F²MC-16LX
16-BIT MICROCONTROLLER
MB90520A Series
HARDWARE MANUAL

FUJITSU LIMITED
CONTENTS

■ Objectives and intended reader

Thank you for purchasing Fujitsu semiconductor products.

The MB90520A series was developed as a group of general-purpose models in the F2MC-16LX family, which is a family of original 16-bit single-chip microcontrollers that can be used for application specific ICs (ASICs).

This manual is intended for engineers who design products using the MB90520A series of microcontrollers. The manual describes the functions and operation of the MB90520A series.

■ Trademarks

F2MC is a registered trademark of Fujitsu Limited and stands for FUJITSU Flexible Microcontroller.

■ Organization of this manual

This manual consists of the following 25 chapters and 5 appendices:

CHAPTER 1 OVERVIEW
This chapter describes the features and basic specifications of MB90520A series.

CHAPTER 2 HANDLING DEVICES
This chapter describes the precautions when handling general-purpose one chip microcontroller.

CHAPTER 3 CPU FUNCTION
This chapter explains the CPU function of MB90520A series.

CHAPTER 4 I/O PORT
This chapter describes the function and operation of the I/O port.

CHAPTER 5 TIMEBASE TIMER
This chapter describes the functions and operations of the timebase timer.

CHAPTER 6 WATCHDOG TIMER
This chapter explains the functions and operation of the watchdog timer.

CHAPTER 7 WATCH TIMER
This section describes the functions and operations of the watch timer.

CHAPTER 8 16-BIT I/O TIMER
This chapter explains the functions and operations of 16-bit input/output timer.

CHAPTER 9 16-BIT RELOAD TIMER
This chapter explains the functions and operations of the 16-bit reload timer.

CHAPTER 10 8-/16-BIT PPG TIMER
This chapter explains the functions and operations of the 8-/16-bit PPG timer.

CHAPTER 11 8-/16-BIT UP/DOWN COUNTER/TIMER
This chapter describes 8-/16-bit up/down counter/timer functions and operations.

CHAPTER 12 DELAYED INTERRUPT GENERATION MODULE
This chapter explains the functions and operations of the delayed interrupt generation module.
CHAPTER 13  DTP/EXTERNAL INTERRUPT
This chapter explains the functions and operations of DTP/external interrupt.

CHAPTER 14  WAKE-UP INTERRUPT
This chapter describes the functions and operation of the wake-up Interrupt

CHAPTER 15  8/10-BIT A/D CONVERTER
This chapter explains the functions and operation of 8-/10-bit A/D converter.

CHAPTER 16  D/A CONVERTER
This chapter describes the function and operation of the D/A converter.

CHAPTER 17  COMMUNICATION PRESCALER
Functions and operations of the communication prescaler are explained.

CHAPTER 18  UART
This chapter describes UART function operation.

CHAPTER 19  I/O EXTENDED SERIAL INTERFACE
In this chapter, the function and operation of the I/O extended serial interface are explained.

CHAPTER 20  LCD CONTROLLER DRIVER
In this chapter, the functions and operation of the LCD controller/driver are explained.

CHAPTER 21  ADDRESS MATCH DETECTING FUNCTION
This chapter explains the address match detection function and its operation.

CHAPTER 22  ROM MIRROR FUNCTION SELECTION MODULE
This chapter describes the functions and operations of the ROM mirroring function select module.

CHAPTER 23  CLOCK MONITOR FUNCTION
The clock monitor function and operations are explained in this chapter.

CHAPTER 24  1M-BIT FLASH MEMORY
In this chapter, the function and operation of the 1M-bit flash memory are explained.

CHAPTER 25  CONNECTION EXAMPLE OF FLASH SERIAL PROGRAMMING
Explanation is given for a serial writing connection example when the flash microcomputer programmer made by Yokogawa Digital Computer Corporation is used.

Appendix
APPENDIX A  Instruction
APPENDIX B  Register Index (address list)
APPENDIX C  Register Index (List by Peripheral Function)
APPENDIX D  Pin Function Index
APPENDIX E  Interrupt Vector Index
• The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.

• The information, such as descriptions of function and application circuit examples, in this document are presented solely for the purpose of reference to show examples of operations and uses of FUJITSU semiconductor device; FUJITSU does not warrant proper operation of the device with respect to use based on such information. When you develop equipment incorporating the device based on such information, you must assume any responsibility arising out of such use of the information. FUJITSU assumes no liability for any damages whatsoever arising out of the use of the information.

• Any information in this document, including descriptions of function and schematic diagrams, shall not be construed as license of the use or exercise of any intellectual property right, such as patent right or copyright, or any other right of FUJITSU or any third party or does FUJITSU warrant non-infringement of any third-party's intellectual property right or other right by using such information. FUJITSU assumes no liability for any infringement of the intellectual property rights or other rights of third parties which would result from the use of information contained herein.

• The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).

Please note that FUJITSU will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.

• Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.

• If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.
# CONTENTS

## CHAPTER 1 OVERVIEW

1.1 Feature of MB90520A Series ................................................................. 2  
1.2 Product Lineup of the MB90520A Series ............................................. 5  
1.3 Block Diagram of the MB90520A Series .............................................. 8  
1.4 Pin Assignment .................................................................................... 10  
1.5 Package Dimension ............................................................................ 11  
1.6 Pin Description .................................................................................. 13  
1.7 I/O Circuit ....................................................................................... 18  

## CHAPTER 2 HANDLING DEVICES

2.1 Precautions when Handling Devices .................................................. 24  

## CHAPTER 3 CPU FUNCTION

3.1 Memory Space .................................................................................... 30  
3.1.1 Mapping of and Access to Memory Space ...................................... 32  
3.1.2 Memory Map ................................................................................ 34  
3.1.3 Addressing ................................................................................. 35  
3.1.4 Linear Addressing ....................................................................... 36  
3.1.5 Bank Addressing ......................................................................... 37  
3.1.6 Multi-byte Data Allocation in Memory .......................................... 39  
3.2 Dedicated Registers and General-purpose Registers .......................... 41  
3.2.1 Dedicated Registers .................................................................... 42  
3.2.2 Accumulator (A) .......................................................................... 44  
3.2.3 Stack Pointers (USP, SSP) ............................................................ 47  
3.2.4 Processor Status (PS) .................................................................. 50  
3.2.4.1 Condition Code Register (PS: CCR) ...................................... 51  
3.2.4.2 Register Bank Pointer (PS: RP) ................................................ 53  
3.2.4.3 Interrupt Level Mask Register (ILM) ....................................... 54  
3.2.5 Program Counter (PC) ................................................................. 55  
3.2.6 Direct Page Register (DPR) ........................................................ 56  
3.2.7 Bank Register (PCB, DTB, USB, SSB, and ADB) ....................... 57  
3.3 General-purpose Register ................................................................. 58  
3.4 Prefix Code ...................................................................................... 60  
3.4.1 Bank Select Prefix (PCB,DTB,ADB,SPB) ..................................... 61  
3.4.2 Common Register Bank Prefix (CMR) ......................................... 63  
3.4.3 Flag Change Inhibit Prefix (NCC) .............................................. 64  
3.4.4 Restrictions on Prefix Code .......................................................... 65  
3.4.5 Precautions when Using the "DIV A, Ri" and "DIVW A, RWi" Commands. ................................................. 67  
3.5 Interrupt ...................................................................................... 69  
3.5.1 Interruption Factors and Interruption Vectors .............................. 71  
3.5.2 Interrupt Control Registers and Peripheral Functions .............. 74  
3.5.3 Interrupt Control Register (ICR00 to ICR15) ............................. 76  
3.5.4 Function of Interrupt Control Register ...................................... 78
CHAPTER 4 I/O PORT ............................................................................................................ 173
4.1 Overview of I/O Ports ........................................................................................................ 174
4.2 Assignment of I/O Port Register and Pins Shared with Flash Memory Writing Pins .... 176
4.3 Port 0 ................................................................................................................................ 178
4.3.1 Registers for Port 0 (PDR0, DDR0, RDR0) ................................................................. 180
4.3.2 Operation of Port 0 ...................................................................................................... 182
4.4 Port 1 ................................................................................................................................ 184
4.4.1 Registers for Ports 1 (PDR1, DDR1, RDR1) ............................................................... 186
4.4.2 Operation of Port 1 ...................................................................................................... 188
4.5 Port 2 ................................................................................................................................ 190
4.5.1 Registers for Port 2 (PDR2, DDR2) ............................................................................. 192
4.5.2 Operation of Port 2 ...................................................................................................... 193
4.6 Port 3 ................................................................................................................................ 195
4.6.1 Registers for Port 3 (PDR3, DDR3) ............................................................................. 197
4.6.2 Operation of Port 3 ...................................................................................................... 198
4.7 Port 4 ................................................................................................................................ 200
4.7.1 Registers for Port 4 (PDR4, DDR4, RDR4) ................................................................. 202
4.7.2 Operation of Port 4 ...................................................................................................... 204
4.8 Port 5 ................................................................................................................................ 206
4.8.1 Registers for Port 5 (PDR5, DDR5) ............................................................................. 209
4.8.2 Operation of Port 5 ...................................................................................................... 210
4.9 Port 6 ................................................................................................................................ 212
4.9.1 Registers for Port 6 (PDR6, DDR6, ADER) ................................................................. 214
4.9.2 Operation of Port 6 ...................................................................................................... 216
4.10 Port 7 ............................................................................................................................... 218
4.10.1 Registers for Port 7 (PDR7, DDR7, LCDCMR) ......................................................... 222
4.10.2 Operation of Port 7 ...................................................................................................... 223
4.11 Port 8 ............................................................................................................................... 225
4.11.1 Registers for Port 8 (PDR8, DDR8, LCR1) ................................................................. 227
4.11.2 Operation of Port 8 ...................................................................................................... 228
4.12 Port 9 ............................................................................................................................... 230
4.12.1 Registers for Port 9 (PDR9, DDR9) ........................................................................... 232
4.12.2 Operation of Port 9 ...................................................................................................... 233
4.13 Port A ............................................................................................................................... 235
4.13.1 Registers for Port A (PDRA, DDRA, LCR1) ............................................................ 237
4.13.2 Operation of Port A .................................................................................................... 238

CHAPTER 5 TIMEBASE TIMER .......................................................................................... 241
5.1 Overview of Timebase Timer ............................................................................................ 242
5.2 Block Diagram of Timebase Timer ................................................................................ 244
5.3 Configuration of Timebase Timer .................................................................................. 246
5.3.1 Timebase Timer Control Register (TBTC) ................................................................. 247
5.4 Interrupt of Timebase Timer .......................................................................................... 249
5.5 Explanation of Operations of Timebase Timer .............................................................. 250
5.6 Precautions when Using Timebase Timer ..................................................................... 254
5.7 Program Example of Timebase Timer ............................................................................ 256
<table>
<thead>
<tr>
<th>CHAPTER 6  WATCHDOG TIMER</th>
<th>257</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Overview of Watchdog Timer</td>
<td>258</td>
</tr>
<tr>
<td>6.2 Block Diagram of Watchdog Timer</td>
<td>260</td>
</tr>
<tr>
<td>6.3 Watchdog Timer Control Register (WDTC)</td>
<td>262</td>
</tr>
<tr>
<td>6.4 Explanation of Operations of Watchdog Timer</td>
<td>264</td>
</tr>
<tr>
<td>6.5 Precautions when Using Watchdog Timer</td>
<td>266</td>
</tr>
<tr>
<td>6.6 Program Examples of Watchdog Timer</td>
<td>267</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 7  WATCH TIMER</th>
<th>269</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Overview of Watch Timer</td>
<td>270</td>
</tr>
<tr>
<td>7.2 Block Diagram of Watch Timer</td>
<td>272</td>
</tr>
<tr>
<td>7.3 Configuration of Watch Timer</td>
<td>274</td>
</tr>
<tr>
<td>7.3.1 Watch Timer Control Register (WTC)</td>
<td>275</td>
</tr>
<tr>
<td>7.4 Watch Timer Interrupt</td>
<td>277</td>
</tr>
<tr>
<td>7.5 Explanation of Operation of Watch Timer</td>
<td>278</td>
</tr>
<tr>
<td>7.6 Program Example of Watch Timer</td>
<td>280</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 8  16-BIT I/O TIMER</th>
<th>281</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Overview of 16-bit Input/Output Timer</td>
<td>282</td>
</tr>
<tr>
<td>8.2 Block Diagram of 16-bit Input/Output Timer</td>
<td>284</td>
</tr>
<tr>
<td>8.2.1 Block Diagram of 16-bit Free-run Timer</td>
<td>286</td>
</tr>
<tr>
<td>8.2.2 Block Diagram of Input Capture</td>
<td>287</td>
</tr>
<tr>
<td>8.2.3 Block Diagram of Output compare</td>
<td>288</td>
</tr>
<tr>
<td>8.3 Configuration of 16-bit Input/Output Timer</td>
<td>290</td>
</tr>
<tr>
<td>8.3.1 Timer Control Status Register (TCCS)</td>
<td>292</td>
</tr>
<tr>
<td>8.3.2 Timer Data Register (TCDT)</td>
<td>294</td>
</tr>
<tr>
<td>8.3.3 Input Capture Control Status Registers (ICS01)</td>
<td>295</td>
</tr>
<tr>
<td>8.3.4 Input Capture Register (IPCP)</td>
<td>297</td>
</tr>
<tr>
<td>8.3.5 High-order Output Compare Control Status Register (OCS)</td>
<td>298</td>
</tr>
<tr>
<td>8.3.6 Low-order Output Compare Control Status Register (OCS)</td>
<td>301</td>
</tr>
<tr>
<td>8.3.7 Output Compare Register</td>
<td>303</td>
</tr>
<tr>
<td>8.4 Interrupts of 16-bit Input/Output Timer</td>
<td>304</td>
</tr>
<tr>
<td>8.5 Explanation of Operation of 16-bit Free-run Timer</td>
<td>306</td>
</tr>
<tr>
<td>8.6 Explanation of Operation of Input Capture</td>
<td>309</td>
</tr>
<tr>
<td>8.7 Explanations of Operation of Output Compare</td>
<td>311</td>
</tr>
<tr>
<td>8.8 Precaution on Using 16-bit I/O Timer</td>
<td>314</td>
</tr>
<tr>
<td>8.9 Program Example of 16-bit Input/Output Timer</td>
<td>315</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 9  16-BIT RELOAD TIMER</th>
<th>319</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1 Overview of 16-bit Reload Timer</td>
<td>320</td>
</tr>
<tr>
<td>9.2 Block Diagram of 16-bit Reload Timer</td>
<td>322</td>
</tr>
<tr>
<td>9.3 Configuration of 16-bit Reload Timer</td>
<td>324</td>
</tr>
<tr>
<td>9.3.1 Timer Control Status Registers (High) (TMCSR0: H, TMCSR1: H)</td>
<td>325</td>
</tr>
<tr>
<td>9.3.2 Timer Control Status Registers (Low) (TMCSR0: L, TMCSR1: L)</td>
<td>327</td>
</tr>
<tr>
<td>9.3.3 16-bit Timer Registers (TMRO, TMRL)</td>
<td>329</td>
</tr>
<tr>
<td>9.3.4 16-bit Reload Registers (TMRLR0, TMRLR1)</td>
<td>330</td>
</tr>
<tr>
<td>9.4 Interrupts of 16-bit Reload Timer</td>
<td>331</td>
</tr>
</tbody>
</table>
CHAPTER 12  DELAYED INTERRUPT GENERATION MODULE ................................. 423
  12.1 Overview of Delayed Interrupt Generation Module ........................................ 424
  12.2 Block Diagram of Delayed Interrupt Generation Module ................................. 425
  12.3 Delayed Interrupt Request Generate/Cancel Register (DIRR) ......................... 426
  12.4 Explanation of Operation of Delayed Interrupt Generation Module .............. 427
  12.5 Precautions when Using Delayed Interrupt Generation Module ................... 428
  12.6 Program Example of Delayed Interrupt Generation Module .......................... 429

CHAPTER 13  DTP/EXTERNAL INTERRUPT ......................................................... 431
  13.1 Overview of DTP/External Interrupt ............................................................... 432
  13.2 Block Diagram of DTP/External Interrupt ..................................................... 433
  13.3 Configuration of DTP/External Interrupt ........................................................ 435
    13.3.1 DTP/External Interrupt Factor Register (EIRR) ........................................ 436
    13.3.2 DTP/External Interrupt Enable Register (ENIR) ....................................... 437
    13.3.3 Detection Level Setting Register (ELVR) (High) ....................................... 438
    13.3.4 Detection Level Setting Register (ELVR) (Low) ....................................... 439
  13.4 Explanation of Operation of DTP/External Interrupt ..................................... 440
    13.4.1 External Interrupt Function ................................................................. 443
    13.4.2 DTP Function ...................................................................................... 444
  13.5 Precautions when Using DTP/External Interrupt ......................................... 445
  13.6 Program Example of DTP/External Interrupt Circuit .................................... 447

CHAPTER 14  WAKE-UP INTERRUPT ................................................................. 451
  14.1 Wake-up Interrupt Outline .............................................................................. 452
  14.2 Block Diagram of Wake-up Interrupt ............................................................. 453
  14.3 Wake-up Interrupt Configuration ................................................................. 455
    14.3.1 Wake-up Interrupt Flag Register (EIFR) ................................................ 456
    14.3.2 Wake-up Interrupt Control Register (EICR) ........................................... 457
  14.4 Explanations for Wake-up Interrupt Operation .............................................. 459
  14.5 Precautions when Using Wake-up Interrupt .................................................. 461
  14.6 Program Examples for Wake-up Interrupt .................................................... 462

CHAPTER 15  8/10-BIT A/D CONVERTER ...................................................... 463
  15.1 Overview of 8-/10-bit A/D Converter ............................................................. 464
  15.2 Block Diagram of 8-/10-bit A/D Converter .................................................... 465
  15.3 Configuration of 8-/10-bit A/D Converter ...................................................... 468
    15.3.1 A/D Control Status Register (High) (ADCS: H) ..................................... 469
    15.3.2 A/D Control Status Register (Low) (ADCS: L) ....................................... 472
    15.3.3 A/D Data Register (High) (ADCR: H) .................................................. 475
    15.3.4 A/D Data Register (Low) (ADCR: L) ..................................................... 477
    15.3.5 Analog Input Enable Register (ADER) .................................................. 478
  15.4 Interrupt of 8-/10-bit A/D Converter ............................................................. 480
  15.5 Explanation of Operation of 8-/10-bit A/D Converter ..................................... 481
    15.5.1 Single-shot Conversion Mode .............................................................. 482
    15.5.2 Continuous Conversion Mode .............................................................. 484
CHAPTER 1

OVERVIEW

This chapter describes the features and basic specifications of MB90520A series.

1.1 Feature of MB90520A Series
1.2 Product Lineup of the MB90520A Series
1.3 Block Diagram of the MB90520A Series
1.4 Pin Assignment
1.5 Package Dimension
1.6 Pin Description
1.7 I/O Circuit
1.1 Feature of MB90520A Series

MB90520A series is a general-purpose, high-performance 16-bit microcontroller designed for control of processors such as consumer products requiring high-speed real-time processing.

The command system continues the F²MC family architecture while adding C language commands, extending the addressing mode and providing better multiplying commands and a full set of bit processing commands. A 32-bit accumulator enables long-word data (32 bits) processing.

Feature of MB90520A series

- **Clock**
  - Built-in PLL clock multiplying circuit
  - The oscillation clock is divided by 2 or multiplied by 1 to 4 (4 to 16 MHz for an oscillation clock frequency of 4 MHz) when used for the machine clock (PLL clock).
  - Sub-clock operation (32.768 kHz) is divided by 4 (8.192 kHz)
  - Minimum instruction execution time: 62.5 ns (at a machine clock frequency of 16 MHz)

- **16-MB CPU memory space**
  - Internal 24-bit addressing

- **Instruction system optimized for controllers**
  - Rich data types (bit, byte, word, long word)
  - 23 types of addressing modes
  - Enhanced signed instructions of multiplication/division and RETI instruction function
  - Improvements in 32-bit accumulator operation accuracy

- **Command system supporting C language/multitasking**
  - System stack pointer
  - Enhanced pointer indirect instructions
  - Barrel shift instructions

- **Improved execution speed**
  - 4-byte instruction queue

- **Interrupt function**
  - 8-level, 34 factor interrupt function

- **CPU-independent automatic data transfer function**
  - Extended intelligent I/O service (EI²OS) function: Maximum 16 channels
CHAPTER 1  OVERVIEW

● Low-power consumption (standby) mode
  - Sleep mode (stops CPU clock)
  - Timebase timer mode (operates only oscillation clock and sub-clock, timebase timer and watch timer)
  - Watch mode (operates only sub-clock and watch timer)
  - Stop mode (stops oscillation clock and sub-clock)
  - CPU Intermittent operation mode
  - Hardware Standby Mode

● I/O port
  - General-purpose I/O ports (CMOS output): 53 ports
  - General-purpose I/O ports (pull-up resistor input): 24 ports
  - General-purpose I/O ports (N-ch open drain output): 8 ports

● Timers
  - Timebase timer, watch timer, watchdog timer: one each per channel
  - 8-/16-bit PPG timer 0, 1:8-bit × 2 channels, or 16-bit × 1 channel
  - 16-bit reload timer 0, 1: 2 channels
  - 16-bit I/O timer:
    - 16-bit free-run timer 0, 1: 2 channels
    - 16-bit input capture 0:2 channels (1 unit, 2 channels)
    - 16-bit output compare 0, 1:8 channels (1 unit, 4 channels)
  - 8-/16-bit up/down counter/timer 0, 1:8-bit × 2 channels, or 16-bit × 1 channel
  - Clock output function: 1 channel

● Communications macro (communication interface)
  - Extended I/O serial interface 0,1:2 channels
  - UART (with full duplex double buffer and SCI: synchronous serial); 1 channel

● External event interrupt control function
  - DTP/external interrupt: 8 channels (rising/falling edge, H level/L level can be received)
  - Wake-up interrupt: 8 channels (only L level accepted)
  - Delayed interrupt generation module: 1 channel (switching task)

● Analog-digital conversion function
  - 8-/10-bit A/D converter: 8 channels (External trigger activation enabled. When minimum change time is 10.2 us; machine clock frequency operates at 16 MHz)
  - 8-bit D/A converter: 2 channel (R-2R system, centring time 12.5 us: machine clock frequency 16 MHz operation)

● Display functions
  - LCD controller driver: 32 segment drivers, 4 common drivers
● Address match detection function
  • Detects address match for two address pointers

● Others
  • Serial write operations to flash memory is possible (only products with flash memory).

● Process
  • CMOS Technology

---

**Note:** The MB90520A series cannot be used under external bus mode. Set the mode pin to the internal vector mode (MD2=0, MD1=1, MD0=1) and set mode data to single-chip mode (M1=0, M0=0). Refer to "3.9.1 Mode Pin" for mode pin details, and refer to "3.9.2 Mode Data" for the mode data details.
1.2 **Product Lineup of the MB90520A Series**

Table 1.2-1 shows product lineup and Table 1.2-3 shows CPU and peripheral functions.

### Product lineup of the MB90520A series

**Table 1.2-1 Product lineup of the MB90520A series**

<table>
<thead>
<tr>
<th></th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Mask ROM</td>
<td>Flash ROM</td>
<td>Evaluation product</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROM Size</td>
<td>64 KB</td>
<td></td>
<td>128 KB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM Size</td>
<td>4KB</td>
<td></td>
<td></td>
<td>6 KB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step-down Circuit</td>
<td>Not provided</td>
<td>Used bit</td>
<td>Not provided</td>
<td>Used bit</td>
<td>Used bit</td>
<td>Used bit</td>
</tr>
<tr>
<td>Power supply for emulator*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Not provided</td>
</tr>
<tr>
<td>Process</td>
<td>CMOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating power supply voltage</td>
<td>3.0V to 5.5V</td>
<td>2.7V to 5.5V</td>
<td>3.0V to 5.5V</td>
<td>2.7V to 5.5V</td>
<td>3.0V to 5.5V</td>
<td></td>
</tr>
</tbody>
</table>

*: Setting of DIP Switch (S2) when using emulation pod (MB2145-507). Refer to the MB2145-507 hardware manual (2.7 Dedicated power pins for emulators) for details.

### MB90520A series package and corresponding products

**Table 1.2-2 MB90520A series package and corresponding products**

<table>
<thead>
<tr>
<th></th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90F523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-120P-M05 (LQFP)</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>-</td>
</tr>
<tr>
<td>FPT-120P-M13 (QFP)</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>-</td>
</tr>
<tr>
<td>PGA-256C-A01 (PGA)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>○</td>
</tr>
</tbody>
</table>
## CPU and peripheral functions of MB90520A series

### Table 1.2-3 CPU and peripheral functions of MB90520A series (1/2)

<table>
<thead>
<tr>
<th>CPU function</th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of basic instructions</td>
<td>351</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction bit length</td>
<td>8-bit, 16-bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction length</td>
<td>1 to 7 bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data bit length</td>
<td>1, 8 and 16 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum instruction execution time: 62.5 ns (at a machine clock frequency of 16 MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt operating time: Min 1.5 μs (at 16-MHz machine clock frequency)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low-power consumption (Standby) Mode</td>
<td>Sleep mode/watch mode/timebase timer mode/stop mode/hardware standby mode/CPU intermittent operation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O port</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (CMOS output)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (pull-up resistor input)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (N-ch open drain output)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>85 ports</td>
</tr>
<tr>
<td>Minimum instruction execution time: 62.5 ns (at a machine clock frequency of 16 MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt operating time: Min 1.5 μs (at 16-MHz machine clock frequency)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timebase timer</td>
<td>18-bit counter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle: 1.024 ms, 4.096 ms, 16.384 ms, 131.072 ms (Assuming an oscillation clock frequency of 4 MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>18-bit counter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle: 3.58ms, 14.33ms, 57.23ms, 458.75ms (the minimum value of oscillation clock frequency 4MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset generation cycle: 0.438s, 3.500s, 7.000s, 14.00s (For sub-clock frequency 8.192 kHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-bit I/O timer</td>
<td>16-bit free-run timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count: 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt by overflow generation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output compare</td>
<td>16-bit free-run timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count: 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pin switching factor: Matching between free-run timer register and output compare register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input capture</td>
<td>16-bit free-run timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count: 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retention of the free-run timer register value through pin input (rising edge, falling edge, and both edges)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-bit reload timer</td>
<td>16-bit reload timer operation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count: 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-bit reload timer operation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Count clock cycle: 0.125μs, 0.5μs, 2.0μs (Assuming a machine clock frequency of 16 MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External event countable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Watch timer</td>
<td>15-bit counter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle: 62.5ms, 125ms, 250ms, 500ms, 1.0s, 2.0s, 4.0s (For sub-clock frequency 8.192 kHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-/16-bit PPG timer</td>
<td>8-/16-bit PPG timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count: 1 (8-bit × 2 channels can also be used.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPG operable with 8 bits × 2 channels or 16 bits × 1 channel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse waveform output at arbitrary cycle and duty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Count clock: 62.5ns to 1μs (Assuming a machine clock frequency of 16 MHz)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 1.2-3 CPU and peripheral functions of MB90520A series (2 / 2)

<table>
<thead>
<tr>
<th>Feature</th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-/16-bit up/down counter/timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>1 (operable with 8 bits × 2 channels)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External event input</td>
<td>6 channels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reload/compare function</td>
<td>8-bit × 2 channels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delayed interrupt generation module</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt generation module for switching task</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Used for real-time OS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DTP/external interrupt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input count</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Started by inputting rising edge, falling edge, &quot;H&quot; level and &quot;L&quot; level.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Available for use external interrupt of extended intelligent I/O service (EI²OS)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wake-up interrupt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input count</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initiated by the &quot;L&quot; level input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8/10-bit A/D converter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>10 or 8 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A number of continuous channels can be converted sequentially.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Up to 8 channels can be set)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single conversion mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selected channel converted once only</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous conversion mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selected channel converted continuously</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop conversion mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selected channel converted and temporary stopped alternately</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D/A converter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>8 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R-2R system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UART (SCI)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock synchronous transfer</td>
<td>62.6 Kbps to 1 Mbps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock asynchronous transfer</td>
<td>1.202 bps to 9,615 kbps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Two-way serial communication function, master/slave-connected communication</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O Extended serial interface</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel count</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock synchronous transfer (When internal shift clock is selected, 31.25 Kbps to 1 Mbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSB first and MSB first can be switched</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCD controller/driver</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Common output</td>
<td>: 4 lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segment output</td>
<td>: 32 lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power supply for driving the LCD</td>
<td>: 4 lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data memory for displaying the LCD</td>
<td>: 16 bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Divided resistance for driving the LCD</td>
<td>: built-in</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1.3 Block Diagram of the MB90520A Series

Figure 1.3-1 shows block diagram of all MB90520A series.

Block diagram of all MB90520A series

---

**Figure 1.3-1 Block diagram of all MB90520A series**

- CPU
- F²MC-16LX core
- Interrupt controller
- Delayed interruption
- Port 6
  - AN0 to AN7
  - ADT0
  - AVcc, AVss
  - AVR0, AVR1
- Port 7
  - P70 to P77
  - TIN0, TIN1
  - TOT0, TOT1
  - OUT4 to OUT7
- Port 8, A
  - LCD controller driver
    - (32 segment, 4 common)
  - P80 to P87, PA0 to PA7
  - V0 to V3
  - SEG16 to SEG23
  - SEG8 to SEG15
  - SEG0 to SEG7
  - COM0 to COM3
  - SEG24 to SEG31
- Port 9
- Port 5
  - P50 to P54
  - AIN1, BIN1, ZIN1
  - SOT2
  - SIN2, SCK2
  - DA0, DA1
  - DVcc, DVss
- Port 4
  - Port 3
    - 8/16-bit PPG timer
      - UART
      - PPG00 to PPG01
      - PPG10 to PPG11
      - SIN0, SCK0
      - SOT0
      - SIN1, SCK1
      - SOT1
      - P40 to P47
- Port 2
  - Port 1
    - Port 0
    - DTP/external interruption
      - P00 to P07
    - INT0 to INT6
    - INT7
      - W10 to W17
    - P20 to P27
    - AIN0, BIN0, ZIN0
    - IN00, IN01, IN10, IN11
    - OUT0 to OUT3
    - CKOT
    - P30 to P37
  - Port 2
    - Port 1
      - Port 0
      - DTP/external interruption
      - P00 to P07
    - INT0 to INT6
      - INT7
        - W10 to W17
      - P20 to P27
      - AIN0, BIN0, ZIN0
      - IN00, IN01, IN10, IN11
      - OUT0 to OUT3
      - CKOT
      - P30 to P37
    - PPG00 to PPG01
      - PPG10 to PPG11
    - SIN0, SCK0
      - SOT0
      - SIN1, SCK1
      - SOT1
      - P40 to P47
- Port 3
  - Port 4
    - Port 3
      - Port 4
      - Port 5
        - Port 6
          - Port 7
            - Port 8, A
              - Port 9
                - Port 0
                  - Port 1
                    - Port 2
                      - Port 3
                        - Port 4
                          - Port 5
                            - Port 6
                              - Port 7
                                - Port 8, A
                                  - Port 9
The clock control circuit is comprised of a watchdog timer, timebase timer, watch timer and low-power consumption control circuit.

Port 0 (P00 to P07), Port 1 (P10 to P17) and Port 4 (P40 to P47) can set up the connection or disconnection of built-in pull-up resistor using the input resistance resistor (RDR).

Port 9 (P90 to P97), when used as the general-purpose input/output port, functions as an N-ch open drain pin. Port 9 (P90 to P97) can also be used as a large current port for driving the LED (I_{OL} = 10mA).
1.4 Pin Assignment

Figure 1.4-1 are pin assignment of the MB90520A series.

**Pin Assignment**

Figure 1.4-1 Pin Assignment

*: As MB90522A, MB90523A: N.C.
1.5 Package Dimension

MB90520A series is available in two types of package. The attached package dimension is for reference only. Contact Fujitsu for the nominal package dimensions.

### Package dimensions of FPT-120P-M05

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead pitch</td>
<td>0.40 mm</td>
</tr>
<tr>
<td>Package width x package length</td>
<td>14.0 x 14.0 mm</td>
</tr>
<tr>
<td>Lead shape</td>
<td>Gullwing</td>
</tr>
<tr>
<td>Sealing method</td>
<td>Plastic mold</td>
</tr>
<tr>
<td>Mounting height</td>
<td>1.70 mm MAX</td>
</tr>
<tr>
<td>Weight</td>
<td>0.62 g</td>
</tr>
<tr>
<td>Code</td>
<td>P-LFQFP120-14x14-0.40</td>
</tr>
</tbody>
</table>

![Diagram of package dimensions](image)
## Package dimensions of FPT-120P-M13

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead pitch</td>
<td>0.50 mm</td>
</tr>
<tr>
<td>Package width ( \times ) package length</td>
<td>20.0 ( \times ) 20.0 mm</td>
</tr>
<tr>
<td>Lead shape</td>
<td>Gullwing</td>
</tr>
<tr>
<td>Sealing method</td>
<td>Plastic mold</td>
</tr>
<tr>
<td>Mounting height</td>
<td>3.85 mm MAX</td>
</tr>
<tr>
<td>Weight</td>
<td>2.58g</td>
</tr>
<tr>
<td>Code (Reference)</td>
<td>P-FQFP120-20( \times )20-0.50</td>
</tr>
</tbody>
</table>

### 120-pin plastic QFP

Dimensions in mm (inches).

- Note 1): These dimensions do not include resin protrusion.
- Note 2): Pins width and pins thickness include plating thickness.
- Note 3): Pins width do not include tie bar cutting remainder.
1.6 Pin Description

Table 1.6-1 shows pin description and function of each pin.

