set the PPG output cycle.

- In 16-bit PPG mode, PPSn1 serves as the upper 8 bits, while PPSn0 serves as the lower 8 bits.
- In 16-bit PPG mode, write the upper bits before the lower bits. When only the upper bits are written, the previously written value is reused in the next load.
- 8-bit mode: Cycle = max. 255 (0xFF) × Input clock cycle
- 16-bit mode: Cycle = max. 65535 (0xFFFF) × Input clock cycle
- PPSn1 and PPSn0 are initialized upon reset.
- Do not set the cycle to "0x00" or "0x01" when using the unit in 8-bit PPG independent mode or 8-bit prescaler mode + 8-bit PPG mode.
- Do not set the cycle to "0x0000" or "0x0001" when using the unit in 16-bit PPG mode.
- If the cycle settings are modified during the operation, the modified settings will be effective from the next PPG cycle.

## 18.7.4 8/16-bit PPG timer n1/n0 Duty Setup Buffer Register (PDSn1/PDSn0)

The 8/16-bit PPG timer n1/n0 duty setup buffer register (PDSn1/PDSn0) sets the duty of the PPG output.

### ■ Register Configuration

| PDSn1         |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | DH7 | DH6 | DH5 | DH4 | DH3 | DH2 | DH1 | DH0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 1   |

| PDSn0         |     |     |     |     |     |     |     |     |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| Field         | DL7 | DL6 | DL5 | DL4 | DL3 | DL2 | DL1 | DL0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 1   |

### ■ Register Functions

The PDSn1 and PDSn0 registers set the duty of the PPG output ("H" pulse width when normal polarity).

- In 16-bit PPG mode, PDSn1 serves as the upper 8 bits while PDSn0 serves as the lower 8 bits.
- In 16-bit PPG mode, write the upper bits before the lower bits. When only the upper bits are written, the previously written value is reused in the next load. Writing data to PDSn0 also updates PDSn1 at the same time.
- PDSn1 and PDSn0 are initialized upon reset.
- To set the duty to 0%, select "0x00".
- To set the duty to 100%, set it to the same value as the 8/16-bit PPG timer n1/n0 cycle setup register (PPSn0, PPSn1).
- When the 8/16-bit PPG timer n0/n1 duty setup register (PDS) is set to a larger value than the setting value of the 8/16-bit PPG cycle setup buffer register (PPS), the PPG output becomes "L" output in the normal polarity (when the output level inversion bit of 8/16-bit PPG output inversion register is "0").
- If the duty settings are modified during operation, the modified value will be effective from the next PPG cycle.

## MB95850K/860K/870K Series

### 18.7.5 8/16-bit PPG Start Register (PPGS)

The 8/16-bit PPG start register (PPGS) starts or stops the downcounter. The operation enable bit of each channel is assigned to the PPGS register, allowing simultaneous activation of the PPG channels.

#### ■ Register Configuration

| bit           | 7 | 6 | 5      | 4      | 3      | 2      | 1     | 0     |
|---------------|---|---|--------|--------|--------|--------|-------|-------|
| Field         | — | — | PEN21* | PEN20* | PEN11* | PEN10* | PEN01 | PEN00 |
| Attribute     | — | — | R/W    | R/W    | R/W    | R/W    | R/W   | R/W   |
| Initial value | 0 | 0 | 0      | 0      | 0      | 0      | 0     | 0     |

\*: Since the number of channels of the 8/16-bit PPG varies among products, these bits may become undefined bits on certain products. For details, refer to the device data sheet.

#### ■ Register Functions

##### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit5] PEN21: PPG timer 21 (ch. 2) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 21 (ch. 2).

| bit5               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 21 (ch. 2).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 21 (ch. 2). |

##### [bit4] PEN20: PPG timer 20 (ch. 2) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 20 (ch. 2).

| bit4               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 20 (ch. 2).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 20 (ch. 2). |

##### [bit3] PEN11: PPG timer 11 (ch. 1) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 11 (ch. 1).

| bit3               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 11 (ch. 1).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 11 (ch. 1). |

##### [bit2] PEN10: PPG timer 10 (ch. 1) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 10 (ch. 1).

| bit2               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 10 (ch. 1).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 10 (ch. 1). |

[bit1] PEN01: PPG timer 01 (ch. 0) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 01 (ch. 0).

| bit1               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 01 (ch. 0).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 01 (ch. 0). |

[bit0] PEN00: PPG timer 00 (ch. 0) downcounter operation enable bit

This bit enables or stops the downcounter operation of PPG timer 00 (ch. 0).

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Stops the downcounter operation of PPG timer 00 (ch. 0).   |
| <b>Writing "1"</b> | Enables the downcounter operation of PPG timer 00 (ch. 0). |



## MB95850K/860K/870K Series

### 18.7.6 8/16-bit PPG Output Reverse Register (REVC)

The 8/16-bit PPG output reverse register (REVC) reverses the PPG output including the initial level.

#### Register Configuration

| bit           | 7 | 6 | 5      | 4      | 3      | 2      | 1     | 0     |
|---------------|---|---|--------|--------|--------|--------|-------|-------|
| Field         | — | — | REV21* | REV20* | REV11* | REV10* | REV01 | REV00 |
| Attribute     | — | — | R/W    | R/W    | R/W    | R/W    | R/W   | R/W   |
| Initial value | 0 | 0 | 0      | 0      | 0      | 0      | 0     | 0     |

\*: Since the number of channels of the 8/16-bit PPG varies among products, these bits may become undefined bits on certain products. For details, refer to the device data sheet.

#### Register Functions

##### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit5] REV21: PPG timer 21 (ch. 2) output level reverse bit

This bit selects the output level of PPG timer 21 (ch. 2).

| bit5               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

##### [bit4] REV20: PPG timer 20 (ch. 2) output level reverse bit

This bit selects the output level of PPG timer 20 (ch. 2).

| bit4               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

##### [bit3] REV11: PPG timer 11 (ch. 1) output level reverse bit

This bit selects the output level of PPG timer 11 (ch. 1).

| bit3               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

##### [bit2] REV10: PPG timer 10 (ch. 1) output level reverse bit

This bit selects the output level of PPG timer 10 (ch. 1).

| bit2               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

[bit1] REV01: PPG timer 01 (ch. 0) output level reverse bit  
This bit selects the output level of PPG timer 01 (ch. 0).

| bit1               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

[bit0] REV00: PPG timer 00 (ch. 0) output level reverse bit  
This bit selects the output level of PPG timer 00 (ch. 0).

| bit0               | Details                   |
|--------------------|---------------------------|
| <b>Writing "0"</b> | Selects normal polarity.  |
| <b>Writing "1"</b> | Selects reverse polarity. |

## MB95850K/860K/870K Series

### 18.8 Notes on Using 8/16-bit PPG

---

This section provides notes on using the 8/16-bit PPG.

---

#### ■ Notes on Using 8/16-bit PPG

- Note on operation

Depending on the timing between the activation of PPG and count clock, an error may occur in the first cycle of the PPG output immediately after the activation. The error varies depending on the count clock selected. The output, however, is performed properly in the succeeding cycles.

- Note on interrupts

A PPG interrupt is generated when the interrupt enable bit (PIE1/PIE0) is set to "1" and the interrupt request flag bit (PUF1/PUF0) in the 8/16-bit PPG timer n1/n0 control register (PCn1/PCn0) is also set to "1". Always clear the interrupt request flag bit (PUF1/PUF0) to "0" in the interrupt service routine.

# MB95850K/860K/870K Series

# **CHAPTER 19**

---

## **UART/SIO**

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

- 19.1 Overview
- 19.2 Configuration
- 19.3 Channel
- 19.4 Pins
- 19.5 Interrupts
- 19.6 Operations and Setting Procedure Example
- 19.7 Registers

## 19.1 Overview

The UART/SIO is a general-purpose serial data communication interface. Serial data transfers of variable-length data can be made with a synchronous or asynchronous clock. The transfer format is NRZ. The transfer rate can be set with the dedicated baud rate generator or external clock (in clock synchronous mode (SIO)).

### ■ Functions of UART/SIO

The UART/SIO is capable of serial data transmission/reception (serial input/output) to and from another CPU or peripheral device.

- Equipped with a full-duplex double buffer that allows 2-way full-duplex communication.
- The synchronous or asynchronous transfer mode can be selected.
- The optimum baud rate can be selected with the dedicated baud rate generator.
- The data length is variable; it can be set to 5 bits to 8 bits when no parity is used or to 6 bits to 9 bits when parity is used. (See Table 19.1-1.)
- The serial data direction (endian) can be selected.
- The data transfer format is NRZ (Non-Return-to-Zero).
- Two operation modes (operation modes 0 and 1) are available.  
Operation mode 0 operates as clock asynchronous mode (UART).  
Operation mode 1 operates as clock synchronous mode (SIO).

**Table 19.1-1 UART/SIO Operation Modes**

| Operation mode | Data length |             | Synchronization mode | Length of stop bit |
|----------------|-------------|-------------|----------------------|--------------------|
|                | No parity   | With parity |                      |                    |
| 0              | 5           | 6           | Asynchronous         | 1 bit or 2 bits    |
|                | 6           | 7           |                      |                    |
|                | 7           | 8           |                      |                    |
|                | 8           | 9           |                      |                    |
| 1              | 5           | -           | Synchronous          | -                  |
|                | 6           | -           |                      |                    |
|                | 7           | -           |                      |                    |
|                | 8           | -           |                      |                    |

# MB95850K/860K/870K Series

## 19.2 Configuration

The UART/SIO consists of the following blocks:

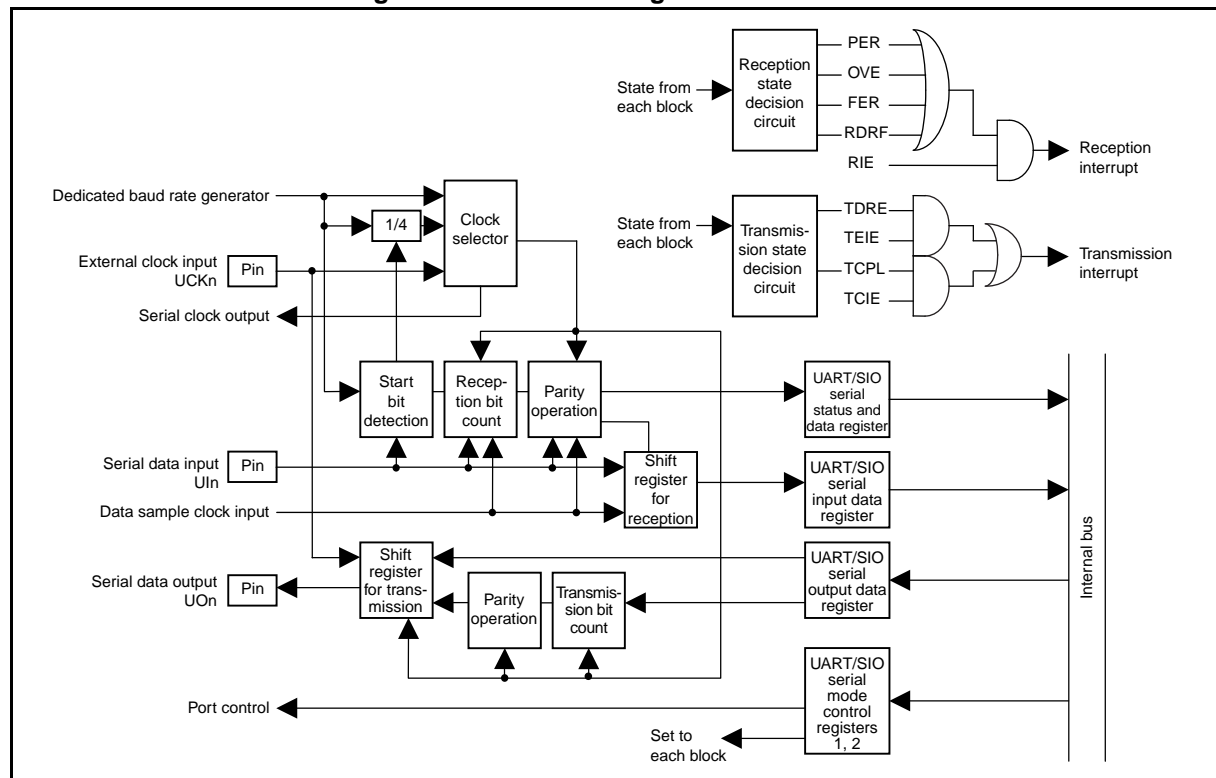
- UART/SIO serial mode control register 1 ch. n (SMC1n)
- UART/SIO serial mode control register 2 ch. n (SMC2n)
- UART/SIO serial status and data register ch. n (SSRn)
- UART/SIO serial input data register ch. n (RDRn)
- UART/SIO serial output data register ch. n (TDRn)

The number of pins and that of channels of the UART/SIO vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

### ■ Block Diagram of UART/SIO

Figure 19.2-1 Block Diagram of UART/SIO



- UART/SIO serial mode control register 1 ch. n (SMC1n)

This register controls UART/SIO operation mode. It is used to set the serial data direction (endian), parity and its polarity, stop bit length, operation mode (synchronous/asynchronous), data length, and serial clock.

- UART/SIO serial mode control register 2 ch. n (SMC2n)

This register controls UART/SIO operation mode. It is used to enable/disable serial clock output, serial data output, transmission/reception, and interrupts and to clear the receive error flag.

- UART/SIO serial status and data register ch. n (SSRn)

This register indicates the transmission/reception status and error status of UART/SIO.

- UART/SIO serial input data register ch. n (RDRn)

This register holds the receive data. The serial input is converted and then stored in this register.

- UART/SIO serial output data register ch. n (TDRn)

This register sets the transmit data. Data written to this register is serial-converted and then output.

### ■ Input Clock

The UART/SIO uses the output clock (internal clock) from the dedicated baud rate generator or the input signal (external clock) from the UCKn pin as its input clock (serial clock).



**MB95850K/860K/870K Series****19.3 Channel**

---

This section describes the channel of UART/SIO.

---

**■ Channel of UART/SIO**

Table 19.3-1 and Table 19.3-2 show the pins and registers of UART/SIO respectively.

**Table 19.3-1 Pins of UART/SIO**

| Pin name | Pin function       |
|----------|--------------------|
| UCKn     | Clock input/output |
| UOn      | Data output        |
| UIn      | Data input         |

**Table 19.3-2 Registers of UART/SIO**

| Register abbreviation | Corresponding register (Name in this manual)   |
|-----------------------|--|
| SMC1n                 | UART/SIO serial mode control register 1 ch. n  |
| SMC2n                 | UART/SIO serial mode control register 2 ch. n  |
| SSRn                  | UART/SIO serial status and data register ch. n |
| TDRn                  | UART/SIO serial output data register ch. n     |
| RDRn                  | UART/SIO serial input data register ch. n      |

## 19.4 Pins

---

**This section describes the pins of the UART/SIO.**

---

### ■ Pins of UART/SIO

The pins of UART/SIO are the clock input and output pin (UCKn), serial data output pin (UOn) and serial data input pin (UIn).

- UCKn

Clock input/output pin for UART/SIO.

When the clock output is enabled (SMC2n:SCKE=1), it serves as a UART/SIO clock output pin (UCKn) regardless of the value of the corresponding port direction register. At this time, do not select the external clock (set SMC1n:CKS = 0).

When it is to be used as a UART/SIO clock input pin, disable the clock output (SMC2n:SCKE = 0) and make sure that it is set as input port by the corresponding port direction register. At this time, be sure to select the external clock (set SMC1n:CKS = 0).

- UOn

Serial data output pin for UART/SIO. When the serial data output is enabled (SMC2n:TXOE = 1), it serves as a UART/SIO serial data output pin (UOn) regardless of the value of the corresponding port direction register.

- UIn

Serial data input pin for UART/SIO. When it is to be used as a UART/SIO serial data input pin, make sure that it is set as input port by the corresponding port direction register.

## MB95850K/860K/870K Series

### 19.5 Interrupts

The UART/SIO has six interrupt-related bits: receive error flag bits (PER, OVE, FER), receive data register full flag bit (RDRF), transmit data register empty flag bit (TDRE), and transmission completion flag bit (TCPL).

#### ■ Interrupts of UART/SIO

Table 19.5-1 lists the UART/SIO interrupt control bits and interrupt sources.

**Table 19.5-1 UART/SIO Interrupt Control Bits and Interrupt Sources**

| Item                         | Description                  |                         |                   |              |               |               |
|------------------------------|------------------------------|-------------------------|-------------------|--------------|---------------|---------------|
| Interrupt request flag bit   | SSRn:TDRE                    | SSRn:TCPL               | SSRn:RDRF         | SSRn:PER     | SSRn:OVE      | SSRn:FER      |
| Interrupt request enable bit | SMC2n:TEIE                   | SMC2n:TCIE              | SMC2n:RIE         | SMC2n:RIE    | SMC2n:RIE     | SMC2n:RIE     |
| Interrupt source             | Transmit data register empty | Transmission completion | Receive data full | Parity error | Overrun error | Framing error |

#### ■ Transmit Interrupt

When transmit data is written to the UART/SIO serial output data register ch. n (TDRn), the data is transferred to the transmission shift register. When the next piece of data can be written, the TDRE bit is set to "1". At this time, an interrupt request to the interrupt controller occurs when transmit data register empty interrupt enable bit has been enabled (SMC2n:TEIE = 1).

The TCPL bit is set to "1" upon completion of transmission of all pieces of transmit data. At this time, an interrupt request to the interrupt controller occurs when transmission completion interrupt enable bit has been enabled (SMC2n:TCIE = 1).

#### ■ Receive Interrupt

If the data is input successfully up to the stop bit, the RDRF bit is set to "1". If an overrun error, a parity error, or a framing error occurs, the corresponding error flag bit (PER, OVE, or FER) is set to "1".

These bits are set when a stop bit is detected. If receive interrupt enable bit has been enabled (SMC2n:RIE = 1), an interrupt request to the interrupt controller will be generated.

## 19.6 Operations and Setting Procedure Example

The UART/SIO has a serial communication function (operation mode 0, 1).

### ■ Operations of UART/SIO

#### ● Operation mode

Two operation modes are available in the UART/SIO. Clock synchronous mode (SIO) or clock asynchronous mode (UART) can be selected (See Table 19.6-1).

**Table 19.6-1 Operation Modes of UART/SIO**

| Operation mode | Data length |             | Synchronization mode | Length of stop bit |
|----------------|-------------|-------------|----------------------|--------------------|
|                | No parity   | With parity |                      |                    |
| 0              | 5           | 6           | Asynchronous         | 1 bit or 2 bits    |
|                | 6           | 7           |                      |                    |
|                | 7           | 8           |                      |                    |
|                | 8           | 9           |                      |                    |
| 1              | 5           | -           | Synchronous          | -                  |
|                | 6           | -           |                      |                    |
|                | 7           | -           |                      |                    |
|                | 8           | -           |                      |                    |

### ■ Setting Procedure Example

Below is an example of procedure for setting the UART/SIO.

#### ● Initial setup

1. Set the port input. (DDR)
2. Set the interrupt level. (ILR\*)
3. Set the prescaler. (PSSRn)
4. Set the baud rate. (BRSRn)
5. Select the clock. (SMC1n:CKS)
6. Set the operation mode. (SMC1n:MD)
7. Enable/disable the serial clock output. (SMC2n:SCKE)
8. Enable reception. (SMC2n:RXE = 1)
9. Enable interrupts. (SMC2n:RIE = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

#### ● Interrupt processing

Read receive data. (RDRn)

## 19.6.1 Operations in Operation Mode 0

Operation mode 0 operates as clock asynchronous mode (UART).

### ■ Operations in UART/SIO Operation Mode 0

Clock asynchronous mode (UART) is selected when the MD bit in the UART/SIO serial mode control register 1 ch. n (SMC1n) is set to "0".

#### ● Baud rate

The serial clock is selected by the CKS bit in the SMC1n register. Be sure to select the dedicated baud rate generator at this time.

The baud rate is equivalent to the output clock frequency of the dedicated baud rate generator, divided by four. The UART can perform communication within the range from -3% to +3% of the selected baud rate.

The baud rate generated by the dedicated baud rate generator is obtained from the equation illustrated below. (For information about the dedicated baud rate generator, see "CHAPTER 20 UART/SIO DEDICATED BAUD RATE GENERATOR".)

**Figure 19.6-1 Baud Rate Calculation when Using Dedicated Baud Rate Generator**

$$\text{Baud rate value} = \frac{\text{Machine clock (MCLK)}}{4 \times \begin{matrix} 1 \\ 2 \\ 4 \\ 8 \end{matrix}} \times \begin{matrix} 2 \\ : \\ 255 \end{matrix} \quad [\text{bps}]$$

UART prescaler select register (PSSRn)  
Prescaler select (PSS[1:0])
UART baud rate setting register (BRSRn)  
Baud rate setting (BRS[7:0])

**Table 19.6-2 Sample Asynchronous Transfer Rates Based on Dedicated Baud Rate Generator (Machine clock = 10 MHz, 16 MHz, 16.25 MHz)**

| Dedicated baud rate generator setting |                                    | UART internal division | Total division ratio (PSS × BRS × 4) | Baud rate (10 MHz / Total division ratio) | Baud rate (16 MHz / Total division ratio) | Baud rate (16.25 MHz / Total division ratio) |
|---------------------------------------|------------------------------------|------------------------|--------------------------------------|---|---|--|
| Prescaler select PSS[1:0]             | Baud rate counter setting BRS[7:0] |                        |                                      |   |   |  |
| 1 (Setting value: 0,0)                | 20                                 | 4                      | 80                                   | 125000                                    | 200000                                    | 203125                                       |
| 1 (Setting value: 0,0)                | 22                                 | 4                      | 88                                   | 113636                                    | 181818                                    | 184659                                       |
| 1 (Setting value: 0,0)                | 44                                 | 4                      | 176                                  | 56818                                     | 90909                                     | 92330  |
| 1 (Setting value: 0,0)                | 87                                 | 4                      | 348                                  | 28736                                     | 45977                                     | 46695  |
| 1 (Setting value: 0,0)                | 130                                | 4                      | 520                                  | 19231                                     | 30769                                     | 31250  |
| 2 (Setting value: 0,1)                | 130                                | 4                      | 1040                                 | 9615                                      | 15385                                     | 15625  |
| 4 (Setting value: 1,0)                | 130                                | 4                      | 2080                                 | 4808                                      | 7692                                      | 7813   |
| 8 (Setting value: 1,1)                | 130                                | 4                      | 4160                                 | 2404                                      | 3846                                      | 3906   |

The baud rate in clock asynchronous mode (UART) can be set in the following range.

**Table 19.6-3 Baud Rate Setting Range in Clock Asynchronous Mode (UART)**

| PSS[1:0]     | BRS[7:0]               |
|--------------|------------------------|
| 0b00 to 0b11 | 0x02 (2) to 0xFF (255) |

● Transfer data format

UART can treat data only in NRZ (Non-Return-to-Zero) format. Figure 19.6-2 shows the data format.

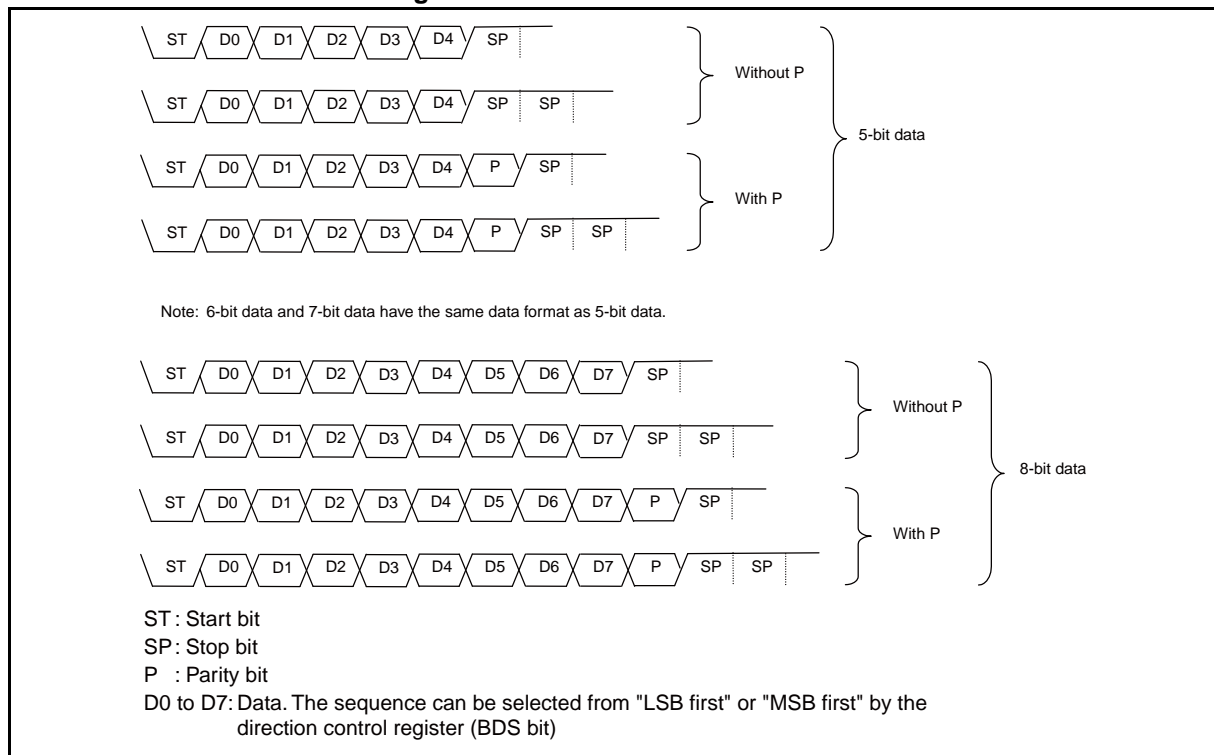
The character bit length can be selected from among 5 to 8 bits depending on the settings of SMC1n:CBL[1:0].

The stop bit length can be set to 1 or 2 bits depending on the setting of SMC1n:SBL.

The PEN bit and TDP bit in the SMC1n register can be used to enable/disable parity and to select parity polarity.

As shown in Figure 19.6-2, the transfer data always starts from the start bit ("L" level) and ends with the stop bit ("H" level) by performing the specified data bit length transfer with MSB first or LSB first ("LSB first" or "MSB first" can be selected by the BDS bit in the SMC1n register). It becomes "H" level at the idle state.

**Figure 19.6-2 Transfer Data Format**



● Reception in clock asynchronous mode (UART)

Use UART/SIO serial mode control register 1 ch. n (SMC1n) to select the serial data direction (endian), parity/non-parity, parity polarity, stop bit length, character bit length, and clock.

Reception remains performed as long as the reception operation enable bit (SMC2n:RXE) contains "1".

Upon detection of a start bit in receive data with the RXE bit set to "1", one frame of data is received according to the data format set in UART/SIO serial control register 1 ch. n (SMC1n).

When the reception of one frame of data has been completed, the received data is transferred to the UART/SIO serial input data register ch. n (RDRn) and the next frame of serial data can be received.

When the RDRn register stores data, the receive data register full flag bit (SSRn:RDRF) is set to "1".

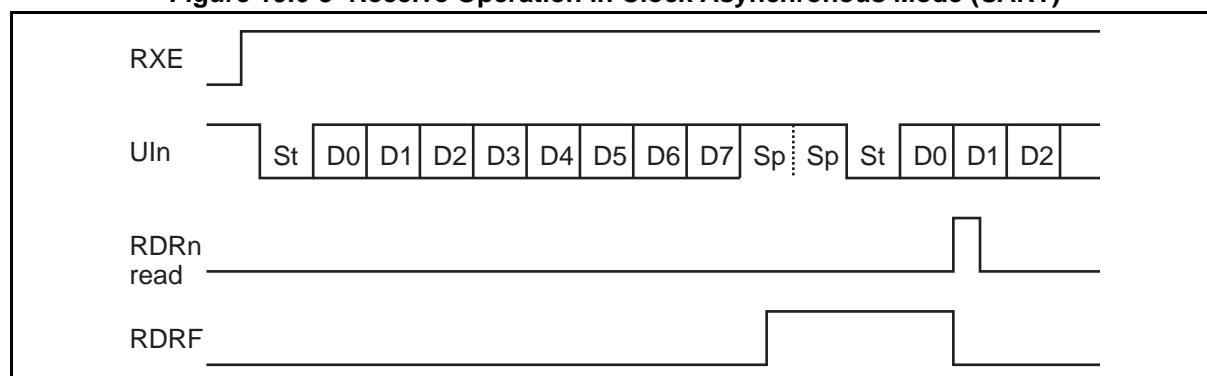
A receive interrupt occurs the moment the RDRF bit is set to "1" when the receive interrupt enable bit (SMC2n:RIE) contains "1".

Received data is read from the RDRn register after each error flag (PER, OVE, FER) in the UART/SIO serial status and data register ch. n (SSRn) is checked.

When received data is read from the RDRn register, the RDRF bit is cleared to "0".

Note that modifying the SMC1n register during reception may result in unpredictable operation. If the RXE bit is set to "0" during reception, the reception is immediately disabled and initialization will be performed. The data received up to that point will not be transferred to the serial input data register.

Figure 19.6-3 Receive Operation in Clock Asynchronous Mode (UART)



● Receive error in clock asynchronous mode (UART)

If any of the following three error flags (PER, FER, OVE) has been set, receive data is not transferred to the UART/SIO serial input data register ch. n (RDRn) and the receive data register full flag bit (RDRF) is not set to "1" either.

- Parity error (PER)

The parity error bit (PER) is set to "1" if the parity bit in received serial data does not match the parity polarity bit (TDP) when the parity control bit (PEN) contains "1".

- Framing error (FER)

The framing error bit (FER) is set to "1" if "1" is not detected at the position of the first stop bit in serial data received in the set character bit length (CBL) under parity control (PEN).

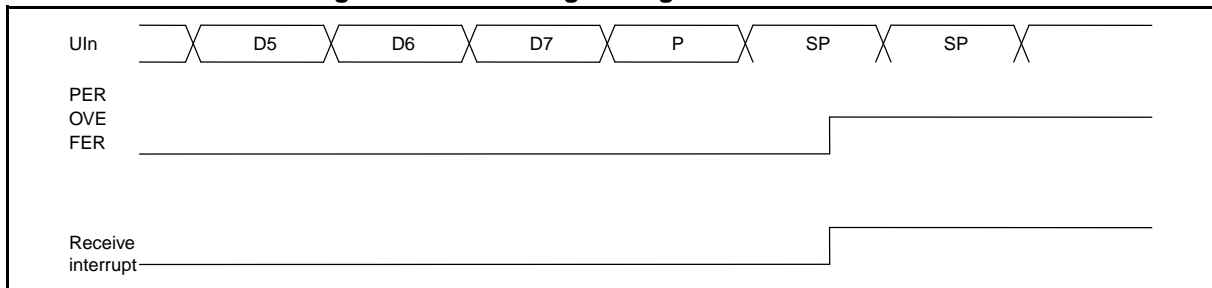
Note that the stop bit is not checked if it appears at the second bit or later.

- Overrun error (OVE)

Upon completion of reception of serial data, the overrun error bit (OVE) is set to "1" if the reception of the next data is performed before the previous receive data is read.

Each flag is set at the position of the first stop bit.

**Figure 19.6-4 Setting Timing for Receive Errors**





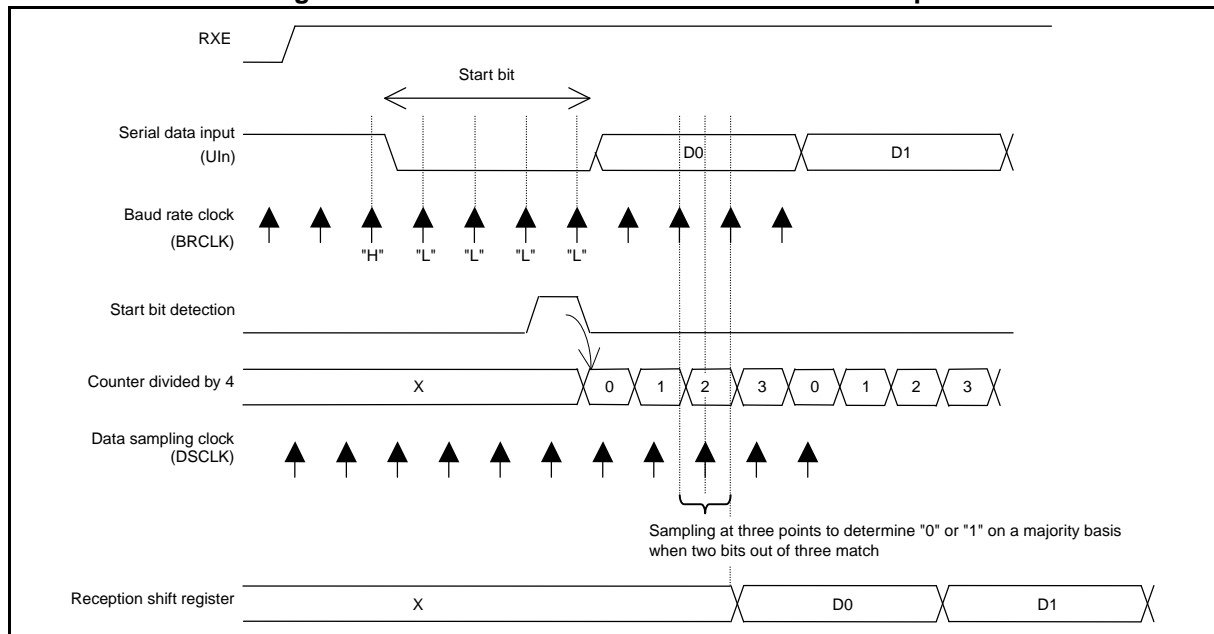
● Start bit detection and confirmation of receive data during reception

The start bit is detected by a falling of the serial input followed by a succession of three "L" levels after the serial data input is sampled according to the clock (BRCLK) signal provided by the dedicated baud rate generator with the reception operation enable bit (RXE) set to "1". When the first "H", "L", "L", "L" train is detected in a BRCLK sample, therefore, the current bit is regarded as the start bit.

The frequency-quartered circuit is activated upon detection of the start bit and serial data is input to the reception shift register at intervals of four periods of BRCLK.

When data is received, sampling is performed at three points of the baud rate clock (BRCLK) and data sampling clock (DSCLK) and received data is confirmed on a majority basis when two bits out of three match.

Figure 19.6-5 Start Bit Detection and Serial Data Input



● Transmission in clock asynchronous mode (UART)

Use UART/SIO serial mode control register 1 ch. n (SMC1n) to select the serial data direction (endian), parity/non-parity, parity polarity, stop bit length, character bit length, and clock.

Either of the following two procedures can be used to initiate the transmission process:

- Set the transmission operation enable bit (TXE) to "1", and then write transmit data to the UART/SIO serial output data register ch. n (TDRn) to start transmission.
- Write transmit data to the UART/SIO serial output data register ch. n (TDRn), and then set the transmission operation enable bit (TXE) to "1" to start transmission.

Transmit data is written to the TDRn register after it is checked that the transmit data register empty flag bit (TDRE) set to "1".

When the transmit data is written to the TDRn register, the TDRE bit is cleared to "0".

The transmit data is transferred from the TDRn register to the transmission shift register, and the TDRE bit is set to "1".

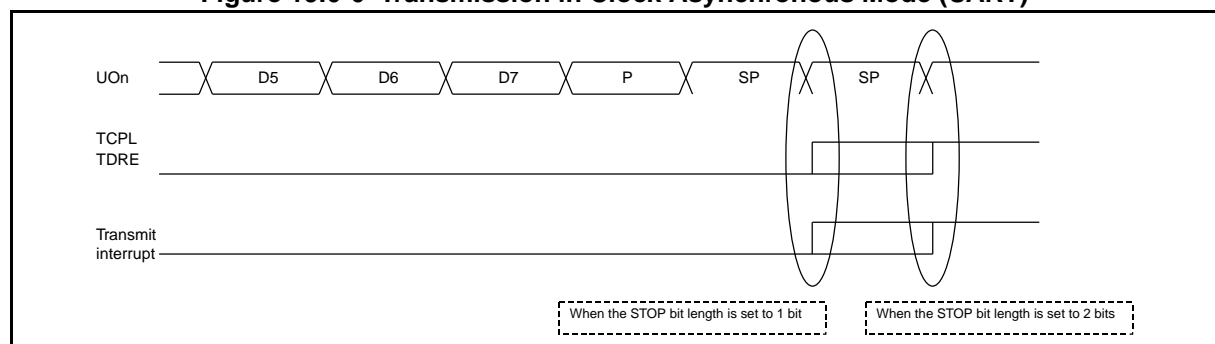
When the transmit interrupt enable bit (TIE) contains "1", a transmit interrupt occurs if the TDRE bit is set to "1". This allows the next piece of transmit data to be written to the TDRn register by interrupt handling.

To detect the completion of serial transmission by transmit interrupt, set the transmission completion interrupt enable bits as follows: TEIE = 0, TCIE = 1. Upon completion of transmission, the transmission completion flag bit (SSRn:TCPL) is set to "1" and a transmit interrupt occurs.

The TCPL bit, and the TDRE bit in consecutive data transmission are set at the position which the transmission of the last bit was completed (it varies depending on the data length, parity enable, or stop bit length setting), as shown in Figure 19.6-6.

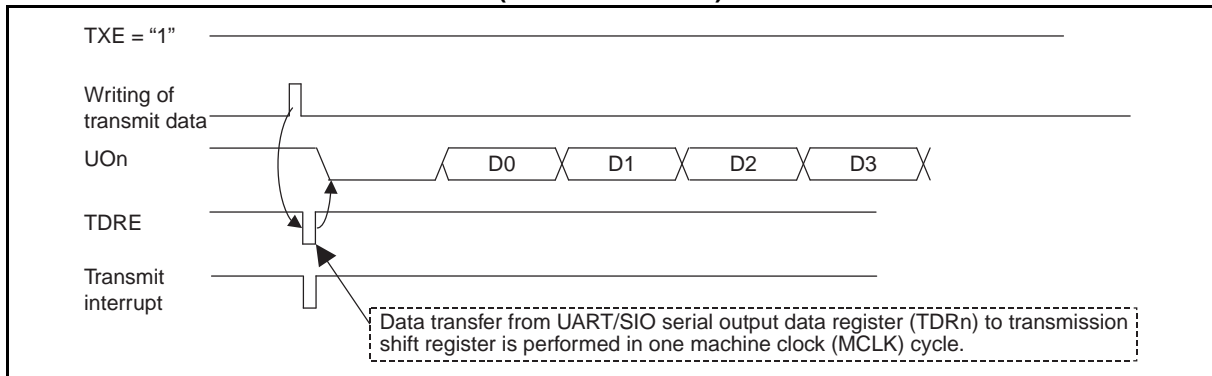
Note that modifying UART/SIO serial mode control register 1 ch. n (SMC1n) during transmission may result in unpredictable operation.

**Figure 19.6-6 Transmission in Clock Asynchronous Mode (UART)**

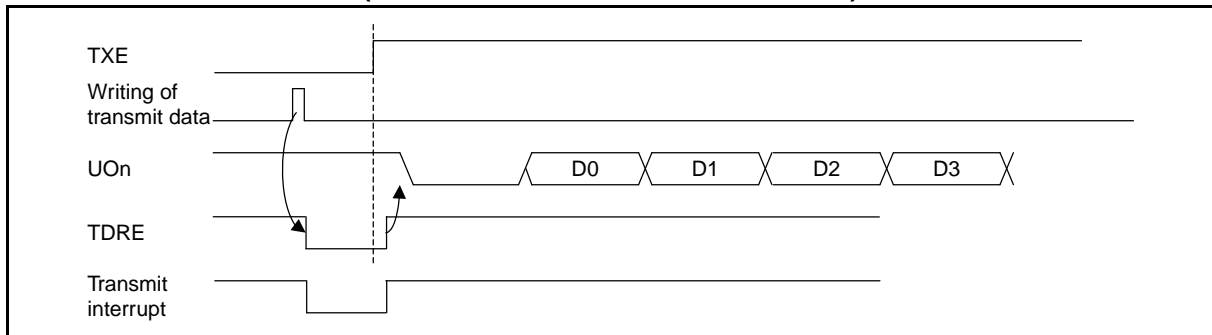


The TDRE bit is set at the point indicated in Figure 19.6-7 or Figure 19.6-8 if the preceding piece of transmit data does not exist in the transmission shift register.

**Figure 19.6-7 Setting Timing 1 for Transmit Data Register Empty Flag Bit (TDRE) (When TXE Is "1")**



**Figure 19.6-8 Setting Timing 2 for Transmit Data Register Empty Flag Bit (TDRE) (When TXE Is Switched from "0" to "1")**



● Concurrent transmission and reception

In clock asynchronous mode (UART), transmission and reception can be performed independently. Therefore, transmission and reception can be performed at the same time or even with transmitting and receiving frames overlapping each other in shifted phases.

## 19.6.2 Operations in Operation Mode 1

**Operation mode 1 operates in clock synchronous mode (SIO).**

### ■ Operations in UART/SIO Operation Mode 1

Setting the MD bit in the UART/SIO serial mode control register 1 ch. n (SMC1n) to "1" selects synchronous clock mode (SIO).

The character bit length in synchronous clock mode (SIO) is variable between 5 bits and 8 bits.

Note, however, that parity is disabled and no stop bit is used.

The serial clock is selected by the CKS bit in the SMC1n register. Select the dedicated baud rate generator or external clock. The SIO performs shift operation using the selected serial clock as a shift clock.

To input the external clock signal, set the SMC2n:SCKE bit to "0".

To output the dedicated baud rate generator output as a shift clock signal, set the SCKE bit to "1". The serial clock signal is obtained by dividing clock by two, which is supplied by the dedicated baud rate generator. The baud rate in the SIO mode can be set in the following range. (For more information about the dedicated baud rate generator, also see "CHAPTER 20 UART/SIO DEDICATED BAUD RATE GENERATOR".)

**Table 19.6-4 Baud Rate Setting Range in Clock Synchronous Mode (SIO)**

| PSS[1:0]     | BRS[7:0]  |
|--------------|---|
| 0b00 to 0b11 | 0x01(1) to 0xFF(255), 0x00(256)<br>(The highest and lowest baud rate settings are 0x01 and 0x00, respectively.) |

The baud rate applied when the external clock or dedicated baud rate generator is used is obtained from the corresponding equation illustrated below.

**Figure 19.6-9 Calculating Baud Rate Based on External Clock**

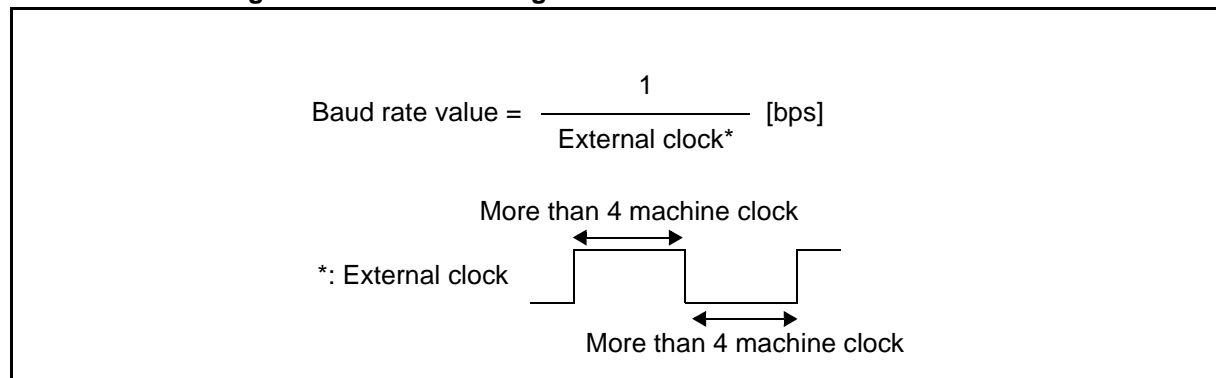


Figure 19.6-10 Baud Rate Calculation Formula for Using Dedicated Baud Rate Generator

$$\text{Baud rate value} = \frac{\text{Machine clock (MCLK)}}{2 \times \begin{matrix} 1 \\ 2 \\ 4 \\ 8 \end{matrix} \times \begin{matrix} 1 \\ : \\ 256 \end{matrix}} \quad [\text{bps}]$$

UART prescaler select register ch. n (PSSRn)  
Prescaler select (PSS[1:0])

UART baud rate setting register ch. n (BRSRn)  
Baud rate setting (BRS[7:0])

- Serial clock

The serial clock signal is output under control of the output for transmit data. When only reception is performed, therefore, set transmission control (SMC2n:TXE = 1) to write dummy transmit data to the UART/SIO serial output register. Refer to the device data sheet for the UCKn clock value.

● Reception in UART/SIO operation mode 1

For reception in operation mode 1, each register is used as shown in Figure 19.6-11.

**Figure 19.6-11 Registers Used for Reception in Operation Mode 1**

|  |      |      |      |      |      |      |      |
|--|------|------|------|------|------|------|------|
| SMC1n (UART/SIO serial mode control register 1)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| BDS  | PEN  | TDP  | SBL  | CBL1 | CBL0 | CKS  | MD   |
| ⊙  | ×    | ×    | ×    | ⊙    | ⊙    | ⊙    | 1    |
| SMC2n (UART/SIO serial mode control register 2)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| SCKE   | TXOE | RERC | RXE  | TXE  | RIE  | TCIE | TEIE |
| ⊙  | 0    | ⊙    | ⊙    | ⊙    | ⊙    | ×    | ×    |
| SSRn (UART/SIO serial status and data register)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| -  | -    | PER  | OVE  | FER  | RDRF | TCPL | TDRE |
| ×  | ×    | ×    | ⊙    | ×    | ⊙    | ×    | ×    |
| TDRn (UART/SIO serial output data register)                        |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| TD7  | TD6  | TD5  | TD4  | TD3  | TD2  | TD1  | TD0  |
| ×  | ×    | ×    | ×    | ×    | ×    | ×    | ×    |
| RDRn (UART/SIO serial input data register)                         |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| RD7  | RD6  | RD5  | RD4  | RD3  | RD2  | RD1  | RD0  |
| ⊙  | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    |
| ⊙ : Used bit<br>× : Unused bit<br>1 : Set to "1"<br>0 : Set to "0" |      |      |      |      |      |      |      |

The reception depends on whether the serial clock has been set to external or internal clock.

**<When external clock is enabled>**

When the reception operation enable bit (RXE) contains "1", serial data is received always at the rising edge of the external clock signal.

**<When internal clock is enabled>**

The serial clock signal is output in accordance with transmission. Therefore, transmission must be performed even when only performing reception. The following two procedures can be used.

- Set the transmission operation enable bit (TXE) to "1", then write transmit data to the UART/SIO serial output data register to generate the serial clock signal and start reception.
- Write transmit data to the TDRn register, then set the TXE bit to "1" to generate the serial clock signal and start reception.

When 5-bit to 8-bit serial data is received by the reception shift register, the received data is transferred to the UART/SIO serial input data register ch. n (RDRn) and the next piece of serial data can be received.

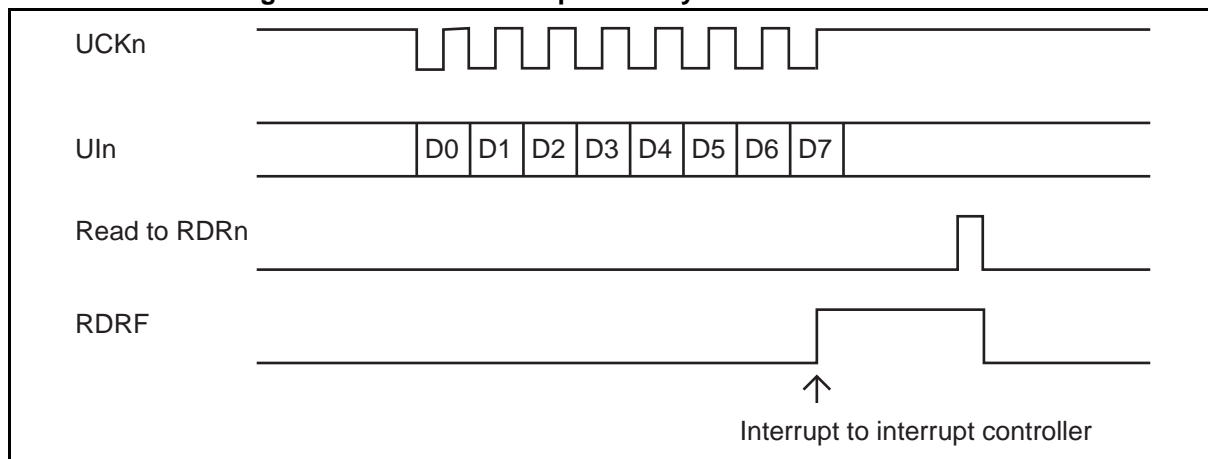
When the RDRn register stores data, the receive data register full flag bit (RDRF) is set to "1".

A receive interrupt occurs the moment the RDRF bit is set to "1" when the receive interrupt enable bit (RIE) contains "1".

To read received data, read it from the RDRn register after checking the overrun error flag bit (OVE) in the UART/SIO serial status and data register ch. n (SSRn).

When received data is read from the RDRn register, the RDRF bit is cleared to "0".

Figure 19.6-12 8-bit Reception of Synchronous Clock Mode



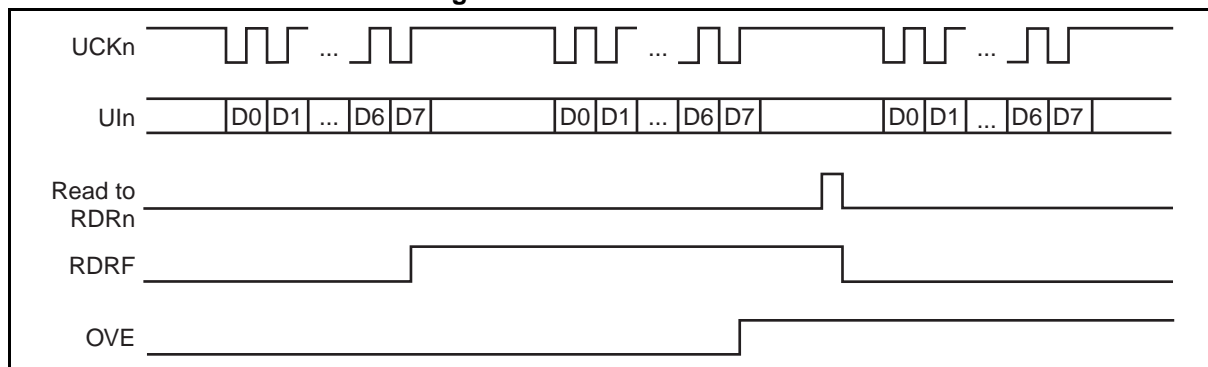
**Operation when receive error occurs**

When an overrun error (OVE = 1) occurs, received data is not transferred to the RDRn register.

**Overrun error (OVE = 1)**

Upon completion of reception for serial data, the OVE bit is set to "1" if the RDRF bit has been set to "1" by the reception for the preceding piece of data.

Figure 19.6-13 Overrun Error



● Transmission in UART/SIO operation mode 1

For transmission in operation mode 1, each register is used as shown below.

**Figure 19.6-14 Registers Used for Transmission in Operation Mode 1**

|  |      |      |      |      |      |      |      |
|--|------|------|------|------|------|------|------|
| SMC1n (UART/SIO serial mode control register 1)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| BDS  | PEN  | TDP  | SBL  | CBL1 | CBL0 | CKS  | MD   |
| ⊙  | ×    | ×    | ×    | ⊙    | ⊙    | ⊙    | 1    |
| SMC2n (UART/SIO serial mode control register 2)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| SCKE   | TXOE | RERC | RXE  | TXE  | RIE  | TCIE | TEIE |
| ⊙  | 1    | ×    | ×    | ⊙    | ×    | ⊙    | ⊙    |
| SSRn (UART/SIO serial status and data register)                    |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| -  | -    | PER  | OVE  | FER  | RDRF | TCPL | TDRE |
| ×  | ×    | ×    | ×    | ×    | ×    | ⊙    | ⊙    |
| TDRn (UART/SIO serial output data register)                        |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| TD7  | TD6  | TD5  | TD4  | TD3  | TD2  | TD1  | TD0  |
| ⊙  | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    | ⊙    |
| RDRn (UART/SIO serial input data register)                         |      |      |      |      |      |      |      |
| bit7   | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| RD7  | RD6  | RD5  | RD4  | RD3  | RD2  | RD1  | RD0  |
| ×  | ×    | ×    | ×    | ×    | ×    | ×    | ×    |
| ⊙ : Used bit<br>× : Unused bit<br>1 : Set to "1"<br>0 : Set to "0" |      |      |      |      |      |      |      |

The following two procedures can be used to initiate the transmission process:

- Set the transmission operation enable bit (TXE) to "1", then write transmit data to the TDRn register to start transmission.
- Write transmit data to the TDRn register, then set the TXE bit to "1" to start transmission.

Transmit data is written to the TDRn register after it is checked that the transmit data register empty flag bit (TDRE) is set to "1".

When the transmit data is written to the TDRn register, the TDRE bit is cleared to "0".

When serial transmission is started after transmit data is transferred from the TDRn register to the transmission shift register, the TDRE bit is set to "1".

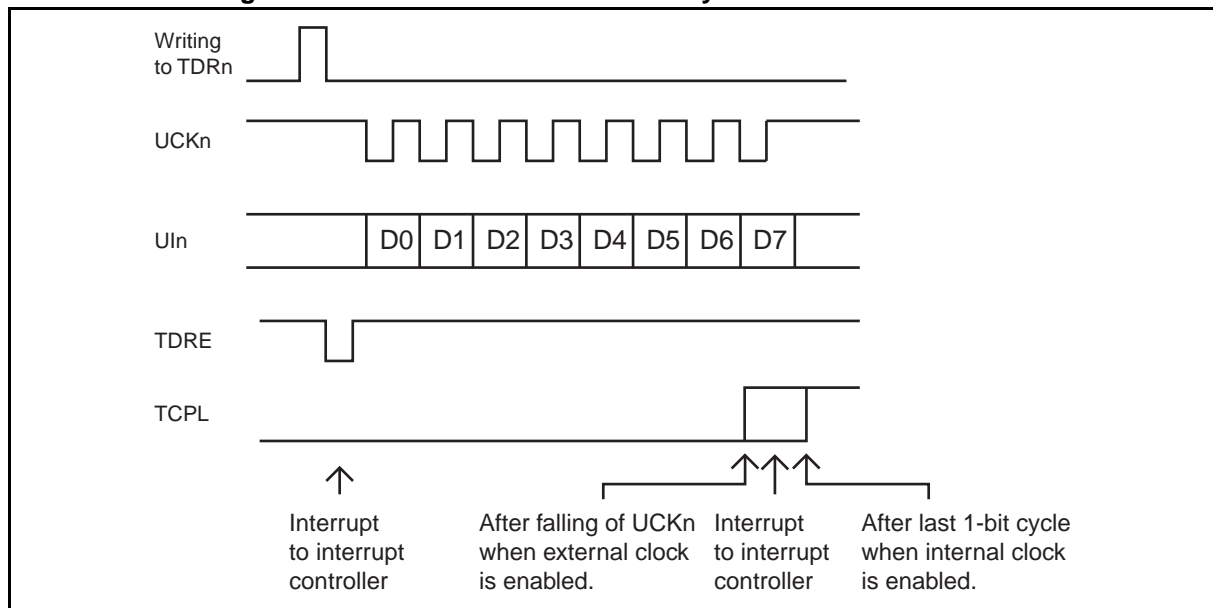


When the use of the external clock signal has been set, serial data transmission starts at the fall of the first serial clock signal after the transmission process is started.

A transmission completion interrupt occurs the moment the TDRE bit is set to "1" when the transmit interrupt enable bit (TIE) contains "1". At this time, the next piece of transmit data can be written to the TDRn register. Serial transmission can be continued with the TXE bit set to "1".

To use a transmission completion interrupt to detect the completion of serial transmission, enable transmission completion interrupt output this way: TEIE = 0, TCIE = 1. Upon completion of transmission, the transmission completion flag bit (SSRn:TCPL) is set to "1" and a transmission completion interrupt occurs.

**Figure 19.6-15 8-bit Transmission in Synchronous Clock Mode**



● Concurrent transmission and reception

<When external clock is enabled>

Transmission and reception can be performed independently of each other. Transmission and reception can therefore be performed at the same time or even when their phases are shifted from each other and overlapping.

<When internal clock is enabled>

As the transmitting side generates a serial clock, reception is influenced.

If transmission stops during reception, the receiving side is suspended. It resumes reception when the transmitting side is restarted.

See "19.4 Pins" for operation with serial clock output and operation with serial clock input.

## 19.7 Registers

---

**This section describes the registers of the UART/SIO.**

---

**Table 19.7-1 List of UART/SIO Registers**

| Register abbreviation | Register name                                  | Reference |
|-----------------------|--|-----------|
| SMC1n                 | UART/SIO serial mode control register 1 ch. n  | 19.7.1    |
| SMC2n                 | UART/SIO serial mode control register 2 ch. n  | 19.7.2    |
| SSRn                  | UART/SIO serial status and data register ch. n | 19.7.3    |
| RDRn                  | UART/SIO serial input data register ch. n      | 19.7.4    |
| TDRn                  | UART/SIO serial output data register ch. n     | 19.7.5    |

## MB95850K/860K/870K Series

### 19.7.1 UART/SIO Serial Mode Control Register 1 ch. n (SMC1n)

The UART/SIO serial mode control register 1 ch. n (SMC1n) controls the UART/SIO operation mode. The register is used to set the serial data direction (endian), parity and its polarity, stop bit length, operation mode (clock synchronous mode / clock asynchronous mode), data length, and serial clock.

#### Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3    | 2    | 1   | 0   |
|---------------|-----|-----|-----|-----|------|------|-----|-----|
| Field         | BDS | PEN | TDP | SBL | CBL1 | CBL0 | CKS | MD  |
| Attribute     | R/W | R/W | R/W | R/W | R/W  | R/W  | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0    | 0    | 0   | 0   |

#### Register Functions

[bit7] BDS: Serial data direction control bit

This bit controls the serial data direction (endian).

| bit7               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Transmission or reception starts from the LSB in the UART/SIO serial input/output data register ch. n (RDRn/TDRn). |
| <b>Writing "1"</b> | Transmission or reception starts from the MSB in the UART/SIO serial input/output data register ch. n (RDRn/TDRn). |

[bit6] PEN: Parity control bit

This bit enables or disables the parity in clock asynchronous mode (UART).

| bit6               | Details              |
|--------------------|----------------------|
| <b>Writing "0"</b> | Disables the parity. |
| <b>Writing "1"</b> | Enables the parity.  |

[bit5] TDP: Parity polarity control bit

This bit controls the even/odd parity.

| bit5               | Details                  |
|--------------------|--------------------------|
| <b>Writing "0"</b> | Selects the even parity. |
| <b>Writing "1"</b> | Selects the odd parity.  |

[bit4] SBL: Stop bit length control bit

This bit controls the stop bit length in clock asynchronous mode (UART).

| bit4               | Details |
|--------------------|---------|
| <b>Writing "0"</b> | 1 bit   |
| <b>Writing "1"</b> | 2 bits  |

Note: The setting of this bit is only valid for transmission operation in clock asynchronous mode (UART). In a receive operation, regardless of the setting of this bit, the UART/SIO completes the receive operation when detecting a stop bit (one bit), and sets the receive data register full flag bit (SSRn:RDRF) to "1".

[bit3:2] CBL[1:0]: Character bit length control bits

These bits select the character bit length.