## Pin Description

Table 1.6-1 Pin Description (1 / 5)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Functional description</th>
</tr>
</thead>
<tbody>
<tr>
<td>93/92</td>
<td>X0/X1</td>
<td>A</td>
<td>High-speed oscillation pin (The oscillation clock is input.)</td>
</tr>
<tr>
<td>74/73</td>
<td>X0A/X1A</td>
<td>B</td>
<td>Low-speed oscillation pin (Sub oscillation clock is input.)</td>
</tr>
<tr>
<td>90</td>
<td>RST</td>
<td>C</td>
<td>Input pin for external reset</td>
</tr>
<tr>
<td>86</td>
<td>HST</td>
<td>C</td>
<td>This is the hardware standby input pin.</td>
</tr>
<tr>
<td>95 to 101</td>
<td>P00 to P06</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR0). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td>95 to 101</td>
<td>INT0 to INT6</td>
<td>D</td>
<td>This functions as the external interrupt input pin. This pin should be set to input port.</td>
</tr>
<tr>
<td>102</td>
<td>P07</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR0). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td>103 to 110</td>
<td>P10 to P17</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR1). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td>103 to 110</td>
<td>WI0 to WI7</td>
<td>D</td>
<td>This functions as the input pin for wake-up interruption. This pin should be set to input port.</td>
</tr>
<tr>
<td>111 to 114</td>
<td>P20 to P23</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td>111 to 114</td>
<td>IN00/IN01, IN10/IN11</td>
<td>E</td>
<td>This functions as trigger input pins for input capture channels 0 and 1. When this is used as the trigger input pin of the input capture channels 0 and 1, set it to the input port for use.</td>
</tr>
<tr>
<td>115</td>
<td>P24</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td>115</td>
<td>AIN0</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 0 count clock input (B) pin. When this is used as the 8-/16-bit up/down counter channel 0 count clock input (B) pin, set it to the input port for use.</td>
</tr>
<tr>
<td>116</td>
<td>P25</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td>116</td>
<td>BIN0</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 0 count clock input (B) pin. When this is used as the 8-/16-bit up/down counter channel 0 count clock input (B) pin, set it to the input port for use.</td>
</tr>
<tr>
<td>Pin No.</td>
<td>Pin Name</td>
<td>Circuit Type</td>
<td>Functional description</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>--------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>117</td>
<td>P26</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>ZIN0</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 0 count control input (Z) pin. When this is used as the 8-/16-bit up/down counter channel 0 count control input (Z) pin, set it to the input port for use.</td>
</tr>
<tr>
<td></td>
<td>INT7</td>
<td>E</td>
<td>This functions as the external interrupt input pin. This pin should be set to input port.</td>
</tr>
<tr>
<td>118</td>
<td>P27</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>ADTG</td>
<td>E</td>
<td>External trigger input pin for A/D converter. This pin should be set to input port.</td>
</tr>
<tr>
<td>120</td>
<td>P30</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td>1</td>
<td>P31</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>CKOT</td>
<td>E</td>
<td>This functions as the output pin for the clock monitor function. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>2 to 5</td>
<td>P32 to P35</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>OUT0 to OUT3</td>
<td>E</td>
<td>This functions as the event output pin for output compare channels 0 to 3. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>6/7</td>
<td>P36/P37</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>PPG00/PPG01</td>
<td>E</td>
<td>Output pins for PPG timer 0. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>9/10</td>
<td>P40/P41</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td></td>
<td>PPG10/PPG11</td>
<td>D</td>
<td>Output pins for PPG timer 1. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>11</td>
<td>P42</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td></td>
<td>SIN0</td>
<td>D</td>
<td>Serial data input pin of UART. This pin should be set to input port.</td>
</tr>
<tr>
<td>12</td>
<td>P43</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td></td>
<td>SOT0</td>
<td>D</td>
<td>Serial data output pin of UART. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>13</td>
<td>P44</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid.</td>
</tr>
<tr>
<td></td>
<td>SCK0</td>
<td>D</td>
<td>Serial clock output input of UART. When output set up for the UART serial clock output function is enabled, the clock output will be valid.</td>
</tr>
</tbody>
</table>
Table 1.6-1 Pin Description (3 / 5)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Functional description</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>P45</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid. This functions as the serial data input pin of the extended serial I/O channel 1. When this is used as the serial data input pin of the extended serial I/O channel 1, set it to the input port for use.</td>
</tr>
<tr>
<td>15</td>
<td>P46</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid. This functions as the serial data output pin of the extended serial I/O channel 1. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>16</td>
<td>P47</td>
<td>D</td>
<td>General-purpose I/O port. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid. This functions as the serial clock input/output pin of the extended serial I/O channel 1. Clock output is valid when the output setting of the extended serial I/O channel 1 serial clock output is enabled.</td>
</tr>
<tr>
<td>17 to 24</td>
<td>SEG0 to SEG7</td>
<td>F</td>
<td>A segment output pin of the LCD controller driver.</td>
</tr>
<tr>
<td>25 to 32</td>
<td>PA0 to PA7, SEG8 to SEG15</td>
<td>L</td>
<td>General-purpose I/O port A segment output pin of the LCD controller/driver. The general-purpose ports and segment pins are switched in units of eight at a time. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>34</td>
<td>C*</td>
<td>G</td>
<td>Capacity pin for stabilizing power supply. This pin should be connected to a ceramic capacitor of approx.0.1 µF.</td>
</tr>
<tr>
<td>35</td>
<td>P50</td>
<td>E</td>
<td>General-purpose I/O port This functions as the serial data input pin of the extended serial I/O channel 2. When this is used as the serial data input pin of the extended serial I/O channel 2, set it to the input port for use.</td>
</tr>
<tr>
<td>35</td>
<td>SIN2</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 1 count clock input (A) pin. When this is used as the 8-/16-bit up/down counter channel 1 count clock input (A) pin, set it to the input port for use.</td>
</tr>
<tr>
<td>36</td>
<td>P51</td>
<td>E</td>
<td>General-purpose I/O port This functions as the serial data output pin of the extended serial I/O channel 2. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>36</td>
<td>SOT2</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 1 count clock input (B) pin. When this is used as the 8-/16-bit up/down counter channel 1 count clock input (B) pin, set it to the input port for use.</td>
</tr>
</tbody>
</table>

Note: P45, P46, and P47 are general-purpose I/O ports. When input port is set, pull-up resistance can be connected by the pull-up resistance setting register (RDR4). When output port is set, pull-up resistance connection will be invalid. The table lists various ports and their functions, including serial data input and output, general-purpose I/O ports, and segment output pins for the LCD controller driver.
<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Functional description</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>SCK2</td>
<td>E</td>
<td>This functions as the serial clock input/output pin of the extended serial I/O channel 2. Clock output is valid when the output setting of the extended serial I/O channel 2 serial clock output is enabled.</td>
</tr>
<tr>
<td></td>
<td>ZIN1</td>
<td>E</td>
<td>This functions as the 8-/16-bit up/down counter channel 1 count control input (Z) pin. When this is used as the 8-/16-bit up/down counter channel 1 count control input (Z) pin, set it to the input port for use.</td>
</tr>
<tr>
<td>38</td>
<td>DV_{CC}</td>
<td>H</td>
<td>This is the D/A converter reference power supply (Vref +) input pin. Enter a voltage that does not exceed Vcc.</td>
</tr>
<tr>
<td>39</td>
<td>DV_{SS}</td>
<td>H</td>
<td>GND input pin for D/A converter. Enter Vss</td>
</tr>
<tr>
<td>40/41</td>
<td>P53/P54</td>
<td>I</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>DA0/DA1</td>
<td>I</td>
<td>It functions as a D/A converter analog output pin. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>42</td>
<td>AV_{CC}</td>
<td>H</td>
<td>V_{CC} power input pin for A/D converter</td>
</tr>
<tr>
<td>43</td>
<td>AV_{RH}</td>
<td>J</td>
<td>This is the D/A converter reference power supply (Vref +) input pin. Enter a voltage that does not exceed Vcc.</td>
</tr>
<tr>
<td>44</td>
<td>AV_{RL}</td>
<td>H</td>
<td>Power (Vref -) input pin for A/D converter. Do not enter a voltage of Vss or lower.</td>
</tr>
<tr>
<td>45</td>
<td>AV_{SS}</td>
<td>H</td>
<td>V_{SS} power input pin for A/D converter</td>
</tr>
<tr>
<td>46 to 53</td>
<td>P60 to P67</td>
<td>K</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>AN0 to AN7</td>
<td>K</td>
<td>It functions as a D/A converter analog output pin. Becomes valid when analog input setting register is set to enabled.</td>
</tr>
<tr>
<td>55</td>
<td>P70</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>TIN0</td>
<td>E</td>
<td>Event input pin for reload timer 0. This pin should be set to input port.</td>
</tr>
<tr>
<td></td>
<td>OUT4</td>
<td>E</td>
<td>This functions as the event output pin for output compare channel 4. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>56</td>
<td>P71</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>TOT0</td>
<td>E</td>
<td>Event output pin for reload timer 0. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td></td>
<td>OUT5</td>
<td>E</td>
<td>This functions as the event output pin for output compare channel 5. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>57</td>
<td>P72</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>TIN1</td>
<td>E</td>
<td>Event input pin for reload timer 1. This pin should be set to input port.</td>
</tr>
<tr>
<td></td>
<td>OUT6</td>
<td>E</td>
<td>This functions as the event output pin for output compare channel 6. This pin is enabled when the output setting is enabled.</td>
</tr>
</tbody>
</table>
### Table 1.6-1 Pin Description (5/5)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Functional description</th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td>P73</td>
<td>E</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>TOT1</td>
<td></td>
<td>Event output pin for reload timer 1. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td></td>
<td>OUT7</td>
<td></td>
<td>This functions as the event output pin for output compare channel 7. This pin is enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>59 to 62</td>
<td>P74 to P77</td>
<td>L</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>COM0 to COM3</td>
<td></td>
<td>A common output pin of the LCD controller/driver.</td>
</tr>
<tr>
<td>64 to 71</td>
<td>P80 to P87</td>
<td>L</td>
<td>General-purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>SEG16 to SEG23</td>
<td></td>
<td>A segment output pin of the LCD controller/driver. The general-purpose ports and segment pins are switched in units of eight at a time. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>72, 75 to 81</td>
<td>P90 to P97</td>
<td>M</td>
<td>General-purpose I/O port (Suitable for up to IOL=10 mA)</td>
</tr>
<tr>
<td></td>
<td>SEG24 to SEG31</td>
<td></td>
<td>A segment output pin of the LCD controller/driver. The general-purpose ports and segment pins are switched in units of eight at a time. These pins are enabled when the output setting is enabled.</td>
</tr>
<tr>
<td>82 to 85</td>
<td>V0 to V3</td>
<td>N</td>
<td>A reference power supply pin for an LCD controller/driver</td>
</tr>
<tr>
<td>87 to 89</td>
<td>MD2, MD1, MD0</td>
<td>C</td>
<td>This is an input pin for setting the operating mode.</td>
</tr>
<tr>
<td>8, 54, 94</td>
<td>Vcc</td>
<td></td>
<td>Power (5 V) input pin.</td>
</tr>
<tr>
<td>33, 63, 91, 119</td>
<td>Vss</td>
<td></td>
<td>Power (0 V) input pin.</td>
</tr>
</tbody>
</table>

*:* N.C. pin on MB90522A and MB90523A

**Note:**

The reload timer output takes priority when the output compare outputs (OUT5, 7) and reload timer outputs (TOT0, 1). For P71, 73 are both enabled to output. Disable the reload timer output to use the output compare outputs (OUT5, 7).
## 1.7 I/O Circuit

Table 1.7-1 shows I/O circuit.

**I/O Circuit**

<table>
<thead>
<tr>
<th>Classification</th>
<th>Circuit</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td><img src="image" alt="Circuit Diagram A" /></td>
<td>• Approximately 1 MΩ high speed oscillation feedback resistor.</td>
</tr>
<tr>
<td>B</td>
<td><img src="image" alt="Circuit Diagram B" /></td>
<td>• Approximately 10 MΩ low speed oscillation feedback resistor.</td>
</tr>
<tr>
<td>C</td>
<td><img src="image" alt="Circuit Diagram C" /></td>
<td>• Hysteresis input</td>
</tr>
</tbody>
</table>
| D              | ![Circuit Diagram D](image) | • Pull-up options can be selected  
• CMOS hysteresis input  
• CMOS level output  
• Standby control provided |
<table>
<thead>
<tr>
<th>Classification</th>
<th>Circuit</th>
<th>Remark</th>
</tr>
</thead>
</table>
| **E** | ![Circuit Diagram](image) | • CMOS hysteresis input  
• CMOS level output  
• Standby control provided |
| **F** | ![Circuit Diagram](image) | • Segment output |
| **G** | ![Circuit Diagram](image) | • Capacitor connection pin (N.C. pin on MB90522A and MB90523A) |
| **H** | ![Circuit Diagram](image) | • Analog power supply input protection circuit |
### Table 1.7-1 I/O Circuit (3 / 4)

<table>
<thead>
<tr>
<th>Classification</th>
<th>Circuit</th>
<th>Remark</th>
</tr>
</thead>
</table>
| I | ![I Circuit Diagram](image) | - CMOS hysteresis input  
- CMOS level output (CMOS is not output during analog output)  
- Combined with an analog output (analog output has priority)  
- Standby control provided |
| J | ![J Circuit Diagram](image) | - A/D converter ref + power supply input pin (with power supply protection circuit) |
| K | ![K Circuit Diagram](image) | - CMOS hysteresis input  
- CMOS-level output  
- Also used as analog input pin  
- Standby control provided |
| L | ![L Circuit Diagram](image) | - CMOS hysteresis input  
- CMOS-level output  
- Combined with segment output pin  
- Standby control (valid for output other than segment output) |
### Table 1.7-1 I/O Circuit (4 / 4)

<table>
<thead>
<tr>
<th>Classification</th>
<th>Circuit</th>
<th>Remark</th>
</tr>
</thead>
</table>
| **M**          | ![Circuit Diagram](image) | • CMOS hysteresis input  
• N-ch open drain output  
• Combined with segment output pin  
• Standby control (valid for output other than segment output) |
| **N**          | ![Circuit Diagram](image) | • LCD controller reference power supply pin |
This chapter describes the precautions when handling general-purpose one chip micro-controller.

2.1 Precautions when Handling Devices
2.1 Precautions when Handling Devices

Care must be taken concerning the following when handling devices.
- Observing the maximum rated voltage (prevention of latch-up)
- Stabilization of supply voltage
- At power on
- Power supply pin
- Crystal oscillation circuit
- When external clock is used
- Note on during operation of PLL clock mode
- When sub-clock mode is not used
- Treatment of unused pins
- Handling N.C. pin
- When output from ports 0 & 1 is insufficient
- Pin processing when the A/D converter is not used.
- Power pin of the A/D converter, power voltage application to the analog input pin, and disconnection order
- Using both general-purpose input/output port and SEG/COM pin for the LCD controller/driver
- Initialization
- When the "DIV A,Ri" and "DIVW A,RWi" commands are used
- When REALOS is used

Precautions when Handling Devices

- Observing the maximum rated voltage (prevention of latch-up)
  - For a CMOS IC, latch-up may occur when a voltage higher than $V_{CC}$ or a voltage lower than $V_{SS}$ is impressed to the I/O pin other than medium-/high-voltage withstand I/O pins, or when a voltage that exceeds the rated voltage is impressed between $V_{CC}$ and $V_{SS}$. When latch up phenomenon is generated, the power supply current significantly increases, and elements may be damaged due to heat, so care must be taken not to exceed the maximum rating during use).
  - When turning the power on or off to analog power supply, care must be taken that the analog power supply voltage ($AVcc$, $AVRH$, and $DVcc$) and analog input voltage do not exceed the digital power supply voltage ($Vcc$).
  - When the voltage is applied to the power pin for LCD (V3 to V0 pins), care must be taken not to exceed the power supply voltage ($Vcc$).

- Stabilization of supply voltage

Even within the operation guarantee range of the Vcc power supply voltage, if the power supply voltage undergoes a rapid change, erroneous operation may occur, so the Vcc power supply voltage must be stable.

In terms of voltage stabilization guidelines, the Vcc ripple fluctuations (peak to peak value) should be suppressed to within 10% of the standard power supply voltage value (Vcc), and the power supply voltage
should be suppressed to within 0.1 V/ms for the transient fluctuation rate during momentary changes such as when the power supply is switched.

● **At power on**

When the power is turned on, retain 50 µs (at between 0.2 V and 2.7 V) or more while starting up the power supply voltage (Vcc) to prevent erroneous operation of built-in suppression circuit.

● **Power supply pin**

- When there are a number of Vcc/Vss pins, for device design purposes, those pins that should have the same potential are connected within the device in order to prevent a latch-up or other erroneous operation. To reduce unnecessary radiation, prevent erroneous operation of the strobe signal due to increases in the ground level, and to maintain the total output current standard, connect the Vcc/Vss pins to an external power supply and ground them.
- Connect Vcc/Vss to the device from the current supply source at low impedance.
- Connect a capacitor of approximately 0.1 µF between the Vcc/Vss pins near the Vcc/Vss pins of the device as a bypass capacitor to counter any power noise.

● **Crystal oscillation circuit**

- Noise in the X0 and X1 pin may cause erroneous operation of the device. The X0 and X1 pins, crystal oscillator (or ceramic oscillator), and the bypass capacitor to ground connect near the X0 and X1 pins. Design the printed circuit board so that the wiring for the X0 and X1 pins does not intersect any other wiring.
- Stable operation can be expected from the printed circuit artwork that encloses the X0 and X1 pins with the ground.

● **When external clock is used**

When an external clock is used, drive the X0 pin only, and open the X1 pin. Figure 2.1-1 shows example of using external clock.

![Figure 2.1-1 Example of Using External Clock](image)

● **Note on during operation of PLL clock mode**

If the PLL clock mode is selected, the microcontroller attempt to be working with the self oscillating circuit even when there is no external oscillator or external clock input is stopped. Performance of this operation, however, cannot be guaranteed.

● **When sub-clock mode is not used**

Even if sub-clock mode is not used, be sure to connect an oscillator to X0A and X1A pins.
CHAPTER 2  HANDLING DEVICES

● Treatment of unused pins

If unused input pins are kept open, permanent damage may result through erroneous operation and latch up, so they should be pulled up or down by resistance of 2kΩ or more.

If there are any unused input/output pins, open them by setting the output status, or execute the same action as the input pin by setting the input status.

● Handling N.C. pin

Use the Non Connect (N.C.) pin while open.

● When output from ports 0 & 1 is insufficient

After the power is turned on, output from ports 0, 1, P53, P54, P71 and P73 is undefined within the oscillation stabilization wait time (during power on reset) of the suppression circuit. Care must be taken with the timing, as it is shown in Figure 2.1-2.

It does not output because there is no suppression circuit stabilization wait time on models that do not feature the suppression circuit.

**Figure 2.1-2  Timing chart for ports 0 and 1 to be undefined output**

Reference: Refer to "Table 1.2-1 MB90520A series model configuration" for whether or not an internal suppression circuit is on each model in the MB90520A series.

● Pin processing when the A/D converter is not used.

When the A/D and D/A converters are not used, connect them as follows. AVcc = DVcc = AVRHi = Vcc, AVss = AVRL = Vss
● Power pin of the A/D converter, power voltage application to the analog input pin, and disconnection order

  - Application of power to the A/D and D/A converters (AVcc, AVRH, AVRL, DVcc and DVss), and voltage to the analog input pin (AN0 to AN7) must be carried out after applying voltage to the digital power pin (Vcc).
  - When the power is turned off, disconnect digital power after disconnecting A/D converter power and analog input.
  - Apply and disconnect AVRH and DVcc without exceeding AVcc. (Simultaneous application/disconnection of analog and digital power supply does not cause any problem.)

● Using both general-purpose input/output port and SEG/COM pin for the LCD controller/driver

  The SEG08 to SEG31 and COM0 to COM3 devices are used as the general-purpose input/output ports. In terms of electrical standards, the warranted value for SEG08 to SEG23 and COM0 to COM3 will be same as the CMOS output port, while SEG24 to SEG31 will be the same as the Nch open drain port.

● Initialization

  There is built-in register that can only be initialized by a power on reset in the device. Turn on the power again to execute initialization.

● When the "DIV A, Ri" and "DIVW A, RWi" commands are used

  When the signed division command "DIV A, Ri" and "DIVW A, RWi" commands are used, set the supported bank registers" (DTB, ADB, USB, and SSB) values to "00H".
  When the supported bank registers (DTB, ADB, USB, and SSB) value is set to other than "00H", any surplus gained through the command results will not be stored in the register of the command operand.

Reference: Refer to "3.4.5 Precautions when Using the "DIV A, Ri" and "DIVW A, RWi" Commands." for details.

● When REALOS is used

  When using REALOS, it’s not used the extended intelligent I/O service (EIOŚ).
CHAPTER 3
CPU FUNCTION

This chapter explains the CPU function of MB90520A series.

3.1 Memory Space
3.2 Dedicated Registers and General-purpose Registers
3.3 General-purpose Register
3.4 Prefix Code
3.5 Interrupt
3.6 Reset
3.7 Clock
3.8 Low-power Consumption Mode
3.9 CPU Mode
3.1 Memory Space

All I/O, programs and data are allocated in the 16 Mbyte memory space of F^2MC-16LX. Part of the memory space is used for specific uses such as the extended intelligent I/O service (EI^2OS) descriptors, the general-purpose registers, and the vector tables.

---

**Memory Space**

All I/O, programs and data area allocated in the 16 Mbyte memory space. The CPU is able to access each resource using a 24-bit address.

The relationship between the F^2MC-16LX system and the memory map is shown below.

---

*1 : The capacity of the internal RAM depends on the product.
*2 : The capacity of the internal ROM depends on the product.
*3 : Address match detection function register area
CHAPTER 3 CPU FUNCTION

■ ROM Area

● Vector table area (address: FFFC00H to FFFFFFFH)
  - The vector table is provided for reset and interrupts.
  - This area is allocated to the ROM area. The starting address of the corresponding processing routine can be set as data in each vector table address.

● Program area (address: FFBFFFH to FFFFFFFH)
  - Program data can be set.
  - The capacity of the internal ROM depends on the product.

■ RAM Area

● Data area (address: 000100H to 001100H)
  - Built-in static RAM
  - The capacity of the internal RAM depends on the product.

● General-purpose register area (address: 000180H to 00037FH)
  - Registers used for 8-bit, 16-bit and 32-bit operation and transfer operation are allocated in this area.
  - Since this area is allocated the RAM area, it can also be used as RAM function.
  - When this area is used as a general-purpose register, general-purpose register addressing enables to process with short instructions cycle.

● Extended intelligent I/O service (EIO2OS) descriptor area
  (Address: 000100H to 00017FH)
  - This area can be set the transfer mode, I/O addresses, transfer count and buffer addresses.
  - Since this area is allocated the RAM area, it can also be used as RAM function.

■ I/O Area

● Interrupt control register area (address: 0000B0H to 0000BFH)
  - The interrupt control registers (ICR00 to ICR15) set the interrupt level and control the extended intelligent I/O service (EIO2OS).

● Resource function control register area (address: 000020H to 0000AFH)
  - This area controls the resource function and data I/O.

● I/O port control register area (address: 000000H to 00001FH)
  - This area controls the I/O ports and data I/O.

■ Register Area

● Program address detection register area (address: 001FF0H to 001FF5H)

● Address match detection function register reservation area (address: 001FF6H to 001FFFH)
3.1.1 Mapping of and Access to Memory Space

The MB90520A series can be set only in single-chip mode as memory access mode.

- Memory allocation of MB90520A

The MB90520A series runs only in single-chip mode that does not use the external data bus. Figure 3.1-2 shows the memory map when the ROM mirroring function is enabled/disabled.

![Figure 3.1-2 Memory allocation of MB90520A](image)

Address #1 and #2 are different from depending on the kind of products.

*1: Address match detection function register area

*2: MB90V520A does not have a ROM, but it works as well as built-in ROM by operation of dedicated developing tools.

<table>
<thead>
<tr>
<th>Products</th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address #1</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001900H</td>
</tr>
</tbody>
</table>
| Address #2 | FF0000H | FE0000H | FE0000H | FE0000H | FE0000H | — | *2

: Access prohibitance
ROM mirroring function

F2MC-16LX allows such settings so that internal ROM data can be seen in the higher 00 bank. This function, called the ROM mirroring function, is enabled when using the small model of C compiler. When the ROM mirroring function is enabled, the lower 16-bit addresses in the FF bank are the same as the lower 16-bit addresses in the 00 bank, the internal ROM table can be referenced without the "far" specification with a pointer.

For example, when "00C000H" is accessed, the data in the internal ROM at "FFC000H" is actually read. However, it should be noted that since the ROM mirroring function only allows 48 Kbytes area referencing while the ROM area in the FF bank is 64 Kbytes, all areas of the FF bank cannot be seen in the 00 bank. (Because ROM data in the area, "FF4000H to FFFFFFFH" can be referenced in the area, "004000H to 00FFFFFFH", set the ROM data table in the area, "FF4000H to FFFFFFFH".

Reference: Refer to "22.1 Overview of ROM Mirroring Function Select Module" for the ROM mirror function setup method (ROMM:MI = 0).
### Memory Map

MB90520A series memory map is shown for each device.

<table>
<thead>
<tr>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000H</td>
<td>000000H</td>
<td>000000H</td>
</tr>
<tr>
<td>000000H</td>
<td>000000H</td>
<td>000000H</td>
</tr>
<tr>
<td>000100H</td>
<td>000100H</td>
<td>000100H</td>
</tr>
<tr>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
</tr>
<tr>
<td>001FF0H</td>
<td>001FF0H</td>
<td>001FF0H</td>
</tr>
<tr>
<td>004000H</td>
<td>004000H</td>
<td>004000H</td>
</tr>
<tr>
<td>0000FFH</td>
<td>0000FFH</td>
<td>0000FFH</td>
</tr>
<tr>
<td>FF0000H</td>
<td>FF0000H</td>
<td>FF0000H</td>
</tr>
<tr>
<td>FFFFFFFH</td>
<td>FFFFFFFH</td>
<td>FFFFFFFH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MB90523B</th>
<th>MB905V23B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000H</td>
<td>000000H</td>
<td>000000H</td>
</tr>
<tr>
<td>000000H</td>
<td>000000H</td>
<td>000000H</td>
</tr>
<tr>
<td>000100H</td>
<td>000100H</td>
<td>000100H</td>
</tr>
<tr>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
</tr>
<tr>
<td>001FF0H</td>
<td>001FF0H</td>
<td>001FF0H</td>
</tr>
<tr>
<td>004000H</td>
<td>004000H</td>
<td>004000H</td>
</tr>
<tr>
<td>0000FFH</td>
<td>0000FFH</td>
<td>0000FFH</td>
</tr>
<tr>
<td>FF0000H</td>
<td>FF0000H</td>
<td>FF0000H</td>
</tr>
<tr>
<td>FFFFFFFH</td>
<td>FFFFFFFH</td>
<td>FFFFFFFH</td>
</tr>
</tbody>
</table>

- **I/O**: Memory area enabling to access
- **X**: Access prohibition

*1: Address match detection function register area
*2: MB90V520A does not have a ROM, but it works as well as built-in ROM by operation of dedicated developing tools.
3.1.3 Addressing

Linear and bank types are available for addressing. F2MC-16LX family in principle uses bank addressing.

- **Linear addressing**: A complete 24-bit address can directly be specified by an command.
- **Bank addressing**: The upper 8-bit of an address can be specified by an appropriate bank register and the lower 16-bit of the address can be specified by an bit command.

### Linear Addressing and Bank Addressing

In linear addressing, 16 Mbyte memory space is accessed by directly specifying an address. In bank addressing, the 16 Mbyte space is divided into 256 64-Kbyte banks and accesses are made by setting up banks and bank addresses.

Figure 3.1-4 shows overview of memory management in linear and bank type.

#### Figure 3.1-4 Memory Management in Linear and Bank Types
3.1.4 Linear Addressing

The following methods are available for linear addressing.
- A method to directly specify 24-bit addresses using an command
- A method to specify addresses using the lower 24-bit of a 32-bit general-purpose register

- Linear addressing specified by 24-bit operand

  Figure 3.1-5 Example of directly specified 24-bit physical address in linear addressing

- Addressing by indirect specification with a 32-bit register

  Figure 3.1-6 Example of indirectly specified 32-bit general-purpose register in linear addressing

RL1 : 32-bit (long ward) general purpose register
3.1.5 Bank Addressing

In bank addressing, divide the 16 Mbyte memory space into 256 for 64-Kbyte banks, specify the bank address in each bank register, and specify the lower 16 bits of the address using the command.

Bank register has the following five types depending on the use.
- Program counter bank register (PCB)
- Data bank register (DTB)
- User stack bank register (USB)
- System stack bank register (SSB)
- Additional data bank register (ADB)

### Bank Registers and Access Space

Table 3.1-1 shows the access space for each bank register and the major use of it.

<table>
<thead>
<tr>
<th>Bank Register Name</th>
<th>Access Space</th>
<th>Major Use</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program counter bank register (PCB)</td>
<td>Program (PC) space</td>
<td>Used to set up addresses, such as command codes, vector tables, and immediate data.</td>
<td>0FFH</td>
</tr>
<tr>
<td>Data bank register (DTB)</td>
<td>Data (DT) space</td>
<td>Used for the read/write data area, or when the addresses for the control and data registers of the peripheral functions are set.</td>
<td>00H</td>
</tr>
<tr>
<td>User stack bank register (USB)</td>
<td>Stack (SP) space</td>
<td>This is used to set up the stack access address for the save interrupt register under the PUSH/POP command. When the stack flag (CCR:S) is &quot;1&quot;, SSB can be used, when it is &quot;0&quot;, USB can be used.*</td>
<td>00H</td>
</tr>
<tr>
<td>System stack bank register (SSB)*</td>
<td></td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>Additional data bank register (ADB)</td>
<td>Additional (AD) space</td>
<td>Used when setting address of data that could not be entered to the data (DT) space.</td>
<td>00H</td>
</tr>
</tbody>
</table>

*: SSB must be used to stack hardware interruption.
Figure 3.1-7 shows the relationships between the memory space divided into banks and each register.

**Figure 3.1-7 Example of Bank Addressing**

![Diagram of memory space divided into banks and registers]

Reference: For details, see Section "3.2 Dedicated Registers and General-purpose Registers".

### Bank Addressing and Default Space

To improve command coding efficiency, each command has a pre-determined default space for each addressing type, as shown in Table 3.1-2. To use a space other than the default space, specify a prefix code for a bank before the command. This enables access to the bank space that corresponds to the specified prefix code.

**Table 3.1-2 Addressing and Default Spaces**

<table>
<thead>
<tr>
<th>Default Spaces</th>
<th>Addressing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program space</td>
<td>PC indirect addressing, program-access addressing, branch instruction addressing</td>
</tr>
<tr>
<td>Data space</td>
<td>Addressing with @RW0, @RW1, @RW4, @RW5, @A, addr16, and dir</td>
</tr>
<tr>
<td>Stack space</td>
<td>Addressing with PUSHW, POPW, @RW3, and @RW7</td>
</tr>
<tr>
<td>Additional space</td>
<td>Addressing with @RW2 and @RW6</td>
</tr>
</tbody>
</table>

Reference: For details on the prefix codes, see "Section 3.4 Prefix Code".
3.1.6 Multi-byte Data Allocation in Memory

Multi-byte data is written to memory in sequence starting from the low address. For 32-bit length data, the lower 16-bit are written first, and then the higher 16-bit are written. If a reset signal is occurred immediately after the lower 16-bit is written, the higher 16-bit may not be written.

■ Sequence of multi-byte data setting in RAM

Figure 3.1-8 shows the sequence of multi-byte data setting. The lower 8-bit of the data are located at address n, and subsequent bits are located at address n + 1, address n + 2, address n + 3, and so on, in this sequence.

![Figure 3.1-8 Sequence of multi-byte data setting in RAM](image)

■ Sequence of multi-byte operand setting

Figure 3.1-9 shows the sequence of multi-byte operand setting in memory.

![Figure 3.1-9 Sequence of multi-byte operand setting](image)
**Sequence of multi-byte data setting in a stack**

Figure 3.1-10 shows the sequence of multi-byte data setting in a stack.

**Figure 3.1-10 Sequence of multi-byte data setting in a stack**

Accessing to multi-byte data is in principle performed within a bank. For an command that accesses multi-byte data, the address following "FFFFH" address is "0000H" address in the same bank.

Figure 3.1-11 shows an example of access instruction for multi-byte data on the bank boundary.

**Figure 3.1-11 Access to Multi-byte Data on Bank Boundary**
3.2 Dedicated Registers and General-purpose Registers

F²MC-16LX registers are classified into dedicated registers in the CPU and general-purpose registers to be set in the internal RAM.

- **Dedicated registers and general-purpose registers**

  The dedicated registers depend on the hardware architecture of the CPU.

  The general-purpose registers are in the internal RAM area in the CPU address space. It can be used for addressing same as the dedicated registers and not particularly dependent on the hardware architecture.

  Figure 3.2-1 shows dedicated registers and general-purpose registers.

![Figure 3.2-1 Dedicated registers and general-purpose registers](image-url)
3.2.1 Dedicated Registers

The dedicated registers in the CPU are comprised of the following 11 registers.
- Accumulator (A)
- User stack pointer (USP)
- System stack pointer (SSP)
- Processor status (PS)
- Program counter (PC)
- Direct page register (DPR)
- Program counter bank register (PCB)
- Data bank register (DTB)
- User stack bank register (USB)
- System stack bank register (SSB)
- Additional data bank register (ADB)

### Configuration of Dedicated Registers

![Figure 3.2-2 Configuration of Dedicated Registers](image)

- **AH**: Accumulator (A)
  - Using for storing of the arithmetic operation results. 16-bit × 2 registers. Also using as 32-bit register continuously.
- **USP**: User stack pointer (USP)
  - 16-bit stack pointer indicating user stack address
- **SSP**: System stack pointer (SSP)
  - 16-bit stack pointer indicating system stack address
- **PS**: Processor status (PS)
  - 16-bit status register indicating system state
- **PC**: Program counter (PC)
  - 16-bit count register indicating location of storing instruction at present.
- **DPR**: Direct page register (DPR)
  - When using of shortening direct addressing, set bit 8 to bit 15 of 24-bit address. 8-bit page register.
- **PCB**: Program bank register (PCB)
  - 8-bit bank register indicating program space
- **DTB**: Data bank register (DTB)
  - 8-bit bank register indicating data space
- **USB**: User stack bank register (USB)
  - 8-bit bank register indicating user stack space
- **SSB**: System stack bank register (SSB)
  - 8-bit bank register indicating system stack space
- **ADB**: Additional data bank register (ADB)
  - 8-bit bank register indicating additional space
### Table 3.2-1 Reset Values of Dedicated Registers

<table>
<thead>
<tr>
<th>Dedicated Registers</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulator (A)</td>
<td>Undefined</td>
</tr>
<tr>
<td>User stack pointer (USP)</td>
<td>Undefined</td>
</tr>
<tr>
<td>System stack pointer (SSP)</td>
<td>Undefined</td>
</tr>
<tr>
<td>Processor status (PS)</td>
<td>![Processor status (PS)]</td>
</tr>
<tr>
<td>Program counter (PC)</td>
<td>Value of reset vector (data at FFFFDCH and FFFFDH)</td>
</tr>
<tr>
<td>Direct page register (DPR)</td>
<td>01H</td>
</tr>
<tr>
<td>Program counter bank register (PCB)</td>
<td>Value of reset vector (data at FFFFEH)</td>
</tr>
<tr>
<td>Data bank register (DTB)</td>
<td>00H</td>
</tr>
<tr>
<td>User stack bank register (USB)</td>
<td>00H</td>
</tr>
<tr>
<td>System stack bank register (SSB)</td>
<td>00H</td>
</tr>
<tr>
<td>Additional data bank register (ADB)</td>
<td>00H</td>
</tr>
</tbody>
</table>

**Note:** The reset value is the value when the device is reset. This will differ from ICE (emulator).
The accumulator (A) consists of two 16-bit length operation registers (AH and AL) and is used to temporarily store the results of an operation and data. The accumulator can be used as a 32-bit, 16-bit, or 8-bit register.

Accumulator (A)

- Data transfer to accumulator
  
  The accumulator can process 32-bit length (long word), 16-bit length (word), and 8-bit length (byte) data. The 4-bit data transfer command (MOVN) is an exception. The processing of 8-bit data also applies to 4-bit data.
  
  - For 32-bit data processing, the higher operation register (AH) and lower operation register (AL) are concatenated.
  - When processing 16-bit data or 8-bit data, only low-order operation registers (AL) are used and the high-order operation registers (AH) retain the data of the low-order operation registers (AL).

Data retention function

The accumulator has a data retention function that automatically transfers pre-transfer data from the AL register to the AH register when data of word length or less is transferred to the AL register.

Code-extended function and zero-extended function

When data of byte length or less is transferred to the AL register, the data becomes 16-bit length by sign extension (MOVX command) or zero extension (MOV command), and is stored in the AL register. The data in the AL register can be handled as word-length or byte-length data. Figure 3.2-3 shows specific examples of data transfer to the accumulator.

![Figure 3.2-3 Data transfer to accumulator](image)

- 32-bit data transmission
- 16-bit data transmission
- 8-bit data transmission

*00H or FFH* (Zero extension or symbol extension)
● Byte processing arithmetic operation of accumulator

When a byte-processing arithmetic operation command is executed for the AL register, the upper 8 bits of
the AL register before the operation is executed are ignored and the upper 8 bits of the arithmetic operation
results are cleared to "00H".

● Reset value of accumulator

The reset value is undefined.

**Figure 3.2-4  Example of 16-bit Data Transfer to Accumulator (A) (Data Saving)**

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH: XXXXH</td>
<td>AL: 2456H</td>
</tr>
<tr>
<td>DTB: B5H</td>
<td></td>
</tr>
<tr>
<td>2456H</td>
<td>7788H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSB</th>
<th>Memory space</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>77H</td>
<td>88H</td>
<td></td>
</tr>
<tr>
<td>B53001H</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X: Undefined  
MSB: The uppermost bit  
LSB: The lowest bit  
DTB: Data bank register

**Figure 3.2-5  Example of 8-bit Data Transfer to Accumulator (A) (Data Saving, Zero-extended)**

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH: XXXXH</td>
<td>AL: 2456H</td>
</tr>
<tr>
<td>DTB: B5H</td>
<td></td>
</tr>
<tr>
<td>2456H</td>
<td>0088H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSB</th>
<th>Memory space</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>77H</td>
<td>88H</td>
<td></td>
</tr>
<tr>
<td>B53001H</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X: Undefined  
MSB: The uppermost bit  
LSB: The lowest bit  
DTB: Data bank register
Figure 3.2-6  Example of 16-bit length Data Transfer to Accumulator (A) (Data Saving)

```
(MOVW  A, @RW1+6)

Before execution

<table>
<thead>
<tr>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXH</td>
<td>1234H</td>
</tr>
</tbody>
</table>

DTB A6H

MSB  | Memory space  | LSB  |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>15H 38H</td>
<td></td>
</tr>
</tbody>
</table>

After execution

<table>
<thead>
<tr>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234H</td>
<td>2B52H</td>
</tr>
</tbody>
</table>

MSB  | Memory space  | LSB  |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>2Bh 52h</td>
<td></td>
</tr>
</tbody>
</table>

X: Undefined
MSB: The uppermost bit
LSB: The lowest bit
DTB: Data bank register

(Instruction to read long-word data from the address calculated as RW1 content + 8-bit offset, then writing the result to the A register)
```