The setting of these bits is valid in both clock asynchronous mode (UART) and clock synchronous mode (SIO).

| bit3:2              | Details |
|---------------------|---------|
| <b>Writing "00"</b> | 5 bits  |
| <b>Writing "01"</b> | 6 bits  |
| <b>Writing "10"</b> | 7 bits  |
| <b>Writing "11"</b> | 8 bits  |

[bit1] CKS: Clock select bit

This bit selects a serial clock from the external clock or the UART/SIO dedicated baud rate generator.

| bit1               | Details                                |
|--------------------|--|
| <b>Writing "0"</b> | UART/SIO dedicated baud rate generator |
| <b>Writing "1"</b> | External clock                         |

Note: Setting this bit to "1" forcibly disables the output of the UCKn pin. The external clock cannot be used in clock asynchronous mode (UART).

[bit0] MD: Operation mode select bit

This bit selects an operation mode from the clock asynchronous mode (UART) or the clock synchronous mode (SIO).

| bit0               | Details                        |
|--------------------|--------------------------------|
| <b>Writing "0"</b> | Clock asynchronous mode (UART) |
| <b>Writing "1"</b> | Clock synchronous mode (SIO)   |

---

Note:

During data transmission or reception, do not modify the settings of the UART/SIO serial mode control register 1 ch. n (SMC1n).

---

## MB95850K/860K/870K Series

### 19.7.2 UART/SIO Serial Mode Control Register 2 ch. n (SMC2n)

The UART/SIO serial mode control register 2 ch. n (SMC2n) controls the UART/SIO operation mode. The register enables or disables serial clock output, serial data output, transmission/reception, and interrupts, and clears the receive error flag.

#### Register Configuration

| bit           | 7    | 6    | 5    | 4   | 3   | 2   | 1    | 0    |
|---------------|------|------|------|-----|-----|-----|------|------|
| Field         | SCKE | TXOE | RERC | RXE | TXE | RIE | TCIE | TEIE |
| Attribute     | R/W  | R/W  | R/W  | R/W | R/W | R/W | R/W  | R/W  |
| Initial value | 0    | 0    | 1    | 0   | 0   | 0   | 0    | 0    |

#### Register Functions

[bit7] SCKE: Serial clock output enable bit

This bit controls the input/output of the serial clock pin (UCKn) in clock synchronous mode (SIO).

| bit7               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the serial clock, and makes the UCKn pin function as a general purpose I/O port. |
| <b>Writing "1"</b> | Enables the serial clock, and makes the UCKn pin function as a serial clock output pin.   |

Note: With the clock select bit (SMC1n:CKS) already set to "1", no internal clock signal is output even when this bit set to "1".

In clock asynchronous mode (UART) (SMC1n:MD = 0), when this bit is set to "0", the output from the UCKn bit will always be "H".

[bit6] TXOE: Serial data output enable bit

This bit controls the output of the serial data pin (UOn).

| bit6               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables serial data output, and makes the UOn pin function as a general purpose I/O port. |
| <b>Writing "1"</b> | Enables serial data output, and makes the UOn pin function as a serial data output pin.    |

[bit5] RERC: Receive error flag clear bit

This bit clears the receive error flags.

The read value of this bit is always "1".

| bit5               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Clears the receive error flags (PER, OVE and FER) in the SSRn register. |
| <b>Writing "1"</b> | Has no effect on operation.   |

[bit4] RXE: Receive operation enable bit

This bit enables or disables the reception of serial data.

If this bit is set to "0" during a receive operation, the receive operation is immediately disabled and initialized. The data received up to that point is not transferred to the UART/SIO serial input data register.

| bit4               | Details                                |
|--------------------|--|
| <b>Writing "0"</b> | Disables the reception of serial data. |
| <b>Writing "1"</b> | Enables the reception of serial data.  |

Note: Setting this bit to "0" initializes the receive operation. It has no effect on the error flags (PER, OVE, FER, RDRF) in the SSRn register.

[bit3] TXE: Transmit operation enable bit

This bit enables or disables the transmission of serial data.

If this bit is set to "0" during a transmit operation, the transmit operation is immediately disabled and initialized. The transmission completion flag bit (SSRn:TCPL) is then be set to "1", and the transmit data register empty flag bit (SSRn:TDRE) is also be set to "1".

| bit3               | Details                                   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the transmission of serial data. |
| <b>Writing "1"</b> | Enables the transmission of serial data.  |

[bit2] RIE: Receive interrupt enable bit

This bit enables or disables the receive interrupt.

With this bit set to "1" (receive interrupt enabled), a receive interrupt is generated immediately after either the receive data register full flag bit (SSRn:RDRF) or any of the receive error flag bits (SSRn:PER, OVE, FER) is set to "1".

| bit2               | Details                         |
|--------------------|---------------------------------|
| <b>Writing "0"</b> | Disables the receive interrupt. |
| <b>Writing "1"</b> | Enables the receive interrupt.  |

[bit1] TCIE: Transmission completion interrupt enable bit

This bit enables or disables the transmission completion interrupt.

With this bit set to "1" (transmission completion interrupt enabled), a transmission completion interrupt is generated immediately after the transmission completion flag bit (SSRn:TCPL) is set to "1".

| bit1               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the transmission completion interrupt. |
| <b>Writing "1"</b> | Enables the transmission completion interrupt.  |

[bit0] TEIE: Transmit data register empty interrupt enable bit

This bit enables or disables the transmit data register empty interrupt.

With this bit set to "1" (transmit data register empty interrupt enabled), a transmit data register empty interrupt is generated immediately after the transmit data register empty flag bit (SSRn:TDRE) is set to "1".

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the transmit data register empty interrupt. |
| <b>Writing "1"</b> | Enables the transmit data register empty interrupt.  |

## MB95850K/860K/870K Series

### 19.7.3 UART/SIO Serial Status and Data Register ch. n (SSRn)

The UART/SIO serial status and data register ch. n (SSRn) indicates the transmission/reception status and error status of the UART/SIO.

#### ■ Register Configuration

| bit           | 7 | 6 | 5   | 4   | 3   | 2    | 1    | 0    |
|---------------|---|---|-----|-----|-----|------|------|------|
| Field         | — | — | PER | OVE | FER | RDRF | TCPL | TDRE |
| Attribute     | — | — | R   | R   | R   | R    | R/W  | R    |
| Initial value | 0 | 0 | 0   | 0   | 0   | 0    | 0    | 1    |

#### ■ Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5] PER: Parity error flag bit

This bit detects the parity error in receive data.

This bit is set to "1" when a parity error occurs during a receive operation reception. Writing "0" to the RERC bit in the SMC2n register clears this bit.

When a parity error is detected at the same time as clearing this bit by writing "0" to the RERC bit, setting this bit to "1" is given priority.

| bit5               | Details                                      |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that no parity error has occurred. |
| <b>Reading "1"</b> | Indicates that a parity error has occurred.  |

[bit4] OVE: Overrun error flag bit

This bit detects the overrun error in receive data.

This bit is set to "1" when an overrun error occurs during a receive operation reception. Writing "0" to the RERC bit in the SMC2n register clears this bit.

When an overrun error is detected at the same time as clearing this bit by writing "0" to the RERC bit, setting this bit to "1" is given priority.

| bit4               | Details                                       |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no overrun error has occurred. |
| <b>Reading "1"</b> | Indicates that an overrun error has occurred. |

[bit3] FER: Framing error flag bit

This bit detects the framing error in receive data.

This bit is set to "1" when a framing error occurs during a receive operation reception. Writing "0" to the RERC bit in the SMC2n register clears this bit.

When a framing error is detected at the same time as clearing this bit by writing "0" to the RERC bit, setting this bit to "1" is given priority.

| bit3               | Details                                       |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no framing error has occurred. |
| <b>Reading "1"</b> | Indicates that a framing error has occurred.  |

[bit2] RDRF: Receive data register full flag bit

This bit indicates the state of the UART/SIO serial input data register ch. n (RDRn).

When receive data is loaded to the RDRn register, this bit is set to "1".

When data in the RDRn register is read, this bit is cleared to "0".

| bit2               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that there is no receive data in the RDRn register. |
| <b>Reading "1"</b> | Indicates that there is receive data in the RDRn register.    |

[bit1] TCPL: Transmission completion flag bit

This bit indicates the data transmission state.

When serial transmission is completed, this bit is set to "1". However, when the UART/SIO serial output data register ch. n (TDRn) contains data to be transmitted successively, this bit is not set to "1" even after one time of transmission is completed.

Writing "0" to this bit clears it.

When transmission completion setting this bit to "1" and writing "0" to this bit to clear it occur simultaneously, the former one is given priority.

Writing "1" to this bit has no effect on operation.

| bit1               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that data transmission has not been completed. |
| <b>Reading "1"</b> | Indicates that data transmission has been completed.     |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.                              |

[bit0] TDRE: Transmit data register empty flag bit

This bit indicates the state of the UART/SIO serial output data register ch. n (TDRn).

When transmit data is written to the TDRn register, this bit is set to "0".

When transmit data is loaded to the shift register for the transmission and data transmission starts, this bit is set to "1".

| bit0               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that there is transmit data in the TDRn register.    |
| <b>Reading "1"</b> | Indicates that there is no transmit data in the TDRn register. |



## MB95850K/860K/870K Series

### 19.7.4 UART/SIO Serial Input Data Register ch. n (RDRn)

The UART/SIO serial input data register ch. n (RDRn) is used for inputting (receiving) serial data.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Field         | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| Attribute     | R   | R   | R   | R   | R   | R   | R   | R   |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### ■ Register Functions

This register stores received data. The serial data signals sent to the serial data input pin (UIn) is converted by the shift register and stored in this register.

When received data is set correctly in this register, the receive data register full flag bit (SSRn:RDRF) is set to "1". At this time, an interrupt occurs if receive interrupt requests have been enabled. If an RDRF bit check by the program or using an interruption shows that received data is stored in this register, the reading of the content for this register clears the RDRF bit to "0".

When the character bit length (SMC1n:CBL[1:0]) is set to shorter than eight bits, the excess upper bits (beyond the set bit length) are set to "0".

## 19.7.5 UART/SIO Serial Output Data Register ch. n (TDRn)

The UART/SIO serial output data register ch. n (TDRn) used for outputting (transmitting) serial data.

### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Field         | TD7 | TD6 | TD5 | TD4 | TD3 | TD2 | TD1 | TD0 |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

### ■ Register Functions

This register holds data to be transmitted. The register accepts a write when the transmit data register empty flag bit (TDRE) is "1". An attempt to write to the bit is ignored when the bit contains "0".

When transmit data is written to the UART/SIO serial output data register ch. n (TDRn), the TDRE bit is set to "0". Upon completion of transfer of transmit data to the transmission shift register, the TDRE bit is set to "1", enabling the next transmit data to be written to the TDRn register. At this time, an interrupt occurs if transmit data register empty interrupts have been enabled. Write the next piece of transmit data when transmit data empty occurs or the TDRE bit is set to "1".

When the character bit length (SMC1n:CBL[1:0]) is set to shorter than eight bits, the excess upper bits (beyond the set bit length) are ignored.

#### Note:

The data in this register cannot be updated when the TDRE bit in UART/SIO serial status and data register ch.n (SSRn) is "0".

When this register is updated at writing complete the transmission data and TDRE = 0 (regardless of the setting of the TXE bit in the SMC2n register), the transmission operation is initialized by writing "0" to TXE, the TDRE bit becomes "1", and updating this register is enabled.

Moreover, when "0" is written to the TXE bit without transmission having started (when the transmit data is written to the TDRn register, and the TXE bit has not been set to "1" yet), the TCPL bit is not set to "1". In the case of modifying the transmit data, make the TDRE bit become "1" once by writing "0" to the TXE bit before modifying the transmit data.

# **CHAPTER 20**

---

# ***UART/SIO DEDICATED BAUD RATE GENERATOR***

**This chapter describes the functions and operations of the dedicated baud rate generator for the UART/SIO.**

- 20.1 Overview
- 20.2 Channel
- 20.3 Operations
- 20.4 Registers

## 20.1 Overview

The UART/SIO dedicated baud rate generator generates the baud rate for the UART/SIO.

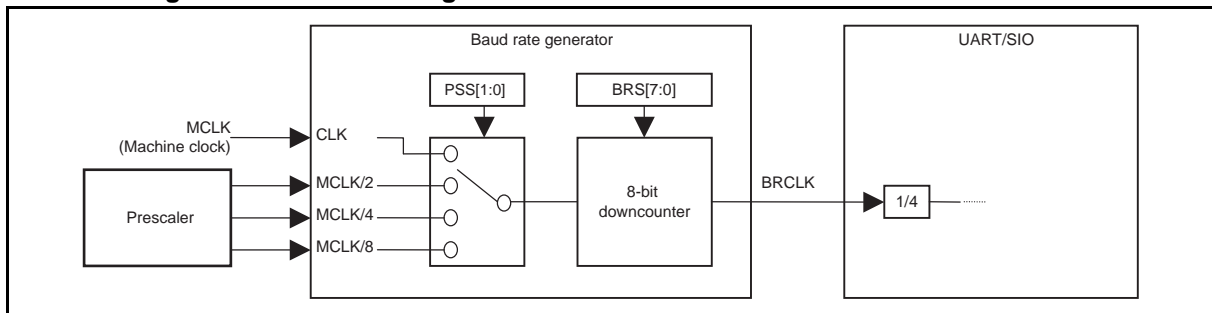
The generator consists of the UART/SIO dedicated baud rate generator prescaler select register ch. n (PSSRn) and UART/SIO dedicated baud rate generator baud rate setting register ch. n (BRSRn).

The number of pins and that of channels of the UART/SIO dedicated baud rate generator vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

### ■ Block Diagram of UART/SIO Dedicated Baud Rate Generator

Figure 20.1-1 Block Diagram of UART/SIO Dedicated Baud Rate Generator



### ■ Input Clock

The UART/SIO dedicated baud rate generator uses the output clock from the prescaler or the machine clock as its input clock.

### ■ Output Clock

The UART/SIO dedicated baud rate generator supplies its clock to the UART/SIO.

## 20.2 Channel

---

This section describes the channel of the UART/SIO dedicated baud rate generator.

---

### ■ Channel of UART/SIO Dedicated Baud Rate Generator

Table 20.2-1 shows the registers of the UART/SIO dedicated baud rate generator.

**Table 20.2-1 Registers of Dedicated Baud Rate Generator**

| Register abbreviation | Corresponding register (Name in this manual)                            |
|-----------------------|---|
| PSSRn                 | UART/SIO dedicated baud rate generator prescaler select register ch. n  |
| BRSRn                 | UART/SIO dedicated baud rate generator baud rate setting register ch. n |

## 20.3 Operations

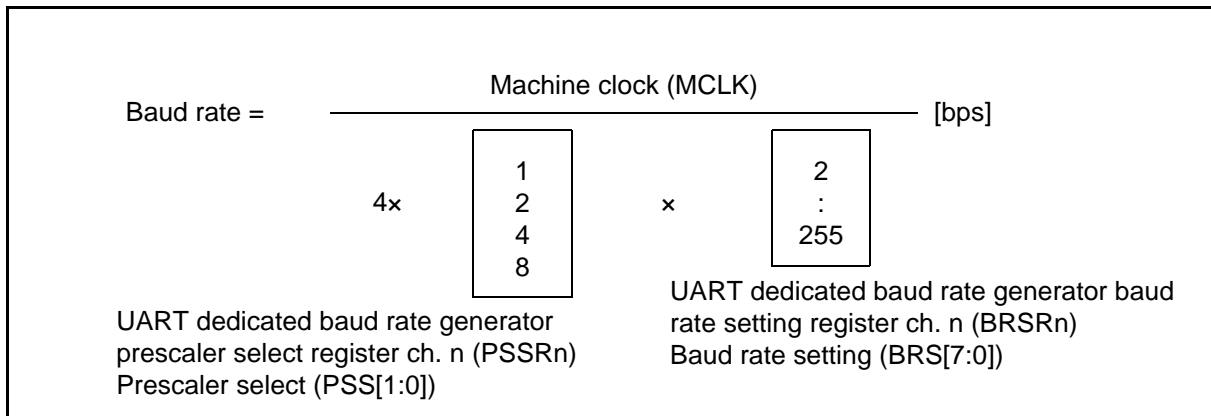
The UART/SIO dedicated baud rate generator serves as the baud rate generator in clock asynchronous mode (UART).

### ■ Baud Rate Setting

The CKS bit in the SMC1n register of the UART/SIO is used to select the serial clock. This selects the UART/SIO dedicated baud rate generator.

In asynchronous clock mode, the shift clock that is selected by the CKS bit and divided by four is used and transfers can be performed within the range from -3% to +3%. The baud rate calculation formula for the UART/SIO dedicated baud rate generator is shown below.

**Figure 20.3-1 Baud Rate Calculation Formula when UART/SIO Dedicated Baud Rate Generator Is Used**



**Table 20.3-1 Sample Asynchronous Transfer Rates by Baud Rate Generator (Machine Clock = 10 MHz, 16 MHz, 16.25 MHz)**

| UART/SIO dedicated baud rate generator setting |                                     | UART internal division | Total division ratio (PSS × BRS × 4) | Baud rate (10 MHz / Total division ratio) | Baud rate (16 MHz / Total division ratio) | Baud rate (16.25 MHz / Total division ratio) |
|--|-------------------------------------|------------------------|--------------------------------------|---|---|--|
| Prescaler select PSS[1:0]                      | Baud rate counter setting BRS [7:0] |                        |                                      |   |   |  |
| 1 (Setting value: 0, 0)                        | 20                                  | 4                      | 80                                   | 125000                                    | 200000                                    | 203125                                       |
| 1 (Setting value: 0, 0)                        | 22                                  | 4                      | 88                                   | 113636                                    | 181818                                    | 184659                                       |
| 1 (Setting value: 0, 0)                        | 44                                  | 4                      | 176                                  | 56818                                     | 90909                                     | 92330  |
| 1 (Setting value: 0, 0)                        | 87                                  | 4                      | 348                                  | 28736                                     | 45977                                     | 46695  |
| 1 (Setting value: 0, 0)                        | 130                                 | 4                      | 520                                  | 19231                                     | 30769                                     | 31250  |
| 2 (Setting value: 0, 1)                        | 130                                 | 4                      | 1040                                 | 9615                                      | 15385                                     | 15625  |
| 4 (Setting value: 1, 0)                        | 130                                 | 4                      | 2080                                 | 4808                                      | 7692                                      | 7813   |
| 8 (Setting value: 1, 1)                        | 130                                 | 4                      | 4160                                 | 2404                                      | 3846                                      | 3906   |

The baud rate can be set in UART mode within the following range.

**Table 20.3-2 Baud Rate Setting Range in Clock Asynchronous Mode (UART)**

| PSS[1:0]     | BRS[7:0]               |
|--------------|------------------------|
| 0b00 to 0b11 | 0x02 (2) to 0xFF (255) |

## 20.4 Registers

---

This section describes the registers of the UART/SIO dedicated baud rate generator.

---

**Table 20.4-1 List of UART/SIO Baud Rate Generator Registers**

| Register abbreviation | Register name   | Reference |
|-----------------------|---|-----------|
| PSSRn                 | UART/SIO dedicated baud rate generator prescaler select register ch. n  | 20.4.1    |
| BRSRn                 | UART/SIO dedicated baud rate generator baud rate setting register ch. n | 20.4.2    |

## 20.4.1 UART/SIO Dedicated Baud Rate Generator Prescaler Select Register ch. n (PSSRn)

The UART/SIO dedicated baud rate generator prescaler select register ch. n (PSSRn) controls the output of the baud rate clock and the prescaler.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3 | 2    | 1    | 0    |
|---------------|---|---|---|---|---|------|------|------|
| Field         | — | — | — | — | — | BRGE | PSS1 | PSS0 |
| Attribute     | — | — | — | — | — | R/W  | R/W  | R/W  |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0    | 0    | 0    |

### ■ Register Functions

[bit7:3] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit2] BRGE: Baud rate clock output enable bit

This bit enables or disables outputting the baud rate clock "BRCLK".

When "1" is written to this bit, the value of the BRS[7:0] bits in the BRSRn register is loaded to the 8-bit downcounter and BRCLK to be supplied to the UART/SIO is output.

Writing "0" to this bit stops the output of BRCLK.

| bit2               | Details                                  |
|--------------------|--|
| <b>Writing "0"</b> | Disables outputting the baud rate clock. |
| <b>Writing "1"</b> | Enables outputting the baud rate clock.  |

[bit1:0] PSS[1:0]: Prescaler select bits

These bits select a prescaler.

| bit1:0              | Details |
|---------------------|---------|
| <b>Writing "00"</b> | 1       |
| <b>Writing "01"</b> | 1/2     |
| <b>Writing "10"</b> | 1/4     |
| <b>Writing "11"</b> | 1/8     |



## 20.4.2 UART/SIO Dedicated Baud Rate Generator Baud Rate Setting Register ch. n (BRSRn)

The UART/SIO dedicated baud rate generator baud rate setting register ch.n (BRSRn) controls the baud rate settings.

### ■ Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | BRS7 | BRS6 | BRS5 | BRS4 | BRS3 | BRS2 | BRS1 | BRS0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

### ■ Register Functions

This register sets the cycle of the 8-bit downcounter and can be used to set any baud rate clock (BRCLK). Stop the UART/SIO operation before writing a value to this register.

In clock asynchronous mode (UART), do not set BRS[7:0] to "0x00" or "0x01".



# CHAPTER 21

---

## *I<sup>2</sup>C BUS INTERFACE*

**This chapter describes functions and operations of the I<sup>2</sup>C bus interface.**

- 21.1 Overview
- 21.2 Configuration
- 21.3 Channel
- 21.4 Pins
- 21.5 Interrupts
- 21.6 Operations and Setting Procedure Example
- 21.7 Registers
- 21.8 Notes on Using I<sup>2</sup>C Bus Interface

## 21.1 Overview

The I<sup>2</sup>C bus interface provides the functions of transmission and reception in master and slave modes, detection of arbitration lost, detection of slave address and general call address, generation and detection of start and stop conditions, bus error detection, and MCU standby wakeup.

### ■ I<sup>2</sup>C Bus Interface Functions

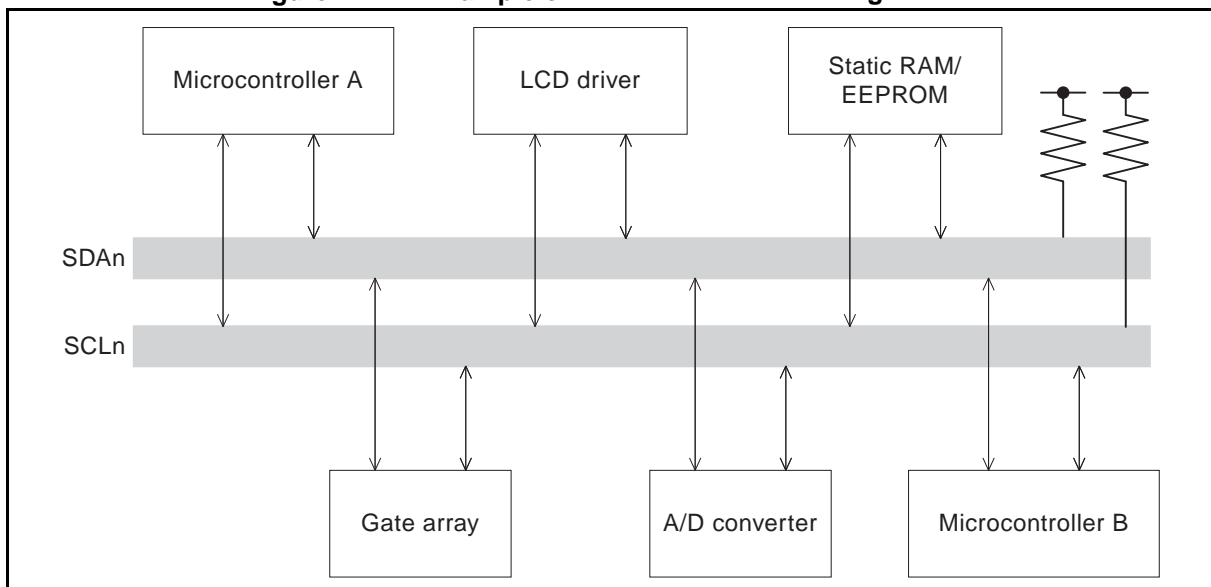
The I<sup>2</sup>C bus interface is a two-wire, bi-directional bus consisting of a serial data line (SDAn) and serial clock line (SCLn). The devices connected to the bus via these two wires can exchange data, and each device can operate as a sender or receiver in accordance with their respective functions based on the unique address assigned to each device. Furthermore, the interface establishes a master/slave relationship between devices.

The I<sup>2</sup>C bus interface can connect multiple devices provided the bus capacitance does not exceed an upper limit of 400 pF. The I<sup>2</sup>C bus interface is a true multi-master bus with collision detection and a communication control protocol that prevent loss of data even if more than one master attempts to start a data transfer at the same time.

The communication control protocol ensures that only one master is able to take control of the bus at a time, even if multiple masters attempt to take control of the bus simultaneously, without messages being lost or data being altered. Multi-master means that more than one master can attempt to take control of the bus at the same time without causing messages to be lost.

The I<sup>2</sup>C bus interface includes a function to wake up the MCU from standby mode.

Figure 21.1-1 Example of I<sup>2</sup>C Bus Interface Configuration



## MB95850K/860K/870K Series

### 21.2 Configuration

---

The I<sup>2</sup>C bus interface consists of the following blocks:

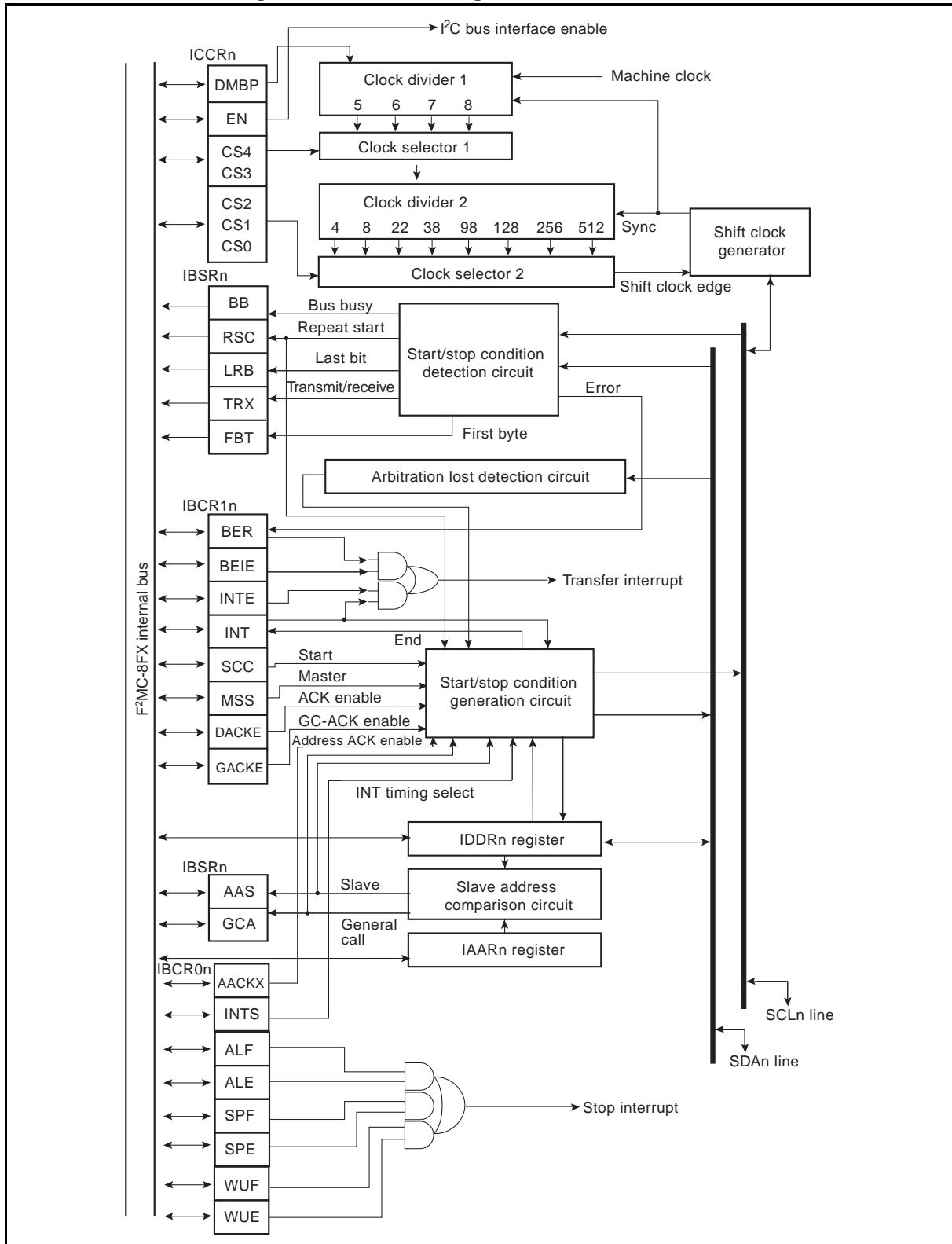
- Clock selector
  - Clock divider
  - Shift clock generator
  - Start/stop condition generation circuit
  - Start/stop condition detection circuit
  - Arbitration lost detection circuit
  - Slave address comparison circuit
  - IBSRn register
  - IBCR0n register
  - IBCR1n register
  - ICCRn register
  - IAARn register
  - IDDRn register
- 

The number of pins and that of channels of the I<sup>2</sup>C bus interface vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

■ Block Diagram of I<sup>2</sup>C Bus Interface

Figure 21.2-1 Block Diagram of I<sup>2</sup>C Bus Interface



**MB95850K/860K/870K Series**

- Clock selector, clock divider, and shift clock generator

This circuit uses the machine clock to generate the shift clock for the I<sup>2</sup>C bus.

- Start/stop condition generation circuit

When a start condition is transmitted with the bus idle (SCL<sub>n</sub> and SDA<sub>n</sub> at the "H" level), a master starts communications. When SCL<sub>n</sub> = "H", a start condition is generated by changing the SDA<sub>n</sub> line from "H" to "L". The master can terminate its communication by generating a stop condition. When SCL<sub>n</sub> = "H", a stop condition is generated by changing the SDA<sub>n</sub> line from "L" to "H".

- Start/stop condition detection circuit

This circuit detects a start/stop condition for data transfer.

- Arbitration lost detection circuit

This interface circuit supports multi-master systems. If two or more masters attempt to transmit at the same time, the arbitration lost condition (if logic level "1" is sent when the SDA<sub>n</sub> line goes to the "L" level) occurs. When the arbitration lost is detected, IBCR0<sub>n</sub>:ALF is set to "1" and the master changes to a slave automatically.

- Slave address comparison circuit

The slave address comparison circuit receives the slave address after the start condition to compare it with its own slave address. The address is seven-bit data followed by a data direction (R/W) bit in the eighth bit position. If the received address matches the own slave address, the comparison circuit transmits an acknowledgment.

- IBSR<sub>n</sub> register

The IBSR<sub>n</sub> register shows the status of the I<sup>2</sup>C bus interface.

- IBCR0<sub>n</sub> register and IBCR1<sub>n</sub> register

The IBCR0<sub>n</sub> register and the IBCR1<sub>n</sub> register are used to select the operating mode and to enable or disable interrupts, acknowledgment, general call acknowledgment, and the function to wake up the MCU from standby mode.

- ICCR<sub>n</sub> register

The ICCR<sub>n</sub> register is used to enable I<sup>2</sup>C bus interface operations and select the shift clock frequency.

- IAAR<sub>n</sub> register

The IAAR<sub>n</sub> register is used to set the slave address.

- IDDR<sub>n</sub> register

The IDDR<sub>n</sub> register holds the transmit or receive shift data or address. When transmitted, the data or address written to this register is transferred from the MSB first to the bus.

**■ Input Clock**

The I<sup>2</sup>C bus interface uses the machine clock as the input clock (shift clock).

## 21.3 Channel

---

**This section describes the channel of the I<sup>2</sup>C bus interface.**

---

### ■ Channel of I<sup>2</sup>C Bus Interface

Table 21.3-1 and Table 21.3-2 show the pins and registers on a channel of the I<sup>2</sup>C bus interface respectively.

**Table 21.3-1 Pins of I<sup>2</sup>C Bus Interface**

| Pin name | Pin function                       |
|----------|------------------------------------|
| SDAn     | I <sup>2</sup> C bus interface I/O |
| SCLn     |                                    |

**Table 21.3-2 Registers of I<sup>2</sup>C Bus Interface**

| Register abbreviation | Corresponding register (Name in this manual)  |
|-----------------------|---|
| IBCR0n                | I <sup>2</sup> C bus control register 0 ch. n |
| IBCR1n                | I <sup>2</sup> C bus control register 1 ch. n |
| IBSRn                 | I <sup>2</sup> C bus status register ch. n    |
| IDDRn                 | I <sup>2</sup> C data register ch. n          |
| IAARn                 | I <sup>2</sup> C address register ch. n       |
| ICCRn                 | I <sup>2</sup> C clock control register ch. n |



## MB95850K/860K/870K Series

### 21.4 Pins

---

This section describes the pins of the I<sup>2</sup>C bus interface and gives their block diagram.

---

#### ■ Pins of I<sup>2</sup>C Bus Interface

The pins of the I<sup>2</sup>C bus interface are SDAn and SCLn.

- SDAn pin

The SDAn pin is the data I/O pin of the I<sup>2</sup>C bus interface.

When the I<sup>2</sup>C bus interface is enabled (ICCRn:EN = 1), the SDAn pin is automatically set as a data I/O pin to function as the SDAn pin.

- SCLn pin

The SCLn pin is the serial clock I/O pin of the I<sup>2</sup>C bus interface.

When the I<sup>2</sup>C bus interface is enabled (ICCRn:EN = 1), the SCLn pin is automatically set as a shift clock I/O pin to function as the SCLn pin.

## 21.5 Interrupts

The I<sup>2</sup>C bus interface has a transfer interrupt and a stop interrupt which are triggered by the following events.

- **Transfer interrupt**  
A transfer interrupt occurs either upon completion of data transfer or when a bus error occurs.
- **Stop interrupt**  
A stop interrupt occurs upon detection of a stop condition or arbitration lost or upon access to the I<sup>2</sup>C bus interface in stop/watch mode.

### ■ Transfer Interrupt

Table 21.5-1 shows the transfer interrupt control bits and I<sup>2</sup>C bus interface interrupt sources.

**Table 21.5-1 Transfer Interrupt Control Bits and I<sup>2</sup>C Bus Interface Interrupt Sources**

| Item                         | End of transfer        | Bus error          |
|------------------------------|------------------------|--------------------|
| Interrupt request flag bit   | IBCR1n:INT =1          | IBCR1n:BER =1      |
| Interrupt request enable bit | IBCR1n:INTE =1         | IBCR1n:BEIE =1     |
| Interrupt source             | Data transfer complete | Bus error occurred |

- **Interrupt upon completion of transfer**  
An interrupt request is output to the CPU upon completion of data transfer if the transfer completion interrupt request enable bit has been set to enable (IBCR1n:INTE = 1). In the interrupt service routine, write "0" to the transfer completion interrupt request flag bit (IBCR1n:INT) to clear the interrupt request. When data transfer is completed, the IBCR1n:INT bit is set to "1" regardless of the value of the IBCR1n:INTE bit.
- **Interrupt in response to a bus error**  
When the following conditions are met, a bus error is deemed to have occurred, and the I<sup>2</sup>C bus interface will be stopped.
  - When a stop condition is detected in master mode.
  - When a start or stop condition is detected during transmission or reception of the first byte.
  - When a start or stop condition is detected during transmission or reception of data (excluding the start, first data, and stop bits).

In these cases, an interrupt request is output to the CPU if the bus error interrupt request enable bit has been set to enable (IBCR1n:BEIE = 1). In the interrupt service routine, write "0" to the bus error interrupt request flag bit (IBCR1n:BER) to clear the interrupt request. When a bus error occurs, the IBCR1n:BER bit is set to "1" regardless of the value of the IBCR1n:BEIE bit.

## MB95850K/860K/870K Series

### ■ Stop Interrupt

Table 21.5-2 shows the stop interrupt control bits and I<sup>2</sup>C interrupt sources (trigger events).

**Table 21.5-2 Stop Interrupt Control Bits and I<sup>2</sup>C Interrupt Sources**

| Item                         | Detection of stop condition | Detection of arbitration lost | MCU wakeup from stop/watch mode |
|------------------------------|-----------------------------|-------------------------------|---------------------------------|
| Interrupt request flag bit   | IBCR0n:SPF =1               | IBCR0n:ALF =1                 | IBCR0n:WUF =1                   |
| Interrupt request enable bit | IBCR0n:SPE =1               | IBCR0n:ALE =1                 | IBCR0n:WUE =1                   |
| Interrupt source             | Stop condition detected     | Arbitration lost detected     | Start condition detected        |

- Interrupt upon detection of a stop condition  
A stop condition is considered to be valid if all of the following conditions are satisfied when the stop condition is detected.
  - The bus is busy (state which the start condition is detected).
  - IBCR1n:MSS = 0
  - After transfer of one byte of data completes, including the acknowledgment.

In this case, an interrupt request is output to the CPU if the stop condition detection interrupt request enable bit has been set to enable (IBCR0n:SPE =1). In the interrupt service routine, write "0" to the IBCR0n:SPF bit to clear the interrupt request.

The IBCR0n:SPF bit is set to "1" when a valid stop condition occurs regardless of the value of the IBCR0n:SPE bit.

- Interrupt upon detection of arbitration lost  
When arbitration lost is detected, an interrupt request is output to the CPU if the arbitration lost detection interrupt request enable bit has been set to enable (IBCR0n:ALE = 1). Either write "0" to the arbitration lost interrupt request flag bit (IBCR0n:ALF) while the bus is idle or write "0" to the IBCR1n:INT bit from the interrupt service routine while the bus is busy to clear the interrupt request.  
When arbitration lost occurs, the IBCR0n:ALF bit is set to "1" regardless of the value for the IBCR0n:ALE bit.
- Interrupt for MCU wakeup from stop mode or watch mode  
When a start condition is detected, an interrupt request is output to the CPU if the function to wake up the MCU from stop or watch mode has been enabled (IBCR0n:WUE = 1).  
In the interrupt service routine, write "0" to the MCU standby mode wakeup interrupt request flag bit (IBCR0n:WUF) to clear the interrupt request.

## 21.6 Operations and Setting Procedure Example

---

This section describes the operations of the I<sup>2</sup>C bus interface.

---

### ■ Operations of I<sup>2</sup>C Bus Interface

- I<sup>2</sup>C bus interface

The I<sup>2</sup>C bus interface is an 8-bit serial interface synchronized with a shift clock.

- MCU standby mode wakeup function

The wakeup function wakes up the MCU upon detection of a start condition, from low power consumption mode such as stop or watch mode.

### ■ Setting Procedure Example

Below is an example of procedure for setting the I<sup>2</sup>C bus interface.

- Initial settings

1. Set the port for input. (DDR)
2. Set the interrupt level. (ILR\*)
3. Set the slave address. (IAAR<sub>n</sub>)
4. Select the clock and enable I<sup>2</sup>C operation. (ICCR<sub>n</sub>)
5. Enable bus error interrupt requests. (IBCR<sub>1n</sub>:BEIE = 1)

\*: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

- Interrupt processing

1. Execute any process.
2. Clear the bus error interrupt request flag. (IBCR<sub>1n</sub>:BER = 0)

## 21.6.1 I<sup>2</sup>C Bus Interface

The I<sup>2</sup>C bus interface is an eight-bit serial interface synchronized with the shift clock.

### ■ I<sup>2</sup>C System

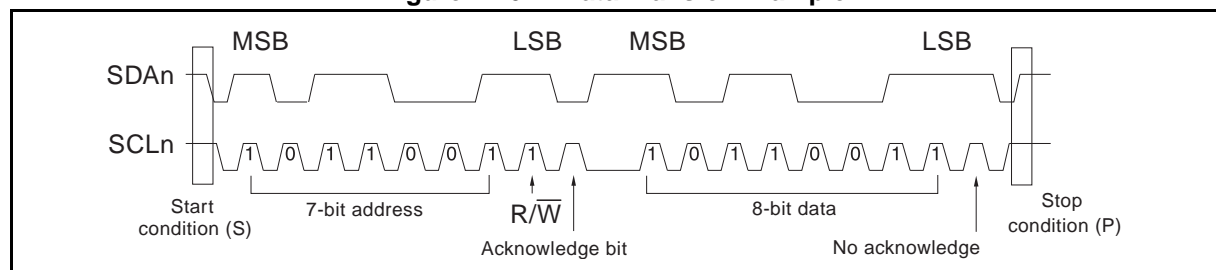
The I<sup>2</sup>C bus system uses the serial data line (SDAn) and serial clock line (SCLn) for data transfers. All the devices connected to the bus require open drain or open collector outputs which must be connected with a pull-up resistor.

Each of the devices connected to the bus has a unique address which can be set up using software. The devices always operate in a simple master/slave relationship, where the master functions as the master transmitter or master receiver. The I<sup>2</sup>C bus interface is a true multi-master bus with a collision detection function and arbitration function to prevent data from being lost if more than one master attempts to start data transfer at the same time.

### ■ I<sup>2</sup>C Protocol

Figure 21.6-1 shows the format required for data transfer.

Figure 21.6-1 Data Transfer Example



The slave address is transmitted after a start condition (S) is generated. This address is seven bits long followed by the data direction bit ( $R/\overline{W}$ ) in the eighth bit position. Data is transmitted after the address. The data is eight bits followed by an acknowledgment.

Data can be transmitted continuously to the same slave address in consecutive units of eight bits plus acknowledgment.

Data transfer is always ended in the master stop condition (P). However, the repeated start condition (S) can be used to transmit the address which indicates a different slave without generating a stop condition.

## ■ Start Conditions

While the bus is idle (SCL<sub>n</sub> and SDA<sub>n</sub> are both at the logical "H" level), the master generates a start condition to start transmission. As shown in Figure 21.6-1, a start condition is triggered when the SDA<sub>n</sub> line is changed from "H" to "L" while SCL<sub>n</sub> = "H". This starts a new data transfer and commences master/slave operation.

A start condition can be generated in either of the following two ways.

- By writing "1" to the IBCR1<sub>n</sub>:MSS bit while the I<sup>2</sup>C bus is not in use (IBCR1<sub>n</sub>:MSS = 0, IBSR<sub>n</sub>:BB = 0, IBCR1<sub>n</sub>:INT = 0, and IBCR0<sub>n</sub>:ALF = 0). (Next, IBSR<sub>n</sub>:BB is set to "1" to indicate that the bus is busy.)
- By writing "1" to the IBCR1<sub>n</sub>:SCC bit during an interrupt while in master mode (IBCR1<sub>n</sub>:MSS = 1, IBSR<sub>n</sub>:BB = 1, IBCR1<sub>n</sub>:INT = 1, and IBCR0<sub>n</sub>:ALF = 0). (This generates a repeated START condition.)

Writing "1" to the IBCR1<sub>n</sub>:MSS or IBCR1<sub>n</sub>:SCC bit is ignored in any circumstances other than those mentioned above. If another system is using the bus when "1" is written to the IBCR1<sub>n</sub>:MSS bit, the IBCR0<sub>n</sub>:ALF bit is set to "1".

## ■ Addressing

### ● Slave addressing in master mode

In master mode, IBSR<sub>n</sub>:BB and IBSR<sub>n</sub>:TRX are set to "1" after the start condition is generated, and the slave address in the IDDR<sub>n</sub> register is output to the bus starting with the MSB. The address data consists of eight bits: the 7-bit slave address and the data transfer direction  $R/\overline{W}$  bit (bit0 in the IDDR<sub>n</sub> register).

The acknowledgment from the slave is received after the address data is sent. SDA<sub>n</sub> goes to "L" in the ninth clock cycle and the acknowledge bit from the receiving device is received (See Figure 21.6-1). In this case, the  $R/\overline{W}$  bit (IDDR<sub>n</sub>:bit0) is inverted logically and stored in the IBSR<sub>n</sub>:TRX bit as "1" if the SDA<sub>n</sub> level is "L".

### ● Addressing in slave mode

In slave mode, after the start condition is detected, IBSR<sub>n</sub>:BB is set to "1" and IBSR<sub>n</sub>:TRX is set to "0", and the data received from the master is stored in the IDDR<sub>n</sub> register. After the address data is received, the IDDR<sub>n</sub> and IAAR<sub>n</sub> registers are compared. If the addresses match, IBSR<sub>n</sub>:AAS is set to "1" and an acknowledgment is sent to the master. Afterward, bit0 in the receive data (bit0 in the IDDR<sub>n</sub> register) is saved in the IBSR<sub>n</sub>:TRX bit.

## ■ Data Transfer

If the MCU is addressed as a slave, data can be sent or received byte by byte with the direction determined by the  $R/\overline{W}$  bit sent by the master.

Each byte to be output on the SDA<sub>n</sub> line is fixed at eight bits. As shown in Figure 21.6-1, the receiver sends an acknowledgment to the sender by forcing the SDA<sub>n</sub> line to the stable "L" level while the acknowledge clock pulse is "H". Data is transferred at one clock pulse per bit with MSB at the head. Sending and receiving an acknowledgment is required after each byte is transferred. Therefore, nine clock pulses are required to transfer one complete data byte.

**■ Acknowledgment**

An acknowledgment is sent by the receiver in the ninth clock cycle for data byte transfer by the sender based on the following conditions.

An address acknowledgment is generated in the following cases.

- The received address matches the address set in IAARn, and the address acknowledgment is output automatically (IBCR0n:AACKX = 0).
- A general call address (0x00) is received and the general call address acknowledgment output is enabled (IBCR1n:GACKE = 1).

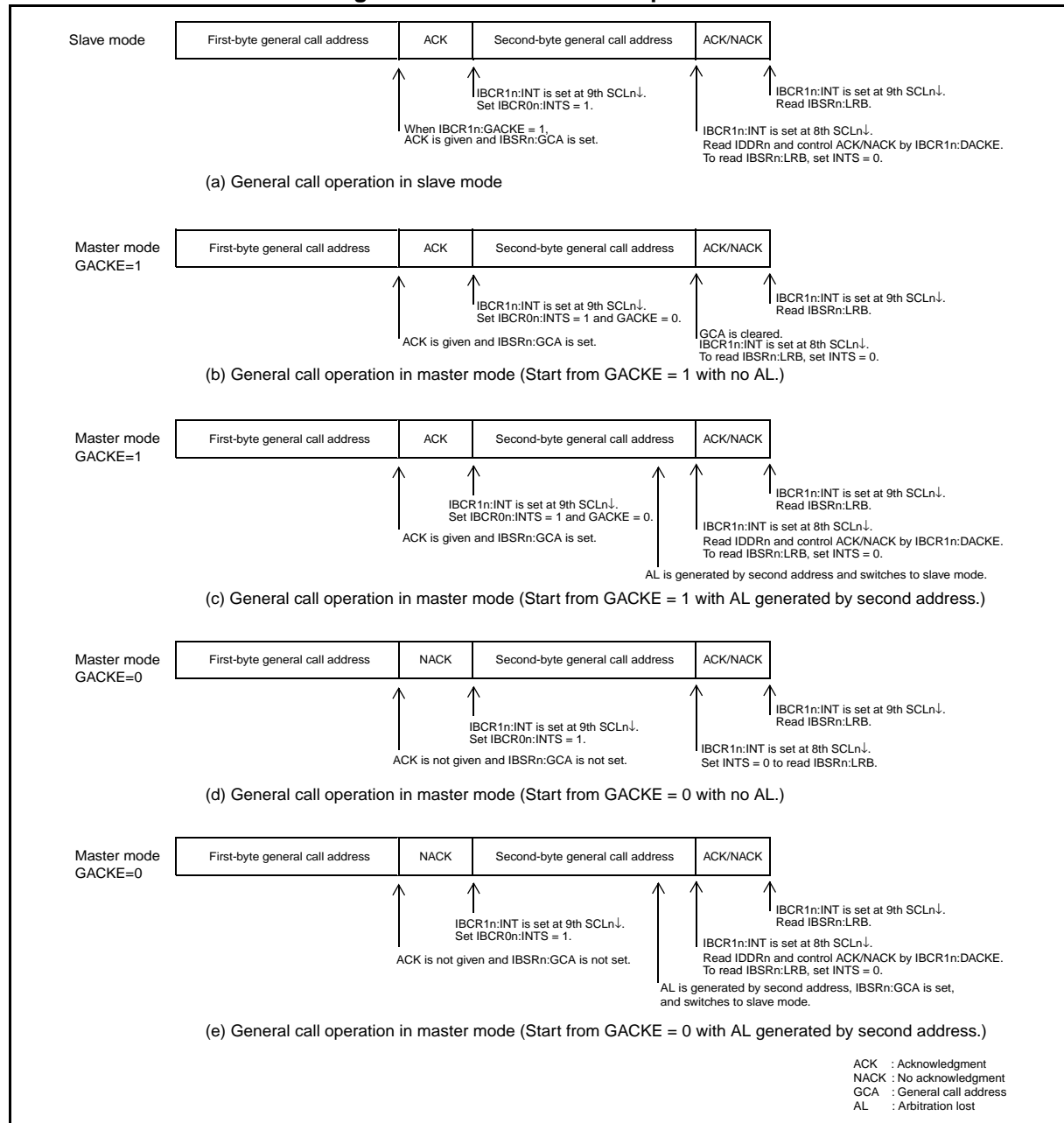
A data acknowledge bit used when data is received can be enabled or disabled by the IBCR1n:DACKE bit. In master mode, a data acknowledgment is generated if IBCR1n:DACKE = 1. In slave mode, a data acknowledgment is generated if an address acknowledgment has already been generated and IBCR1n:DACKE = 1. The received acknowledgment is saved in IBSRn:LRB in the ninth SCLn cycle.

- If the data ACK depends on the content of received data (such as packet error checking used by the SM bus), control the data ACK by setting the data ACK enable bit (IBCR1n:DACKE) after writing "1" to the IBCR0n:INTS bit (for example, by a previous transfer completion interrupt) so that the latest received data can be read.
- The latest data ACK (IBSRn:LRB) can be read after the ACK has been received (IBSRn:LRB must be read during the transfer completion interrupt triggered by the ninth SCLn cycle). Accordingly, if ACK is read when the IBCR0n:INTS bit is "1", write "0" to this bit in the transfer completion interrupt triggered by the eighth SCLn cycle so that another transfer completion interrupt will be triggered by the ninth SCLn cycle.

■ **General Call Address**

A general call address consists of the start address byte (0x00) and the second address byte that follows. To use a general call address, set IBCR1n:GACKE=1 before the acknowledge of the first byte general call address. In addition, the acknowledgment for the second address byte can be controlled as shown below.

**Figure 21.6-2 General Call Operation**



If this module sends a general call address at the same time as another device, you can determine whether the module successfully seized control of the bus by checking whether arbitration lost was detected when the second address byte was transferred. If arbitration lost was detected, the module goes to slave mode and continues to receive data from the master.



## ■ Stop Condition

The master can release the bus and end communications by generating a stop condition. Changing the SDA<sub>n</sub> line from "L" to "H" while SCL<sub>n</sub> is "H" generates a stop condition. This signals to the other devices on the bus that the master has finished communications (referred to below as "bus free"). However, the master can continue to generate start conditions without generating a stop condition. This is called a repeated start condition.

Writing "0" to the IBCR1<sub>n</sub>:MSS bit during an interrupt while in master mode (IBCR1<sub>n</sub>:MSS = 1, IBSR<sub>n</sub>:BB = 1, IBCR1<sub>n</sub>:INT = 1, and IBCR0<sub>n</sub>:ALF = 0) generates a stop condition and changes to slave mode. In any other circumstances other than those mentioned above, writing "0" to the IBCR1<sub>n</sub>:MSS bit is ignored.

## ■ Arbitration

The interface circuit is a true multi-master bus able to connect multiple master devices. Arbitration occurs when another master within the system simultaneously transfers data during a master transfer.

Arbitration occurs on the SDA<sub>n</sub> line while the SCL<sub>n</sub> line is at the "H" level. When the send data is "1" and the data on the SDA<sub>n</sub> line is "L" at the master, this is treated as arbitration lost. In this case, data output is halted and IBCR0<sub>n</sub>:ALF is set to "1". If this occurs, an interrupt is generated if arbitration lost interrupts have been enabled (IBCR0<sub>n</sub>:ALE = 1). If IBCR0<sub>n</sub>:ALF is set to "1", the module sets IBCR1<sub>n</sub>:MSS = 0 and IBSR<sub>n</sub>:TRX = 0, clears TRX, and goes to slave receive mode.

If IBCR0<sub>n</sub>:ALF is set to "1" when IBSR<sub>n</sub>:BB = 0, IBCR0<sub>n</sub>:ALF is cleared only by writing "0". If IBCR0<sub>n</sub>:ALF is set to "1" when IBSR<sub>n</sub>:BB = 1, IBCR0<sub>n</sub>:ALF is cleared only by clearing IBCR1<sub>n</sub>:INT to "0".

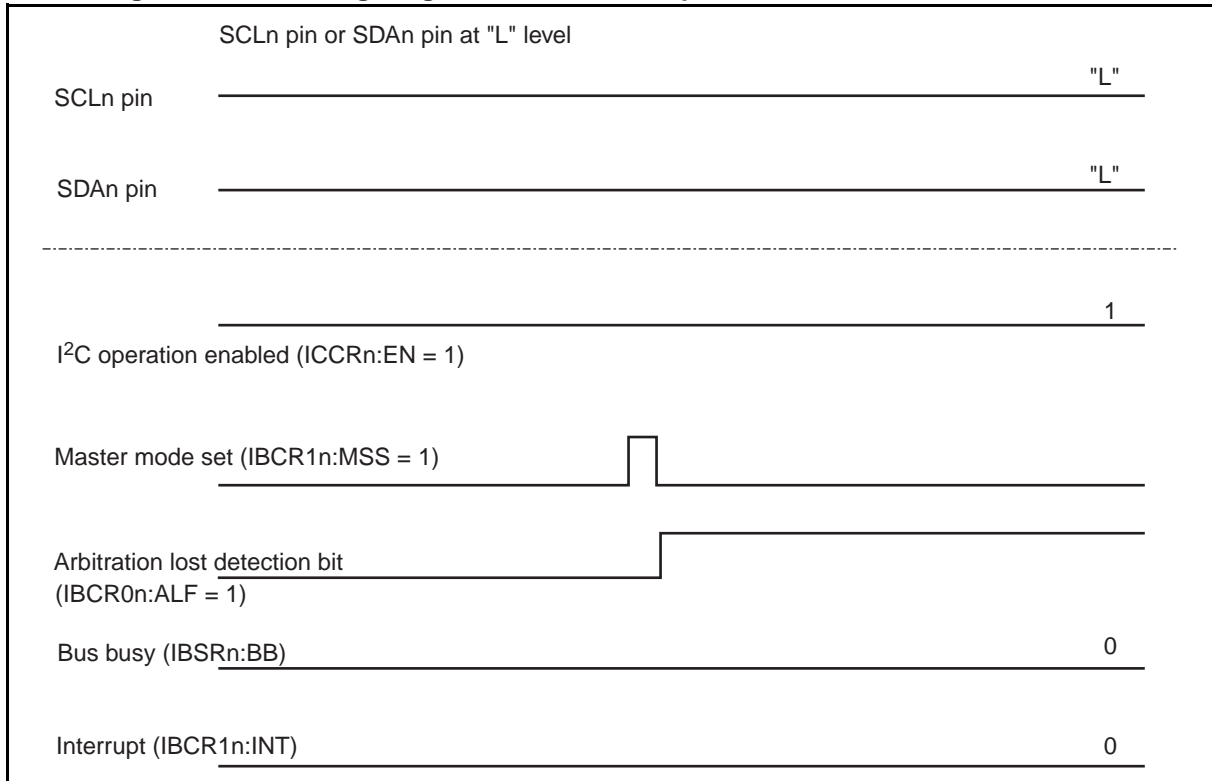
### ● Conditions for generating an arbitration lost interrupt when IBSR<sub>n</sub>:BB = 0

When a start condition is generated by the program (by setting the IBCR1<sub>n</sub>:MSS bit to "1") at the timing shown in Figure 21.6-3 or Figure 21.6-4, interrupt generation (IBCR1<sub>n</sub>:INT = 1) is prohibited by arbitration lost detection (IBCR0<sub>n</sub>:ALF = 1).

- Conditions (1) in which no interrupt is generated due to arbitration lost

If the program triggers a start condition (by setting the IBCR1<sub>n</sub>:MSS bit to "1") when no start condition has been detected (IBSR<sub>n</sub>:BB = 0) and the SDA<sub>n</sub> and SCL<sub>n</sub> line pins are at the "L" level.

**Figure 21.6-3 Timing Diagram with No Interrupt Generated with IBCR0n:ALF = 1**

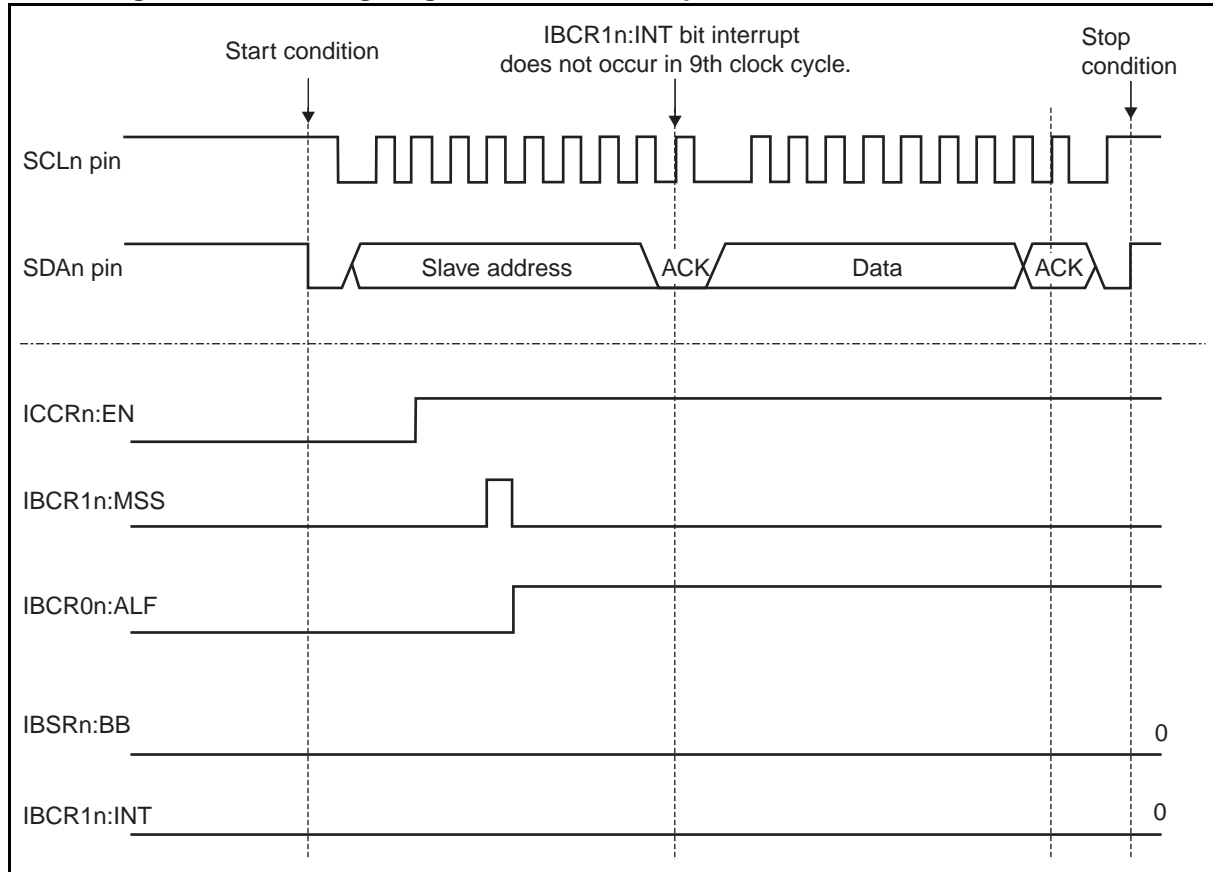


- Conditions (2) in which no interrupt is generated due to arbitration lost

If the program enables I<sup>2</sup>C bus interface operation (by setting the ICCRn:EN bit to "1") and triggers a start condition (by setting the IBCR1n:MSS bit to "1") when the I<sup>2</sup>C bus is in use by another master.