Figure 3.2-7  Example of 32-bit Data Transfer to Accumulator (A) (Register Indirect)

```
(MOVL  A, @RW1+6)

Before execution

<table>
<thead>
<tr>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXH</td>
<td>XXXH</td>
</tr>
</tbody>
</table>

DTB A6H

MSB  | Memory space  | LSB  |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>15H 38H</td>
<td></td>
</tr>
</tbody>
</table>

After execution

<table>
<thead>
<tr>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>8F74H</td>
<td>2B52H</td>
</tr>
</tbody>
</table>

MSB  | Memory space  | LSB  |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>2Bh 52h</td>
<td></td>
</tr>
</tbody>
</table>

X: Undefined
MSB: The uppermost bit
LSB: The lowest bit
DTB: Data bank register

(Instruction to read long-word data from the address calculated as RW1 content + 8-bit offset, then writing the result to the A register)
3.2.3 Stack Pointers (USP, SSP)

The stack pointers include a user stack pointer (USP) and a system stack pointer (SSP). Both these pointers indicate the address where saved data and return data are stored when the PUSH instruction, the POP instruction, and the subroutine is executed.

- The higher 8 bits of the stack address are set by the user stack bank register (USB) or the system stack bank register (SSB).
- When the stack flag (PS: CCR: S) is 0, the USP and USB register are enabled. When the stack flag is 1, the SSP and SSB register are enabled.

### Stack Selection

F2MC-16LX can use two types of stack pointers: the system stack pointer and the user stack pointer.

The stack pointer address is determined by the stack flag (CCR:S) in the condition code register, as shown in Table 3.2-2.

![Table 3.2-2 Stack Address Specification](image)

<table>
<thead>
<tr>
<th>S Flag</th>
<th>Stack Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Higher 8 Bits</td>
</tr>
<tr>
<td>0</td>
<td>User stack bank register (USB)</td>
</tr>
<tr>
<td>1*</td>
<td>System stack bank register (SSB)</td>
</tr>
</tbody>
</table>

*: Reset value

Because the stack flag (CCR:S) is initialized to "1" by a reset, the system stack is used as default. The system stack pointer is usually used for stack processing during the interrupt routine and the user stack pointer is used for stack processing performed outside the interrupt routine. When the separation of the stack space is not required, only the system stack pointer should be used.

### Note

When an interrupt is accepted, the stack flag (CCR:S) is set, and the system stack pointer must be used.
Figure 3.2-8 shows an example of stack operation using the system stack.

**Figure 3.2-8 Stack operation instruction and stack pointer**

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PUSHW A when S flag is 0</strong></td>
<td><strong>PUSHW A when S flag is 1</strong></td>
</tr>
<tr>
<td>AL: A624H</td>
<td>AL: A624H</td>
</tr>
<tr>
<td>USB: C6H</td>
<td>USB: C6H</td>
</tr>
<tr>
<td>USP: F328H</td>
<td>USP: F328H</td>
</tr>
<tr>
<td>S flag: 0</td>
<td>S flag: 1</td>
</tr>
<tr>
<td>SSB: 56H</td>
<td>SSB: 56H</td>
</tr>
<tr>
<td>SSP: 1234H</td>
<td>SSP: 1234H</td>
</tr>
<tr>
<td>MSB: XXH</td>
<td>MSB: XXH</td>
</tr>
<tr>
<td>LSB: XXH</td>
<td>LSB: XXH</td>
</tr>
<tr>
<td>Use the user stack pointer because S flag is 0.</td>
<td>Use the system stack pointer because S flag is 1.</td>
</tr>
</tbody>
</table>

**Notes:**
- When stack pointer address is set, even addresses should be set. Whenever an odd address is set, word access is divided by two, and access efficiency is degraded.
- The reset value is undefined after the USP register and SSP register is reset.

**System stack pointer (SSP)**
To use the system stack pointer (SSP), set the stack flag (CCR:S) to "1". The upper 8 bits of the address that will be used for stack processing should be set by the system stack bank register (SSB).

**User stack pointer (USP)**
To use the user stack pointer (USP), set the stack flag (CCR:S) to "0". The upper 8 bits of the address that will be used for stack processing should be set by the user stack bank register (USB).
## Stack Area

### Securing stack area

The stack area is also used for saving/returning the program counter (PC) when executing the interrupt processing, subroutine call command (CALL) or vector call command (CALLV) as well as for saving/returning temporary registers using the PUSHW and POPW commands.

The stack area is allocated in RAM with the data area.

The stack area is as shown below:

**Figure 3.2-9 Stack Area**

<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000H</td>
<td>I/O area</td>
</tr>
<tr>
<td>00000H</td>
<td>Built-in RAM area</td>
</tr>
<tr>
<td>000100H</td>
<td>General purpose register bank area</td>
</tr>
<tr>
<td>000180H</td>
<td></td>
</tr>
<tr>
<td>000380H</td>
<td></td>
</tr>
<tr>
<td>001100H</td>
<td></td>
</tr>
<tr>
<td>FF000H</td>
<td>ROM area</td>
</tr>
<tr>
<td>FFC00H</td>
<td></td>
</tr>
<tr>
<td>FFFF0H</td>
<td></td>
</tr>
</tbody>
</table>

*: Built-in ROM capacitance is different depending on products.

**Notes:**
- Set an even address in the stack pointer (SSP, USP). Whenever an odd address is set, word access is divided by two, and access efficiency is degraded.
- The system stack area, user stack area, and data area should not overlap.

### System stack and user stack

The system stack area is used for interrupt processing. Forcibly switches to the system stack when an interrupt is generated even when using the user stack area. Therefore, be sure to correctly set up the system stack area if the system primarily uses the user stack area.

When the separation of the stack space is not required, only the system stack should be used.
3.2.4 Processor Status (PS)

The processor status (PS) consists of CPU control bits and bits that indicate the CPU status. The processor status (PS) consists of the following three registers.
- Interrupt level mask register (ILM)
- Register bank pointer (RP)
- Condition code register (CCR)

### Configuration of Processor Status (PS)

The processor status (PS) consists of CPU control bits and bits that indicate the CPU status. Figure 3.2-10 shows the configuration of the processor status (PS).

#### Figure 3.2-10 Processor status (PS)

<table>
<thead>
<tr>
<th>ILM</th>
<th>RP</th>
<th>CCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit15 14 13 12 11 10 9 8</td>
<td>7 6 5 4 3 2 1 bit0</td>
<td></td>
</tr>
<tr>
<td>PS</td>
<td>ILM2 ILM1 ILM0 B4 B3 B2 B1 B0</td>
<td>I S T N Z V C</td>
</tr>
<tr>
<td>Reset value</td>
<td>0 0 0 0 0 0 0 0 × 0 1 × × × × ×</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>Undefined</td>
<td></td>
</tr>
</tbody>
</table>

- **Interrupt level mask register (ILM)**
  
  This register indicates the level of the interrupt that the CPU is currently accepting. Compared with the value of the interrupt level setting bit (ICR:IL0 to IL2) in the interrupt control register set according to the interrupt request from each peripheral function.

- **Register bank pointer (RP)**
  
  This register sets the memory block (register bank) to be used for the general-purpose registers allocated in the internal RAM area.

  Up to 32 general-purpose register banks can be set up and the general-purpose register bank to be used can be specified by setting the register bank pointer (RP) to "00H to 1FH".

- **Condition code register (CCR)**
  
  Consists of seven types of flags that are either set to "1" or cleared to "0" based on the command execution result or interrupt request acceptance.
3.2.4.1 Condition Code Register (PS: CCR)

The condition code register (CCR) is an 8-bit register that consists of the bits that indicate the processing result of an command executed and the bits that enable or disable an interrupt request.

Configuration of Condition Code Register (CCR)

Figure 3.2-11 shows the configuration of the CCR register.

- **Interrupt enable flag (I)**
  
  In response to all interrupt requests other than software interrupts when the interrupt enable flag (CCR:I) is set to "1", interrupts are enabled. When set to "0", interrupts are disabled. This flag is cleared to "0" by a reset.

- **Stack flag (S)**
  
  The stack flag specifies the pointer used for stack processing.

  When the stack flag (CCR:S) is set to "0", the user stack pointer (USP) is enabled. When it is set to "1", the system stack pointer (SSP) is enabled. Set to "1" when an interrupt is accepted or a reset signal is input.

- **Sticky flag (T)**
  
  The sticky flag bit is set to "1" when the data shifted out by the carry contains at least one that is set to "1" when a logical right shift command or arithmetic right shift command is executed. Cleared to "0" when the data shifted out are all "0" or the shift amount is zero.

- **Negative flag (N)**
  
  Set to "1" when the most significant bit is "1" as the result of an operation or cleared to "0" when the most significant bit is "0".
Zero flag (Z)
Set to "1" when the bits are all "0s" as the result of an operation or cleared to "0" when there is at least one bit that is set to "1".

Overflow flag (V)
Set to "1" if a signed numeric value overflows when an arithmetic operation is executed or cleared to "0" if no overflow occurs.

Carry flag (C)
Set to "1" when there is an carry from the most significant bit or an carry from the least significant bit when an operation is executed, or cleared to "0" when there is no overflow or underflow when an operation is executed.

Reference: For the state of the condition code register (CCR) during instruction execution, refer to the Programming Manual.
3.2.4.2 Register Bank Pointer (PS: RP)

The register bank pointer (RP) is a 5-bit register that indicates the starting address of the currently used general-purpose register bank.

■ Register bank pointer (RP)

Figure 3.2-12 shows the configuration of the register bank pointer (RP).

Figure 3.2-12 Configuration of Register Bank Pointer (RP)

![Figure 3.2-12 Configuration of Register Bank Pointer (RP)](image)

■ General-purpose Register Area and Register Bank Pointer

The register bank pointer (RP) indicates the first address of the general-purpose register bank currently used. Figure 3.2-13 shows the conversion rules applicable to the relationship between the contents of the register bank pointer (RP) and the real address.

Figure 3.2-13 Physical Address Conversion Rules in General-purpose Register Area

![Figure 3.2-13 Physical Address Conversion Rules in General-purpose Register Area](image)

- Since the register bank pointer (RP) can take a value of "00H to 1FH", the first address of the register bank can be set in the range from "000180H to 00037FH".
- Although an assembler command can use an 8-bit immediate value transfer command for transfer to the register bank pointer (RP), only the lower 5-bit of the transferred data will be stored in the register bank pointer (RP).
- Register bank pointer (RP) will be "00H" after a reset.
3.2.4.3 Interrupt Level Mask Register (ILM)

The interrupt level mask register (ILM) is a 3-bit register that indicates the level of the interrupt currently accepted by the CPU.

■ Interrupt level mask register (ILM)

Figure 3.2-14 shows the configuration of the interrupt level mask register (ILM).

The interrupt level mask register (ILM) indicates the level of the interrupt currently accepted by the CPU and compares the value of the interrupt level setting bit (ICR:IL2 to IL0) set according to the interrupt request from each peripheral resource. Only when an interrupt request with a value (that defines the interrupt level) smaller than the value of the interrupt level mask register (ILM) is made while the interrupt is enabled (CCR:I=1), the CPU accepts the interrupt and performs interrupt processing.

- When an interrupt is accepted, the value that defines the level of the accepted interrupt will be set in the interrupt level mask register (ILM). Thereafter, an interrupt with a level value lower than the set level value is not accepted.
- The interrupt level mask register (ILM) will reach interrupt disabled state (maximum interrupt level) when it is initialized to all "0s" by the input of a reset signal.
- Although an assembler command can use an 8-bit immediate value transfer command for transfer to the interrupt level mask register (ILM), only the lower 3-bit of the transferred data will be stored in the interrupt level mask register (ILM).

Table 3.2-3 Interrupt level mask register (ILM) and strength/weakness of the interrupt level

<table>
<thead>
<tr>
<th>ILM2</th>
<th>ILM1</th>
<th>ILM0</th>
<th>Interrupt level</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

High (interrupt prohibited)

Reference: For details of interrupt, see "3.5 Interrupt".
3.2.5 Program Counter (PC)

The program counter (PC) is a 16-bit counter that indicates the lower 16-bit of the address of the next command code to be executed by the CPU.

Program counter (PC)

- The program bank register (PCB) indicates the higher 8-bit of addresses where the next command code to be executed by the CPU is stored; the program counter (PC) indicates the lower 16-bit. The actual address is combined to become 24-bit, as shown in Figure 3.2-15.
- The contents of the program counter (PC) are updated by the conditional branch command, subroutine call command, interrupt and reset.
- The program counter (PC) can also be used as the base pointer when reading the operand.

Figure 3.2-15 Program counter (PC)

![Diagram of Program counter (PC)]

Note: Neither the program counter (PC) nor the program bank register (PCB) can be rewritten directly by a program (such as MOV PC, #FF).
3.2.6 Direct Page Register (DPR)

To execute a short direct addressing command, specify 8 intermediate bits (bit15 to bit8) for the lower 8 bits of the address (bit7 to bit0) directly specified by the operand in the direct page register (DPR).

Direct page register (DPR)

To execute a short direct addressing command, specify 8 intermediate bits (bit15 to bit8) for the lower 8 bits of the address (bit7 to bit0) directly specified by the operand in the direct page register (DPR). The direct page register (DPR) is 8-bit length and initialized to "01H" by a reset. The DPR can be read and written.

Figure 3.2-16 Generation of physical address by the direct page register (DPR)

Figure 3.2-17 shows an example of a direct page register (DPR) setting and data access.

Figure 3.2-17 Example of direct page register (DPR) setting and data access
3.2.7 Bank Register (PCB, DTB, USB, SSB, and ADB)

The bank register specifies the most significant 8-bit of a 24-bit address when making access by bank addressing. It consists of the five registers shown below.

- Program counter bank register (PCB)
- Data bank register (DTB)
- User stack bank register (USB)
- System stack bank register (SSB)
- Additional data bank register (ADB)

Specify a bank register to access memory banks where the program (PCB), data (DTB), user stack (USB), system stack (SSB) and additional (ADB) space are located.

■ Program counter bank register (PCB)
   The program counter bank register (PCB) specifies the program (PC) address bank.
   The PCB is rewritten when the JMPP, CALLP, RETP or RETI command that branches out into the entire 16 Mbyte space, a software interrupt command or a hardware interrupt is executed or when an exceptional interrupt occurs.

■ Data bank register (DTB)
   The data bank register (DTB) specifies the data address bank.

■ User Stack Bank Register (USB) and System Stack Bank Register (SSB)
   User Stack Bank Register (USB) and System Stack Bank Register (SSB) are setting register for Stack (SP) address bank. The USB and SSB can be switched according to the set value of the stack flag (CCR:S).

■ Additional data bank register (ADB)
   The additional data bank register (ADB) specifies the additional address bank.

■ Setting of Each Bank and Data Access
   Bank registers are 8-bit length. The program bank register (PCB) is initialized to reset vector value by a reset. Also initializes bank registers other than PCB to "00H".
   The program bank register (PCB) is a read-only register. Bank registers other than PCB can be read or written.

Reference: For the operation of each bank register, see "Section 3.1 Memory Space".
### 3.3 General-purpose Register

The general-purpose registers are located at addresses from "000180H" to "00037FH" of the internal RAM, with 16-bit $\times$ 8 allocated as a bank, as shown below.

- General-purpose 8-bit register (byte registers R0 to R7)
- 16-bit register (word registers RW0 to RW7)
- 32-bit register (long-word registers RL0 to RL7)

#### Configuration of a general-purpose register

The general-purpose registers are assigned to 32 banks in the internal RAM addressed from "000180H" to "00037FH" and the banks to be used are specified by the register bank pointer (RP). By reading the register bank pointer (RP), the bank address currently in use can be referenced.

The register bank pointer (RP) specifies the first address of the general-purpose register bank, using the following formula.

**Starting address of general-purpose register = 000180H + RP $\times 10H**

Figure 3.3-1 shows the location and configuration of the general-purpose register banks.

---

**Figure 3.3-1 Allocation and Configuration of General-Purpose Register Banks in Memory Space**

<table>
<thead>
<tr>
<th>Address</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>000180H</td>
<td>Register bank 0</td>
</tr>
<tr>
<td>000190H</td>
<td>Register bank 1</td>
</tr>
<tr>
<td>0001A0H</td>
<td>Register bank 2</td>
</tr>
<tr>
<td>0002B0H</td>
<td>Register bank 19</td>
</tr>
<tr>
<td>0002C0H</td>
<td>Register bank 20</td>
</tr>
<tr>
<td>0002E0H</td>
<td>Register bank 21</td>
</tr>
<tr>
<td>000360H</td>
<td>Register bank 30</td>
</tr>
<tr>
<td>000370H</td>
<td>Register bank 31</td>
</tr>
</tbody>
</table>

- **Byte address**: 02C0H to 02CFH
- **Word address**: 02C1H to 02CDH
- **Long-word address**: 02C2H to 02CFH

**Conversion formula**: [000180H + RP $\times 10H$]

- R0 to R7: Byte register
- RW0 to RW7: Word register
- RL0 to RL3: Long word register

**Note**: The register bank pointer (RP) is initialized to "00000B" by a reset.
- **Register Bank**

A register bank can be used as general-purpose registers (byte registers R0 to R7, word registers RW0 to RW7 and long-word register RL0 to RL3) for operations and pointers. A long-word register can be used for linear addressing to make direct address access.

The contents of the general-purpose register, like ordinary RAM, are not initialized by a reset. The state before a reset is retained. However, at power on, the contents are undefined.

Table 3.3-1 shows the typical function of the general-purpose register.

**Table 3.3-1 Typical Function of the General-purpose Register**

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0 to R7</td>
<td>This can be used as the operand of the command. <strong>Note:</strong> R0 can also be used as the barrel shift counter or the normalized command counter.</td>
</tr>
<tr>
<td>RW0 to RW7</td>
<td>This can be used when address setting is performed. This can be used as the operand of the command. <strong>Note:</strong> RW0 can also be used as the string instruction counter.</td>
</tr>
<tr>
<td>RL0 to RL3</td>
<td>This can be used when linear method address setting is performed. This can be used as the operand of the command.</td>
</tr>
</tbody>
</table>
3.4 Prefix Code

By placing a prefix code before an command to be executed, the operation of the command can partially be changed. The three types of prefix codes are as follows:
- Bank select prefix (PCB, DTB, ADB, and SPB)
- Common register bank prefix (CMR)
- Flag change inhibit prefix (NCC)

### Prefix Code

- **Bank select prefix (PCB, DTB, ADB, and SPB)**
  By placing a bank select prefix code (PCB, DTB, ADB, and SPB) before an command to be executed, the address to be accessed by the command can be specified regardless of the addressing method currently in use.

- **Common register bank prefix (CMR)**
  By placing a common register bank prefix code (CMR) before an command that accesses a general-purpose register, the register accessed by the command that comes immediately after the CMR can be changed to the common bank, "000180H" to "00018FH", (register bank selected when the register bank pointer is set to 0), regardless of the current register bank pointer (RP) value.

- **Flag change inhibit prefix (NCC)**
  By placing a flag change suppression prefix code (NCC) before a command to change each flag in the condition code register, the flag change accompanying the execution of the command can be suppressed.
3.4.1 Bank Select Prefix (PCB, DTB, ADB, SPB)

By placing a bank select prefix code before an command to be executed, the address to be accessed by the command can be specified regardless of the addressing method currently in use.

■ Bank select prefix (PCB, DTB, ADB, and SPB)

The address used for data access is defined for each addressing method. However, by placing a bank select prefix code before an command, the address to be accessed by the command can be specified regardless of the addressing method currently in use. Table 3.4-1 shows the bank select prefix code and the specified address space.

Table 3.4-1 Bank Select Prefix

<table>
<thead>
<tr>
<th>Bank Select Prefix</th>
<th>Selected Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCB</td>
<td>Program space</td>
</tr>
<tr>
<td>DTB</td>
<td>Data space</td>
</tr>
<tr>
<td>ADB</td>
<td>Additional space</td>
</tr>
<tr>
<td>SPB</td>
<td>When the stack flag (CCR:S) is &quot;0&quot;, user stack space will be set. When it is &quot;1&quot;, system stack space will be set.</td>
</tr>
</tbody>
</table>

It should be noted that when a bank select prefix code (PCB, DTB, ADB, SPB) is used, some commands may lead to unexpected operations as shown below.

Table 3.4-2 shows commands that are not affected by the bank select prefix code. On the other hand, Table 3.4-3 shows commands that require caution when using the bank select prefix code.

Table 3.4-2 Instructions Unaffected by Bank Select Prefix

<table>
<thead>
<tr>
<th>Instruction Types</th>
<th>Instruction</th>
<th>Effect of bank select prefix code</th>
</tr>
</thead>
<tbody>
<tr>
<td>String instruction</td>
<td>MOVS, SCEQ, FILS</td>
<td>The bank register specified for the operand is used irrespective of the presence or absence of the bank select prefix code.</td>
</tr>
<tr>
<td>Stack instruction</td>
<td>PUSHW, POPW</td>
<td>Irrespective of the presence or absence of the bank select prefix code, the user stack bank (USB) is used when the S flag is 0; and the system stack bank (SSB) is used when the S flag is 1.</td>
</tr>
<tr>
<td>I/O Access instruction</td>
<td>MOV A, io, MOV io, A, MOV io,#imm8, MOV A,io:bp, MOV io:bp, rel</td>
<td>The I/O space (&quot;000000H&quot; to &quot;0000BFH&quot;) is accessed irrespective of the presence or absence of the bank select prefix code.</td>
</tr>
<tr>
<td>Interrupt return instruction</td>
<td>RETI</td>
<td>The system stack bank (SSB) is used irrespective of the presence or absence of the bank select prefix code.</td>
</tr>
</tbody>
</table>
### Table 3.4-3 Instructions Requiring Precaution When Using Bank Select Prefix

<table>
<thead>
<tr>
<th>Instruction Types</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag change instruction</td>
<td>AND CCR, #imm8 OR CCR, #imm8</td>
<td>The bank select prefix code affects up to the next instruction.</td>
</tr>
<tr>
<td>ILM setting instruction</td>
<td>MOV ILM, #imm8</td>
<td>The bank select prefix code affects up to the next instruction</td>
</tr>
<tr>
<td>PS return instruction</td>
<td>POPW PS</td>
<td>Do not add the bank select prefix code to the PS return instruction.</td>
</tr>
</tbody>
</table>
3.4.2 Common Register Bank Prefix (CMR)

By placing a common register bank prefix code (CMR) before an command that accesses a general-purpose register, the register accessed by the command that comes immediately after the CMR can be changed to the common bank, "000180\text{H}" to "00018\text{FH}\text{H}", regardless of the current register bank pointer (RP) value.

**Common Register Bank Prefix (CMR)**

F\textsuperscript{2}MC-16LX has a common register bank at "000180\text{H}" to "00018\text{FH}\text{H}" for common accesses regardless of the value of the register bank pointer (RP). The use of the common banks facilitates data exchange between two or more tasks.

By placing a common register bank prefix code (CMR) before an command that accesses a general-purpose register, the register accessed by the command that comes immediately after the CMR can be changed to the common bank, "000180\text{H}" to "00018\text{FH}\text{H}" (register bank selected when RP=0) regardless of the current register bank pointer (RP) value.

Table 3.4-4 shows the instructions requiring precaution when using the common register bank prefix.

<table>
<thead>
<tr>
<th>Instruction Types</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String instruction</td>
<td>MOV, SCEQ, FILS</td>
<td>Do not add the CMR code to string instructions.</td>
</tr>
<tr>
<td></td>
<td>MOVSW, SCWEQ, FILSW</td>
<td></td>
</tr>
<tr>
<td>Flag change instruction</td>
<td>AND, OR</td>
<td>The CMR code affects up to the next instruction.</td>
</tr>
<tr>
<td></td>
<td>CCR, #imm8, CCR, #imm8</td>
<td></td>
</tr>
<tr>
<td>PS return instruction</td>
<td>POP#W, PS</td>
<td>The CMR code affects up to the next instruction.</td>
</tr>
<tr>
<td>ILM setting instruction</td>
<td>MOV, ILM, #imm8</td>
<td>The CMR code affects up to the next instruction.</td>
</tr>
</tbody>
</table>
3.4.3 Flag Change Inhibit Prefix (NCC)

By placing an NCC prefix code before an command to change each flag in the condition code register, the flag change accompanying the execution of the command can be suppressed.

■ Flag change inhibit prefix (NCC)

The flag change inhibit prefix code (NCC) is used to suppress the flag change in the condition code register.

By placing a prefix code (NCC) before an command to change each flag, the flag change in the condition code register accompanying the execution of the command can be suppressed. Flag changes that are suppressed are shown below.

- Sticky-bit flag (CCR: T)
- Negative flag (CCR: N)
- Zero flag (CCR: Z)
- Overflow flag (CCR: V)
- Carry flag (CCR: C)

Table 3.4-5 shows the instructions requiring precaution when using the flag change inhibit prefix.

<table>
<thead>
<tr>
<th>Instruction Types</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String instruction</td>
<td>MOVSS</td>
<td>Do not add the NCC code to the string instruction.</td>
</tr>
<tr>
<td></td>
<td>SCEQS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FILSS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOVSW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SCWESQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FILSW</td>
<td></td>
</tr>
<tr>
<td>Flag change instruction</td>
<td>AND CCR,#imm8</td>
<td>The CCR changes by execution of an instruction, regardless of the presence or absence of the NCC code. The NCC code affects the next instruction.</td>
</tr>
<tr>
<td></td>
<td>OR CCR,#imm8</td>
<td></td>
</tr>
<tr>
<td>PS return instruction</td>
<td>POPW PS</td>
<td>The CCR changes by execution of an instruction, regardless of the presence or absence of the NCC code. The NCC code affects the next instruction.</td>
</tr>
<tr>
<td>ILM setting instruction</td>
<td>MOV ILM,#imm8</td>
<td>The NCC code affects the next instruction.</td>
</tr>
<tr>
<td>Interrupt instruction</td>
<td>INT #vct8</td>
<td>The CCR changes by execution of an instruction, regardless of the presence or absence of the NCC code.</td>
</tr>
<tr>
<td>Interrupt return instruction</td>
<td>INT9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>INTP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>addr16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>addr24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RETI</td>
<td></td>
</tr>
<tr>
<td>Context Switch command</td>
<td>JCTX @A</td>
<td>The CCR changes by execution of an instruction, regardless of the presence or absence of the NCC code.</td>
</tr>
</tbody>
</table>
3.4.4 Restrictions on Prefix Code

The use of the prefix codes is restricted as follows:
- Interrupt requests are not accepted during the execution of a prefix code or an interrupt inhibit instruction.
- If a prefix code is placed before an interrupt inhibit instruction, the effect of the prefix code is delayed.
- When conflicting prefix codes are used in succession, the last prefix code is enabled.

### Prefix Code and Interrupt Inhibit Instruction

Interrupt inhibit instruction and prefix code are restricted as shown Table 3.4-6.

<table>
<thead>
<tr>
<th>Instruction that does not accept interrupt request</th>
<th>Prefix Code</th>
<th>Interrupt inhibit instruction (The command that delays effects of prefix code)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCB</td>
<td></td>
<td>MOV ILM, #imm8</td>
</tr>
<tr>
<td>DTB</td>
<td></td>
<td>OR CCR, #imm8</td>
</tr>
<tr>
<td>ADB</td>
<td></td>
<td>AND CCR, #imm8</td>
</tr>
<tr>
<td>SPB</td>
<td></td>
<td>POPW PS</td>
</tr>
<tr>
<td>CMR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NCC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Interrupt inhibit**

Interrupt requests are not accepted during the execution of a prefix code or an interrupt inhibit instruction. If an command is executed after a prefix code or an interrupt inhibit instruction is executed, an interrupt request is accepted and interrupt processing is performed.

**Figure 3.4-1 Interrupt inhibit**

![Diagram showing interruption inhibit instruction and interrupt request generating](image-url)
● Delay of the effect of the prefix code

When a prefix code precedes an interrupt inhibit instruction, it affects an instruction next to the interrupt inhibit instruction.

**Figure 3.4-2 Interrupt Inhibit Instruction and Prefix Code**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV A, FFH</td>
<td>NCC</td>
</tr>
<tr>
<td>MOV ILM, #imm8</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>ADD A, 01H</td>
<td></td>
</tr>
</tbody>
</table>

CCR: XXX10XXe

CCR does not change by NCC.

**Array of Prefix Codes**

Conflicting prefix codes (PCB, ADB, DTB, and SPB) is arrayed, the last one is enabled.

**Figure 3.4-3 Array of Prefix Codes**

<table>
<thead>
<tr>
<th>Prefix code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>ADB</td>
</tr>
<tr>
<td>...</td>
<td>DTB</td>
</tr>
<tr>
<td>...</td>
<td>PCB</td>
</tr>
<tr>
<td>ADD A, 01H</td>
<td></td>
</tr>
</tbody>
</table>

The effective prefix code is PCB.
3.4.5 Precautions when Using the "DIV A, Ri" and "DIVW A, RWi" Commands.

When using the "DIV A, Ri" and "DIVW A, RWi" commands, set the bank register to "00H".

Precautions when using the "DIV A, Ri" and "DIVW A, RWi" commands.

Table 3.4-7 Precautions when using the "DIV A, Ri" and "DIVW A, RWi" commands. (i=0 to 7)

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Use of command shown at left effects bank register name</th>
<th>Address storing excess</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIV A,R0</td>
<td>DTB</td>
<td>(ADB: high-order 8 bits) + (0180H + RP × 10H + 8H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R1</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + 9H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R4</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + CH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R5</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + DH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW0</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + 0H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW1</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + 2H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW4</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + 8H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW5</td>
<td>DTB</td>
<td>(DTB: high-order 8 bits) + (0180H + RP × 10H + AH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R2</td>
<td>ADB</td>
<td>(ADB: high-order 8 bits) + (0180H + RP × 10H + A0H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R6</td>
<td>ADB</td>
<td>(ADB: high-order 8 bits) + (0180H + RP × 10H + EH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW2</td>
<td>ADB</td>
<td>(ADB: high-order 8 bits) + (0180H + RP × 10H + 4H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW6</td>
<td>ADB</td>
<td>(ADB: high-order 8 bits) + (0180H + RP × 10H + EH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R3</td>
<td>USB</td>
<td>(USB*: high-order 8 bits) + (0180H + RP × 10H + BH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIV A,R7</td>
<td>USB</td>
<td>(USB*: high-order 8 bits) + (0180H + RP × 10H + FH: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW3</td>
<td>USB</td>
<td>(USB*: high-order 8 bits) + (0180H + RP × 10H + 6H: low-order 16 bits)</td>
</tr>
<tr>
<td>DIVW A,RW7</td>
<td>USB</td>
<td>(USB*: high-order 8 bits) + (0180H + RP × 10H + Eh: low-order 16 bits)</td>
</tr>
</tbody>
</table>

*1: CCR register S bit
*2: When CCR register S bit is 0

If the values of the bank registers (DTB, ADB, USB, and SSB) are set to "00H", the remainder of the division result is stored in the command operand register. If the bank register value is not "00H", the high-order 8-bit address is specified by the bank register that corresponds to the command operand register, the low-order 16-bit address is the same as that of the command operand register and the remainder is stored in the bank register specified by the upper 8 bits.
[example]
If "DIV A, R0" is executed when DTB= "053H" and RP= "03H", the R0 address becomes "0180H" + RP ("03H") × "10H" + "08H" (R0 equivalent address) = "0001B8H".
Since the bank register specified by "DIV A, R0" here is the data bank register (DTB), the remainder is stored in the address "05301B8H" that has the bank address "053H" added.

<table>
<thead>
<tr>
<th>Reference:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• For more information on the bank register, refer to &quot;3.2.7 Bank Register (PCB, DTB, USB, SSB, and ADB)&quot;.</td>
</tr>
<tr>
<td>• For more information on the Ri and RWi registers, refer to &quot;3.3 General-purpose Register&quot;.</td>
</tr>
</tbody>
</table>

### Avoidance of the precautions
To allow you to avoid precautions when using the "DIV A, Ri" and "DIVW A, RWi" commands, in developing programs, compilers altered not to generate the commands shown in Table 3.4-7 and assemblers provided with a function to replace commands shown in Table 3.4-7 with equivalent command rows have been developed. With regard to MB90520A series compilers and assemblers, make sure that ones meeting the following requirements are used:

- **Compiler**
  - V02L06 or later version of cc907 and V30L02 or later version of fcc907s

- **Assembler**
  - V03L04 or later version of asm907a and V30L04(Rev.300004) or later version of fasm907s

---

For more information on the bank register, refer to "3.2.7 Bank Register (PCB, DTB, USB, SSB, and ADB)".
For more information on the Ri and RWi registers, refer to "3.3 General-purpose Register".
3.5 Interrupt

F2MC-16LX has 4 interruption functions that stop the current processing temporarily and execute an interruption processing program when an event occurs.

- Hardware Interrupt
- Software interrupt
- Interrupts by extended intelligent I/O service (El²OS)
- Exception processing

Type and Function of Interrupt

- Hardware Interrupt
  A user-defined interruption processing program is executed in response to an interruption request from a peripheral function.

- Software interrupt
  By executing command dedicated for software interruptions (INT command, etc.), a user-defined interruption processing program is carried out.

- Interrupts by extended intelligent I/O service (El²OS)
  The extended intelligent I/O service (El²OS) provides automatic data transfer between peripheral function and memory. When the completion program and the setting program for activating an El²OS have been defined by the user, data transfers can be performed. Upon completion of data transfer processing, an interruption processing program is executed automatically.

  Interrupt activating extended intelligent I/O service (El²OS) is the same as hardware interruption.

- Exception processing
  In the exceptional processing, if an exceptional event (such as the execution of an undefined command) is detected, the normal processing is stopped temporarily and a software interruption is performed. Processing that is equivalent to the software interruption command "INT#10" is performed.
CHAPTER 3 CPU FUNCTION

■ Interrupt Operation

Figure 3.5-1 shows interrupt start and return processing.

Figure 3.5-1 General Flow of Interrupt Operation

*: When a string type instruction is being executed, checks for interrupts are made in each step.
3.5.1 Interruption Factors and Interruption Vectors

The F²MC-16LX family has vector tables corresponding to 256 types of interrupt factor.

**Interruption vector**

The interruption vector table that is referred to during interruption processing is allocated to the top address ("FFFFC0H" to "FFFFFFFH") in the memory area. Interruption vectors use the same area for extended intelligent I/O services (EI²OS), exception processing and hardware and software interruptions.

- Interrupts (INT0 to INT255) are used as software interrupts.
- For hardware interruptions, the interruption vector and interrupt control register (ICR) for each peripheral function are fixed.

Table 3.5-1 shows the assignment of interruption numbers and interruption vectors.

<table>
<thead>
<tr>
<th>Software Interrupt Instruction</th>
<th>Vector address (Low)</th>
<th>Vector address (Middle)</th>
<th>Vector address (High)</th>
<th>Mode Data</th>
<th>Interrupt number</th>
<th>Hardware Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT0</td>
<td>FFFFC0H</td>
<td>FFFFD0H</td>
<td>FFFFFE0H</td>
<td>Unused</td>
<td>#0</td>
<td>None</td>
</tr>
<tr>
<td>INT7</td>
<td>FFFFE0H</td>
<td>FFFFE1H</td>
<td>FFFFE2H</td>
<td>Unused</td>
<td>#7</td>
<td>None</td>
</tr>
<tr>
<td>INT8</td>
<td>FFFFDCH</td>
<td>FFFFD0H</td>
<td>FFFDF0H</td>
<td>#8</td>
<td>(RESET vector)</td>
<td>None</td>
</tr>
<tr>
<td>INT9</td>
<td>FFFFD9H</td>
<td>FFFFD8H</td>
<td>FFFFD8H</td>
<td>Unused</td>
<td>#9</td>
<td>None</td>
</tr>
<tr>
<td>INT10</td>
<td>FFFFD4H</td>
<td>FFFFD5H</td>
<td>FFFFD6H</td>
<td>Unused</td>
<td>#10</td>
<td>&lt;Exception processing&gt;</td>
</tr>
<tr>
<td>INT11</td>
<td>FFFFD0H</td>
<td>FFFFD1H</td>
<td>FFFFD2H</td>
<td>Unused</td>
<td>#11</td>
<td>Peripheral function interrupt #0</td>
</tr>
<tr>
<td>INT12</td>
<td>FFFFC0H</td>
<td>FFFFC1H</td>
<td>FFFFC2H</td>
<td>Unused</td>
<td>#12</td>
<td>Peripheral function interrupt #1</td>
</tr>
<tr>
<td>INT13</td>
<td>FFFFC9H</td>
<td>FFFFC8H</td>
<td>FFFFC9H</td>
<td>Unused</td>
<td>#13</td>
<td>Peripheral function interrupt #2</td>
</tr>
<tr>
<td>INT14</td>
<td>FFFFC4H</td>
<td>FFFFC5H</td>
<td>FFFFC6H</td>
<td>Unused</td>
<td>#14</td>
<td>Peripheral function interrupt #3</td>
</tr>
<tr>
<td>INT254</td>
<td>FFFC04H</td>
<td>FFFC05H</td>
<td>FFFC06H</td>
<td>Unused</td>
<td>#254</td>
<td>None</td>
</tr>
<tr>
<td>INT255</td>
<td>FFFC00H</td>
<td>FFFC01H</td>
<td>FFFC02H</td>
<td>Unused</td>
<td>#255</td>
<td>None</td>
</tr>
</tbody>
</table>

**Reference:** It is recommended that unused interruption vectors be set to the address, such as exception processing.
### Interruption factors, interruption vectors, and interrupt control register

Table 3.5-2 shows the relationships between the interruption vectors, interrupt control registers and interruption factors (except for software interruptions).

<table>
<thead>
<tr>
<th>Interrupt Factor</th>
<th>EI(\text{OS})-Corresponded</th>
<th>Interrupt vector</th>
<th>Interrupt Control Register</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>✗</td>
<td>#08 08H</td>
<td>FFFFDC(\text{H}) -</td>
<td>High</td>
</tr>
<tr>
<td>INT9 instruction</td>
<td>✗</td>
<td>#09 09H</td>
<td>FFFFFD(\text{H}) -</td>
<td></td>
</tr>
<tr>
<td>Exception processing</td>
<td>✗</td>
<td>#10 0AH</td>
<td>FFFFFD(\text{H}) -</td>
<td></td>
</tr>
<tr>
<td>8-/10-bit A/D converter</td>
<td>✗</td>
<td>#11 0B(\text{H})</td>
<td>FFFFFD(\text{H})</td>
<td></td>
</tr>
<tr>
<td>Timebase time</td>
<td>✗</td>
<td>#12 0C(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>External interrupt 0 (DTP0)/External interrupt 1 (DTP1)</td>
<td>✗</td>
<td>#13 0D(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>16-bit free-run timer 0 overflow</td>
<td>✗</td>
<td>#14 0E(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>I/O Extended serial interface 1</td>
<td>✗</td>
<td>#15 0F(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>Wake-up interrupted</td>
<td>✗</td>
<td>#16 10(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>I/O Extended serial interface 2</td>
<td>✗</td>
<td>#17 11(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>External interrupt 2 (DTP2)/External interrupt 3 (DTP3)</td>
<td>✗</td>
<td>#18 12(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>8-/16-bit PPG Timer 0 counter borrow</td>
<td>✗</td>
<td>#19 13(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>External interrupt 4 (DTP4)/External interrupt 5 (DTP5)</td>
<td>✗</td>
<td>#20 14(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>8-/16-bit up/down counter/Timer 0 compare match</td>
<td>✗</td>
<td>#21 15(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>8-/16-bit up/down counter/Timer 0 overflow, up/down reverse</td>
<td>✗</td>
<td>#22 16(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>8-/16-bit PPG Timer 1 counter borrow</td>
<td>✗</td>
<td>#23 17(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>External interrupt 6 (DTP6)/External interrupt 7 (DTP7)</td>
<td>✗</td>
<td>#24 18(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>Output compare (unit 1) ch4, ch5 match</td>
<td>✗</td>
<td>#25 19(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>Watch timer</td>
<td>✗</td>
<td>#26 1A(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>Output compare (unit 1) ch6, ch7 match</td>
<td>✗</td>
<td>#27 1B(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
<tr>
<td>16-bit free-run timer 1 overflow</td>
<td>✗</td>
<td>#28 1C(\text{H})</td>
<td>FFFFC(\text{H})</td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.5-2 Interruption factors, interruption vectors, and interruption control register (2 / 2)

<table>
<thead>
<tr>
<th>Interrupt Factor</th>
<th>Ei²OS-Corresponded</th>
<th>Interrupt vector</th>
<th>Interrupt Control Register</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Number</td>
<td>Address</td>
<td>ICR</td>
</tr>
<tr>
<td>8-/16-bit up/down counter / Timer 1 compare match</td>
<td>O</td>
<td>#29</td>
<td>1D_H</td>
<td>FFFF88H</td>
</tr>
<tr>
<td>8-/16-bit up/down counter/ Timer 1 overflow, up/down reverse</td>
<td>O</td>
<td>#30</td>
<td>1E_H</td>
<td>FFFF84H</td>
</tr>
<tr>
<td>Input capture 0 fetched</td>
<td>O</td>
<td>#31</td>
<td>1F_H</td>
<td>FFFF80H</td>
</tr>
<tr>
<td>Input capture 1 fetched</td>
<td>O</td>
<td>#32</td>
<td>20_H</td>
<td>FFFF7CH</td>
</tr>
<tr>
<td>Output compare (unit 0) ch0 match</td>
<td>O</td>
<td>#33</td>
<td>21_H</td>
<td>FFFF78H</td>
</tr>
<tr>
<td>Output compare (unit 0) ch1 match</td>
<td>O</td>
<td>#34</td>
<td>22_H</td>
<td>FFFF74H</td>
</tr>
<tr>
<td>Output compare (unit 0) ch2 match</td>
<td>O</td>
<td>#35</td>
<td>23_H</td>
<td>FFFF70H</td>
</tr>
<tr>
<td>Output compare (unit 0) ch3 match</td>
<td>O</td>
<td>#36</td>
<td>24_H</td>
<td>FFFF6CH</td>
</tr>
<tr>
<td>UART receive completed</td>
<td>⊗</td>
<td>#37</td>
<td>25_H</td>
<td>FFFF68H</td>
</tr>
<tr>
<td>16-bit reload timer 0</td>
<td>O</td>
<td>#38</td>
<td>26_H</td>
<td>FFFF64H</td>
</tr>
<tr>
<td>UART transmit completed</td>
<td>⊗</td>
<td>#39</td>
<td>27_H</td>
<td>FFFF60H</td>
</tr>
<tr>
<td>16-bit reload timer 1</td>
<td>O</td>
<td>#40</td>
<td>28_H</td>
<td>FFFF5CH</td>
</tr>
<tr>
<td>Flash memory</td>
<td>×</td>
<td>#41</td>
<td>29_H</td>
<td>FFFF58H</td>
</tr>
<tr>
<td>Delayed interrupt generation module</td>
<td>×</td>
<td>#42</td>
<td>2A_H</td>
<td>FFFF54H</td>
</tr>
</tbody>
</table>

O: Can be used  ×: Cannot be used  ⊗: Can be used, with the Ei²OS stop function

*: The priority is given when plural interrupts with the same level are generated simultaneously.

- Peripheral functions that share the same ICR register have the same interruption level.
- When two peripheral functions share an ICR register, only one can use the Ei²OS.
- When two peripheral functions share an ICR register and one specifies the Ei²OS, the remaining peripheral functions cannot use the interrupt.
3.5.2 Interrupt Control Registers and Peripheral Functions

The interrupt control registers (ICR00 to ICR15) are located in the interruption controller and support all peripheral functions with an interruption function. The registers control the interrupt and extended intelligent I/O service (EI²OS).

■ Interrupt Control Register List

Table 3.5-3 lists the peripheral functions corresponding to the interrupt control registers.

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers</th>
<th>Abbreviation</th>
<th>Corresponding peripheral function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000B0H</td>
<td>Interrupt control register 00</td>
<td>ICR00</td>
<td>8-/10-bit A/D converter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Timebase timer</td>
</tr>
<tr>
<td>0000B1H</td>
<td>Interrupt control register 01</td>
<td>ICR01</td>
<td>External interrupt 0, 1 (DTP0, 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16-bit free-run timer 0</td>
</tr>
<tr>
<td>0000B2H</td>
<td>Interrupt control register 02</td>
<td>ICR02</td>
<td>I/O Extended serial interface 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Wake-up interrupted</td>
</tr>
<tr>
<td>0000B3H</td>
<td>Interrupt control register 03</td>
<td>ICR03</td>
<td>I/O Extended serial interface 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>External interrupt 2, 3 (DTP2, 3)</td>
</tr>
<tr>
<td>0000B4H</td>
<td>Interrupt control register 04</td>
<td>ICR04</td>
<td>8-/16-bit PPG timer 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>External interrupt 4, 5 (DTP4, 5)</td>
</tr>
<tr>
<td>0000B5H</td>
<td>Interrupt control register 05</td>
<td>ICR05</td>
<td>8-/16-bit up/down counter/timer 0</td>
</tr>
<tr>
<td>0000B6H</td>
<td>Interrupt control register 06</td>
<td>ICR06</td>
<td>8-/16-bit PPG timer 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>External interrupt 6, 7 (DTP6, 7)</td>
</tr>
<tr>
<td>0000B7H</td>
<td>Interrupt control register 07</td>
<td>ICR07</td>
<td>16-bit I/O timer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Watch timer</td>
</tr>
<tr>
<td>0000B8H</td>
<td>Interrupt control register 08</td>
<td>ICR08</td>
<td>16-bit I/O timer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16-bit free-run timer 1</td>
</tr>
<tr>
<td>0000B9H</td>
<td>Interrupt control register 09</td>
<td>ICR09</td>
<td>8-/16-bit up/down counter/timer 1</td>
</tr>
<tr>
<td>0000BAH</td>
<td>Interrupt control register 10</td>
<td>ICR10</td>
<td>16-bit I/O timer</td>
</tr>
<tr>
<td>0000BBH</td>
<td>Interrupt control register 11</td>
<td>ICR11</td>
<td>16-bit I/O timer</td>
</tr>
<tr>
<td>0000BCH</td>
<td>Interrupt control register 12</td>
<td>ICR12</td>
<td>16-bit I/O timer</td>
</tr>
<tr>
<td>0000BDH</td>
<td>Interrupt control register 13</td>
<td>ICR13</td>
<td>UART</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16-bit reload timer 0</td>
</tr>
<tr>
<td>0000BEH</td>
<td>Interrupt control register 14</td>
<td>ICR14</td>
<td>UART</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16-bit reload timer 1</td>
</tr>
<tr>
<td>0000BFH</td>
<td>Interrupt control register 15</td>
<td>ICR15</td>
<td>Flash memory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Delayed interrupt generation module</td>
</tr>
</tbody>
</table>
Interrupt control register (ICR) has following four functions below.

The interrupt control register (ICR) functions associated with writing differ from those associated with reading.

- The interruption levels of the corresponding peripheral functions can be specified.
- Interruption setting for the corresponding peripheral functions can be selected between ordinary interruptions and extended intelligent I/O services.
- Channel of extended intelligent I/O service (EI^2OS) can be selected.
- Completed state of extended intelligent I/O service (EI^2OS) can be displayed.

**Note:** The interrupt control register (ICR) should not access via the read-modify-write command.
### Interrupt Control Register (ICR00 to ICR15)

The functions of the interrupt control registers are shown below.

#### Interrupt Control Register (ICR00 to ICR15)

The interrupt control register functions associated with writing differ from those associated with reading.

**Figure 3.5-2 Interrupt Control Register (ICR00 to ICR15) during writing**

<table>
<thead>
<tr>
<th>IL2</th>
<th>IL1</th>
<th>IL0</th>
<th>ISE</th>
<th>EFOs enbale bit</th>
<th>Interrupt level set bit</th>
<th>Interrupt level (No interrupt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table: EFOs Channel select bit**

<table>
<thead>
<tr>
<th>Channel Descriptor Address</th>
<th>Channel</th>
<th>EFOs Descriptor Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>000100h</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>000108h</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>000110h</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>000118h</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>000120h</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>000128h</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>000130h</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>000138h</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>000140h</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>000148h</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>000150h</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>000158h</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>000160h</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>000168h</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>000170h</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>000178h</td>
</tr>
</tbody>
</table>

**Legend:**
- **R/W**: Readable and Writable
- **W**: Write only
- **Reset value**
### Figure 3.5-3  Interrupt Control Register (ICR00 to ICR15) during reading

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>IL2</th>
<th>IL1</th>
<th>IL0</th>
<th>Interrupt level setting bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Interrupt level 0 (Strongest)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Interrupt level 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Interrupt level 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Interrupt level 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Interrupt level 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Interrupt level 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Interrupt level 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Interrupt level 7 (No interrupt)</td>
</tr>
</tbody>
</table>

**R/W**: Readable and Writable  
**W**: Write only  
**–**: Unused  
**X**: Undefined  
**X**: Reset value

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 3</th>
<th>ISE</th>
<th>El2OS status bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>When El2OS in operation or not started</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Stopped status due to count termination</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Stopped status due to a request from the peripheral function</td>
</tr>
</tbody>
</table>

**ISE**: El2OS enable bit  
0: Activating the normal interrupt processing at generating interruption  
1: Activating El2OS at generating interruption
3.5.4 Function of Interrupt Control Register

Interrupt Control Register (ICR00 to ICR15) can be set the following four functions below.
- Interrupt level set bits (IL2 to IL0)
- EI²OS enable bit (ISE)
- EI²OS channel selection bits (ICS3 to ICS0)
- EI²OS status bits (S1 and S0)

■ Bit Configuration of Interrupt Control Register (ICR)

Figure 3.5-4 shows the bit configuration of interrupt control register (ICR)

![Configuration of Interrupt control register (ICR) writing](image)

![Configuration of Interrupt control register (ICR) reading](image)

Reference:
- The setting of the channel selection bits (ICR: ICS3 to ICS0) is enabled only when starting the EI²OS. When EI²OS is activated, set the EI²OS enable bit (ICR:ISE) to "1". When it is not activated, set the EI²OS enable bit to "0".
- The channel select bits (ICR:ICS3 to 0) are only valid when writing is performed. The EI²OS status bits (ICR:S1, S0) are only valid when reading is performed.
### Function of Interrupt Control Register

- **Interrupt level set bits (IL2 to IL0)**
  
  The interruption levels of peripheral functions can be specified. A reset results in an initializes to level 7 (IL2 to 0 = "111B": no interruptions).
  
  Table 3.5-4 shows the relationships between the interrupt level set bits and interruption levels.

#### Table 3.5-4 Relationship between interrupt level and interrupt level setting bits

<table>
<thead>
<tr>
<th>IL2</th>
<th>IL1</th>
<th>IL0</th>
<th>Interrupt level</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 (Priority: highest)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6 (Priority: lowest)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7 (No interrupt)</td>
</tr>
</tbody>
</table>

- **EI^2^OS enable bit (ISE)**
  
  If an interruption occurs when the interrupt control register ISE bit (ICR:ISE) is set to "1", an EI^2^OS is activated. If an interruption occurs when the interrupt control register ISE bit is set to "0", the normal interruption processing is activated. If a condition that causes the completion of the EI^2^OS (i.e. when the status bits S1 and S0 become other than "00B") occurs, the ISE bit is cleared to "0". Even when the peripheral functions do not have an EI^2^OS function, make sure that the interrupt control register ISE bit (ICR:ISE) is set to "0" by the program. The ISE bit is initialized to "0" upon a reset.

- **EI^2^OS channel select bits (ICS3 to ICS0)**

  EI^2^OS channel can be set. The EI^2^OS descriptor address can be specified through channel select bits (ICR:ICS3 to 0) setting. It will be initialized to "0000B" upon a reset.

  Table 3.5-5 shows the correspondence between the EI^2^OS channel selection bits and descriptor addresses.

#### Table 3.5-5 Correspondence between EI^2^OS Channel Selection Bits and Descriptor Addresses (1/2)

<table>
<thead>
<tr>
<th>ICS3</th>
<th>ICS2</th>
<th>ICS1</th>
<th>ICS0</th>
<th>Channel to be selected</th>
<th>Descriptor address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>000100H</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>000108H</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>000110H</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>000118H</td>
</tr>
</tbody>
</table>
CHAPTER 3  CPU FUNCTION

### Table 3.5-5  Correspondence between EI\(^2\)OS Channel Selection Bits and Descriptor Addresses (2 / 2)

<table>
<thead>
<tr>
<th>ICS3</th>
<th>ICS2</th>
<th>ICS1</th>
<th>ICS0</th>
<th>Channel to be selected</th>
<th>Descriptor address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>000120(_H)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>000128(_H)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>000130(_H)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>000138(_H)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>000140(_H)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>000148(_H)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>000150(_H)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>11</td>
<td>000158(_H)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>000160(_H)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>13</td>
<td>000168(_H)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>14</td>
<td>000170(_H)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>15</td>
<td>000178(_H)</td>
</tr>
</tbody>
</table>

- **EI\(^2\)OS status bits (S1 and S0)**

  By reading this after completion of an EI\(^2\)OS, the operation and completion statuses can be confirmed. It will be initialized to “00\(_B\)” upon a reset.

Table 3.5-6  shows the relationship between the EI\(^2\)OS status bits (ICR: S1, S0) and the EI\(^2\)OS status.

### Table 3.5-6  Relationships Between EI\(^2\)OS Status Bits and EI\(^2\)OS Status

<table>
<thead>
<tr>
<th>S1</th>
<th>S0</th>
<th>EI(^2)OS Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>When EI(^2)OS in operation or not started</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Stopped status due to count termination</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Stopped status due to a request from the peripheral function</td>
</tr>
</tbody>
</table>
3.5.5 Hardware Interrupt

Hardware interrupt temporarily stop the current program processing and execute the user-defined interruption processing program if an interruption request is issued by a peripheral function.

Also corresponding to extended intelligent I/O service (EI²OS).

## Hardware Interrupt

- **Function of hardware interrupt**

  If a request to perform a hardware interrupt is issued by a peripheral function, the interruption level (IR:IL) is compared with the interruption level mask register (PS:ILM) and the interruption permission flag (CCR:I) is confirmed to determine whether the interruption can be accepted.

  If a hardware interrupt is accepted, the register in the CPU is automatically evacuated to the system stack and the interruption level of the accepted interruption is stored in the interruption level mask register (ILM), before which a branching to the interruption vector takes place.

- **Multiple interrupts**

  Multiple hardware interrupts can be activated.

- **Extended intelligent I/O service (EI²OS)**

  Upon completion of an EI²OS, the normal interruption processing is performed. EI²OS can only be activated one at a time. During the EI²OS processing, all other interrupt requests and EI²OS requests are kept on hold.

- **External interrupt**

  External interruptions (including wake-up interruptions) are treated as hardware interrupt when they are accepted.

- **Interrupt**

  The interruption vector table referred to perform interruption processing is deployed in the "FFFF00H" to "FFFFF0H" in memory and is used for software interruptions as well.
CHAPTER 3  CPU FUNCTION

■ Hardware Interrupt

For a hardware interrupt, the following 4 parts are used.

Before activating a hardware interrupt, make settings for the following 4 parts by using the program.

Table 3.5-7 Mechanism Related to Hardware Interrupt

<table>
<thead>
<tr>
<th>Mechanism Related to Hardware Interrupt</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peripheral function</td>
<td></td>
</tr>
<tr>
<td>Interrupt enable bit, interrupt request bit</td>
<td>Controls interrupt request from peripheral function</td>
</tr>
<tr>
<td>Interrupt controller</td>
<td></td>
</tr>
<tr>
<td>Interrupt control register (ICR)</td>
<td>Interrupt level set up and the EI²OS control</td>
</tr>
<tr>
<td>CPU</td>
<td></td>
</tr>
<tr>
<td>Interrupt enable flag (I)</td>
<td>Identifies interrupt enable state</td>
</tr>
<tr>
<td>Interrupt level mask register (ILM)</td>
<td>Compare interrupt request level and current interrupt level</td>
</tr>
<tr>
<td>FFFC00H to FFFFFFH in memory</td>
<td>Interrupt vector table</td>
</tr>
</tbody>
</table>

■ Hardware Interrupt Inhibition

When the following condition exists, no hardware interrupt can be accepted.

- Hardware Interrupt Inhibition during the Writing to Peripheral Function Control Register in I/O area.

During the execution of a command to write to a peripheral function control register, no hardware interrupts are accepted. This is necessary to prevent the CPU from malfunctioning during its interruption processing for an interruption request issued while an interrupt control register for each resource is being rewritten.

Figure 3.5-5 shows the operation of a hardware interrupt during execution of a command to write to a peripheral function control register.

Figure 3.5-5 Hardware interruption request during execution of a command to write to a peripheral function control register

MOV   A,#08
MOV   io,A
MOV   A,2000H

Transit to interrupt processing

Not transit to interrupt processing

Generating interrupt request at this point
Suppression of hardware interruptions during execution of an interruption suppression command

Table 3.5-8 shows the hardware interrupt inhibit instructions.

If a hardware interruption occurs during execution of a hardware interruption suppression command, the interruption processing is performed after the hardware interruption suppression command is processed and the following command is executed.

<table>
<thead>
<tr>
<th>Instruction not reception of interrupt request</th>
<th>Interrupt Inhibit Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCB</td>
<td>MOV ILM,#imm8</td>
</tr>
<tr>
<td>DTB</td>
<td>OR CCR,#imm8</td>
</tr>
<tr>
<td>ADB</td>
<td>AND CCR,#imm8</td>
</tr>
<tr>
<td>SPB</td>
<td>POPW PS</td>
</tr>
<tr>
<td>CMR</td>
<td></td>
</tr>
<tr>
<td>NCC</td>
<td></td>
</tr>
</tbody>
</table>

Hardware interrupt inhibition during execution of software interrupt

If a software interrupt is activated, the interruption permission flag (CCR:I) is cleared to "0" to prohibit interruptions.
3.5.6 Operation of Hardware Interrupt

The operation from the generation of hardware interrupt request to the completion of interrupt processing is explained below.

■ Start of Hardware Interrupt

● Operation of peripheral function (generation of interrupt request)

Peripheral functions with the hardware interruption request function have an "interrupt request flag" that indicates whether an interruption request has been issued and an "interrupt permission bit" that specifies whether interruption requests are to be permitted or prohibited. The interruption request flag is set by the occurrence of an event unique to a peripheral function. If the current interruption permission bit setting is "permit", an interruption request occurs for CPU interruption controller.

● Operation of the interruption controller (interrupt request control)

The CPU interruption controller compares the interruption levels of the interruption requests that occurred simultaneously (ICR:IL2 to 0), selects the highest level request (the one with the smallest IL setting value) and issues an interruption request to the CPU. If there are two or more requests of the same level, they are handled in the ascending order according to their smallest interruption numbers.

● Operation of CPU (interrupt request acceptance and interrupt processing)

The CPU compares the interruption level (ICR:IL2 to 0) value received from the interruption controller with the interruption level mask register (ILM) value. If the interruption level (IL) value is smaller than the interruption level mask register (ILM) value and interruption is permitted (CCR:I=1), the interruption processing micro-code is activated after completion of the command currently being executed.

The EI²OS permission bit (ICR:ISE) is referred at the beginning of the interruption processing micro-code. If the EI²OS permission bit (ICR:ISE) is "0", the normal interruption processing is performed. If the EI²OS permission bit (ICR:ISE) is "1", the EI²OS is performed.

In the interruption processing, the contents of the dedicated registers (A, DPR, ADB, DTB, PCB, PC and PS) are first evacuated to the system stack (i.e. the system stack space specified by SSB and SSP).

After this, the interruption vector address of the vector table that corresponds to the interruption occurred is loaded into the program counter (PCB, PC), the interruption level mask register (ILM) data is renewed and then the stack flag (CCR:S) is set to "1".

■ Return from Hardware Interrupt

In case of clearing of the interruption request flag of the peripheral function that constituted the interruption factor to "0" and executing the RETI command in the interruption processing program, the dedicated registers data evacuated to the system stack is retransferred to the original locations and the operation is resumed from the point that is subsequent to the command that was being executed before the shift to the interruption processing. (Return of interruption processing)

By clearing the interruption request flag, an interruption request that has been output to the interruption controller by a peripheral function can be cancelled.
- **Operation of Hardware Interrupt**

  Figure 3.5-6 shows the operation from the generation of hardware interrupt to the completion of interrupt processing.

**Figure 3.5-6 Operation of Hardware Interrupt**

1. A peripheral function issues an interruption request.
2. If the interruption permission bit in that peripheral function has been set by this point to permit interruptions, an interruption request is output to the interruption controller.
3. After receiving the interruption request, the interruption controller determines the priorities of the interruptions that have been requested simultaneously and notifies the CPU of the interruption level (IL) of the interruption requests to be accepted.
4. The CPU compares the interruption level (IL) of the interruptions requested by the interruption controller with the interruption level mask register (ILM) values.
5. If an interruption request is higher priority than the interruption level mask register (ILM) value, the interruption permission flag (CCR:I) is checked.
6. If the interruption permission flag has been set by this point to permit interruptions (CCR:I=1), the interruption level mask register (ILM) is set to the interruption level (IL) of the interruption that has been requested after completion of the command processing currently being performed.
7. The dedicated registers contents are evacuated and the interruption processing program is executed.
8. If the interruption request flag is cleared to "0" and an interruption return command (RETI) is executed in the interruption processing program, the interruption processing is terminated.
3.5.7 Procedure for Use of Hardware Interrupt

When hardware interrupts are to be used, it is necessary to make system stack area, peripheral function and interrupt control register (ICR) settings.

**Hardware Interrupt**

Figure 3.5-7 shows an example procedure for a case where hardware interrupts are used.

1. Set the system stack area.
2. Make settings for interruptions by the peripheral function(s) that has the interruption request function.
3. Set the interrupt control register (ICR in the interruption controller).
4. Make the peripheral function(s) ready for operation and set the interruption permission bit to "permit".
5. Set the interruption level mask register (ILM) and interruption permission flag (CCR:I) to accept interruptions (CCR:I=1).
6. If an interruption by a peripheral function occurs, a hardware interruption request to the CPU is issued.
7. The dedicate registers are evacuated in the interruption controller and a branching to the interruption processing program takes place.
8. In the interruption processing, the program for the interruption is executed.
9. Clear the interruption request flag of the peripheral function to "0" in the interruption processing program.
10. Execute an interruption return command (RETI). Returns to the program that is subsequent to the branching.
3.5.8  Multiple Interrupts

With regard to hardware interrupts, multiple interrupts can be achieved by specifying different interruption levels for the interruption level setting bits (ICR:IL2 to IL0) when two or more interruption requests from peripheral functions occur. However, extended intelligent I/O services can only be activated one at a time.

Multiple interrupts

- Multiple Interrupts

If a request for an interruption of a level that is higher than the level of the current interruption occurs during the interruption processing, the current interruption processing is stopped temporarily and the higher level interruption request is accepted. After completion of the interruption processing for the higher level interruption, the suspended interruption processing is resumed.

Interruption level (IL) of 0 to 7 can be specified, but no interruption request that has been set to level 7 can be accepted.

If a request for an interruption of a level that is equivalent to or lower than the level of the current interruption occurs during the interruption processing, the new interruption request is kept on hold until the current interruption processing is completed.

If the interruption level mask register (ILM) is set to "interruption prohibited" (ILM = "000B") or the interruption permission flag (CCR:I) is set to "interruption prohibited" (CCR:I=0) while interruption processing is being performed, activation of multiple interruptions during the interruption processing can be temporarily prohibited.

Note: Multiple EI²OS cannot be started. During EI²OS processing, other interrupt requests and other EI²OS requests are all held.
Example of multiple interrupts

As an example of multiple interruption processing, a case in which timer interruptions are given higher priority than A/D converter interruptions is explained. Figure 3.5-8 shows the processing that takes place when a timer interruption occurs during the processing for an A/D converter interruption when the A/D converter interruption level setting is 2 (IL= "010B") and the timer interruption level setting is 1 (IL= "001B").

Figure 3.5-8 Example of multiple interrupts

- When the processing for the A/D converter interruption is started, the interruption level mask register (ILM) is set to the A/D converter interruption level (ICR:IL2 to IL0) value (2 in this example). If a request for an interruption of level 1 or level 0 (a timer interruption request in this example) occurs after this, timer interruption processing is given higher priority.

- When the interruption processing is completed and an interruption processing return command (RETI) is executed, the values of the dedicated registers (A, DPR, ADB, DTB, PCB, PC, PS) that have been evacuated to the system stack are replaced into the registers and the value of the interruption level mask register (ILM) is set back to the value before the suspension of the processing (2 in this example).
3.5.9 Software Interrupt

Software interrupts are a function that allows a user-defined interruption processing program to be executed with a software interruption command from a main program being executed. During a software interruption, all hardware interrupt requests are kept on hold.

■ Start and operation of software interrupt

● Start of Software Interrupt

A software interrupt is activated by executing an INT command. There are no interruption request flag or interruption permission flag for software interrupts. The execution of an INT command immediately generates an interruption request.

● Hardware interrupt inhibition

INT command-based interruptions have no interruption level. The interruption level mask register (ILM) will not be renewed. During execution of an INT command, the interruption permission flag (CCR:I) is cleared to "0" and hardware interruptions are masked.

To permit hardware interrupts during software interruption processing, set the interruption permission flag (CCR:I) to "1" in the software interruption processing program.

● Operation of software interrupt

The execution of an INT command activates the micro-code for software interruption processing. The micro-code for software interruption processing evacuates the contents of the dedicated registers to the system stack, masks hardware interruptions (CCR:I=0) and then causes a branching to the interruption vector table interruption address.

■ Return from Software Interrupt

If an interruption processing return command (RETI) is executed in the interruption processing program, the dedicated registers values that have been evacuated to the system stack are replaced to their original locations and a return to the command that succeeds the interruption processing branching takes place.

Note: When the program bank register (PCB) is FFH, the vector area for the CALLV instruction overlaps the table for the INT #vc8 instruction. A CALLV and INT #vc8 instructions cannot use the same address in creating a software.
3.5.10 Interrupts by Extended Intelligent I/O Service (EI²OS)

Extended Intelligent I/O Services (EI²OS) are a function that automatically transfers data between the peripheral function (I/O) and RAM. After completion of the data transfer, hardware interruptions will occur.

■ Extended intelligent I/O service (EI²OS)

Extended Intelligent I/O Services (EI²OS) are a function that automatically transfers data between the I/O area and RAM. Upon completion of the data transfer(s), the factor for the completion (completion condition) is set to the status bit in the interrupt control register (ICR:S1,S0) and an automatic branching to the interruption processing routine takes place. You can have data transfers performed just by creating in advance a completion program and a setting program for the EI²OS activation.

● Advantages of EI²OS

The advantages over interruption processing routine-based data transfers are as follows:

- Since the creation of transfer program is not required, the program size can be reduced.
- Because the number of transfers can be specified, unnecessary data is not transferred.
- Because you can specify whether the buffer address pointer is to be renewed or not, all you have to do is make settings for the first and last when the addresses are continuous.
- Because you can specify whether the I/O address pointer is to be renewed or not, all you have to do is make settings for the first and last when the addresses are continuous.

● Interrupt by EI²OS termination

Upon completion of the EI²OS data transfer(s), the completion condition is set to the EI²OS status bit in the interrupt control register (ICR:S1,S0) and an automatic shift to the interruption processing takes place.

The factor for an EI²OS completion can be confirmed by checking the EI²OS status bit (ICR:S1,S0) by the interruption processing program.

● Interrupt control register (ICR)

This is located in the interruption controller and is used for EI²OS activation, EI²OS channel setting and EI²OS completion status display.

● EI²OS descriptor (ISD)

This is an 8-byte data register that is deployed in the "000100H" to "00017FH" area of built-in RAM and used to specify the transfer mode, address, number of bytes to be transferred and buffer address. It has 16 channels, and a channel number is allocated to each of these channels by the interrupt control register (ICR: ICS3 to ICS0).

Note:
- The CPU stops while the EI²OS is in operation.
- When using REALOS, extended intelligent I/O service (EI²OS) can not be used.
### Operation of EI^2^OS

Figure 3.5-9 shows the operation of the EI^2^OS.

1. If an interruption request occurs, the EI^2^OS will be activated.
2. The interruption controller selects the EI^2^OS descriptor.
3. The address pointers for the transfer origination and destination are read from the EI^2^OS descriptor.
4. The data is transferred based on the address pointers for the transfer origination and destination.
5. An interrupt factor is cleared automatically.
3.5.11 EI²OS Descriptor (ISD)

EI²OS descriptor (ISD) which is in "000100H" to "00017FH" of built-in RAM is consists of 8-byte × 16 channels.

Configuration of EI²OS Descriptor (ISD)

An EI²OS descriptor (ISD) comprises 8-byte × 16 channels. Figure 3.5-10 shows ISD configurations. Table 3.5-9 shows the correspondence between the channel number and ISD address.

Figure 3.5-10 Configuration of EI²OS Descriptor (ISD)

Table 3.5-9 EI²OS Descriptor (ISD) Area (1/2)

<table>
<thead>
<tr>
<th>Channel (ICR: ICS3 to ICS0)</th>
<th>Descriptor Starting Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (0000B)</td>
<td>000100H</td>
</tr>
<tr>
<td>1 (0001B)</td>
<td>000108H</td>
</tr>
<tr>
<td>2 (0010B)</td>
<td>000110H</td>
</tr>
<tr>
<td>3 (0011B)</td>
<td>000118H</td>
</tr>
<tr>
<td>4 (0100B)</td>
<td>000120H</td>
</tr>
<tr>
<td>5 (0101B)</td>
<td>000128H</td>
</tr>
<tr>
<td>6 (0110B)</td>
<td>000130H</td>
</tr>
<tr>
<td>7 (0111B)</td>
<td>000138H</td>
</tr>
<tr>
<td>8 (1000B)</td>
<td>000140H</td>
</tr>
</tbody>
</table>
### Table 3.5-9  El²OS Descriptor (ISD) Area (2 / 2)

<table>
<thead>
<tr>
<th>Channel (ICR: ICS3 to ICS0)</th>
<th>Descriptor Starting Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>9 (1001B)</td>
<td>000148H</td>
</tr>
<tr>
<td>10 (1010B)</td>
<td>000150H</td>
</tr>
<tr>
<td>11 (1011B)</td>
<td>000158H</td>
</tr>
<tr>
<td>12 (1100B)</td>
<td>000160H</td>
</tr>
<tr>
<td>13 (1101B)</td>
<td>000168H</td>
</tr>
<tr>
<td>14 (1110B)</td>
<td>000170H</td>
</tr>
<tr>
<td>15 (1111B)</td>
<td>000178H</td>
</tr>
</tbody>
</table>
3.5.12 Each Register of EI²OS Descriptor (ISD)

Extended intelligent I/O service (EI²OS) descriptor (ISD) is configured by following registers.
- Data counter (DCT)
- I/O address pointer (IOA)
- EI²OS status register (ISCS)
- Buffer address pointer (BAP)

Note that if the reset setting is made, the register reset values will become indeterminate.

■ Data counter (DCT)

Data counter (DCT) is 16-bit register. Specifies the number of bytes of transfer data. When a data transfer is made, 1 is subtracted. When the data counter (DCT) value becomes "0000H", the EI²OS is completed and the interruption processing program is executed.

Figure 3.5-11 shows the configuration of the data counter (DCT).

![Figure 3.5-11 Configuration of Data Counter (DCT)](image)

■ I/O address pointer (IOA)

I/O address pointer (IOA) is 16-bit register. Specifies the lower-order address (A15 to A0) for the data transfer. The higher-order address (A23 to A16) is set to "00H". The area from "000000H" to "00FFFFH" can be used when specifying an address.

Figure 3.5-12 shows the configuration of I/O address pointer (IOA).

![Figure 3.5-12 Configuration of I/O Address Pointer (IOA)](image)
**EI²OS Status Register (ISCS)**

EI²OS status register (ISCS) is an 8-bit register. The methods to renew the buffer address pointer and I/O address pointer, the transfer data type (byte/word) and the transfer direction can be specified.

Figure 3.5-13 shows the bit configuration of the EI²OS status register (ISCS).

### Figure 3.5-13 Configuration of EI²OS Status Register (ISCS)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reset value</td>
<td>XXXXXX X B</td>
</tr>
<tr>
<td>1</td>
<td>SE</td>
<td>EF²OS termination control bit</td>
</tr>
<tr>
<td>2</td>
<td>DIR</td>
<td>Data transfer direction specification bit</td>
</tr>
<tr>
<td>3</td>
<td>BF</td>
<td>BAP update/fix selection bit</td>
</tr>
<tr>
<td>4</td>
<td>BW</td>
<td>Transfer data length specification bit</td>
</tr>
<tr>
<td>5</td>
<td>IF</td>
<td>IOA update/fix selection bit</td>
</tr>
<tr>
<td>6-7</td>
<td>Reserved</td>
<td>Reserved bit</td>
</tr>
</tbody>
</table>

- **R/W**: Readable and Writable
- **X**: Undefined
- **B**: Buffer address pointer varies only in the lower 16 bits and can be incremented.
- **I/O**: I/O address pointer allows only incrementing.
- **若**: Always set this bit to “0”.

---

- **bit0**
  - SE: EF²OS termination control bit
    - 0: Don’t terminate by the termination request from a peripheral resource.
    - 1: Terminate by the termination request from a peripheral resource.

- **bit1**
  - DIR: Data transfer direction specification bit
    - 0: I/O address pointer -> transfer to Buffer address pointer
    - 1: Buffer address pointer -> transfer to I/O address pointer

- **bit2**
  - BF: BAP update/fix selection bit
    - 0: Update Buffer address pointer after completion transfer.
    - 1: Don’t update Buffer address pointer after completion transfer.

- **bit3**
  - BW: Transfer data length specification bit
    - 0: Byte
    - 1: Word

- **bit4**
  - IF: IOA update/fix selection bit
    - 0: Update I/O address pointer after completion transfer.
    - 1: Don’t update I/O address pointer after completion transfer.

- **bit5-7**
  - Reserved bit
    - 0 0 0: Always set this bit to “0”.
Buffer address pointer (BAP)

Buffer address pointer (BAP) is a 24-bit register. EI²OS operation sets the memory address of the data transferring source. Buffer address pointer exists in each channel. So, the data can be transferring between the 16M-byte memory address and the peripheral function (resource) address. If the BAP updated/fixed setting bit (BF) is set to "0", the lower 16-bit (BAPM, BAPL) is incremented, and the higher 8-bit (BAPH) is not incremented.

![Figure 3.5-14 Configuration of Buffer Address Pointer (BAP)](image)

### Reference:
- The area that can be set by the I/O address pointer (IOA) is "000000H" to "00FFFFH".
- The area that can be set by the buffer address pointer (BAP) is "000000H" to "FFFFFFFH".
- The maximum transfer count that can be set by the data counter (DCT) is 65,536.
3.5.13 Operation of EI²OS

The flowchart of operation of the EI²OS is shown below:

### Operation of EI²OS

![Flowchart of Operation of EI²OS](image)

**Figure 3.5-15 Flowchart of Operation of EI²OS**

- **ISE=1?**
  - NO: Interrupt request generation of Peripheral function
  - YES: ISD/ISCS Read

- **With finish request from peripheral functions?**
  - NO: Address is set to IOA (data transfer)
  - YES: Address is set to BAP

- **DIR=1?**
  - NO: Address is set to IOA (data transfer)
  - YES: Address is set to BAP (data transfer)

- **IF=0?**
  - NO: Renewal value depends on BW
  - YES: IOA Update

- **BF=0?**
  - NO: Renewal value depends on BW
  - YES: BAP Update

- **DCT decrement**
  - **DCT=“00H”?**
    - YES: EI²OS End processing
    - NO: Set S1, S0 to “00s”

- **Clear Peripheral function interrupt request**

- **CPU Operation return**

- **Clear ISE to “0”**

- **Interuption processing**
3.5.14 Procedure for Use of EI²OS

The procedure for using the EI²OS is shown below:

### Procedure for Use of EI²OS

<table>
<thead>
<tr>
<th>Processing by Software</th>
<th>Processing by Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>System stack area setting</td>
<td></td>
</tr>
<tr>
<td>EI²OS descriptor setting</td>
<td></td>
</tr>
<tr>
<td>Peripheral function interruption setting</td>
<td></td>
</tr>
<tr>
<td>Interrupt control register (ICR) setting</td>
<td></td>
</tr>
<tr>
<td>Peripheral function operation start setting and Interruption enable bit setting</td>
<td></td>
</tr>
<tr>
<td>Setting of ILM, I inside PS</td>
<td></td>
</tr>
<tr>
<td>Execution of user program</td>
<td>(Interruption request) and (ISE=1)</td>
</tr>
<tr>
<td>Data transfer</td>
<td></td>
</tr>
<tr>
<td>Determining whether interrupt branching was performed because of count-out or because of a completion request from the resource</td>
<td></td>
</tr>
<tr>
<td>YES</td>
<td>S1, S0=&quot;01s&quot; or S1, S0=&quot;11s&quot;</td>
</tr>
<tr>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>Expansion intelligent I/O service resetting (channel changing)</td>
<td></td>
</tr>
<tr>
<td>Processing of data in the buffer</td>
<td></td>
</tr>
<tr>
<td>RETI</td>
<td></td>
</tr>
</tbody>
</table>

ISE : EI²OS enable bit (ICR)
S1, S0 : EI²OS status (ICR)
3.5.15 EI²OS Processing Time

The time for processing of extended intelligent I/O service (EI²OS) depends on the following set.

- Setting of EI²OS status register (ISCS)
- Address setting of I/O register address pointer (IOA)
- Address setting of buffer address pointer (BAP)
- External data bus width at external access
- Data length of transfer data

An interruption handling time is necessary between completion of a data transfer and the shift to a hardware interruption.

■ EI²OS Processing Time (time for one transfer)

At continuing data transfer (DCT1 ≠ 0, ISCS: SE=0)

The EI²OS processing time at continuing data transfer is determined by the setting of the EI²OS status register (ISCS) as shown in Table 3.5-10.

Table 3.5-10 Extended Intelligent I/O Service Execution Time

<table>
<thead>
<tr>
<th>Setting of the EI²OS termination control bit (SE)</th>
<th>IOA update/fixed selection bit (IF) setting</th>
<th>BAP address update/fixed selection bit (BF) setting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>Termination by the termination request from a peripheral resource</td>
<td>Updated</td>
<td>Updated</td>
</tr>
<tr>
<td>Fixed</td>
<td>32</td>
<td>34</td>
</tr>
<tr>
<td>Updated</td>
<td>34</td>
<td>36</td>
</tr>
</tbody>
</table>

Unit: Machine cycle (One machine cycle is equal to one clock cycle of the machine clock (\(\phi\)).)

In addition, compensation is required depending on the conditions at executing EI²OS as shown in Table 3.5-11.

Table 3.5-11 Compensation Value for Data Transfer at EI²OS Processing Time

<table>
<thead>
<tr>
<th>Buffer address pointer</th>
<th>B/even</th>
<th>Odd</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O address pointer (IOA)</td>
<td>B/even</td>
<td>Odd</td>
</tr>
<tr>
<td>B/even</td>
<td>0</td>
<td>+2</td>
</tr>
<tr>
<td>Odd</td>
<td>+2</td>
<td>+4</td>
</tr>
</tbody>
</table>

B: Byte data transfer
Even: Word transfer at even address
Odd: Word transfer at odd address
● At end of data counter (DCT) (DCT ≠ 0, ISCS: SE=0)

When an EI²OS data transfer is complete, a hardware interruption is activated. Thus an interruption handling time is added. The EI²OS processing time for the completion of the counting by the data counter is calculated as follows:

\[
\text{EI²OS processing time after the end of data counter count} = \text{EI²OS processing time at continuing data transfer} + (21 + 6 \times Z) \text{ Machine cycle}
\]

\((Z: \text{Correction value of interrupt handling time})\)

The interruption handling time differs depending on the address specified by the stack pointer. Table 3.5-12 shows compensation values \((Z)\) of interruption handling time.

**Table 3.5-12 Compensation value of Interrupt handling time**

<table>
<thead>
<tr>
<th>Address Set by Stack Pointer</th>
<th>Compensation Value (Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even address</td>
<td>0</td>
</tr>
<tr>
<td>Odd address</td>
<td>+2</td>
</tr>
</tbody>
</table>

● Completion by a completion request from a peripheral function (DTC ≠ 0, ISCS=1).

When an EI²OS data transfer is terminated by a completion request from a peripheral function before completion of the transfer processing \((\text{ICR}: S1, S0 = "11B")\), a shift to the interruption processing program occurs. The EI²OS processing time by using completion request from a peripheral function is calculated as follows:

\[
\text{EI²OS processing time at termination during processing} = 36 + 6 \times Z \text{ Machine cycle}
\]

\((Z: \text{Compensation value of interrupt handling time})\)

**Reference:** One machine cycle is equal to one clock cycle of the machine clock (\(\phi\)).
### 3.5.16 Exception Processing Interrupt

The F²MC-16LX family performs exception processing when an undefined instruction is executed.

The processing of an exceptional event is the same as an interruption. If an exceptional event is detected between commands, the normal processing is stopped temporarily to handle the event.

Because exceptional processing is what takes place after the occurrence of an unexpected event during operation, make sure that it is used only for debugging or for the activation of recovery software programs in emergency cases.

#### Exception processing

- **Operation of exception processing**

  The F²MC-16LX family treats all instruction codes not defined in the instruction map as undefined instructions. If an undefined instruction is executed, the processing equal to the software interrupt instruction INT # 9 is performed.

  At exception processing, the following processing is performed before the transition to interrupt processing:
  - The contents of dedicated registers (A, DPR, ADB, DTB, PCB, PC, and PS) saved to the system stack
  - The interrupt enable flag (CCR: I) cleared to "0" and an interrupt disabled
  - The stack flag (CCR: S) set to "1"

  The value of the program counter (PC) saved in the stack is a value of the address where undefined instructions are stored. For instruction codes of 2 bytes or more, the value of the program counter (PC) is a value of the address where instruction codes that can be identified as undefined are stored. When it is necessary to make a cause analysis by exceptional processing refer to the evacuated program counter (PC).

- **Return from exception processing**

  If the program counter (PC) indicates that there has been an undefined command, the execution of an interruption return command (RETI) from the exceptional processing causes a return to exceptional processing. Some measures such as performing a software reset should be taken when returning from exception processing.
### 3.5.17 Waiting Times Associated with Interruption Processing

Between the occurrence of an interruption request and the start of the execution of the interruption processing, a time is required that equals the time remaining to the completion of the execution of the current command plus the interruption handling time.

<table>
<thead>
<tr>
<th>Time Required to Start Interrupt Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Between the occurrence of an interruption request and the acceptance of the interruption request and the start of the execution of the interruption processing, a time is required that equals the interruption request sampling waiting time plus the interruption handling time (the time required to prepare for the interruption processing). Figure 3.5-17 shows the interrupt processing time.</td>
</tr>
</tbody>
</table>

#### Figure 3.5-17 Interrupt Processing Time

- **CPU operation**
  - Execute normal instruction
  - Interrupt handling
  - Interrupt processing

- **Interrupt wait time**
  - Interrupt request sampling waiting time
  - Interrupt handling time (\(\phi\) : Machine cycle)

- **Interrupt request generation**

  - Instruction last cycle, sampling interrupt request here
  - \(\star\) : 1 machine cycle is equal to 1 clock cycle of machine clock (\(\phi\)).

- **Interrupt request sampling wait time**

  - It indicates a time from the generation of the interrupt request to the termination of the currently executing instruction.

  - Whether there has been an interruption request or not is determined by performing interruption request sampling in last cycle of the command. There is a waiting time during execution of each command, because the CPU cannot recognize the interruption request.

<table>
<thead>
<tr>
<th>Reference:</th>
</tr>
</thead>
<tbody>
<tr>
<td>The interrupt request sampling wait time is longest when the interrupt request is generated immediately after starting execution of the POPW, RW0... RW7 instructions with the longest execution cycle (45 machine cycles).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>When a software interrupt occurs, the interrupt request sampling time is not generated.</td>
</tr>
</tbody>
</table>
Interrupt handling time (θ machine cycle)

The CPU requires an interruption handling time of θ machine cycle after accepting an interruption request, to evacuate the dedicated registers to the system stack, take in the interruption vector table addresses. The interruption handling time (θ) is calculated using the following equation:

\[ θ = 24 + 6 \times Z \]  

machine cycle (Z: Compensation value of interruption handling time)

The interruption handling time differs depending on the address specified by the stack pointer. Table 3.5-13 shows compensation value (Z) of interruption handling time.

**Table 3.5-13  Compensation value (Z) of interrupt handling time**

<table>
<thead>
<tr>
<th>Address specified by stack pointer</th>
<th>Compensation value (Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even address</td>
<td>0</td>
</tr>
<tr>
<td>Odd address</td>
<td>+2</td>
</tr>
</tbody>
</table>

Reference: One machine cycle is equal to one clock cycle of the machine clock (φ).
CHAPTER 3  CPU FUNCTION

3.5.18 Stack Operation for Interrupt Processing

When an interruption is accepted, the dedicated registers data is automatically evacuated to the system stack before the shift to the interruption processing. When the interruption processing is completed, the dedicated registers data is automatically replaced from the system stack.

■ Stack Operation at Starting Interrupt Processing

When an interrupt is accepted, the CPU automatically saves the values of the current-dedicated registers to the system stack in the following order.

- Accumulator (AH, AL)
- Direct page register (DPR)
- Additional data bank register (ADB)
- Data bank register (DTB)
- Program bank register (PCB)
- Program counter (PC)
- Processor status (PS)

Figure 3.5-18 shows the stack operation at starting interrupt processing.

![Figure 3.5-18 Stack Operation at Starting Interrupt Processing](image)

■ Stack Operation at Return from Interrupt Processing

If an interruption return command (RETI) is executed after completion of interruption processing, the values of the dedicated registers (PS, PC, PCB, DTB, ADB, DPR, AL, and AH) are replaced to the registers and the registers return to the original states (i.e. the states immediately before the interruption).
3.5.19 Program Example of Interrupt Processing

This section gives a program example of interrupt processing.

### Program Example of Interrupt Processing

#### Processing specification

An example of interruption program that uses external interruption 0 (INT0)

#### Coding example

```assembly
DDR0 EQU 000010H ;Port 1 direction register
ENIR EQU 030H ;DTP/external interrupt enable register
EIRR EQU 031H ;DTP/external interruption flag
ELVR EQU 032H ;Request level setting register
ICR01 EQU 0B1H ;Interrupt control register

;----------Main Program-----------------------------------------------
CODE CSEG
;
START:
    MOV DDR0,#00000000B ;P10/INT0 pin is set to "input".
    OR CCR,#40H ;I flag of CCR in PS set to interrupt enabled.
    MOV #00H,ICR01 ;interuption level 0 (highest)
    MOV #00000001B,ELVR ;make INT0 H level request
    MOV #00H,EIRR ;clearing of the INT0 interruption-causing
    MOV #01H,ENIR ;INT0 input permitted

    ;LOOP:
    BRA LOOP

;----------Interrupt Program------------------------------------------
ED_INT1:
    MOV #00H,EIRR ;New acceptance of INT0 prohibited
    NOP
    NOP
    NOP
    NOP
    NOP
    RETI ;Returning from interrupt processing

;----------Vector Setting---------------------------------------------
VECT CSEG ABS=0FFH

;vector set in interrupt number #11 (0BH)
ORG 0FFC8H
DSL ED_INT1
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode

CODE ENDS

END START
```
■ Extended Intelligent I/O Service (EI²OS) program

● Processing specification

- The EI²OS is activated by detecting "H" level to be inputted to the INT0 pin from outside.
- If "H" level is input into the INT0 pin from outside, an EI²OS is activated and the port 0 data is transferred to the "3000H" area of the memory.
- The number of transfer data bytes is 100. After completion of the transfer of 100 bytes, an EI²OS transfer completion-based interruption occurs.

● Coding example

```assembly
DDR1 EQU 000016H ;Port 1 direction register
ENIR  EQU 000030H ;DTP/external interrupt enable register
EIRR  EQU 000031H ;DTP/external interrupt factor register
ELVR  EQU 000032H ;a register to specify the required level
ICR01 EQU 0000B1H ;Interrupt control register
BAPL  EQU 000100H ;Lower Buffer address pointer
BAPM  EQU 000101H ;Middle Buffer address pointer
BAPH  EQU 000102H ;Higher Buffer address pointer
ISCS  EQU 000103H ;EI²OS Status
IOAL  EQU 000104H ;Lower I/O address pointer
IOAH  EQU 000105H ;Higher I/O address pointer
DCTL  EQU 000106H ;Lower data counter
DCTH  EQU 000107H ;Higher Data counter
ER0   EQU EIRR:0 ;definition of the external interruption request flag bit