This is because, as shown in Figure 21.6-4, this I<sup>2</sup>C bus interface cannot detect the start condition (IBSRn:BB = 0) if another master starts communications on the I<sup>2</sup>C bus when the operation of this I<sup>2</sup>C bus interface has been disabled (ICCRn:EN = 0).

**Figure 21.6-4 Timing Diagram with No Interrupt Generated with IBCR0n:ALF = 1**



If this situation can occur, use the following procedure to set up the module from the software.

1. Trigger a start condition from the program (by setting the IBCR1n:MSS bit to "1").
2. Check the IBCR0n:ALF and IBSRn:BB bits in the arbitration lost interrupt.

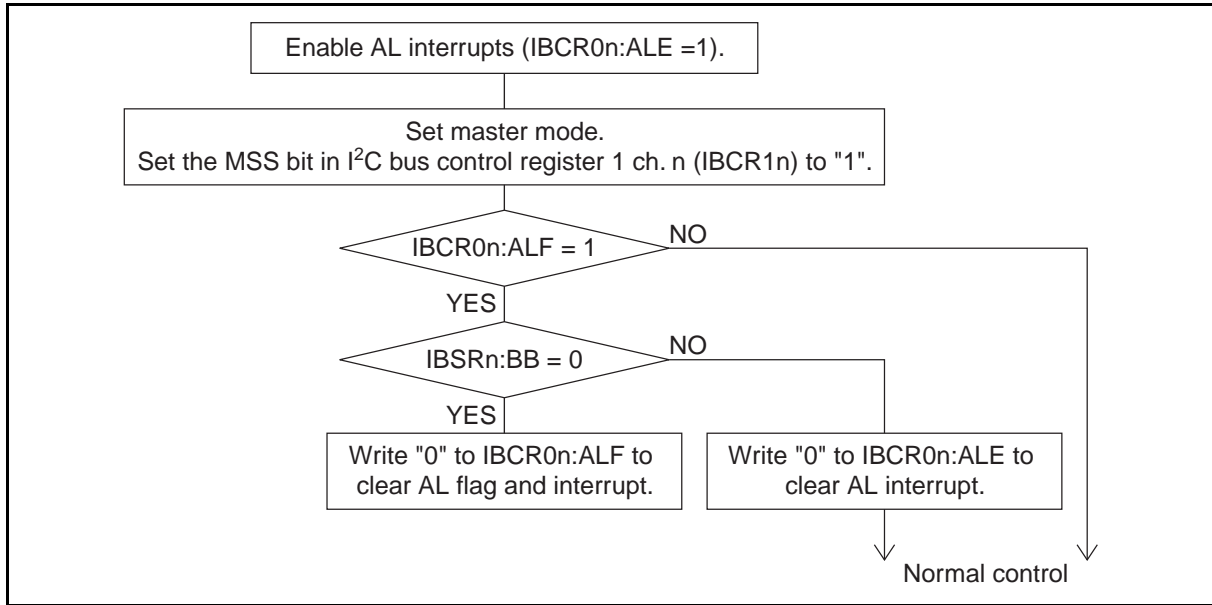
If IBCR0n:ALF = 1 and IBSRn:BB = 0, clear the IBCR0n:ALF bit to "0".

If IBCR0n:ALF = 1 and IBSRn:BB = 1, clear the IBCR0n:ALE bit to "0" and perform control as normal. (Normal control means writing "0" to the IBCR1n:INT bit in the INT interrupt to clear IBCR0n:ALF to "0".)

In other cases, perform control as normal (Normal control means writing "0" to the IBCR1n:INT bit in the INT interrupt to clear IBCR0n:ALF.)

The following sample flow chart illustrates the procedure:

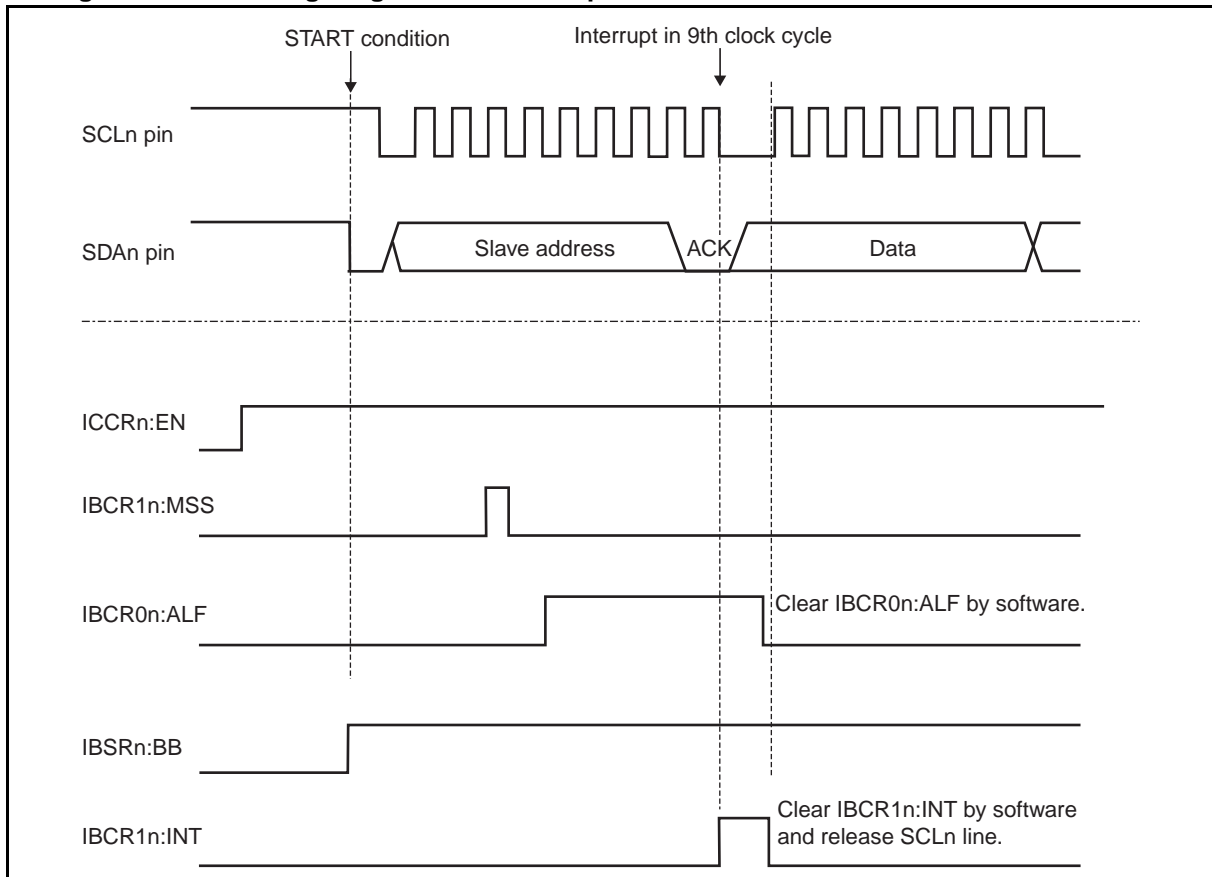
**Figure 21.6-5 Sample Flow Chart 1**



- Example of generating an interrupt (IBCR1n:INT = 1) with "IBCR0n:ALF = 1" detected

If a START condition is generated by the program (by setting the IBCR1n:MSS bit to "1") with the bus busy (IBSRn:BB = 1) and arbitration lost detected, a IBCR1n:INT bit interrupt occurs upon detection of "IBCR0n:ALF = 1".

**Figure 21.6-6 Timing Diagram with Interrupt Generated with "IBCR0n:ALF = 1" Detected**



## 21.6.2 Function to Wake up the MCU from Standby Mode

The wakeup function enables the I<sup>2</sup>C macro to be accessed while the MCU is in stop or watch mode.

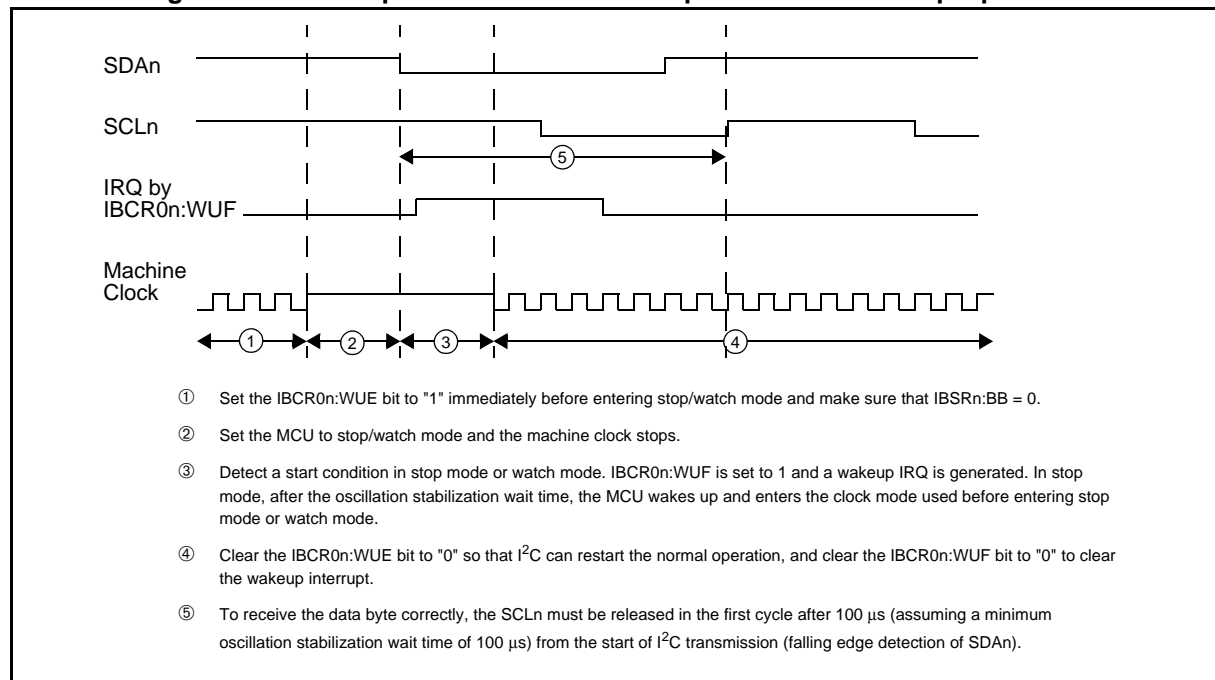
### ■ Function to Wake Up the MCU from Standby Mode

The I<sup>2</sup>C macro includes a function to wake up the MCU from standby mode. The function is enabled by writing "1" to the IBCR0n:WUE bit.

With the MCU in stop mode or watch mode and the IBCR0n:WUE bit set to "1", if a start condition is detected on the I<sup>2</sup>C bus, the wakeup interrupt request flag bit (IBCR0n:WUF) is set to "1" and the wakeup interrupt request is generated to wake up the MCU from stop/watch mode.

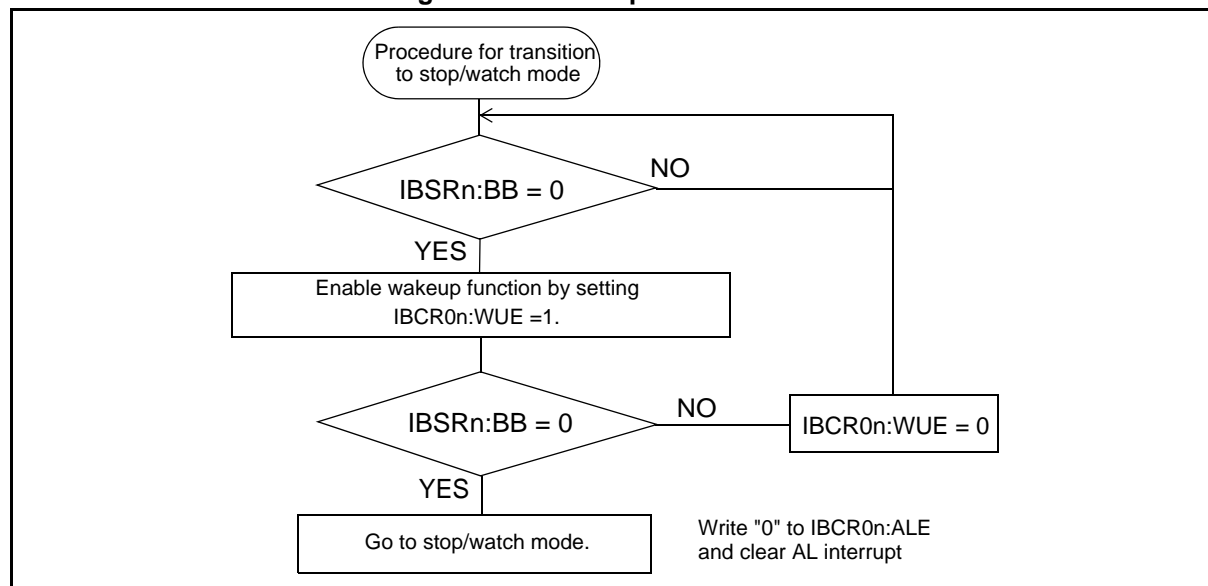
- Set IBCR0n:WUE to "1" immediately before setting the MCU to stop or watch mode. Similarly, clear IBCR0n:WUE (by writing "0") after the MCU wakes up from stop or watch mode so that I<sup>2</sup>C operation can restart as soon as possible.
- The wakeup function only applies to the MCU stop and watch modes.

**Figure 21.6-7 Comparison of Normal I<sup>2</sup>C Operation and Wakeup Operation**



The following sample flow chart illustrates the wakeup function.

**Figure 21.6-8 Sample Flow Chart 2**



## 21.7 Registers

---

**This section describes the registers of the I<sup>2</sup>C bus interface.**

---

**Table 21.7-1 List of I<sup>2</sup>C Bus Interface Registers**

| Register abbreviation | Register name                                 | Reference |
|-----------------------|---|-----------|
| IBCR0n                | I <sup>2</sup> C bus control register 0 ch. n | 21.7.1    |
| IBCR1n                | I <sup>2</sup> C bus control register 1 ch. n | 21.7.2    |
| IBSRn                 | I <sup>2</sup> C bus status register ch. n    | 21.7.3    |
| IDDRn                 | I <sup>2</sup> C data register ch. n          | 21.7.4    |
| IAARn                 | I <sup>2</sup> C address register ch. n       | 21.7.5    |
| ICCRn                 | I <sup>2</sup> C clock control register ch. n | 21.7.6    |

## 21.7.1 I<sup>2</sup>C Bus Control Register 0 ch. n (IBCR0n)

The I<sup>2</sup>C bus control register 0 ch. n (IBCR0n) controls the address acknowledge in the transmission of the first byte, selects the timing of the transfer completion interrupt, and enables or disables the arbitration lost interrupt, the STOP condition detection interrupt and MCU standby wakeup function.

### Register Configuration

| bit           | 7     | 6    | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-------|------|-----|-----|-----|-----|-----|-----|
| Field         | AACKX | INTS | ALF | ALE | SPF | SPE | WUF | WUE |
| Attribute     | R/W   | R/W  | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0     | 0    | 0   | 0   | 0   | 0   | 0   | 0   |

### Register Functions

#### [bit7] AACKX: Address acknowledge disable bit

This bit controls the address acknowledge in the transmission of the first byte.

Writing "0" to this bit causes the address acknowledge to be output automatically (The address acknowledge is returned automatically if the slave address matches).

Writing "1" to this bit prevents the address acknowledge from being output.

Modify the setting of this bit in either of the following ways:

- Write "1" to this bit in master mode.
- Clear this bit to "0" after checking that the bus busy bit (IBSRn:BB) is "0".

Notes:

- If AACKX = 1 and IBSRn:FBT = 0 when a transfer completion interrupt is generated (IBCR1n:INT = 1), no address acknowledge is output even though the I<sup>2</sup>C address matches the slave address. Clear the IBCR1n:INT bit to "0" as an interrupt is generated upon completion of transfer of each byte of address/data in the same way as during addressing.
- If AACKX = 1 and IBSRn:FBT = 1 when a transfer completion interrupt is generated (IBCR1n:INT = 1), "1" might be written to AACKX after addressing as in slave mode. Either continue normal communication after setting AACKX to "0" again or restart communication after disabling I<sup>2</sup>C operation (ICCRn:EN = 0).

| bit7        | Details                       |
|-------------|-------------------------------|
| Writing "0" | Enable address acknowledge.   |
| Writing "1" | Disables address acknowledge. |

#### [bit6] INTS: Timing select bit for transfer completion flag bit at data reception

This bit selects the timing of the transfer completion interrupt (IBCR1n:INT) when data is received. Modify this bit only when IBSRn:TRX = 0 and IBSRn:FBT = 0.

Writing "0" to this bit sets the transfer completion interrupt request flag bit (IBCR1n:INT) to "1" in the ninth SCLn cycle.

Writing "1" to this bit sets the transfer completion interrupt request flag bit (IBCR1n:INT) to "1" in the eighth SCLn cycle.

Notes:

- The transfer completion interrupt request flag bit (IBCR1n:INT) is set to "1" always in the ninth SCLn cycle except during data reception (IBSRn:TRX = 1 or IBSRn:FBT = 1).
- If the data acknowledge depends on the content of the received data (such as packet error checking used by the SM bus), control the data acknowledge by setting the data acknowledge enable bit (IBCR1n:DACKE) after writing "1" to this bit (for example, using a previous transfer completion interrupt) to read latest received data.



## MB95850K/860K/870K Series

- The latest data acknowledge (IBSRn:LRB) can be read after the acknowledge has been received (IBSRn:LRB must be read during the transfer completion interrupt in the ninth SCLn cycle.) If acknowledge is read when this bit is "1", therefore, write "0" to this bit in the transfer completion interrupt in the eighth SCLn cycle so that another transfer completion interrupt will occur in the ninth SCLn cycle.

| bit6               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Sets the INT bit to "1" in the ninth SCLn cycle.  |
| <b>Writing "1"</b> | Sets the INT bit to "1" in the eighth SCLn cycle. |

### [bit5] ALF: Arbitration lost interrupt request flag bit

This bit detects the arbitration lost.

An arbitration lost interrupt request is generated if this bit and the IBCR0n:ALE bit are both "1".

If one of the following conditions is satisfied, this bit is set to "1".

- An arbitration lost is detected when this device is transmitting data/address as a master.
- "1" is written to the IBCR1n:MSS bit with the I<sup>2</sup>C bus being used by another system. However, when "1" is written to the MSS bit after this device returns AACK or GACK as a slave, the ALF bit is not set to "1".

If one of the following conditions is satisfied, this bit is set to "0".

- With IBSRn:BB = 0, "0" is written to the ALF bit.
- "0" is written to the IBCR1n:INT bit to clear the transmission completion flag bit.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit5               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no arbitration lost has been detected. |
| <b>Reading "1"</b> | Indicates that an arbitration lost has been detected. |
| <b>Writing "0"</b> | Clears this bit.                                      |
| <b>Writing "1"</b> | Has no effect on operation.                           |

### [bit4] ALE: Arbitration lost interrupt enable bit

This bit enables or disables the arbitration lost interrupt.

When this bit and the ALF bit are both set to "1", an arbitration lost interrupt request is generated.

| bit4               | Details                                  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the arbitration lost interrupt. |
| <b>Writing "1"</b> | Enables the arbitration lost interrupt.  |

### [bit3] SPF: STOP detection interrupt request flag bit

This bit detects the STOP condition.

When this bit and the IBCR0n:SPE bit are both set to "1", a STOP detection interrupt request is generated.

With the bus busy, when a valid STOP condition is correctly detected, this bit is set to "1".

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit3               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no STOP condition has been detected. |
| <b>Reading "1"</b> | Indicates that a STOP condition has been detected.  |
| <b>Writing "0"</b> | Clears this bit.                                    |
| <b>Writing "1"</b> | Has no effect on operation.                         |

[bit2] SPE: STOP detection interrupt enable bit

This bit enables or disables the STOP detection interrupt.

When this bit and the SPF bit are both set to "1", a STOP detection interrupt request is generated.

| bit2               | Details                                |
|--------------------|--|
| <b>Writing "0"</b> | Disables the STOP detection interrupt. |
| <b>Writing "1"</b> | Enables the STOP detection interrupt.  |

[bit1] WUF: MCU standby mode wakeup interrupt request flag bit

This bit detects an MCU standby mode wakeup in stop mode or watch mode.

When this bit and the IBCR0n:WUE bit are both set to "1", a wakeup interrupt request is generated.

With the wakeup function enabled (WUE = 1), when a START condition is detected, the WUF bit is set to "1".

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit1               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that no START condition has been detected. |
| <b>Reading "1"</b> | Indicates that a START condition has been detected.  |
| <b>Writing "0"</b> | Clears this bit.                                     |
| <b>Writing "1"</b> | Has no effect on operation.                          |

[bit0] WUE: MCU standby mode wakeup function enable bit

This bit enables or disables the MCU standby mode wakeup function in stop mode or watch mode.

In stop mode or watch mode, when this bit is set to "1" and a START condition is generated, a wakeup interrupt request is generated to start the I<sup>2</sup>C operation.

| bit0               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the MCU standby mode wakeup function in stop mode or watch mode. |
| <b>Writing "1"</b> | Enables the MCU standby mode wakeup function in stop mode or watch mode.  |

Notes:

- Write "1" to this bit right before the MCU enters stop mode or watch mode. To ensure that the I<sup>2</sup>C operation can restart immediately after the MCU wakes up from stop mode or watch mode, clear (write "0" to) this bit as soon as possible.
- When a wakeup interrupt request is generated, the MCU wakes up after the oscillation stabilization wait time elapses. In order to prevent data loss from occurring immediately after the MCU wakes up, after 100 μs (assuming that the minimum oscillation stabilization wait time is 100 μs) elapses since a wakeup caused by the start of I<sup>2</sup>C transmission (upon detection of the falling edge of SDA<sub>n</sub>), the SCL<sub>n</sub> must rise in the first cycle and the first bit must be received as data.
- In standby mode of the MCU, the status flags, state machine, and I<sup>2</sup>C bus output for the I<sup>2</sup>C function keep their states existing before the MCU entered standby mode. To prevent a hang-up of the entire I<sup>2</sup>C bus system, ensure that IBSRn:BB is set to "0" before making the MCU enter standby mode.
- The wakeup function does not support the transition of the MCU to stop mode or watch mode with the BB bit set to "1". When the MCU enters stop mode or watch mode with the BB bit set to "1", a bus error occurs upon detection of a START condition.
- The wakeup function is effective only when the MCU is in stop mode or watch mode.

Note:

The values of the AACKX, INTS, and WUE bits in the IBCR0n register become "0" and non-writable when the I<sup>2</sup>C operation is disabled (ICCRn:EN = 0).

## MB95850K/860K/870K Series

### 21.7.2 I<sup>2</sup>C Bus Control Register 1 ch. n (IBCR1n)

The I<sup>2</sup>C bus control register 1 ch. n (IBCR1n) controls the following functions: bus error interrupt, START condition generation, master/slave mode selection, data acknowledge, general call acknowledge and transfer completion interrupt.

#### ■ Register Configuration

|               |     |      |     |     |       |       |      |     |
|---------------|-----|------|-----|-----|-------|-------|------|-----|
| bit           | 7   | 6    | 5   | 4   | 3     | 2     | 1    | 0   |
| Field         | BER | BEIE | SCC | MSS | DACKE | GACKE | INTE | INT |
| Attribute     | R/W | R/W  | R/W | R/W | R/W   | R/W   | R/W  | R/W |
| Initial value | 0   | 0    | 0   | 0   | 0     | 0     | 0    | 0   |

#### ■ Register Functions

[bit7] BER: Bus error interrupt request flag bit

This bit detects the bus error.

When this bit and the BEIE bit are both set to "1", a bus error interrupt is generated.

This bit is set to "1" when an invalid START condition or an invalid STOP condition is detected.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit7               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no bus error has been detected.  |
| <b>Reading "1"</b> | Indicates that an invalid START condition or an invalid STOP condition has been detected. |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.   |

[bit6] BEIE: Bus error interrupt enable bit

This bit enables or disables the bus error interrupt.

When this bit and the BER bit are both set to "1", a bus error interrupt request is generated.

| bit6               | Details                           |
|--------------------|-----------------------------------|
| <b>Writing "0"</b> | Disables the bus error interrupt. |
| <b>Writing "1"</b> | Enables the bus error interrupt.  |

[bit5] SCC: START condition generation bit

This bit generates a repeated START condition to restart communications in master mode.

In master mode, writing "1" to this bit generates a repeated START condition.

Writing "0" to this bit has no effect on operation.

| bit5               | Details  |
|--------------------|--|
| <b>Read access</b> | The read value is always "0".                        |
| <b>Writing "0"</b> | Has no effect on operation.                          |
| <b>Writing "1"</b> | Generates a repeated START condition in master mode. |

Notes:

- Do not set this bit to "1" or the IBCR1n:MSS bit to "0" at the same time.
- With the IBCR1n:INT bit set to "0", an attempt to write "1" to the SCC bit is ignored (no START condition is generated). In addition, with the INT bit set to "1", when writing "1" to the SCC bit and writing "0" to the INT bit occur simultaneously, writing "1" to the SCC bit is given priority.

[bit4] MSS: Master/slave select bit

This bit selects an operation mode from master mode and slave mode.

Writing "1" to this bit while the I<sup>2</sup>C bus is in the idle state (IBSRn:BB = 0) selects master mode, generates a START condition, and then starts address transfer.

Writing "0" to the bit while the I<sup>2</sup>C bus is in the busy state (IBSRn:BB = 1) selects slave mode, generates a STOP condition, and then terminates data transfer.

When an arbitration lost occurs during data or address transfer in master mode, this bit is cleared to "0" and the operation mode switches to slave mode.

| bit4               | Details     |
|--------------------|-------------|
| <b>Writing "0"</b> | Slave mode  |
| <b>Writing "1"</b> | Master mode |

Notes:

- Do not set this bit to "0" or the SCC bit to "1" at the same time.
- With the INT bit set to "0", an attempt to write "0" to the MSS bit is ignored. With the INT bit set to "1", when writing "0" to the MSS bit and writing "0" to the INT bit occur simultaneously, writing "0" to the MSS bit is given priority.
- In slave mode, during transmission or reception, writing "1" to the MSS bit does not set the ALF bit to "1". Do not write "1" to the MSS bit during transmission or reception in slave mode.

[bit3] DACKE: Data acknowledge enable bit

This bit controls the data acknowledge in data reception.

Writing "0" to this bit disables data acknowledge output.

Writing "1" to this bit enables data acknowledge output. In master mode, with this bit set to "1", a data acknowledge is output in the ninth SCLn cycle during data reception. In slave mode, a data acknowledge is output in the ninth SCLn cycle only when an address acknowledgment has already been output.

| bit3               | Details                           |
|--------------------|-----------------------------------|
| <b>Writing "0"</b> | Disables data acknowledge output. |
| <b>Writing "1"</b> | Enables data acknowledge output.  |

[bit2] GACKE: General call address acknowledge enable bit

This bit controls the general call address acknowledge.

Writing "0" to this bit disables general call address acknowledge output.

With this bit set to "1", in master mode or slave mode, when a general call address acknowledge (0x00) is received, a general call address acknowledge is output.

| bit2               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the general call address acknowledge. |
| <b>Writing "1"</b> | Enables the general call address acknowledge.  |

[bit1] INTE: Transfer completion interrupt enable bit

This bit enables or disables the transfer completion interrupt.

When this bit and the IBCR1n:INT bit are both set to "1", a transfer completion interrupt request is generated.

| bit1               | Details                                     |
|--------------------|---|
| <b>Writing "0"</b> | Disables the transfer completion interrupt. |
| <b>Writing "1"</b> | Enables the transfer completion interrupt.  |

## MB95850K/860K/870K Series

### [bit0] INT: Transfer completion interrupt request flag bit

This bit detects the transfer completion.

When this bit and the INTE bit are both set to "1", a transfer completion interrupt request is generated.

If one of the following four conditions is satisfied, upon completion of transferring 1-byte address or data (the setting of the INTS bit determines whether the 1-byte address or data includes an acknowledge.), this bit is set to "1".

- In bus master mode
- The device is addressed as slave.
- The I<sup>2</sup>C bus interface has received a general call address.
- The I<sup>2</sup>C bus interface has detected an arbitration lost.
- Arbitration lost detected

If one of the following two conditions is satisfied, this bit is set to "0".

- "0" is written to this bit.
- In master mode, a repeated START condition (IBCR1n:SCC = 1) or a STOP condition (IBCR1n:MSS = 0) is generated.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

Writing "0" to clear this bit (its value becomes "0") releases the SCLn line, and the transmission of the next byte of data is then enabled.

| bit0               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that data transfer has not been completed.                               |
| <b>Reading "1"</b> | Indicates that 1-byte data (including an acknowledge) transfer has been completed. |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.  |

#### Notes:

- In the case of writing "1" to the SCC bit while this bit is "0", the setting of the SCC bit is given priority, and a START condition is generated.
- In the case of writing "0" to the MSS bit while this bit is "0", the setting of the MSS bit is given priority, and a STOP condition is generated.
- During data reception, with the IBCR0n:INTS bit already set to "1", this bit becomes "1" after 1-byte data (not including an acknowledge) transfer is completed. If the INTS bit is set to "0", this bit becomes "1" after the transmission/reception of 1-byte data/address (including an acknowledge) is completed.

---

#### Notes:

- When clearing the interrupt request flag bit (IBCR1n:BER) by writing "0" to it, do not update the interrupt request enable bit (IBCR1n:BEIE) at the same time.
  - All bits in the IBCR1n register except the BER and BEIE bits are cleared to "0" either when the I<sup>2</sup>C bus interface operation is disabled (ICCRn:EN = 0).
-

### 21.7.3 I<sup>2</sup>C Bus Status Register ch. n (IBSRn)

The I<sup>2</sup>C bus status register ch. n (IBSRn) indicates the status of the I<sup>2</sup>C bus interface.

#### Register Configuration

|               |    |     |   |     |     |     |     |     |
|---------------|----|-----|---|-----|-----|-----|-----|-----|
| bit           | 7  | 6   | 5 | 4   | 3   | 2   | 1   | 0   |
| Field         | BB | RSC | — | LRB | TRX | AAS | GCA | FBT |
| Attribute     | R  | R   | — | R   | R   | R   | R   | R   |
| Initial value | 0  | 0   | 0 | 0   | 0   | 0   | 0   | 0   |

#### Register Functions

[bit7] **BB: Bus busy bit**

This bit indicates the bus state.

| bit7               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that a STOP condition has been detected and the bus has entered the idle state.  |
| <b>Reading "1"</b> | Indicates that a START condition has been detected and the bus has entered the busy state. |

[bit6] **RSC: Repeated START condition detection bit**

This bit detects the repeated START condition.

This bit is set to "1" when a repeated START condition is detected.

If one of the following conditions is satisfied, this bit is set to "0".

- "0" is written to the IBCR1n:INT bit.
- In slave mode, the slave address does not match the address set in the IAARn register.
- In slave mode, the slave address matches the address set in the IAARn register but the IBCR0n:AACKX bit is set to "1".
- In slave mode, the device receives a general call address, but the IBCR1n:GACKE bit is set to "0".
- A STOP condition is detected.

| bit6               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that no repeated START condition has been detected.                      |
| <b>Reading "1"</b> | Indicates that the bus is in use and a repeated START condition has been detected. |

[bit5] **Undefined bit**

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

## MB95850K/860K/870K Series

### [bit4] LRB: Acknowledge storage bit

This bit captures the value of the SDAn line in the ninth shift clock cycle during data byte transfer.

This bit is set to "1" when no acknowledge has been detected (SDAn = "H").

If one of the following conditions is satisfied, this bit is set to "0".

- An acknowledge is detected (SDAn = "L").
- A START condition or a STOP condition is detected.

| bit4               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that an acknowledge has been detected in the ninth shift clock cycle. |
| <b>Reading "1"</b> | Indicates that no acknowledge has been detected in the ninth shift clock cycle. |

Note: According to the above description, this bit must be read after an acknowledge (Read the bit value at a transfer completion interrupt in the ninth SCLn cycle). Therefore, if an acknowledge is read with the IBCR0n:INTS bit set to "1", write "0" to the INTS bit at a transfer completion interrupt generated in the eighth SCLn cycle so that another transfer completion interrupt is to be generated in the ninth SCLn cycle.

### [bit3] TRX: Data transfer status bit

This bit indicates the data transfer mode.

This bit is set to "1" when data transfer is executed in transmission mode.

If one of the following conditions is satisfied, this bit is set to "0".

- In receive mode, data transfer is executed.
- The device receives an NACK in slave transmit mode.

| bit3               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the data transfer mode is receive mode.  |
| <b>Reading "1"</b> | Indicates that the data transfer mode is transmit mode. |

### [bit2] AAS: Addressing detection bit

This bit indicates whether the MCU has undergone addressing in slave mode.

This bit is set to "1" when the MCU has undergone addressing in slave mode.

This bit is set to "0" when a START condition or a STOP condition has been detected.

| bit2               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the MCU has not undergone addressing in slave mode. |
| <b>Reading "1"</b> | Indicates that the MCU has undergone addressing in slave mode.     |

[bit1] GCA: General call address detection bit

This bit detects a general call address.

If one of the following conditions is satisfied, this bit is set to "1".

- The device receives a general call address (0x00) in slave mode.
- With IBCR1n:GACKE set to "1", the device receives a general call address (0x00) in master mode.
- In master mode, an arbitration lost is detected during the transmission of the second byte of a general call address.

If one of the following conditions is satisfied, this bit is set to "0".

- A START condition or a STOP condition is detected.
- In master mode, no arbitration lost is detected during the transmission of the second byte of a general call address.

| bit1               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the I <sup>2</sup> C bus interface has not received a general call address (0x00) in slave mode. |
| <b>Reading "1"</b> | Indicates that the I <sup>2</sup> C bus interface has received a general call address (0x00) in slave mode.     |

[bit0] FBT: First byte detection bit

This bit detects the first byte.

This bit is set to "1" when a START condition is detected.

If one of the following conditions is satisfied, this bit is set to "0".

- "0" is written to the IBCR1n:INT bit.
- In slave mode, the slave address does not match the address set in the IAARn register.
- In slave mode, the slave address matches the address set in the IAARn register, but the IBCR0n:AACKX bit is "1".
- In slave mode, the device receives a general call address, but the IBCR1n:GACKE bit is "0".
- In slave mode, a STOP condition is detected.

| bit0               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the receive data is not the first byte in data reception.       |
| <b>Reading "1"</b> | Indicates that the receive data is the first byte (address) in data reception. |



## MB95850K/860K/870K Series

### 21.7.4 I<sup>2</sup>C Data Register ch. n (IDDRn)

The I<sup>2</sup>C data register ch. n (IDDRn) sets the data or address to be transmitted, and holds the data or address received.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Field         | D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
| Attribute     | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

#### ■ Register Functions

In transmit mode, each bit of the data or address value written to the register is shifted to the SDA<sub>n</sub> line, starting with the MSB. The write side of this register is double-buffered, where if the bus is in use (IBSR<sub>n</sub>:BB = 1), the write data is loaded to the 8-bit shift register either when the current data transfer completion interrupt is cleared (writing "0" to the IBCR1<sub>n</sub>:INT bit) or when a repeated start condition is generated (writing "1" to the IBCR1<sub>n</sub>:SCC bit). Each bit of the shift register data is output (shifted) to the SDA<sub>n</sub> line.

Note that writing to this register has no effect on the current data transfer. In slave mode, however, data is transferred to the shift register after the address is determined.

The received data or address can be read from this register at the transfer completion interrupt (IBCR1<sub>n</sub>:INT = 1). However, since the serial transfer register is directly read from when the received data or address is read, the receive data is valid only when the INT bit is "1".

## 21.7.5 I<sup>2</sup>C Address Register ch. n (IAARn)

The I<sup>2</sup>C address register ch. n (IAARn) register sets the slave address.  
 In slave mode, the I<sup>2</sup>C bus interface receives address data from the master and compares it with the value of this register.

### ■ Register Configuration

| bit           | 7 | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|---|-----|-----|-----|-----|-----|-----|-----|
| Field         | — | A6  | A5  | A4  | A3  | A2  | A1  | A0  |
| Attribute     | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

### ■ Register Functions

[bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

[bit6:0] A[6:0]: Address bits

These bits set the slave address.

## MB95850K/860K/870K Series

### 21.7.6 I<sup>2</sup>C Clock Control Register ch. n (ICCRn)

The I<sup>2</sup>C clock control register ch. n (ICCRn) register enables the I<sup>2</sup>C operation and selects the shift clock frequency.

#### ■ Register Configuration

| bit           | 7    | 6        | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|------|----------|-----|-----|-----|-----|-----|-----|
| Field         | DMBP | Reserved | EN  | CS4 | CS3 | CS2 | CS1 | CS0 |
| Attribute     | R/W  | W        | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0    | 0        | 0   | 0   | 0   | 0   | 0   | 0   |

#### ■ Register Functions

##### [bit7] DMBP: Divider m bypass bit

This bit is used to bypass the divider m to generate the shift clock frequency.

Writing "0" to this bit sets the value set in the CS[4:3] bits as the divider m value ( $m = \text{ICCRn:CS}[4:3]$ ).

When "1" is written to this bit, the divider m is to be bypassed.

Do not write "1" to this bit when the value of divider n is "4" ( $\text{ICCRn:CS}[2:0] = 0b000$ ).

| bit7               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | The settings of ICCRn:CS[4:3] (clock divide m) are valid.   |
| <b>Writing "1"</b> | The settings of ICCRn:CS[4:3] (clock divide m) are invalid. |

##### [bit6] Reserved bit

Always set this bit "0".

##### [bit5] EN: I<sup>2</sup>C bus interface operation enable bit

This bit enables the I<sup>2</sup>C bus interface operation.

Writing "0" to this bit disables the I<sup>2</sup>C bus interface operation and clears the following bits to "0".

- AACKX, INTS, and WUE bits in the IBCR0n register
- All bits in the IBCR1n register except the BER and BEIE bits
- All bits in the IBSRn register

Writing "1" to this bit enables the I<sup>2</sup>C bus interface operation.

| bit5               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the I <sup>2</sup> C bus interface operation. |
| <b>Writing "1"</b> | Enables the I <sup>2</sup> C bus interface operation.  |

[bit4:3] CS[4:3]: Clock-1 select bits (Divider m)

[bit2:0] CS[2:0]: Clock-2 select bits (Divider n)

These bits set the shift clock frequency.

The shift clock frequency (Fsck) is set by the following equation:

$$F_{sck} = \frac{\phi}{(m \times n + 2)}$$

$\phi$  represents the machine clock frequency (MCLK).

| bit4:3       | Details |
|--------------|---------|
| Writing "00" | 5       |
| Writing "01" | 6       |
| Writing "10" | 7       |
| Writing "11" | 8       |

| bit2:0        | Details |
|---------------|---------|
| Writing "000" | 4       |
| Writing "001" | 8       |
| Writing "010" | 22      |
| Writing "011" | 38      |
| Writing "100" | 98      |
| Writing "101" | 128     |
| Writing "110" | 256     |
| Writing "111" | 512     |

---

Note:

If the standby mode wakeup function is not used, disable the I<sup>2</sup>C bus interface operation before making the MCU transit to stop mode or watch mode.

---

## 21.8 Notes on Using I<sup>2</sup>C Bus Interface

---

This section provides notes on using the I<sup>2</sup>C bus interface.

---

### ■ Notes on Using I<sup>2</sup>C Bus Interface

- Notes on setting I<sup>2</sup>C bus interface registers
  - Enable the I<sup>2</sup>C bus interface operation (ICCRn:EN) before setting the I<sup>2</sup>C bus control registers ch. n (IBCR0n and IBCR1n).
  - Setting the master/slave select bit (IBCR1n:MSS) to "1" starts data transfer.
- Notes on setting the shift clock frequency
  - The shift clock frequency can be calculated by determining the m, n, and DMBP values using the F<sub>sck</sub> equation. See "21.7.6 I<sup>2</sup>C Clock Control Register ch. n (ICCRn)" for details of the F<sub>sck</sub> equation.
  - Do not write "1" to the DMBP bit in the ICCRn register if the value of n is 4 (ICCRn:CS[2:0] = 0b000).
- Notes on priority for simultaneous write operations
  - Conflict between next byte transfer and stop condition  
When writing "0" to IBCR1n:MSS and clearing IBCR1n:INT occur simultaneously, the MSS bit is given priority and a STOP condition is generated.
  - Conflict between next byte transfer and start condition  
When writing "1" to IBCR1n:SCC and clearing IBCR1n:INT occur simultaneously, the SCC bit is given priority and a START condition is generated.
- Notes on setting up using software
  - Do not select the repeated START condition (IBCR1n:SCC = 1) or slave mode (IBCR1n:MSS = 0) simultaneously.
  - The I<sup>2</sup>C bus interface cannot return from interrupt processing if an interrupt request enable bit is enabled (IBCR1n:BEIE = 1 or IBCR1n:INTE = 1) with the interrupt request flag bit (IBCR1n:BER or IBCR1n:INT) set to "1". Clear the BER bit or the INT bit.
  - The following bits are cleared to "0" when the I<sup>2</sup>C bus interface operation is disabled (ICCRn:EN = 0).
    - AACKX, INTS, and WUE bits in the IBCR0n register
    - All bits in the IBCR1n register except the BER bit and the BEIE bit
    - All bits in the IBSRn register
- Notes on data acknowledgment
 

In slave mode, a data acknowledge is generated if one of the following conditions is satisfied.

  - The received address matches the value in the address register (IAARn) and IBCR0n:AACKX is "0".
  - A general call address (0x00) is received and IBCR1n:GACKE is "1".

- Notes on selecting the transfer complete timing
  - The transfer complete timing select bit (IBCR0n:INTS) is valid only during data reception (IBSRn:TRX = 0 and IBSRn:FBT = 0).
  - In an operation other than data reception (IBSRn:TRX = 1 or IBSRn:FBT = 1), the transfer completion interrupt (IBCR1n:INT) is always generated in the ninth SCLn cycle.
  - If the data acknowledge depends on the content of the received data (such as packet error checking used by the SM bus), control the data acknowledge by setting the data acknowledge enable bit (IBCR1n:DACKE) after writing "1" to the IBCR0n:INTS bit (for example, using a previous transfer completion interrupt) to read latest received data.
  - The latest data acknowledge (IBSRn:LRB) can be read after the acknowledge is received (IBSRn:LRB must be read at a transfer completion interrupt in the ninth SCLn cycle.) Therefore, if an acknowledge is read with the IBCR0n:INTS bit set to "1", write "0" to the INTS bit at a transfer completion interrupt generated in the eighth SCLn cycle so that another transfer completion interrupt is to be generated in the ninth SCLn cycle.
  
- Notes on using the MCU standby mode wakeup function
  - Write "1" to the IBCR0n:WUE bit right before the MCU enters stop mode or watch mode. To ensure that the I<sup>2</sup>C operation can restart immediately after the MCU wakes up from stop mode or watch mode, clear (write "0" to) this bit as soon as possible.
  - When a wakeup interrupt request is generated, the MCU wakes up after the oscillation stabilization wait time elapses. In order to prevent data loss from occurring immediately after the MCU wakes up, after 100 μs (assuming that the minimum oscillation stabilization wait time is 100 μs) elapses since a wakeup caused by the start of I<sup>2</sup>C transmission (upon detection of the falling edge of SDA<sub>n</sub>), the SCL<sub>n</sub> must rise in the first cycle and the first bit must be received as data.
  - In standby mode of the MCU, the status flags, state machine, and I<sup>2</sup>C bus output for the I<sup>2</sup>C function keep their states existing before the MCU entered standby mode. To prevent a hang-up of the entire I<sup>2</sup>C bus system, ensure that IBSRn:BB is set to "0" before making the MCU enter standby mode.
  - The wakeup function does not support the transition of the MCU to stop mode or watch mode with the BB bit set to "1". When the MCU enters stop mode or watch mode with the BB bit set to "1", a bus error occurs upon detection of a START condition.
  - To ensure that the I<sup>2</sup>C bus interface operation correctly executes its operation, always clear IBCR0n:WUE to "0" after the MCU wakes up from stop mode or watch mode, regardless of whether the MCU has been woken up by to the I<sup>2</sup>C wakeup function or the wakeup function of another resource (such as an external interrupt).

# **CHAPTER 22**

---

# **EXAMPLE OF SERIAL PROGRAMMING CONNECTION**

**This chapter describes the example of serial programming connection.**

- 22.1 Basic Configuration of Serial Programming Connection
- 22.2 Example of Serial Programming Connection

## 22.1 Basic Configuration of Serial Programming Connection

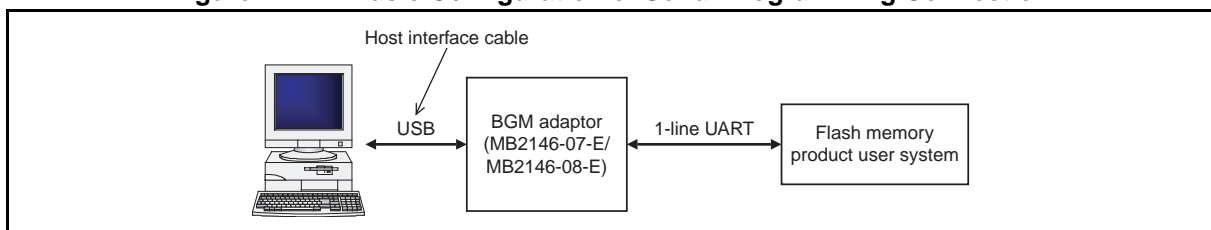
The MB95850K/860K/870K Series support Flash memory serial on-board programming. This section describes the configuration.

### ■ Basic Configuration of Serial Programming Connection

The BGM adaptor MB2146-07-E or MB2146-08-E, manufactured by Fujitsu Semiconductor Limited, is used for serial onboard programming.

Figure 22.1-1 shows the basic configuration of serial programming connection.

**Figure 22.1-1 Basic Configuration of Serial Programming Connection**



**Table 22.1-1 Pins Used for Fujitsu Semiconductor Standard Serial Onboard Programming**

| Pin                     | Function                                    | Details   |
|-------------------------|---|---|
| V <sub>CC</sub>         | Power supply voltage supply pin             | The programming voltage (2.4 V to 5.5 V) is supplied from the user system.  |
| V <sub>SS</sub>         | GND pin                                     | It is shared with the GND of the Flash microcontroller programmer.  |
| C                       | Decoupling capacitor connection             | Connect it to a decoupling capacitor and then to the ground.  |
| $\overline{\text{RST}}$ | Reset                                       | The $\overline{\text{RST}}$ pin is pulled up to V <sub>CC</sub> .   |
| DBG                     | 1-line UART setting serial programming mode | The DBG pin provides 1-line UART communication with the programmer. The serial programming mode is set if voltage is supplied to the DBG pin and the V <sub>CC</sub> pin at specific timings. (For the timings, see Figure 22.2-1.) |

- UART clock

The UART clock is supplied from the main CR clock.



## 22.2 Example of Serial Programming Connection

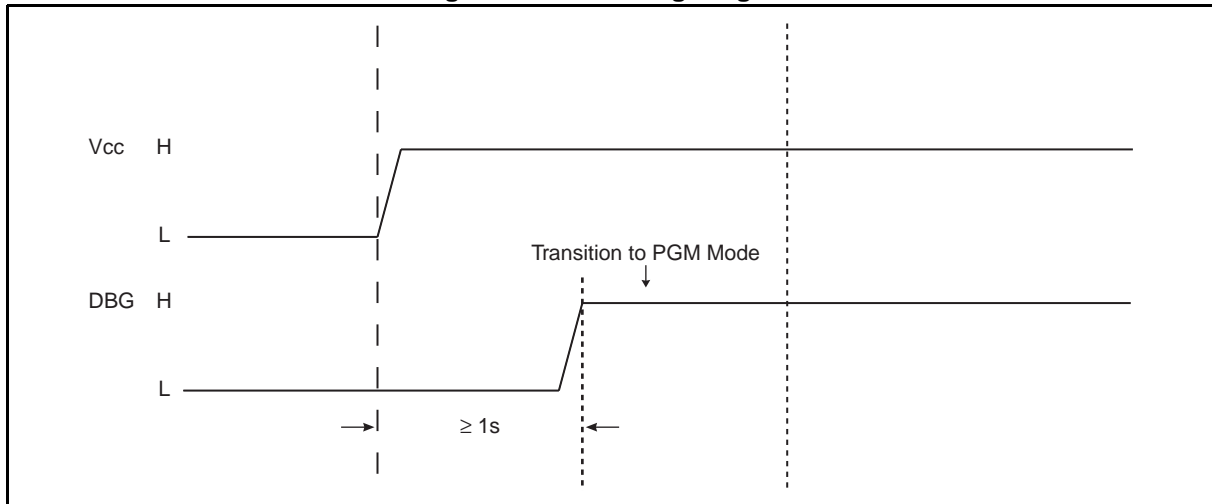
The MCU enters the PGM mode at the following timing.

### ■ MCU Transiting to PGM Mode

The MCU enters the PGM mode at the following timing.

The serial programmer controls the DBG pin according to  $V_{CC}$  input.

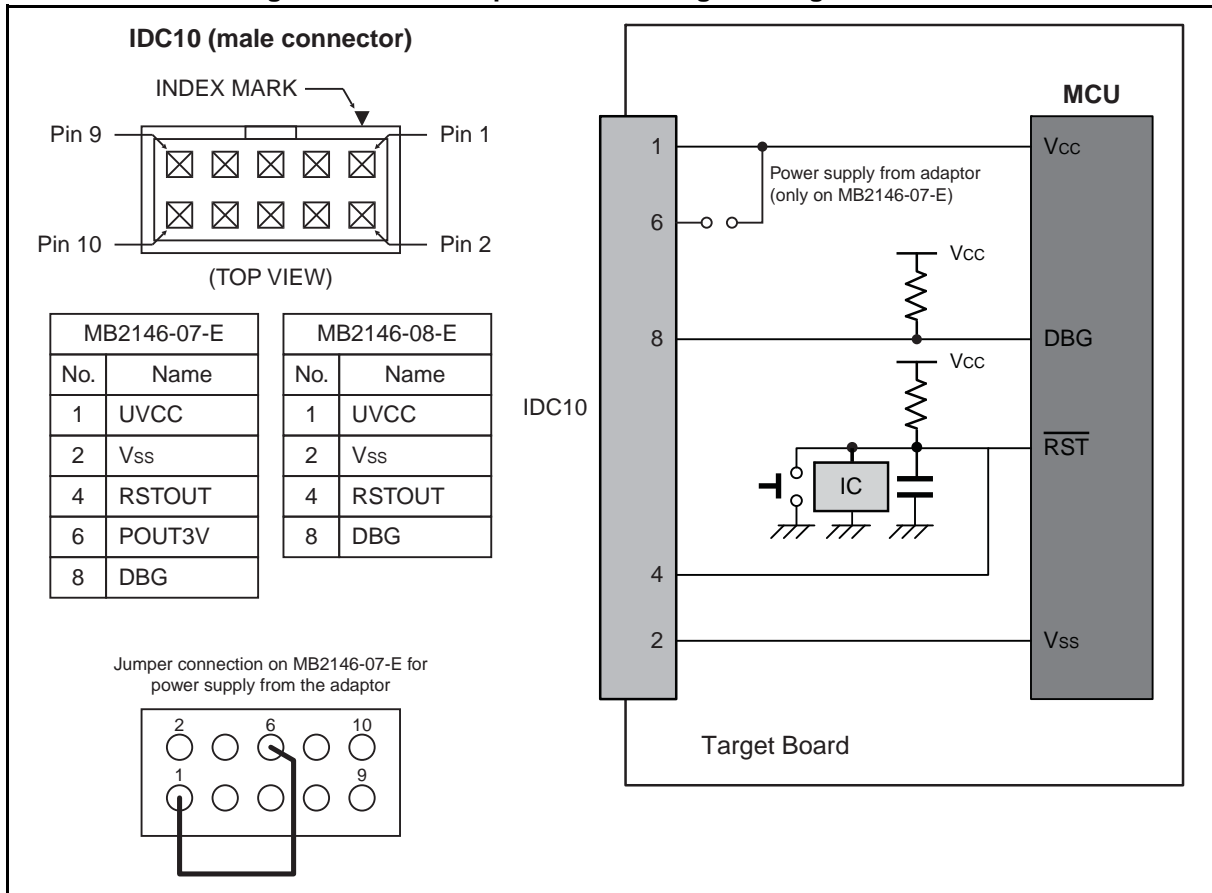
Figure 22.2-1 Timing Diagram



■ **Example of Serial Programming Connection**

Figure 22.2-2 shows an example of connection for serial programming.

**Figure 22.2-2 Example of Serial Programming Connection**



Since the pull-up resistor that can be used depends on the tool used and the interconnection length, refer to the tool document when selecting a pull-up resistor.

In the case of using the BGM adaptor MB2146-07-E of Fujitsu Semiconductor Limited, it is recommended to use a pull-up resistor of approximately 2 kΩ to 10 kΩ.

# **CHAPTER 23**

---

# ***DUAL OPERATION FLASH MEMORY***

**This chapter describes the function and operations of the 288 Kbit dual operation Flash memory.**

- 23.1 Overview
- 23.2 Sector/Bank Configuration
- 23.3 Invoking Flash Memory Automatic Algorithm
- 23.4 Checking Automatic Algorithm Execution Status
- 23.5 Programming/Erasing Flash Memory
- 23.6 Operations
- 23.7 Flash Security
- 23.8 Registers
- 23.9 Notes on Using Dual Operation Flash Memory

## 23.1 Overview

---

The dual operation Flash memory is located at 0x1000 to 0x1FFF and at 0x8000 to 0xFFFF for 288 Kbit Flash memory on the CPU memory map.

The dual operation Flash memory consists of an upper bank and a lower bank\*. Unlike conventional Flash products, programming/erasing data to/from one bank and reading data from another bank can be executed simultaneously.

\*: Upper bank: 32 Kbyte × 1; lower bank: 2 Kbyte × 2

---

### ■ Overview of Dual Operation Flash Memory

The following methods can be used to write data into and erase data from the Flash memory:

- Programming/erasing using a dedicated serial programmer
- Programming/erasing by program execution

Since data can be written into and erased from the dual operation Flash memory by instructions from the CPU via the Flash memory interface circuit, program code and data can be efficiently updated with the device mounted on a circuit board. The minimum sector size of the dual operation Flash is 2 Kbyte, which is a sector configuration facilitating the management of the program/data area.

Data can be updated by executing a program in RAM or by executing a program in the Flash memory in dual operation. The erase/program operation and the read operation can be executed in different banks (upper bank/lower bank) simultaneously.

The dual operation Flash can use the following combinations:

| Upper bank                   | Lower bank                   |
|------------------------------|------------------------------|
| Read                         |                              |
| Read                         | Program/sector erase         |
| Program/sector erase         | Read                         |
| Chip erase                   |                              |
| Sector erase (erase suspend) | Program                      |
| Program                      | Sector erase (erase suspend) |

**■ Features of Dual Operation Flash Memory**

- Sector configuration
  - 36 Kbyte (32 Kbyte + 2 Kbyte × 2)
- Two-bank configuration, enabling simultaneous execution of a program/erase operation and a read operation
- Automatic algorithm (Embedded Algorithm)
- Erase-suspend/erase-resume functions integrated
- Detecting the completion of programming/erasing using the data polling flag or the toggle bit
- Detecting the completion of programming/erasing by CPU interrupts
- Capable of erasing data in specific sectors (any combination of sectors)
- Compatible with JEDEC standard commands
- Number of program/erase cycles (minimum): 100000
- Flash read cycle time (minimum): 1 machine cycle

**■ Programming and Erasing Flash Memory**

- Programming data to and reading data from the same bank of the Flash memory cannot be executed simultaneously.
- To program data to or erase data from a bank in the Flash memory, copy the program for programming/erasing either to another bank or to the RAM first, and then execute the program.
- The dual operation Flash memory enables programming in the Flash memory and controlling programming by using interrupts. In addition, it is not necessary to download a program to RAM in order to program data to a bank, thereby reducing the time of program download and eliminating the need to protecting RAM data against power interruption.

## 23.2 Sector/Bank Configuration

This section shows the sector/bank configuration of the Flash memory.

### ■ Sector/Bank Configuration of Dual Operation Flash Memory

Figure 23.2-1 shows the sector configuration of the dual operation Flash memory. The upper and lower addresses of each sector are shown in the figure.

- Bank configuration

The lower bank of the Flash memory is SA0 and SA1 and the upper bank SA2.

**Figure 23.2-1 Sector/Bank Configuration of Dual Operation Flash Memory**

| Flash memory (36 Kbyte) |            | CPU address |
|-------------------------|------------|-------------|
| SA0: 2 Kbyte            | Lower bank | 0x1000      |
| SA1: 2 Kbyte            |            | 0x17FF      |
|                         |            | 0x1800      |
|                         |            | 0x1FFF      |
|                         |            | 0x2000      |
|                         |            |             |
|                         |            | 0x7FFF      |
|                         |            | 0x8000      |
| SA2: 32 Kbyte           | Upper bank |             |
|                         |            | 0xFFFF      |

## 23.3 Invoking Flash Memory Automatic Algorithm

There are four commands that invoke the Flash memory automatic algorithm: read/reset, program, chip erase, and sector erase. The sector erase command is capable of suspending and resuming sector erase.

### ■ Command Sequence Table

Table 23.3-1 lists commands used in programming/erasing Flash memory.

**Table 23.3-1 Command Sequence**

| Command sequence      | Bus write cycle   | 1st bus write cycle |      | 2nd bus write cycle |      | 3rd bus write cycle |      | 4th bus write cycle |      | 5th bus write cycle |      | 6th bus write cycle |      |
|-----------------------|---|---------------------|------|---------------------|------|---------------------|------|---------------------|------|---------------------|------|---------------------|------|
|                       |   | Address             | Data | Address             | Data | Address             | Data | Address             | Data | Address             | Data | Address             | Data |
| Read/reset            | 1   | 0xUXXX              | 0xF0 | -                   | -    | -                   | -    | -                   | -    | -                   | -    | -                   | -    |
| Program               | 4   | 0xUAAA              | 0xAA | 0xU554              | 0x55 | 0xUAAA              | 0xA0 | PA                  | PD   | -                   | -    | -                   | -    |
| Chip erase            | 6   | 0xUAAA              | 0xAA | 0xU554              | 0x55 | 0xUAAA              | 0x80 | 0xUAAA              | 0xAA | 0xU554              | 0x55 | 0xUAAA              | 0x10 |
| Sector erase          | 6   | 0xUAAA              | 0xAA | 0xU554              | 0x55 | 0xUAAA              | 0x80 | 0xUAAA              | 0xAA | 0xU554              | 0x55 | SA                  | 0x30 |
| Unlock bypass entry   | 3   | 0xUAAA              | 0xAA | 0xU554              | 0x55 | 0xUAAA              | 0x20 | -                   | -    | -                   | -    | -                   | -    |
| Unlock bypass program | 2   | 0xUXXX              | 0xA0 | PA                  | PD   | -                   | -    | -                   | -    | -                   | -    | -                   | -    |
| Unlock bypass reset   | 2   | 0xUXXX              | 0x90 | 0xUXXX              | any  | -                   | -    | -                   | -    | -                   | -    | -                   | -    |
| Sector erase suspend  | Programming data "0xB0" to the address "0xUXXX" suspends erasing during sector erase. |                     |      |                     |      |                     |      |                     |      |                     |      |                     |      |
| Sector erase resume   | Programming data "0x30" to the address "0xUXXX" resumes suspended sector erase.       |                     |      |                     |      |                     |      |                     |      |                     |      |                     |      |
| Erase sector add      | Programming data "0x30" to the SA adds a new sector to be erased.                     |                     |      |                     |      |                     |      |                     |      |                     |      |                     |      |

PA : Program address

SA : Sector address (Specify any address in a sector.)