----------Main Program-------------------------------------------
CODE   CSEG
START:
    AND  CCR,#0BFH ;clearing of the CCR I flag in PS to prohibit interruptions
    MOV  I:DDR6,#00000000B ;Sets the P10/INT0 pin for input.
    MOV  BAPL,#00H ;Sets buffer address (003000H).
    MOV  BAPM,#30H
    MOV  BAPH,#00H
    MOV  ISCS, #00010001B ;No I/O address update, byte transfer
    MOV  IOAL, #00H ;Sets transfer source address (port 0: 000000H)
    MOV  IOAH,#00H
    MOV  DCTL, #64H ;Sets transfer byte count (100 bytes)
    MOV  DCTH,#00H
    MOV  I:ICR01,#00001000B ;EI²OS channel 0, EI²OS permitted
                      ;Interrupt levels 0 (strength)
    MOV  I:ELVR,#0000001B ;make INT0 H level request
    MOV  I:EIRR,#00H ;clearing of the INT0 interruption-causing
    MOV  I:ENIR,#01H ;INT0 interruption permitted
    MOV  ILM, #07H ;Sets ILM in PS to level 7
    OR  CCR,#40H ;CCR I flag setting (PS) to permit interruptions
LOOP:    BRA LOOP ;infinite loop
----------Interrupt Program------------------------------------------
```
WARI CLR BER0 ;DTP/clearing of the external interruption request flag :
user processing ;confirmation of the cause of the EI²OS :
;completion, processing of data in the buffer, :
;EI²OS re-setting, etc.
RETI
CODE ENDS
----------Vector setting ---------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFC8H ;vector set in interrupt number #11 (0B_H)
DSL WARI
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START
CHAPTER 3 CPU FUNCTION

3.6 Reset

If a reset factor occurs, the CPU stops the current processing temporarily to shift to the reset operation.

When a reset is cancelled, the mode data and reset vector will be taken in and the program processing will be started from the address specified by the reset vector.

There are 5 types of reset factors:
- Generating power on reset
- Cancelling hardware standby Mode
- Watchdog timer overflow
- An external reset signal input from the \texttt{RST} pin
- Generation of an internal reset signal by the software

\section*{Reset Factor}

Table 3.6-1 shows the 5 reset-causing event types.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Reset & Factor & Machine clock & Watchdog Timers & Oscillation stability wait \\
\hline
Power on & At power on & Main clock (MCLK) & Stops & Yes \\
\hline
Cancelling hardware standby & Switching HST pin from "L" level input to "H" level input & Main clock (MCLK) & Stops & Yes \\
\hline
Watchdog timer & Elapsing Interval Time of Watchdog Timer & Main clock (MCLK) & Stops & None \\
\hline
External pin & Input "L" level to \texttt{RST} pin & Hold the state immediately before & Hold the state immediately before & None \\
\hline
Software & Set the RST bit of low-power consumption mode control register (LPMCR) to "0" & Hold the state immediately before & Hold the state immediately before & None \\
\hline
\end{tabular}
\caption{Reset Factor}
\end{table}

MCLK: Main clock: (oscillation clock frequency divided by 2)

- Power on reset

A power on reset is a reset that occurs upon power on.

- The main clock oscillation stabilization time is fixed at $2^{18}$ or $2^{17}$ oscillation clock cycles ($2^{18}$/HCLK or $2^{17}$/HCLK) (For MB90522B, MB90523B, MB90F523B and MB90V520A, $2^{18}$/HCLK is used. For MB90523A and MB90F523A, $2^{17}$/HCLK is used.) After expiration of the oscillation stabilization time, the reset vector is taken in.
● Reset by canceling hardware standby mode

When the \text{RST} pin signal input is switched from level "L" to level "H", the hardware standby mode will be cancelled and an internal reset will occur.

- Main clock oscillation stabilization wait time is generated. The oscillation stabilization wait time can be specified using the clock selection register (CKSCR).

● Watchdog reset

If the watchdog timer is not cleared within the pre-specified interval time after activation of the watchdog timer, a watchdog timer overflow causes a reset.

- The watchdog timer can be cleared by setting the watchdog control bit (WTE) of the watchdog timer control register (WDTC) to "0".
- Main clock oscillation stabilization wait time is generated. The oscillation stabilization wait time can be specified using the clock selection register (CKSCR).

● External reset

When level "L" signals are input to the external reset pin (\text{RST} pin), a reset will occur.

- To cause an external pin-based reset, input level "L" signals to the \text{RST} pin for 16 machine cycles (16/\phi) or more.
- There is no main clock oscillation stabilization wait time for an external pin-based reset.

Notes:

- When an external reset is input during write operations by executing MOV or other transfer commands, a reset occurs after write operations have ended normally. Note a reset may occur before the set number of transfers complete when string type commands (MOVS, etc.) are used.
- A recovery cannot be effected by performing a reset via the external pin in the sub watch clock mode. For details, refer to "3.8.7 Watch Mode".

● Software reset

When the internal reset signal generation bit of the low-power consumption mode control register is set to "0" (LPMCR:RST=0), internal reset signals lasting for 3-machine cycles (3/\phi) will be generated.

- There is no main clock oscillation stabilization wait time for a software-based reset.

Reference: Clock definition

- HCLK: Oscillation clock frequency
- MCLK: Main clock frequency
- \phi: Machine clock (CPU operating clock) frequency
- 1/\phi: Machine cycle (CPU operating clock period)

See "3.7 Clock" for details.
3.6.1 Reset Factors and Oscillation Stabilization Wait Times

The MB90520A series products has the following reset factors. The main clock oscillation stabilization wait time differs depending on the type of the reset factors.

Table 3.6-2 shows reset factors and oscillation stabilization wait times. Table 3.6-3 shows oscillation stabilization wait time is set by the clock selection register (CKSCR).

<table>
<thead>
<tr>
<th>Reset Factor</th>
<th>Oscillation Stabilization Wait Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power on reset</td>
<td>MB90522B/MB90523B/MB90F523B/MB90V520A: 2^{18}/HCLK (65.536ms)</td>
</tr>
<tr>
<td></td>
<td>MB90522A/MB90523A: 2^{17}/HCLK (Approx. 32.77ms)</td>
</tr>
<tr>
<td>Hardware standby (HST pin)</td>
<td>Setting by Clock selection register*</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>Setting by Clock selection register*</td>
</tr>
<tr>
<td>External reset pin (RST pin)</td>
<td>None</td>
</tr>
<tr>
<td>Software reset</td>
<td>None</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency

*: Oscillation stabilization wait time is determined by the oscillation stabilization wait time setting bit (CKSCR:WS1, WS0) in the clock selection register.
Figure 3.6-1  A power on reset occurs for the oscillation stabilization wait time of the MB90520A series.

Table 3.6-3  Oscillation stabilization wait time is set by the clock selection register (CKSCR)

<table>
<thead>
<tr>
<th>WS1</th>
<th>WS0</th>
<th>Oscillation Stabilization Wait Time</th>
<th>Parenthesized values are examples calculated at an oscillation clock frequency of 4 MHz.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$2^{10}$/HCLK (approx. 256µs)</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$2^{13}$/HCLK (approx. 2.05ms)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$2^{15}$/HCLK (approx. 8.19ms)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$2^{17}$/HCLK (approx. 32.77ms)</td>
<td></td>
</tr>
</tbody>
</table>

Note:  Ceramics and quartz and other oscillators normally take from a few ms to tens of ms until they stabilize at a specific number of oscillator after start of oscillation. Set the oscillation stabilization wait time required for the oscillator to be used.

Reference: Refer to "3.7.5 Oscillation Stabilization Wait Time" for details.

- Oscillation Stabilization Waiting reset state

If power on reset, hardware standby mode cancellation and watchdog timer overflow-based reset are generated, the reset vector is taken in after cancellation of the reset state (main clock oscillation stabilization).

- If an external reset is being input (i.e. level "L" signals are being input to the RST pin), the reset vector is taken in after cancellation of the external reset (RST pin = level "H").
3.6.2 External Reset Pin

The external reset pin (RST pin) is a pin for reset input only. By level "L" signal input, synchronization with the CPU operation clock is made and an internal reset is generated.

■ Block Diagram of External Reset Pin

Figure 3.6-2 Block Diagram of External Reset Pin

Notes:
- When data is written to memory, inputs to the RST pin are received in cycles that do not corrupt memory to ensure correct writing operation.
- External resets require that the internal clock operates to perform internal clock sampling. When an external clock is operated to input, an external clock is required to input when an external reset is inputted.
3.6.3 Reset Operation

When a reset occurs, the cancellation of a reset causes the mode data and reset vector to be taken in. The CPU processing resumes from the address specified by the reset vector that has been taken in.

For power on reset, hardware standby mode cancellation and watchdog timer overflow-based reset, the reset sequence is executed after expiration of the oscillation stabilization wait time.

■ Overview of Reset Operation

Figure 3.6-3 shows operation flow during a reset.

![Figure 3.6-3 Operation Flow during a Reset](image)

■ Operation Status Changes during a Reset

1. If a reset occurs, the current processing is stopped temporarily and a shift to the reset operation takes place. For power on reset, hardware standby mode cancellation and watchdog timer overflow, shift to the waiting status for main clock oscillation stabilization takes place.

2. When the reset is cancelled, the CPU mode data and reset vector will be taken in by the reset sequence (mode fetch).

3. The CPU processing resumes from the address specified by the reset vector that has been taken in by the mode fetch.
CHAPTER 3 CPU FUNCTION

● Reset sequence

- The destination from which data is to be read (internal ROM) with respect to the mode data/reset vector and the CPU operation mode are specified based on the mode pin (MD0 to MD2) settings.
- If the mode pins (MD0 to MD2) have been set to the internal vector mode (MD0 to MD2 = "110B") by this point, the mode data and reset vector are read from the specified areas. The mode data is stored in the mode register and the reset vector is stored in the program counter and program bank register.
- The mode data and reset vector are assigned to the 4 bytes "FFFFDFH" to "FFFFDCH".

Figure 3.6-4 shows how the reset vector and mode data are transferred.

![Figure 3.6-4 Reset Vector and CPU Mode Data Transfers](image)

● Mode Pin

The mode pins (MD0 to MD2) specify the method to take in the mode data and reset vector. For MB90520A, only the internal vector mode is valid. In the case of a product with a built-in flash memory, the flash serial writing mode or flash memory mode can also be selected.

● Mode data (address: FFFFDFH)

- The mode data is a bit that specifies the CPU memory access mode. For MB90520A, only the single-chip mode is valid.
- The mode register can be altered only during execution of a reset sequence it becomes the value specified by the mode data (FFFFDFH).
- The mode register setting becomes effective after the reset sequence.

Reference: For details on mode pin and mode data, refer to "3.9 CPU Mode".

● Reset vectors (FFFFDCH to FFFFDEH)

Specify the address from which the CPU processing is to be started after completion of the reset sequence. The CPU processing starts from the address specified by the reset vector.
3.6.4 Reset Factor

Reset generating factor can be recognized by reading the watchdog timer control register (WDTC).

■ Reset factor flag bit

In the case of occurrence of a reset or resets, checks can be made by reading the watchdog timer control register (WDTC).

Figure 3.6-5 shows a block diagram of the reset factor.

---

**Figure 3.6-5 Reset factor flag bit block diagram**

- At power-on generation detection circuit
- Hardware stand-by cancellation detection circuit
- External reset request detection circuit
- Watchdog timer reset generation detection circuit
- LPMCR: RST bit writing detection circuit
- Power-on generation detection circuit
- Hardware stand-by cancellation detection circuit
- External reset request detection circuit
- Watchdog timer reset generation detection circuit
- LPMCR: RST bit writing detection circuit
- Set to RST=“0”
- Clear
- Delayed circuit
- Read watchdog timer control register (WDTC)

S: Set
R: Reset
Q: Output
F/F: Flipflop
CHAPTER 3 CPU FUNCTION

■ Relationships between the reset factor flag bit bits and the reset factor

Figure 3.6-6 shows the compositions of the reset factor flag bit bits of the watchdog timer control register (WDTC), and Table 3.6-4 shows the relationships between the reset factor flag bits and the reset factor.

Figure 3.6-6 Configuration of Reset Factor Flag Bit (Watchdog timer control register)

<table>
<thead>
<tr>
<th>Watchdog timer control register (WDTC)</th>
<th>bit7</th>
<th>bit6</th>
<th>bit5</th>
<th>bit4</th>
<th>bit3</th>
<th>bit2</th>
<th>bit1</th>
<th>bit0</th>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PONR</td>
<td>STBR</td>
<td>WRST</td>
<td>ERST</td>
<td>SRST</td>
<td>WTE</td>
<td>W1</td>
<td>W0</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>R: Read only</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.6-4 Reset factor flag bit content and reset factor

<table>
<thead>
<tr>
<th>Reset Factor</th>
<th>PONR</th>
<th>STBR</th>
<th>WRST</th>
<th>ERST</th>
<th>SRST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generating power on reset</td>
<td>1</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Release of hardware standby mode</td>
<td>*</td>
<td>1</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Reset by watchdog timer</td>
<td>*</td>
<td>*</td>
<td>1</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Input of external reset signal to RST pin</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>1</td>
<td>*</td>
</tr>
<tr>
<td>Software reset</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>1</td>
</tr>
</tbody>
</table>

*: The previous state is held.
×: Indeterminate

■ Setting/Clearing a reset factor flag bit

- At two or more reset factors

A reset caused by two or more factors also results in the setting of the corresponding reset factor flag bit of the watchdog timer control register (WDTC) to "1".

Therefore, if an external reset (RST pin) and a watchdog timer function-based reset occur simultaneously, both the ERST and WRST bits are set to "1".

- Power on reset

If a power on reset occurs and the PONR bit is set to "1", the reset factor bit flag other than the PONR bit become invalid.

Make sure that, when the PONR bit is set to "1", the reset factor flag bits other than the PONR bit are ignored software processing (program).

- Clearing of reset factor bit

When reset factor bit is cleared, read watchdog timer control register (WDTC). Reset factor bit settings in the WDTC register cannot be cleared unless the WDTC register is read or a power on reset is performed.
### 3.6.5 State of Each Pin at Reset

This section explains the state of each pin at reset.

#### State of Pins at Reset

- The states of the pins during a reset are determined by the mode pin (MD0 to MD2) and mode data (M1,M0) settings.
- For the MB90520A series products, only the internal vector mode and single-chip mode can be set.
- The I/O pins (the input and output pins for the peripheral functions) are all in the high impedance, input shutoff state (although there are some pins through which signals can be input).
- The segment output pins (SEG0 to SEG7) of the LCD controller/driver are in the "L" output state.

**Note:** When a reset factor is generated, make sure that devices connected to high impedance pins do not malfunction.

**Reference:** For details on the status of each pin during reset, refer to "3.8 Low-power Consumption Mode".
3.6.6 Registers that are initialized by reset factors

Some of the registers in the CPU for the MB90520A series products are initialized when certain reset factors occur.

**Registers that are initialized by reset factors**

Table 3.6-5 shows registers that are initialized by specific reset factors.

<table>
<thead>
<tr>
<th>Registers</th>
<th>Bit (bit No.)</th>
<th>Bit name</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPMCR Low-power consumption</td>
<td>CG1, CG0</td>
<td>CPU clock pause cycle number set bit</td>
<td>Power on, cancelling hardware standby mode, setting with the watchdog function, initializes the register to &quot;00B&quot;, and the CPU clock pause cycle number becomes 0 cycles (CPU clock = machine clock).</td>
</tr>
<tr>
<td>mode control register</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CKSCR Clock selection register</td>
<td>WS1, WS0</td>
<td>Oscillation stabilization wait time selection bit</td>
<td>Initialized to &quot;11B&quot; only in a power on reset, the oscillation stability wait time is $2^{17}$/HCLK (approx. 32.77 ms for a 4 MHz oscillation clock).</td>
</tr>
<tr>
<td></td>
<td>MCS (bit10)</td>
<td>PLL clock selection bits</td>
<td>Power on, hardware standby mode cancellation, watchdog function reset initializes to &quot;1&quot; and the main clock is selected.</td>
</tr>
<tr>
<td></td>
<td>CS1, CS0</td>
<td>Multiplication rate selection bit</td>
<td>Initialization to &quot;00B&quot; is performed only during power on reset when operating frequency = oscillation frequency (HCLK).</td>
</tr>
<tr>
<td>WDTC Watchdog timers control</td>
<td>WTE (bit2)</td>
<td>Watchdog timer control bit</td>
<td>Power on, hardware standby mode cancellation, watchdog function reset stops the watchdog timer.</td>
</tr>
<tr>
<td>registers</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The register that can not be initialized by reset input

MB90520A series have the registers that are only initialized by power-on. Table 3.6-6 shows the register that can not be initialized by the reset factors.

### Table 3.6-6 The register that can not be initialized by reset input

<table>
<thead>
<tr>
<th>Reset type</th>
<th>CKSCR</th>
<th>WTC</th>
<th>LPMCR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WS1</td>
<td>WS0</td>
<td>MCS</td>
</tr>
<tr>
<td>Software reset with using only RST pin</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Watch-dog reset</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
<tr>
<td>Power-on reset</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Hardware standby</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
</tbody>
</table>

WS1, WS0: Setting of main clock oscillation stabilization wait time
MCS: Machine clock setting ("0"= PLL clock, "1"= main clock)
CS1, CS0: PLL clock multiplied setting
WDCS: Watch-dog input clock ("0"= watch timer, "1"= timebase timer)
O: Initialized
X: No initialized

Pay attention in particular to the MCS bit, because the bit is the machine clock setting bit. For example, if the power-on reset condition is not satisfied as power-on sequence, the power-on reset should not be occurred. So, because the MCS bit is not be initialized, internal operating frequency should be without the operation guarantee range, and the device should not operate normally.

Or, if CPU overrunning from some factor causes writing MCS/CS1/CS0 bit, internal operating frequency should be without the operation guarantee range, and should not be able to reset by RST pin. (If the internal watch-dog is generating, MCS bit should be initialized. So the device should operate normally)

If these case will be occur in plan, using HST+RST (shorted both HST pin and RST pin) is recommended.

Table 3.6-7 shows the register that can not be initialized by reset input with using HST+RST. Pay attention to difference between HST+RST reset input and only RST reset input at operating status of releasing reset.

### Table 3.6-7 The register that can not be initialized by reset input

<table>
<thead>
<tr>
<th>Reset type</th>
<th>CKSCR</th>
<th>WTC</th>
<th>LPMCR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WS1</td>
<td>WS0</td>
<td>MCS</td>
</tr>
<tr>
<td>HST+RST</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
</tbody>
</table>

X: Initialized
O: No initialized
Figure 3.6-7 Operating transition from reset input

[Operating transition from reset input]

Reset Input (RST, HST+RST)

A. Oscillation status

<table>
<thead>
<tr>
<th>Oscillation</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td>Oscillation</td>
</tr>
<tr>
<td>Sub</td>
<td>Oscillation</td>
</tr>
</tbody>
</table>

Using only RST (HST = "H")

Using HST+RST

<table>
<thead>
<tr>
<th>Main</th>
<th>Oscillation</th>
<th>Stop</th>
<th>Main oscillation stabilization wait</th>
<th>Main run operating enable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub</td>
<td></td>
<td></td>
<td>Sub oscillation stabilization wait</td>
<td>Sub run operating enable</td>
</tr>
</tbody>
</table>

B. Instruction executing timing ("L": Stop, "H": Start)

- Using only RST (HST = "H")

- Using HST+RST

  Main mode

  Oscillation stabilization wait
time of setting before reset input

  When sub mode is requested,
main clock operation is enabled.
During the main clock operation,
writing to SCS bits is possible.

  Sub mode

  2^{18}-count of sub oscillation (32 kHz) = about 2 s

- At power-on reset

  Vcc (power supply)

<table>
<thead>
<tr>
<th>Power-on reset</th>
<th>Oscillation</th>
<th>Stop</th>
<th>Main oscillation stabilization wait</th>
<th>Main run operating enable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Main</td>
<td></td>
<td>Sub oscillation stabilization wait</td>
<td>Sub run operating enable</td>
</tr>
</tbody>
</table>

  Main mode

  Oscillation stabilization wait
time of 2^{18}-count of main oscillation

  When sub mode is requested,
main clock operation is enabled.
During the main clock operation,
writing to SCS bits is possible.

  Sub mode

  2^{18}-count of sub oscillation (32 kHz) = about 2 s
3.7 Clock

The clock generation section controls the operation clock supplied to the CPU and peripheral functions.
- The operation clock supplied to the CPU and peripheral functions is called the machine clock (\( \phi \)). One cycle of the machine clock is the machine cycle.
- The clock supplied from the high-speed oscillation pins is called the oscillation clock (HCLK).
- A clock frequency divided by 4 of the clock supplied from the low-speed oscillation pins is called the sub-clock (SCLK).
- The clock obtained by multiplying the oscillation clock (HCLK) by a multiplier by means of internal PLL oscillation is called the PLL clock.

Clock

Two separate oscillation circuits, one for the high-speed clock and the other for the low-speed clocks, are built into the clock generation section.
- By connecting an external oscillator, the oscillation clock (HCLK) and sub-clock (SCLK) are generated.
- It is also possible to input an externally generated clock and turn it into the oscillation clock or sub-clock.
- A multiplication clock of the oscillation clock (HCLK) can be generated by using built-in PLL multiplication circuit. There are 4 multiplication rate to select from.

- Oscillation clock (HCLK)
  A clock generated by connecting an oscillator to the high-speed oscillation pins (X0 and X1 pins) or inputting an external clock.

- Sub-clock (SCLK)
  A clock generated by connecting an oscillator to the low-speed oscillation pins (X0A and X1A pins) or inputting an external clock and internally clock frequency divided by 4. Used for sub-operation and clock timers.

- Main clock (MCLK)
  A clock is generated by internally the oscillation clock (HCLK) frequency divided by 2 and input into the timebase timer and clock selector.

- PLL clock (PCLK)
  A clock generated by multiplying the oscillation clock (HCLK) by means of built-in PLL multiplication circuit (PLL oscillation circuit). There are 4 multiplication rate to select from.
CHAPTER 3  CPU FUNCTION

● Machine clock (φ)

- Operation clock for CPU and peripheral functions. One cycle of the machine clock is the machine cycle \((1/\phi)\).
- A machine clock is selected from the following:
  - Main clock: oscillation clock (HCLK) frequency divided by 2
  - Sub-clock (SCLK)
  - 4 types of PLL clocks

---

**Notes:**

- When the operating voltage is 5 V, the oscillation clock (HCLK) of 3 MHz to 16 MHz can be generated. The maximum operating frequency of the CPU and the peripheral functions are both 16 MHz.
- If a multiplier rate of the PLL clock exceeding the maximum operating frequency is specified, devices will not operate correctly. For example, when the oscillation clock (HCLK) of 16 MHz is generated, only a multiplier of 1 can be specified.
- The minimum operating frequency for PLL oscillation is 4 MHz and the setting of a lower frequency is not allowed.

---

■ Clock Supply Map

The CPU and peripheral functions use as their operation clock (machine clock) by a clock generated in the clock generation section.

For this reason, the operation of the CPU and peripheral functions is affected by a change in the PLL clock multiplication rate and machine clock selection (main clock, sub-clock, PLL clock).

Some of the peripheral functions are provided with the frequency-divided outputs (timebase timer, watch timer, etc.). The peripheral functions can select supplied operation clock.

Figure 3.7-1 shows clock supply map.
Figure 3.7-1 Clock Supply Map

- **Peripheral function**
  - 16-bit reload timer-0
  - TIN0
  - Pin
  - SCLK
  - Pin
  - MCLK
  - Pin
  - PCLK
  - Pin

- **CPU Function**
  - Sub clock
  - Oscillation clock
  - PLL clock
  - Machine clock

- **Clock generation unit**
  - Timebase timer
  - PLL multiplying circuit
  - 2-frequency division
  - 4-frequency division

- **CPU intermittent operation**

- **CPU**

- **wake-up interruption**

- **External interrupt**

- **Watchdog timer**

- **Clock output module**

- **oscillation stabilization controller**

- **Communication prescaler**

- **UART**

- **Extended serial I/O interface 1,2**

- **Output compare 4 to 7**
  - SCLK
  - Pin

- **Input capture**

- **Output compare 0 to 3**

- **16-bit free run timer-0**

- **16-bit free run timer-1**

- **Output**
  - OUT4 to 7
  - Pin
  - IN00,01
  - IN10,11

- **FREQUENCY DIVIDERS**
  - 2 - frequency
  - 4 - frequency

- **Pin Connections**
  - X0, X1, X0A, X1A, TIN0, TOT0, SCLK, MCLK, PCLK, HCLK

- **Additional Pins**
  - INT0 to INT7
  - WI0 to WI7
  - CKOT

- **Other Components**
  - UART
  - Communication prescaler
  - Extended serial I/O interface 1,2
  - Output compare 4 to 7
  - Input capture
  - Output compare 0 to 3
  - Oscillation clock stabilization controller

- **Other Functions**
  - CPU intermittent operation
  - Wake-up interruption
  - External interrupt

- **Pin Descriptions**
  - HCLK: Oscillation clock
  - SCLK: Sub clock
  - MCLK: Main clock
  - PCLK: PLL clock
  - φ: Machine clock
3.7.1 Block Diagram of Clock Generation Section

Block Diagram of Clock Generation Section

Figure 3.7-2 Block Diagram of Clock Generation Section

- Standby control circuit
- Low-power consumption mode control register (LPMCR)
- CPU intermittent operation cycle selector
- CPU clock control circuit
- CPU operation clock
- Peripheral clock control circuit
- Peripheral function operation clock
- Standby circuit
- Operation clock selector
- Oscillation stabilization wait interval selector
- Main clock
- Timebase timer
- Sub clock
- Oscillation clock (HCLK)
- Oscillation clock (HCLK) generation circuit
- Clock selection register (CKSCR)
- Watch mode
- Sleep signal
- Stop signal
- Reset
- Interrupt
- Hardware standby
- X0, X1, X0A, X1A pins
- Sub clock (HCLK) generation circuit
- 4-freq. division
- 2-freq. division
- Oscillation clock
- Machine clock
- OSC
- CMCM
- WS1
- WS0
- SCS
- MCS
- CS1
- CS0
- PLL multiplying circuit
- Watch timer
- CPU
- Watch mode
- Sleep signal
- Stop signal
- Reset
- Interrupt

Legend:
- S: Set
- R: Reset
- Q: Output
Figure 3.7-2 includes the blocks of standby control circuit, timebase timer and watch timer. Returning from the sub watch mode through external reset is not possible. For details, see Section "3.8.7 Watch Mode".

- **Oscillation clock (HCLK) generator circuit**
  A clock (HCLK) is generated by the oscillator connected to the high-speed oscillation pins (X0 and X1 pins). It is also possible to input an external clock.

- **Sub-clock generator circuit**
  A low-speed clock is generated by the oscillator connected to the low-speed oscillation pins (X0A and X1A pins). It is also possible to input an external clock. The clock obtained by internally turning the generated clock into clock frequency divided by 4 is used as the sub-clock (SCLK).

- **PLL multiplying circuit**
  The oscillation clock (HCLK) is supplied to the operation clock selector after being multiplied by means of PLL oscillation.

- **Operation clock selector**
  The clock to be supplied to the CPU clock control circuit and peripheral clock control circuit is selected from main clock, sub-clock and PLL clock.

- **Clock selection register (CKSCR)**
  This register is switching between the main clock and sub-clock, switching between the main clock and PLL clock, oscillation stabilization wait time or PLL clock multiplier selection, etc.

- **Oscillation stabilization wait time selector**
  This selector is Main clock oscillation stabilization wait time selection circuit when stop mode cancellation, hardware standby mode cancellation or watchdog function-based reset occurs. Select the four kinds of timebase timer.
### 3.7.2 Clock Selection Register (CKSCR)

The clock selection register (CKSCR) is used for machine clock switching.

#### Clock selection register (CKSCR)

**Figure 3.7-3 Configuration of the Clock Selection Register (CKSCR)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>SCM</td>
</tr>
<tr>
<td>14</td>
<td>MCM</td>
</tr>
<tr>
<td>13</td>
<td>WS1</td>
</tr>
<tr>
<td>12</td>
<td>WS0</td>
</tr>
<tr>
<td>11</td>
<td>SCS</td>
</tr>
<tr>
<td>10</td>
<td>CS1</td>
</tr>
<tr>
<td>9</td>
<td>CS0</td>
</tr>
<tr>
<td>8</td>
<td>Reset value</td>
</tr>
</tbody>
</table>

**Note:**

The sub-clock selection bit and the PLL clock selection bit are initialized to the state in which the main clock is selected at a reset by a power on, the cancellation of the hardware standby or the watchdog function. (SCS=1, MCS=1)
### Table 3.7-1 Function of clock selection register (CKSCR) (1 / 2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit9, bit8</td>
<td>CS1, CS0: Multiplication rate selection bits</td>
</tr>
<tr>
<td>bit10</td>
<td>MCS: PLL clock selection bit</td>
</tr>
<tr>
<td>bit11</td>
<td>SCS: Sub-clock selection bit</td>
</tr>
<tr>
<td>bit13, bit12</td>
<td>WS1, WS0: Oscillation stabilization wait time selection bits</td>
</tr>
<tr>
<td>bit14</td>
<td>MCM: PLL clock operation underway bit</td>
</tr>
</tbody>
</table>
CHAPTER 3  CPU FUNCTION

Table 3.7-1  Function of clock selection register (CKSCR)  (2 / 2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| SCM: Sub-clock operation underway bit | Indicates which of the main clock or the sub-clock is operating as the machine clock. **If SCM= "0":** The sub-clock is operating as the machine clock. **If SCM= "1":** The main clock or the PLL clock is operating as the machine clock.  
  - If SCM is set to 1 when the sub-clock mode is selected (SCS=0) by the sub-clock selection bit, it indicates that the main clock switches to the sub-clock. If SCM is set to 0 when the main clock mode is selected (SCS=1), it indicates that the sub-clock switches to the main clock.  
  - Writing this bit has no effect on operation. |
3.7.3 Clock Mode

The clock modes are the main clock mode, PLL clock mode and sub-clock mode.

■ Main Clock Mode, PLL Clock Mode, and sub-clock Mode

● Main clock mode

In the main clock mode, a main clock whose frequency is the oscillation clock (HCLK) frequency divided by 2 is used as the operation clock for the CPU and peripheral functions and the PLL clock is disabled.

● PLL clock

In the PLL clock mode, the PLL clock is used as the operation clock for the CPU and peripheral functions. The PLL clock multiplication rate to be used is selected by means of the clock selection registers (CKSCR:CS1,CS0).

● Sub-clock mode

In the sub-clock mode, the sub-clock is used as the operation clock for the CPU and peripheral functions and the main clock and PLL clock are disabled.

■ Switching of clock modes

The clock mode is determined by the setting of the sub-clock selection bit (SCS) and PLL clock selection bit (MCS) of the clock selection register (CKSCR).

● Transition from main clock mode to PLL clock mode

If the PLL clock selection bit (MCS) value is changed from "1" to "0" when the sub-clock selection bit (SCS) is "1", the mode will change from the main clock mode to the PLL clock mode after expiration of the PLL clock oscillation stabilization wait time (2^13/HCLK).

● Transition from PLL clock mode to main clock mode

If the PLL clock selection bit (MCS) value is changed from "0" to "1" when the sub-clock selection bit (SCS) is "1", the mode will change to the main clock mode at the point where the PLL clock and main clock edges coincide (i.e. after 1 to 8 PLL clocks).

● Transition from main clock mode to sub-clock mode

If the sub-clock selection bit (SCS) value is changed from "1" to "0", the mode will change from the main clock mode to the sub-clock mode in such a manner that synchronization with the sub-clock is achieved (approx. 130µs).

Note: When turning on the power or canceling the hardware standby mode or stop mode, the sub-clock oscillation stabilization wait time (approx. 2 seconds) is generated. In the meantime, when switching from the main clock mode to the sub-clock mode, the oscillation stabilization wait time is generated.
● Transition from sub-clock mode to main clock mode

If the sub-clock selection bit (SCS) value is changed from "0" to "1", the mode changes from the sub-clock mode to the main clock mode after expiration of the main clock oscillation stabilization wait time.

● Transition from PLL clock mode to sub-clock mode

If the sub-clock selection bit (SCS) value is changed from "1" to "0", the mode will change from the PLL clock mode to the sub-clock mode as soon as a sub-clock edge is detected.

● Transition from sub-clock mode to PLL clock mode

If the sub-clock selection bit (SCS) value is changed from "0" to "1" when the PLL clock selection bit (MCS) is "0", the mode will change to the PLL clock mode after expiration of the main clock oscillation stabilization wait time and then the PLL clock oscillation stabilization wait time \((2^{13}/HCLK)\) is performed.

| Note: | Resetting the sub-clock selection bit (SCS) and the PLL clock selection bit (MCS) does not enable immediate switching of the machine clock. To operate peripheral functions that depend on the machine clock, first check that machine clock switching has been completed by referencing the sub-clock operation underway bit (SCM) and the PLL clock operation underway bit (MCM) in the clock selection register (CKSCR), and then operate peripheral functions. |

### Selection of PLL Clock Multiplication Rate

By setting the CS1 and CS0 bits of the clock selection register (CKSCR) to appropriate values ("00B" to "11B"), the most suitable PLL clock multiplication rate can be selected from the 4 PLL clock multipliers (1, 2, 3 and 4).

### Machine clock

As the machine clock, one clock can be selected from the 3 different clocks; main clock, PLL clock or sub-clock. The selected clock (machine clock) is supplied to the CPU and peripheral functions.

The setting of PLL clock selection bit (MCS) and sub-clock selection bit (SCS) in the clock selection register (CKSCR) determine which one of the 3 different clocks (main clock, PLL clock and sub-clock) is to be used as the machine clock.

| Note: | When transiting a clock mode, do not transit a clock mode to any other clock mode or a low-power consumption mode until the completion of transition. Refer to the MCM and SCM bits in the clock selection register (CKSCR) to check that the transition of clock mode is completed. If the mode is switched to another clock mode or low-power-consumption mode before completion of switching, the mode may not be switched. |
3.7.4 Machine Clock Selection

Machine clock selection uses the SCS, MCS, CS1 and CS0 bits of the clock control register (CKSCR).

- Machine clock selection

  Figure 3.7-4 shows state transmission by changing machine clock.

  For simplicity, events that occur simultaneously are shown as a sequence of events in the status change diagram. Actual status changes take place instantly.
Figure 3.7-4 Clock Mode Transition Diagram

- Power on reset
- Hardware standby mode cancellation
- Watchdog reset

**Main stabilization**

**Main clock mode**
- SCS=1 MCM=1
- SCM=1 MCM=1
- MainO PLL x SubO

**PLL → Main**
- SCS=1 MCM=1
- SCM=1 MCM=0
- MainO PLL x SubO

**[PLL → Main] transition**
- SCS=1 MCM=1
- SCM=1 MCM=0
- MainO PLL x SubO

**PLL stabilization**

**PLL clock mode**
- SCS=1 MCM=1
- SCM=1 MCM=0
- MainO PLL x SubO

**[PLL → Sub] transition**
- SCS=0 MCM=0
- SCM=1 MCM=1
- MainO PLL x SubO

**Sub clock mode**
- SCS=0 MCM=0
- SCM=0 MCM=1
- MainO PLL x SubO

**[Sub → PLL] transition**
- SCS=0 MCM=0
- SCM=0 MCM=1
- MainO PLL x SubO

**[Sub → P PLL] transition**
- SCS=1 MCM=0
- SCM=0 MCM=1
- MainO PLL x SubO

**[Sub → Main] transition**
- SCS=1 MCM=1
- SCM=0 MCM=1
- MainO PLL x SubO

**Reset * / Interruption**
- SCS=0
- SCM=1 MCM=1
- MainO PLL x SubO

**Main oscillation stabilization waiting**
- SCS=0
- SCM=0 MCM=1
- MainO PLL x SubO

**Power on reset**
- SCS=1 MCM=1
- SCM=1 MCM=1
- MainO PLL x SubO

**Sub mode**
- SCS=0 MCM=0
- SCM=0 MCM=1
- MainO PLL x SubO

**Main mode**
- SCS=1 MCM=1
- SCM=1 MCM=1
- MainO PLL x SubO

**Main stabilization**

Main : Main clock
Sub : Sub clock
PLL : PLL clock
O : Operation
x : Stop

* : Reset means external reset and software reset, and excludes power-on reset, reset by release of hardware standby mode and watchdog reset.

**Notes:**
- The reset value for the machine clock is the main clock (CKSCR:SCS=1, MCS=1).
- The SCS bit and the MCS bit are initialized to the state in which the main clock is selected at a reset at a power on reset, the cancellation of the hardware standby or the watchdog function. (SCS=1, MCS=1)
- Not initialized by an external reset or a software reset.
3.7.5 Oscillation Stabilization Wait Time

In the following case, the main clock oscillation stabilization wait time is necessary after the start of oscillation because the oscillation of the oscillation clock (HCLK) is disabled.

- Power on reset
- Cancellation of stop mode/watch mode/hardware standby mode
- Watchdog function-based reset

To cancel the stop mode, a waiting time (sub-clock oscillation stabilization time) is necessary because the sub-clock is disabled.

In the case of a switching from the main clock to the PLL clock, the oscillation stabilization wait time is necessary after the start of PLL oscillation.

### Oscillation Stabilization Wait Time

A ceramic or crystal oscillator generally requires several milliseconds to several tens of milliseconds between the start of oscillation and the stabilization of oscillation to the natural frequency (oscillation frequency).

Therefore, it is necessary to prohibit the operation of the CPU immediately after the start of oscillation and supply the clock to the CPU only after the oscillation stabilization wait time has elapsed and sufficient stabilization of oscillation has been achieved.

- It is necessary to select a oscillation stabilization wait time appropriate to an oscillator to be used because the oscillation stabilization time varies with the type of oscillator (ceramic, crystal, etc.).
- During a switching from the main clock to the PLL clock, the CPU operates on the main clock while waiting for the oscillation to stabilize. The CPU clock is then switched to the PLL clock.
- When turning on the power or canceling the hardware standby mode or stop mode, the sub-clock oscillation stabilization wait time (approx. 2 seconds) is generated. In the meantime, when switching from the main clock mode to the sub-clock mode, the oscillation stabilization wait time is generated.

Figure 3.7-5 shows operation immediately after oscillation starts.

#### Figure 3.7-5 Operation Immediately after Oscillation Starts

![Oscillation Stabilization Wait Time Diagram](image)

**Note:** Returning from the sub watch mode through the external reset is not possible. For details, see Section "3.8.7 Watch Mode".
CHAPTER 3 CPU FUNCTION

Main Clock Oscillation Stabilization Wait Time

To start operation in the main clock mode or PLL clock mode when the oscillation clock (HCLK) is disabled, it is necessary to have the main clock oscillation stabilization time.

- The main clock oscillation stabilization wait time expires when the timebase timer counter value is counted up from "0" and becomes the time reached the specified oscillation stabilization wait time.
- For hardware standby cancellation/watchdog function-based reset, the main clock oscillation stabilization wait time is specified by the WS1 and WS0 bits of the clock selection register (CKSCR).
  For power on reset, the main clock oscillation stabilization wait time is fixed at $2^{18}/HCLK$ or $2^{17}/HCLK$.

Oscillation stabilization wait time of sub-clock

To start operation in the sub-clock mode when the sub-clock is disabled, it is necessary to have the sub-clock oscillation stabilization wait time.

The sub-clock oscillation stabilization time is fixed at $2^{14}/SCLK$ and is counted by a watch timer.

Oscillation stabilization wait time of PLL clock

To start operation in the PLL clock mode when the PLL clock is disabled, it is necessary to have the PLL clock oscillation stabilization wait time.

The PLL clock oscillation stabilization time is fixed at $2^{13}/HCLK$ and is counted by a timebase timer.
3.7.6 Connection of Oscillator and External Clock

The MB90520A series products have built-in clock generation circuit. Clock signals can be generated by connecting an external oscillator. It is also possible to input an externally generated clock.

■ Connection of Oscillator and External clock

● Example of connection of crystal oscillator or ceramic oscillator

When connecting a crystal or ceramic oscillator, make sure that it is connected as shown in Figure 3.7-6.

Figure 3.7-6 Example of connection of crystal oscillator or ceramic oscillator

● Example of connection of external clock

Connect the external clock to the X0 (X0A) pin and leave the X1 (X1A) pin open. Figure 3.7-7 shows a connection example.

Figure 3.7-7 Example of connection of external clock
3.8 Low-power Consumption Mode

The MB90520A series products have the following CPU operation modes:
- Operation modes by 3 different types of operation clocks
- CPU intermittent operation mode
- Standby mode
The modes other than the PLL clock mode are low-power consumption modes.

Low-power Consumption Mode

Figure 3.8-1 shows the relationships between the CPU operation mode and current consumption.

Figure 3.8-1 CPU Operation Modes and Current Consumption

This figure shows the image of each mode, so some parts are different from actual consumption current.
CHAPTER 3 CPU FUNCTION

### Operation modes by 3 different types of clocks

- **PLL clock mode**
  
  The CPU and peripheral functions are operated on a PLL clock generated by multiplying the oscillation clock (HCLK).

- **Main clock mode**
  
  The CPU and peripheral functions are operated on a main clock divided by 2 the oscillation clock (HCLK) supplied from the high-speed oscillation pins.
  
  Because the PLL multiplication circuit is disabled, power consumption is reduced.

- **Sub-clock mode**
  
  The CPU and peripheral functions are operated on a sub-clock (SCLK) divided by 4 internally turning the clock supplied from the low-speed oscillation pins.
  
  Power consumption is lower than that in the main clock mode, because the operation is on a low speed sub-clock with the oscillation clock (HCLK) disabled.

| Reference: | For the clock mode, see "Section 3.7 Clock". |

### CPU Intermittent operation mode

This is a reduced-power consumption mode that uses the CPU intermittently while supplying an operation clock to built-in peripheral functions to allow them to operate normally.

- In the CPU intermittent operation mode, the CPU operation clock is temporarily stopped when the CPU accesses a register, built-in memory or built-in peripheral functions to slow down the execution cycle.

- The available modes are the PLL clock intermittent operation mode, main clock operation mode and sub-clock intermittent operation mode.

### Standby Mode

In the standby mode, the supply of the operation clock to the CPU is disabled regardless of which mode is being used. In addition, power consumption is reduced by disabling the supply of the operation clocks to all components except certain peripheral functions or by disabling all operation clocks.

- **PLL sleep mode**
  
  In the PLL sleep mode, the PLL clock is used as the machine clock and only the CPU operation clock is disabled.
  
  All components except the CPU operate on the PLL clock.

- **Main sleep mode**
  
  In the main sleep mode, the main clock is used as the machine clock and only the CPU operation clock is disabled.
  
  All components except the CPU operate on the main clock.
CHAPTER 3  CPU FUNCTION

● Sub sleep mode

  In the sub sleep mode, the sub-clock is used as the machine clock and only the oscillation clock (HCLK) and the CPU operation clock are disabled.
  All components except the CPU operate on the sub-clock.

● Timebase timer mode

  In the timebase timer mode, only the oscillation clock (HCLK), sub-clock (SCLK), timebase timer and watch timer operate.
  The CPU and peripheral functions are disabled.

● Watch mode

  In the mode, only the sub-clock and watch timer operate.
  The oscillation clock (HCLK) is disabled and the CPU and peripheral functions are disabled.
  Returning from the sub watch mode through the external reset is not possible.
  For details, see Section "3.8.7 Watch Mode".

● Stop mode

  In the stop mode, the oscillation clock (HCLK) and sub-clock are disabled.
  All operation clocks are disabled.
  The stop mode is the least-power-consuming mode because it disables all operation clocks.

● Hardware standby mode

  The hardware standby mode disables all oscillation functions and put all input and output pins into the high impedance state.
  • The hardware standby mode continues as long as "L" level signals are being input to the HST pins and can only be cancelled by inputting "H" level signals to the HST pins.
  • Like the stop mode, the hardware standby mode is the least-power-consuming mode because it disables all operation clocks.

Note: When transiting a clock mode, do not transit a clock mode to any other clock mode or a low-power consumption mode until the completion of transition. Refer to the MCM and SCM bits in the clock selection register (CKSCR) to check that the transition of clock mode is completed. If the mode is switched to another clock mode or low-power-consumption mode before completion of switching, the mode may not be switched.
3.8.1 Block Diagram of the Low-power Consumption Control Circuit

The following is a block diagram of the low-power consumption control circuit.

**Block Diagram of the Low-power consumption control circuit**

![Block Diagram of the Low-power consumption control circuit](image-url)
CHAPTER 3  CPU FUNCTION

- CPU intermittent operation selector

  Specifies the number of suspended clock for the CPU intermittent operation mode.

- Standby controller circuit

  This controller causes the CPU clock controller circuit and peripheral clock controller circuit to switch between the CPU operating clock and the peripheral functions operating clock, and to transit and cancel the standby mode.

- CPU clock controller circuit

  This controller supplies an operating clock to the CPU.

- Peripheral clock controller circuit

  This circuit supplies the operation clock to the peripheral functions.

- Pin high-impedance controller circuit

  This circuit is used to put the input and output pins into the high-impedance state in the timebase timer, watch or stop mode.

  For pins selected as "the pull-up resistances", pull-up resistances are detached in the stop and hardware standby modes.

- Internal reset generation circuit

  This circuit generates internal reset signals.

- Low-power consumption mode control register (LPMCR)

  This register is used to make a shift to the standby mode or cancel the standby mode and to make CPU intermittent operation function settings.
3.8.2 Low-power Consumption Mode Control Register (LPMCR)

The low-power consumption mode control register (LPMCR) is used to make a shift to the low-power consumption mode or cancel the low-power consumption mode and to specify the number of CPU clock suspension cycles for the CPU intermittent operation mode.

![Configuration of the low-power consumption mode control register (LPMCR)](image)

- **SSR**
  - 0: Specifies not perform the self refresh control of the DRAMC in the main/PLL sleep mode, clock mode and the stop mode.
  - 1: Specifies perform the self refresh control of the DRAMC in the main/PLL sleep mode, clock mode and the stop mode.

- **CG0 CG1**
  - CPU halt cycle count select bit
    - 0 0: 0 cycle (CPU clock=Peripheral clock)
    - 0 1: 9 cycle (CPU clock: Peripheral clock=1: Approx.3 to 4)
    - 1 0: 17 cycle (CPU clock: Peripheral clock=1: Approx.5 to 6)
    - 1 1: 33 cycle (CPU clock: Peripheral clock=1: Approx.9 to 10)

- **TMD**
  - Watch mode bit
    - 0: Change Clock mode
    - 1: No effect

- **RST**
  - Internal reset signal generation bit
    - 0: Generate Internal reset signal of 3 machine cycle
    - 1: No effect

- **SPL**
  - Pin state specification bit (in Timebase timer · watch · stop mode)
    - 0: Hold state preserved
    - 1: High impedance

- **SLP**
  - Sleep mode bit
    - 0: No effect
    - 1: Change Sleep mode

- **STP**
  - Stop mode bit
    - For PLL clock mode
      - 0: No effect
      - 1: Change Timebase timer mode
    - For Main clock mode
      - 0: No effect
      - 1: Change Stop mode
    - For Sub clock mode
      - 0: No effect
      - 1: Change Stop mode

**Reset value**
00011000B
### Table 3.8-1 Function of low-power consumption mode control register (LPMCR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| **bit0** | **SSR:** Self refresh control bit | Specifies whether or not to perform the self refresh control of the DRAMC in the main/PLL sleep mode, watch mode and the stop mode.  
* **When the bit is set to "0":** No self refresh control is performed.  
* **When the bit is set to "1":** Performs the self refresh control of the DRAMC.  
**Note:**  
This function is invalid when there is no built-in DRAMC. |
| **bit2** | **CG1, CG0:** CPU halt cycle count selection bits | Specifies the number of temporary stops during the cycle of the CPU operating clock in the CPU intermittent operation.  
* Initialized to "00b" only by a reset by a power on, the cancellation of the hardware standby or the watchdog function. |
| **bit3** | **TMD:** Watch mode bit | Transition to watch mode  
* **When the bit is set to "0":** Transit to the watch mode.  
* **When the bit is set to "1":** No effect.  
* Set to "1" by the cancellation of the watch mode or the stop mode.  
* Read value is always "0".  
**Note:**  
The sub watch mode cannot be canceled by the external reset (RST pin). For details, see Section "3.8.7 Watch Mode". |
| **bit4** | **RST:** Internal reset signal generation bit | Used to generate a software reset.  
* **When the bit is set to "0":** An internal reset signal for three machine cycles is generated.  
* **When the bit is set to "1":** No effect.  
* Read value is always "0". |
| **bit5** | **SPL:** Pin state specification bit | Sets the state of I/O pin in the watch mode, timebase timer mode and stop mode.  
* **When the bit is set to "0":** Retains the immediately preceding level of the I/O pin.  
* **When the bit is set to "1":** The I/O pin enter a high impedance state.  
* The bit is initialized to "0" at a reset. |
| **bit6** | **SLP:** Sleep mode bit | Transition to sleep mode  
* **When the bit is set to "0":** No effect.  
* **When the bit is set to "1":** Transit to the sleep mode.  
* Cleared to "0" by a reset or by the cancellation of the sleep mode or the stop mode.  
* If the SLP bit and the STP bit are set to "1" simultaneously, the STP bit takes precedence, and a transition to the timebase timer mode or the stop mode occurs.  
* Read value is always "0". |
| **bit7** | **STP:** Stop mode bit | Specifies a transition to the timebase timer mode or the stop mode.  