PD : Program data

U : The upper four bits represent an address in a sector to which data can be programmed.

X : Any address value

any : Any program data

---

**Notes:**

- Addresses in Table 23.3-1 are values on the CPU memory map. All addresses and data are in hexadecimal notation. However, "X" is an arbitrary value.
  - "U" in an address in Table 23.3-1 is not arbitrary, but represents the upper four bits (bit15 to bit12) of an address.
  - The chip erase command is accepted only when programming data into all sectors has been enabled. The chip erase command is ignored if the bit for any sector in the flash memory sector write control register 0 (SWRE0) has been set to "0" (to disable programming data to that sector).
- 

**■ Note on Issuing Commands**

Do the following when issuing commands in command sequence table:

- Enable programming data into a required sector before issuing the first command.



## 23.4 Checking Automatic Algorithm Execution Status

Since the Flash memory uses the automatic algorithm to execute the program/erase flow, its internal operating status can be checked through the hardware sequence flags.

### ■ Hardware Sequence Flags

#### ● Overview of hardware sequence flags

The hardware sequence flag consists of the following 5-bit output:

- Data polling flag (DQ7)
- Toggle bit flag (DQ6)
- Execution timeout flag (DQ5)
- Sector erase timer flag (DQ3)
- Toggle bit2 flag (DQ2)

The hardware sequence flags can tell whether a program command, a chip erase command or a sector erase command has been terminated, whether an erase code can be written and whether an erase sector or a non-erase sector is being read.

The value of a hardware sequence flag can be checked by a read access to the address of a target sector in the Flash memory after a command sequence is set. Note that a hardware sequence flag is output only to the bank from which a command has been issued.

Table 23.4-1 shows the bit allocation of the hardware sequence flags.

**Table 23.4-1 Bit Allocation of Hardware Sequence Flag**

| Bit no.                | 7   | 6   | 5   | 4 | 3   | 2   | 1 | 0 |
|------------------------|-----|-----|-----|---|-----|-----|---|---|
| Hardware sequence flag | DQ7 | DQ6 | DQ5 | - | DQ3 | DQ2 | - | - |

- To decide whether a program command, a chip erase command or a sector erase command is being executed or has been terminated, check the respective hardware sequence flags or the flash memory program/erase status bit in the flash memory status register (FSR:RDY). After programming/erasing is terminated, the Flash memory returns to the read/reset state.
- When creating a program/erase program, read data after confirming the termination of programming/erasing using the DQ2, DQ3, DQ5, DQ6 and DQ7 flags.
- The hardware sequence flags can also be used to check whether the second sector erase code write and those to be executed afterward are valid or not.

- Description of hardware sequence flags

Table 23.4-2 lists the functions of the hardware sequence flags.

**Table 23.4-2 List of Hardware Sequence Flag Functions**

| State                                    |   | DQ7                                     | DQ6                 | DQ5            | DQ3            | DQ2            |
|--|---|---|---------------------|----------------|----------------|----------------|
| State transition during normal operation | Programming → Programming completed (when program address has been specified) | $\overline{DQ7} \rightarrow$<br>DATA: 7 | Toggle →<br>DATA: 6 | 0 →<br>DATA: 5 | 0 →<br>DATA: 3 | 0 →<br>DATA: 2 |
|  | Chip/sector erase → Erase completed   | 0 → 1                                   | Toggle → 1          | 0 → 1          | 1              | Toggle → 1     |
|  | Sector erase wait → Erase started   | 0                                       | Toggle              | 0              | 0 → 1          | Toggle         |
|  | Erasing → sector erase suspended (Sector being erased)                        | 0                                       | Toggle → 0          | 0              | 1              | Toggle         |
|  | Sector erase suspended → Erasing resumed (Sector being erased)                | 0                                       | 0 → Toggle          | 0              | 1              | Toggle         |
|  | Sector erase being suspended (Sector not being erased)                        | DATA: 7                                 | DATA: 6             | DATA: 5        | DATA: 3        | DATA: 2        |
| Abnormal operation                       | Programming   | $\overline{DQ7}$                        | Toggle              | 1              | 0              | 0              |
|  | Chip/sector erase   | 0                                       | Toggle              | 1              | 1              | Toggle         |

## 23.4.1 Data Polling Flag (DQ7)

The data polling flag (DQ7) is a hardware sequence flag indicating that the automatic algorithm is being executing or has been completed using the data polling function.

### ■ Data Polling Flag (DQ7)

Table 23.4-3 and Table 23.4-4 show the state transition of the data polling flag during normal operation and the one during abnormal operation respectively.

**Table 23.4-3 State Transition of Data Polling Flag (During Normal Operation)**

| Operating state | Programming → Programming completed         | Chip/sector erase → Erasing completed | Sector erase wait → Erasing started | Sector erase → Sector erase suspended (Sector being erased) | Sector erase suspended → Erasing resumed (Sector being erased) | Sector erase being suspended (Sector not being erased) |
|-----------------|---|---------------------------------------|-------------------------------------|---|--|--|
| DQ7             | $\overline{DQ7} \rightarrow \text{DATA: 7}$ | 0 → 1                                 | 0                                   | 0   | 0  | DATA: 7  |

**Table 23.4-4 State Transition of Data Polling Flag (During Abnormal Operation)**

| Operating state | Programming      | Chip/sector erase |
|-----------------|------------------|-------------------|
| DQ7             | $\overline{DQ7}$ | 0                 |

- At programming

When read access takes place during execution of the automatic write algorithm, the Flash memory outputs the inverted value of bit7 in the last data written to DQ7.

If read access takes place on completion of the automatic write algorithm, the Flash memory outputs bit7 of the value read from the read-accessed address to DQ7.

- At chip/sector erase

When read access is made to the sector currently being erased during execution of the chip/sector erase algorithm, bit7 of Flash memory outputs "0". Bit7 of Flash memory outputs "1" upon completion of chip/sector erase.

- At sector erase suspension

- When read access takes place with a sector erase operation suspended, the Flash memory outputs "0" to DQ7 if the read address is the sector being erased. If not, the Flash memory outputs bit7 (DATA:7) of the value read from the read address to DQ7.
- Referring the data polling flag (DQ7) together with the toggle bit flag (DQ6) permits a decision on whether Flash memory is in the sector erase suspended state or which sector is being erased.

Note:

Once the automatic algorithm has been started, read access to the specified address is ignored. Data reading is allowed after the data polling flag (DQ7) is set to "1". Data reading after the end of the automatic algorithm should be performed following read access made to confirm the completion of data polling.

---

## 23.4.2 Toggle Bit Flag (DQ6)

The toggle bit flag (DQ6) is a hardware sequence flag using the toggle bit function to indicate whether the automatic algorithm is being executed or has terminated.

### ■ Toggle Bit Flag (DQ6)

Table 23.4-5 and Table 23.4-6 show the state transition of the toggle bit flag during normal operation and the one during abnormal operation respectively.

**Table 23.4-5 State Transition of Toggle Bit Flag (During Normal Operation)**

| Operating state | Programming → Programming completed | Chip/sector erase → Erasing completed | Sector erase wait → Erasing started | Sector erase → Sector erase suspended (Sector being erased) | Sector erase suspended → Erasing resumed (Sector being erased) | Sector erase being suspended (Sector not being erased) |
|-----------------|-------------------------------------|---------------------------------------|-------------------------------------|---|--|--|
| DQ6             | Toggle → DATA: 6                    | Toggle → 1                            | Toggle                              | Toggle → 0  | 0 → Toggle   | DATA: 6  |

**Table 23.4-6 State Transition of Toggle Bit Flag (During Abnormal Operation)**

| Operating state | Programming | Chip/sector erase |
|-----------------|-------------|-------------------|
| DQ6             | Toggle      | Toggle            |

- At programming and chip/sector erase

- When read accesses are made continuously while the automatic write algorithm or the automatic chip/sector erase algorithm is being executed, the Flash memory toggles the output between "1" and "0" at each read access.
- When read accesses are made continuously after the automatic write algorithm or the chip/sector erase algorithm terminates, the Flash memory outputs bit6 (DATA:6) of the value read from the read address at each read access.

- At sector erase suspension

When a read access is made with a sector erase operation suspended, the Flash memory outputs "0" if the read address is the sector being erased. Otherwise, the Flash memory outputs bit6 (DATA: 6) of the value read from the read address.

**Note:**

When using dual-operation Flash memory (Flash memory write control program is executed on the Flash memory), the toggle bit flag (DQ6) cannot be used to check the operating state of programming/erasing. See the notes in "23.9 Notes on Using Dual Operation Flash Memory" when writing a program.

The note above does not apply if the Flash memory write control program is executed on the RAM.

### 23.4.3 Execution Timeout Flag (DQ5)

The execution timeout flag (DQ5) is a hardware sequence flag indicating that the execution time of the automatic algorithm exceeds a specified time (required for programming/erasing) in the Flash memory.

#### ■ Execution Timeout Flag (DQ5)

Table 23.4-7 and Table 23.4-8 show the state transition of the execution timeout flag during normal operation and the one during abnormal operation respectively.

**Table 23.4-7 State Transition of Execution Timeout Flag (During Normal Operation)**

| Operating state | Programming → Programming completed | Chip/sector erase → Erasing completed | Sector erase wait → Erasing started | Sector erase → Sector erase suspended (Sector being erased) | Sector erase suspended → Erasing resumed (Sector being erased) | Sector erase being suspended (Sector not being erased) |
|-----------------|-------------------------------------|---------------------------------------|-------------------------------------|---|--|--|
| DQ5             | 0 → DATA: 5                         | 0 → 1                                 | 0                                   | 0   | 0  | DATA: 5  |

**Table 23.4-8 State Transition of Execution Timeout Flag (During Abnormal Operation)**

| Operating state | Programming | Chip/sector erase |
|-----------------|-------------|-------------------|
| DQ5             | 1           | 1                 |

- At programming and chip/sector erase

When a read access is made with the automatic write algorithm or the automatic chip/sector erase algorithm invoked, the flag outputs "0" when the algorithm execution time is within the specified time (required for programming/erasing) or "1" when it exceeds that time.

The execution timeout flag (DQ5) can be used to check whether programming/erasing has succeeded or failed regardless of whether the automatic algorithm has been running or terminated. When the execution timeout flag (DQ5) outputs "1", it can be judged that programming fails if flash memory program/erase status bit (RDY) in the flash memory status register (FSR) is "0".

If an attempt is made to write "1" to a Flash memory address holding "0", for example, the Flash memory is locked, the time limit is exceeded and the execution timeout flag (DQ5) outputs "1". The state in which the execution timeout flag (DQ5) outputs "1" means that the Flash memory has not been used correctly; it does not mean that the Flash memory is defective. When this state occurs, execute the reset command.

### 23.4.4 Sector Erase Timer Flag (DQ3)

The sector erase timer flag (DQ3) is a hardware sequence flag indicating whether the Flash memory is waiting for sector erase after the sector erase command has started.

#### ■ Sector Erase Timer Flag (DQ3)

Table 23.4-9 and Table 23.4-10 show the state transition of the sector erase timer flag during normal operation and the one during abnormal operation respectively.

**Table 23.4-9 State Transition of Sector Erase Timer Flag (During Normal Operation)**

| Operating state | Programming → Programming completed | Chip/sector erase → Erasing completed | Sector erase wait → Erasing started | Sector erase → Sector erase suspended (Sector being erased) | Sector erase suspended → Erasing resumed (Sector being erased) | Sector erase being suspended (Sector not being erased) |
|-----------------|-------------------------------------|---------------------------------------|-------------------------------------|---|--|--|
| DQ3             | 0 → DATA: 3                         | 1                                     | 0 → 1                               | 1   | 1  | DATA: 3  |

**Table 23.4-10 State Transition of Sector Erase Timer Flag (During Abnormal Operation)**

| Operating state | Programming | Chip/sector erase |
|-----------------|-------------|-------------------|
| DQ3             | 0           | 1                 |

- At sector erase

- When a read access is made after the sector erase command has started, the sector erase timer flag (DQ3) outputs "0" within the sector erase wait period. The flag outputs "1" if the sector erase wait period has elapsed.
- When the data polling function or the toggle bit function indicates that the erase algorithm is being executed (DQ7 = 0, DQ6: toggle output), the Flash memory executes sector erase. If the command subsequently set is not a sector erase suspend command, it is ignored until sector erase is terminated.
- If the sector erase timer flag (DQ3) is "0", the Flash memory can accept the sector erase command. Before writing the sector erase command to the Flash memory, make sure that the sector erase timer flag (DQ3) is "0". If the flag is "1", the Flash memory may not accept suspending the sector erase command.

- At sector erase suspension

When a read access is made with the sector erase operation suspended, the Flash memory outputs "1" if the read address of that read access is the address of a sector being erased. If the read address is not the address of a sector being erased, the Flash memory outputs bit3 (DATA: 3) of the value read from the read address.

### 23.4.5 Toggle Bit2 Flag (DQ2)

The toggle bit2 flag (DQ2) is a hardware sequence flag using the toggle bit function to indicate whether a read address is an erase target sector in the sector erase suspend state and whether output data is toggled.

#### ■ Toggle Bit2 Flag (DQ2)

Table 23.4-11 and Table 23.4-12 show the state transition of the toggle bit2 flag during normal operation and the one during abnormal operation respectively.

**Table 23.4-11 State Transition of Toggle Bit2 Flag (During Normal Operation)**

| Operating state | Programming → Programming completed | Chip/sector erase → Erasing completed | Sector erase wait → Erasing started | Sector erase → Sector erase suspended (Sector being erased) | Sector erase suspended → Erasing resumed (Sector being erased) | Sector erase being suspended (Sector not being erased) |
|-----------------|-------------------------------------|---------------------------------------|-------------------------------------|---|--|--|
| DQ2             | 0 → DATA: 2                         | Toggle → 1                            | Toggle                              | Toggle  | Toggle   | DATA: 2  |

**Table 23.4-12 State Transition of Toggle Bit2 Flag (During Abnormal Operation)**

| Operating state | Programming | Chip/sector erase |
|-----------------|-------------|-------------------|
| DQ2             | 0           | Toggle            |

#### ● At chip/sector erase

- When read accesses are continuously made to a sector to be erased while the automatic chip/sector erase algorithm is being executed, the Flash memory toggles the output between "1" and "0" at each read access.
- When read accesses are continuously made to a sector not to be erased while the automatic chip/sector erase algorithm is being executed, the Flash memory outputs bit2 (DATA: 2) of the value read from a read address of each read access.

#### ● At sector erase suspension

- With a sector erase operation suspended, when read accesses are continuously made to a sector to be erased, the Flash memory toggles the output between "1" and "0" whenever a read access is made.
- With a sector erase operation suspended, when read accesses are continuously made to a sector not to be erased, the Flash memory outputs bit2 (DATA: 2) of the read value of a read address whenever a read access is made.



## 23.5 Programming/Erasing Flash Memory

---

**This section describes the respective procedures for reading/resetting the Flash memory, programming, chip-erasing, sector-erasing, sector erase suspending and sector erase resuming by entering respective commands to invoke the automatic algorithm.**

---

### ■ Details of Programming/Erasing Flash Memory

The automatic algorithm can be invoked by programming the read/reset, program, chip erase, sector erase, sector-erase suspend, and sector erase resume command sequence to the Flash memory from the CPU. Always write the commands of a command sequence continuously from the CPU to the Flash memory. The termination of the automatic algorithm can be checked by the data polling function. After the automatic algorithm terminates normally, the Flash memory returns to the read/reset state.

The operations are explained in the following order:

- Enter the read/reset state
- Program data
- Erase all data (chip erase)
- Erase arbitrary data (sector erase)
- Suspend sector erase
- Resume sector erase
- Unlock bypass program

## 23.5.1 Placing Flash Memory in Read/Reset State

---

This section explains the procedure for entering the read/reset command to place the Flash memory in read/reset state.

---

### ■ Placing Flash Memory in Read/Reset State

- To place the Flash memory in the read/reset state, send read/reset commands in the command sequence table from the CPU to the Flash memory.
- Since the read/reset state is the initial state of the Flash memory, the Flash memory always enters this state after power-on or the normal termination of a command. The read/reset state is also regarded as the command input wait state.
- In the read/reset state, data in the Flash memory can be read by a read access to the Flash memory.
- In the case of a read access to the Flash memory, no read/reset commands are required. If a command does not terminate normally, use a read/reset command to initialize the automatic algorithm.

## 23.5.2 Programming Data to Flash Memory

---

This section explains the procedure for entering the program command to program data to the Flash memory.

---

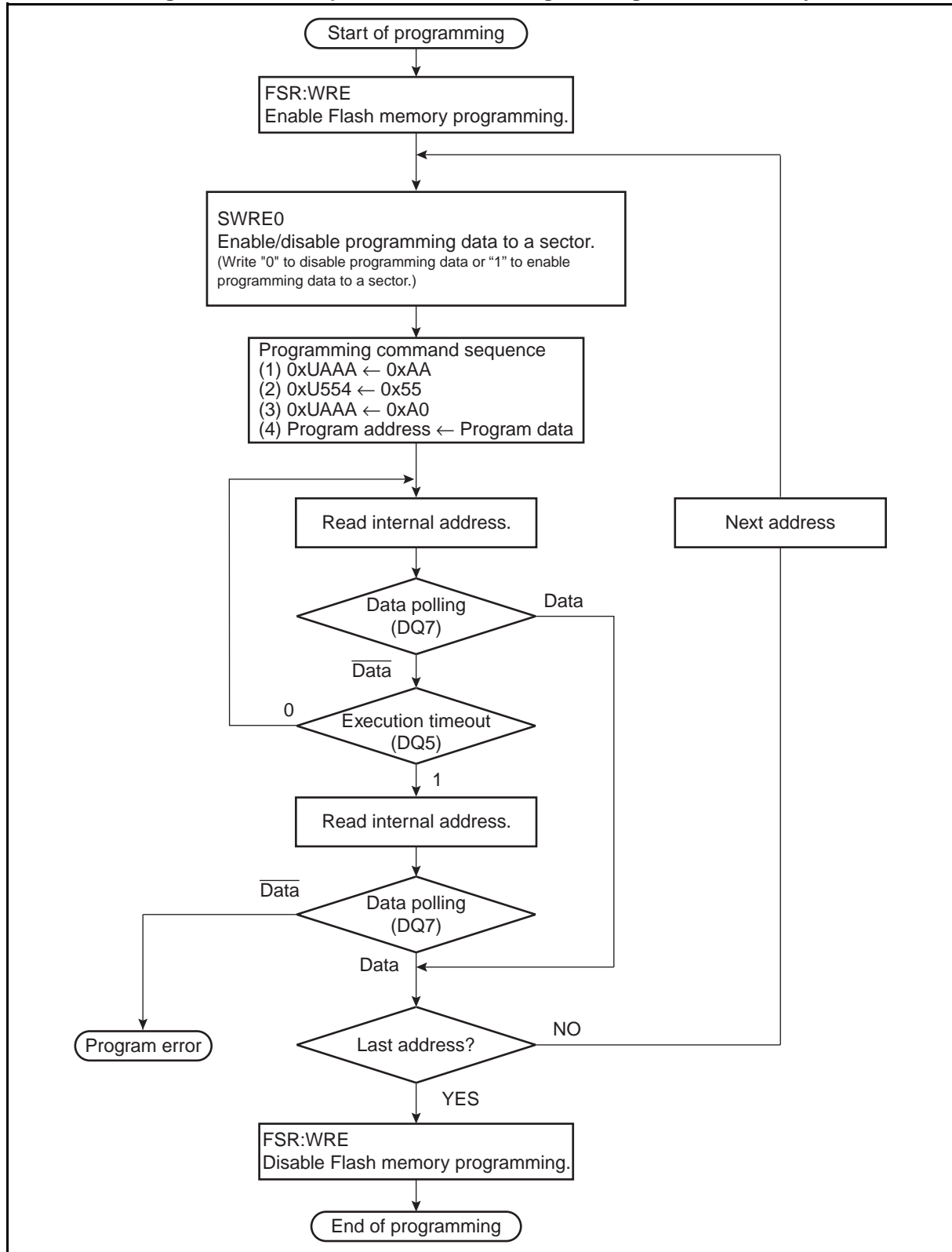
### ■ Programming Data to Flash Memory

- To invoke the automatic algorithm for programming data to the Flash memory, send program commands in the command sequence table consecutively from the CPU to the Flash memory.
  - When data is programmed to a target address in the fourth cycle, the automatic algorithm is invoked and starts automatic programming.
- Addressing method
    - Programming can be performed in any order of addresses and across a sector boundary. The size of data that can be written by a single program command is one byte only.
  - Note on programming data
    - Bit data cannot be returned from "0" to "1" by programming. When "1" is written to bit data that is currently "0", the data polling function (DQ7) or toggle operation (DQ6) is not terminated, it is determined that Flash memory component is defective, and the execution timeout flag (DQ5) indicates that an error has occurred because the execution time of the automatic algorithm exceeds the programming time specified. When data is read in the read/reset state, the bit data remains "0". To make the bit data return from "0" to "1", erase the Flash memory.
    - All commands are ignored during programming.
    - During programming, if a hardware reset occurs, the integrity of data being written to the current address is not guaranteed. Start programming the data from the chip erase command or the sector erase command again.

### ■ Flash Memory Programming Procedure

- Figure 23.5-1 gives an example of the procedure for programming data to the Flash memory. The hardware sequence flag can be used to check the operating state of the automatic algorithm in the Flash memory. The data polling flag (DQ7) is used for checking the end of programming data into Flash memory in this example.
- Data for flag checking is read from the address to which data has been last written.
- Since the data polling flag (DQ7) and the execution timeout flag (DQ5) are changed simultaneously, check the data polling flag (DQ7) even when the execution timeout flag (DQ5) is "1".
- Similarly, since the toggle bit flag (DQ6) stops toggling at the same time as the execution timeout flag (DQ5) changes to "1", check DQ6 after DQ5 changes to "1".

**Figure 23.5-1 Sample Procedure for Programming to Flash Memory**



## MB95850K/860K/870K Series

### 23.5.3 Erasing All Data from Flash Memory (Chip Erase)

---

This section explains the procedure for issuing the chip erase command to erase all data in the Flash memory.

---

#### ■ Erasing Data from Flash Memory (Chip Erase)

- To erase all data from the Flash memory, send the chip erase command mentioned in the command sequence table continuously from the CPU to the Flash memory.
- The chip erase command is executed in six bus operations. Chip erasing starts at the point when the sixth cycle of programming commands is complete.
- In chip erase, the user does not need to program data to the Flash memory before starting erasing data. While the automatic erase algorithm is running, it automatically writes "0" to all cells in the Flash memory before erasing data.

#### ■ Note on Chip Erase

- The chip erase command is accepted only when programming data to all sectors has been enabled. The chip erase command is ignored even if only one bit for a sector in the flash memory sector write control register 0 (SWRE0) has been set to "0" (to disable programming data to that sector).
- During chip erase, if a hardware reset occurs, the integrity of data in the Flash memory is not guaranteed.

## 23.5.4 Erasing Specific Data from Flash Memory (Sector Erase)

---

**This section explains the procedure for entering the sector erase command to erase a specific sector in the Flash memory. Sector-by-sector erase is enabled and multiple sectors can also be specified at a time.**

---

### ■ Erasing Specific Data from Flash Memory (Sector Erase)

To erase data from a specific sector in the Flash memory, send the sector erase command mentioned in the command sequence table continuously from the CPU to the Flash memory.

#### ● Specifying a sector

- The sector erase command is executed in six bus operations. A minimum of 35  $\mu$ s sector erase wait time starts as an address in the sector to be erased is specified as the address for the sixth cycle and the sector erase code (0x30) is written as data.
- To erase data from multiple sectors, write the erase code (0x30) to an address in sector to be erased after programming the sector erase code to the address of the first sector to be erased as explained above.

#### ● Note on specifying multiple sectors

- Sector erase starts as a minimum of 35  $\mu$ s sector erase wait time elapses after the last sector erase code has been written.
- To erase data from multiple sectors simultaneously, input the addresses of sectors to be erased and the erase code (in the sixth cycle of the command sequence) within 35  $\mu$ s. If the erase code is input after 35  $\mu$ s elapses, it will not be accepted due to the end of the sector erase wait time.
- The sector erase timer flag (DQ3) can be used to check whether it is valid to write sector erase codes continuously.
- Specify the address of a sector to be erased as the address at which the sector erase timer flag (DQ3) is read.

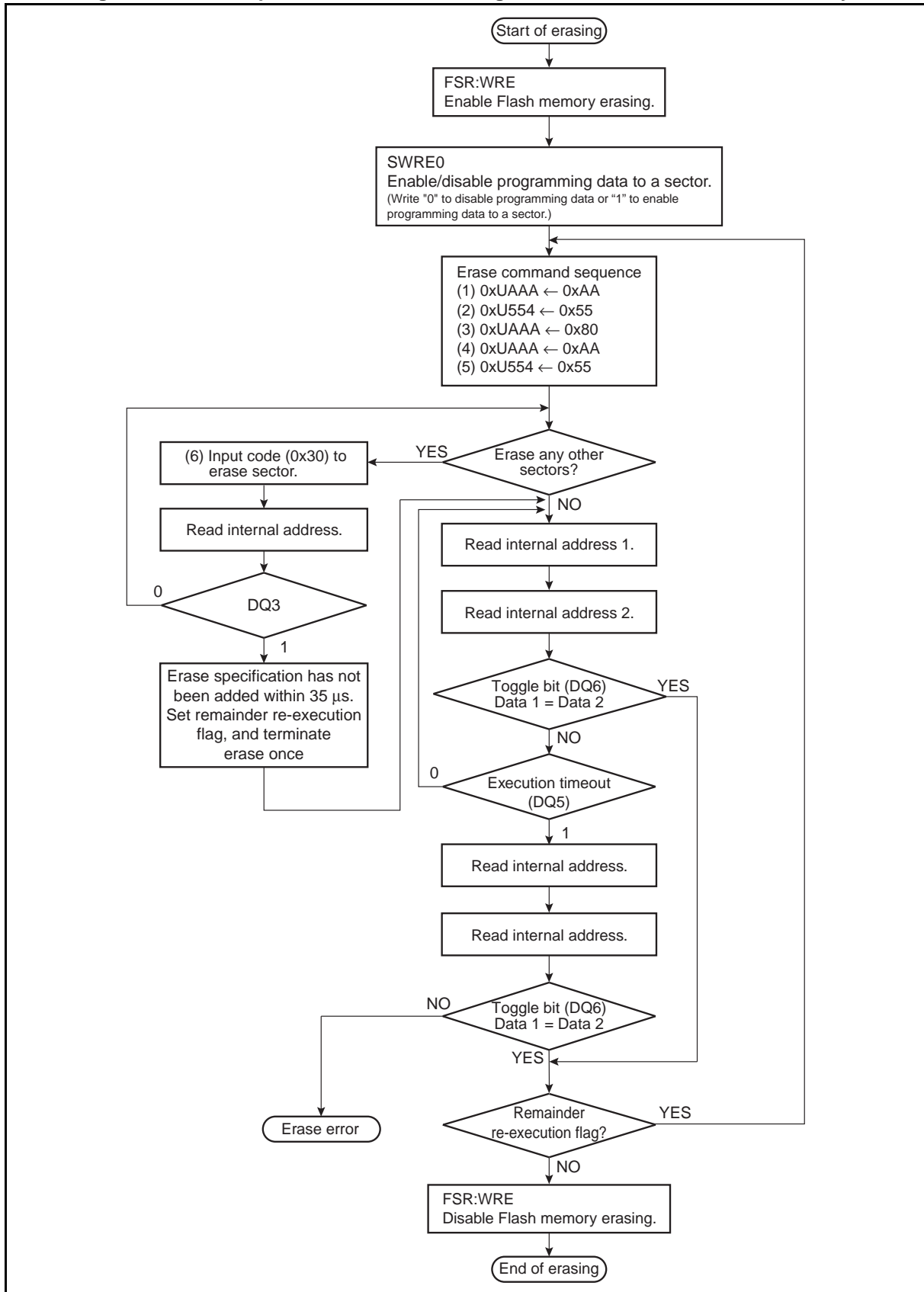
### ■ Flash Memory Sector Erase Procedure

- Hardware sequence flags can be used to check the state of the automatic algorithm in the Flash memory. Figure 23.5-2 gives an example of the Flash memory sector erase procedure. In this example, the toggle bit flag (DQ6) is used to check the end of sector erase.
- The toggle bit flag (DQ6) stops toggling the output at the same time as the execution timeout flag (DQ5) changes to "1". Check the toggle bit flag (DQ6) even when the execution timeout flag (DQ5) is "1".
- Since the data polling flag (DQ7) and the execution timeout flag (DQ5) are changed simultaneously, check the data polling flag (DQ7).

### ■ Note on Erasing Data from Sectors

If a hardware reset occurs while data is being erased, the integrity of data in the Flash memory is not guaranteed. Therefore, run the sector erase procedure again after a hardware reset occurs.

Figure 23.5-2 Sample Procedure for Erasing Data from Sectors in Flash Memory



## **23.5.5 Suspending Sector Erase from Flash Memory**

---

**This section explains the procedure for entering the sector erase suspend command to suspend sector erase from the Flash memory. Data can be read from sectors not being erased.**

---

### **■ Suspending Sector Erase from Flash Memory**

- To suspend the Flash memory sector erase, send the sector erase suspend command mentioned in the command sequence table from the CPU to the Flash memory.
- The sector erase suspend command suspends the current sector erase operation, allowing data to be read from sectors that are not being erased.
- The sector erase suspend command is only enabled during the sector erase period including the erase wait time; it is ignored in chip erasing or programming.
- The sector erase suspend command is executed when the sector erase suspend code (0xB0) is written. Specify an address in the sector selected to be erased. If an attempt is made to execute the sector erase suspend command again when sector erase has been suspended, the new sector erase suspend command input is ignored.
- When a sector erase suspend command is input during the sector erase wait period, the sector erase wait time ends immediately, the sector erase operation is stopped, and the Flash memory enters the erase stop state.
- When a sector erase suspend command is input during sector erase after the sector erase wait period, the erase suspend state occurs after a maximum of 35  $\mu$ s has elapsed since the issue of the sector erase suspend command.

---

**Note:**

To suspend sector erase by issuing a sector erase suspend command, issue the command after 35  $\mu$ s + 2 MCLK (machine clock) or longer has elapsed since the issue of a sector erase command or a sector erase resume command.

To suspend sector erase command again after resuming sector erase by issuing a sector erase resume command, issue the command after 2 ms or longer has elapsed since the issue of the sector erase resume command.

---



## MB95850K/860K/870K Series

### 23.5.6 Resuming Sector Erase of Flash Memory

---

This section explains the procedure for entering the sector erase resume command to resume suspended erasing of a sector in the Flash memory.

---

#### ■ Resuming Sector Erase of Flash Memory

- To resume suspended sector erase, send the sector erase resume command mentioned in the command sequence table from the CPU to the Flash memory.
- The sector erase resume command resumes a sector erase operation suspended by the sector erase suspend command. The sector erase resume command is executed by writing erase resume code (0x30). Specify an address in the sector selected to be erased.
- A sector erase resume command input during sector erase is ignored.

## **23.5.7 Unlock Bypass Program**

---

**This sections explains details of the unlock bypass state.**

---

### **■ Transiting from Normal Command State to Unlock Bypass State**

If an unlock bypass program command is input in the normal command state, the Flash memory will transit to the unlock bypass state. In this state, a program command can be executed if the command is input within two cycles as mentioned in Table 23.3-1.

### **■ Returning from Unlock Bypass State to Normal Command State**

If an unlock bypass reset command is input in the unlock bypass state, the Flash memory will return to the normal command state from the unlock bypass state. In addition, executing a hardware reset in the unlock bypass state will also make Flash memory return to the normal command state.

## 23.6 Operations

Pay attention in particular to the following points when using dual operation Flash memory:

- Interrupt generated when upper banks are updated
- Procedure of setting the sector swap enable bit in the flash memory status register (FSR:SSEN)

### ■ Interrupt Generated When Upper Banks Are Updated

The dual operation Flash memory consists of two banks. Like conventional Flash products, however, it cannot be erased/programmed and read at the same time in banks on the same side.

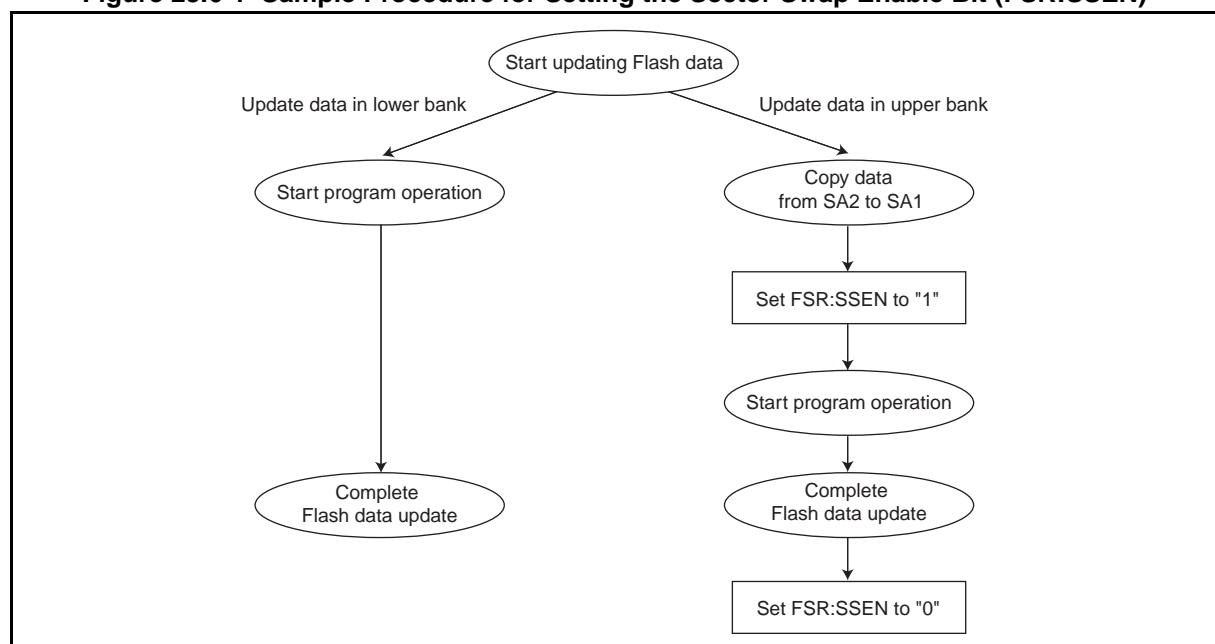
As SA2 contains an interrupt vector, an interrupt vector from the CPU cannot be read normally when an interrupt occurs during programming data to an upper bank. Before an upper bank can be updated, the sector swap enable bit must be set to "1" (FSR:SSEN = 1). When an interrupt occurs, therefore, SA1 is accessed to read interrupt vector data. The same data must be copied to SA1 and SA2 before the sector swap enable bit (FSR:SSEN) is set.

### ■ Procedure for Setting Sector Swap Enable Bit (FSR:SSEN)

Figure 23.6-1 shows a sample procedure of setting the sector swap enable bit (FSR:SSEN).

To modify data in the upper bank, it is necessary to set FSR:SSEN to "1". While data is being written to the Flash memory, modifying the setting of FSR:SSEN is prohibited. The setting of FSR:SSEN can only be modified before the start of programming data to the Flash memory or after the completion of programming data to the Flash memory. In addition, control the Flash memory interrupts while setting FSR:SSEN as follows: before setting FSR:SSEN, disable the Flash memory interrupts; after setting FSR:SSEN, enable the interrupts.

Figure 23.6-1 Sample Procedure for Setting the Sector Swap Enable Bit (FSR:SSEN)



■ **Operation during Programming/Erasing**

It is prohibited to program data to the Flash memory within an interrupt routine when an interrupt occurs during Flash memory programming/erasing.

When two or more program/erase routines exist, wait for one program/erase routine to finish before executing another program/erase routine.

While data is being written to or erased from the Flash memory, state transition in the current mode (clock mode or standby mode) is prohibited. Ensure that programming data to or erasing data from the Flash memory ends before state transition occurs.

## 23.7 Flash Security

---

**The Flash security controller function prevents contents of the Flash memory from being read by external pins.**

---

### ■ Flash Security

Writing protection code "0x01" to the Flash memory address (0xFFFC) restricts access to the Flash memory, disabling any read/write access to the Flash memory from any external pin. Once the protection of the Flash memory is enabled, the function cannot be unlocked until a chip erase command operation is executed.

It is advisable to write the protection code at the end of Flash programming to avoid enabling unnecessary protection during writing.

Once Flash security is enabled, a chip erase operation must be executed before data can be written to the Flash memory again.

## 23.8 Registers

---

**This section describes the registers for the dual operation Flash memory.**

---

**Table 23.8-1 List of Dual Operation Flash Memory Registers**

| Register abbreviation | Register name                                | Reference |
|-----------------------|--|-----------|
| FSR2                  | Flash memory status register 2               | 23.8.1    |
| FSR                   | Flash memory status register                 | 23.8.2    |
| SWRE0                 | Flash memory sector write control register 0 | 23.8.3    |
| FSR3                  | Flash memory status register 3               | 23.8.4    |
| FSR4                  | Flash memory status register 4               | 23.8.5    |

## 23.8.1 Flash Memory Status Register 2 (FSR2)

This section describes the Flash memory status register 2 (FSR2).

### Register Configuration

|               |       |        |       |       |       |        |       |       |
|---------------|-------|--------|-------|-------|-------|--------|-------|-------|
| bit           | 7     | 6      | 5     | 4     | 3     | 2      | 1     | 0     |
| Field         | PEIEN | PGMEND | PTIEN | PGMTO | EEIEN | ERSEND | ETIEN | ERSTO |
| Attribute     | R/W   | R/W    | R/W   | R/W   | R/W   | R/W    | R/W   | R/W   |
| Initial value | 0     | 0      | 0     | 0     | 0     | 0      | 0     | 0     |

### Register Functions

[bit7] PEIEN: PGMEND interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the completion of Flash memory programming.

| bit7               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the interrupt request upon completion of Flash memory programming (FSR2:PGMEND = 1). |
| <b>Writing "1"</b> | Enables the interrupt request upon completion of Flash memory programming (FSR2:PGMEND = 1).  |

[bit6] PGMEND: PGMEND interrupt request flag bit

This bit indicates the completion of Flash memory programming.

The PGMEND bit is set to "1" upon completion of the Flash memory automatic algorithm.

An interrupt request is generated when the PGMEND bit is set to "1", provided that generating an interrupt request upon completion of Flash memory programming has been enabled (FSR2:PEIEN = 1).

When the PGMEND bit is set to "0" after Flash memory programming is completed, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

When Flash memory programming fails (FSR3:HANG = 1), the PGMEND bit is cleared to "0".

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit6               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory programming is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory programming has been completed.  |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.  |

[bit5] PTIEN: PGMTO interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the failure of Flash memory programming.

| bit5               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the interrupt request upon failure of Flash memory programming (FSR2:PGMTO = 1). |
| <b>Writing "1"</b> | Enables the interrupt request upon failure of Flash memory programming (FSR2:PGMTO = 1).  |

[bit4] PGMTO: PGMTO interrupt request flag bit

This bit indicates that Flash memory programming has failed.

When Flash memory programming fails, the PGMTO bit is set to "1" upon completion of the Flash memory automatic algorithm. Afterward, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

An interrupt request is generated when the PGMTO bit is set to "1", provided that generating an interrupt request upon failure of Flash memory programming has been enabled (FSR2:PTIEN = 1).

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit4               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory programming is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory programming has failed.  |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.  |

[bit3] EEIEN: ERSEND interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the completion of Flash memory sector erase.

| bit3               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the interrupt request upon completion of Flash memory sector erase (FSR2:ERSEND = 1). |
| <b>Writing "1"</b> | Enables the interrupt request upon completion of Flash memory sector erase (FSR2:ERSEND = 1).  |

[bit2] ERSEND: ERSEND interrupt request flag bit

This bit indicates the completion of Flash memory sector erase.

The ERSEND bit is set to "1" upon completion of the Flash memory automatic algorithm.

An interrupt request is generated when the ERSEND bit is set to "1", provided that generating an interrupt request upon completion of Flash memory sector erase has been enabled (FSR2:EEIEN = 1).

When the ERSEND bit is set to "0" after Flash memory sector erase is completed, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

When Flash memory sector erase fails (FSR3:HANG = 1), the ERSEND bit is cleared to "0".

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit2               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory erase is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory sector erase has been completed.                                       |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.  |



[bit1] ETIEN: ERSTO interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the failure of Flash memory sector erase.

| bit1               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the interrupt request upon failure of Flash memory sector erase (FSR2:ERSTO = 1). |
| <b>Writing "1"</b> | Enables the interrupt request upon failure of Flash memory sector erase (FSR2:ERSTO = 1).  |

[bit0] ERSTO: ERSTO interrupt request flag bit

This bit indicates that Flash memory sector erase has failed.

When Flash memory sector erase fails, the ERSTO bit is set to "1" upon completion of the Flash memory automatic algorithm. Afterward, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

An interrupt request is generated when the ERSTO bit is set to "1", provided that generating an interrupt request upon failure of Flash memory sector erase has been enabled (FSR2:ETIEN = 1).

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit0               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory erase is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory sector erase has failed.   |
| <b>Writing "0"</b> | Clears this bit.   |
| <b>Writing "1"</b> | Has no effect on operation.  |

## 23.8.2 Flash Memory Status Register (FSR)

This section describes the Flash memory status register (FSR).

### Register Configuration

| bit           | 7 | 6 | 5      | 4   | 3        | 2     | 1   | 0    |
|---------------|---|---|--------|-----|----------|-------|-----|------|
| Field         | — | — | RDYIRQ | RDY | Reserved | IRQEN | WRE | SSEN |
| Attribute     | — | — | R/W    | R   | W        | R/W   | R/W | R/W  |
| Initial value | 0 | 0 | 0      | X   | 0        | 0     | 0   | 0    |

### Register Functions

#### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit5] RDYIRQ: Flash memory operation flag bit

This bit indicates the operating state of the Flash memory.

After the Flash memory programming/erasing is completed, the RDYIRQ bit is set to "1" at the point when the automatic algorithm of the Flash memory ends.

With the interrupt triggered by the completion of Flash memory programming/erasing having been enabled (FSR:IRQEN = 1), if the RDYIRQ bit is set to "1", an interrupt request occurs.

After Flash memory programming/erasing is completed, if the RDYIRQ bit has already been set to "0", further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit5               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that Flash memory programming/erasing is in progress.     |
| <b>Reading "1"</b> | Indicates that Flash memory programming/erasing has been completed. |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.   |

#### [bit4] RDY: Flash memory program/erase status bit

This bit indicates the program/erase status of the Flash memory.

When the RDY bit is "0", programming data into and erasing data from the Flash memory are disabled.

The read/reset command/sector erase suspend command can still be accepted when the RDY bit is "0". When programming or erasing ends, the RDY bit is set to "1".

After a program/erase command is issued, there is a delay of two machine clock (MCLK) cycles before the RDY bit becomes "0". After the issue of a program/erase command, wait for those two machine clock cycles to elapse (e.g. inserting NOP twice) before reading this bit.

| bit4               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that data is being programmed/erased. (Programming/erasing next data is disabled.) |
| <b>Reading "1"</b> | Indicates that data has been programmed/erased. (Programming/erasing next data is enabled.)  |

#### [bit3] Reserved bit

Always set this bit to "0".

[bit2] IRQEN: Flash memory program/erase interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the completion of Flash memory programming/erasing.

| bit2               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables generating an interrupt upon completion of Flash memory programming/erasing. |
| <b>Writing "1"</b> | Enables generating an interrupt upon completion of Flash memory programming/erasing.  |

[bit1] WRE: Flash memory program/erase enable bit

This bit enables or disables the programming/erasing of data into/from the Flash memory area.

Set the WRE bit before invoking a Flash memory program/erase command.

Writing "0" to this bit disables generating program/erase signals even when a program/erase command is input.

Writing "1" to this bit enables programming/erasing Flash memory data after a program/erase command is input.

When not programming data into or erasing data from the Flash memory, set the WRE bit to "0" in order to prevent data from being accidentally written into or erased from the Flash memory.

To program data to the Flash memory, set FSR:WRE to "1" to enable programming data to the Flash memory, and set the flash memory sector write control register 0 (SWRE0) according to the Flash memory sector into which data is to be written. When Flash memory programming is disabled (FSR:WRE = 0), no write access to a sector in the Flash memory can be executed even though it has been enabled by setting a bit corresponding to that sector in the Flash memory sector write control register 0 (SWRE0) to "1".

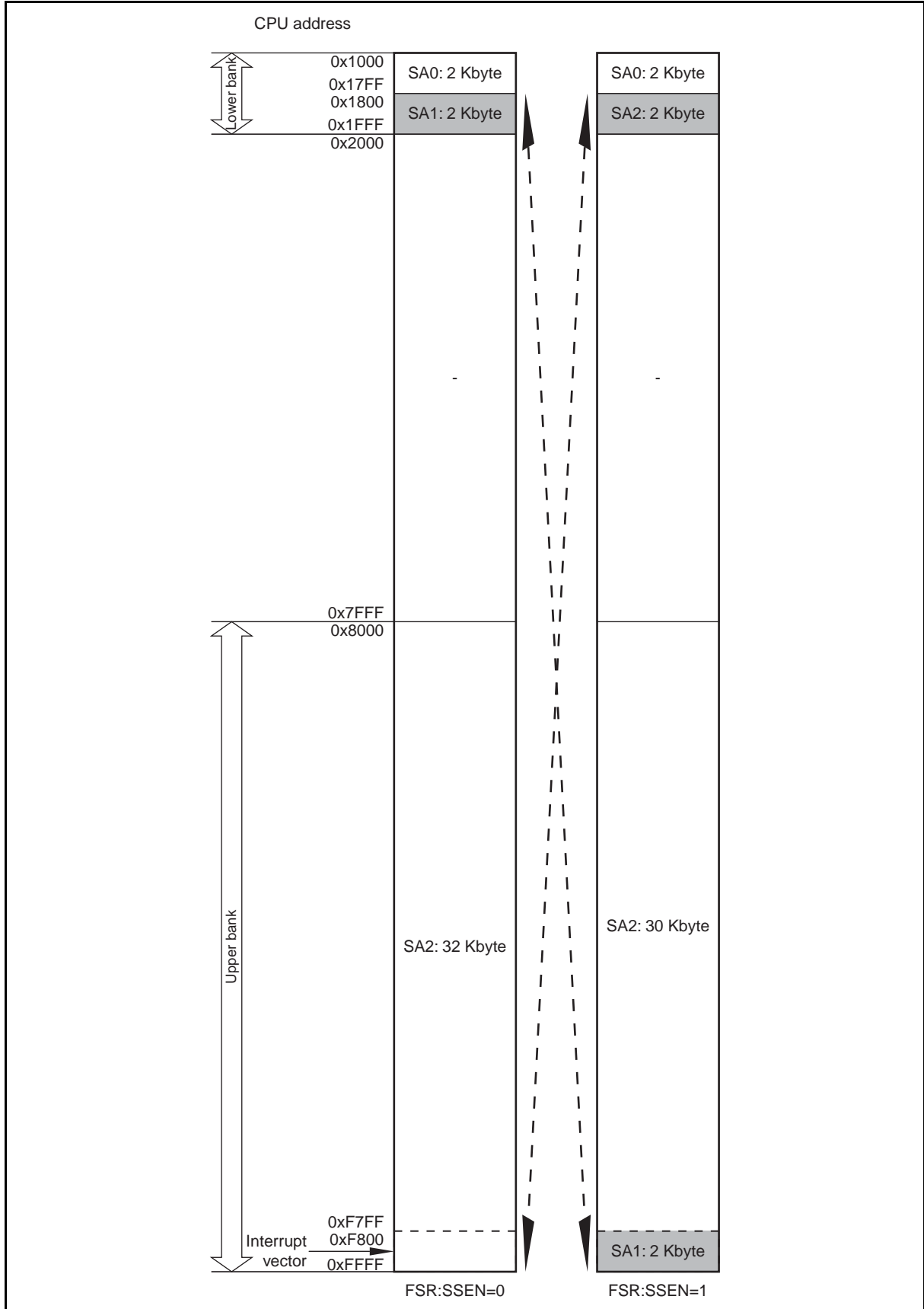
| bit1               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables Flash memory area programming/erasing. |
| <b>Writing "1"</b> | Enables Flash memory area programming/erasing.  |

[bit0] SSEN: Sector swap enable bit

This bit is used to swap part of SA2 in the upper bank, at which interrupt vectors are kept, for SA1 in the lower bank.

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Maps SA1 to 0x1800-0x1FFF, and the 2 Kbyte address area of SA2 to 0xF800-0xFFFF. |
| <b>Writing "1"</b> | Maps the 2 Kbyte address area of SA2 to 0x1800-0x1FFF, and SA1 to 0xF800-0xFFFF. |

**Figure 23.8-1 Access Sector Map by FSR:SSEN Value**



### 23.8.3 Flash Memory Sector Write Control Register 0 (SWRE0)

The flash memory sector write control register 0 (SWRE0) is installed in the Flash memory interface and used to set the function of protecting the Flash memory against spurious writes.

The flash memory sector write control register 0 (SWRE0) has bits for enabling/disabling programming data into individual sectors (SA0 to SA2). The initial value of each bit is "0", meaning programming data is disabled. Writing "1" to a bit in SWRE0 enables programming data into the sector corresponding to that bit. Writing "0" to a bit in SWRE0 prevents data from being accidentally written into the sector corresponding to that bit. When "0" is written to a bit in SWRE0, even though "1" is written to that bit afterward, data cannot be programmed into the sector corresponding to that bit. To re-program the data, execute a reset operation.

Only write data to SWRE0 by the byte. Setting the bits in SWRE0 using the bit manipulation instruction is prohibited.

#### ■ Register Configuration

|               |          |          |          |          |          |      |      |      |
|---------------|----------|----------|----------|----------|----------|------|------|------|
| bit           | 7        | 6        | 5        | 4        | 3        | 2    | 1    | 0    |
| Field         | Reserved | Reserved | Reserved | Reserved | Reserved | SA2E | SA1E | SA0E |
| Attribute     | W        | W        | W        | W        | W        | R/W  | R/W  | R/W  |
| Initial value | 0        | 0        | 0        | 0        | 0        | 0    | 0    | 0    |

#### ■ Register Functions

##### [bit7:3] Reserved bits

Always set these bits to "0".

##### [bit2:0] SA2E, SA1E, SA0E: Programming function setup bits

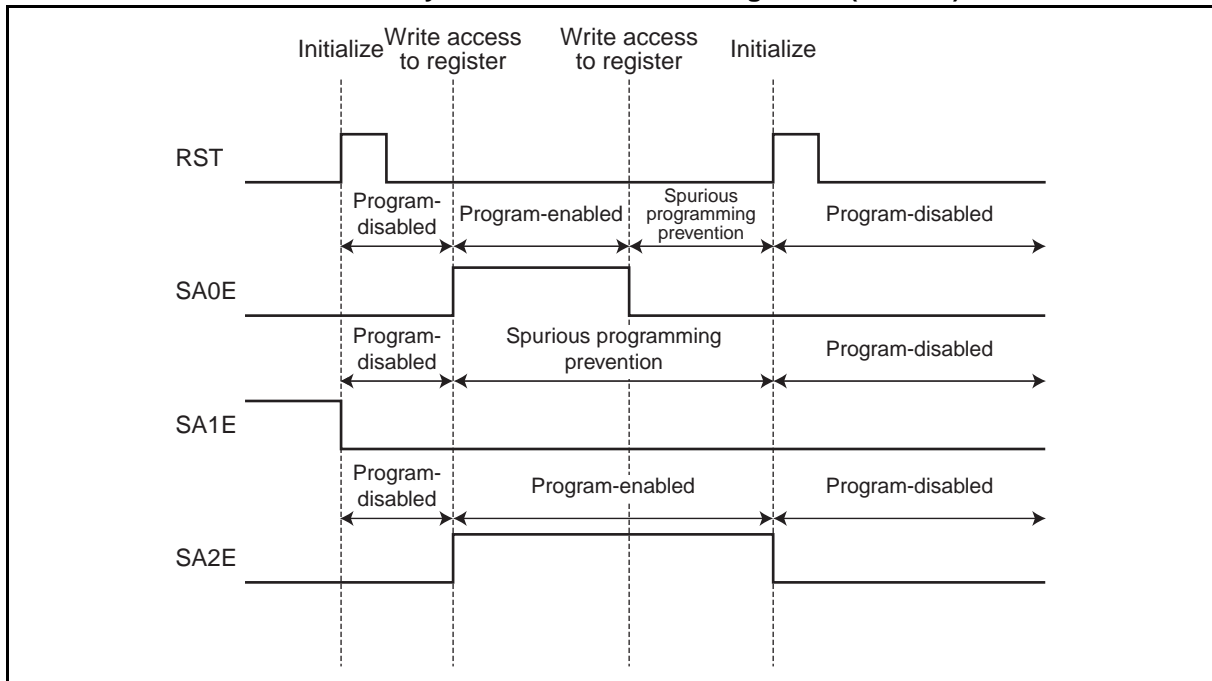
These bits are used to set the function of preventing data from being accidentally written into a sector of the Flash memory. Writing "1" to a bit in SWRE0 enables programming data into the sector corresponding to that bit. Writing "0" to a bit in SWRE0 prevents data from being accidentally written into the sector corresponding to that bit. In addition, a reset initializes this bit to "0" (programming disabled).

| Bit name | Corresponding sector in Flash memory |
|----------|--------------------------------------|
| SA2E     | SA2                                  |
| SA1E     | SA1                                  |
| SA0E     | SA0                                  |

Settings of SAxE (x = 0, 1 or 2) and their respective programming functions:

- Program-disabled (SAxE = 0)  
With "0" not written to the SAxE bit in the flash memory sector write control register 0 (SWRE0), programming data to a sector can be enabled by setting the SAxE bit corresponding to that sector to "1". (This is the state after a reset).
- Program-enabled (SAxE = 1)  
Data can be written to a sector corresponding to the SAxE bit.
- Spurious programming prevention (SAxE = 0)  
With "0" written to the SAxE bit in the flash memory sector write control register 0 (SWRE0), programming data to a sector cannot be enabled even though the SAxE bit corresponding to that sector is set to "1".

**Figure 23.8-2 Examples of Flash Memory Program-disabled, Program-enabled, and Spurious Programming Prevention States Depending on Flash Memory Sector Write Control Register 0 (SWRE0)**



■ **Note on Setting SWRE0 Register**

To program data to or erase data from SA0 (0x1000 to 0x17FF) or SA1 (0x1800 to 0x1FFF) of the Flash memory when FSR:SSEN is "0", set both SA0E and SA1E in the SWRE0 register to "1" first.

To program data to or erase data when FSR:SSEN is "1", set SA0E, SA1E and SA2E in the SWRE0 register to "1" first.

For details of the sector map of the Flash memory, see Figure 23.2-1 and Figure 23.8-1.

## 23.8.4 Flash Memory Status Register 3 (FSR3)

This section describes the flash memory status register 3 (FSR3).

### Register Configuration

| bit           | 7 | 6 | 5 | 4    | 3    | 2    | 1    | 0    |
|---------------|---|---|---|------|------|------|------|------|
| Field         | — | — | — | CERS | ESPS | SERS | PGMS | HANG |
| Attribute     | — | — | — | R    | R    | R    | R    | R    |
| Initial value | 0 | 0 | 0 | X    | X    | X    | X    | X    |

### Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4] CERS: Flash memory chip erase status bit

This bit indicates the chip erase status of the Flash memory.

| bit4        | Details  |
|-------------|--|
| Reading "0" | Indicates that Flash memory chip erase has been completed. |
| Reading "1" | Indicates that Flash memory chip erase is in progress.     |

#### [bit3] ESPS: Flash memory sector erase suspend status bit

This bit indicates the sector erase suspend of the Flash memory.

| bit3        | Details  |
|-------------|--|
| Reading "0" | Indicates that Flash memory sector erase suspend has been completed. |
| Reading "1" | Indicates that Flash memory sector erase suspend is in progress.     |

#### [bit2] SERS: Flash memory sector erase status bit

This bit indicates the sector erase status of the Flash memory.

| bit2        | Details  |
|-------------|--|
| Reading "0" | Indicates that Flash memory sector erase has been completed. |
| Reading "1" | Indicates that Flash memory sector erase is in progress.     |

#### [bit1] PGMS: Flash memory program status bit

This bit indicates the program status of the Flash memory.

The PGMS bit will never be asserted under the condition that the machine clock (MCLK) cycle is longer than 1  $\mu$ s. Use this bit with the machine clock (MCLK) cycle shorter than 1  $\mu$ s.

| bit1        | Details   |
|-------------|---|
| Reading "0" | Indicates that Flash memory program has been completed. |
| Reading "1" | Indicates that Flash memory program is in progress.     |

#### [bit0] HANG: Flash memory hang up status bit

This bit indicates whether the Flash memory has malfunctioned or not.

| bit0        | Details   |
|-------------|---|
| Reading "0" | Indicates that no malfunction of command input has occurred so far. |
| Reading "1" | Indicates that a malfunction of command input has occurred.         |

## 23.8.5 Flash Memory Status Register 4 (FSR4)

This section describes of the flash memory status register 4 (FSR4).

### ■ Register Configuration

| bit           | 7 | 6      | 5     | 4     | 3 | 2 | 1 | 0 |
|---------------|---|--------|-------|-------|---|---|---|---|
| Field         | — | CEREND | CTIEN | CERTO | — | — | — | — |
| Attribute     | — | R/W    | R/W   | R/W   | — | — | — | — |
| Initial value | 0 | 0      | 0     | 0     | 0 | 0 | 0 | 0 |

### ■ Register Functions

#### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

#### [bit6] CEREND: CEREND interrupt request flag bit

This bit indicates the completion of Flash memory chip erase.

The CEREND bit is set to "1" upon completion of the Flash memory automatic algorithm.

When the CEREND bit is set to "0" after Flash memory chip erase is completed, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

When Flash memory chip erase fails (FSR3:HANG = 1), this bit is cleared to "0".

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit6               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory chip erase is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory chip erase has been completed.  |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.   |

#### [bit5] CTIEN: CERTO interrupt enable bit

This bit enables or disables the generation of interrupt requests triggered by the failure of Flash memory chip erase.

| bit5               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Disables the interrupt request upon failure of Flash memory chip erase (FSR4:CERTO = 1). |
| <b>Writing "1"</b> | Enables the interrupt request upon failure of Flash memory chip erase (FSR4:CERTO = 1).  |



[bit4] CERTO: CERTO interrupt request flag bit

This bit indicates that Flash memory chip erase has failed.

When Flash memory chip erase fails, the CERTO bit is set to "1" upon completion of the Flash memory automatic algorithm.

An interrupt request is generated when the CERTO bit is set to "1", provided that generating an interrupt request upon failure of Flash memory chip erase has been enabled (FSR4:CTIEN = 1).

When the CERTO bit is set to "1" after Flash memory chip erase is completed, further Flash memory programming/erasing is disabled. Writing a reset command can make the Flash memory return to the normal command state.

Writing "0" to this bit clears it.