* **When the bit is set to "0":** Makes a transition to the timebase timer mode (during the PLL clock mode) or the stop mode (during the main clock mode and the sub-clock mode).  
* **When the bit is set to "1":** No effect.  
* Cleared to "0" by a reset or by the cancellation of the timebase timer mode or the stop mode.  
* Read value is always "0". |
Notes:

- To set up a transition to the low-power consumption mode in the low-power consumption mode control register, use the commands in Table 3.8-2.
- If a transition to the low-power consumption mode is caused by commands other than those listed in Table 3.8-2, the operation cannot be guaranteed.
- There is no limit to the commands that can be used when making settings other than the setting to make a transition to the low-power consumption mode.
- To write word length data into the low-power consumption mode control register, use even addresses. A transition to the low-power consumption mode caused by the writing of an odd address may result in erroneous operation.

Table 3.8-2 List of commands used to make a transition to the low-power consumption mode

<table>
<thead>
<tr>
<th>MOV  io,#imm8</th>
<th>MOV dir,#imm8</th>
<th>MOV eam,#imm8</th>
<th>MOV eam,Ri</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV  io,A</td>
<td>MOV dir,A</td>
<td>MOV addr16,A</td>
<td>MOV eam,A</td>
</tr>
<tr>
<td>MOVW @RLi+disp8,A</td>
<td>MOVW dir,#imm16</td>
<td>MOVW eam,#imm16</td>
<td>MOVW eam,RWi</td>
</tr>
<tr>
<td>MOVW io,#imm16</td>
<td>MOVW dir,A</td>
<td>MOVW addr16,A</td>
<td>MOVW eam,A</td>
</tr>
<tr>
<td>MOV  @RLi+disp8,A</td>
<td>MOVW dir,#imm16</td>
<td>MOVW eam,#imm16</td>
<td>MOVW eam,A</td>
</tr>
<tr>
<td>SETB io:bp</td>
<td>SETB dir:bp</td>
<td>SETB addr16:bp</td>
<td></td>
</tr>
<tr>
<td>CLRB io:bp</td>
<td>CLRB dir:bp</td>
<td>CLRB addr16:bp</td>
<td></td>
</tr>
</tbody>
</table>
3.8.3 CPU Intermittent Operation Mode

The CPU intermittent operation mode is a reduced-power consumption mode that uses the CPU intermittently while supplying an operation clock to built-in peripheral functions to allow them to operate normally.

- **CPU Intermittent operation mode**
  In the CPU intermittent operation mode, the clock supplied to the CPU is temporarily stopped when the CPU accesses a register, built-in memory or built-in peripheral function to delay the activation of the internal bus. Allows the CPU execution speed to be reduced while supplying a high-speed clock to built-in peripheral functions to reduce power consumption.
  - The number of cycles in which the supply of clock to the CPU suspension is specified by the CG1 and CG0 bits of the low-power consumption mode control register (LPMCR).
  - The command execution time in the CPU intermittent operation mode is calculated by adding the normal execution time to a correction value that is obtained by multiplying the number of times the CPU accesses registers, built-in memory and built-in peripheral functions by the number of suspension cycles.

Figure 3.8-4 shows clock for the CPU intermittent operation mode.

![Clock for the Intermittent CPU Operation Mode](image-url)
### 3.8.4 Standby Mode

In the standby mode, the CPU is disabled regardless of which mode is being used.

#### Operating State of Standby Mode

Table 3.8-3 shows operating state of standby mode.

<table>
<thead>
<tr>
<th>Standby Mode</th>
<th>Transition condition</th>
<th>Oscillation clock (HCLK)</th>
<th>Sub clock</th>
<th>Machine clock</th>
<th>CPU Peripheral function</th>
<th>Pin</th>
<th>Setting disabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep mode</td>
<td>Sub sleep mode</td>
<td>SCS=0 SLP=1</td>
<td>Stops</td>
<td>Operation</td>
<td>Retention of the immediately prior state</td>
<td>External reset or Interrupt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Main sleep mode</td>
<td>MCS=1 SCS=1 SLP=1</td>
<td></td>
<td>Operation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PLL sleep mode</td>
<td>MCS=0 SCS=1 SLP=1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timebase timer mode</td>
<td>Timebase timer mode (SPL=0)</td>
<td>MCS=0 SCS=1 SLP=1</td>
<td>Operation</td>
<td>Operation</td>
<td>Stops*1</td>
<td>Retention of the immediately prior state</td>
<td>External reset or interrupt*4</td>
</tr>
<tr>
<td></td>
<td>Timebase timer mode (SPL=1)</td>
<td>MCS=0 SCS=1 SLP=1</td>
<td></td>
<td></td>
<td></td>
<td>Hi-Z*3</td>
<td></td>
</tr>
<tr>
<td>Watch mode</td>
<td>Watch mode (SPL=0)</td>
<td>TMD=0</td>
<td>Stops</td>
<td></td>
<td>Stops*2</td>
<td>Retention of the immediately prior state</td>
<td>External reset or interrupt*5</td>
</tr>
<tr>
<td></td>
<td>Watch mode (SPL=1)</td>
<td>TMD=0</td>
<td></td>
<td></td>
<td></td>
<td>Hi-Z*3</td>
<td></td>
</tr>
<tr>
<td>Stop mode</td>
<td>Stop mode (SPL=0)</td>
<td>MCS=1 or SCS=0 STP=1</td>
<td>Stops</td>
<td></td>
<td>Stops</td>
<td>Retention of the immediately prior state</td>
<td>External reset or interrupt*6</td>
</tr>
<tr>
<td></td>
<td>Stop mode (SPL=1)</td>
<td>MCS=1 or SCS=0 STP=1</td>
<td></td>
<td></td>
<td></td>
<td>Hi-Z*3</td>
<td></td>
</tr>
<tr>
<td>Hardware Standby Mode</td>
<td>HST=L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Hi-Z</td>
<td>HST=H</td>
</tr>
</tbody>
</table>

*1: The timebase timer and watch timer operate.
*2: The watch timer operates.
*3: DTP/external interrupt input pin and wake-up interrupt input pin do operate.
*4: Timebase timer, watch timer and external interrupt
*5: Watch timer and external interrupt
*6: External interrupt

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR)
SLP: Sleep mode bit of low-power consumption mode control register (LPMCR)
STP: Stop mode bit of low-power consumption mode control register (LPMCR)
TMD: Watch mode bit of low-power consumption mode control register (LPMCR)
MCS: PLL clock selection bit in clock selection register (CKSCR)
SCS: sub-clock selection bit in the clock selection register (CKSCR)
Hi-Z: High impedance
HST: HST pin
**Note:** To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to 0. This applies to the following pins:
P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01
3.8.5 Sleep Mode

In a sleep mode, only the CPU operation clock is disabled. All components except the CPU continue operating. When the setting for shifting to the sleep mode has been made (LPMCR:SLP=1, STP=0), the shift to the sleep mode is made as follows depending on the clock selected as the machine clock:

PLL clock mode → PLL sleep mode
Main clock mode → Main sleep mode
Sub-clock mode → Sub-sleep mode

■ Transition to Sleep Mode

When the setting for shifting to the sleep mode has been made in the low-power consumption mode control register (LPMCR:SLP=1, STP=0), the shift to the sleep mode is made based on the SCS and MCS bits of the clock selection register (CKSCR).

Table 3.8-4 shows the relationship between the setting of the SCS and MCS bits of the clock selection register and the way the shift to the sleep mode is made.

<table>
<thead>
<tr>
<th>CKSCR: SCS</th>
<th>CKSCR: MCS</th>
<th>Sleep mode to be transited</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Sub-sleep mode</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>PLL sleep mode</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Main sleep mode</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Sub-sleep mode</td>
</tr>
</tbody>
</table>

Note: If the SLP bit and the STP bit in the low-power consumption mode control register (LPMCR) are set to "1" simultaneously, the STP bit takes precedence, and a transition to the stop mode or the timebase timer mode occurs. When the SLP bit is set to "1" and the TMD bit is set to "0" simultaneously, the TMD bit takes precedence and a transition to the watch mode occurs.

● Data retention function

In the sleep mode, the contents of the internal RAM and the dedicated registers (such as the accumulator) are retained.

● Operation when interrupt request generated

If an interrupt request occurs when the SLP bit of the low-power consumption mode control register (LPMCR) is set to "1", no shift to the sleep mode takes place.

If the CPU is not in a state to accept interrupt requests, the following command is executed. If the CPU is in a state to accept interrupt requests, a branching to the interrupt processing routine occurs immediately.
CHAPTER 3 CPU FUNCTION

● Pin state

During the sleep mode, the input and output pins stay in the states they were in before the shift to the sleep mode.

■ Cancellation of sleep modes

The low-power consumption control circuit cancels the sleep mode when a reset factor event or an interruption occurs.

● Return by reset factor

In the case of a sleep mode cancellation caused by a reset factor, a shift to the reset operation takes place after the cancellation.

● Return by interrupt

If an interruption request whose interruption level has been set to a level that is higher than 7 using the interrupt control register (ICR) from a peripheral function etc. occurs during the sleep mode, the sleep mode is cancelled.

After the cancellation of the sleep mode, the interruption request is assessed in the same manner as in the normal interruption processing, based on the condition code register (CCR) I flag, interruption level mask register (ILM) and interrupt control register (ICR) settings.

• If the interruption can be accepted, the interruption processing takes place.

• If the interruption cannot be accepted, the processing resumes from the command subsequent to the command for which the sleep mode setting was made.

Figure 3.8-5 shows how the sleep mode is cancelled by an interruption.
Figure 3.8-5  Release of Sleep Mode by Interrupt Occurrence

Reference:  Return from the PLL sleep mode by an external reset
- If an external reset is generated during the PLL sleep mode (the input of the "L" level signal to RST), the PLL clock selection bit in the clock selection register (CKSCR) remains in the state where the PLL clock mode is selected. (CKSCR: MCS=0)
- Since the main clock, sub-clock and PLL clock are oscillating in the PLL sleep mode, the PLL clock is used immediately after the cancellation of the PLL sleep mode to make a return by an external reset.

Figure 3.8-6  shows a return operation from the PLL sleep mode by an external reset.

Figure 3.8-6  Release of Sleep Mode (External Reset)
3.8.6 Timebase Timer Mode

The timebase timer mode is a mode that allows only the watch timer and timebase timer to function by allowing the oscillation clock and sub-clock to operate. All other functions are disabled.

■ Transition to Timebase Timer Mode

If the STP bit of the low-power consumption mode control register (LPMCR) is set to "1" when the current mode is the PLL Clock mode (CKSCR:SCS=1, MCS=0), a shift to the timebase timer mode occurs.

- Data retention function

In the timebase timer mode, the contents of the internal RAM and the dedicated registers (such as the accumulator) are retained.

- Operation when interrupt request generated

If an interrupt request occurs when the STP bit of the low-power consumption mode control register (LPMCR) is set to "1", no shift to the timebase timer mode takes place.

If the CPU is not in a state to accept interrupt requests, the following command is executed. If the CPU is in a state to accept interrupt requests, a branching to the interrupt processing occurs immediately.

- Pin state

Whether to keep the input and output pins in the previous state during the timebase timer mode or put them in the high-impedance state upon entry into the timebase timer mode can be specified using the pin state selection bit (SPL) of the low-power consumption mode control register (LPMCR)

Note: To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to "0".

This applies to the following pins:
P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01
■ Cancellation of timebase timer mode

The low-power consumption control circuit cancels the timebase timer mode when a reset factor or an interruption occurs.

In the case of a return from the timebase timer mode by an external reset input, a shift to the PLL clock mode occurs after the timebase timer mode is cancelled and the PLL clock oscillation stabilization wait time expires.

In the case of operation on the main clock, set the PLL clock selection bit (CKSCR:MCS=1) to cause the shift to the main clock mode.

● Return by reset factor

In the case of a timebase timer mode cancellation caused by a reset factor, a shift to the reset operation takes place after the cancellation.

Return by an external reset

In the case of an external reset, the PLL clock mode selection state is maintained.

If the PLL clock oscillation stabilization wait time has not yet expired at the time of the start of the reset sequence, the reset sequence is performed using the main clock.

Figure 3.8-7 shows a return operation from the timebase timer mode by an external reset.

Figure 3.8-7 Releasing of timebase timer mode (external reset)

<table>
<thead>
<tr>
<th>RST pin</th>
<th>Timebase timer mode</th>
<th>Main clock</th>
<th>Oscillating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timebase timer</td>
<td></td>
<td>Sub clock</td>
<td>Oscillating</td>
</tr>
<tr>
<td>PLL clock</td>
<td>Oscillation</td>
<td>CPU operation clock</td>
<td>Main clock</td>
</tr>
<tr>
<td>CPU operation</td>
<td>Stop</td>
<td>CPU operation</td>
<td>Reset sequence</td>
</tr>
</tbody>
</table>

● Return by interrupt

If an interruption request whose interruption level has been set to a level that is higher than 7 using the interrupt control register (ICR) from the watch timer, timebase timer or an external interruption occurs during the timebase timer mode, the timebase timer mode is cancelled.

After the cancellation of the timebase timer mode, the interruption request is assessed in the same manner as in the normal interruption processing, based on the condition code register (CCR) I flag, interruption level mask register (ILM) and interrupt control register (ICR) settings.

• If the interruption can be accepted, the interruption processing takes place.

• If the interruption cannot be accepted, the processing resumes from the command subsequent to the command for which the timebase timer mode setting was made.
3.8.7 Watch Mode

In the watch mode, only the sub-clock and watch timer operate. All functions except the watch timer is disabled.

■ Transition to Watch mode

If the watch mode bit (TMD) of the low-power consumption mode control register (LPMCR) is set to "0", a shift to the watch mode occurs.

Depending on the clock mode before the shift to the watch mode occurs, a shift to the main watch mode, sub-clock mode or PLL clock mode occurs. In the case of a return from the watch mode caused by an interruption, a shift to the clock mode before the shift to the watch mode occurs.

The differences between the main watch mode, sub watch mode and PLL watch mode is that the operation mode after an interruption-based return becomes main clock mode, sub-clock mode, PLL clock mode. There is no difference with respect to watch mode operation.

It is not possible to return from the sub watch mode by means of an external reset input.

● Data retention function

In the watch mode, the contents of the internal RAM and the dedicated registers (such as the accumulator) are retained.

● Operation when interrupt request generated

If an interrupt request occurs when the TMD bit of the low-power consumption mode control register (LPMCR) is set to "0", no shift to the watch mode takes place.

If the CPU is not in a state to accept interrupt requests, the following command is executed. If the CPU is in a state to accept interrupt requests, a branching to the interrupt processing occurs immediately.

● Pin state

Whether to keep the input and output pins in the previous state during the watch mode or put them in the high-impedance state upon entry into the watch mode can be specified using the pin state selection bit (SPL) of the low-power consumption mode control register (LPMCR).

Note: To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to "0".

This applies to the following pins:

P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01
### Cancellation of watch mode

The low-power consumption control circuit cancels the main watch and the PLL watch mode when a reset factor or an interruption occurs. The sub watch mode is cancelled when the external interrupt or the watch interrupt occurs.

Also, using the HST pin, the watch mode can be cancelled by using the hardware standby cancellation reset.

#### Return by reset factor

In the case of a main watch mode or PLL watch mode cancellation caused by a reset factor, a shift to the reset operation takes place after the cancellation.

If the sub watch mode is cancelled by the hardware standby cancellation reset, the sub watch mode should be cancelled and the device should be to reset operation after oscillation stabilization wait time.

The reset sequence is used from using sub clock.

**Return by an external reset**

- From sub watch mode, can not be returned by the external reset
- If the device is returned from main watch mode or PLL watch mode, the device should be to the main clock oscillation wait status after releasing the watch mode.
- If the device is returned from PLL watch mode, selected PLL clock mode status stack at the external reset. (CMSCR: MCS = 0)

The reset sequence is used from using sub clock. The device should start from PLL clock after the main clock oscillation stabilization wait time + the PLL clock oscillation stabilization wait time.

Figure 3.8-8 shows a return operation from the watch timer mode by an external reset from PLL watch mode.

---

**Figure 3.8-8 Releasing of PLL watch timer mode (external reset)**

<table>
<thead>
<tr>
<th>RST pin</th>
<th>Watch mode</th>
<th>Main clock</th>
<th>Sub clock</th>
<th>PLL clock</th>
<th>CPU clock</th>
<th>CPU operation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Oscillating</td>
<td>Oscillating</td>
<td>Oscillating</td>
<td>Sub clock</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Main clock</td>
<td>Reset sequence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PLL clock</td>
<td>Normal processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Oscillation stabilization wait
Return by interrupt

If an interruption request whose interruption level has been set to a level that is higher than 7 using the interrupt control register (ICR) from the watch timer or an external interruption occurs in the watch mode, the watch mode is cancelled.

In the case of a return from the PLL/Main watch mode, the interruption request is assessed in the same manner as in the normal interruption processing, based on the condition code register (CCR) I flag, interruption level mask register (ILM) and interrupt control register (ICR) settings after expiration of the main clock oscillation stabilization wait time after the cancellation of the watch mode.

In the case of a return from the sub watch mode, the interruption request is assessed immediately after the cancellation of the watch mode in the same manner as in the normal interruption processing.

- If the interruption can be accepted, the interruption processing takes place.
- If the interruption cannot be accepted, the processing resumes from the command subsequent to the command for which the watch mode setting was made.
3.8.8 Stop Mode

The stop mode stops all functions including all oscillation functions. Thus the stop mode allows data to be retained with lowest power consumption.

■ Transition to stop mode

If the STP bit of the low-power consumption mode control register (LPMCR) is set to "1" when the current mode is the main clock mode (CKSCR:SCS=1, MCS=1) or sub-clock mode (CKSCR:SCS=0), a shift to the stop mode occurs.

Depending on the clock mode before the shift to the stop mode occurs, a shift to the main stop mode or sub-stop mode occurs. In the case of a return the stop mode caused by an interruption or a reset factor, a shift to the clock mode before the shift to the stop mode occurs.

● Data retention function

In the stop mode, the contents of the internal RAM and the dedicated registers (such as the accumulator) are retained.

● Operation when interrupt request generated

If an interrupt request occurs when the STP bit of the low-power consumption mode control register (LPMCR) is set to "1", no shift to the stop mode takes place.

If the CPU is not in a state to accept interrupt requests, the following command is executed. If the CPU is in a state to accept interrupt requests, a branching to the interrupt processing occurs immediately.

● Pin state

Whether to keep the input and output pins in the previous state during the stop mode or put them in the high-impedance state upon entry into the stop mode can be specified using the pin state selection bit (SPL) of the low-power consumption mode control register (LPMCR).

Note:

To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to "0".

This applies to the following pins:
P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01
CHAPTER 3  CPU FUNCTION

■ Release of the stop mode

The low-power consumption control circuit cancels the stop mode when a reset factor or an interruption (external interruption/wake-up interruption) occurs.

The transition to the main clock oscillation stabilization wait time occurs, after a return from the stop mode, because the operation clock oscillation is disabled. After expiration of the main clock oscillation stabilization wait time, the stop mode is cancelled.

● Return from the main stop mode by an external reset

In the case of a stop mode cancellation by an external reset, the stop mode is cancelled after expiration of the main clock oscillation stabilization wait time.

The reset sequence is performed after expiration of the oscillation stabilization wait time.
Return from the Sub Stop mode by an external reset

An external reset causes a shift to the sub-clock oscillation stabilization waiting state. After expiration of the sub-clock oscillation stabilization wait time, the reset sequence is performed with the sub-clock.

Figure 3.8-9 shows a return operation from the sub stop mode (SCS=0) by an external reset.

**Figure 3.8-9 Cancellation of the Sub Stop mode (SCS=0, external reset)**

Return by interrupt

If an interruption request whose interruption level has been set to a level that is higher than 7 using the interrupt control register (ICR) from an external interruption occurs during the stop mode, the stop mode is cancelled.

In the case of a return from the main stop mode, the interruption request is assessed in the same manner as in the normal interruption processing, based on the condition code register (CCR) I flag, interruption level mask register (ILM) and interrupt control register (ICR) settings after expiration of the main clock oscillation stabilization wait time after the cancellation of the stop mode.

In the case of a return from the sub stop mode, the interruption request is assessed in the same manner as in the normal interruption processing after expiration of the sub-clock oscillation stabilization wait time after the cancellation of the stop mode.

- If the interruption can be accepted, the interruption processing takes place.
- If the interruption cannot be accepted, the processing resumes from the command subsequent to the command for which the stop mode setting was made.
3.8.9 Hardware Standby Mode

The hardware standby mode disable all oscillation functions and puts all input and output pins in the high-impedance state. The hardware standby mode continues as long as "L" level signals are being input to the HST pins and can only be cancelled by inputting "H" level signals to the HST pins.

- Transition to Hardware Standby Mode
  A shift to the Hardware Standby mode can be made from any state by setting the HST pin to "L" level.

  - Data retention function
    In the Hardware Standby mode, the content of the internal RAM is retained, but the dedicated registers (such as the accumulator) are reset.

  - Pin state setting
    In the Hardware Standby mode, all input and output pins are in the high-impedance state.

- Cancellation of Hardware Standby Mode
  The Hardware Standby mode can only be cancelled by a "H" level input to the HST pin.
  - When the HST pin is set to "H" level, the low-power consumption control circuit accepts a reset by the Hardware Standby mode and a shift to the main clock oscillation stabilization waiting state occurs.
  
  After expiration of the main clock oscillation stabilization wait time, the reset sequence is performed with the main clock.
  
  Figure 3.8-10 shows a return operation from the hardware standby mode.

Figure 3.8-10 Hardware Standby mode operation (in the case of a shift from the PLL RUN mode)
3.8.10 State Transition Diagram

Transition diagram of operation status and the transition conditions in an MB90520A series product are shown.

■ State Transition Diagram

For simplicity, events that occur simultaneously are shown as a sequence of events in the status transition diagram.

Actual status changes take place instantly. If the status transition diagram indicates that when MCS= "1" and SLP= "1" are selected in the PLL clock mode, a "PLL → Main" shift occurs first and then a "PLL → Main → Sleep" shift occurs. In reality, however, an immediate shift from the PLL Clock mode occurs ("PLL → Main → Sleep").

The status transition diagram also indicates that when a reset is activated in the sub sleep mode, a shift to the sub mode occurs first and then a shift to the main oscillation stabilization waiting state occurs. In reality, however, an immediate shift from the sub sleep mode to the main oscillation stabilization waiting state occurs.
Figure 3.8-11 State transition diagram of Low-power Consumption Mode

- Power on reset
- Hardware standby mode cancellation
- Watchdog reset

Main stabilization

[Sub→Main] transition
SCS=1 MCS=1
SCM=0 MCM=0
Main O PLL× Sub O

[Main→Sub] transition
SCS=0 MCS=0
SCM=1 MCM=0
Main O PLL× Sub O

Main clock mode
SCS=1 MCS=1
SCM=1 MCM=0
Main O PLL× Sub O

PLL→Main
SCS=1 MCS=1
SCM=0 MCM=0
Main O PLL× Sub O

PLL stabilization

PLL clock mode
SCS=1 MCS=0
SCM=0 MCM=0
Main O PLL× Sub O

Sub clock mode
SCS=1 MCS=1
SCM=0 MCM=0
Main O PLL× Sub O

Main oscillation stabilization waiting
SCS=0 MCS=0
SCM=0 MCM=0
Main O PLL× Sub O

- Reset*

Main stabilization

Main : Main clock
SCS ... CKSCR: SCS
Sub : Sub clock
MCS ... CKSCR: MCS
PLL : PLL clock
SCM ... CKSCR: SCM
O : Operation
MCM ... CKSCR: MCM
× : Stop

* : Reset means external reset and software reset and excludes power-on reset, reset by release of hardware standby mode and watchdog reset.
Figure 3.8-12 Standby mode transition diagram (Returning to Main Clock Mode)

Main: Main clock
Sub: Sub clock
PLL: PLL clock
O: Operation
×: Stop

* : Reset means external reset and software reset and excludes power-on reset, reset by release of hardware standby mode and watchdog reset.
**Figure 3.8-13 Standby Mode transition Diagram (Returning to sub-clock Mode)**

- **Sub clock mode**
  - SCS=0
  - MCS=0 or 1
  - STP=0
  - SLP=1
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Sub stabilization**
  - TMD+1
  - STP+0
  - SLP+1

- **Main oscillation stabilization wait**
  - SCS=0
  - MCS=0 or 1
  - STP=0
  - SLP=0
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Reset**
  - SCS=1
  - MCS=1
  - STP=0
  - SLP=0
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Sub sleep mode**
  - SCS=0
  - MCS=0 or 1
  - STP=0
  - SLP=1
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Sub stop mode**
  - SCS=0
  - MCS=0 or 1
  - STP=1
  - SLP=0
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Main → Sub**
  - TMD+1
  - STP+0
  - SLP+1

- **[Main → Sub] transition**
  - SCS=0
  - MCS=0 or 1
  - STP=0
  - SLP=0
  - SCM=1
  - MCM=1
  - Main x PLL x Sub O

- **PLL → Main**
  - TMD+1
  - STP+0
  - SLP+0

- **[PLL → Sub] transition**
  - SCS=0
  - MCS=0 or 1
  - STP=0
  - SLP=0
  - SCM=1
  - MCM=0
  - Main x PLL x Sub O

- **PLL → Main**
  - TMD+1
  - STP+0
  - SLP+1

- **Sub stabilization**
  - STP<0
  - SLP<0
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

- **Reset**
  - SCS=1
  - MCS=1
  - STP=0
  - SLP=0
  - SCM=0
  - MCM=1
  - Main x PLL x Sub O

**Notes:**
- Main: Main clock
- SCS: CKSCR: SCS
- STP: LPMCR: STP
- MCS: CKSCR: MCS
- SLP: LPMCR: SLP
- SCM: CKSCR: SCM
- MCM: CKSCR: MCM
- O: Operation
- PLL: PLL clock
- Sub: Sub clock
- PLL: PLL clock
- Operation: MCM
- Stop: MCM
- *: Reset means external reset and software reset and excludes power-on reset, reset by release of hardware standby mode and watchdog reset.
Figure 3.8-14 Standby mode transition diagram (Return to the PLL Mode)

- **PLL clock mode**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

- **PLL sleep mode**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=1
  - TMD=1
  - MainO PLL x SubO

- **Timebase timer mode**
  - SCS=1
  - MCS=0
  - STP=1
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=1
- TMD=1
- MainO PLL x SubO

**Sub stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=1
- TMD=1
- MainO PLL x SubO

**PLL stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

- **Interruption/Reset**
  - TMD=1
  - STP=0
  - SLP=1

- **Main stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

- **PLL-pll stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

- **PLL-pll stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

- ** PLL-pll stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

- **Interruption/Reset**
  - TMD=0
  - STP=1
  - SLP=0

- **Main stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

- **Interruption/Reset**
  - TMD=0
  - STP=1
  - SLP=0

- **Main stabilization**
  - SCS=1
  - MCS=0
  - STP=0
  - SLP=0
  - TMD=1
  - MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**Main stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

**PLL-pll stabilization**
- SCS=1
- MCS=0
- STP=0
- SLP=0
- TMD=1
- MainO PLL x SubO

* : Reset means external reset and software reset and excludes power-on reset, reset by release of hardware standby mode and watchdog reset.

Main : Main clock
Sub : Sub clock
PLL : PLL clock
● : Operation
× : Stop
3.8.11 Pin State in Standby Mode, at Reset

Shows the states of the pins in the standby mode and in case of reset.

■ Pull-up resistance

For pins selected as "the pull-up resistances", pull-up resistances are detached in the stop (SPL=1) and hardware standby modes.

■ I/O Pin state

Table 3.8-5 I/O Pin state

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>In Standby Mode</th>
<th>At a reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>At sleep</td>
<td>Timebase timer, watch, stop</td>
</tr>
<tr>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>Hardware standby</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P07 P20 to P25, P27 P30 to P37 P40 to P47 P50 to P54 P60 to P67 P70 to P73</td>
<td>Immediately prior state held*2</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P00 to P06 (INT0 to 6) P26 (INT7) P10 to P17 (W10 to 7)</td>
<td>Input enabled*1</td>
<td>Input enabled*1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEG0 to 07</td>
<td>Immediately prior state held *2</td>
<td></td>
</tr>
<tr>
<td>P80 to P87 (SEG16 to 23) P90 to P97 (SEG24 to 31) PA0 to PA7 (SEG08 to 15)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P74 to P77 (COM0 to 3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1: When using as the output port, hold the immediately prior state at the sleep and timebase timer, watch and stop mode (SPL=0). In the timebase timer, watch and stop mode (SPL=1), becomes the output Hi-Z. Since input enable means the state in which the input function can be used, the selection of the presence of a pull-up or pull-down resistor or an external input is required.

*2: The retention of the immediately prior state means the continuous output of the state that was output immediately prior to switching to this mode or the disabling of inputs. The continuous output of the state that was output means that if built-in peripheral functions creating outputs are operating, the outputs are continuously made and that if outputs are created by ports, the outputs are retained. "Input disabled" means that the contents of a pin are not accepted internally, because the internal circuit is not in operation although operation of the input gate nearest to the pin is currently enabled.

*3: The input cutoff means the state in which the operation of the input gate immediately near the pin is prohibited, and the output Hi-Z means to set the pin driving transistor to the disabled state and the pin to the high-impedance state.
| Note: | To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to 0. This applies to the following pins: P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01 |
3.8.12 Precautions when Using Low-power Consumption Mode

Take notes of the following points on the use of the low-power consumption mode.

■ Transition to Standby Mode

When an interruption request has been issued to the CPU from a peripheral function, neither the setting of the STP and SLP bits of the low-power consumption mode control register (LPMCR) to "1" nor the setting of the TMD bit to "0" causes a shift to the standby mode (No shift to the standby mode takes place even after completion of the interruption processing).

Even during the processing of an interruption by the CPU, a shift to the standby mode can be made if the interruption request flag bit has been cleared and there is no other interruption request.

However, the input of "L" to the HST pin causes a shift to the hardware standby mode regardless of the current state.

■ Notes on the Transition to Standby Mode

To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode, or timebase timer mode, use the following procedure:

1. Disable the output of peripheral functions.
2. Set the SPL bit to "1", STP bit to "1", or TMD bit to "0" in the low-power consumption mode control register (LPMCR).

■ Cancellation of Standby Mode by Interrupt

• If an interruption request whose interruption level is higher than 7 from an operating peripheral function or an external interruption occurs during the sleep mode, watch mode, timebase timer mode or stop mode, the standby mode is cancelled. Whether or not the interrupt is accepted by the CPU is not related to this operation.

• After cancellation of the standby mode, the interruption request is assessed in the same manner as in the normal interruption processing.
  - If the interruption can be accepted, the interruption processing takes place.
  - If the interruption cannot be accepted, the processing resumes from the command subsequent to the command for which the standby mode setting was made.

Note: To prevent a branch to interrupt processing immediately after a return from the standby mode, measures such as the prohibition of interrupts are required before setting up the standby mode.

■ Notes on Setting Standby Mode

If the STP and SLP bits of the low-power consumption mode control register (LPMCR) are set to "1", the STP bit takes precedence. If the MCS bit of the clock selection register (CKSCR) is set to "0", a shift to the timebase timer mode occurs. If the MCS bit of the clock selection register (CKSCR) is set to "1", a shift to the stop mode occurs.
### Notes on cancelling Standby Mode
- The standby mode can be cancelled by inputting the interrupt request of the external interrupt that has been set before entering the standby mode. "H" level, "L" level, rising edge, and falling edge can be selected as interrupt requests.
- The Hardware Standby mode can only be cancelled by a "H" level input to the HST pin.

### Notes on cancelling timebase timer mode
In the case of a timebase timer mode cancellation, a shift to the PLL clock mode occurs after expiration of the PLL clock oscillation stabilization wait time. When the main clock is to be used for operation, change the MCS bit of the clock selection register (CKSCR) to "1" by means of a reset or interruption processing and cause a shift to the main clock mode.

To cancel the timebase timer mode with an external interruption, make the setting to use a external interruption request to "H" level request. The use of a "L" level request may result in a malfunction. It is not possible to cause a return by means of an edge request.

### Note on cancelling sub watch mode
The sub watch mode can be cancelled by the occurrence of the external or watch interrupt or inputting "L" → "H" level to the HST pin. The external reset cannot be cancelled ("L" level input to the RST pin).

### Oscillation Stabilization Wait Time
- **Oscillation stabilization wait time of main clock**
  A main clock oscillation stabilization wait time is necessary in the sub-clock mode, watch mode, stop mode or hardware standby mode because the main clock oscillation is disabled. The oscillation stabilization wait time of the main clock is set by the WS1 and WS0 bits in the clock selection register (CKSCR).

- **Oscillation stabilization wait time of sub-clock**
  In the sub stop mode, sub-clock oscillation stabilization wait time is necessary when returning because the sub-clock oscillation is disabled. The oscillation stabilization wait time of the sub-clock is fixed at $2^{14}/SCLK$ (SCLK: sub-clock).

- **Oscillation stabilization wait time of PLL clock**
  In the case of a shift from a state where the CPU operates on the main clock and the PLL clock is disabled to a mode in which the CPU or a peripheral function operates on the PLL clock, a shift to the PLL clock oscillation stabilization waiting state occurs first, and during the oscillation stabilization waiting state, the main clock is used for operation.
  The PLL clock oscillation stabilization wait time is fixed at $2^{13}/HCLK$ (HCLK: oscillation clock frequency).

### Transition of clock mode
When transiting a clock mode, do not transit a clock mode to any other clock mode or a low-power consumption mode until the completion of transition. Refer to the MCM and SCM bits in the clock selection register (CKSCR) to check that the transition of a clock mode is completed. If the mode is switched to another clock mode or low-power-consumption mode before completion of switching, the mode may not be switched.
CHAPTER 3  CPU FUNCTION

3.9 CPU Mode

F²MC-16LX has a CPU operation mode that specifies the CPU operation and a memory access mode to control the access area. The CPU mode is determined by the mode pin settings at the time of the reset and the mode data that has been mode-fetched.

■ CPU Mode setting

F²MC-16LX has a CPU operation mode that specifies the CPU operation and a memory access mode to control the access area. Figure 3.9-1 shows the CPU operation modes and memory access modes.

**Figure 3.9-1 Classification of CPU Modes**

<table>
<thead>
<tr>
<th>CPU Operation mode (Setting by mode pin)</th>
<th>Memory access mode (Setting by CPU mode data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Execution mode</td>
<td>Single - chip mode (Internal access)</td>
</tr>
<tr>
<td>Flash serial programming mode</td>
<td></td>
</tr>
<tr>
<td>Flash memory mode</td>
<td></td>
</tr>
</tbody>
</table>

The CPU mode is determined by the mode pin states at the time of the reset and the mode data.

● Mode Pins (MD2 to MD0)

The mode pins are used to specify the area accessed when the reset vector and mode data are read during the reset sequence.

In the case of a product with built-in memory, it is also possible to specify the CPU operation mode in which data etc. are written into the flash memory.

● Mode Data

The mode data is used to specify the memory access mode that is used after cancelling a reset. The mode data is automatically taken into the CPU by a mode fetch from the "FFFFDFH" address of the memory.

■ CPU operation modes

Three CPU operation modes are available; the "CPU Execution" mode, "Flash Serial Writing" mode and "Flash Memory" mode. The CPU operation mode to be used is specified by means of the mode pins (MD2 to MD0).

■ Memory Access Mode

The MB90520A series products can only be used in the single-chip mode because they do not have an external bus access function.

The memory access mode is specified by the mode pins (MD2 to MD0) and the M1 and M0 bits in the mode data.
3.9.1 Mode Pin

The mode pins are 3 external pins (MD2 to MD0). These are used to specify the area to be accessed when the reset vector and mode data are read.

**Mode Pin (MD2 to MD0)**

The CPU mode pins are used to specify the area to be accessed when the reset vector and CPU mode data are read. For the MB90520A series products, the "Internal Vector mode", in which the reset vector is read from the internal ROM, is specified.

In the case of a product with built-in flash memory, it is also possible to specify the CPU operation mode in which programs, etc. are written into the flash memory.

Table 3.9-1 shows the mode pin settings.

---

**Table 3.9-1 Setting of Mode Pins**

<table>
<thead>
<tr>
<th>MD2</th>
<th>MD1</th>
<th>MD0</th>
<th>Mode Name</th>
<th>Reset vectors Access area</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Setting disabled</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Internal vector mode</td>
<td>Internal</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Setting disabled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Flash serial writing mode*</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Flash memory mode*</td>
<td>-</td>
</tr>
</tbody>
</table>

Set 0 = VSS or 1 = VCC.

*: Models with internal write flash memory enable setting.

**Internal vector mode**

The reset vector and CPU mode data are read from built-in ROM. The CPU mode data is stored in the mode register and the reset vector is stored in the program counter and program bank register.

**Flash serial writing mode**

Flash serial writing cannot be performed to write data etc. only by making mode pin settings.

**Flash memory mode**

This mode is set when using a parallel writer.

Reference: For details on flash serial write operations, refer to "CHAPTER 25 CONNECTION EXAMPLE OF FLASH SERIAL PROGRAMMING".
3.9.2 Mode Data

The mode data is used to specify the memory access mode that is used after the reset sequence. The mode data is automatically taken into the CPU by a mode fetch.

CPU Mode Data

During the reset sequence, the mode data in the "FFFFDFH" address is taken into the mode register in the CPU. The CPU specifies the memory access mode by means of the mode data.

In the memory access mode, whether to access the internal or external memory and the bus width to access the external memory are specified. For the MB90520A series products, however, the single-chip mode setting to access only built-in memory can be made.

- The content of the mode register can be changed only in the reset sequence.
- The memory access mode specified by the mode data becomes effective after completion of the reset sequence.

Figure 3.9-2 shows configuration of mode data register.

![Figure 3.9-2 Configuration of Mode Data register](image)

Setting of mode pin and mode data

The memory access mode is specified by a combination of mode pins and mode data.

Table 3.9-2 shows setting of mode pin and mode data. Make sure that valid combinations of mode pins and mode data are used.

Table 3.9-2 Setting of CPU mode pin and CPU mode data

<table>
<thead>
<tr>
<th>Mode Pin</th>
<th>Mode Data</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD2</td>
<td>MD1</td>
<td>MD0</td>
</tr>
<tr>
<td>M1</td>
<td>M0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
| 0        | 0         | Single-chip mode

Set MD2, MD1 and MD0 to 0=Vss, 1=Vcc.
3.9.3 Memory Access Mode

In the memory access mode, whether to access the internal or external memory and the bus width to access the external memory are specified. For the MB90520A series products, however, the single-chip mode setting to access only built-in memory can be made.

■ Single-chip mode

In the Single-Chip mode, only the built-in ROM and built-in RAM are accessed.

● Memory space of Single-chip mode

Figure 3.9-3 shows memory map of single-chip mode.

![Memory map of Single-chip mode](image)

<table>
<thead>
<tr>
<th>Product</th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address 1</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
</tr>
<tr>
<td>Address 2</td>
<td>FF0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>—</td>
</tr>
</tbody>
</table>

If ROM mirror function is ineffective, refer to "Chapter 22  ROM mirror function selection module".

*: Address match detection function register area

Reference: For details on memory access area, refer to "3.1 Memory Space".
This chapter describes the function and operation of the I/O port.

4.1 Overview of I/O Ports
4.2 Assignment of I/O Port Register and Pins Shared with Flash Memory Writing Pins
4.3 Port 0
4.4 Port 1
4.5 Port 2
4.6 Port 3
4.7 Port 4
4.8 Port 5
4.9 Port 6
4.10 Port 7
4.11 Port 8
4.12 Port 9
4.13 Port A
CHAPTER 4 I/O PORT

4.1 Overview of I/O Ports

I/O ports can be used as general-purpose I/O ports (parallel I/O ports). Each port is used both for peripheral functions and for providing input/output pins.

I/O Port Function

The I/O ports enable the port data register (PDR) to output data to the I/O pins from the CPU and fetch signals input to the I/O pins. The port uses the port direction register (DDR) to set the I/O pins input/output direction in units of individual bits.

The following shows the function of each port, and the resources that it also serves as:

- Port 0: Used as general-purpose input/output port or for peripheral functions (external interrupt input pin)
- Port 1: Used as general-purpose input/output port or for peripheral functions (wake-up interrupt input pin)
- Port 2: Used as general-purpose input/output port for peripheral functions (input capture input, up-down counter/timer 0 input, external interrupt input or A/D converter start up trigger pin)
- Port 3: Used as general-purpose input/output port or for peripheral functions (clock monitor output, output compare output (unit 0) or PPG timer output pin)
- Port 4: Used as general-purpose input/output port or for peripheral functions (PPg timer output, UART input/output, I/O extended serial input/output (channel 1) pin)
- Port 5: Used as general-purpose input/output port or for peripheral functions (I/O extended serial input/output (channel 2), up-down counter/timer 1 input or D/A converter output pin)
- Port 6: Used as general-purpose input/output port or for peripheral functions (A/D converter input pin)
- Port 7: Used as general-purpose input/output port or for peripheral functions (reload timer input/output, output compare output (unit 1), LCD common output pin)
- Port 8: Used as general-purpose input/output port or for peripheral functions (LCD segment output pin)
- Port 9: Used as general-purpose input/output port or for peripheral functions (LCD segment output pin)
- Port A: Used as general-purpose input/output port or for peripheral functions (LCD segment output pin)
### Table 4.1-1 List of each port function

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Input Type</th>
<th>Output Type</th>
<th>Function</th>
<th>bit7</th>
<th>bit6</th>
<th>bit5</th>
<th>bit4</th>
<th>bit3</th>
<th>bit2</th>
<th>bit1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0</td>
<td>P00/INT0 to P07</td>
<td>CMOS (hysteresis)</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>INT6</td>
<td>INT5</td>
<td>INT4</td>
<td>INT3</td>
<td>INT2</td>
<td>INT1</td>
<td>INT0</td>
<td>-</td>
</tr>
<tr>
<td>Port 1</td>
<td>P10/WI0 to P17/WI7</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>W17</td>
<td>W16</td>
<td>W15</td>
<td>W14</td>
<td>W13</td>
<td>W12</td>
<td>W11</td>
<td>W10</td>
<td>-</td>
</tr>
<tr>
<td>Port 2</td>
<td>P20/IN0 to P27/ADTG</td>
<td>CMOS</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>ADTG</td>
<td>ZIN0/IN7</td>
<td>BIN0</td>
<td>AIN0</td>
<td>IN11</td>
<td>IN10</td>
<td>IN01</td>
<td>IN00</td>
</tr>
<tr>
<td>Port 3</td>
<td>P30 to P37/PPG01</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>PPG01</td>
<td>PPS00</td>
<td>OUT3</td>
<td>OUT2</td>
<td>OUT1</td>
<td>OUT0</td>
<td>CK0T</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Port 4</td>
<td>P40/PPG11 to P47/SCK1</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>SCK1</td>
<td>SOT1</td>
<td>SIN1</td>
<td>SCK0</td>
<td>SOT0</td>
<td>SIN0</td>
<td>PPG11</td>
<td>PPG10</td>
<td></td>
</tr>
<tr>
<td>Port 5</td>
<td>P54/DA1 to P50/SIN2/AIN1</td>
<td>CMOS</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>DA1</td>
<td>DA0</td>
<td>SCK2/ZIN1</td>
<td>SOT2/BIN1</td>
<td>SIN2/AIN1</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 6</td>
<td>P60/AN0 to P67/AN7</td>
<td>CMOS</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>AN7</td>
<td>AN6</td>
<td>AN5</td>
<td>AN4</td>
<td>AN3</td>
<td>AN2</td>
<td>AN1</td>
<td>AN0</td>
</tr>
<tr>
<td>Port 7</td>
<td>P70/TIN0/OUT4 to P77/COM3</td>
<td>CMOS</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>COM3</td>
<td>COM2</td>
<td>COM1</td>
<td>COM0</td>
<td>TOT1/OUT7</td>
<td>TIN1/OUT6</td>
<td>TOT0/OUT5</td>
<td>TIN0/OUT4</td>
</tr>
<tr>
<td>Port 8</td>
<td>P87/SEG23 to P96/SEG16</td>
<td>N-ch Open drain</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>SEG23</td>
<td>SEG22</td>
<td>SEG21</td>
<td>SEG20</td>
<td>SEG19</td>
<td>SEG18</td>
<td>SEG17</td>
<td>SEG16</td>
</tr>
<tr>
<td>Port 9</td>
<td>P90/SEG24 to P97/SEG31</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>SEG31</td>
<td>SEG30</td>
<td>SEG29</td>
<td>SEG28</td>
<td>SEG27</td>
<td>SEG26</td>
<td>SEG25</td>
<td>SEG24</td>
<td>-</td>
</tr>
<tr>
<td>Port A</td>
<td>PA0/SEG08 to PA7/SEG15</td>
<td>CMOS</td>
<td>General-purpose I/O ports</td>
<td>Resource</td>
<td>SEG15</td>
<td>SEG14</td>
<td>SEG13</td>
<td>SEG12</td>
<td>SEG11</td>
<td>SEG10</td>
<td>SEG9</td>
<td>SEG8</td>
</tr>
</tbody>
</table>

**Notes:**
- Port 6 is also used as analog input pin. If used as a general-purpose port, always set the corresponding analog input enable register (ADER) bit to "0". At a reset, the ADER bit is initialized to "FFH".
- Port 9 is a general-purpose I/O port of open drain output type.
- When port 9 is used as output port, connect the pull-up resistance to external.
4.2 Assignment of I/O Port Register and Pins Shared with Flash Memory Writing Pins

The list of registers related to I/O port setting and assignment of flash memory writing pin are shown below.

### Registers of I/O Ports

Table 4.2-1 shows registers of each port.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Read/Write</th>
<th>Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0 data register (PDR0)</td>
<td>R/W</td>
<td>000000H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>R/W</td>
<td>000001H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 2 data register (PDR2)</td>
<td>R/W</td>
<td>000002H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>R/W</td>
<td>000003H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>R/W</td>
<td>000004H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 5 data register (PDR5)</td>
<td>R/W</td>
<td>000005H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>R/W</td>
<td>000006H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 7 data register (PDR7)</td>
<td>R/W</td>
<td>000007H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 8 data register (PDR8)</td>
<td>R/W</td>
<td>000008H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 9 data register (PDR9)</td>
<td>R/W</td>
<td>000009H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port A data register (PDRA)</td>
<td>R/W</td>
<td>00000AH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Common pin switch register (LCDCMR)</td>
<td>R/W</td>
<td>00000BH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 0 direction register (DDR0)</td>
<td>R/W</td>
<td>000010H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 1 direction register (DDR1)</td>
<td>R/W</td>
<td>000011H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 2 direction register (DDR2)</td>
<td>R/W</td>
<td>000012H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 3 direction register (DDR3)</td>
<td>R/W</td>
<td>000013H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 4 direction register (DDR4)</td>
<td>R/W</td>
<td>000014H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 5 direction register (DDR5)</td>
<td>R/W</td>
<td>000015H</td>
<td>XXXX0000B</td>
</tr>
<tr>
<td>Port 6 direction register (DDR6)</td>
<td>R/W</td>
<td>000016H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 7 direction register (DDR7)</td>
<td>R/W</td>
<td>000017H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 8 direction register (DDR8)</td>
<td>R/W</td>
<td>000018H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 9 direction register (DDR9)</td>
<td>R/W</td>
<td>000019H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port A direction register (DDRA)</td>
<td>R/W</td>
<td>00001AH</td>
<td>00000000B</td>
</tr>
<tr>
<td>Analog input enable register (ADER)</td>
<td>R/W</td>
<td>00001BH</td>
<td>11111111B</td>
</tr>
<tr>
<td>Port 0 input resistance register (RDR0)</td>
<td>R/W</td>