Writing "1" to this bit has no effect on operation.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

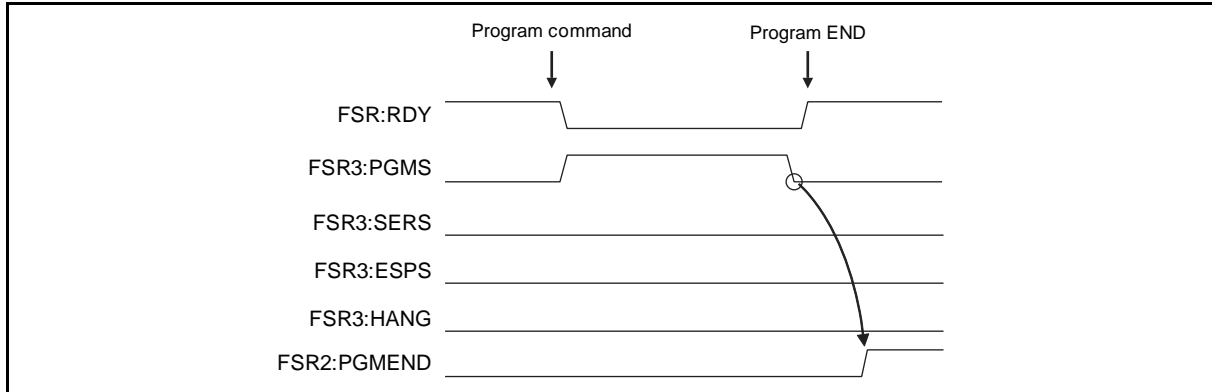
| bit4               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the device is in the command input wait state or Flash memory chip erase is in progress. |
| <b>Reading "1"</b> | Indicates that Flash memory chip erase has failed.  |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.   |

[bit3:0] Undefined bits

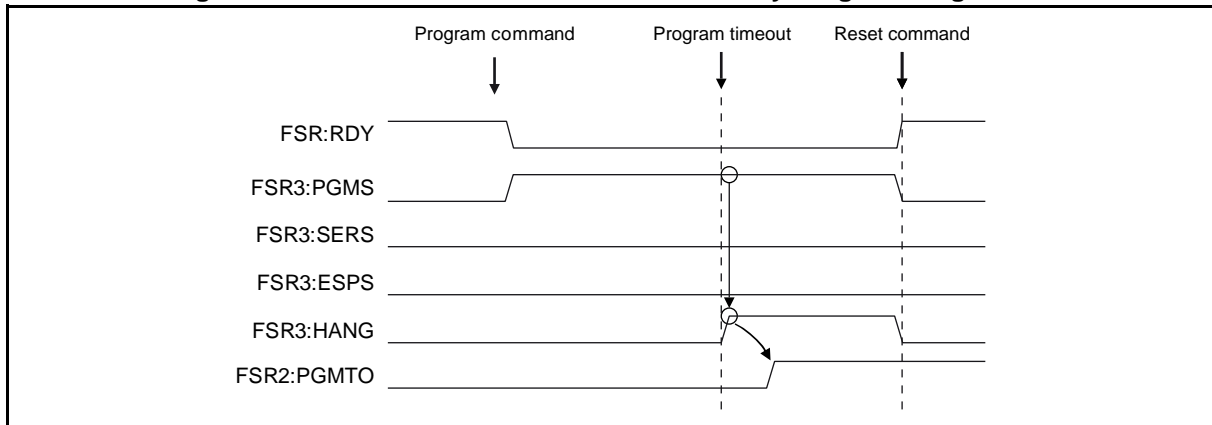
Their read values are always "0". Writing values to these bits has no effect on operation.

■ **Examples of Status of Flash Memory Status Register 2, Flash Memory Status Register 3, Flash Memory Status Register 4 and RDY Bit (FSR:RDY)**

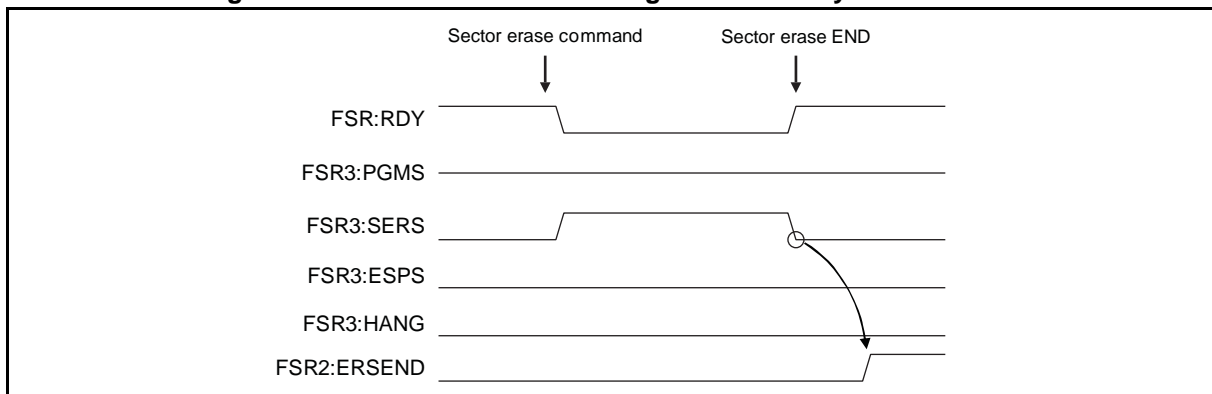
**Figure 23.8-3 FSR2:PGMEND during Flash Memory Programming**



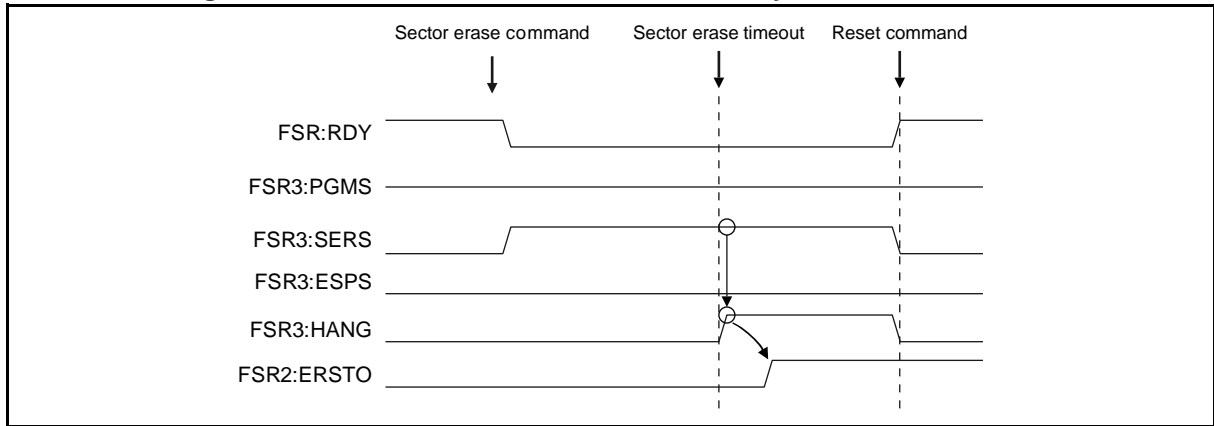
**Figure 23.8-4 FSR2:PGMTO when Flash Memory Programming Failed**



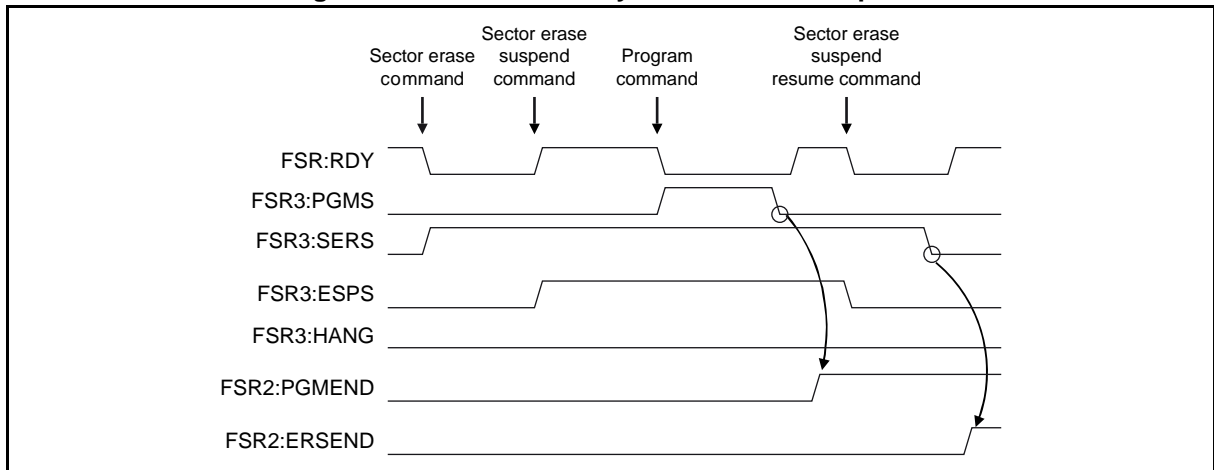
**Figure 23.8-5 FSR2:ERSEND during Flash Memory Sector Erase**



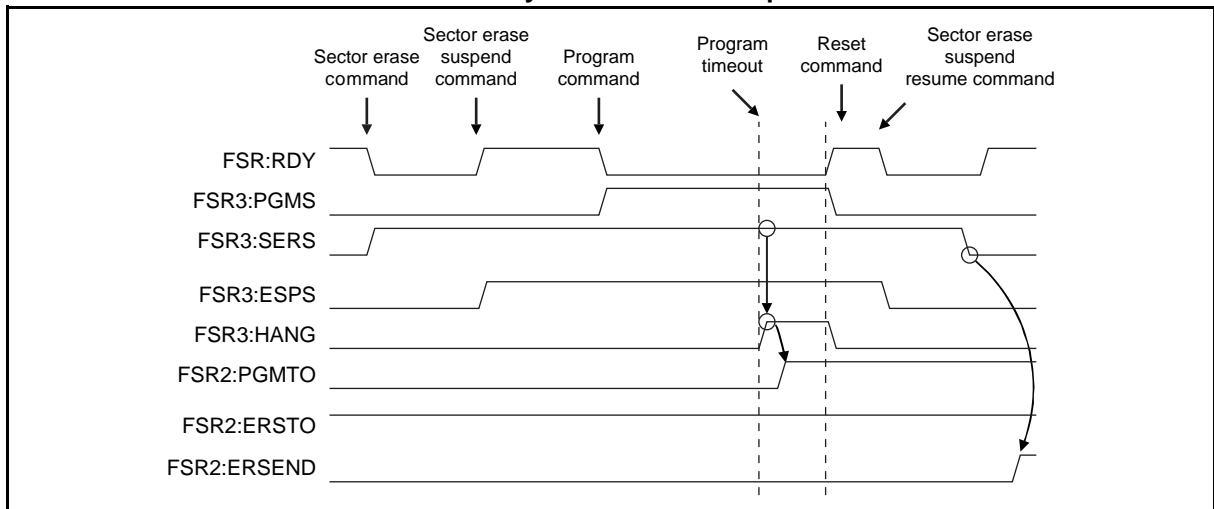
**Figure 23.8-6 FSR2:ERSTO when Flash Memory Sector Erase Failed**



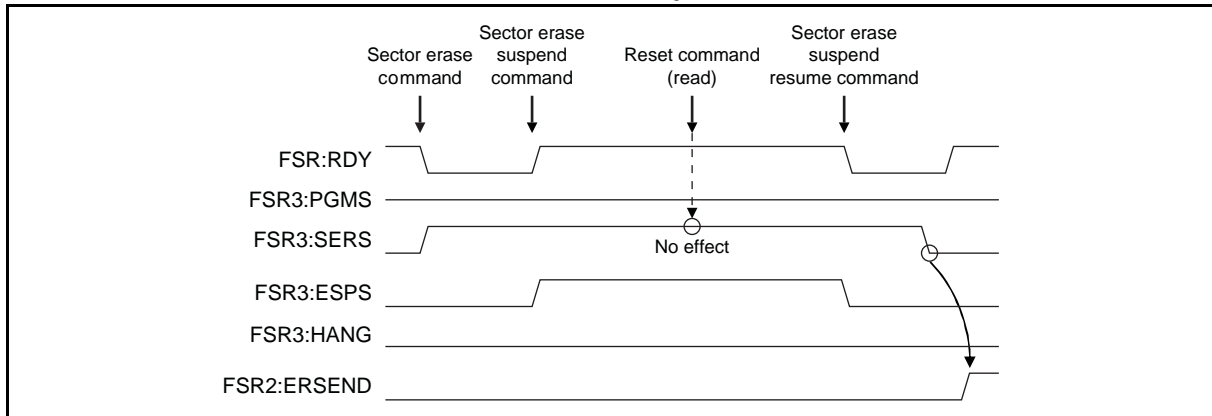
**Figure 23.8-7 FSR2:PGMEND and FSR2:ERSEND when Flash Memory Programming Is in Progress with Flash Memory Sector Erase Suspended**



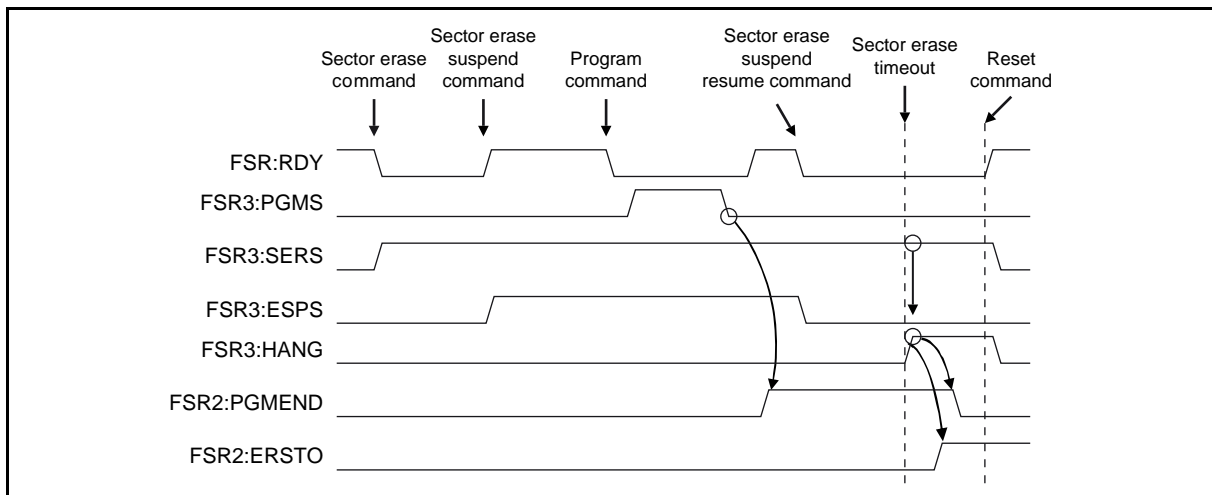
**Figure 23.8-8 FSR2:PGMTO and FSR2:ERSEND when Flash Memory Programming Failed with Flash Memory Sector Erase Suspended**



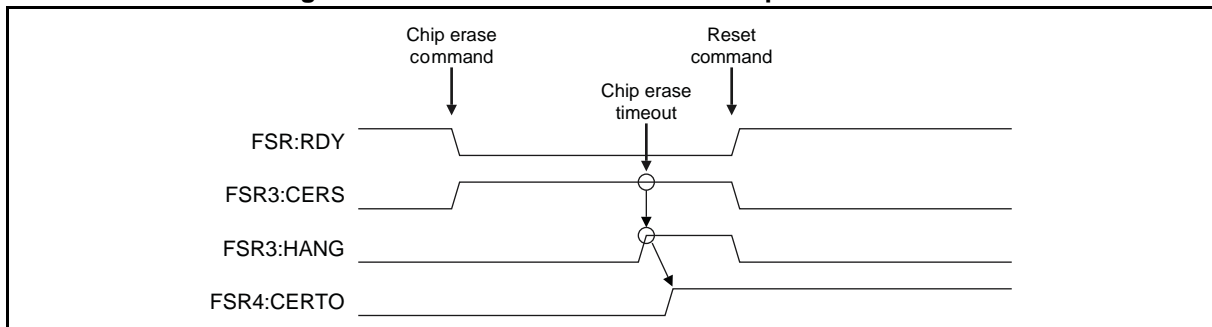
**Figure 23.8-9 FSR2:ERSEND when Flash Memory Read Is in Progress with Flash Memory Sector Erase Suspended**



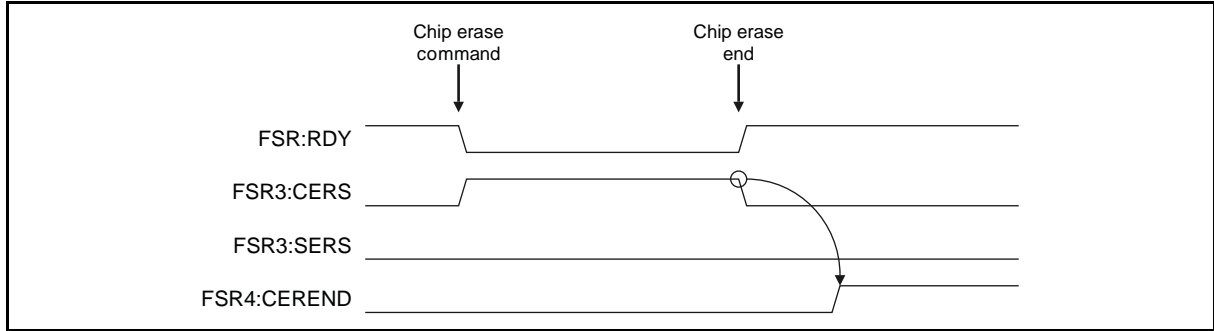
**Figure 23.8-10 FSR2:PGMEND and FSR2:ERSTO when Flash Memory Sector Erase Failed after Sector Erase Has Resumed**



**Figure 23.8-11 FSR4:CERTO when Chip Erase Failed**



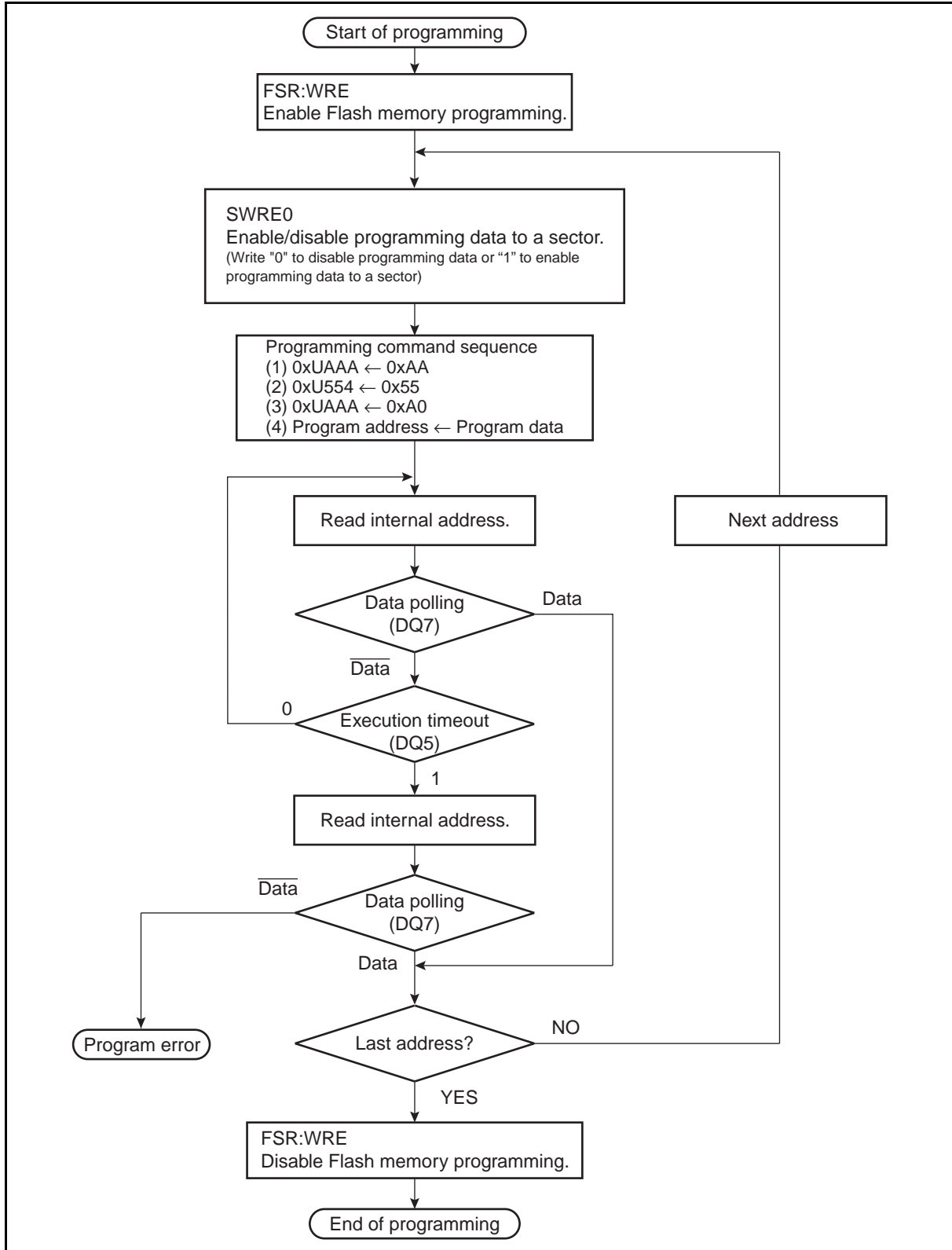
**Figure 23.8-12 FSR4:CEREND during Chip Erase**



**Flash Memory Sector Write Control Register 0 (SWRE0) Setup Flow Chart**

Set the FSR:WRE bit to "1" to enable Flash memory programming, then enable or disable programming data into a sector by setting the corresponding bit in the SWRE0 register to "1" or "0" respectively.

**Figure 23.8-13 Sample Procedure for Enabling/Disabling Flash Memory Programming**



**■ Note on Setting (FSR:WRE)**

To program data to the Flash memory, set the WRE bit to "1" to enable Flash memory programming and then set the bit in the SWRE0 register corresponding to a sector to which data is to be written. When Flash memory programming is disabled by setting the WRE bit to "0", no write access to a sector in the Flash memory can be executed even though it has been enabled by setting a bit corresponding to that sector in the SWRE0 register to "1".

## 23.9 Notes on Using Dual Operation Flash Memory

---

This section provides notes on using the dual operation Flash memory.

---

### ■ Restriction on Using Toggle Bit Flag (DQ6)

When using the dual-operation Flash memory (The Flash memory write control program is executed on the Flash memory), the toggle bit flag (DQ6) cannot be used to check the operating state of the Flash memory during programming or erasing. Therefore, use the data polling flag (DQ7) to check the internal operating state of the Flash memory after programming data to the Flash memory or erasing data from the Flash memory as shown in the examples in Figure 23.5-1 and Figure 23.5-2.

The restriction above does not apply if the Flash memory write control program is executed on the RAM.



# ***CHAPTER 24***

---

# ***NON-VOLATILE REGISTER (NVR) INTERFACE***

**This chapter describes the functions and operations of the NVR interface.**

- 24.1 Overview
- 24.2 Configuration
- 24.3 Registers
- 24.4 Notes on Main CR Clock Trimming
- 24.5 Notes on Using NVR Interface

## 24.1 Overview

The NVR (Non-Volatile Register) area is a reserved area in the Flash that stores system information and option settings. After a reset, data in the NVR Flash area will be fetched and stored in registers in the NVR I/O area. In the MB95850K/860K/870K Series, the NVR interface is used to store the following data:

- Coarse trimming value for main CR Clock (5 bits)
- Fine trimming value for main CR Clock (5 bits)
- Watchdog timer selection ID (16 bits)
- Temperature dependent adjustment value for main CR clock (5 bits)

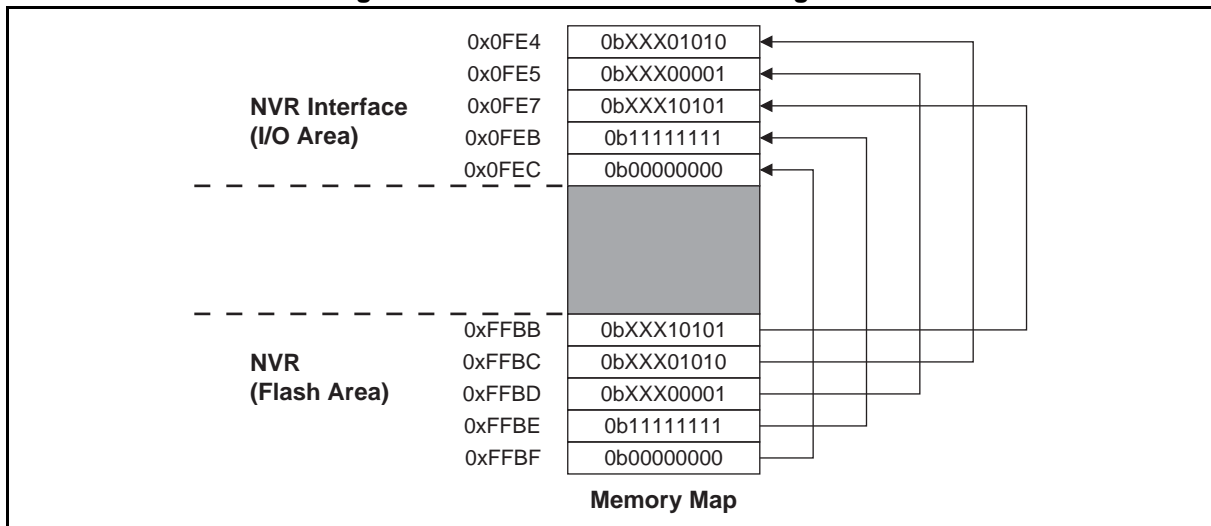
### ■ Functions of NVR Interface

Functions of the NVR interface are as follows:

1. The NVR interface retrieves all data from the NVR Flash area and stores it in the registers in the NVR I/O area after a reset. See Figure 24.1-1 and Figure 24.2-1.
2. The NVR interface enables the user to know the value of the initial CR trimming setting.
3. The NVR interface enables the user to select the hardware watchdog timer or software watchdog timer by modifying the 16-bit watchdog timer selection ID. The watchdog timer selection ID cannot be modified while the CPU is running.

Figure 24.1-1 shows the retrieval of NVR during a reset.

**Figure 24.1-1 Retrieval of NVR during Reset**



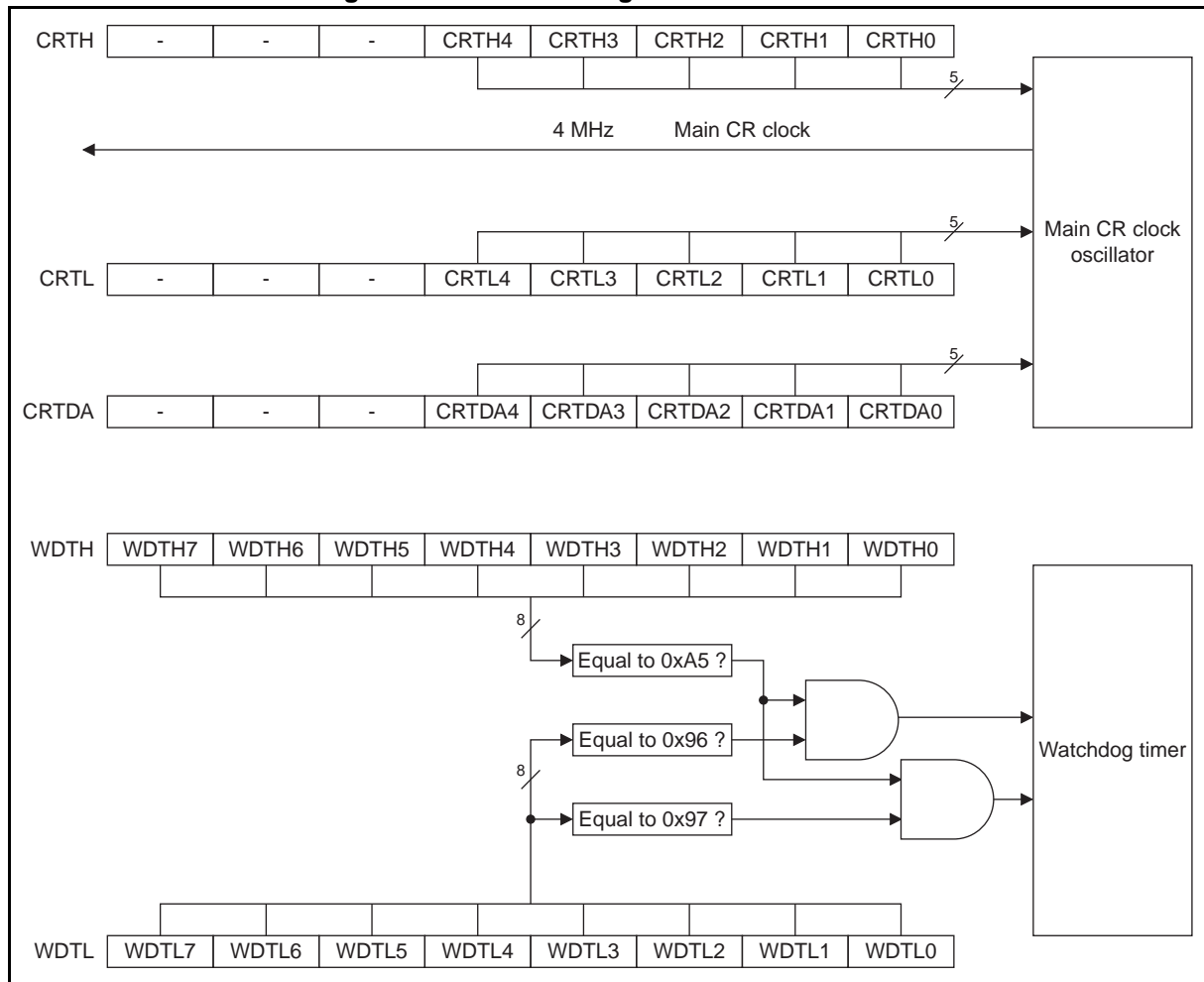
## 24.2 Configuration

The NVR interface consists of the following blocks:

- Trimming of Main CR Clock (CRTH and CRTL)
- Watchdog Timer Selection ID (WDTH and WDTL)
- Main CR Temperature Dependent Adjustment (CRTDA)

### ■ Block Diagram of NVR Interface

**Figure 24.2-1 Block Diagram of NVR Interface**



## 24.3 Registers

---

This section lists the registers of the NVR interface.

---

**Table 24.3-1 List of NVR Interface Registers**

| Register abbreviation | Register name   | Reference |
|-----------------------|---|-----------|
| CRTH                  | Main CR clock trimming register (upper)                 | 24.3.1    |
| CRTL                  | Main CR clock trimming register (lower)                 | 24.3.2    |
| CRTDA                 | Main CR clock temperature dependent adjustment register | 24.3.3    |
| WDTH                  | Watchdog timer selection ID register (upper)            | 24.3.4    |
| WDTL                  | Watchdog timer selection ID register (lower)            | 24.3.4    |

## 24.3.1 Main CR Clock Trimming Register (Upper) (CRTH)

This section describes the main CR clock trimming register (upper) (CRTH).

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|---|-------|-------|-------|-------|-------|
| Field         | — | — | — | CRTH4 | CRTH3 | CRTH2 | CRTH1 | CRTH0 |
| Attribute     | — | — | — | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0 | X     | X     | X     | X     | X     |

### ■ Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4:0] CRTH[4:0]: Main CR clock coarse trimming bits

The settings of these bits are loaded from the Flash address 0xFFBC (bit4:0) after a reset. Their initial values are determined by the pre-loaded values in the NVR Flash area.

Coarse trimming modifies the main CR clock frequency with a bigger step. Increasing the coarse trimming value decreases the main CR clock frequency.

| bit4:0          | Details                         |
|-----------------|---------------------------------|
| Writing "00000" | Highest main CR clock frequency |
| :               | :                               |
| Writing "11111" | Lowest main CR clock frequency  |

See "24.4 Notes on Main CR Clock Trimming" and "24.5 Notes on Using NVR Interface" for details of main CR clock trimming and notes on changing the main CR clock values respectively.

## 24.3.2 Main CR Clock Trimming Register (Lower) (CRTL)

This section describes the main CR clock trimming register (lower) (CRTL).

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|---|-------|-------|-------|-------|-------|
| Field         | — | — | — | CRTL4 | CRTL3 | CRTL2 | CRTL1 | CRTL0 |
| Attribute     | — | — | — | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0 | X     | X     | X     | X     | X     |

### ■ Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4:0] CRTL[4:0]: Main CR clock fine trimming bits

The settings of these bits are loaded from the Flash address 0xFFBD (bit4:0) after a reset. Their initial values are determined by the pre-loaded values in the NVR Flash area.

Fine trimming modifies the main CR clock frequency with a smaller step. Increasing the fine trimming value decreases the main CR clock frequency.

| bit4:0          | Details                         |
|-----------------|---------------------------------|
| Writing "00000" | Highest main CR clock frequency |
| :               | :                               |
| Writing "11111" | Lowest main CR clock frequency  |

See "24.4 Notes on Main CR Clock Trimming" and "24.5 Notes on Using NVR Interface" for details of main CR clock trimming and notes on changing the main CR clock values respectively.

### 24.3.3 Main CR Clock Temperature Dependent Adjustment Register (CRTDA)

This section describes the main CR clock temperature dependent adjustment register (CRTDA).

#### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4      | 3      | 2      | 1      | 0      |
|---------------|---|---|---|--------|--------|--------|--------|--------|
| Field         | — | — | — | CRTDA4 | CRTDA3 | CRTDA2 | CRTDA1 | CRTDA0 |
| Attribute     | — | — | — | R/W    | R/W    | R/W    | R/W    | R/W    |
| Initial value | 0 | 0 | 0 | X      | X      | X      | X      | X      |

#### ■ Register Functions

[bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit4:0] CRTDA[4:0]: Main CR clock temperature dependent adjustment bits

These bits are loaded from the Flash address 0xFFBB (bit4:0) after a reset. Their initial values are determined by the pre-load values in the NVR Flash area.

Temperature dependent adjustment maintains the accuracy of the main CR output frequency within a temperature range. It works in combination with the coarse trimming settings in the CRTH register and the fine trimming settings in the CRTL register. In addition, increasing the value of the CRTDA register decreases the main the main CR clock frequency

| bit4:0          | Details                         |
|-----------------|---------------------------------|
| Writing "00000" | Highest main CR clock frequency |
| :               | :                               |
| Writing "11111" | Lowest main CR clock frequency  |

See "24.4 Notes on Main CR Clock Trimming" and "24.5 Notes on Using NVR Interface" for details of main CR clock trimming and notes on changing the main CR clock values respectively.

### 24.3.4 Watchdog Timer Selection ID Register (Upper/Lower) (WDTH/WDTL)

This section describes the watchdog timer selection ID register (upper/lower) (WDTH/WDTL).

#### ■ Register Configuration

| WDTH          |       |       |       |       |       |       |       |       |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
| Field         | WDTH7 | WDTH6 | WDTH5 | WDTH4 | WDTH3 | WDTH2 | WDTH1 | WDTH0 |
| Attribute     | R     | R     | R     | R     | R     | R     | R     | R     |
| Initial value | X     | X     | X     | X     | X     | X     | X     | X     |

| WDTL          |       |       |       |       |       |       |       |       |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
| Field         | WDTL7 | WDTL6 | WDTL5 | WDTL4 | WDTL3 | WDTL2 | WDTL1 | WDTL0 |
| Attribute     | R     | R     | R     | R     | R     | R     | R     | R     |
| Initial value | X     | X     | X     | X     | X     | X     | X     | X     |

#### ■ Functions of WDTH Register

[bit7:0] WDTH[7:0]: Watchdog timer selection ID (upper) bits

These bits are loaded from the Flash address 0xFFBE (bit7:0) after a reset. The initial values are determined by the pre-loaded values in the NVR Flash area.

These bits cannot be modified while the CPU is running.

See Table 24.3-2 for watchdog timer selection.

See "24.5 Notes on Using NVR Interface" for notes on writing NVR values.

#### ■ Functions of WDTL Register

[bit7:0] WDTL[7:0]: Watchdog timer selection ID (lower) bits

These bits are loaded from the Flash address 0xFFBF (bit7:0) after a reset. The initial values are determined by the pre-loaded values in the NVR Flash area.

These bits cannot be modified while the CPU is running.

See Table 24.3-2 for watchdog timer selection.

See "24.5 Notes on Using NVR Interface" for notes on writing NVR values.

**Table 24.3-2 Watchdog Timer Selection ID**

| WDTH[7:0], WDTL[7:0] | Function  |
|----------------------|---|
| 0xA596               | The hardware watchdog timer is disabled; the software watchdog timer is enabled.  |
| 0xA597               | The hardware watchdog timer is enabled; the software watchdog timer is disabled. The hardware watchdog timer can be stopped in all standby modes (stop mode, sleep mode, time-base timer mode and watch mode).  |
| Other than the above | The hardware watchdog timer is enabled; the software watchdog timer is disabled. The hardware watchdog timer keeps operating in all standby modes (stop mode, sleep mode, time-base timer mode and watch mode). |



## 24.4 Notes on Main CR Clock Trimming

---

**This section provides notes on main CR clock trimming.**

---

After a hardware reset, the 10-bit main CR clock trimming value and the 5-bit temperature dependent adjustment value will be loaded from the NVR Flash area to registers in the NVR I/O area.

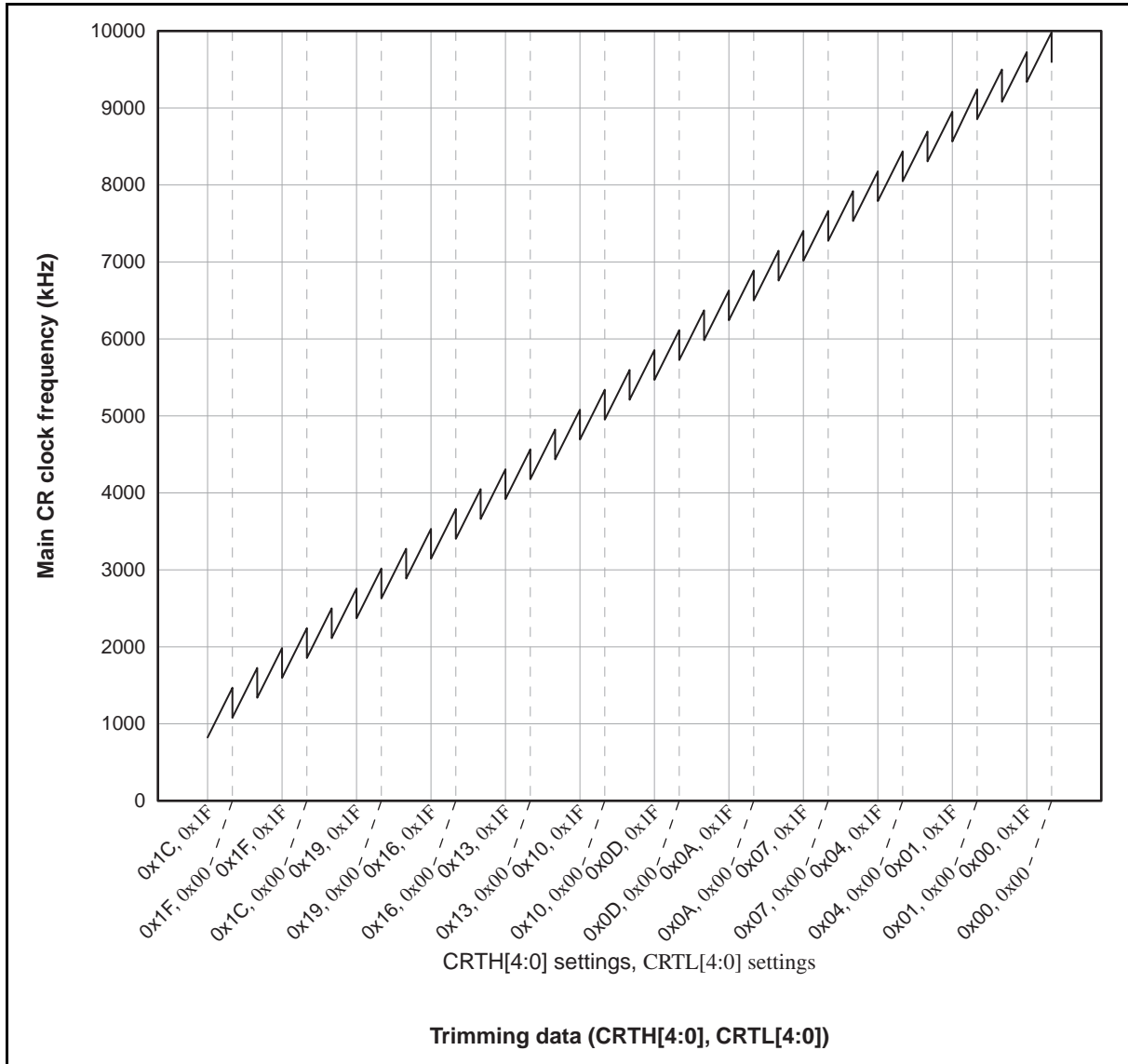
Table 24.4-1 shows the step size of main CR clock trimming.

**Table 24.4-1 Step Size of Main CR Clock Trimming**

| Function                         | Coarse trimming value<br>CRTH[4:0] | Fine trimming value<br>CRTL[4:0] |
|----------------------------------|------------------------------------|----------------------------------|
| To achieve the minimum frequency | 0b11111                            | 0b11111                          |
| To achieve the maximum frequency | 0b00000                            | 0b00000                          |
| Step Size                        | 220 kHz to 300 kHz                 | 14 kHz to 20 kHz                 |

The relationship between main CR clock frequency and trimming step size is shown in the diagram below.

**Figure 24.4-1 Relationship between Main CR Clock Frequency and Trimming Step Size (with CRTDA[4:0] = 0b10000)**



## 24.5 Notes on Using NVR Interface

---

This section provides notes on using the NVR interface.

---

### ■ Note on Changing Main CR Frequency

Please note that the NVR interface does not program a modified value to the NVR Flash area. To modify the CRTH, CRTL and CRTDA registers, program their new values to the NVR Flash with the Flash writer.

### ■ Note on Flash Erase and Trimming Value

1. A Flash erase operation will erase all NVR data.

The Flash writer carries out the following procedure to keep original system settings.

- (1) Make a backup of data in CRTH:CRTH[4:0], CRTL:CRTL[4:0] and CRTDA:CRTDA[4:0].
- (2) Erase the Flash.
- (3) Restore all data in CRTH:CRTH[4:0], CRTL:CRTL[4:0] and CRTDA:CRTDA[4:0] to the NVR Flash area.

If there is new data in CRTH:CRTH[4:0], CRTL:CRTL[4:0] and CRTDA:CRTDA[4:0], the Flash writer will program the new data to the NVR Flash area.

2. The trimming value has been preset before this device is shipped. If the preset trimming value is modified after the device has been shipped, Fujitsu Semiconductor does not warrant proper operation of the device with respect to use based on the modified trimming value.
3. If the Flash operation is performed by the user program code, restore the original trimming data to the NVR Flash area by the user program code. Otherwise, the trimming value, which has been preset before this device is shipped, is erased by the Flash erase operation.



# **CHAPTER 25**

---

# **COMPARATOR**

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

- 25.1 Overview
- 25.2 Configuration
- 25.3 Dedicated BGR
- 25.4 Pins
- 25.5 Interrupt
- 25.6 Operations and Setting Procedure Example
- 25.7 Register

## **25.1 Overview**

---

**The comparator monitors the two analog input voltages, and automatically generates an interrupt upon detection of a change in the edge of comparator output.**

---

### **■ Function of Comparator**

The function of the comparator is to monitor the two analog input voltages and compare them. Using the voltage of the inverting analog input (negative input) or of the dedicated BGR as a reference voltage, the comparator outputs "H" if the voltage of the non-inverting analog input voltage (positive input) is higher than the reference voltage; otherwise, it outputs "L". In addition, upon detection of a rising edge or falling edge of the comparator output, the comparator outputs a corresponding interrupt.

# MB95850K/860K/870K Series

## 25.2 Configuration

The comparator module consists of the following blocks:

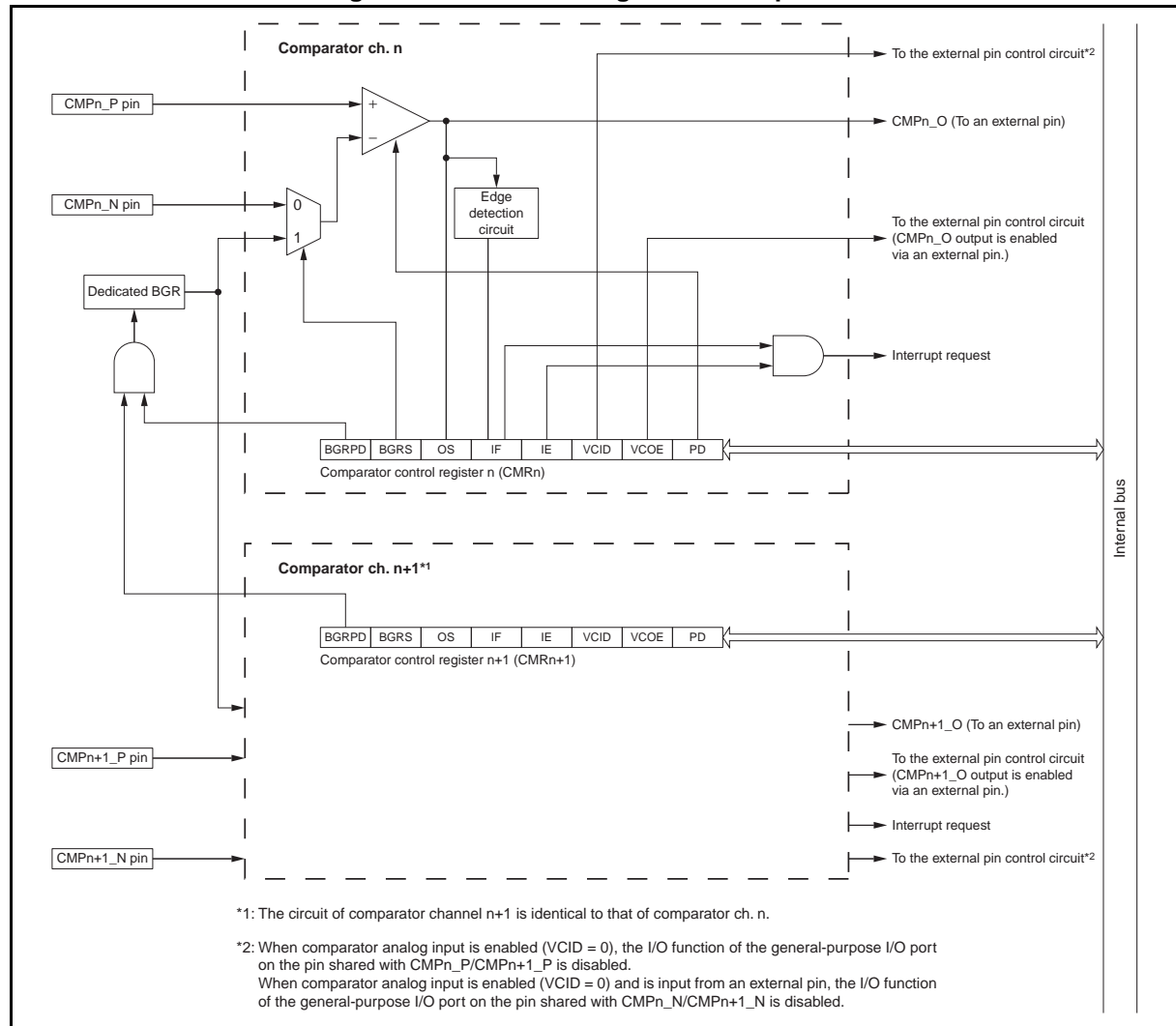
- Comparator
- Edge detection circuit
- Comparator control register ch. n (CMRn)

The number of pins and that of channels of the comparator vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

### ■ Block Diagram of Comparator

Figure 25.2-1 Block Diagram of Comparator



● **Comparator**

The comparator monitors the voltages of two external analog inputs and compares them. Using the voltage of the inverting analog input (negative input) or the dedicated BGR as a reference voltage, the comparator outputs "H" if the voltage of the non-inverting analog input (positive input) is higher than the reference voltage; otherwise, it outputs "L".

● **Edge detection circuit**

Except in stop mode, watch mode or time-base timer mode, upon detection of a rising edge or falling edge of comparator output, the edge detection circuit automatically raises the interrupt flag (CMRn:IF).

● **Comparator control register ch. n (CMRn)**

This register has the following functions:

- To power up or power down the comparator (CMRn:PD)
- To enable or disable comparator output (CMRn:VCOE)
- To enable or disable comparator analog input (CMRn:VCID)
- To power up or power down the dedicated BGR (CMRn:BGRPD)
- To select the comparator reference voltage source (CMRn:BGRS)

Except in stop mode, watch mode or time-base timer mode, if the interrupt request enable bit (IE) in the CMRn register has been set to "1", upon detection of a rising edge or falling edge of comparator output, the comparator generates an interrupt request, and the output edge detection interrupt flag bit (IF) in the CMRn register is automatically set to "1" at the same time.

The output status of a comparator can be read through the output status bit (OS) in the CMRn register.



## MB95850K/860K/870K Series

### 25.3 Dedicated BGR

---

**This section describes the function of the dedicated BGR.**

---

#### ■ Function of Dedicated BGR

The dedicated BGR is a reference voltage generation circuit that outputs an analog voltage to the comparator.

---

#### Note:

The comparator has only one dedicated BGR, which is shared by all channels of the comparator. The dedicated BGR is enabled when a channel uses it. The dedicated BGR can be powered down when the BGRPD bit (CMRn:BGRPD) of each channel is set to "1".

---

## 25.4 Pins

---

**This section describes the pins of the comparator.**

---

### ■ Pins of Comparator

Table 25.4-1 shows details of the pins of the comparator.

**Table 25.4-1 Pins of Comparator**

| Pin name | Pin function   |
|----------|--|
| CMPn_P   | Comparator non-inverting analog input (positive input) |
| CMPn_N   | Comparator inverting analog input (negative input)     |
| CMPn_O   | Comparator digital output                              |

**MB95850K/860K/870K Series****25.5 Interrupt**

---

The comparator generates an interrupt called output edge detection interrupt. An interrupt request number and an interrupt vector are assigned to the interrupt.

---

**■ Output Edge Detection Interrupt**

Table 25.5-1 shows details of the output edge detection interrupt.

**Table 25.5-1 Details of Output Edge Detection Interrupt**

| Item                           | Details  |
|--------------------------------|--|
| Interrupt generating condition | An output rising edge or output falling edge occurs. |
| Interrupt flag                 | CMRn:IF  |
| Interrupt enable bit           | CMRn:IE  |

---

**Note:**

In stop mode, watch mode or time-base timer mode, the edge detection circuit stops operating, and the IF bit in the comparator control register ch. n (CMRn) is not to be updated even if the comparator has been turned on.

---

## 25.6 Operations and Setting Procedure Example

The comparator can be activated by the software according to the setting of the PD bit in the CMRn register.

### ■ Software Activation of Comparator (With Analog Input from External Pin)

To activate the comparator with analog input from an external pin, do the settings shown in Figure 25.6-1.

Figure 25.6-1 Settings for Activating Comparator (With Analog Input from External Pin)

|      | bit7               | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|------|--------------------|------|------|------|------|------|------|------|
| CMRn | BGRPD              | BGRS | OS   | IF   | IE   | VCID | VCOE | PD   |
|      | 1                  | 0    | ○    | ○    | 0    | 0    | 0    | 0    |
|      | ○ : Bit to be used |      |      |      |      |      |      |      |
|      | x : Unused bit     |      |      |      |      |      |      |      |
|      | 0 : Set to "0"     |      |      |      |      |      |      |      |
|      | 1 : Set to "1"     |      |      |      |      |      |      |      |

After the comparator is activated as shown above, it has to wait for the stabilization wait time to elapse before stabilizing. For details of the stabilization wait time of the comparator, refer to the device data sheet.

#### Note:

Before activating the comparator, set the IE bit in the CMRn register to "0" in advance in order to avoid any unexpected interrupt generated due to the comparator being unstable at its startup.

### ■ Setting Procedure Example

Below is an example of procedure for setting the comparator when the inverting analog input (negative input) voltage is input from an external pin:

#### ● Initial settings

1. Disable the comparator interrupt request, and power down the comparator. (CMRn:IE = 0, CMRn:PD = 1)
2. Activate the comparator according to the settings shown in Figure 25.6-1.
3. Wait until the comparator stabilizes.
4. Clear the interrupt flag bit. (CMRn:IF = 0)
5. Enable the comparator interrupt request, and enable the comparator output if necessary. (CMRn:IE = 1, CMRn:VCOE = 1)

### ■ Software Activation of Comparator (With Analog Input from Dedicated BGR)

To activate the comparator with analog input from the dedicated BGR, do the settings shown in Figure 25.6-2.

**Figure 25.6-2 Settings for Activating Comparator (With Analog Input from Dedicated BGR)**

|      | bit7  | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|------|-------|------|------|------|------|------|------|------|
| CMRn | BGRPD | BGRS | OS   | IF   | IE   | VCID | VCOE | PD   |
|      | 0     | 1    | ○    | ○    | 0    | 0    | 0    | 0    |

○ : Bit to be used  
 × : Unused bit  
 0 : Set to "0"  
 1 : Set to "1"

After the dedicated BGR is activated as shown above, it has to wait for the stabilization wait time to elapse before stabilizing. For details of the stabilization wait time of the dedicated BGR, refer to the device data sheet.

#### Note:

Before activating the comparator, set the IE bit in the CMRn register to "0" in advance in order to avoid any unexpected interrupt generated due to the comparator being unstable at its startup.

### ■ Setting Procedure Example

Below is an example of procedure for setting the comparator when the inverting analog input (negative input) voltage is input from the dedicated BGR:

#### ● Initial settings

1. Disable the comparator interrupt request. (CMRn:IE = 0)
2. Activate the comparator and the dedicated BGR according to the settings shown in Figure 25.6-2.
3. Wait until the dedicated BGR stabilizes.
4. Clear the interrupt flag bit. (CMRn:IF = 0)
5. Enable the comparator interrupt request, and enable the comparator output if necessary. (CMRn:IE = 1, CMRn:VCOE = 1)

## 25.7 Register

---

**This section describes the register of the comparator.**

---

**Table 25.7-1 List of Comparator Register**

| Register abbreviation | Register name                     | Reference |
|-----------------------|-----------------------------------|-----------|
| CMRn                  | Comparator control register ch. n | 25.7.1    |

## MB95850K/860K/870K Series

### 25.7.1 Comparator Control Register ch. n (CMRn)

The comparator control register ch. n (CMRn) has the following functions:

- To power up or power down the comparator (CMRn:PD)
- To enable or disable comparator output (CMRn:VCOE)
- To enable and disable comparator analog input (CMRn:VCID)
- To power up or power down the dedicated BGR (CMRn:BGRPD)
- To select the comparator reference voltage source (CMRn:BGRS)

Except in stop mode, watch mode or time-base timer mode, if CMRn:IE has been set to "1", upon detection of a rising edge or falling edge of comparator output, the comparator generates an interrupt request and the interrupt flag bit (CMRn:IF) is automatically set to "1" at the same time.

The output status of the comparator can be read through the OS bit in the comparator control register ch. n (CMRn).

#### ■ Register Configuration

| bit           | 7     | 6    | 5  | 4   | 3   | 2    | 1    | 0   |
|---------------|-------|------|----|-----|-----|------|------|-----|
| Field         | BGRPD | BGRS | OS | IF  | IE  | VCID | VCOE | PD  |
| Attribute     | R/W   | R/W  | R  | R/W | R/W | R/W  | R/W  | R/W |
| Initial value | 1     | 1    | 0  | 0   | 0   | 1    | 0    | 1   |

#### ■ Register Functions

[bit7] BGRPD: Dedicated BGR power down control bit

This bit powers up or powers down the dedicated BGR.

| bit7        | Details                        |
|-------------|--------------------------------|
| Writing "0" | Powers up the dedicated BGR.   |
| Writing "1" | Powers down the dedicated BGR. |

Note: After the dedicated BGR is powered up, it has to wait for the stabilization time to elapse stabilize before starting to operate. For details of the stabilization time of the dedicated BGR, refer to the device data sheet.

[bit6] BGRS: Inverting analog input (negative input) voltage source select bit

This bit selects the external pin or the dedicated BGR as the inverting analog input (negative input) voltage source.

| bit6        | Details  |
|-------------|--|
| Writing "0" | Selects the external pin as the inverting analog input (negative input) voltage source.  |
| Writing "1" | Selects the dedicated BGR as the inverting analog input (negative input) voltage source. |

Note: Power down the comparator (CMRn:PD = 1) before changing the inverting analog input (negative input) voltage source.

[bit5] OS: Output status bit

This bit indicates the output status of the comparator.

| bit5               | Details                                    |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the comparator outputs "L". |
| <b>Reading "1"</b> | Indicates that the comparator outputs "H". |

Note: This bit is not updated in stop mode, watch mode or time-base timer mode. When the PD bit is set to "1" (to power down the comparator), the OS bit becomes "0".

[bit4] IF: Output edge detection interrupt flag bit

This bit detects the output rising edge and the output falling edge of the comparator.

With the comparator in operation, this bit is set to "1" if an output rising edge or an output falling edge occurs.

When read by the read-modify-write (RMW) type of instruction, this bit always returns "1".

| bit4               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that no output rising edge/falling edge has occurred. |
| <b>Reading "1"</b> | Indicates that an output rising edge/falling edge has occurred. |
| <b>Writing "0"</b> | Clears this bit.  |
| <b>Writing "1"</b> | Has no effect on operation.                                     |

Note: This bit is not be updated in stop mode, watch mode or time-base timer mode.

[bit3] IE: Interrupt request enable bit

This bit enables or disables the interrupt request of the comparator.

Writing "0" to this bit disables the interrupt request of the comparator.

Writing "1" to this bit enables the interrupt request of the comparator. With the interrupt request enabled, the comparator generates an interrupt request when detecting an output rising edge or an output falling edge.

| bit3               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the interrupt request of the comparator. |
| <b>Writing "1"</b> | Enables the interrupt request of the comparator.  |

[bit2] VCID: Comparator analog input disable bit

This bit enables or disables comparator analog input.

| bit2               | Details                           |
|--------------------|-----------------------------------|
| <b>Writing "0"</b> | Enables comparator analog input.  |
| <b>Writing "1"</b> | Disables comparator analog input. |

Notes:

- In the case of a pin possessing the non-inverting analog input (positive input) function, the general-purpose I/O port function of that pin is disabled when the analog input function is enabled by this bit.
- In the case of a pin possessing the inverting analog input (negative input) function, the general-purpose I/O port function of that pin is disabled when the analog input function is enabled by this bit and the external pin is selected as the inverting analog input (negative input) voltage source (CMRn:BGRS = 0).



## MB95850K/860K/870K Series

[bit1] VCOE: Comparator output enable bit

This bit enables or disables comparator output.

| bit1               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables comparator output. The output pin of the comparator is used as general-purpose I/O port. |
| <b>Writing "1"</b> | Enables comparator output.  |

[bit0] PD: Comparator power down control bit

This bit powers up or down the comparator.

| bit0               | Details                     |
|--------------------|-----------------------------|
| <b>Writing "0"</b> | Powers up the comparator.   |
| <b>Writing "1"</b> | Powers down the comparator. |



# **CHAPTER 26**

---

# **TOUCH SENSOR**

**This chapter describes the functions and operations of the touch sensor.**

- 26.1 Overview
- 26.2 Configuration
- 26.3 Pins
- 26.4 Interrupts
- 26.5 Operations and Setting Procedure Example
- 26.6 Registers

## 26.1 Overview

---

**The touch sensor (TS) controls touch keys and touch panels for home appliances and consumer electronic products, and computer input devices such as mice and keyboards.**

---

### ■ Features of TS

The TS has the following features.

- It has the configurable Automatic Impedance Calibration (AIC™)\* function. The calibration interval is adjustable.
  - Different sensitivities can be set for individual channels.
  - It has the Adjacent Pattern Interference Suppression (APIS™)\* function, which is a filtering function that eliminates adjacent key or pattern interference. APIS uses touch strength information measured in a sampler. There are three modes in APIS:
    - APIS mode 1 : takes the strongest one from all touch inputs whose strength exceeds a pre-defined strength threshold.
    - APIS mode 2 : takes all touch inputs exceeding whose strength exceeds a pre-defined strength threshold, without a limit on the number of simultaneous sensing operations.
    - APIS mode 3 : takes up to the two strongest ones from all touch inputs whose strength exceeds a predefined strength threshold, with the number of simultaneous sensing operations limited to two.
- \*: AIC™ and APIS™ are registered trademarks of ATLab, Inc., South Korea.
- It has two types of interrupts: GINT (general interrupt) and TINT (touch interrupt).
  - It can read touch strength.
  - It has an idle mode for power saving.
  - It has configurable direct touch output pins (DIO pins). When the touch pad of a touch channel is touched, its corresponding DIO pin lights up an LED.
  - It has programmable registers to characterize applications.
  - It generates beep for tactile feeling. For details, see "CHAPTER 27 BEEP OUTPUT UNIT".

## MB95850K/860K/870K Series

### 26.2 Configuration

The TS consists of the following blocks:

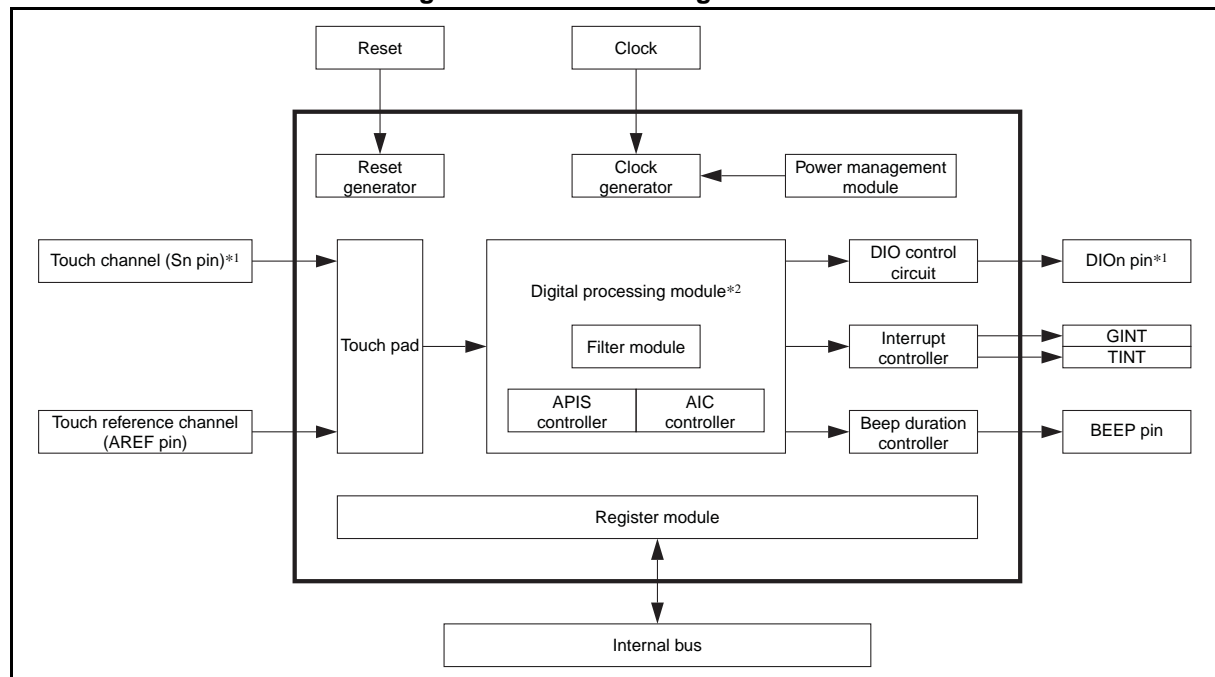
- Reset generator
- Clock generator
- Power management module
- Touch pad
- Digital processing module
- DIO control circuit
- Interrupt control module
- Beep duration controller
- AIC controller
- APIS controller
- Filter module
- Register module

The number of pins, that of touch channels and that of direct output (DIO) channels of the TS vary among products. For details, refer to the device data sheet.

In this chapter, "n" in a pin name and a register abbreviation represents the touch channel number and the direct output channel number. For details of pin names, register names and register abbreviations of a product, refer to the device data sheet.

#### ■ Block Diagram of TS

Figure 26.2-1 Block Diagram of TS



\*1: The number of Sn pins (touch channel input pins) and that of DIO pins (direct output channel pins) of the TS vary among products. For details, refer to the device data sheet.

\*2: For details of this module, see Figure 26.6-2.

- Reset generator

The TS has two resets: chip reset and warm reset. The chip reset initializes all registers to their respective initial values when resetting the TS, while the warm reset maintains the current values of all registers except the INTCR register, the INTPR register, the CALIPn register and the IMPEn register when resetting the TS. In the case of modifying register values and applying the modified values immediately, use the warm reset.

- Clock generator

The clock generator selects an input sensor clock and controls whether the TS runs or stops in stop mode and watch mode.

- Power management module

The power management module makes the TS enter idle mode through the IDLETM register and the CONTROL register.

- Touch pad

The touch pad receives touch channel impedance changes and outputs such changes to the digital processing module.

- Digital processing module

The digital processing module processes touch channel impedance changes according to settings of related registers.

- DIO control circuit

The DIO control circuit controls the relations between touch channels and DIO channels.

- Interrupt controller

The interrupt controller generates interrupts. There are two types of interrupts: GINT for general purpose and TINT for touch detection. The GINT consists of new impedance interrupt, EOC (End Of Calibration) interrupt, IDLE-to-ACTIVE interrupt, ACTIVE-to-IDLE interrupt and TINT.

- Beep duration controller

The beep duration controller selects through the BEEP\_DUR register the time during which beep wave is being output when the touch pad is touched.

- AIC controller

The AIC controller controls the automatic impedance calibration function.

- APIS controller

The APIS controller selects an APIS mode through the FTSEL register.

- Filter module

The filter module stabilizes APIS output and reduces the touch interrupt rate.

- Register module

The register module contains all TS registers, which can be accessed through the internal bus.

- **Input Clock**

The TS uses a clock output from the touch prescaler (selected through the PSC register) as the sensor clock.

The TS uses the machine clock (MCLK) for the read operation and write operation of registers.

## 26.3 Pins

---

**This section describes the pins of the TS.**

---

### ■ Pins of TS

- Sn pin

This pin serves as a touch channel input pin.

The TS processes impedance changes on an Sn pin. To use an Sn pin as a touch channel input pin, write "0" to the bit in the port direction register (DDR<sub>x</sub>) corresponding to that Sn pin, and enable the touch channel corresponding to that Sn pin by writing "1" to the ENS<sub>n</sub> bit in the TER0/TER1 register corresponding to that touch channel.

- DIO<sub>n</sub> pin

This pin serves as a direct output channel pin.

A DIO<sub>n</sub> pin outputs the status of the touch channel. An "L" level is output to the DIO<sub>n</sub> pin when the touch pad of a touch channel is touched; an "H" level is output to the DIO<sub>n</sub> pin when the touch pad of a touch channel is not touched. The DIO<sub>n</sub> pin outputs a DIO signal indicating the status of a touch channel. Use the DIO<sub>n</sub>S[3:0] bits in the DIOR1/DIOR2/DIOR3 register to select the touch channel whose status is to be output to the DIO<sub>n</sub> pin. For details of the DIOR1, DIOR2 and DIOR3 registers, see "26.6.13 TS Direct Output Control Register 1 (DIOR1)", "26.6.14 TS Direct Output Control Register 2 (DIOR2)" and "26.6.15 TS Direct Output Control Register 3 (DIOR3)".

- AREF pin

This pin serves as a touch reference impedance input pin.

To use the AREF pin, write "0" to the bit in the port direction register (DDR<sub>x</sub>) corresponding to the AREF pin, and enable the touch reference channel by writing "1" to the EN\_REF bit in the TER1 register.



## MB95850K/860K/870K Series

### 26.4 Interrupts

The TS generates two types of interrupts: GINT (general interrupt) and TINT (touch interrupt).

- **GINT (general interrupt)**

GINT consists of five interrupts: new impedance interrupt, EOC interrupt, IDLE-to-ACTIVE interrupt, ACTIVE-to-IDLE interrupt and TINT. All GINT can be masked. The INTPR register indicates the GINT source.

- **TINT (touch interrupt)**

A TINT occurs when a touch pad is pressed or released. A TINT is automatically cleared when the TS touch data register (upper) (TOUCHH) and the TS touch data register (lower) (TOUCHL) are read.

#### ■ Details of TINT

A TINT occurs when a touch pad is pressed or released (when there is any change in the values of the TOUCHH and TOUCHL registers). A TINT is automatically cleared when the TOUCHH and TOUCHL registers are read. The TS cannot mask the TINT.

Note:

To mask the TINT (not process the TINT), write "0b11" to the L07[1:0] bits in the ILR1 register. For details of the ILR1 register, see "CHAPTER 5 INTERRUPTS".

#### ■ Details of GINT

Table 26.4-1 shows details of the registers and bits related to the GINT.

**Table 26.4-1 Details of Registers and Bits Related to GINT**

| Item  | Details                                  |                   |  |  |   |
|---|--|-------------------|--|--|---|
|   | New impedance                            | EOC               | IDLE-to-ACTIVE                                 | ACTIVE-to-IDLE                                 | TINT  |
| Interrupt source  | The value of the IMPEN register changes. | Calibration ends. | The TS transits from idle mode to active mode. | The TS transits from active mode to idle mode. | A touch pad is pressed or released. In other words, there is a change in the values of the TOUCHH and TOUCHL registers. |
| Interrupt mask bit in the TS interrupt mask register (INTMR)    | NEW                                      | EOC               | ITA  | ATI  | TINT  |
| Interrupt clear bit in the TS interrupt clear register (INTCR)  | NEW                                      | EOC               | ITA  | ATI  | TINT  |
| Interrupt flag bit in the TS interrupt pending register (INTPR) | NEW                                      | EOC               | ITA  | ATI  | TINT  |

- New impedance interrupt

When the value of the IMPEN register changes, the new impedance interrupt flag bit in the TS interrupt pending register (INTPR:NEW) is set to "1", and a new impedance interrupt request is made to the interrupt controller provided that the masking of the new impedance interrupt has been disabled (INTMR:NEW = 0). Writing "1" to the new impedance interrupt clear bit in the TS interrupt clear register (INTCR:NEW) clears the new impedance interrupt request.

- EOC (End Of Calibration) interrupt

When the TS finishes calibration, the EOC interrupt flag bit in the TS interrupt pending register (INTPR:EOC) is set to "1", and an EOC interrupt request is made to the interrupt controller provided that the masking of the EOC interrupt has been disabled (INTMR:EOC = 0). Writing "1" to the EOC interrupt clear bit in the TS interrupt clear register (INTCR:EOC) clears the EOC interrupt request.

- IDLE-to-ACTIVE interrupt

When the TS transits from idle mode to active mode, the IDLE-to-ACTIVE interrupt flag bit in the TS interrupt pending register (INTPR:ITA) is set to "1", and an IDLE-to-ACTIVE interrupt request is made to the interrupt controller provided that the masking of the IDLE-to-ACTIVE interrupt has been disabled (INTMR:ITA = 0). Writing "1" to the IDLE-to-ACTIVE interrupt clear bit in the TS interrupt clear register (INTCR:ITA) clears the IDLE-to-ACTIVE interrupt request.

- ACTIVE-to-IDLE interrupt

When the TS transits from active mode to idle mode, the ACTIVE-to-IDLE interrupt flag bit in the TS interrupt pending register (INTPR:ATI) is set to "1", and an ACTIVE-to-IDLE interrupt request is made to the interrupt controller provided that the masking of the ACTIVE-to-IDLE interrupt has been disabled (INTMR:ATI = 0). Writing "1" to the ACTIVE-to-IDLE interrupt clear bit in the TS interrupt clear register (INTCR:ATI) clears the ACTIVE-to-IDLE interrupt request.

- TINT

When a touch pad is touched, the TINT flag bit in the TS interrupt pending register (INTPR:TINT) is set to "1", and a TINT request is made to the interrupt controller provided that the masking of the TINT has been disabled (INTMR:TINT = 0). Reading the TOUCHH and TOUCHL registers and then writing "1" to the TINT clear bit in the TS interrupt clear register (INTCR:TINT) clears the TINT request.

---

**Note:**

The warm reset can initialize the INTPR register and the INTCR register. The minimum TINT interval\* depends on the respective settings of the ITGTM register and the FLTP register.

---

\*: Minimum TINT interval = Setting of ITGTM × Setting of FLTP × SNCLKs

## 26.5 Operations and Setting Procedure Example

The TS has an impedance change detection (ICD) engine, which detects impedance difference between reference input and sensor input.

### ■ Touch Detection

Figure 26.5-1 When a Touch Pad Is Not Touched (No-touch State)

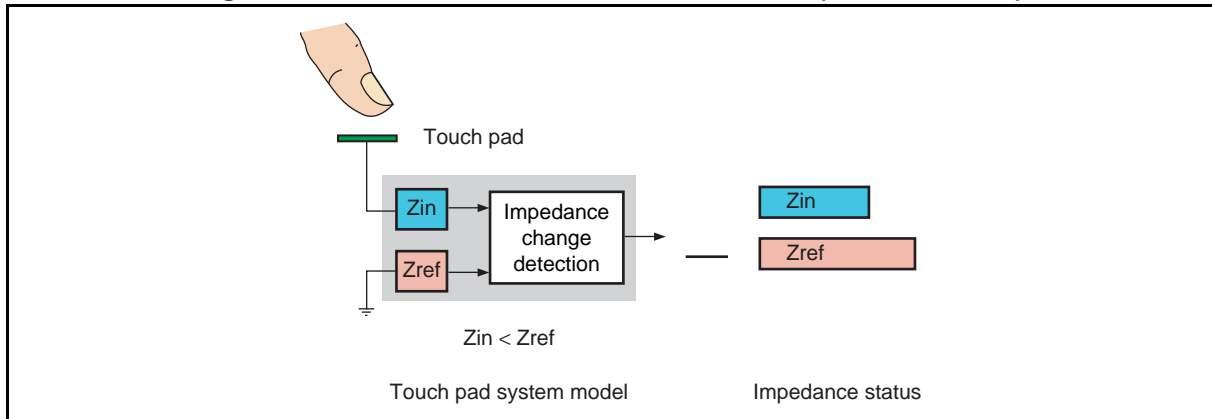
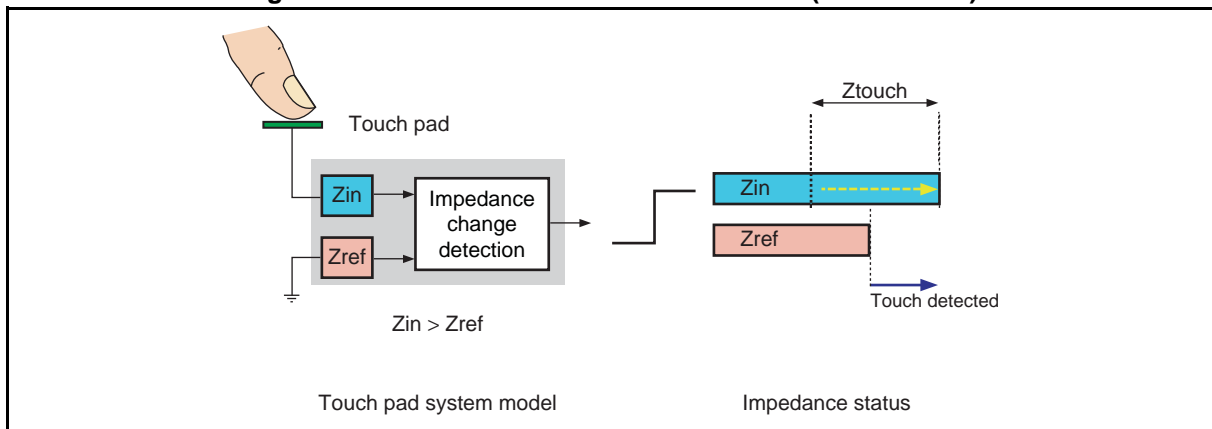


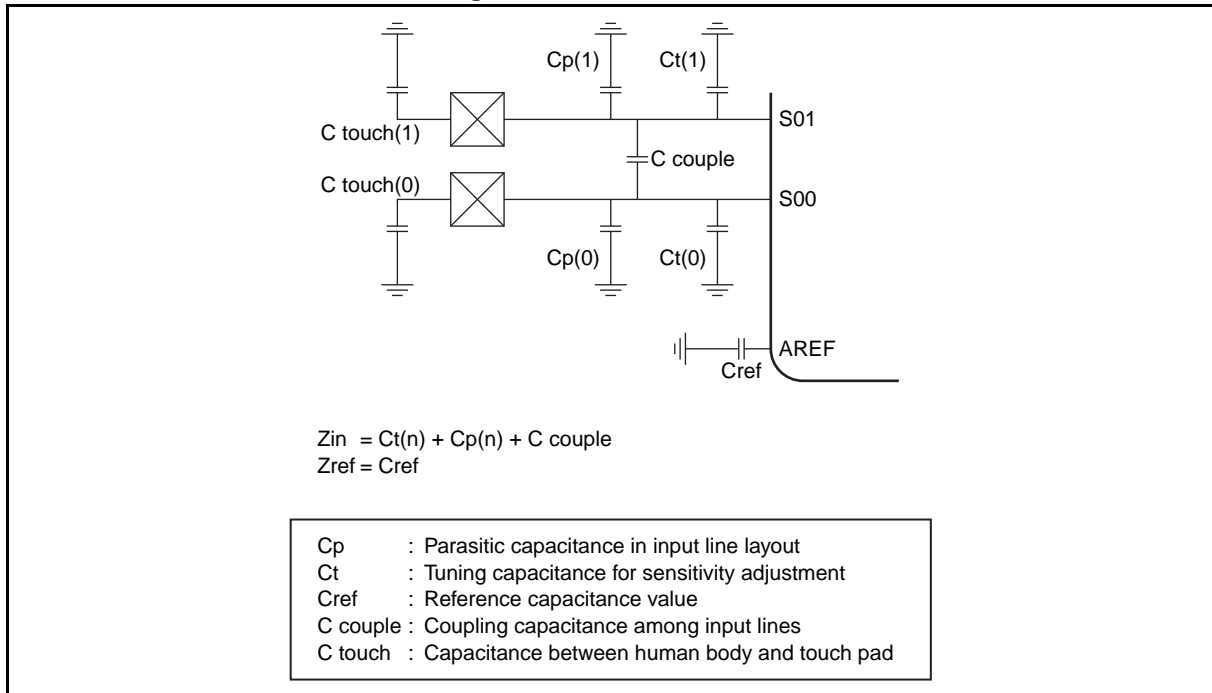
Figure 26.5-2 When a Touch Pad Is Touched (Touch State)



As shown in Figure 26.5-1, when the touch pad is not touched, the impedance of the sensor input  $Z_{in}$  should be smaller than that of the reference input  $Z_{ref}$ . As shown in Figure 26.5-2, when the touch pad is touched,  $Z_{in}$  increases by  $Z_{touch}$ . When  $Z_{touch}$  becomes greater than the difference between  $Z_{in}$  and  $Z_{ref}$  in no-touch state, the ICD engine in the TS generates the acknowledged output signal.

$Z_{in}$  consists of parasitic capacitance of the input line, tuning capacitance of the input pin, and on-chip input impedance, while  $Z_{ref}$  includes on-chip impedance and AIC control values. External tuning capacitance can be added to  $Z_{ref}$  if necessary. For details, see Figure 26.5-3.

**Figure 26.5-3 Zin and Zref**



■ **AIC™ (Automatic Impedance Calibration)**

AIC maintains consistent sensitivity against external environmental changes such as temperature, supply voltage and current, humidity, and system-level variations. This helps users develop their applications more conveniently by providing the actual impedance value of each sensor input.

The ICD engine residing in the TS controls reference impedance values for each sensor input pin by acquiring each input impedance data. It periodically updates all reference impedance values under the condition that all twelve touch pads remain in no-touch state. This auto-calibration function absorbs environmental changes and guarantees product stability.

■ APIS™ Touch Output

When touch pads are arranged too closely to each other, it is sometimes difficult to identify which pad is touched. APIS™ (Adjacent Pattern Interference Suppression) is a filtering function to identify which pads are intentionally touched. If no APIS mode is defined, all touch data without APIS filtering are transmitted to the MCU. For example, if the application is a numeric keypad, the user can use APIS mode 1 to get the strongest output and filter out all other weakly touched inputs.

There are three modes in APIS:

- APIS mode 1 : reports only the strongest output among outputs exceeding the threshold defined in the STRTHn. See Figure 26.5-4.
- APIS mode 2 : reports all outputs that exceed the threshold defined in the STRTHn register. See Figure 26.5-5.
- APIS mode 3 : reports the two strongest outputs among outputs exceeding the threshold defined in the STRTHn register (suitable for multi-touch applications). See Figure 26.5-6.

Figure 26.5-4 APIS Mode 1

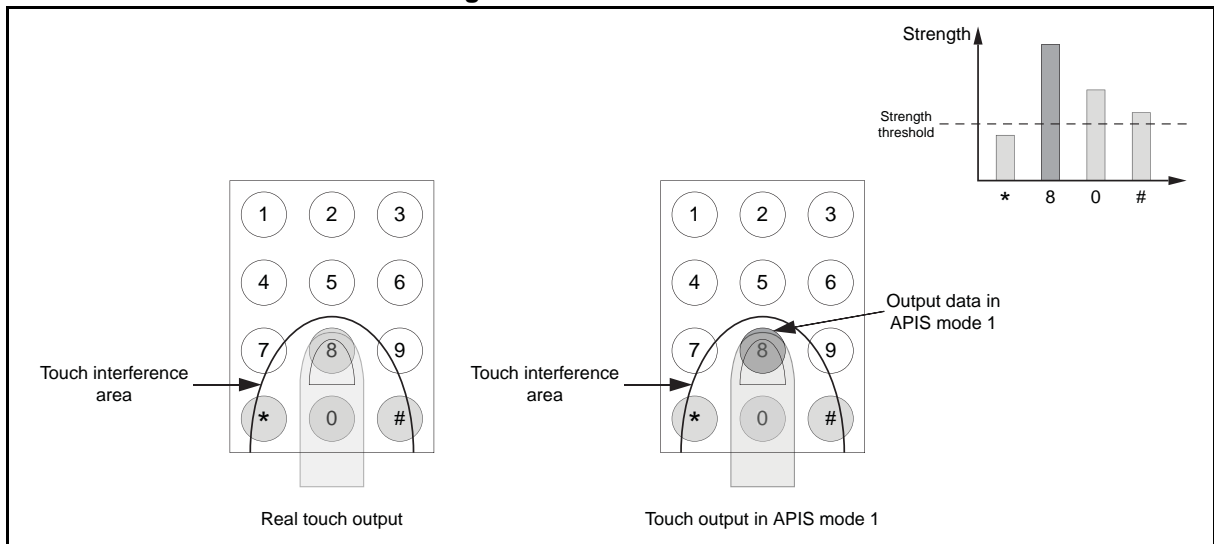
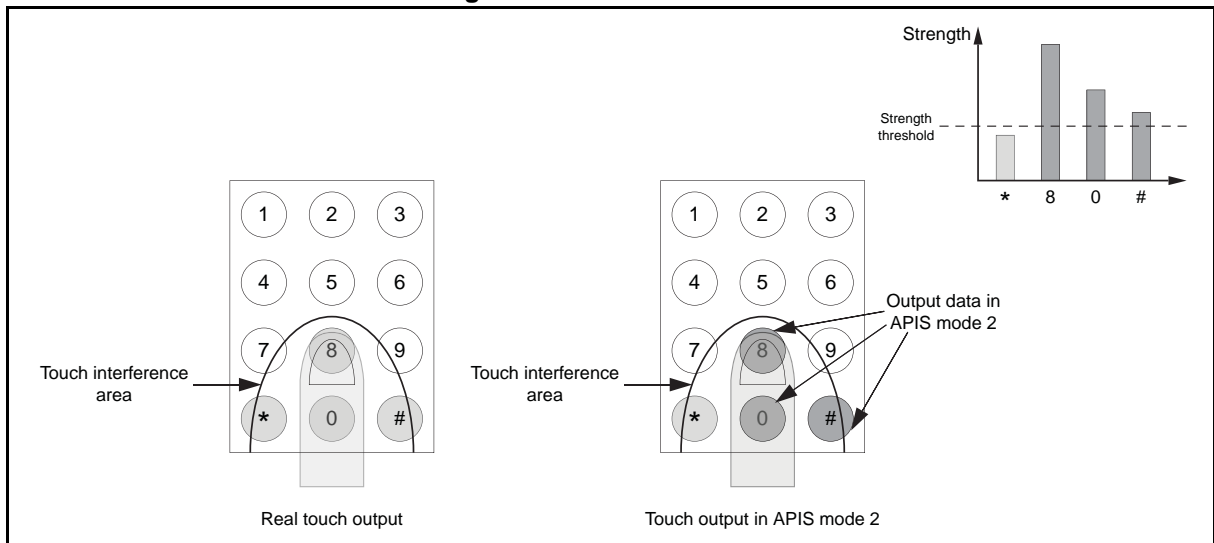
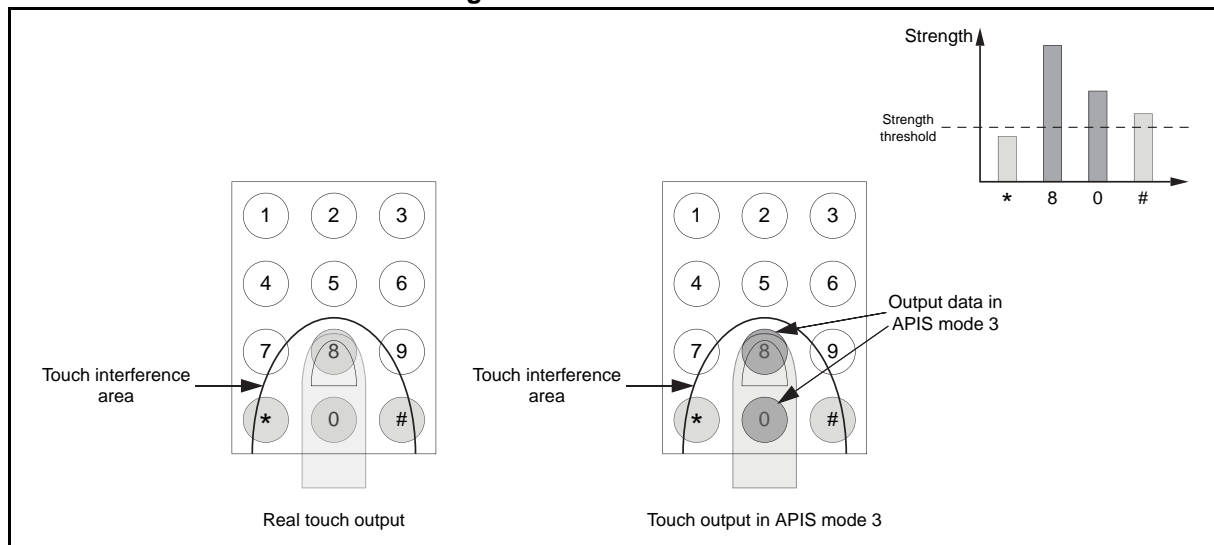


Figure 26.5-5 APIS Mode 2



**Figure 26.5-6 APIS Mode 3**



■ Operations of TS

- Measuring impedance by comparing rising edge phase difference between SNCLK of AREF pin and SNCLK of Sn pin

The impedance is measured by comparing the rising edge phase difference between the SNCLK of the AREF pin (AREF SNCLK) and the SNCLK of the Sn pin (Sn SNCLK). The rising edge phase of Sn SNCLK depends on the impedance of an electrode. The larger the impedance of an electrode is, the longer the delay of Sn SNCLK becomes. The reference delay depends on the value of the REFDLY register and on whether there is an external capacitor on the AREF pin. The rising edge phase difference between AREF SNCLK and Sn SNCLK is measured by the delay chain.

When the rising edge phase difference exceeds the range that the delay chain can measure, use the reference delay to keep the rising edge phase difference within the range. For details, see Figure 26.5-7 and Figure 26.5-8.

Figure 26.5-7 Waveform for Measuring Impedance

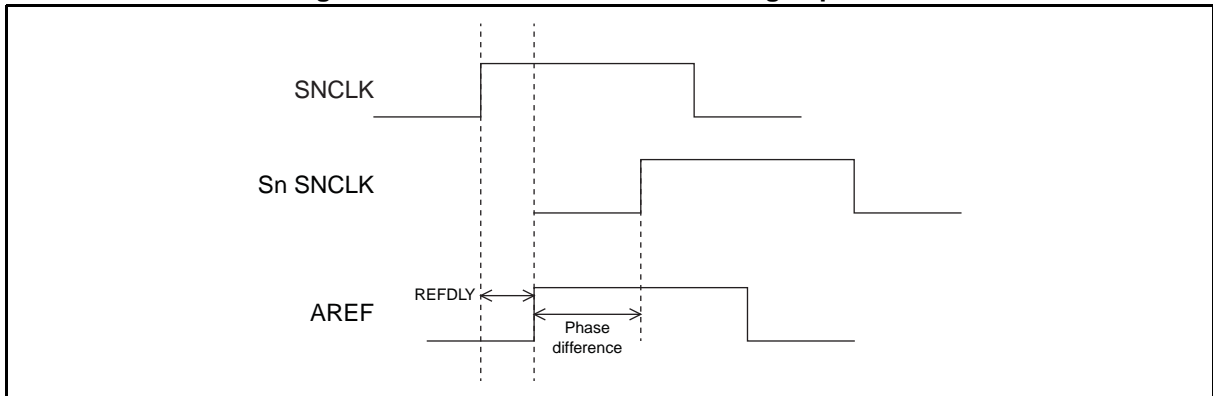
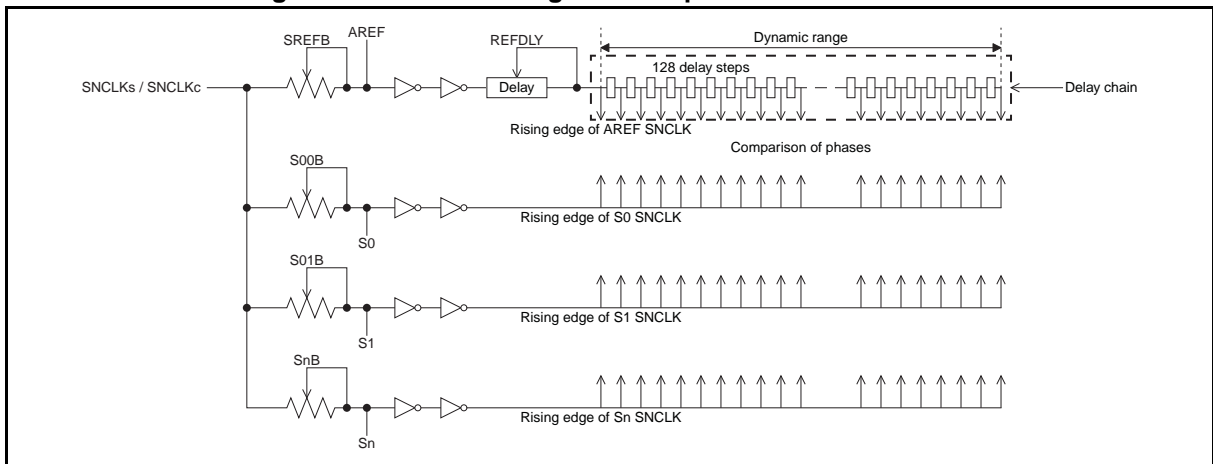


Figure 26.5-8 Block Diagram of Impedance Measurement



Note:

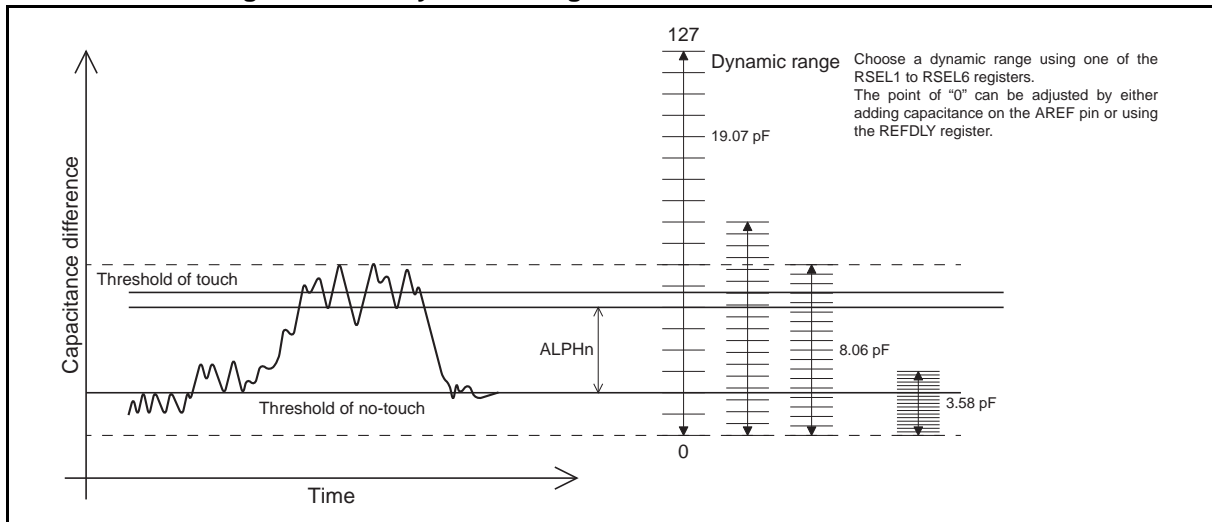
The IMPE<sub>n</sub> register keeps the value of impedance of an electrode. According to the circuit operation principle, the increase in impedance of an electrode is measured as a decrease in the value of the IMPE<sub>n</sub> register and vice versa. The actual impedance can be obtained by subtracting the value of the IMPE<sub>n</sub> register from 127. The actual calibrated impedance can be obtained by subtracting the value of the CALIP<sub>n</sub> register from 127.

● Dynamic range and touch detection threshold

Selecting a proper dynamic range is important in the operation of the TS. In the example illustrated in Figure 26.5-9, the dynamic range of 8.06 pF is relatively more suitable to measure capacitance completely and accurately. The ALPH<sub>n</sub> register controls the threshold of touch and the threshold of no-touch.

It is recommended that the initial value of the ALPH<sub>n</sub> register be set to a value equivalent to 80% of the average capacitance difference between a touch and a no-touch. When an APIS mode is used, the dynamic range to be used is determined by the STR<sub>n</sub> register and the STRTH<sub>n</sub> register.

**Figure 26.5-9 Dynamic Range and Touch Detection Threshold**

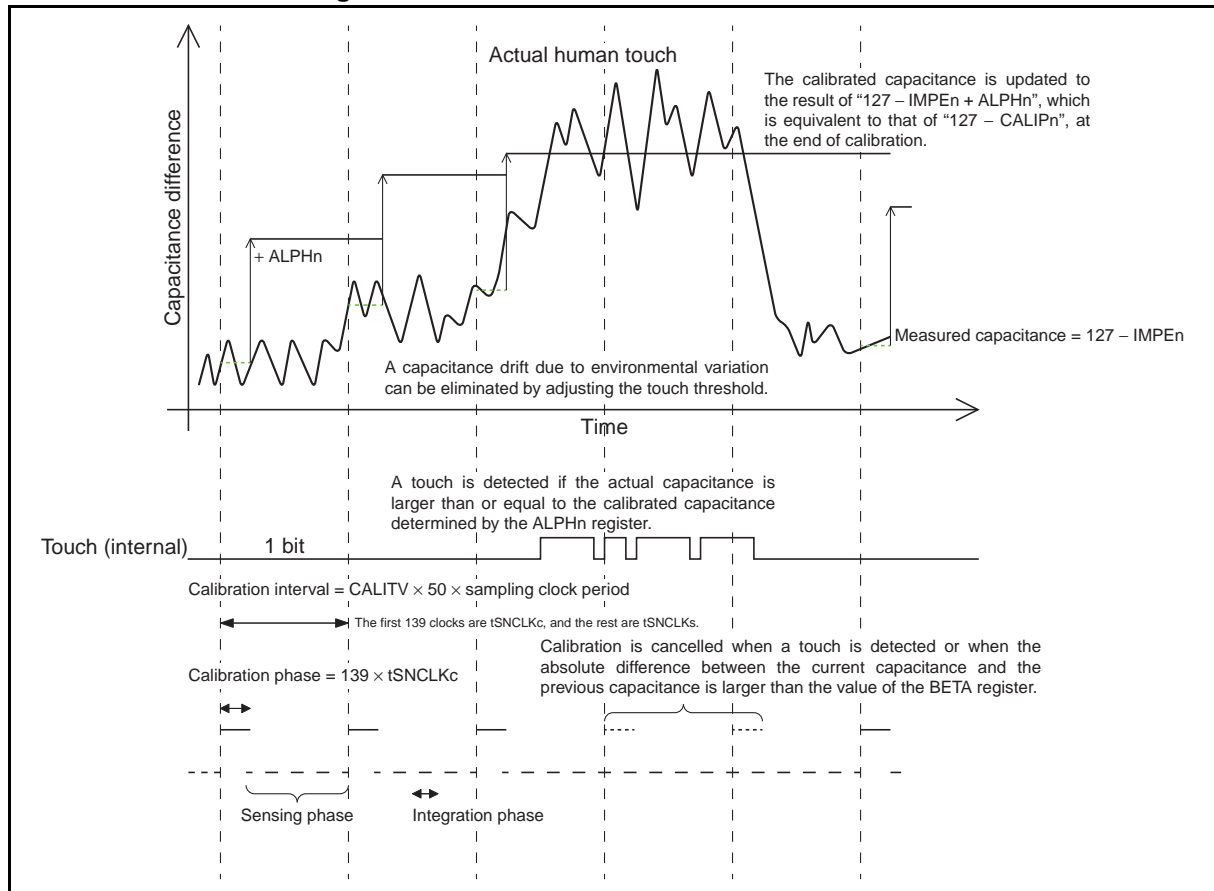




● Touch detection and calibration

Figure 26.5-10 illustrates the operations of touch detection and calibration work.

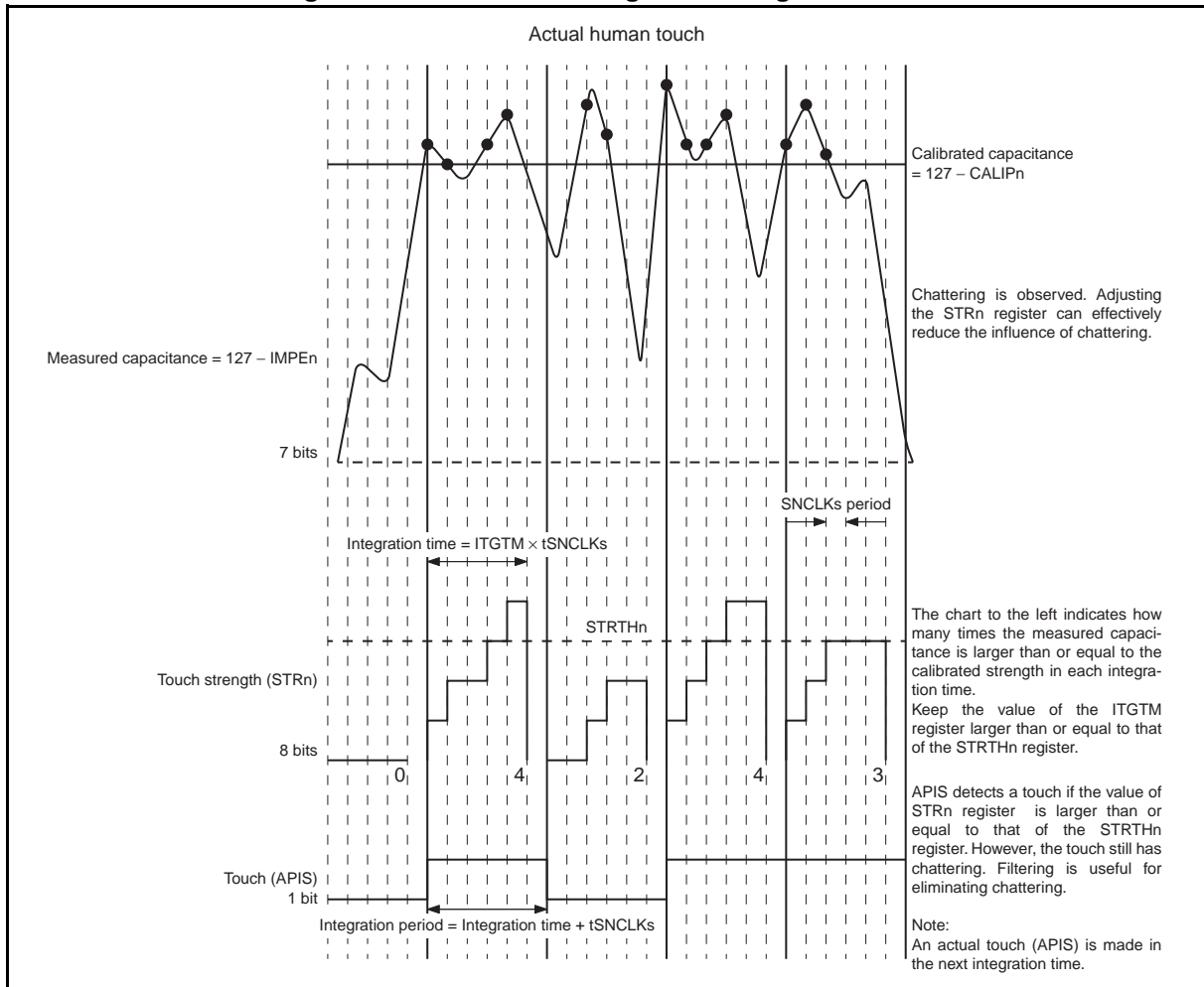
Figure 26.5-10 Touch Detection and Calibration



● Touch strength and integration time

Figure 26.5-11 illustrates details of touch strength and integration time.

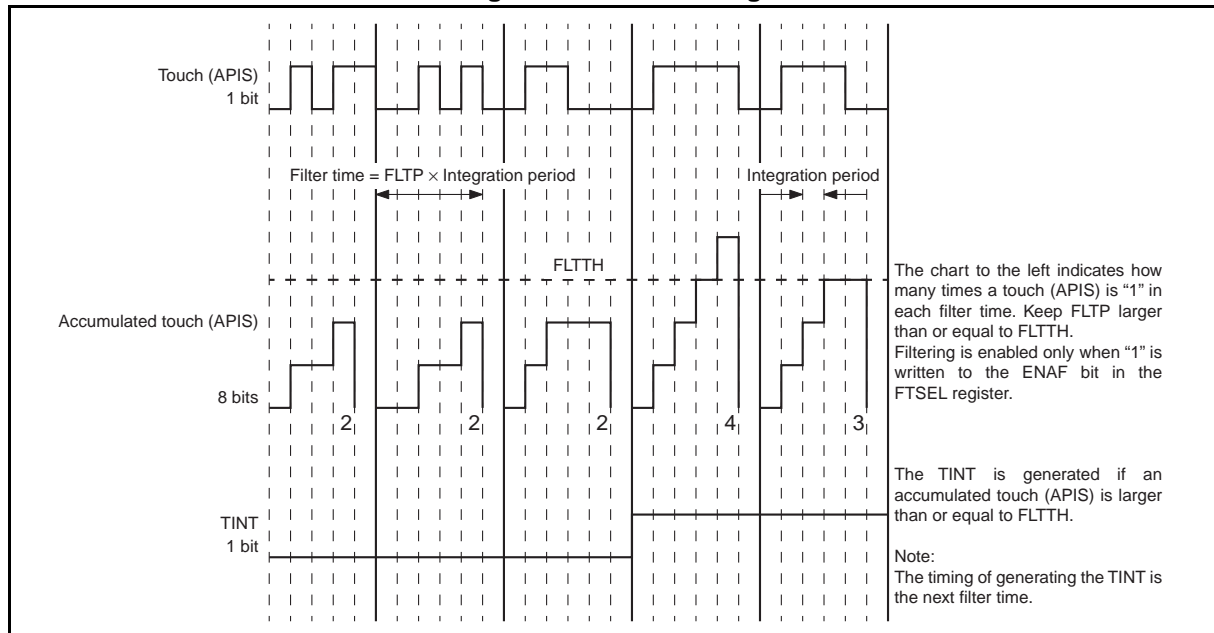
**Figure 26.5-11 Touch Strength and Integration Time**



● Filtering

Figure 26.5-12 illustrates details of filtering.

Figure 26.5-12 Filtering



### ■ Setting Procedure Example

Below is an example of procedure for setting the TS.

#### ● Initial settings

1. Set the input port. (DDR)
2. Set the interrupt level. (ILR\*<sup>1</sup>)
3. Enable touch input. (TIDR0/TIDR1)
4. Enable a touch channel. (TER0/TER1\*<sup>2</sup>)
5. Set necessary touch sensor parameters such as APIS mode (FTSEL), alpha value (ALPHn), etc.
6. Set the parameters of the beep function. (BPDUR and BPFREQ)
7. Set the direct output function. (DIOR1/DIOR2/DIOR3)
8. Enable interrupts. (GINT\*<sup>3</sup>)
9. Set the prescaler control register. (PSC)
10. Generate a warm reset. (WRESET:WRER[7:0] = 0b01011010)

\*1: For details of the interrupt level setting register (ILR), refer to "CHAPTER 5 INTERRUPTS" in this hardware manual and "■ INTERRUPT SOURCE TABLE" in the device data sheet.

\*2: Always enable the reference channel of the TS (TER1:EN\_REF = 1, TIDR0:bit3 = 0) when using any touch channel.

\*3: The TINT is always enabled and cannot be masked by the TS.

#### ● Switching touch channels

1. Disable the channel in use. (TER0/TER1:ENS<sub>n</sub> = 0)
2. Enable the desired channel. (TER0/TER1:ENS<sub>n</sub> = 1)

---

#### Notes:

- Use the warm reset to restart the TS after switching touch channels.
  - It is recommended that a warm reset be generated after the settings of any TS register have been modified.
- 

#### ● Interrupt processing

1. Check the interrupt source. (INTPR)
2. Mask all interrupts. (INTMR = 0b00011111)
3. Execute the interrupt service routine.
4. Clear the bit in the INTPR register corresponding to that interrupt source to "0".
5. Unmask the interrupt source the INTPR register indicates by writing "0" to the bit in the INTMR register corresponding to that interrupt source.

## 26.6 Registers

This section describes the registers of the TS.

**Table 26.6-1 List of TS Registers**

| Register abbreviation | Register name                                      | Reference |
|-----------------------|--|-----------|
| TER0                  | TS touch channel enable register 0                 | 26.6.1    |
| TER1                  | TS touch channel enable register 1                 | 26.6.2    |
| PSC                   | TS prescaler control register                      | 26.6.3    |
| WRESET                | TS warm reset register                             | 26.6.4    |
| RSEL0                 | TS sensitivity select register 0                   | 26.6.5    |
| RSEL1                 | TS sensitivity select register 1                   | 26.6.6    |
| RSEL2                 | TS sensitivity select register 2                   | 26.6.7    |
| RSEL3                 | TS sensitivity select register 3                   | 26.6.8    |
| RSEL4                 | TS sensitivity select register 4                   | 26.6.9    |
| RSEL5                 | TS sensitivity select register 5                   | 26.6.10   |
| RSEL6                 | TS sensitivity select register 6                   | 26.6.11   |
| BPDUR                 | TS beep duration setting register                  | 26.6.12   |
| DIOR1                 | TS direct output control register 1                | 26.6.13   |
| DIOR2                 | TS direct output control register 2                | 26.6.14   |
| DIOR3                 | TS direct output control register 3                | 26.6.15   |
| FTSEL                 | TS feature select register                         | 26.6.16   |
| AICWAT                | TS AIC wait time setting register                  | 26.6.17   |
| CALITV                | TS calibration interval setting register           | 26.6.18   |
| ITGTM                 | TS integration time setting register               | 26.6.19   |
| IDLETM                | TS idle time setting register                      | 26.6.20   |
| CONTROL               | TS control register                                | 26.6.21   |
| INTMR                 | TS interrupt mask register                         | 26.6.22   |
| INTCR                 | TS interrupt clear register                        | 26.6.23   |
| INTPR                 | TS interrupt pending register                      | 26.6.24   |
| FLTP                  | TS filter period setting register                  | 26.6.25   |
| FLTTH                 | TS filter threshold setting register               | 26.6.26   |
| REFDLY                | TS reference delay setting register                | 26.6.27   |
| ALPHn                 | TS alpha value setting register ch. n              | 26.6.28   |
| BETA                  | TS beta value setting register                     | 26.6.29   |
| STRTHn                | TS touch strength threshold setting register ch. n | 26.6.30   |
| STRn                  | TS touch strength register ch. n                   | 26.6.31   |
| CALIPn                | TS calibrated impedance register ch. n             | 26.6.32   |
| IMPEn                 | TS impedance register ch. n                        | 26.6.33   |
| TOUCHL                | TS touch data register (lower)                     | 26.6.34   |
| TOUCHH                | TS touch data register (upper)                     | 26.6.35   |

## 26.6.1 TS Touch Channel Enable Register 0 (TER0)

The TS touch channel enable register 0 (TER0) enables or disables touch ch. 0 to touch ch. 7.

### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | ENS07 | ENS06 | ENS05 | ENS04 | ENS03 | ENS02 | ENS01 | ENS00 |
| Attribute     | R/W   | R/W   | R     | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

### ■ Register Functions

[bit7] ENS07: Touch ch. 7 enable bit

This bit enables or disables touch ch. 7.

| bit7               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 7. |
| <b>Writing "1"</b> | Enables touch ch. 7.  |

[bit6] ENS06: Touch ch. 6 enable bit

This bit enables or disables touch ch. 6.

| bit6               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 6. |
| <b>Writing "1"</b> | Enables touch ch. 6.  |

[bit5] ENS05: Touch ch. 5 enable bit

This bit enables or disables touch ch. 5.

| bit5               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 5. |
| <b>Writing "1"</b> | Enables touch ch. 5.  |

[bit4] ENS04: Touch ch. 4 enable bit

This bit enables or disables touch ch. 4.

| bit4               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 4. |
| <b>Writing "1"</b> | Enables touch ch. 4.  |

[bit3] ENS03: Touch ch. 3 enable bit

This bit enables or disables touch ch. 3.

| bit3               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 3. |
| <b>Writing "1"</b> | Enables touch ch. 3.  |

## MB95850K/860K/870K Series

[bit2] ENS02: Touch ch. 2 enable bit

This bit enables or disables touch ch. 2.

| bit2               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 2. |
| <b>Writing "1"</b> | Enables touch ch. 2.  |

[bit1] ENS01: Touch ch. 1 enable bit

This bit enables or disables touch ch. 1.

| bit1               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 1. |
| <b>Writing "1"</b> | Enables touch ch. 1.  |

[bit0] ENS00: Touch ch. 0 enable bit

This bit enables or disables touch ch. 0.

| bit0               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 0. |
| <b>Writing "1"</b> | Enables touch ch. 0.  |

---

Notes:

- Always enable the reference channel of the TS (TER1:EN\_REF = 1, TIDR0:bit3 = 0) when using any touch channel.
  - The number of touch channels varies among products. For details, refer to the device data sheet.
-

## 26.6.2 TS Touch Channel Enable Register 1 (TER1)

The TS touch channel enable register 1 (TER1) enables or disables the touch reference channel and touch ch. 8 to touch ch. 11.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4      | 3     | 2     | 1     | 0     |
|---------------|---|---|---|--------|-------|-------|-------|-------|
| Field         | — | — | — | EN_REF | ENS11 | ENS10 | ENS09 | ENS08 |
| Attribute     | — | — | — | R/W    | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0 | 0      | 0     | 0     | 0     | 0     |

### ■ Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4] EN\_REF: Touch reference channel enable bit

This bit enables or disables the touch reference channel.

| bit4               | Details                               |
|--------------------|---------------------------------------|
| <b>Writing "0"</b> | Disables the touch reference channel. |
| <b>Writing "1"</b> | Enables the touch reference channel.  |

#### [bit3] ENS11: Touch ch. 11 enable bit

This bit enables or disables touch ch. 11.

| bit3               | Details                |
|--------------------|------------------------|
| <b>Writing "0"</b> | Disables touch ch. 11. |
| <b>Writing "1"</b> | Enables touch ch. 11.  |

#### [bit2] ENS10: Touch ch. 10 enable bit

This bit enables or disables touch ch. 10.

| bit2               | Details                |
|--------------------|------------------------|
| <b>Writing "0"</b> | Disables touch ch. 10. |
| <b>Writing "1"</b> | Enables touch ch. 10.  |

#### [bit1] ENS09: Touch ch. 9 enable bit

This bit enables or disables touch ch. 9.

| bit1               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 9. |
| <b>Writing "1"</b> | Enables touch ch. 9.  |

#### [bit0] ENS08: Touch ch. 8 enable bit

This bit enables or disables touch ch. 8.

| bit0               | Details               |
|--------------------|-----------------------|
| <b>Writing "0"</b> | Disables touch ch. 8. |
| <b>Writing "1"</b> | Enables touch ch. 8.  |



---

Notes:

- Always enable the reference channel of the TS (TER1:EN\_REF = 1, TIDR0:bit3 = 0) when using any touch channel.
  - The number of touch channels varies among products. For details, refer to the device data sheet.
-

### 26.6.3 TS Prescaler Control Register (PSC)

The TS prescaler control register (PSC) selects the sensor clock frequency (SNCLK) and determines whether the TS can operate in stop mode and watch mode.

#### ■ Register Configuration

| bit           | 7 | 6 | 5    | 4    | 3       | 2       | 1       | 0       |
|---------------|---|---|------|------|---------|---------|---------|---------|
| Field         | — | — | TRDY | STPE | DIVSEL3 | DIVSEL2 | DIVSEL1 | DIVSEL0 |
| Attribute     | — | — | R    | R/W  | R/W     | R/W     | R/W     | R/W     |
| Initial value | 0 | 0 | 1    | 0    | 0       | 0       | 0       | 0       |

#### ■ Register Functions

##### [bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit5] TRDY: TS ready bit

This bit indicates whether the registers of the TS can be accessed.

| bit5               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the registers of the TS cannot be accessed. |
| <b>Reading "1"</b> | Indicates that the registers of the TS can be accessed.    |

Note: After waking up the TS from stop mode or watch mode, ensure that this bit is reading "1" before accessing any other register of the TS.

##### [bit4] STPE: Stop enable bit

This bit determines whether the TS can operate in stop mode and watch mode. For details, see Table 3.1-4 in "CHAPTER 3 CLOCK CONTROLLER".

| bit4               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the TS to operate in stop mode and watch mode. |
| <b>Writing "1"</b> | Enables the TS to operate in stop mode and watch mode.  |

## MB95850K/860K/870K Series

[bit3:0] DIVSEL[3:0]: Divider select bits

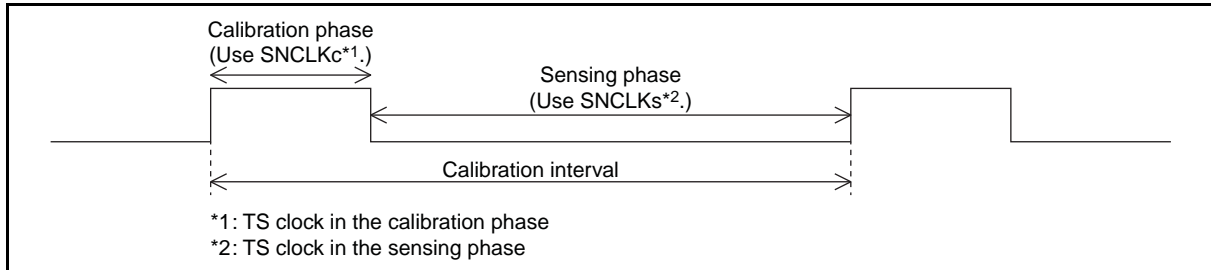
These bits select a divider for the SNCLK frequency of the TS.

The calibration interval comprises a calibration phase and a sensing phase as shown in Figure 26.6-1.

In the calibration phase, the TS automatically executes impedance calibration or a host MCU executes impedance calibration. In the sensing phase, the TS generates touch output.

The calibration phase and the sensing phase may have different SNCLK frequencies.

**Figure 26.6-1 Calibration Interval**



| bit3:0         | Details<br>(MCLK: machine clock)   |
|----------------|--|
| Writing "0000" | Disables the use of SNCLK.   |
| Writing "0001" |  |
| Writing "0010" | SNCLKs frequency = MCLK/2<br>SNCLKc frequency = MCLK/2                             |
| Writing "0011" | SNCLKs frequency = MCLK/2 <sup>2</sup><br>SNCLKc frequency = MCLK/2                |
| Writing "0100" | SNCLKs frequency = MCLK/2 <sup>3</sup><br>SNCLKc frequency = MCLK/2                |
| Writing "0101" | SNCLKs frequency = MCLK/2 <sup>4</sup><br>SNCLKc frequency = MCLK/2                |
| Writing "0110" | SNCLKs frequency = MCLK/2 <sup>5</sup><br>SNCLKc frequency = MCLK/2 <sup>2</sup>   |
| Writing "0111" | SNCLKs frequency = MCLK/2 <sup>6</sup><br>SNCLKc frequency = MCLK/2 <sup>3</sup>   |
| Writing "1000" | SNCLKs frequency = MCLK/2 <sup>7</sup><br>SNCLKc frequency = MCLK/2 <sup>4</sup>   |
| Writing "1001" | SNCLKs frequency = MCLK/2 <sup>8</sup><br>SNCLKc frequency = MCLK/2 <sup>5</sup>   |
| Writing "1010" | SNCLKs frequency = MCLK/2 <sup>9</sup><br>SNCLKc frequency = MCLK/2 <sup>6</sup>   |
| Writing "1011" | SNCLKs frequency = MCLK/2 <sup>10</sup><br>SNCLKc frequency = MCLK/2 <sup>7</sup>  |
| Writing "1100" | SNCLKs frequency = MCLK/2 <sup>11</sup><br>SNCLKc frequency = MCLK/2 <sup>8</sup>  |
| Writing "1101" | SNCLKs frequency = MCLK/2 <sup>12</sup><br>SNCLKc frequency = MCLK/2 <sup>9</sup>  |
| Writing "1110" | SNCLKs frequency = MCLK/2 <sup>13</sup><br>SNCLKc frequency = MCLK/2 <sup>10</sup> |
| Writing "1111" | SNCLKs frequency = MCLK/2 <sup>14</sup><br>SNCLKc frequency = MCLK/2 <sup>11</sup> |

For details of the SNCLKs frequency range, refer to the device data sheet.

## MB95850K/860K/870K Series

### 26.6.4 TS Warm Reset Register (WRESET)

The TS warm reset register (WRESET) controls warm reset generation of the TS.

#### ■ Register Configuration

|               |       |       |       |       |       |       |       |       |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
| Field         | WRER7 | WRER6 | WRER5 | WRER5 | WRER3 | WRER2 | WRER1 | WRER0 |
| Attribute     | W     | W     | W     | W     | W     | W     | W     | W     |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

#### ■ Register Functions

[bit7:0] WRER[7:0]: Warm reset bits

These bits controls warm reset generation of the TS.

In the case of modifying the value of a register and then immediately applying the modified value, use the warm reset.

| bit7:0                                       | Details                                  |
|--|--|
| <b>Writing "01011010"</b>                    | Generates a warm reset signal to the TS. |
| <b>Writing a value other than "01011010"</b> | Has no effect on operation.              |

## 26.6.5 TS Sensitivity Select Register 0 (RSEL0)

The TS sensitivity select register 0 (RSEL0) selects a sensitivity value for the touch reference channel.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3 | 2      | 1      | 0      |
|---------------|---|---|---|---|---|--------|--------|--------|
| Field         | — | — | — | — | — | SREFB2 | SREFB1 | SREFB0 |
| Attribute     | — | — | — | — | — | R/W    | R/W    | R/W    |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0      | 1      | 0      |

### ■ Register Functions

[bit7:3] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit2:0] SREFB[2:0]: Sensitivity of touch reference channel bits

These bits select a sensitivity value for the touch reference channel.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

## MB95850K/860K/870K Series

### 26.6.6 TS Sensitivity Select Register 1 (RSEL1)

The TS sensitivity select register 1 (RSEL1) selects the respective sensitivity values for touch ch. 0 and touch ch. 1.

#### Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S01B2 | S01B1 | S01B0 | S00B2 | S00B1 | S00B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

#### Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S01B[2:0]: Sensitivity of touch ch. 1 bits

These bits select a sensitivity value for touch ch. 1.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S00B[2:0]: Sensitivity of touch ch. 0 bits

These bits select a sensitivity value for touch ch. 0.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## 26.6.7 TS Sensitivity Select Register 2 (RSEL2)

The TS sensitivity select register 2 (RSEL2) selects the respective sensitivity values for touch ch. 2 and touch ch. 3.

### ■ Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S03B2 | S03B1 | S03B0 | S02B2 | S02B1 | S02B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

### ■ Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S03B[2:0]: Sensitivity of touch ch. 3 bits

These bits select a sensitivity value for touch ch. 3.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S02B[2:0]: Sensitivity of touch ch. 2 bits

These bits select a resistance value determining the sensitivity of touch ch. 2.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.



## MB95850K/860K/870K Series

### 26.6.8 TS Sensitivity Select Register 3 (RSEL3)

The TS sensitivity select register 3 (RSEL3) selects the respective sensitivity values for touch ch. 4 and touch ch. 5.

#### Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S05B2 | S05B1 | S05B0 | S04B2 | S04B1 | S04B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

#### Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S05B[2:0]: Sensitivity of touch ch. 5 bits

These bits select a sensitivity value for touch ch. 5.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S04B[2:0]: Sensitivity of touch ch. 4 bits

These bits select a sensitivity value for touch ch. 4.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## 26.6.9 TS Sensitivity Select Register 4 (RSEL4)

The TS sensitivity select register 4 (RSEL4) selects the respective sensitivity values for touch ch. 6 and touch ch. 7.

### Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S07B2 | S07B1 | S07B0 | S06B2 | S06B1 | S06B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

### Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S07B[2:0]: Sensitivity of touch ch. 7 bits

These bits select a sensitivity value for touch ch. 7.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S06B[2:0]: Sensitivity of touch ch. 6 bits

These bits select a sensitivity value for touch ch. 6.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## MB95850K/860K/870K Series

### 26.6.10 TS Sensitivity Select Register 5 (RSEL5)

The TS sensitivity select register 5 (RSEL5) selects the respective sensitivity values for touch ch. 8 and touch ch. 9.

#### Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S09B2 | S09B1 | S09B0 | S08B2 | S08B1 | S08B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

#### Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S09B[2:0]: Sensitivity of touch ch. 9 bits

These bits select a sensitivity value for touch ch. 9.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S08B[2:0]: Sensitivity of touch ch. 8 bits

These bits select a sensitivity value for touch ch. 8.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## 26.6.11 TS Sensitivity Select Register 6 (RSEL6)

The TS sensitivity select register 6 (RSEL6) selects the respective sensitivity values for touch ch. 10 and touch ch. 11.

### ■ Register Configuration

| bit           | 7 | 6 | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|---|-------|-------|-------|-------|-------|-------|
| Field         | — | — | S11B2 | S11B1 | S11B0 | S10B2 | S10B1 | S10B0 |
| Attribute     | — | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0 | 0     | 1     | 0     | 0     | 1     | 0     |

### ■ Register Functions

[bit7:6] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

[bit5:3] S11B[2:0]: Sensitivity of touch ch. 11 bits

These bits select a sensitivity value for touch ch. 11.

| bit5:3        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

[bit2:0] S10B[2:0]: Sensitivity of touch ch. 10 bits

These bits select a sensitivity value for touch ch. 10.

| bit2:0        | Details            |                  |
|---------------|--------------------|------------------|
|               | Dynamic range (pF) | Resolution* (pF) |
| Writing "000" | 19.10              | 0.149            |
| Writing "001" | 10.96              | 0.086            |
| Writing "010" | 8.05               | 0.063            |
| Writing "011" | 6.03               | 0.047            |
| Writing "100" | 5.12               | 0.040            |
| Writing "101" | 4.27               | 0.033            |
| Writing "110" | 3.84               | 0.030            |
| Writing "111" | 3.42               | 0.027            |

\*: Resolution = Dynamic range / 128

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## MB95850K/860K/870K Series

### 26.6.12 TS Beep Duration Setting Register (BPDUR)

The TS beep duration setting register (BPDUR) sets the duration of beep output triggered by a touch on a touch pad.

#### ■ Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | BDR7 | BDR6 | BDR5 | BDR4 | BDR3 | BDR2 | BDR1 | BDR0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

#### ■ Register Functions

[bit7:0] BDR[7:0]: Beep output duration bits

These bits set the duration of beep output.

Below is the equation of calculating the duration of beep output.

Beep output duration = value of BPDUR register × tSNCLKs × 32

tSNCLKs: TS clock period in the sensing phase

Example:

BPDUR = 0b00000001

tSNCLKs = 0.1 ms

Beep output duration =  $1 \times 0.1 \times 32 = 3.2$  ms

Note:

For details of the beep output function, see "CHAPTER 27 BEEP OUTPUT UNIT".

## 26.6.13 TS Direct Output Control Register 1 (DIOR1)

The TS direct output control register 1 (DIOR1) controls the touch channel status output of DIO ch. 0 and DIO ch. 1.

### ■ Register Configuration

| bit           | 7       | 6       | 5       | 4       | 3       | 2       | 1       | 0       |
|---------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Field         | DIO01S3 | DIO01S2 | DIO01S1 | DIO01S0 | DIO00S3 | DIO00S2 | DIO00S1 | DIO00S0 |
| Attribute     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     |
| Initial value | 0       | 0       | 0       | 0       | 0       | 0       | 0       | 0       |

### ■ Register Functions

[bit7:4] DIO01S[3:0]: DIO ch. 1 control bits

These bits control the touch channel status output of DIO ch. 1.

When the touch pad of the touch channel whose status is to be output by DIO ch. 1 is touched, DIO ch. 1 outputs "L", otherwise DIO ch. 1 outputs "H".

| bit7:4         | Details  |
|----------------|--|
| Writing "0000" | Disables DIO ch. 1.                                |
| Writing "0001" | Makes DIO ch. 1 output the status of touch ch. 0.  |
| Writing "0010" | Makes DIO ch. 1 output the status of touch ch. 1.  |
| Writing "0011" | Makes DIO ch. 1 output the status of touch ch. 2.  |
| Writing "0100" | Makes DIO ch. 1 output the status of touch ch. 3.  |
| Writing "0101" | Makes DIO ch. 1 output the status of touch ch. 4.  |
| Writing "0110" | Makes DIO ch. 1 output the status of touch ch. 5.  |
| Writing "0111" | Makes DIO ch. 1 output the status of touch ch. 6.  |
| Writing "1000" | Makes DIO ch. 1 output the status of touch ch. 7.  |
| Writing "1001" | Makes DIO ch. 1 output the status of touch ch. 8.  |
| Writing "1010" | Makes DIO ch. 1 output the status of touch ch. 9.  |
| Writing "1011" | Makes DIO ch. 1 output the status of touch ch. 10. |
| Writing "1100" | Makes DIO ch. 1 output the status of touch ch. 11. |
| Writing "1101" |  |
| Writing "1110" |  |
| Writing "1111" |  |

## MB95850K/860K/870K Series

[bit3:0] DIO00S[3:0]: DIO ch. 0 control bits

These bits control the touch channel status output of DIO ch. 0.

When the touch pad of the touch channel whose status is to be output by DIO ch. 0 is touched, DIO ch. 0 outputs "L", otherwise DIO ch. 0 outputs "H".

| bit3:0                | Details  |
|-----------------------|--|
| <b>Writing "0000"</b> | Disables DIO ch. 0.                                |
| <b>Writing "0001"</b> | Makes DIO ch. 0 output the status of touch ch. 0.  |
| <b>Writing "0010"</b> | Makes DIO ch. 0 output the status of touch ch. 1.  |
| <b>Writing "0011"</b> | Makes DIO ch. 0 output the status of touch ch. 2.  |
| <b>Writing "0100"</b> | Makes DIO ch. 0 output the status of touch ch. 3.  |
| <b>Writing "0101"</b> | Makes DIO ch. 0 output the status of touch ch. 4.  |
| <b>Writing "0110"</b> | Makes DIO ch. 0 output the status of touch ch. 5.  |
| <b>Writing "0111"</b> | Makes DIO ch. 0 output the status of touch ch. 6.  |
| <b>Writing "1000"</b> | Makes DIO ch. 0 output the status of touch ch. 7.  |
| <b>Writing "1001"</b> | Makes DIO ch. 0 output the status of touch ch. 8.  |
| <b>Writing "1010"</b> | Makes DIO ch. 0 output the status of touch ch. 9.  |
| <b>Writing "1011"</b> | Makes DIO ch. 0 output the status of touch ch. 10. |
| <b>Writing "1100"</b> | Makes DIO ch. 0 output the status of touch ch. 11. |
| <b>Writing "1101"</b> |  |
| <b>Writing "1110"</b> |  |
| <b>Writing "1111"</b> |  |

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## 26.6.14 TS Direct Output Control Register 2 (DIOR2)

The TS direct output control register 2 (DIOR2) controls the touch channel status output of DIO ch. 2 and DIO ch. 3.

### ■ Register Configuration

| bit           | 7       | 6       | 5       | 4       | 3       | 2       | 1       | 0       |
|---------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Field         | DIO03S3 | DIO03S2 | DIO03S1 | DIO03S0 | DIO02S3 | DIO02S2 | DIO02S1 | DIO02S0 |
| Attribute     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     | R/W     |
| Initial value | 0       | 0       | 0       | 0       | 0       | 0       | 0       | 0       |

### ■ Register Functions

[bit7:4] DIO03S[3:0]: DIO ch. 3 control bits

These bits control the touch channel status output of DIO ch. 3.

When the touch pad of the touch channel whose status is to be output by DIO ch. 3 is touched, DIO ch. 3 outputs "L", otherwise DIO ch. 3 outputs "H".

| bit7:4         | Details  |
|----------------|--|
| Writing "0000" | Disables DIO ch. 3.                                |
| Writing "0001" | Makes DIO ch. 3 output the status of touch ch. 0.  |
| Writing "0010" | Makes DIO ch. 3 output the status of touch ch. 1.  |
| Writing "0011" | Makes DIO ch. 3 output the status of touch ch. 2.  |
| Writing "0100" | Makes DIO ch. 3 output the status of touch ch. 3.  |
| Writing "0101" | Makes DIO ch. 3 output the status of touch ch. 4.  |
| Writing "0110" | Makes DIO ch. 3 output the status of touch ch. 5.  |
| Writing "0111" | Makes DIO ch. 3 output the status of touch ch. 6.  |
| Writing "1000" | Makes DIO ch. 3 output the status of touch ch. 7.  |
| Writing "1001" | Makes DIO ch. 3 output the status of touch ch. 8.  |
| Writing "1010" | Makes DIO ch. 3 output the status of touch ch. 9.  |
| Writing "1011" | Makes DIO ch. 3 output the status of touch ch. 10. |
| Writing "1100" | Makes DIO ch. 3 output the status of touch ch. 11. |
| Writing "1101" |  |
| Writing "1110" |  |
| Writing "1111" |  |



## MB95850K/860K/870K Series

[bit3:0] DIO02S[3:0]: DIO ch. 2 control bits

These bits control the touch channel status output of DIO ch. 2.

When the touch pad of the touch channel whose status is to be output by DIO ch. 2 is touched, DIO ch. 2 outputs "L", otherwise DIO ch. 2 outputs "H".

| bit3:0         | Details  |
|----------------|--|
| Writing "0000" | Disables DIO ch. 2.                                |
| Writing "0001" | Makes DIO ch. 2 output the status of touch ch. 0.  |
| Writing "0010" | Makes DIO ch. 2 output the status of touch ch. 1.  |
| Writing "0011" | Makes DIO ch. 2 output the status of touch ch. 2.  |
| Writing "0100" | Makes DIO ch. 2 output the status of touch ch. 3.  |
| Writing "0101" | Makes DIO ch. 2 output the status of touch ch. 4.  |
| Writing "0110" | Makes DIO ch. 2 output the status of touch ch. 5.  |
| Writing "0111" | Makes DIO ch. 2 output the status of touch ch. 6.  |
| Writing "1000" | Makes DIO ch. 2 output the status of touch ch. 7.  |
| Writing "1001" | Makes DIO ch. 2 output the status of touch ch. 8.  |
| Writing "1010" | Makes DIO ch. 2 output the status of touch ch. 9.  |
| Writing "1011" | Makes DIO ch. 2 output the status of touch ch. 10. |
| Writing "1100" | Makes DIO ch. 2 output the status of touch ch. 11. |
| Writing "1101" |  |
| Writing "1110" |  |
| Writing "1111" |  |

Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## 26.6.15 TS Direct Output Control Register 3 (DIOR3)

The TS direct output control register 3 (DIOR3) controls the touch channel status output of DIO ch. 4.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3       | 2       | 1       | 0       |
|---------------|---|---|---|---|---------|---------|---------|---------|
| Field         | — | — | — | — | DIO04S3 | DIO04S2 | DIO04S1 | DIO04S0 |
| Attribute     | — | — | — | — | R/W     | R/W     | R/W     | R/W     |
| Initial value | 0 | 0 | 0 | 0 | 0       | 0       | 0       | 0       |

### ■ Register Functions

#### [bit7:4] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit3:0] DIO04S[3:0]: DIO ch. 4 control bits

These bits control the touch channel status output of DIO ch. 4.

When the touch pad of the touch channel whose status is to be output by DIO ch. 4 is touched, DIO ch. 4 outputs "L", otherwise DIO ch. 4 outputs "H".

| bit3:0         | Details  |
|----------------|--|
| Writing "0000" | Disables DIO ch. 4.                                |
| Writing "0001" | Makes DIO ch. 4 output the status of touch ch. 0.  |
| Writing "0010" | Makes DIO ch. 4 output the status of touch ch. 1.  |
| Writing "0011" | Makes DIO ch. 4 output the status of touch ch. 2.  |
| Writing "0100" | Makes DIO ch. 4 output the status of touch ch. 3.  |
| Writing "0101" | Makes DIO ch. 4 output the status of touch ch. 4.  |
| Writing "0110" | Makes DIO ch. 4 output the status of touch ch. 5.  |
| Writing "0111" | Makes DIO ch. 4 output the status of touch ch. 6.  |
| Writing "1000" | Makes DIO ch. 4 output the status of touch ch. 7.  |
| Writing "1001" | Makes DIO ch. 4 output the status of touch ch. 8.  |
| Writing "1010" | Makes DIO ch. 4 output the status of touch ch. 9.  |
| Writing "1011" | Makes DIO ch. 4 output the status of touch ch. 10. |
| Writing "1100" | Makes DIO ch. 4 output the status of touch ch. 11. |
| Writing "1101" |  |
| Writing "1110" |  |
| Writing "1111" |  |

#### Note:

The number of touch channels varies among products. For details, refer to the device data sheet.

## MB95850K/860K/870K Series

### 26.6.16 TS Feature Select Register (FTSEL)

The TS feature select register (FTSEL) enables or disables the filter function, and selects an APIS (Adjacent Pattern Interference Suppression) mode.

#### Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3 | 2      | 1      | 0    |
|---------------|---|---|---|---|---|--------|--------|------|
| Field         | — | — | — | — | — | APSEL1 | APSEL0 | ENAF |
| Attribute     | — | — | — | — | — | R/W    | R/W    | R/W  |
| Initial value | 0 | 0 | 0 | 0 | 0 | 1      | 0      | 0    |

#### Register Functions

##### [bit7:3] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit2:1] APSEL[1:0]: APIS mode select bits

These bits select a mode for APIS, which eliminates adjacent key interference or adjacent pattern interference.

| bit2:1              | Details   |
|---------------------|---|
| <b>Writing "00"</b> | Bypasses APIS.  |
| <b>Writing "01"</b> | Selects APIS mode 1.<br>In APIS mode 1, only the strongest one of all sensor outputs that exceed the threshold defined in the TS touch strength threshold setting register ch. n (STRTHn), is saved in the corresponding TS touch strength register ch. n (STRn) within the integration time set in the TS integration time setting register (ITGTM). |
| <b>Writing "10"</b> | Selects APIS mode 2.<br>In APIS mode 2, all sensor outputs that exceed the threshold defined in the STRTHn register are saved in the corresponding STRn register within the integration time set in the ITGTM register.   |
| <b>Writing "11"</b> | Selects APIS mode 3.<br>In APIS mode 3, the two strongest ones of all sensor outputs that exceed the threshold defined in the STRTHn register are saved in the corresponding STRn register within the integration time set in the ITGTM register.   |

##### [bit0] ENAF: Filter function enable bit

This bit enables or disables the filter function. For details of the filter function, see "26.6.26 TS Filter Threshold Setting Register (FLTTH)".

| bit0               | Details                       |
|--------------------|-------------------------------|
| <b>Writing "0"</b> | Disables the filter function. |
| <b>Writing "1"</b> | Enables the filter function.  |

## 26.6.17 TS AIC Wait Time Setting Register (AICWAT)

The TS AIC wait time setting register (AICWAT) sets the AIC (Automatic Impedance Calibration) wait time.

### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | AICW7 | AICW6 | AICW5 | AICW4 | AICW3 | AICW2 | AICW1 | AICW0 |
| Attribute     | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 1     | 0     | 0     | 1     | 1     | 1     |

### ■ Register Functions

[bit7:0] AICW[7:0]: AIC wait time set bits

These bits set the AIC wait time.

Below is the equation of calculating the AIC wait time.

AIC wait time = value of AICWAT register × tSNCLKs × 64

tSNCLKs: TS clock period in the sensing phase

Example:

AICWAT = 0b00000001

tSNCLKs = 0.1 ms

AIC wait time = 1 × 0.1 × 64 = 6.4 ms

Using the AICWAT register and the TS beta value setting register (BETA) together can stabilize the AIC operation by preventing false calibration from occurring when a finger is being removed from a touch pad very slowly.

AIC is disabled while any touch pad is being touched. When no touch pad has been touched and the AIC wait time elapses, AIC starts.

During the AIC wait time, when a touch pad is touched, the AIC controller reloads the AIC wait time and the AIC wait time restarts.

The bigger the value of the AICWAT register, the longer the wait time before the start of AIC becomes.

## MB95850K/860K/870K Series

### 26.6.18 TS Calibration Interval Setting Register (CALITV)

The TS calibration interval setting register (CALITV) sets the calibration interval.

#### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | CALI7 | CALI6 | CALI5 | CALI4 | CALI3 | CALI2 | CALI1 | CALI0 |
| Attribute     | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 1     | 1     | 0     | 0     | 0     | 0     |

#### ■ Register Functions

[bit7:0] CALI[7:0]: Calibration interval set bits

These bits set the calibration interval.

Below is the equation of calculating the calibration interval.

Calibration interval = value of CALITV register  $\times$  tSNCLKs  $\times$  50 – 139  $\times$  tSNCLKs + 139  $\times$  tSNCLKc

tSNCLKc: TS clock period in the calibration phase

tSNCLKs: TS clock period in the sensing phase

For details of SNCLKs and SNCLKc, see "26.6.3 TS Prescaler Control Register (PSC)".

Note:

Set the calibration interval as shown below.

Calibration interval > 139  $\times$  tSNCLKc + integration time in the sensing phase

For example, when the integration time is one tSNCLKs, set the value of the CALITV register to a value larger than or equal to "3".

## 26.6.19 TS Integration Time Setting Register (ITGTM)

The TS integration time setting register (ITGTM) sets the integration time for APIS.

### Register Configuration

| bit           | 7      | 6      | 5      | 4      | 3      | 2      | 1      | 0      |
|---------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Field         | INTET7 | INTET6 | INTET5 | INTET4 | INTET3 | INTET2 | INTET1 | INTET0 |
| Attribute     | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    |
| Initial value | 0      | 0      | 0      | 0      | 1      | 1      | 1      | 1      |

### Register Functions

[bit7:0] INTET[7:0]: Integration time set bits

These bits set the integration time.

Below is the equation of calculating the integration time.

Integration time = value of ITGTM register × tSNCLKs

tSNCLKs: TS clock period in the sensing phase

Example:

ITGTM = 0b00000001

tSNCLKs = 0.1 ms

Integration time = 1 × 0.1 = 0.1 ms

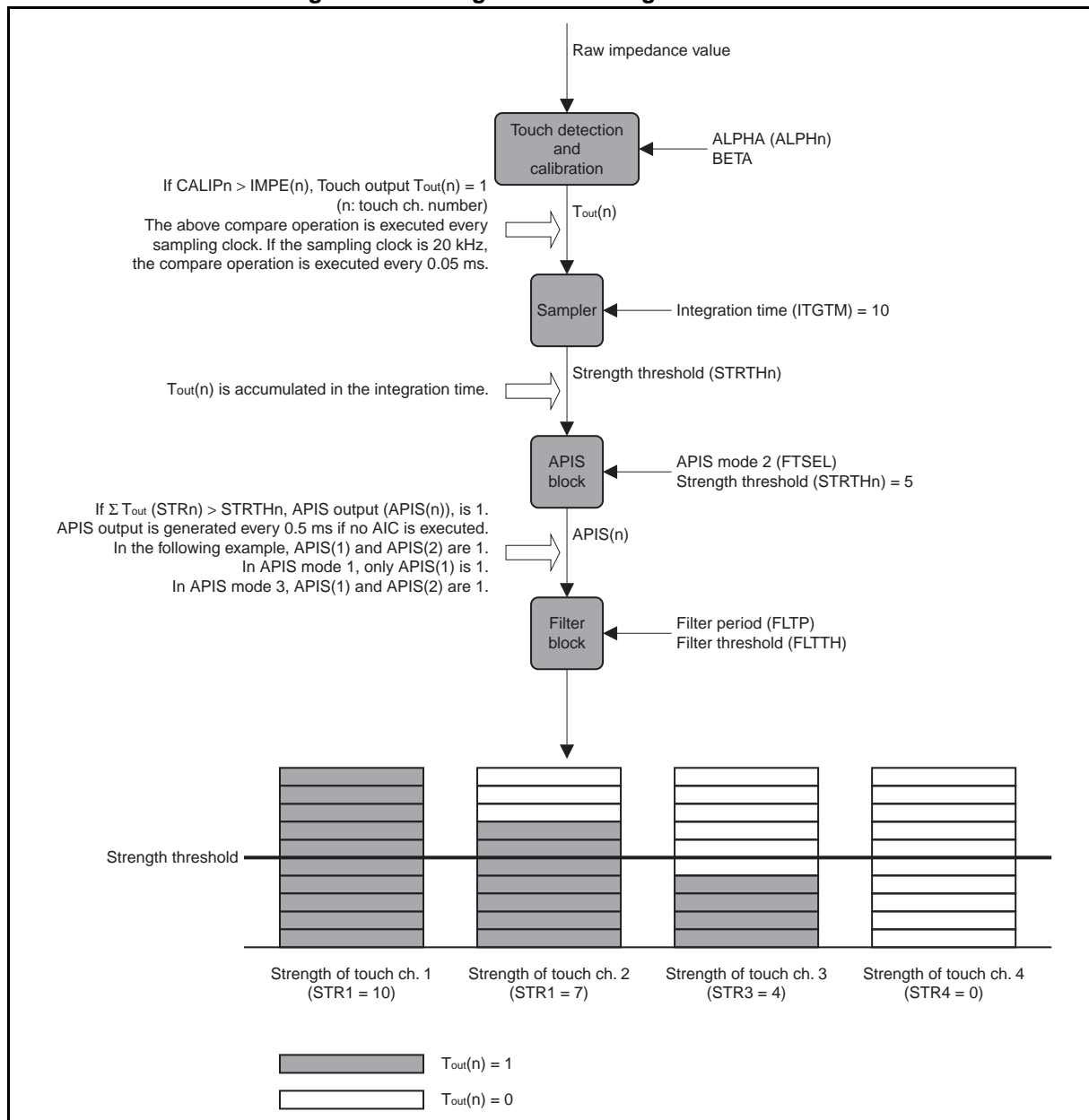
The integration time is the period of accumulating touch output as shown in Figure 26.6-2.

APIS determines its output at each integration time interval.

Set the value of the ITGTM register larger than or equal to that of the STRTHn register.

The smaller the value of the ITGTM register, the shorter the response time of touch output becomes. On the other hand, the APIS filtering capacity increases as the value of the ITGTM register increases.

Figure 26.6-2 Digital Processing Module of TS



## 26.6.20 TS Idle Time Setting Register (IDLETM)

The TS idle time setting register (IDLETM) sets how much idle time has to elapse before the TS enters idle mode after all touch pads have entered no-touch state.

### ■ Register Configuration

| bit           | 7      | 6      | 5      | 4      | 3      | 2      | 1      | 0      |
|---------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Field         | IDLET7 | IDLET6 | IDLET5 | IDLET4 | IDLET3 | IDLET2 | IDLET1 | IDLET0 |
| Attribute     | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    | R/W    |
| Initial value | 0      | 0      | 0      | 0      | 1      | 1      | 1      | 1      |

### ■ Register Functions

[bit7:0] IDLET[7:0]: Idle time set bits

These bits set the idle time.

Below is the equation of calculating the idle time.

Idle time = value of IDLETM register × tSNCLKs × 5000

tSNCLKs: TS clock period in the sensing phase

When no touch pad has been touched for the length of the idle time, a timeout condition is met and the TS enters idle mode.



## MB95850K/860K/870K Series

### 26.6.21 TS Control Register (CONTROL)

The TS control register (CONTROL) controls AIC and the operating mode of the TS.

#### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3   | 2     | 1     | 0    |
|---------------|---|---|---|---|-----|-------|-------|------|
| Field         | — | — | — | — | F2A | HDC_U | HDC_C | HOLD |
| Attribute     | — | — | — | — | R/W | R/W   | R/W   | R/W  |
| Initial value | 0 | 0 | 0 | 0 | 0   | 0     | 0     | 0    |

#### ■ Register Functions

##### [bit7:4] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit3] F2A: Force to active bit

Writing this bit to "1" makes the TS transit from idle mode to active mode and stay in active mode even a timeout condition is met.

| bit3               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Has no effect on operation.   |
| <b>Writing "1"</b> | Makes the TS transit from idle mode to active mode and stay in active mode even a timeout condition is met. |

##### [bit2] HDC\_U: Unconditional host driven bit

Writing this bit to "1" makes the TS execute calibration unconditionally. This bit is automatically cleared to "0" upon completion of the calibration cycle.

Disable AIC (CONTROL:HOLD = 1) before making the TS execute calibration unconditionally.

When calibration is required without any condition, write "1" to this bit.

| bit2               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Has no effect on operation.                       |
| <b>Writing "1"</b> | Makes the TS execute calibration unconditionally. |

##### [bit1] HDC\_C: Conditional host driven bit

Writing this bit to "1" makes the TS execute calibration only when no touch pad is touched to avoid false calibration. This bit is automatically cleared to "0" upon completion of the calibration cycle.

Disable AIC (CONTROL:HOLD = 1) before making the TS execute calibration only when no touch pad is touched.

| bit1               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Has no effect on operation.   |
| <b>Writing "1"</b> | Makes the TS execute calibration only when no touch pad is touched. |

##### [bit0] HOLD: AIC hold bit

This bit enables or disables (holds) AIC.

| bit0               | Details       |
|--------------------|---------------|
| <b>Writing "0"</b> | Enables AIC.  |
| <b>Writing "1"</b> | Disables AIC. |

## 26.6.22 TS Interrupt Mask Register (INTMR)

The TS interrupt mask register (INTMR) controls masking interrupts of the TS to be output to the CPU when GINT is used.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4   | 3   | 2   | 1   | 0    |
|---------------|---|---|---|-----|-----|-----|-----|------|
| Field         | — | — | — | NEW | EOC | ITA | ATI | TINT |
| Attribute     | — | — | — | R/W | R/W | R/W | R/W | R/W  |
| Initial value | 0 | 0 | 0 | 1   | 1   | 0   | 0   | 0    |

### ■ Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4] NEW: New impedance interrupt mask bit

This bit masks or unmaskes the new impedance interrupt.

| bit4               | Details                              |
|--------------------|--------------------------------------|
| <b>Writing "0"</b> | Unmasks the new impedance interrupt. |
| <b>Writing "1"</b> | Masks the new impedance interrupt.   |

#### [bit3] EOC: EOC interrupt mask bit

This bit masks or unmaskes the EOC interrupt.

| bit3               | Details                    |
|--------------------|----------------------------|
| <b>Writing "0"</b> | Unmasks the EOC interrupt. |
| <b>Writing "1"</b> | Masks the EOC interrupt.   |

#### [bit2] ITA: IDLE-to-ACTIVE interrupt mask bit

This bit masks or unmaskes the IDLE-to-ACTIVE interrupt.

| bit2               | Details                               |
|--------------------|---------------------------------------|
| <b>Writing "0"</b> | Unmasks the IDLE-to-ACTIVE interrupt. |
| <b>Writing "1"</b> | Masks the IDLE-to-ACTIVE interrupt.   |

#### [bit1] ATI: ACTIVE-to-IDLE interrupt mask bit

This bit masks or unmaskes the ACTIVE-to-IDLE interrupt.

| bit1               | Details                               |
|--------------------|---------------------------------------|
| <b>Writing "0"</b> | Unmasks the ACTIVE-to-IDLE interrupt. |
| <b>Writing "1"</b> | Masks the ACTIVE-to-IDLE interrupt.   |

#### [bit0] TINT: TINT mask bit

This bit masks or unmaskes the TINT.

| bit0               | Details           |
|--------------------|-------------------|
| <b>Writing "0"</b> | Unmasks the TINT. |
| <b>Writing "1"</b> | Masks the TINT.   |

## MB95850K/860K/870K Series

### 26.6.23 TS Interrupt Clear Register (INTCR)

The TS interrupt clear register (INTCR) clears interrupts of the TS.

#### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4   | 3   | 2   | 1   | 0    |
|---------------|---|---|---|-----|-----|-----|-----|------|
| Field         | — | — | — | NEW | EOC | ITA | ATI | TINT |
| Attribute     | — | — | — | R/W | R/W | R/W | R/W | R/W  |
| Initial value | 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0    |

#### ■ Register Functions

##### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit4] NEW: New impedance interrupt clear bit

This bit clears the new impedance interrupt. The bit is automatically cleared to "0" after the new impedance interrupt flag bit in the TS interrupt pending register (INTPR:NEW) is cleared to "0".

| bit4               | Details                             |
|--------------------|-------------------------------------|
| <b>Writing "0"</b> | Has no effect on operation.         |
| <b>Writing "1"</b> | Clears the new impedance interrupt. |

##### [bit3] EOC: EOC interrupt clear bit

This bit clears the EOC interrupt. The bit is automatically cleared to "0" after the EOC interrupt flag bit in the TS interrupt pending register (INTPR:EOC) is cleared to "0".

| bit3               | Details                     |
|--------------------|-----------------------------|
| <b>Writing "0"</b> | Has no effect on operation. |
| <b>Writing "1"</b> | Clears the EOC interrupt.   |

##### [bit2] ITA: IDLE-to-ACTIVE interrupt clear bit

This bit clears the IDLE-to-ACTIVE interrupt. The bit is automatically cleared to "0" after the IDLE-to-ACTIVE interrupt flag bit in the TS interrupt pending register (INTPR:ITA) is cleared to "0".

| bit2               | Details                              |
|--------------------|--------------------------------------|
| <b>Writing "0"</b> | Has no effect on operation.          |
| <b>Writing "1"</b> | Clears the IDLE-to-ACTIVE interrupt. |

##### [bit1] ATI: ACTIVE-to-IDLE interrupt clear bit

This bit clears the ACTIVE-to-IDLE interrupt. The bit is automatically cleared to "0" after the ACTIVE-to-IDLE interrupt flag bit in the TS interrupt pending register (INTPR:ATI) is cleared to "0".

| bit1               | Details                              |
|--------------------|--------------------------------------|
| <b>Writing "0"</b> | Has no effect on operation.          |
| <b>Writing "1"</b> | Clears the ACTIVE-to-IDLE interrupt. |

[bit0] TINT: TINT clear bit

This bit clears the TINT. The bit is automatically cleared to "0" after the TINT flag bit in the TS interrupt pending register (INTPR:TINT) is cleared to "0".

| bit0               | Details                     |
|--------------------|-----------------------------|
| <b>Writing "0"</b> | Has no effect on operation. |
| <b>Writing "1"</b> | Clears the TINT.            |

## MB95850K/860K/870K Series

### 26.6.24 TS Interrupt Pending Register (INTPR)

The TS interrupt pending register (INTPR) indicates the interrupt sources of the TS.

#### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4   | 3   | 2   | 1   | 0    |
|---------------|---|---|---|-----|-----|-----|-----|------|
| Field         | — | — | — | NEW | EOC | ITA | ATI | TINT |
| Attribute     | — | — | — | R   | R   | R   | R   | R    |
| Initial value | 0 | 0 | 0 | X   | X   | X   | X   | X    |

#### ■ Register Functions

##### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

##### [bit4] NEW: New impedance interrupt flag bit

This bit indicates whether the value of the IMPEN register has changed.

This bit is read-only. Writing a value to this bit has no effect on operation. This bit is cleared by writing "1" to the NEW bit in the INTCR register.

| bit4        | Details   |
|-------------|---|
| Reading "0" | Indicates that there is no change in the value of the IMPEN register. |
| Reading "1" | Indicates that the value of the IMPEN register has changed.           |

##### [bit3] EOC: EOC interrupt flag bit

This bit indicates whether calibration has ended.

This bit is read-only. Writing a value to this bit has no effect on operation. This bit is cleared by writing "1" to the EOC bit in the INTCR register.

| bit3        | Details                                    |
|-------------|--|
| Reading "0" | Indicates that calibration is in progress. |
| Reading "1" | Indicates that calibration has ended.      |

##### [bit2] ITA: IDLE-to-ACTIVE interrupt flag bit

This bit indicates whether the TS has transited from idle mode to active mode.

This bit is read-only. Writing a value to this bit has no effect on operation. This bit is cleared by writing "1" to the ITA bit in the INTCR register.

| bit2        | Details  |
|-------------|--|
| Reading "0" | Indicates that the TS is still in idle mode.                       |
| Reading "1" | Indicates that the TS has transited from idle mode to active mode. |

##### [bit1] ATI: ACTIVE-to-IDLE interrupt flag bit

This bit indicates the TS has transited from active mode to idle mode.

This bit is read-only. Writing a value to this bit has no effect on operation. This bit is cleared by writing "1" to the ATI bit in the INTCR register.

| bit1        | Details  |
|-------------|--|
| Reading "0" | Indicates that the TS is still in active mode.                     |
| Reading "1" | Indicates that the TS has transited from active mode to idle mode. |

[bit0] TINT: TINT flag bit

This bit indicates that no touch operation or touch release operation has been executed, or that a touch operation or a touch release operation has been executed.

This bit is read-only. Writing a value to this bit has no effect on operation. This bit is cleared by writing "1" to the TINT bit in the INTCR register.

| bit0               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that no touch operation or touch release operation has been executed.  |
| <b>Reading "1"</b> | Indicates that a touch operation or a touch release operation has been executed. |

## MB95850K/860K/870K Series

### 26.6.25 TS Filter Period Setting Register (FLTP)

The TS filter period setting register (FLTP) determines the length of the additional APIS filtering period.

#### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | FILP7 | FILP6 | FILP5 | FILP4 | FILP3 | FILP2 | FILP1 | FILP0 |
| Attribute     | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

#### ■ Register Functions

[bit7:0] FILP[7:0]: Filter period set bits

These bits determine the length of the additional APIS filtering period.

During a period of the length of "value of FLTP register × integration time (see "26.6.19 TS Integration Time Setting Register (ITGTM))", a filter accumulates APIS output. (The algorithm for this operation is similar to APIS.) When the cumulative APIS output exceeds the value of the TS filter threshold setting register (FLTTH), the filter outputs touch output.

The filter can stabilize APIS output and reduce TINT generation.

## 26.6.26 TS Filter Threshold Setting Register (FLTTH)

The TS filter threshold setting register (FLTTH) sets the threshold for filtered touch output.

### ■ Register Configuration

| bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Field         | FILT7 | FILT6 | FILT5 | FILT4 | FILT3 | FILT2 | FILT1 | FILT0 |
| Attribute     | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

### ■ Register Functions

[bit7:0] FILT[7:0]: Filter threshold set bits

These bits set the threshold for filtered touch output.

APIS output is accumulated in the filter during the period of the length of "value of FLTP register × integration time (see "26.6.19 TS Integration Time Setting Register (ITGTM))" and is compared with the threshold set in this register. A touch on the touch pad is deemed valid only when the cumulative APIS output is larger than the threshold set in this register.



## MB95850K/860K/870K Series

### 26.6.27 TS Reference Delay Setting Register (REFDLY)

The TS reference delay setting register (REFDLY) sets the reference impedance.

#### ■ Register Configuration

| bit           | 7 | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|-------|-------|-------|-------|-------|-------|-------|
| Field         | — | REFD6 | REFD5 | REFD4 | REFD3 | REFD2 | REFD1 | REFD0 |
| Attribute     | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

#### ■ Register Functions

##### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

##### [bit6:0] REFD[6:0]: Reference delay set bits

These bits set the reference impedance in the range of zero to 127 inclusive.

The reference impedance is necessary for determining the touch sensor operating point, giving the touch sensor an appropriate capacitive input dynamic range.

The reference delay has the same effect as installing an external capacitor on the TS reference input pin (AREF). The reference delay chain has 128 delay steps. For details, see the table in details of the SREFB[2:0] bits in the RSEL0 register.

For example, when the SREFB[2:0] bits are set to "0b010", the maximum value of the REFD[6:0] bits is equivalent to the capacitance generated by putting an 8.05 pF capacitor at the AREF pin.

#### Note:

After modifying the value of the REFDLY register, always use a warm reset to restart the TS.

## 26.6.28 TS Alpha Value Setting Register ch. n (ALPHn)

The TS alpha value setting register ch. n (ALPHn) sets the sensitivity of touch ch. n.

### ■ Register Configuration

| bit           | 7 | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|---|------|------|------|------|------|------|------|
| Field         | — | ALP6 | ALP5 | ALP4 | ALP3 | ALP2 | ALP1 | ALP0 |
| Attribute     | — | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0 | 0    | 0    | 0    | 1    | 0    | 0    | 0    |

### ■ Register Functions

#### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

#### [bit6:0] ALP[6:0]: Alpha value set bits

These bits set the alpha value (sensitivity) of touch ch. n.

The smaller the alpha value is, the greater the sensitivity becomes.

For example, with the sensitivity of touch ch. n bits (SnB[2:0]) set to "0b010", each step of alpha value has a resolution of about 0.063 pF. For details, see the table in the SnB[2:0] bits in the RSEL1/RSEL2/RSEL3/RSEL4/RSEL5/RSEL6 register.

When the alpha value is set to "5", a touch has to induce a capacitance of 0.315 pF ( $5 \times 0.063$  pF) or larger to activate touch output. The capacitance induced by a touch depends on the thickness of a pad cover and the surface area of a metal electrode.

## MB95850K/860K/870K Series

### 26.6.29 TS Beta Value Setting Register (BETA)

The TS beta value setting register (BETA) sets the threshold for starting AIC.

#### ■ Register Configuration

| bit           | 7 | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|---|-------|-------|-------|-------|-------|-------|-------|
| Field         | — | BETA6 | BETA5 | BETA4 | BETA3 | BETA2 | BETA1 | BETA0 |
| Attribute     | — | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial value | 0 | 0     | 0     | 0     | 0     | 1     | 0     | 0     |

#### ■ Register Functions

##### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

##### [bit6:0] BETA[6:0]: Beta value set bits

These bits set the beta value for starting AIC.

AIC is enabled only when no touch pad is touched. The beta value is a criterion for checking that no touch pad of any touch channel is being touched.

The TS compares the current impedance and previous impedance of all sensor inputs before starting AIC. When the difference between the current impedance and the previous impedance is larger than the beta value, the TS cancels AIC. For details, see Figure 26.5-10.

The default beta value is "4". The minimum beta value is "0" and the maximum beta value "127". Increasing the beta value decreases the possibility of AIC being cancelled for a certain period of time.

## 26.6.30 TS Touch Strength Threshold Setting Register ch. n (STRTHn)

The TS touch strength threshold setting register ch. n (STRTHn) sets the touch strength threshold for touch ch. n.

### ■ Register Configuration

|               |      |      |      |      |      |      |      |      |
|---------------|------|------|------|------|------|------|------|------|
| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
| Field         | STH7 | STH6 | STH5 | STH4 | STH3 | STH2 | STH1 | STH0 |
| Attribute     | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  | R/W  |
| Initial value | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 1    |

### ■ Register Functions

[bit7:0] STH[7:0]: Strength threshold set bits

These bits set the touch strength threshold for touch ch. n.

The internal sampler of the TS measures the strength of each touch. APIS compares the touch strength threshold set in the STH[7:0] bits with the strength of a touch to determine whether that touch is the final touch.

Increasing the touch strength threshold reduces touch output sensitivity, but it increases the filtering capacity of APIS by eliminating more unintentional touches.

#### Note:

Set the value of the STRTHn register smaller than or equal to the value of the ITGTM register.

## MB95850K/860K/870K Series

### 26.6.31 TS Touch Strength Register ch. n (STRn)

The TS touch strength register ch. n (STRn) reads the touch strength of touch ch. n at every integration time in APIS mode 1/2/3.

#### ■ Register Configuration

| bit           | 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Field         | SR7 | SR6 | SR5 | SR4 | SR3 | SR2 | SR1 | SR0 |
| Attribute     | R   | R   | R   | R   | R   | R   | R   | R   |
| Initial value | X   | X   | X   | X   | X   | X   | X   | X   |

#### ■ Register Functions

[bit7:0] SR[7:0]: Strength read bits

These bits read the strength for touch ch. n and store the touch strength of touch ch. n at every integration time.

The maximum value of the STRn register is equal to the value of the ITGTM register. Therefore, to increase the maximum touch strength, increase the value of the ITGTM register.

## 26.6.32 TS Calibrated Impedance Register ch. n (CALIPn)

The TS calibrated impedance register ch. n (CALIPn) monitors the calibrated impedance resulting from AIC.

### ■ Register Configuration

| bit           | 7 | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|---|------|------|------|------|------|------|------|
| Field         | — | CIR6 | CIR5 | CIR4 | CIR3 | CIR2 | CIR1 | CIR0 |
| Attribute     | — | R    | R    | R    | R    | R    | R    | R    |
| Initial value | 0 | X    | X    | X    | X    | X    | X    | X    |

### ■ Register Functions

#### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

#### [bit6:0] CIR[6:0]: Calibrated impedance read bits

These bits read the calibrated impedance of touch ch. n. The calibrated impedance is the reference impedance for touch ch. n and is monitored by these bits.

AIC measures the impedance of touch ch. n when there is no touch on the touch pad, and subtracts the alpha value (ALPHn register) from the impedance (IMPEn register) to acquire the calibrated impedance of touch ch. n.

When there is no touch on the touch pad, the calibrated impedance of touch ch. n is the remainder of subtracting the value of the ALPHn register from the value of the IMPEn register.

When the touch pad is touched and the decrease\* in impedance (value of the IMPEn register) resulting from the touch is larger than the alpha value, the touch can be recognized by the TS.

\*: According to the circuit operation principle, the increase in impedance of an electrode is measured as a decrease in the value of the IMPEn register and vice versa. The actual impedance can be obtained by subtracting the value of the IMPEn register from 127. The actual calibrated impedance can be obtained by subtracting the value of the CALIPn register from 127.

## MB95850K/860K/870K Series

### 26.6.33 TS Impedance Register ch. n (IMPE<sub>n</sub>)

The TS impedance register ch. n (IMPE<sub>n</sub>) reflects the current impedance change on the touch pad of touch ch. n.

#### ■ Register Configuration

| bit           | 7 | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
|---------------|---|-----|-----|-----|-----|-----|-----|-----|
| Field         | — | IR6 | IR5 | IR4 | IR3 | IR2 | IR1 | IR0 |
| Attribute     | — | R   | R   | R   | R   | R   | R   | R   |
| Initial value | 0 | X   | X   | X   | X   | X   | X   | X   |

#### ■ Register Functions

##### [bit7] Undefined bit

The read value of this bit is always "0". Writing a value to this bit has no effect on operation.

##### [bit6:0] IR[6:0]: Impedance read bits

These bits reflect the impedance change on the touch pad of touch ch. n. The AIC controller uses the value of the IMPE<sub>n</sub> register for calibrating touch ch. n.

## 26.6.34 TS Touch Data Register (Lower) (TOUHL)

The TS touch data register (lower) (TOUHL) stores the final touch status of touch ch. n.

### ■ Register Configuration

| bit           | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|---------------|------|------|------|------|------|------|------|------|
| Field         | TS07 | TS06 | TS05 | TS04 | TS03 | TS02 | TS01 | TS00 |
| Attribute     | R    | R    | R    | R    | R    | R    | R    | R    |
| Initial value | X    | X    | X    | X    | X    | X    | X    | X    |

### ■ Register Functions

[bit7] TS07: Touch status of touch ch. 7 bit

This bit stores the final touch status of touch ch. 7.

| bit7               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 7 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 7 has been touched. |

[bit6] TS06: Touch status of touch ch. 6 bit

This bit stores the final touch status of touch ch. 6.

| bit6               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 6 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 6 has been touched. |

[bit5] TS05: Touch status of touch ch. 5 bit

This bit stores the final touch status of touch ch. 5.

| bit5               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 5 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 5 has been touched. |

[bit4] TS04: Touch status of touch ch. 4 bit

This bit stores the final touch status of touch ch. 4.

| bit4               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 4 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 4 has been touched. |

[bit3] TS03: Touch status of touch ch. 3 bit

This bit stores the final touch status of touch ch. 3.

| bit3               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 3 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 3 has been touched. |



## MB95850K/860K/870K Series

[bit2] TS02: Touch status of touch ch. 2 bit

This bit stores the final touch status of touch ch. 2.

| bit2               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 2 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 2 has been touched. |

[bit1] TS01: Touch status of touch ch. 1 bit

This bit stores the final touch status of touch ch. 1.

| bit1               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 1 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 1 has been touched. |

[bit0] TS00: Touch status of touch ch. 0 bit

This bit stores the final touch status of touch ch. 0.

| bit0               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 0 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 0 has been touched. |

---

Notes:

- The number of touch channels varies among products. For details, refer to the device data sheet.
  - Whenever there is a change in the value of a bit in the TOUCHL register, TINT rises to "H" to indicate a change in the touch status of the touch channel corresponding to the bit. After rising to "H", TINT remains "H" until the TOUCHL register is read. It falls to "L" immediately after the TOUCHL register is read. Therefore, to enable TINT to rise from "L" to "H" again, read the TOUCHL register.
-

## 26.6.35 TS Touch Data Register (Upper) (TOUCHH)

The TS touch data register (upper) (TOUCHH) stores the final touch status of touch ch. n.

### ■ Register Configuration

| bit           | 7 | 6 | 5 | 4 | 3    | 2    | 1    | 0    |
|---------------|---|---|---|---|------|------|------|------|
| Field         | — | — | — | — | TS11 | TS10 | TS09 | TS08 |
| Attribute     | — | — | — | — | R    | R    | R    | R    |
| Initial value | 0 | 0 | 0 | 0 | X    | X    | X    | X    |

### ■ Register Functions

#### [bit7:4] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit3] TS11: Touch status of touch ch. 11 bit

This bit stores the final touch status of touch ch. 11.

| bit3               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 11 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 11 has been touched. |

#### [bit2] TS10: Touch status of touch ch. 10 bit

This bit stores the final touch status of touch ch. 10.

| bit2               | Details  |
|--------------------|--|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 10 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 10 has been touched. |

#### [bit1] TS09: Touch status of touch ch. 9 bit

This bit stores the final touch status of touch ch. 9.

| bit1               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 9 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 9 has been touched. |

#### [bit0] TS08: Touch status of touch ch. 8 bit

This bit stores the final touch status of touch ch. 8.

| bit0               | Details   |
|--------------------|---|
| <b>Reading "0"</b> | Indicates that the touch pad of touch ch. 8 is not touched.   |
| <b>Reading "1"</b> | Indicates that the touch pad of touch ch. 8 has been touched. |

---

Notes:

- The number of touch channels varies among products. For details, refer to the device data sheet.
  - Whenever there is a change in the value of a bit in the TOUCHH register, TINT rises to "H" to indicate a change in the touch status of the touch channel corresponding to the bit. After rising to "H", TINT remains "H" until the TOUCHH register is read. It falls to "L" immediately after the TOUCHH register is read. Therefore, to enable TINT to rise from "L" to "H" again, read the TOUCHH register.
-



# **CHAPTER 27**

---

# ***BEEP OUTPUT UNIT***

**This chapter describes the functions and operations of the beep output unit.**

27.1 Overview

27.2 Configuration

27.3 Pin

27.4 Operations and Setting Procedure Example

27.5 Register

## **27.1 Overview**

---

**The beep output unit has 14 beep output frequency (square wave) options. It can be used for applications such as sounding a buzzer to confirm key input.**

---

### **■ Features of Beep Output Unit**

The beep output unit has the following features.

- The beep output unit outputs a frequency (square wave) suitable for applications such as sounding a buzzer to confirm an operation.
  - There are 14 beep output frequency options, ten of which are output from the time-base timer and four of which from the watch prescaler. The beep output function can be disabled.
  - The beep output unit can be activated by using the software or the TS, depending on the setting of the software activate signal select bit in the beep output frequency register (BPFREQ:SW).
- 

#### Notes:

- Since frequencies output from the time-base timer and the watch prescaler are input to the beep output unit as beep input signals, when the selected beep input signal source (time-base timer or watch prescaler) is cleared, the beep output unit cannot output any frequency.
  - In subclock mode and sub-CR clock mode, since the time-base timer stops operating as both main clock and main CR clock stop, do not select a frequency output from the time-base timer as a beep output frequency.
-

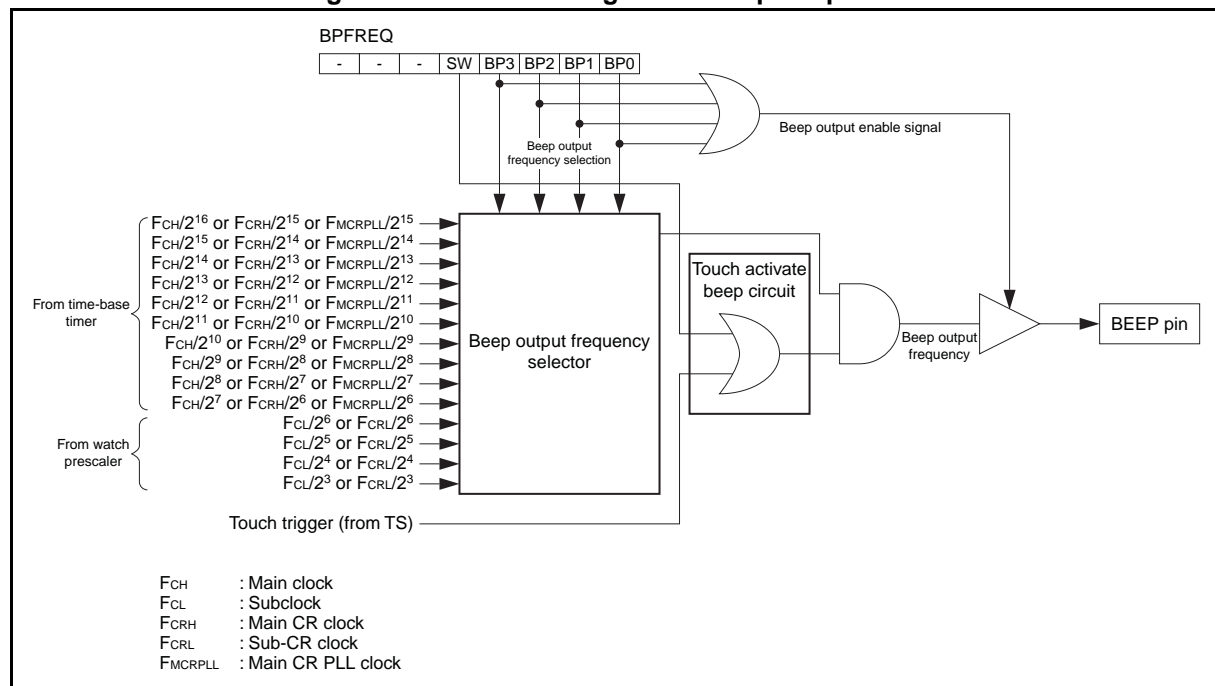
## 27.2 Configuration

The beep output unit consists of the following blocks:

- Beep output frequency selector
- Beep output frequency register (BPFREQ)
- Touch activate beep circuit

### ■ Block Diagram of Beep Output Unit

Figure 27.2-1 Block Diagram of Beep Output Unit



#### ● Beep frequency output selector

The beep frequency output selector is a circuit for selecting a beep output frequency (square wave) from 14 options, ten of which are output from the time-base timer and four of which from the watch prescaler, and outputting the selected beep output frequency.

#### ● Beep output frequency register (BPFREQ)

The beep output frequency register (BPFREQ) selects a method for activating the beep output unit, enables the beep output function and selects a beep output frequency. Writing a value to the BP[3:0] bits in this register to select a beep output frequency automatically enables the beep output function.

#### ● Touch activate beep circuit

The touch activate beep circuit selects a method for activating the beep output unit.

## **27.3 Pin**

---

**This section describes the pin of the beep output unit.**

---

### **■ Pin of Beep Output Unit**

- BEEP pin

This pin serves as a beep output pin.

The BEEP pin outputs a square wave of a frequency specified by the BP[3:0] bits in the BPFREQ register. Writing a value to the BP[3:0] bits to select a beep output frequency automatically make this pin become the beep output pin regardless of the setting of the port direction register (DDR).



## 27.4 Operations and Setting Procedure Example

The beep output unit can be activated by using the software or the TS.

### ■ Activating Beep Output Unit

The beep output unit can be activated by using the software or the TS. To activate the beep output unit by using the software, write "1" to the software activate signal select bit in the beep output frequency register (BPFREQ:SW). To activate the beep output unit by using the TS, write "0" to the SW bit in the BPFREQ register and set the beep duration by using the TS beep duration setting register (BPDUR) of the TS. For details of the BPDUR register, see "26.6.12 TS Beep Duration Setting Register (BPDUR)".

The BP[3:0] bits in the BPFREQ register select a beep output frequency and enables the beep output function.

#### ● Activating by using software

To activate the beep output unit by using the software, do the settings shown in Figure 27.4-1.

Figure 27.4-1 Settings for Activating Beep Output Unit by Using Software

|        | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|--------|------|------|------|------|------|------|------|------|
| BPFREQ | —    | —    | —    | SW   | BP3  | BP2  | BP1  | BP0  |
|        | x    | x    | x    | 1    | ○    | ○    | ○    | ○    |
|        | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| BPDUR  | BDR7 | BDR6 | BDR5 | BDR4 | BDR3 | BDR2 | BDR1 | BDR0 |
|        | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

○ : Bit to be used  
x : Unused bit  
0 : Set to "0"  
1 : Set to "1"

#### ● Activating by using TS

To activate the beep output unit by using the TS, do the settings shown in Figure 27.4-2.

Figure 27.4-2 Settings for Activating Beep Output Unit by Using TS

|        | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|--------|------|------|------|------|------|------|------|------|
| BPFREQ | —    | —    | —    | SW   | BP3  | BP2  | BP1  | BP0  |
|        | x    | x    | x    | 0    | ○    | ○    | ○    | ○    |
|        | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| BPDUR  | BDR7 | BDR6 | BDR5 | BDR4 | BDR3 | BDR2 | BDR1 | BDR0 |
|        | ○    | ○    | ○    | ○    | ○    | ○    | ○    | ○    |

○ : Bit to be used  
x : Unused bit  
0 : Set to "0"  
1 : Set to "1"

■ **Setting Procedure Example**

Below is an example of procedure for setting the beep output unit.

● Initial settings

1. Set the beep output duration. (BPDUR)
2. Select a method for activating the beep output unit. (BPFREQ:SW)
3. Select a beep output frequency. (BPFREQ:BP[3:0])

## 27.5 Register

---

This section describes the register of the beep output unit.

---

**Table 27.5-1 List of Beep Output Unit Register**

| Register abbreviation | Register name                  | Reference |
|-----------------------|--------------------------------|-----------|
| BPFREQ                | Beep output frequency register | 27.5.1    |

## 27.5.1 Beep Output Frequency Register (BPFREQ)

The beep output frequency register (BPFREQ) selects a method for activating the beep output unit, enables the beep output function and selects a beep output frequency.

### Register Configuration

| bit           | 7 | 6 | 5 | 4   | 3   | 2   | 1   | 0   |
|---------------|---|---|---|-----|-----|-----|-----|-----|
| Field         | — | — | — | SW  | BP3 | BP2 | BP1 | BP0 |
| Attribute     | — | — | — | R/W | R/W | R/W | R/W | R/W |
| Initial value | 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   |

### Register Functions

#### [bit7:5] Undefined bits

Their read values are always "0". Writing values to these bits has no effect on operation.

#### [bit4] SW: Software activate signal select bit

This bit selects a method for activating the beep output unit.

| bit4               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Selects activating the beep output unit by using the BPDUR register of the TS. |
| <b>Writing "1"</b> | Selects activating the beep output unit by using the software.                 |

[bit3:0] BP[3:0]: Beep output frequency select bits

These bits select a beep output frequency to be output to the BEEP pin, and enables or disables the beep output function. Selecting a beep output frequency automatically enables the beep output function.

| bit3:0         | Details            |                                      |                                       |  |
|----------------|--------------------|--------------------------------------|---------------------------------------|--|
|                | Frequency supplier | Main clock mode                      | Main CR clock mode                    | Main CR PLL clock mode                   |
|                |                    | $F_{CH} = 4 \text{ MHz}$             | $F_{CRH} = 4 \text{ MHz}$             | $F_{MCRPLL} = 8 \text{ MHz}$             |
| Writing "0000" | —                  | Disables the beep output function.   |                                       |  |
| Writing "0001" | Time-base timer    | Selects $F_{CH}/2^{16}$ (61 Hz).     | Selects $F_{CRH}/2^{15}$ (122 Hz).    | Selects $F_{MCRPLL}/2^{15}$ (244 Hz).    |
| Writing "0010" |                    | Selects $F_{CH}/2^{15}$ (122 Hz).    | Selects $F_{CRH}/2^{14}$ (244 Hz).    | Selects $F_{MCRPLL}/2^{14}$ (488 Hz).    |
| Writing "0011" |                    | Selects $F_{CH}/2^{14}$ (244 Hz).    | Selects $F_{CRH}/2^{13}$ (488 Hz).    | Selects $F_{MCRPLL}/2^{13}$ (977 Hz).    |
| Writing "0100" |                    | Selects $F_{CH}/2^{13}$ (488 Hz).    | Selects $F_{CRH}/2^{12}$ (977 Hz).    | Selects $F_{MCRPLL}/2^{12}$ (1.953 kHz). |
| Writing "0101" |                    | Selects $F_{CH}/2^{12}$ (977 Hz).    | Selects $F_{CRH}/2^{11}$ (1.953 kHz). | Selects $F_{MCRPLL}/2^{11}$ (3.906 kHz). |
| Writing "0110" |                    | Selects $F_{CH}/2^{11}$ (1.953 kHz). | Selects $F_{CRH}/2^{10}$ (3.906 kHz). | Selects $F_{MCRPLL}/2^{10}$ (7.813 kHz). |
| Writing "0111" |                    | Selects $F_{CH}/2^{10}$ (3.906 kHz). | Selects $F_{CRH}/2^9$ (7.813 kHz).    | Selects $F_{MCRPLL}/2^9$ (15.625 kHz).   |
| Writing "1000" |                    | Selects $F_{CH}/2^9$ (7.813 kHz).    | Selects $F_{CRH}/2^8$ (15.625 kHz).   | Selects $F_{MCRPLL}/2^8$ (31.25 kHz).    |
| Writing "1001" |                    | Selects $F_{CH}/2^8$ (15.625 kHz).   | Selects $F_{CRH}/2^7$ (31.25 kHz).    | Selects $F_{MCRPLL}/2^7$ (62.5 kHz).     |
| Writing "1010" |                    | Selects $F_{CH}/2^7$ (31.25 kHz).    | Selects $F_{CRH}/2^6$ (62.5 kHz).     | Selects $F_{MCRPLL}/2^6$ (125 kHz).      |

| bit3:0         | Details            |  |                                     |
|----------------|--------------------|--|-------------------------------------|
|                | Frequency supplier | Subclock mode  | Sub-CR clock mode                   |
|                |                    | $F_{CL} = 32.768 \text{ kHz}$                          | $F_{CRL} = 100 \text{ kHz}$         |
| Writing "1011" | Watch prescaler    | Selects $F_{CL}/2^6$ (512 Hz).                         | Selects $F_{CRL}/2^6$ (1.5625 kHz). |
| Writing "1100" |                    | Selects $F_{CL}/2^5$ (1.024 kHz).                      | Selects $F_{CRL}/2^5$ (3.125 kHz).  |
| Writing "1101" |                    | Selects $F_{CL}/2^4$ (2.048 kHz).                      | Selects $F_{CRL}/2^4$ (6.25 kHz).   |
| Writing "1110" |                    | Selects $F_{CL}/2^3$ (4.096 kHz).                      | Selects $F_{CRL}/2^3$ (12.5 kHz).   |
| Writing "1111" | —                  | Selects "1" as the value to be output to the BEEP pin. |                                     |

Notes:

- In subclock mode and sub-CR clock mode, since the time-base timer stops operating as both main clock and main CR clock stop, do not select a frequency output from the time-base timer as a beep output frequency.
- In main stop mode, the subclock and the sub-CR clock can operate when the respective oscillation enable bits, the SOSCE bit and SCRE bit in the SYCC2 register, are set to "1". Therefore, to use the beep output function in main stop mode, do the following settings:
  - Write "0" to the pin state setting bit in the standby control register (STBC:SPL).
  - Write "1" to the SOSCE bit in the SYCC2 register to enable the subclock oscillation, or "1" to the SCRE bit in the SYCC2 register to enable the sub-CR clock oscillation.
  - Select a beep frequency output from the watch prescaler (BPFREQ:BP[3:0] = 0b1011, 0b1100, 0b1101 or 0b1110).



# **CHAPTER 28**

---

# **SYSTEM CONFIGURATION CONTROLLER**

**This chapter describes the functions and operations of the system configuration controller (called the "controller" in this chapter).**

- 28.1 Overview
- 28.2 Register
- 28.3 Notes on Using Controller

## 28.1 Overview

---

The controller consists of the system configuration register (SYSC), which is an 8-bit register used to configure the clock and reset system, and to select the touch sensor (TS) direct output pins.

---

### ■ Functions of SYSC

- Selecting the general purpose I/O port/reset function for the PF2/ $\overline{\text{RST}}$  pin
- Enabling/disabling reset output for the  $\overline{\text{RST}}$  pin
- Selecting the general purpose I/O port/oscillation function for the PF0/X0 pin and that for the PF1/X1 pin
- Selecting the general purpose I/O port/oscillation function for the PG1/X0A/DIO04 pin and that for the PG2/X1A/DIO03 pin
- Mapping the TS direct output pins DIO03 and DIO04 to P44 and P45 respectively or to PG2 and PG1 respectively



## 28.2 Register

---

This section describes the register of the controller.

---

**Table 28.2-1 List of Controller Register**

| Register abbreviation | Register name                 | Reference |
|-----------------------|-------------------------------|-----------|
| SYSC                  | System configuration register | 28.2.1    |

## 28.2.1 System Configuration Register (SYSC)

This section describes the system configuration register (SYSC).

### ■ Register Configuration

|               |       |       |          |          |          |        |       |       |
|---------------|-------|-------|----------|----------|----------|--------|-------|-------|
| bit           | 7     | 6     | 5        | 4        | 3        | 2      | 1     | 0     |
| Field         | PGSEL | PFSEL | Reserved | Reserved | Reserved | DIOSEL | RSTOE | RSTEN |
| Attribute     | R/W   | R/W   | W        | W        | W        | R/W    | R/W   | R/W   |
| Initial value | 1     | 1     | 0        | 0        | 0        | 0      | 1     | 1     |

### ■ Register Functions

#### [bit7] PGSEL: PG1 and PG2 function select bit

This bit selects the function of the PG1 and PG2 pins.

Writing "0" to this bit makes the PG1 and PG2 pins function as subclock oscillation pins. The subclock oscillation is enabled or disabled by the subclock oscillation enable bit (SYCC2:SOSCE).

Writing "1" to this bit makes the PG1 and PG2 pins function as general-purpose I/O ports.

| bit7               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Makes the PG1 and PG2 pins function as subclock oscillation pins. |
| <b>Writing "1"</b> | Makes the PG1 and PG2 pins function as general-purpose I/O ports. |

#### [bit6] PFSEL: PF0 and PF1 function select bit

This bit selects the function of the PF0 and PF1 pins.

Writing "0" to this bit makes the PF0 and PF1 pins function as main clock oscillation pins. The main clock oscillation is enabled or disabled by the main clock oscillation enable bit (SYCC2:MOSCE).

Writing "1" to this bit makes the PF0 and PF1 pins function as general-purpose I/O ports.

| bit6               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Makes the PF0 and PF1 pins function as main clock oscillation pins. |
| <b>Writing "1"</b> | Makes the PF0 and PF1 pins function as general-purpose I/O ports.   |

#### [bit5:3] Reserved bits

Always set these bits to "0".

#### [bit2] DIOSEL: TS direct output pins select bit

This bit selects which pins the DIO03 pin (TS direct output ch. 3 pin) and the DIO04 pin (TS direct output ch. 4 pin) are mapped.

| bit2               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Selects the PG2 and PG1 pins to which the DIO03 and DIO04 pins are respectively mapped. |
| <b>Writing "1"</b> | Selects the P44 and P45 pins to which the DIO03 and DIO04 pins are respectively mapped. |

#### Relation between TS direct output channels and pins

| TS direct output channel | DIOSEL = 0 | DIOSEL = 1 |
|--------------------------|------------|------------|
| ch. 3                    | PG2        | P44        |
| ch. 4                    | PG1        | P45        |

[bit1] RSTOE: Reset output enable/disable bit

This bit enables or disables the reset output function of the PF2/ $\overline{\text{RST}}$  pin with the reset input function enabled. When the reset input function is disabled (SYSC:RSTEN = 0), the reset output function is disabled regardless of the setting of this bit.

See the PF2 function select bit (SYSC:RSTEN) for details of selecting the reset input function.

| bit1               | Details   |
|--------------------|---|
| <b>Writing "0"</b> | Disables the reset output function of the PF2/ $\overline{\text{RST}}$ pin. |
| <b>Writing "1"</b> | Enables the reset output function of the PF2/ $\overline{\text{RST}}$ pin.  |

[bit0] RSTEN: PF2 function select bit

This bit enables or disables the reset input function of the PF2/ $\overline{\text{RST}}$  pin.

Writing "0" to this bit disables the reset input function of the PF2/ $\overline{\text{RST}}$  pin and enables the general-purpose I/O port function.

Writing "1" to this bit enables the reset input function of the PF2/ $\overline{\text{RST}}$  pin and disables the general-purpose I/O port function.

Set bit2 in the PDRF register to "1" before modifying this bit.

| bit0               | Details  |
|--------------------|--|
| <b>Writing "0"</b> | Selects the general-purpose I/O port function of the PF2/ $\overline{\text{RST}}$ pin. |
| <b>Writing "1"</b> | Selects the reset output function of the PF2/ $\overline{\text{RST}}$ pin.             |

## Note:

To keep the reset input/output function after the reset, SYSC:RSTEN and SYSC:RSTOE are initialized to "1" after the power is switched on. They are not initialized by any other type of reset.

When the reset input/output functions have to be used in a system, it is strongly recommended that SYSC:RSTEN be initialized to "1" in the initialize program routine after a reset for stable operation. With the reset input/output functions having been enabled, all types of reset, including the watchdog reset, can be used.

## 28.3 Notes on Using Controller

---

This section provides notes on using the controller.

---

### ■ Setting DIOSEL to "0" When Using 8/16-bit Composite Timer

While the output function of the 8/16-bit composite timer is in use, the P44 and P45 pins are being used as 8/16-bit composite timer output pins. In this situation, to use the direct output function of the TS, set the DIOSEL bit to "0" to map the TS direct output pins to the PG1 and PG2 pins.

# APPENDIX

---

**This section provides an overview of instructions.**

- A.1 Addressing
- A.2 Special Instruction
- A.3 Bit Manipulation Instructions (SETB, CLRB)
- A.4 F<sup>2</sup>MC-8FX Instructions
- A.5 Instruction Map

## APPENDIX A Instruction Overview

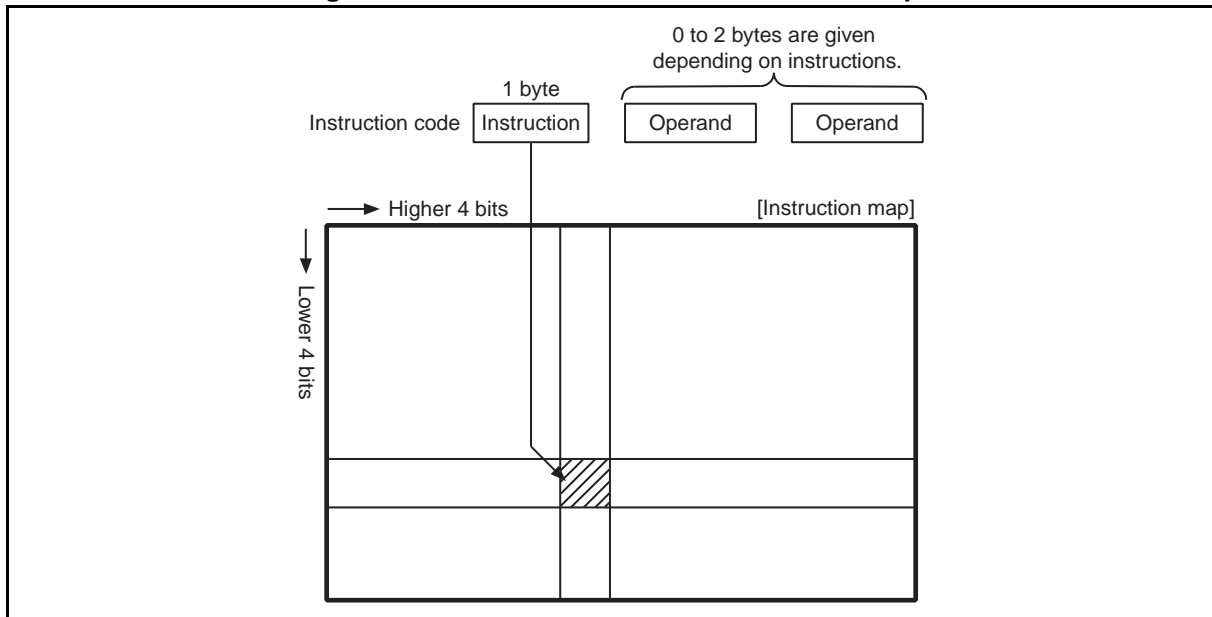
This section explains the instructions used in F<sup>2</sup>MC-8FX.

### ■ Instruction Overview of F<sup>2</sup>MC-8FX

In the F<sup>2</sup>MC-8FX, there are 140 kinds of one byte instructions (256 bytes on the map), and the instruction code is composed of the instruction and the operand following it.

Figure A-1 shows the correspondence of the instruction code and the instruction map.

**Figure A-1 Instruction Code and Instruction Map**



- The instruction is classified into following four types; forwarding system, operation system, branch system and others.
- There are various methods of addressing, and ten kinds of addressing can be selected by the selection and the operand specification of the instruction.
- This provides with the bit operation instruction, and can execute the read-modify-write (RMW) type of instruction.
- There is an instruction that directs special operation.

Code: CM26-00118-1EA

## MB95850K/860K/870K Series

### ■ Meanings of Signs in Instruction Codes

Table A-1 shows the meanings of signs used in explaining instruction codes in APPENDIX A.

**Table A-1 Meanings of Signs in Instruction Codes**

| Sign  | Meanings   |
|-------|--|
| dir   | Direct address (8-bit length)  |
| off   | Offset (8-bit length)  |
| ext   | Extended address (16-bit length)   |
| #vct  | Vector table number (3-bit length)   |
| #d8   | Immediate data (8-bit length)  |
| #d16  | Immediate data (16-bit length)   |
| dir:b | Bit direct address (8-bit length: 3-bit length)  |
| rel   | Branch relative address (8-bit length)   |
| @     | Register indirect (Example: @A, @IX, @EP)  |
| A     | Accumulator (Whether 8- bit length or 16- bit length is decided by the instruction used.)  |
| AH    | Upper 8-bit of accumulator (8-bit length)  |
| AL    | Lower 8-bit of accumulator (8-bit length)  |
| T     | Temporary accumulator (Whether 8- bit length or 16- bit length is decided by the instruction used.)  |
| TH    | Upper 8-bit of temporary accumulator (8-bit length)  |
| TL    | Lower 8-bit of temporary accumulator (8-bit length)  |
| IX    | Index register (16-bit length)   |
| EP    | Extra pointer (16-bit length)  |
| PC    | Program counter (16-bit length)  |
| SP    | Stack pointer (16-bit length)  |
| PS    | Program status (16-bit length)   |
| dr    | Either of accumulator or index register (16-bit length)  |
| CCR   | Condition code register (8-bit length)   |
| RP    | Register bank pointer (5-bit length)   |
| DP    | Direct bank pointer (3-bit length)   |
| Ri    | General-purpose register (8-bit length, i = 0 to 7)  |
| x     | This shows that x is immediate data.<br>(Whether 8- bit length or 16- bit length is decided by the instruction used.)  |
| (x)   | This shows that contents of x are objects of the access.<br>(Whether 8- bit length or 16- bit length is decided by the instruction used.)                        |
| ((x)) | This shows that the address that contents of x show is an object of the access.<br>(Whether 8- bit length or 16- bit length is decided by the instruction used.) |

## ■ Meanings of Items in Instruction Table

**Table A-2 Meanings of Items in Instruction Table**

| Item       | Meaning   |
|------------|---|
| MNEMONIC   | It shows the assembly description of the instruction.   |
| ~          | It shows the number of cycles of the instruction. One instruction cycle is a machine cycle.<br>Note:<br>The number of cycles of the instruction can be delayed by 1 cycle by the immediately preceding instruction. Moreover, the number of cycles of the instruction might be extended in the access to the I/O area.  |
| #          | It shows the number of bytes for the instruction.   |
| Operation  | It shows the operations for the instruction.  |
| TL, TH, AH | They show the change (auto forwarding from A to T) in the content when each TL, TH, and AH instruction is executed. The sign in the column indicates the followings respectively. <ul style="list-style-type: none"> <li>• -: No change</li> <li>• dH: upper 8 bits of the data described in operation.</li> <li>• AL and AH: the contents become those of the immediately preceding instruction's AL and AH.</li> <li>• 00: Become 00</li> </ul> |
| N, Z, V, C | They show the instruction into which the corresponding flag is changed respectively. The sign in the column shows the followings respectively. <ul style="list-style-type: none"> <li>• -: No change</li> <li>• +: Change</li> <li>• R: Become "0"</li> <li>• S: Become "1"</li> </ul>  |
| OP CODE    | It shows the code of the instruction. When a pertinent instruction occupies two or more codes, it follows the following description rules.<br>[Example] 48 to 4F: This shows 48, 49....4F.  |



# MB95850K/860K/870K Series

## A.1 Addressing

F<sup>2</sup>MC-8FX has the following ten types of addressing:

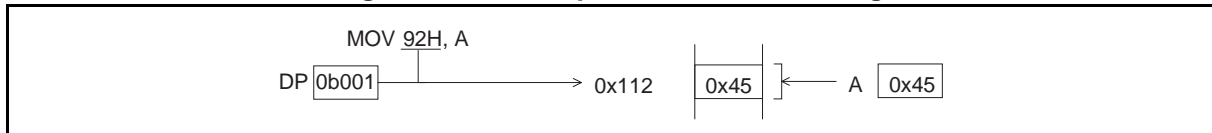
- Direct addressing
- Extended addressing
- Bit direct addressing
- Index addressing
- Pointer addressing
- General-purpose register addressing
- Immediate addressing
- Vector addressing
- Relative addressing
- Inherent addressing

### ■ Explanation of Addressing

#### ● Direct addressing

This is used when accessing the direct area of "0x0000" to "0x047F" with addressing indicated "dir" in instruction table. In this addressing, when the operand address is "0x00" to "0x7F", it is accessed into "0x0000" to "0x007F". Moreover, when the operand address is "0x80" to "0xFF", the access can be mapped in "0x0080" to "0x047F" by setting of direct bank pointer DP. Figure A.1-1 shows an example.

**Figure A.1-1 Example of Direct Addressing**

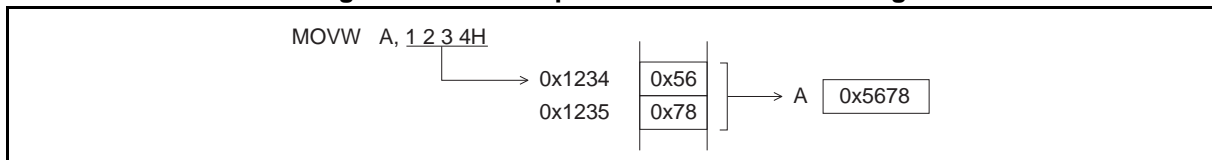


#### ● Extended addressing

This is used when the area of the entire 64 Kbyte is accessed by addressing shown "ext" in the instruction table. In this addressing, the first operand specifies one high rank byte of the address and the second operand specifies one subordinate position byte of the address.

Figure A.1-2 shows an example.

**Figure A.1-2 Example of Extended Addressing**

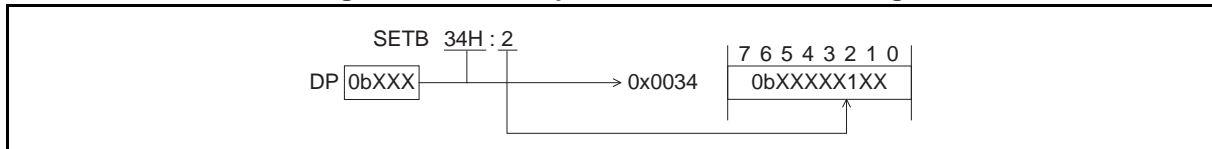


● Bit direct addressing

This is used when accessing the direct area of "0x0000" to "0x047F" in bit unit with addressing indicated "dir:b" in instruction table. In this addressing, when the operand address is "0x00" to "0x7F", it is accessed into "0x0000" to "0x007F". Moreover, when the operand address is "0x80" to "0xFF", the access can be mapped in "0x0080" to "0x047F" by setting of direct bank pointer DP. The position of the bit in the specified address is specified by the values of the instruction code of three subordinate position bits.

Figure A.1-3 shows an example.

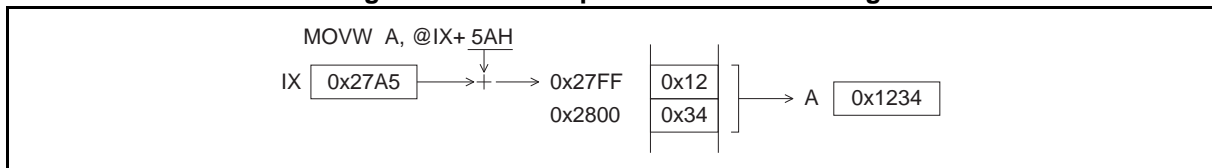
**Figure A.1-3 Example of Bit Direct Addressing**



● Index addressing

This is used when the area of the entire 64 Kbyte is accessed by addressing shown "@IX+off" in the instruction table. In this addressing, the content of the first operand is sign extended and added to IX (index register) to the resulting address. Figure A.1-4 shows an example.

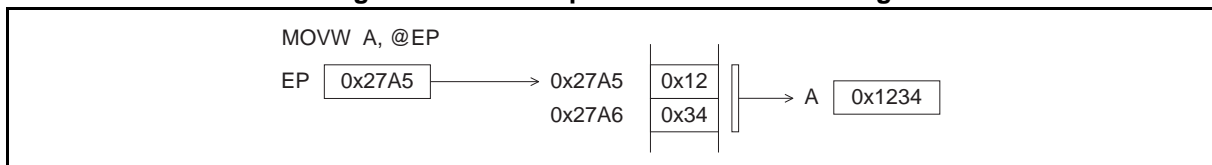
**Figure A.1-4 Example of Index Addressing**



● Pointer addressing

This is used when the area of the entire 64 Kbyte is accessed by addressing shown "@EP" in the instruction table. In this addressing, the content of EP (extra pointer) is assumed to be an address. Figure A.1-5 shows an example.

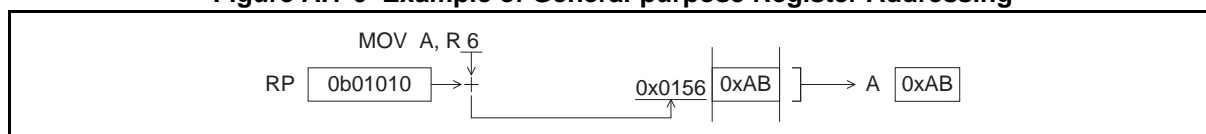
**Figure A.1-5 Example of Pointer Addressing**



● General-purpose register addressing

This is used when accessing the register bank in general-purpose register area with the addressing shown "Ri" in instruction table. In this addressing, fix one high rank byte of the address to "01" and create one subordinate position byte from the contents of RP (register bank pointer) and three subordinate bits of the operation code to access to this address. Figure A.1-6 shows an example.

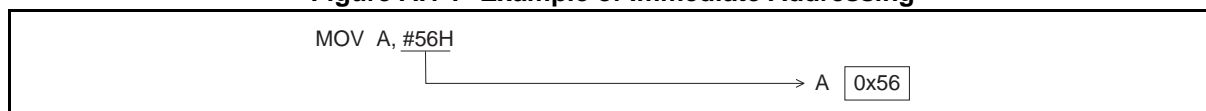
Figure A.1-6 Example of General-purpose Register Addressing



● Immediate addressing

This is used when immediate data is needed in addressing shown "#d8" in the instruction table. In this addressing, the operand becomes immediate data as it is. The specification of byte/word depends on the operation code. Figure A.1-7 shows an example.

Figure A.1-7 Example of Immediate Addressing



● Vector addressing

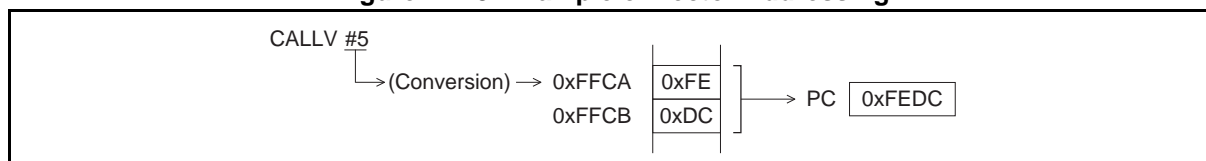
This is used when branching to the subroutine address registered in the table with the addressing shown "#vct" in the instruction table. In this addressing, information on "#vct" is contained in the operation code, and the address of the table is created using the combinations shown in Table A.1-1.

Table A.1-1 Vector Table Address Corresponding to "#vct"

| #vct | Vector table address<br>(jump destination high-ranking address: subordinate address) |
|------|--|
| 0    | 0xFFC0 : 0xFFC1  |
| 1    | 0xFFC2 : 0xFFC3  |
| 2    | 0xFFC4 : 0xFFC5  |
| 3    | 0xFFC6 : 0xFFC7  |
| 4    | 0xFFC8 : 0xFFC9  |
| 5    | 0xFFCA : 0xFFCB  |
| 6    | 0xFFCC : 0xFFCD  |
| 7    | 0xFFCE : 0xFFCF  |

Figure A.1-8 shows an example.

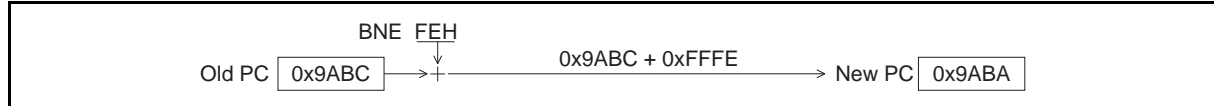
Figure A.1-8 Example of Vector Addressing



## ● Relative addressing

This is used when branching to the area in 128 bytes before and behind PC (program counter) with the addressing shown "rel" in the instruction table. In this addressing, add the content of the operand to PC with the sign and store the result in PC. Figure A.1-9 shows an example.

**Figure A.1-9 Example of Relative Addressing**

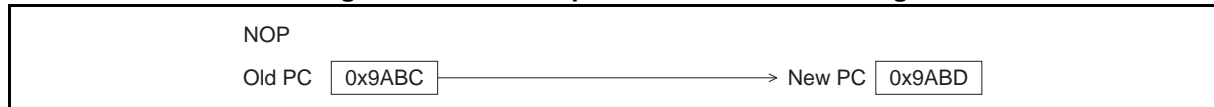


In this example, by jumping to the address where the operation code of BNE is stored, it results in an infinite loop.

## ● Inherent addressing

This is used when doing the operation decided by the operation code with the addressing that does not have the operand in the instruction table. In this addressing, the operation depends on each instruction. Figure A.1-10 shows an example.

**Figure A.1-10 Example of Inherent Addressing**



# MB95850K/860K/870K Series

## A.2 Special Instruction

This section explains special instructions other than the addressing.

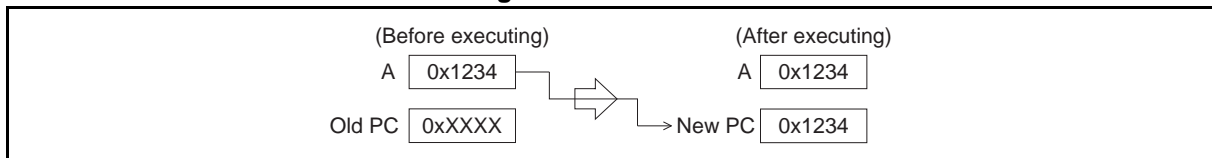
### ■ Special Instruction

#### ● JMP @A

This instruction is to branch the content of A (accumulator) to PC (program counter) as an address. N pieces of the jump destination is arranged on the table, and one of the contents is selected and transferred to A. N branch processing can be done by executing this instruction.

Figure A.2-1 shows a summary of the instruction.

**Figure A.2-1 JMP @A**

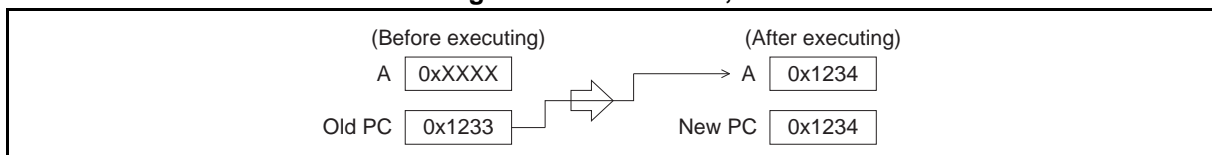


#### ● MOVW A, PC

This instruction works as the opposite of "JMP @A". That is, it stores the content of PC to A. When you have executed this instruction in the main routine and set it to call a specific subroutine, you can make sure that the content of A is the specified value in the subroutine. Also, you can identify that the branch is not from the part that cannot be expected, and use it for the reckless driving judgment.

Figure A.2-2 shows a summary of the instruction.

**Figure A.2-2 MOVW A, PC**



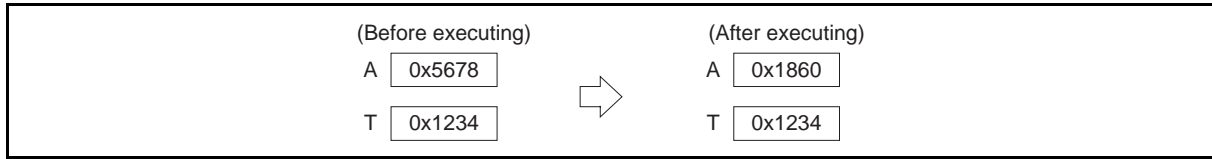
When this instruction is executed, the content of A reaches the same value as the address where the following instruction is stored, rather than the address where operation code of this instruction is stored. Therefore, in Figure A.2-2, the value "0x1234" stored in A corresponds to the address where the following operation code of "MOVW A, PC" is stored.

#### ● MULU A

This instruction performs an unsigned multiplication of AL (lower 8-bit of the accumulator) and TL (lower 8-bit 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 8-bit of the accumulator) and TH (higher 8-bit of the temporary accumulator) before execution of the instruction are not used for the operation. Note that since the instruction does not change the flags, a branch may occur depending on the multiplication result.

Figure A.2-3 shows a summary of the instruction.

**Figure A.2-3 MULU A**

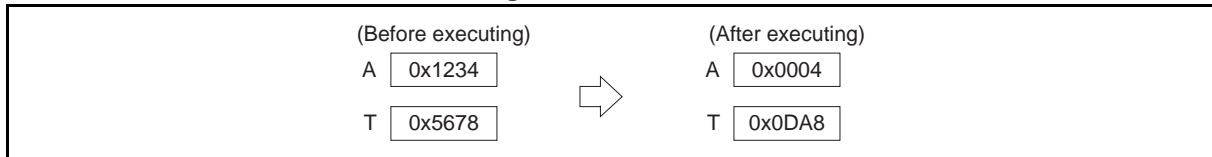


● **DIVU A**

This instruction divides the 16-bit value in T by the unsigned 16-bit value in A, and stores the 16-bit result and the 16-bit remainder in A and T, respectively. When the value in A before execution of instruction is "0", the Z flag becomes "1" to indicate zero-division is executed. Note that since the instruction does not change other flags, a branch may occur depending on the division result.

Figure A.2-4 shows a summary of the instruction.

**Figure A.2-4 DIVU A**

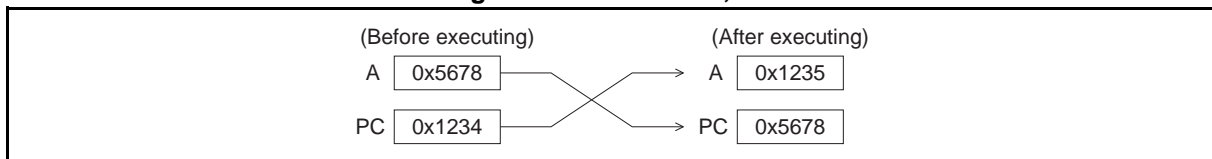


● **XCHW A, PC**

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 becomes the address that follows the address where the operation code of "XCHW A, PC" is stored. This instruction is effective especially when it is used in the main routine to specify a table for use in a subroutine.

Figure A.2-5 shows a summary of the instruction.

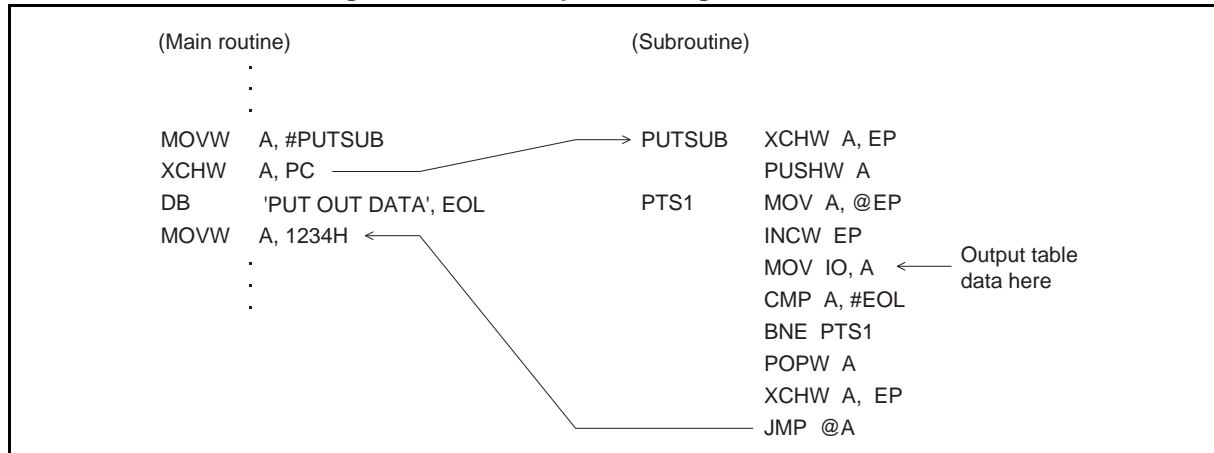
**Figure A.2-5 XCHW A, PC**



When this instruction is executed, the content of A reaches the same value as the address where the following instruction is stored, rather than the address where operation code of this instruction is stored. Therefore, in Figure A.2-5, the value "0x1235" stored in A corresponds to the address where the following operation code of "XCHW A, PC" is stored. This is why "0x1235" is stored instead of "0x1234".

Figure A.2-6 shows an assembler language example.

Figure A.2-6 Example of Using "XCHW A, PC"

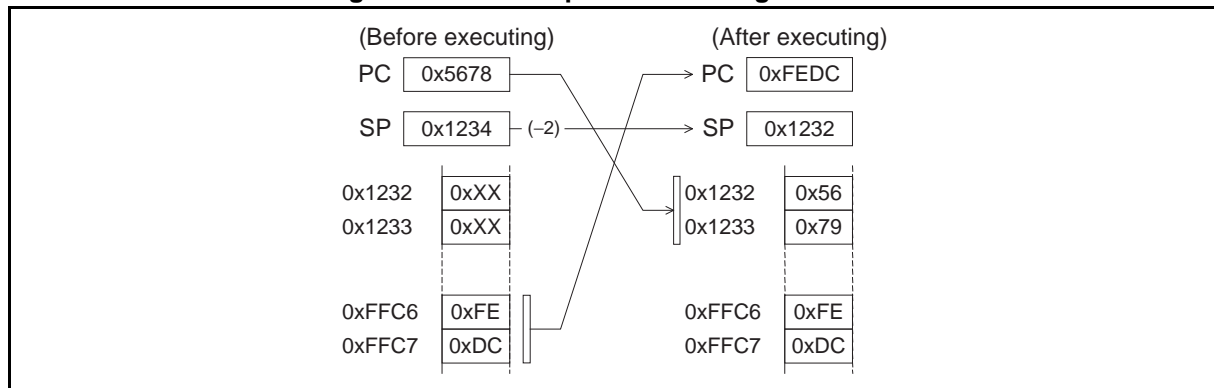


● CALLV #vct

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 A.2-7 shows a summary of the instruction.

Figure A.2-7 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 A.2-7 shows that the value saved in the stack (0x1232 and 0x1233) is 0x5679, which is the address of the operation code of the instruction that follows "CALLV vct" (return address).

**Table A.2-1 Vector Table**

| Vector use<br>(call instruction) | Vector table address |        |
|----------------------------------|----------------------|--------|
|                                  | Upper                | Lower  |
| CALLV #7                         | 0xFFCE               | 0xFFCF |
| CALLV #6                         | 0xFFCC               | 0xFFCD |
| CALLV #5                         | 0xFFCA               | 0xFFCB |
| CALLV #4                         | 0xFFC8               | 0xFFC9 |
| CALLV #3                         | 0xFFC6               | 0xFFC7 |
| CALLV #2                         | 0xFFC4               | 0xFFC5 |
| CALLV #1                         | 0xFFC2               | 0xFFC3 |
| CALLV #0                         | 0xFFC0               | 0xFFC1 |



### A.3 Bit Manipulation Instructions (SETB, CLRB)

Some peripheral function registers include bits that are read differently than usual by a bit manipulation instruction.

#### ■ Read-modify-write Operation

By using these bit manipulation instructions, you can set only the specified bit in a register or RAM location to "1" (SETB) or clear to "0" (CLRB). However, as the CPU operates 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 A.3-1 shows bus operation for bit manipulation instructions.

**Table A.3-1 Bus Operation for Bit Manipulation Instructions**

| CODE     | MNEMONIC   | ~ | Cycle | Address bus | Data bus               | RD | WR | RMW |
|----------|------------|---|-------|-------------|------------------------|----|----|-----|
| A0 to A7 | CLRB dir:b | 4 | 1     | N+2         | Next instruction       | 1  | 0  | 1   |
|          |            |   | 2     | dir address | Data                   | 1  | 0  | 1   |
| A8 to AF | SETB dir:b |   | 3     | dir address | Data                   | 0  | 1  | 0   |
|          |            |   | 4     | N+3         | Instruction after next | 1  | 0  | 0   |

#### ■ Read Destination on the Execution of Bit Manipulation Instructions

For some I/O ports and the interrupt request flag bits, the read destination 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 a port data register value is read during a bit manipulation. This prevents the other port data register 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 writing the value "0" to the interrupt request flag bit when manipulating another bit.

## A.4 F<sup>2</sup>MC-8FX Instructions

Table A.4-1 to Table A.4-4 show the instructions used by F<sup>2</sup>MC-8FX.

### ■ Transfer Instructions

Table A.4-1 Transfer Instructions

| No. | MNEMONIC | ~              | # | Operation | TL   | TH | AH | N  | Z | V | C | OPCODE   |
|-----|----------|----------------|---|-----------|--|----|----|----|---|---|---|----------|
| 1   | MOV      | dir, A         | 3 | 2         | (dir) ← (A)                                  | -  | -  | -  | - | - | - | 45       |
| 2   | MOV      | @IX + off, A   | 3 | 2         | ((IX) + off) ← (A)                           | -  | -  | -  | - | - | - | 46       |
| 3   | MOV      | ext, A         | 4 | 3         | (ext) ← (A)                                  | -  | -  | -  | - | - | - | 61       |
| 4   | MOV      | @EP, A         | 2 | 1         | ((EP)) ← (A)                                 | -  | -  | -  | - | - | - | 47       |
| 5   | MOV      | Ri, A          | 2 | 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   | 3 | 2         | (A) ← ((IX) + off)                           | AL | -  | -  | + | + | - | 06       |
| 9   | MOV      | A, ext         | 4 | 3         | (A) ← (ext)                                  | AL | -  | -  | + | + | - | 60       |
| 10  | MOV      | A, @A          | 2 | 1         | (A) ← ((A))                                  | AL | -  | -  | + | + | - | 92       |
| 11  | MOV      | A, @EP         | 2 | 1         | (A) ← ((EP))                                 | AL | -  | -  | + | + | - | 07       |
| 12  | MOV      | A, Ri          | 2 | 1         | (A) ← (Ri)                                   | AL | -  | -  | + | + | - | 08 to 0F |
| 13  | MOV      | dir, #d8       | 4 | 3         | (dir) ← d8                                   | -  | -  | -  | - | - | - | 85       |
| 14  | MOV      | @IX + off, #d8 | 4 | 3         | ((IX) + off) ← d8                            | -  | -  | -  | - | - | - | 86       |
| 15  | MOV      | @EP, #d8       | 3 | 2         | ((EP)) ← d8                                  | -  | -  | -  | - | - | - | 87       |
| 16  | MOV      | Ri, #d8        | 3 | 2         | (Ri) ← d8                                    | -  | -  | -  | - | - | - | 88 to 8F |
| 17  | MOVW     | dir, A         | 4 | 2         | (dir) ← (AH), (dir + 1) ← (AL)               | -  | -  | -  | - | - | - | D5       |
| 18  | MOVW     | @IX + off, A   | 4 | 2         | ((IX) + off) ← (AH), ((IX) + off + 1) ← (AL) | -  | -  | -  | - | - | - | D6       |
| 19  | MOVW     | ext, A         | 5 | 3         | (ext) ← (AH), (ext + 1) ← (AL)               | -  | -  | -  | - | - | - | D4       |
| 20  | MOVW     | @EP, A         | 3 | 1         | ((EP)) ← (AH), ((EP) + 1) ← (AL)             | -  | -  | -  | - | - | - | D7       |
| 21  | MOVW     | EP, A          | 1 | 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       |
| 24  | MOVW     | A, @IX + off   | 4 | 2         | (AH) ← ((IX) + off), (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          | 3 | 1         | (AH) ← ((A)), (AL) ← ((A) + 1)               | AL | AH | dH | + | + | - | 93       |
| 27  | MOVW     | A, @EP         | 3 | 1         | (AH) ← ((EP)), (AL) ← ((EP) + 1)             | AL | AH | dH | + | + | - | C7       |
| 28  | MOVW     | A, EP          | 1 | 1         | (A) ← (EP)                                   | -  | -  | dH | - | - | - | F3       |
| 29  | MOVW     | EP, #d16       | 3 | 3         | (EP) ← d16                                   | -  | -  | -  | - | - | - | E7       |
| 30  | MOVW     | IX, A          | 1 | 1         | (IX) ← (A)                                   | -  | -  | -  | - | - | - | E2       |
| 31  | MOVW     | A, IX          | 1 | 1         | (A) ← (IX)                                   | -  | -  | dH | - | - | - | F2       |
| 32  | MOVW     | SP, A          | 1 | 1         | (SP) ← (A)                                   | -  | -  | -  | - | - | - | E1       |
| 33  | MOVW     | A, SP          | 1 | 1         | (A) ← (SP)                                   | -  | -  | dH | - | - | - | F1       |
| 34  | MOV      | @A, T          | 2 | 1         | ((A)) ← (T)                                  | -  | -  | -  | - | - | - | 82       |
| 35  | MOVW     | @A, T          | 3 | 1         | ((A)) ← (TH), ((A) + 1) ← (TL)               | -  | -  | -  | - | - | - | 83       |
| 36  | MOVW     | IX, #d16       | 3 | 3         | (IX) ← d16                                   | -  | -  | -  | - | - | - | E6       |
| 37  | MOVW     | A, PS          | 1 | 1         | (A) ← (PS)                                   | -  | -  | dH | - | - | - | 70       |
| 38  | MOVW     | PS, A          | 1 | 1         | (PS) ← (A)                                   | -  | -  | -  | + | + | + | 71       |
| 39  | MOVW     | SP, #d16       | 3 | 3         | (SP) ← d16                                   | -  | -  | -  | - | - | - | E5       |
| 40  | SWAP     |                | 1 | 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           | 1 | 1         | (AL) ↔ (TL)                                  | AL | -  | -  | - | - | - | 42       |
| 44  | XCHW     | A, T           | 1 | 1         | (A) ↔ (T)                                    | AL | AH | dH | - | - | - | 43       |
| 45  | XCHW     | A, EP          | 1 | 1         | (A) ↔ (EP)                                   | -  | -  | dH | - | - | - | F7       |
| 46  | XCHW     | A, IX          | 1 | 1         | (A) ↔ (IX)                                   | -  | -  | dH | - | - | - | F6       |
| 47  | XCHW     | A, SP          | 1 | 1         | (A) ↔ (SP)                                   | -  | -  | dH | - | - | - | F5       |
| 48  | MOVW     | A, PC          | 2 | 1         | (A) ← (PC)                                   | -  | -  | dH | - | - | - | F0       |

Note:

In automatic transfer to T during byte transfer to A, AL is transferred to TL.  
If an instruction has plural operands, they are saved in the order indicated by MNEMONIC.

## ■ Arithmetic Operation Instructions

Table A.4-2 Arithmetic Operation Instruction (1 / 2)

| No. | MNEMONIC          | ~  | # | Operation  | TL | TH | AH | N | Z | V | C | OPCODE   |
|-----|-------------------|----|---|--|----|----|----|---|---|---|---|----------|
| 1   | ADDC A, Ri        | 2  | 1 | $(A) \leftarrow (A) + (Ri) + C$  | -  | -  | -  | + | + | + | + | 28 to 2F |
| 2   | ADDC A, #d8       | 2  | 2 | $(A) \leftarrow (A) + d8 + C$  | -  | -  | -  | + | + | + | + | 24       |
| 3   | ADDC A, dir       | 3  | 2 | $(A) \leftarrow (A) + (dir) + C$   | -  | -  | -  | + | + | + | + | 25       |
| 4   | ADDC A, @IX + off | 3  | 2 | $(A) \leftarrow (A) + ((IX) + off) + C$  | -  | -  | -  | + | + | + | + | 26       |
| 5   | ADDC A, @EP       | 2  | 1 | $(A) \leftarrow (A) + ((EP)) + C$  | -  | -  | -  | + | + | + | + | 27       |
| 6   | ADDCW A           | 1  | 1 | $(A) \leftarrow (A) + (T) + C$   | -  | -  | dH | + | + | + | + | 23       |
| 7   | ADDC A            | 1  | 1 | $(AL) \leftarrow (AL) + (TL) + C$  | -  | -  | -  | + | + | + | + | 22       |
| 8   | SUBC A, Ri        | 2  | 1 | $(A) \leftarrow (A) - (Ri) - C$  | -  | -  | -  | + | + | + | + | 38 to 3F |
| 9   | SUBC A, #d8       | 2  | 2 | $(A) \leftarrow (A) - d8 - C$  | -  | -  | -  | + | + | + | + | 34       |
| 10  | SUBC A, dir       | 3  | 2 | $(A) \leftarrow (A) - (dir) - C$   | -  | -  | -  | + | + | + | + | 35       |
| 11  | SUBC A, @IX + off | 3  | 2 | $(A) \leftarrow (A) - ((IX) + off) - C$  | -  | -  | -  | + | + | + | + | 36       |
| 12  | SUBC A, @EP       | 2  | 1 | $(A) \leftarrow (A) - ((EP)) - C$  | -  | -  | -  | + | + | + | + | 37       |
| 13  | SUBCW A           | 1  | 1 | $(A) \leftarrow (T) - (A) - C$   | -  | -  | dH | + | + | + | + | 33       |
| 14  | SUBC A            | 1  | 1 | $(AL) \leftarrow (TL) - (AL) - C$  | -  | -  | -  | + | + | + | + | 32       |
| 15  | INC Ri            | 3  | 1 | $(Ri) \leftarrow (Ri) + 1$   | -  | -  | -  | + | + | + | - | C8 to CF |
| 16  | INCW EP           | 1  | 1 | $(EP) \leftarrow (EP) + 1$   | -  | -  | -  | - | - | - | - | C3       |
| 17  | INCW IX           | 1  | 1 | $(IX) \leftarrow (IX) + 1$   | -  | -  | -  | - | - | - | - | C2       |
| 18  | INCW A            | 1  | 1 | $(A) \leftarrow (A) + 1$   | -  | -  | dH | + | + | - | - | C0       |
| 19  | DEC Ri            | 3  | 1 | $(Ri) \leftarrow (Ri) - 1$   | -  | -  | -  | + | + | + | + | D8 to DF |
| 20  | DECW EP           | 1  | 1 | $(EP) \leftarrow (EP) - 1$   | -  | -  | -  | - | - | - | - | D3       |
| 21  | DECW IX           | 1  | 1 | $(IX) \leftarrow (IX) - 1$   | -  | -  | -  | - | - | - | - | D2       |
| 22  | DECW A            | 1  | 1 | $(A) \leftarrow (A) - 1$   | -  | -  | dH | + | + | - | - | D0       |
| 23  | MULU A            | 8  | 1 | $(A) \leftarrow (AL) \times (TL)$  | -  | -  | dH | - | - | - | - | 01       |
| 24  | DIVU A            | 17 | 1 | $(A) \leftarrow (T) / (A), \text{MOD} \rightarrow (T)$                                   | dL | dH | dH | - | + | - | - | 11       |
| 25  | ANDW A            | 1  | 1 | $(A) \leftarrow (A) \wedge (T)$  | -  | -  | dH | + | + | R | - | 63       |
| 26  | ORW A             | 1  | 1 | $(A) \leftarrow (A) \vee (T)$  | -  | -  | dH | + | + | R | - | 73       |
| 27  | XORW A            | 1  | 1 | $(A) \leftarrow (A) \vee (T)$  | -  | -  | dH | + | + | R | - | 53       |
| 28  | CMP A             | 1  | 1 | $(TL) - (AL)$  | -  | -  | -  | + | + | + | + | 12       |
| 29  | CMPW A            | 1  | 1 | $(T) - (A)$  | -  | -  | -  | + | + | + | + | 13       |
| 30  | RORC A            | 1  | 1 | $\overset{\curvearrowright}{\curvearrowright} C \rightarrow A \leftarrow A \leftarrow C$ | -  | -  | -  | + | + | - | + | 03       |
| 31  | ROLC A            | 1  | 1 | $\overset{\curvearrowleft}{\curvearrowleft} C \leftarrow A \leftarrow A \leftarrow C$    | -  | -  | -  | + | + | - | + | 02       |
| 32  | CMP A, #d8        | 2  | 2 | $(A) - d8$   | -  | -  | -  | + | + | + | + | 14       |
| 33  | CMP A, dir        | 3  | 2 | $(A) - (dir)$  | -  | -  | -  | + | + | + | + | 15       |
| 34  | CMP A, @EP        | 2  | 1 | $(A) - ((EP))$   | -  | -  | -  | + | + | + | + | 17       |
| 35  | CMP A, @IX + off  | 3  | 2 | $(A) - ((IX) + off)$   | -  | -  | -  | + | + | + | + | 16       |
| 36  | CMP A, Ri         | 2  | 1 | $(A) - (Ri)$   | -  | -  | -  | + | + | + | + | 18 to 1F |
| 37  | DAA               | 1  | 1 | decimal adjust for addition  | -  | -  | -  | + | + | + | + | 84       |
| 38  | DAS               | 1  | 1 | decimal adjust for subtraction   | -  | -  | -  | + | + | + | + | 94       |
| 39  | XOR A             | 1  | 1 | $(A) \leftarrow (AL) \vee (TL)$  | -  | -  | -  | + | + | R | - | 52       |
| 40  | XOR A, #d8        | 2  | 2 | $(A) \leftarrow (AL) \vee d8$  | -  | -  | -  | + | + | R | - | 54       |
| 41  | XOR A, dir        | 3  | 2 | $(A) \leftarrow (AL) \vee (dir)$   | -  | -  | -  | + | + | R | - | 55       |
| 42  | XOR A, @EP        | 2  | 1 | $(A) \leftarrow (AL) \vee ((EP))$  | -  | -  | -  | + | + | R | - | 57       |
| 43  | XOR A, @IX + off  | 3  | 2 | $(A) \leftarrow (AL) \vee ((IX) + off)$  | -  | -  | -  | + | + | R | - | 56       |
| 44  | XOR A, Ri         | 2  | 1 | $(A) \leftarrow (AL) \vee (Ri)$  | -  | -  | -  | + | + | R | - | 58 to 5F |
| 45  | AND A             | 1  | 1 | $(A) \leftarrow (AL) \wedge (TL)$  | -  | -  | -  | + | + | R | - | 62       |
| 46  | AND A, #d8        | 2  | 2 | $(A) \leftarrow (AL) \wedge d8$  | -  | -  | -  | + | + | R | - | 64       |
| 47  | AND A, dir        | 3  | 2 | $(A) \leftarrow (AL) \wedge (dir)$   | -  | -  | -  | + | + | R | - | 65       |
| 48  | AND A, @EP        | 2  | 1 | $(A) \leftarrow (AL) \wedge ((EP))$  | -  | -  | -  | + | + | R | - | 67       |
| 49  | AND A, @IX + off  | 3  | 2 | $(A) \leftarrow (AL) \wedge ((IX) + off)$  | -  | -  | -  | + | + | R | - | 66       |

**Table A.4-2 Arithmetic Operation Instruction (2 / 2)**

| No. | MNEMONIC           | ~ | # | Operation                 | TL | TH | AH | N | Z | V | C | OPCODE   |
|-----|--------------------|---|---|---------------------------|----|----|----|---|---|---|---|----------|
| 50  | AND A, Ri          | 2 | 1 | (A) ← (AL) ∧ (Ri)         | -  | -  | -  | + | + | R | - | 68 to 6F |
| 51  | OR A               | 1 | 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          | 2 | 1 | (A) ← (AL) ∨ ((EP))       | -  | -  | -  | + | + | R | - | 77       |
| 55  | OR A, @IX + off    | 3 | 2 | (A) ← (AL) ∨ ((IX) + off) | -  | -  | -  | + | + | R | - | 76       |
| 56  | OR A, Ri           | 2 | 1 | (A) ← (AL) ∨ (Ri)         | -  | -  | -  | + | + | R | - | 78 to 7F |
| 57  | CMP dir, #d8       | 4 | 3 | (dir) - d8                | -  | -  | -  | + | + | + | + | 95       |
| 58  | CMP @EP, #d8       | 3 | 2 | ((EP)) - d8               | -  | -  | -  | + | + | + | + | 97       |
| 59  | CMP @IX + off, #d8 | 4 | 3 | ((IX) + off) - d8         | -  | -  | -  | + | + | + | + | 96       |
| 60  | CMP Ri, #d8        | 3 | 2 | (Ri) - d8                 | -  | -  | -  | + | + | + | + | 98 to 9F |
| 61  | INCW SP            | 1 | 1 | (SP) ← (SP) + 1           | -  | -  | -  | - | - | - | - | C1       |
| 62  | DECW SP            | 1 | 1 | (SP) ← (SP) - 1           | -  | -  | -  | - | - | - | - | D1       |

■ **Branch Instructions**

**Table A.4-3 Branch Instructions**

| No. | MNEMONIC                  | ~ | # | Operation                           | TL | TH | AH | N | Z       | V | C | OPCODE   |
|-----|---------------------------|---|---|-------------------------------------|----|----|----|---|---------|---|---|----------|
| 1   | BZ/BEQ rel(at branch)     | 4 | 2 | if Z = 1 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | FD       |
|     | BZ/BEQ rel(at no branch)  | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 2   | BNZ/BNE rel(at branch)    | 4 | 2 | if Z = 0 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | FC       |
|     | BNZ/BNE rel(at no branch) | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 3   | BC/BLO rel(at branch)     | 4 | 2 | if C = 1 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | F9       |
|     | BC/BLO rel(at no branch)  | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 4   | BNC/BHS rel(at branch)    | 4 | 2 | if C = 0 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | F8       |
|     | BNC/BHS rel(at no branch) | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 5   | BN rel(at branch)         | 4 | 2 | if N = 1 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | FB       |
|     | BN rel(at no branch)      | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 6   | BP rel(at branch)         | 4 | 2 | if N = 0 then PC ← PC + rel         | -  | -  | -  | - | -       | - | - | FA       |
|     | BP rel(at no branch)      | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 7   | BLT rel(at branch)        | 4 | 2 | if V ∨ N = 1 then PC ← PC + rel     | -  | -  | -  | - | -       | - | - | FF       |
|     | BLT rel(at no branch)     | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 8   | BGE rel(at branch)        | 4 | 2 | if V ∨ N = 0 then PC ← PC + rel     | -  | -  | -  | - | -       | - | - | FE       |
|     | BGE rel(at no branch)     | 2 |   |                                     |    |    |    |   |         |   |   |          |
| 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                    | 3 | 1 | (PC) ← (A)                          | -  | -  | -  | - | -       | - | - | E0       |
| 12  | JMP ext                   | 4 | 3 | (PC) ← ext                          | -  | -  | -  | - | -       | - | - | 21       |
| 13  | CALLV #vct                | 7 | 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                       | 6 | 1 | return from subroutine              | -  | -  | -  | - | -       | - | - | 20       |
| 17  | RETI                      | 8 | 1 | return from interrupt               | -  | -  | -  | - | restore | - | - | 30       |

■ **Other Instructions**

**Table A.4-4 Other Instructions**

| No. | MNEMONIC | ~ | # | Operation                      | TL | TH | AH | N | Z | V | C | OPCODE |
|-----|----------|---|---|--------------------------------|----|----|----|---|---|---|---|--------|
| 1   | PUSHW A  | 4 | 1 | ((SP)) ← (A), (SP) ← (SP) - 2  | -  | -  | -  | - | - | - | - | 40     |
| 2   | POPW A   | 3 | 1 | (A) ← ((SP)), (SP) ← (SP) + 2  | -  | -  | dH | - | - | - | - | 50     |
| 3   | PUSHW IX | 4 | 1 | ((SP)) ← (IX), (SP) ← (SP) - 2 | -  | -  | -  | - | - | - | - | 41     |
| 4   | POPW IX  | 3 | 1 | (IX) ← ((SP)), (SP) ← (SP) + 2 | -  | -  | -  | - | - | - | - | 51     |
| 5   | NOP      | 1 | 1 | No operation                   | -  | -  | -  | - | - | - | - | 00     |
| 6   | CLRC     | 1 | 1 | (C) ← 0                        | -  | -  | -  | - | - | - | R | 81     |
| 7   | SETC     | 1 | 1 | (C) ← 1                        | -  | -  | -  | - | - | - | S | 91     |
| 8   | CLRI     | 1 | 1 | (I) ← 0                        | -  | -  | -  | - | - | - | - | 80     |
| 9   | SETI     | 1 | 1 | (I) ← 1                        | -  | -  | -  | - | - | - | - | 90     |

# MB95850K/860K/870K Series

## A.5 Instruction Map

Table A.5-1 shows the instruction map of F<sup>2</sup>MC-8FX.

### ■ Instruction Map

Table A.5-1 Instruction Map of F<sup>2</sup>MC-8FX

| L | H    | 0    | 1     | 2     | 3    | 4    | 5    | 6    | 7    | 8    | 9    | A    | B   | C    | D    | E     | F    |
|---|------|------|-------|-------|------|------|------|------|------|------|------|------|-----|------|------|-------|------|
| 0 | NOP  | SWAP | RET   | RET   | POPW | MOV  | MOVW | MOVW | MOVW | CLRI | SETI | CLRB | BBC | INCW | DECW | JMP   | MOVW |
| 1 | MULU | DIVU | JMP   | CALL  | POPW | MOV  | MOVW | MOVW | MOVW | CLRC | SETC | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 2 | ROLC | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | MOV  | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 3 | RORC | CMPW | ADDCW | SUBCW | XORW | ANDW | ORW  | ANDW | ORW  | MOVW | MOVW | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 4 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | DAA  | DAS  | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 5 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 6 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 7 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | CLRB | BBC | INCW | DECW | MOVW  | MOVW |
| 8 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BNC  |
| 9 | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BC   |
| A | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BP   |
| B | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BN   |
| C | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BNZ  |
| D | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BZ   |
| E | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BGE  |
| F | MOV  | CMP  | ADDC  | SUBC  | XOR  | AND  | OR   | AND  | OR   | MOV  | CMP  | SETB | BBS | INC  | DEC  | CALLV | BLT  |

# MB95850K/860K/870K Series

MN702-00010-2v0-E

---

**FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL**  
8-BIT MICROCONTROLLER  
New 8FX  
MB95850K/860K/870K Series  
HARDWARE MANUAL

---

February 2013 the second edition

Published **FUJITSU SEMICONDUCTOR LIMITED**

Edited Sales Promotion Department

---