<td>00008CH</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 1 input resistance register (RDR1)</td>
<td>R/W</td>
<td>00008DH</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 0 input resistance register (RDR4)</td>
<td>R/W</td>
<td>00008EH</td>
<td>00000000B</td>
</tr>
<tr>
<td>LCD control register (LCR1)</td>
<td>R/W</td>
<td>00006BH</td>
<td>00000000B</td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
### Assignment of flash memory writing pin

#### Table 4.2-2 Assignment of flash memory writing pin

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-Chip</td>
</tr>
<tr>
<td>P07 to P00</td>
<td>D07 to D00</td>
</tr>
<tr>
<td>P17 to P10</td>
<td>-</td>
</tr>
<tr>
<td>P27 to P20</td>
<td>A07 to A00</td>
</tr>
<tr>
<td>P30</td>
<td>A16</td>
</tr>
<tr>
<td>P31</td>
<td>CE</td>
</tr>
<tr>
<td>P32</td>
<td>OE</td>
</tr>
<tr>
<td>P33</td>
<td>FGM</td>
</tr>
<tr>
<td>P34</td>
<td>Unused</td>
</tr>
<tr>
<td>P35</td>
<td></td>
</tr>
<tr>
<td>P36</td>
<td></td>
</tr>
<tr>
<td>P37</td>
<td></td>
</tr>
</tbody>
</table>
4.3 Port 0

Port 0 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 0.

■ Configuration of Port 0

Port 0 consists of following factors.
- General-purpose I/O port / External interrupt input pin (P00/INT0 to P06/INT6)
- General-purpose I/O port (P07)
- Port 0 data register (PDR0)
- Port 0 direction register (DDR0)
- Port 0 input resistance register (RDR0)

■ Pins Assignment of Port 0

- Use port 0 by switching between the resource pin and the general-purpose I/O port.
- Since port 0 serves as resource pin, when used as a resource, port 0 cannot be used as general-purpose I/O port.
- When using port 0 as the input pin of the resource, set the pin corresponding to the resource in the DDR0 as an input port.

Table 4.3-1 shows pins assignment of port 0.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P00/INT0</td>
<td>P00</td>
<td>INT0</td>
<td>Input</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>P01/INT1</td>
<td>P01</td>
<td>INT1</td>
<td>Output</td>
<td>Hysteresis</td>
</tr>
<tr>
<td></td>
<td>P02/INT2</td>
<td>P02</td>
<td>INT2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P03/INT3</td>
<td>P03</td>
<td>INT3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P04/INT4</td>
<td>P04</td>
<td>INT4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P05/INT5</td>
<td>P05</td>
<td>INT5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P06/INT6</td>
<td>P06</td>
<td>INT6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P07</td>
<td>P07</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
Block Diagram of Pins of Port 0

Figure 4.3-1  Block Diagram of Pins of Port 0

Registers for Port 0

- The registers for port 0 are PDR0 and DDR0.
- Port 0 input resistance register (RDR0) connects or cuts off the pull-up resistance to the pin set in input port.
- The bits composing each register correspond to the pins of port 0 one-to-one.

Table 4.3-2 shows correspondence between port 0 registers and pins.

Table 4.3-2  Correspondence between Port 0 Registers and Pins

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0</td>
<td>PDR0, DDR0, RDR0 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P07 P06 P05 P04 P03 P02 P01 P00</td>
</tr>
</tbody>
</table>
### 4.3.1 Registers for Port 0 (PDR0, DDR0, RDR0)

The registers for port 0 are explained.

#### Function of Registers for Port 0

- **Port 0 data register (PDR0)**
  - Port 0 data register indicates the state of the pins.

- **Port 0 direction register (DDR0)**
  - The port 0 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- **Port 0 input resistance register (RDR0)**
  - Port 0 input resistance register (PDR0) sets to connect/cut off the pull-up resistance.
  - It is valid when the pin is set to the input port. When setting the bit corresponding to input port to "1" or "0", pull-up resistance is connected or cut off respectively. It is invalid when the pin is set to the output port and pull-up resistance is cut off.
Table 4.3-3 shows function of registers for port 0.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0 data register (PDR0)</td>
<td>0</td>
<td>The pin state is Low level.</td>
<td>Outputs L level if used as an output port.</td>
<td>R/W</td>
<td>000000H</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is High level.</td>
<td>Outputs H level if used as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 0 direction register (DDR0)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000010H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 0 input resistance register (RDR0)</td>
<td>0</td>
<td>The pull-up resistance sets to &quot;cut off&quot;.</td>
<td>Sets the pull-up resistance to cut off.</td>
<td>R/W</td>
<td>00008CH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pull-up resistance sets to &quot;connect&quot;.</td>
<td>Sets the pull-up resistance to connect.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate

Reference: When using as an input pin of peripheral function, set "0" to the bit of port 0 direction register (DDR0) corresponding to the input pin of peripheral functions.
4.3.2 Operation of Port 0

This section describes the operation of port 0.

■ Operation of Port 0

● Operation of output port
  - With the corresponding DDR0 register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDR0 register is retained in the PDR.
  - Output latch and then output to the pins as is.
  - Reading the PDR0 register allows the pin values (output level) to be read.
  - Setting of port 0 input resistance register (RDR0) is invalid.

Notes:
  - If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
  - To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.

● Operation of input port
  - With the corresponding DDR0 register bit set to "0", the port works as an input port.
  - If data is written to the PDR0 register, it is retained in the PDR0 output latch but not output to the pins.
  - If data is read the PDR0 register, it is read pin level ("L" or "H").
  - Setting of port 0 input resistance register (RDR0) is valid.

● Operation of resource input
  - The state of the pin that serves as the resource input is input to the resource.
  - When using as an input pin of peripheral function, set the DDR0 register bit corresponding to input pin of peripheral function to "0" (input port).

● Operation at reset
  - At CPU reset, value of the DDR0 register is initialized to "00H" (input port).
  - In a reset operation, the PDR0 register is not initialized. Therefore, if used as an output port, set the output data in the PDR0 register and then set the corresponding DDR0 register to output.

● Operation in stop mode, timebase timer mode or watch mode
  - When a transition to the stop mode or timebase timer mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

● Operation of hardware standby mode
  - When setting HST pin to L level and a transition to the hardware standby mode, the pins are set to
"high-impedance." This is because, irrespective of the DDR0 register’s value, the output buffer is forcibly set to "OFF."
However, even if setting \texttt{HST} pin to L level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.3-4 shows the pin state of port 0.

### Table 4.3-4 State of port 0 pins

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00/INT0</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected) When INT0 to INT6 is enabled, input is not intercepted. For details, refer to &quot;13.3.2 DTP/External Interrupt Enable Register (ENIR)&quot;.</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P01/INT1</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P02/INT2</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P03/INT3</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P04/INT4</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P05/INT5</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P06/INT6</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>P07</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
<td>Input cut off, and output becomes Hi-Z (Pull-up resistor disconnected)</td>
</tr>
</tbody>
</table>

Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z: High impedance
Port 1 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 1.

■ Configuration of Port 1
Port 1 consists of following factors.
- General purpose I/O port / Wake-up interrupt input pin (P10/WI0 to P17/WI7)
- Port 1 data register (PDR1)
- Port 1 direction register (DDR1)
- Port 1 input resistance register (RDR1)

■ Pin Assignment of Port 1
- Use Port 1 by switching between the resource pin and the general-purpose I/O port.
- Since port 1 serves as resource pin, when used as a resource, port 1 cannot be used as general-purpose I/O port.
- When using port 1 as the input pin of the resource, set the pin corresponding to the resource in the DDR1 as an input port.

Table 4.4-1 shows pin assignment of port 1.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>P10/WI0</td>
<td>P10</td>
<td>WI0</td>
<td>Input</td>
<td>CMOS Hysteresis</td>
</tr>
<tr>
<td></td>
<td>P11/WI1</td>
<td>P11</td>
<td>WI1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P12/WI2</td>
<td>P12</td>
<td>WI2</td>
<td></td>
<td>CMOS Hysteresis</td>
</tr>
<tr>
<td></td>
<td>P13/WI3</td>
<td>P13</td>
<td>WI3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P14/WI4</td>
<td>P14</td>
<td>WI4</td>
<td></td>
<td>CMOS Hysteresis</td>
</tr>
<tr>
<td></td>
<td>P15/WI5</td>
<td>P15</td>
<td>WI5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P16/WI6</td>
<td>P16</td>
<td>WI6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P17/WI7</td>
<td>P17</td>
<td>WI7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
**Block Diagram of Pins of Port 1**

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

- The registers for port 1 are PDR1 and DDR1.
- Port 1 input resistance register (RDR1) connects or cuts off the pull-up resistance to the pin set in input port.
- The bits composing each register correspond to the pins of port 1 one-to-one.
  
  Table 4.4-2 shows the correspondence between the port 1 registers and pins.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>PDR1, DDR1, RDR1 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P17 P16 P15 P14 P13 P12 P11 P10</td>
</tr>
</tbody>
</table>

**Registers for port 1**

- Standby control: Stop mode (SPL=1), timebase timer mode (SPL=1), Watch mode (SPL=1), Control of Hardware standby mode.
4.4.1 Registers for Ports 1 (PDR1, DDR1, RDR1)

The registers for port 1 are explained.

■ Function of Registers for Port 1

- **Port 1 data register (PDR1)**
  - Port 1 data register indicates the state of the pins.

- **Port 1 direction register (DDR1)**
  - The port 1 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- **Port 1 input resistance register (RDR1)**
  - Port 1 input resistance register (PDR1) sets to connect/cut off the pull-up resistance.
  - It is valid when the pin is set to the input port. When setting the bit corresponding to input port to "1" or "0", pull-up resistance is connected or cut off respectively. It is invalid when the pin is set to the output port and pull-up resistance is cut off.
Table 4.4-3 shows function of registers for port 1.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>0</td>
<td>The pin state is Low level.</td>
<td>Outputs L level if used as an output port.</td>
<td>R/W</td>
<td>000001H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is High level.</td>
<td>Outputs H level if used as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1 direction register (DDR1)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000011H</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1 input resistance register (RDR1)</td>
<td>0</td>
<td>Pull-up resistance sets to shut off.</td>
<td>Sets the pull-up resistance to cut off.</td>
<td>R/W</td>
<td>00008DH</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pull-up resistance sets to connect.</td>
<td>Sets the pull-up resistance to connect.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write  
X: Indeterminate

Reference: When using as an input pin of peripheral function, set "0" to the bit of port 1 direction register (DDR1) corresponding to the input pin of peripheral functions.
4.4.2 Operation of Port 1

The operation of port 1 is explained.

- **Operation of Port 1**
  - **Operation of output port**
    - With the corresponding DDR1 register bit set to "1", the port works as an output port.
    - When used as an output port, any data written to the PDR1 register is retained in the PDR output latch and then output to the pins as is.
    - Reading the PDR1 register allows the pin values (output level) to be read.
    - Setting of port 1 input resistance register (RDR1) is invalid.
  - **Operation of input port**
    - With the corresponding DDR1 register bit set to "0", the port works as an input port.
    - If data is written to the PDR1 register, it is retained in the PDR1 output latch but not output to the pins.
    - If data is read the PDR1 register, it is read pin level ("L" or "H").
    - Setting of port 1 input resistance register (RDR1) is valid.
  - **Operation of resource input**
    - The state of the pin that serves as the resource input is input to the resource. The pin which is also used for peripheral function input is input the pin state to peripheral functions.
    - When using as an input pin of peripheral function, set "0" to the bit of port 1 direction register (DDR1) corresponding to the input pin of peripheral functions.
  - **Operation at reset**
    - At CPU reset, value of the DDR1 register is initialized to "00H" (input port).
    - In a reset operation, the PDR1 register is not initialized. Therefore, if used as an output port, set the output data in the PDR1 register and then set DDR1 register corresponding to output to "1".
  - **Operation in stop mode, timebase timer mode or watch mode**
    - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."
Operation of hardware standby mode

- When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR1 register's value, the output buffer is forcibly set to "OFF."

However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance."

Table 4.4-4 shows state of port 1 pins.

Table 4.4-4  State of Port 1 Pins

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>P10/W10 to P17/W17</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z (Pull-up resistor disconnected)</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
4.5 Port 2

Port 2 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 1.

■ Configuration of Port 2

Port 2 consists of following factors.
- General-purpose I/O port/resource input pin (P20/IN00 to P27/ADTG)
- Port 2 data register (PDR2)
- Port 2 direction register (DDR2)

■ Pin Assignment of Port 2

- Use port 2 by switching between the resource pin and the general-purpose I/O port.
- Since port 2 serves as resource pin, when used as a resource, port 2 cannot be used as general-purpose I/O port.
- When using port 2 as the input pin of the resource, set the pin corresponding to the resource in the DDR2 as an input port.

Table 4.5-1 shows pin assignment of port 2.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2</td>
<td>P20/IN00</td>
<td>P20</td>
<td>IN00</td>
<td>Input</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>P21/IN01</td>
<td>P21</td>
<td>IN01</td>
<td>Output</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>P22/IN10</td>
<td>P22</td>
<td>IN10</td>
<td></td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>P23/IN11</td>
<td>P23</td>
<td>IN11</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P24/AIN0</td>
<td>P24</td>
<td>AIN0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P25/BIN0</td>
<td>P25</td>
<td>BIN0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P26/ZIN0</td>
<td>P26</td>
<td>ZIN0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>INT7</td>
<td></td>
<td>INT7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P27/ADTG</td>
<td>P27</td>
<td>ADTG</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Block Diagram of Pins of Port 2

*Figure 4.5-1 Block Diagram of Pins of Port 2*

---

#### Registers for Port 2

- The registers for port 2 are PDR2 and DDR2.
- The bits composing each register correspond to the pins of port 2 one-to-one.

Table 4.5-2 shows the correspondence between port 2 registers and pins.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port 2</strong></td>
<td>PDR2, DDR2 bit7, bit6, bit5, bit4, bit3, bit2, bit1, bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P27, P26, P25, P24, P23, P22, P21, P20</td>
</tr>
</tbody>
</table>

---

Reference: Refer to "1.7 I/O Circuit" for circuit type.
CHAPTER 4 I/O PORT

4.5.1 Registers for Port 2 (PDR2, DDR2)

The registers for port 2 are explained.

■ Function of Registers for Port 2

- Port 2 data register (PDR2)

  Port 2 data register indicates the state of the pins.

- Port 2 direction register (DDR2)
  - The port 2 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

  Table 4.5-3 shows function of registers for port 2.

Table 4.5-3 Function of Registers for Port 2

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2 data register (PDR2)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000002H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 2 direction register (DDR2)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000012H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate

Reference: When using as an input pin of peripheral function, set "0" to the bit of port 2 direction register (DDR2) corresponding to the input pin of peripheral functions.
4.5.2 Operation of Port 2

The operation of port 2 is explained.

### Operation of Port 2

- **Operation of output port**
  - With the corresponding DDR2 register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDR2 register is retained in the PDR output latch and then output to the pins as is.
  - Reading the PDR2 register allows the pin values (output level) to be read.

### Notes:
- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR2 register and then set to the DDR register by the PDR.

- **Operation of input port**
  - With the corresponding DDR2 register bit set to "0", the port works as an input port.
  - If data is written to the PDR2 register, it is retained in the PDR2 output latch but not output to the pins.
  - If data is read the PDR2 register, it is read pin level ("L" or "H").

- **Operation of resource input**
  - The state of the pin that serves as the resource input is input to the resource.
  - When using as an input pin of peripheral function, set "0" to the bit of port 2 direction register (DDR2) corresponding to the input pin of peripheral functions.

- **Operation at reset**
  - At CPU reset, value of the DDR2 register is initialized to "00H" (input port).
  - In a reset operation, the PDR2 register is not initialized. Therefore, if used as an output port, set the output data in the PDR2 register and then set DDR2 register corresponding to output to "1".
● Operation in stop mode, timebase timer mode or watch mode

  - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

● Operation of hardware standby mode

  - When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR2 register’s value, the output buffer is forcibly set to "OFF."

    However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance."

    Table 4.5-4 shows the state of the port 2 pins.

**Table 4.5-4 State of Sport 2 pins**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode</th>
<th>Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0  SPL=1</td>
<td></td>
</tr>
<tr>
<td>P20/IN00</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P21/IN01</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P22/IN10</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P23/IN11</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P24/AIN0</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P25/BIN0</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>P26/ZIN0/</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
<tr>
<td>INT7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P27/ADTG</td>
<td>General-purpose I/O</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
Port 3 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 3.

### Configuration of Port 3

Port 3 consists of following factors.
- General-purpose I/O port (P30)
- General-purpose I/O port/resource output pin (P30/CKOT to P37/PPG01)
- Port 3 data register (PDR3)
- Port 3 direction register (DDR3)

### Pin Assignment of Port 3

- Use port 3 by switching between the resource pin and the general-purpose I/O port.
- Since port 3 serves as resource pin, when used as a resource, port 3 cannot be used as general-purpose I/O port.
- When using port 3 as the output pin of the resource, set the output of the corresponding resource to enabled. Port 3 functions as the output pin of the resource regardless of the settings of the DDR3.

Table 4.6-1 shows pin assignment of port 3.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3</td>
<td>P30</td>
<td>P30</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P31/CKOT</td>
<td>P31</td>
<td>CKOT</td>
<td>Clock monitor output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P32/OUT0</td>
<td>P32</td>
<td>OUT0</td>
<td>I/O timer (Output compare 0) output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P33/OUT1</td>
<td>P33</td>
<td>OUT1</td>
<td>I/O timer (Output compare 1) output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P34/OUT2</td>
<td>P34</td>
<td>OUT2</td>
<td>I/O timer (Output compare 2) output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P35/OUT3</td>
<td>P35</td>
<td>OUT3</td>
<td>I/O timer (Output compare 3) output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P36/PPG00</td>
<td>P36</td>
<td>PPG00</td>
<td>PPG timer 0 output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P37/PPG01</td>
<td>P37</td>
<td>PPG01</td>
<td>PPG timer 0 output</td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
CHAPTER 4 I/O PORT

■ Block Diagram of Pins of Port 3

Figure 4.6-1 Block Diagram of Pins of Port 3

■ Registers for Port 3

- The registers for port 3 are PDR3 and DDR3.
- The bits composing each register correspond to the pins of port 3 one-to-one.

Table 4.6-2 shows correspondence between port 3 registers and pins.

Table 4.6-2 Correspondence between Port 3 Registers and Pins

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3</td>
<td>PDR3, DDR3 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P37 P36 P35 P34 P33 P32 P31 P30</td>
</tr>
</tbody>
</table>
4.6.1 Registers for Port 3 (PDR3, DDR3)

The registers for port 3 are explained.

- **Function of Registers for Port 3**

- **Port 3 data register (PDR3)**
  - Port 3 data register indicates the state of the pins.

- **Port 3 direction register (DDR3)**
  - The port 3 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Table 4.6-3 shows function of registers for ports.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000003H</td>
<td>XXXXXXXXB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>00000000B</td>
<td></td>
</tr>
<tr>
<td>Port 3 direction register (DDR3)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000013H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
4.6.2 Operation of Port 3

The operation of port 3 is explained.

- **Operation of Port 3**
  - **Operation of output port**
    - With the corresponding DDR3 register bit set to "1", the port works as an output port.
    - When used as an output port, any data written to the PDR3 register is retained in the PDR output latch and then output to the pins as is.
    - Reading the PDR3 register allows the pin values (output level) to be read.

  - **Operation of input port**
    - With the corresponding DDR3 register bit set to "0", the port works as an input port.
    - If data is written to the PDR3 register, it is retained in the PDR3 output latch but not output to the pins.
    - If data is read the PDR3 register, it is read pin level ("L" or "H").

  - **Operation of resource output**
    - When using as output pin of peripheral functions, output is set to enable with the corresponding peripheral functions.
    - Output enable of peripheral functions is prioritized in spite of the setting of DDR3 register.
    - When setting the output of peripheral functions to enable and then reading the pin state, the output state of the peripheral functions is read.

  - **Operation at reset**
    - If the CPU is reset, port 3 direction register value is initialized to "00H" (input port).
    - Port 3 data register is not initialized by reset. When using as the output port, after setting of output data to PDR3 register, set the bit of DDR3 register corresponding to output pin to "1".

  - **Operation in stop mode, timebase timer mode or watch mode**
    - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

---

Notes:
- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.
**Note:** To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode or timebase timer mode, disable the output of peripheral functions, and set the STP bit of the low-power consumption mode control register (LPMCR) to 1 or set the TMD bit to "0".

This applies to the following pins:
P31/CKOT, P32/OUT0, P33/OUT1, P34/OUT2, P35/OUT3, P36/PPG00, P37/PPG01

- **Operation of hardware standby mode**
  
  - When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance". This is because, irrespective of the DDR3 register's value, the output buffer is forcibly set to "OFF."
  
  However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

  Table 4.6-4 shows pin state of port 3.

**Table 4.6-4 State of port 3 pins**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>P30 to P37/PPG01</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z: High impedance
4.7 Port 4

Port 4 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 4.

- **Configuration of Port 4**
  Port 4 consists of the following factors:
  - General-purpose I/O port/resource I/O pin (P40/PPG10 to P47/SCK1)
  - Port 4 data register (PDR4)
  - Port 4 direction register (DDR4)
  - Port 4 input resistance register (RDR4)

- **Pin Assignment of Port 4**
  - Use port 4 by switching between the resource pin and the general-purpose I/O port.
  - Since port 4 serves as resource pin, when used as a resource, port 4 cannot be used as general-purpose I/O port.
  - When using port 4 as the input pin of the resource, set the pin corresponding to the resource in the DDR4 as an input port.
  - When using port 4 as the output pin of the resource, set the output of the corresponding resource to enabled. Port 4 functions as the output pin of the resource regardless of the settings of the DDR4.

Table 4.7-1 shows pin assignment of port 4.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>P40/PPG10</td>
<td>P40</td>
<td>PPG10</td>
<td>PPG timer 1 output</td>
<td>CMOS</td>
<td>Hysteresis</td>
</tr>
<tr>
<td>P41/PPG11</td>
<td>P41</td>
<td>PPG11</td>
<td></td>
<td></td>
<td>CMOS</td>
</tr>
<tr>
<td>P42/SIN0</td>
<td>P42</td>
<td>SIN0</td>
<td>UART serial data input</td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>P43/SOT0</td>
<td>P43</td>
<td>SOT0</td>
<td>UART serial data output</td>
<td></td>
<td>CMOS</td>
</tr>
<tr>
<td>P44/SCK0</td>
<td>P44</td>
<td>SCK0</td>
<td>UART serial clock I/O</td>
<td></td>
<td>CMOS</td>
</tr>
<tr>
<td>P45/SIN1</td>
<td>P45</td>
<td>SIN1</td>
<td>I/O Extended serial 1 serial data input</td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>P46/SOT1</td>
<td>P46</td>
<td>SOT1</td>
<td>I/O Extended serial 1 serial data output</td>
<td></td>
<td>CMOS</td>
</tr>
<tr>
<td>P47/SCK1</td>
<td>P47</td>
<td>SCK1</td>
<td>I/O Extended serial 1 serial clock I/O</td>
<td></td>
<td>CMOS</td>
</tr>
</tbody>
</table>
■ Block Diagram of Pins of Port 4

![Block Diagram of Pins of Port 4](image)

**Figure 4.7-1 Block Diagram of Pins of Port 4**

- Resource input
- Resource output
- Resource enables output
- Pull-up resistance options
- Selected connection or disconnection
- Pch
- Nch
- Standby control (SPL=1)

Reference: Refer to "1.7 I/O Circuit" for circuit type.

■ Registers for Port 4

- The registers for port 4 are PDR4 and DDR4.
- Port 4 input resistance register (RDR4) connects or cuts off the pull-up resistance to the pin set in input port.
- The bits composing each register correspond to the pins of port 4 one-to-one.

Table 4.7-2 shows correspondence between port 4 registers and pins.

**Table 4.7-2 Correspondence between Port 4 Registers and Pins**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4</td>
<td>PDR4, DDR4, RDR4 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P47 P46 P45 P44 P43 P42 P41 P40</td>
</tr>
</tbody>
</table>

* : Input and output for the peripheral functions corresponds to the input and output for the peripheral functions.
4.7.1 Registers for Port 4 (PDR4, DDR4, RDR4)

The registers for port 4 are explained.

■ Function of Registers for Port 4

- Port 4 data register (PDR4)
  - Port 4 data register indicates the state of the pins.

- Port 4 direction register (DDR4)
  - The port 4 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- Port 0 input resistance register (RDR4)
  - Port 4 input resistance register (PDR4) sets to connect/cut off the pull-up resistance.
  - It is valid when the pin is set to the input port. When setting the bit corresponding to input port to "1" or "0", pull-up resistance is connected or cut off respectively. It is invalid when the pin is set to the output port and pull-up resistance is cut off.

Table 4.7-3 shows function of registers for port 4.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000004H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000004H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 4 direction register (DDR4)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000014H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td>000014H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 0 input resistance register (RDR4)</td>
<td>0</td>
<td>Pull-up resistance sets to shut off.</td>
<td>Sets the pull-up resistance to cut off.</td>
<td>R/W</td>
<td>00008EH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pull-up resistance sets to connect.</td>
<td>Sets the pull-up resistance to connect.</td>
<td>R/W</td>
<td>00008EH</td>
<td>00000000B</td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
| **Reference:** | When using as an input pin of peripheral function, set "0" to the bit of port 4 direction register (DDR4) corresponding to the input pin of peripheral functions. |
CHAPTER 4 I/O PORT

4.7.2 Operation of Port 4

The operation of port 4 is explained.

■ Operation of Port 4

- **Operation of output port**
  - With the corresponding DDR4 register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDR register is retained in the PDR output latch and then output to the pins as is.
  - Reading the PDR register allows the pin values (output level) to be read.
  - Setting of port 4 input resistance register (RDR1) is invalid.

**Notes:**
- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.

- **Operation of input port**
  - With the corresponding DDR4 register bit set to "0", the port works as an input port.
  - If data is written to the PDR4 register, it is retained in the PDR4 output latch but not output to the pins.
  - If data is read the PDR4 register, it is read pin level ("L" or "H").
  - Setting of port 4 input resistance register (RDR0) is valid.

- **Operation of resource output**
  - When using as output pin of peripheral functions, output is set to enable with the corresponding peripheral functions.
  - Output enable of peripheral functions is prioritized in spite of the setting of DDR4 register.
  - When setting the output of peripheral functions to enable and then reading the pin state, the output state of the peripheral functions is read.

- **Operation of resource input**
  - The state of the pin that serves as the resource input is input to the resource.
  - When using as the input of peripheral functions, set the bit of DDR4 register corresponding to input pin of peripheral function to "0" (input port).

- **Operation at reset**
  - At CPU reset, value of the DDR4 register is initialized to "00H" (input port).
  - In a reset operation, the PDR4 register is not initialized. Therefore, if used as an output port, set the output data in the PDR4 register and then set DDR4 register corresponding to output to "1".
● Operation in stop mode, timebase timer mode or watch mode
  
  - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

● Operation of hardware standby mode
  
  - When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR4 register’s value, the output buffer is forcibly set to "OFF".
  
  However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.7-4 shows the pin state of port 4.

### Table 4.7-4 State of port 4 pins

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>SPL=0</td>
<td></td>
<td>SPL=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P40/PPG10 to P47/SCK1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
### 4.8 Port 5

Port 5 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 5.

#### Configuration of Port 5

Port 5 consists of following factors.
- General-purpose I/O port/resource I/O pin (P50/SIN2 to P54/DA1)
- Port 5 data register (PDR5)
- Port 5 direction register (DDR5)

#### Pins Assignment of Port 5

- Use port 5 by switching between the resource pin and the general-purpose I/O port.
- Since port 5 serves as resource pin, when used as a resource, port 5 cannot be used as general-purpose I/O port.
- When using port 5 as the input pin of the resource, set the pin corresponding to the resource in the DDR5 as an input port.
- When using port 5 as the output pin of the resource, set the output of the corresponding resource to enabled. Port 5 functions as the output pin of the resource regardless of the settings of the DDR5.

Table 4.8-1  shows pins assignment of port 5.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P50/SIN2/AIN1</td>
<td>P50</td>
<td>SIN2</td>
<td>Input</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>I/O Extended serial 2 serial data input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P51/SOT2/BIN1</td>
<td>P51</td>
<td>SOT2</td>
<td>Output</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>I/O Extended serial 2 serial data output</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CMOS Hysteresis</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P52/SCK2/ZIN1</td>
<td>P52</td>
<td>SCK2</td>
<td>Input</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>I/O Extended serial 2 serial clock I/O</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P53/DA0</td>
<td>P53</td>
<td>DA0</td>
<td>Output</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D/A converter 0 analog output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P54/DA1</td>
<td>P54</td>
<td>DA1</td>
<td>Output</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D/A converter 1 analog output</td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
■ Block Diagram of Pins of Port 5

Figure 4.8-1 Block Diagram of Pins of Port 5 (used as resource I/O pin)

Figure 4.8-2 Block Diagram of Pins of Port 5 (used as D/A analog output pin)
CHAPTER 4 I/O PORT

- Registers for Port 5
  - The registers for port 5 are PDR5 and DDR5.
  - The bits composing each register correspond to the pins of port 5 one-to-one.

Table 4.8-2 shows correspondence between port 5 registers and pins.

Table 4.8-2 Correspondence between Port 5 Registers and Pins

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5</td>
<td>PDR5, DDR5</td>
</tr>
<tr>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>-</td>
</tr>
</tbody>
</table>
4.8.1 Registers for Port 5 (PDR5, DDR5)

The registers for port 5 are explained.

### Function of Registers for Port 5

- **Port 5 data register (PDR5)**
  - Port 5 data register indicates the state of the pins.

- **Port 5 direction register (DDR5)**
  - The port 5 direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Table 4.8-3 shows function of register for port 5.

#### Table 4.8-3 Function of Registers for Port 5

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5 data register (PDR5)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000005H</td>
<td>XXXXXXXX&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000015H</td>
<td>XXXX0000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td>Port 5 direction register (DDR5)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000015H</td>
<td>XXXX0000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td>000015H</td>
<td>XXXX0000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

R/W: Read/Write  
X: Indeterminate

**Reference:** When using as an input pin of peripheral function, set "0" to the bit of port 5 direction register (DDR5) corresponding to the input pin of peripheral functions.
4.8.2 Operation of Port 5

The operation of port 5 is explained.

- **Operation of Port 5**

  - **Operation of output port**
    - With the corresponding DDR5 register bit set to "1", the port works as an output port.
    - When used as an output port, any data written to the PDR5 register is retained in the PDR output latch and then output to the pins as is.
    - Reading the PDR5 register allows the pin values (output level) to be read.

  - **Operation of input port**
    - With the corresponding DDR5 register bit set to "0", the port works as an input port.
    - If data is written to the PDR5 register, it is retained in the PDR5 output latch but not output to the pins.
    - If data is read from the PDR5 register, it is read pin level ("L" or "H").

  - **Operation of resource output**
    - When using as output pin of peripheral functions, output is set to enable with the corresponding peripheral functions.
    - Output enable of peripheral functions is prioritized in spite of the setting of DDR5 register.
    - When setting the output of peripheral functions to enable and then reading the pin state, the output state of the peripheral functions is read.

  - **Operation of resource input**
    - The state of the pin that serves as the resource input is input to the resource.
    - When using as an input pin of peripheral function, set "0" to the bit of port 5 direction register (DDR5) corresponding to the input pin of peripheral functions.

- **Notes:**
  - If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
  - To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.
CHAPTER 4 I/O PORT

● Operation at reset
  • At CPU reset, value of the DDR5 register is initialized to "00H" (input port).
  • In a reset operation, the PDR5 register is not initialized. Therefore, if used as an output port, set the output data in the PDR5 register and then set DDR5 register corresponding to output to "1".

● Operation in stop mode, timebase timer mode or watch mode
  • When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

● Operation of hardware standby mode
  • When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR5 register’s value, the output buffer is forcibly set to "OFF."
  However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance."

Table 4.8-4 shows the state of the port 5 pins.

Table 4.8-4 State of port 5 pins

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>P50/SIN2 AIN1 to P54/DA1</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
4.9 Port 6

Port 6 is a general-purpose input/output port and also used for input/output of the peripheral functions. For each pin, use for a peripheral function or as port is switched in units of individual bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 6.

■ Configuration of Port 6

Port 6 consists of following factors.

- General-purpose I/O port/analog input pins. (P60/AN0 to P67/AN7)
- Port 6 data register (PDR6)
- Port 6 direction register (DDR6)
- Analog input enable register (ADER)

■ Pins Assignment of Port 6

- Use port 6 by switching between the analog input pin and the general-purpose I/O port.
- Since port 6 serves as an analog input pin, it cannot be used as a general-purpose I/O port when used as an analog input pin.
- When using port 6 as an analog input pin, set the pin corresponding to the analog input in the DDR6 as an input port.
- When using port 6 as a general-purpose I/O port, do not input any analog signal.

Table 4.9-1 shows pins assignment of port 6.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>P60/AN0</td>
<td>P60</td>
<td>General-purpose I/O</td>
<td>AN0</td>
<td>Analog input channel 0</td>
<td>CMOS</td>
</tr>
<tr>
<td>P61/AN1</td>
<td>P61</td>
<td>General-purpose I/O</td>
<td>AN1</td>
<td>Analog input channel 1</td>
<td>CMOS</td>
</tr>
<tr>
<td>P62/AN2</td>
<td>P62</td>
<td>General-purpose I/O</td>
<td>AN2</td>
<td>Analog input channel 2</td>
<td>CMOS</td>
</tr>
<tr>
<td>P63/AN3</td>
<td>P63</td>
<td>General-purpose I/O</td>
<td>AN3</td>
<td>Analog input channel 3</td>
<td>CMOS</td>
</tr>
<tr>
<td>P66/AN4</td>
<td>P66</td>
<td>General-purpose I/O</td>
<td>AN4</td>
<td>Analog input channel 4</td>
<td>CMOS</td>
</tr>
<tr>
<td>P66/AN5</td>
<td>P66</td>
<td>General-purpose I/O</td>
<td>AN5</td>
<td>Analog input channel 5</td>
<td>CMOS</td>
</tr>
<tr>
<td>P66/AN6</td>
<td>P66</td>
<td>General-purpose I/O</td>
<td>AN6</td>
<td>Analog input channel 6</td>
<td>CMOS</td>
</tr>
<tr>
<td>P67/AN7</td>
<td>P67</td>
<td>General-purpose I/O</td>
<td>AN7</td>
<td>Analog input channel 7</td>
<td>CMOS</td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
■ Block Diagram of Pins of Port 6

Figure 4.9-1  Block Diagram of Pins of Port 6

■ Registers for port 6

- The registers for port 6 are PDR6, DDR6, and ADER.
- Analog input enable register (ADER) sets to enable/disable the analog signal input to analog input pin.
- The bits correspondence each register correspond to the pins of port 6 one-to-one.

Table 4.9-2  Correspondence between port 6 registers and pins

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 6</td>
<td>PDR6, DDR6: bit7, bit6, bit5, bit4, bit3, bit2, bit1, bit0</td>
</tr>
<tr>
<td>ADER</td>
<td>ADE7, ADE6, ADE5, ADE4, ADE3, ADE2, ADE1, ADE0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P67, P66, P65, P64, P63, P62, P61, P60</td>
</tr>
</tbody>
</table>
4.9.1 Registers for Port 6 (PDR6, DDR6, ADER)

The registers for port 6 are explained.

Function of Registers for Port 6

- **Port 6 data register (PDR6)**
  - Port 6 data register indicates the state of the pins.

- **Port 6 direction register (DDR6)**
  - Port 6 direction register sets the I/O direction of pins.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- **Analog input enable register (ADER)**
  - The analog input enable register (ADER) sets the general-purpose I/O ports and analog input pin in unit of ports.
  - The pin works as analog input if the ADE bit corresponding to the analog input pin is set to "1" or as a general-purpose I/O port if the bit is set to "0".

Table 4.9-3 shows function of registers for port 6.

Table 4.9-3 Function of Registers for port 6

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000006H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000006H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 6 direction register (DDR6)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000016H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 6 direction register (DDR6)</td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td>000016H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Analog input enable register (ADER)</td>
<td>0</td>
<td>Set general-purpose I/O port</td>
<td></td>
<td>R/W</td>
<td>00001B</td>
<td>11111111B</td>
</tr>
<tr>
<td>Analog input enable register (ADER)</td>
<td>1</td>
<td>Set analog input pin</td>
<td></td>
<td>R/W</td>
<td>00001B</td>
<td>11111111B</td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate

Note: When used as input port, if a medium level signal is input, an input leak current flows. To avoid this, pins used for analog input must have their corresponding ADER bits set to "1".
Reference: When using as an analog input pin, set "0" (input port) to the bit of DDR6 register corresponding to the analog input pin.
4.9.2 Operation of Port 6

The operation of port 6 is explained.

Operation of Port 6

- **Operation of output port**
  - With the corresponding DDR6 register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDR6 register is retained in the PDR output latch and then output to the pins as is.
  - Reading the PDR6 register allows the pin values (output level) to be read.

- **Operation of input port**
  - With the corresponding DDR6 register bit set to "0", the port works as an input port.
  - If data is written to the PDR6 register, it is retained in the PDR6 output latch but not output to the pins.
  - If data is read the PDR6 register, it is read pin level ("L" or "H").

- **Operation of analog input**
  - When using port 6 as an analog input pin, set the bit in the ADER corresponding to the analog input pin to "1". Port 6 is disabled to operate as a general-purpose I/O port, and functions as an analog input pin.
  - When setting the analog input to enable and then reading the PDR6 register, reading value is "00H".

- **Operation at reset**
  - At CPU reset, value of the DDR6 register is initialized to "00H" (input port). Consequently, all output buffers are set to OFF (input port pin), and the pin enters the high-impedance state.
  - In a reset operation, the PDR6 register is not initialized. Therefore, if used as an output port, set the output data in the PDR6 register and then set DDR6 register corresponding to output to "1".

- **Operation in stop mode, timebase timer mode or watch mode**
  - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance." This is because, irrespective of the DDR6 register’s value, the output buffer is forcibly set to "OFF".

Notes:

- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.
Operation of hardware standby mode

- When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR6 register's value, the output buffer is forcibly set to "OFF."

However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.9-4 shows the state of the port 6 pins.

**Table 4.9-4 State of port 6 pins**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>P60/AN0 to P67/AN7</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input cutoff/output Hi-Z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z: High impedance
4.10 Port 7

Port 7 is a general-purpose input/output port and also used for LCD controller/driver common output pin and input/output of the peripheral functions. The lower 4-bit pins use for switching the input/output pin of the peripheral functions and general purpose I/O port, while the upper 4-bit pins use for switching the common output of LCD controller and general purpose I/O port. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 7.

■ Configuration of Port 7

Port 7 consists of following factors.

• General-purpose I/O port/resource I/O pin (P70/TIN0/OUT4 to P73/TOT1/OUT7)
• General-purpose I/O port/ LCD common output pin (P74/COM0 to P77/COM3)
• Port 7 data register (PDR7)
• Port 7 direction register (DDR7)
• Common pin switch register (LCDCMR)

■ Pins Assignment of Port 7

• Port 7 pins are used for peripheral function or switching common output pin and general purpose I/O port.
• Port 7 pins are used both for peripheral function and common output pin, therefore if using as peripheral function or LCD controller/driver, they are not used for general purpose I/O port.
• When using port 7 as the input pin of the resource, set the pin corresponding to the resource in the DDR7 as an input port.
• When using port 7 as the output of the resource, set the output of the corresponding resource to enabled. Port 7 functions as the output pin of the resource regardless of the settings of the DDR7.
Table 4.10-1 shows pins assignment of port 7.

**Table 4.10-1  Pins Assignment of Port 7**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>Port 7</td>
<td>P70/TIN0/OUT4</td>
<td>P70</td>
<td>TIN0</td>
<td>CMOS</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUT4</td>
<td>CMOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P71/TOT0/OUT5</td>
<td>P71</td>
<td>TOT0</td>
<td>CMOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUT5</td>
<td>CMOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P72/TIN1/OUT6</td>
<td>P72</td>
<td>TIN1</td>
<td>Hysteresis</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUT6</td>
<td>Hysteresis</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P73/TOT1/OUT7</td>
<td>P73</td>
<td>TOT1</td>
<td>Hysteresis</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUT7</td>
<td>Hysteresis</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P74/COM0</td>
<td>P74</td>
<td>COM0</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P75/COM1</td>
<td>P75</td>
<td>COM1</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P76/COM2</td>
<td>P76</td>
<td>COM2</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P77/COM3</td>
<td>P77</td>
<td>COM3</td>
<td>L</td>
<td></td>
</tr>
</tbody>
</table>

**Reference:** Refer to "1.7 I/O Circuit" for circuit type.
Chapter 4 I/O PORT

Block Diagram of Pins of Port 7

Figure 4.10-1 Block Diagram of Pins of Port 7 (used as resource I/O pin)

*Resource input*

- PDR (Port data register)
  - PDR read
  - PDR write

- DDR (Port direction register)
  - DDR read
  - DDR write

- Output latch

- Direction latch

- Standby control (SPL=1)

- Resource output*
  - Resource enables output*

- Pch
  - Nch

Standby control: Control of Stop mode (SPL=1), Timebase timer mode (SPL=1), Watch Mode (SPL=1) and Hardware standby mode

*: Input and output for the peripheral functions corresponds to the Input and output for the peripheral functions.

Figure 4.10-2 Block Diagram of Pins of Port 7 (used as common output pin)

- PDR (Port data register)
  - PDR read
  - PDR write

- DDR (Port direction register)
  - DDR read
  - DDR write

- Output latch

- Direction latch

- LCD common output

- Pch
  - Nch

- Standby control (SPL=1)

- Common pin output enabled

Standby control: Stop mode (SPL=1), timebase timer mode (SPL=1), Watch mode (SPL=1), Control of Hardware standby mode
### Registers for Port 7

- The port 7 registers are PDR7, DDR7 and LCDCMR.
- LCDCMR is used for setting to enable/disable output of common signal to the common output pin.
- The bits composing each register correspond to the pins of port 7 one-to-one.

Table 4.10-2 shows correspondence between port 7 registers and pins.

**Table 4.10-2  Correspondence between port 7 registers and pins**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 7</td>
<td>PDR7, DDR7</td>
</tr>
<tr>
<td></td>
<td>LCDCMR</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P77</td>
</tr>
</tbody>
</table>
4.10.1 Registers for Port 7 (PDR7, DDR7, LCDCMR)

The registers for port 7 are explained.

**Function of Registers for Port 7**

- **Port 7 data register (PDR7)**
  - Port 7 data register indicates the state of the pins.

- **Port 7 direction register (DDR7)**
  - DDR7 register is used to set the pin I/O direction.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- **Common pin switch register (LCDCMR)**
  - LCDCMR is used to set the general-purpose I/O port and LCD common output pin for port unit.
  - When the COM bit corresponding to the LCD common output pin is set to "1", it functions as LCD common output pin. When the bit is set to "0", it functions as a general-purpose I/O port.

Table 4.10-3 shows function of registers for port 7.

**Table 4.10-3 Function of Registers for Port 7**

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 7 data register (PDR7)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W 000007H</td>
<td>XXXXXXXX0B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 7 direction register (DDR7)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W 000017H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Common pin switch register (LCDCMR)</td>
<td>0</td>
<td>COM bit is &quot;0&quot;.</td>
<td>Sets the COM bit to &quot;0&quot; to use it as a general purpose I/O port.</td>
<td>R/W 00000BH</td>
<td>XXXX0000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>COM bit is &quot;1&quot;</td>
<td>Sets the COM bit to &quot;1&quot; to use it as a common pin.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
### 4.10.2 Operation of Port 7

The operation of port 7 is explained.

#### Operation of Port 7

- **Operation of output port**
  
  - With the corresponding DDR7 register bit set to "1", the port works as an output port.
  
  - When used as an output port, any data written to the PDR7 register is retained in the PDR output latch and then output to the pins as is.
  
  - Reading the PDR7 register allows the pin values (output level) to be read.

#### Operation of input port

- With the corresponding DDR7 register bit set to "0", the port works as an input port.

- If data is written to the PDR7 register, it is retained in the PDR7 output latch but not output to the pins.

- If data is read the PDR7 register, it is read pin level ("L" or "H").

#### Operation of resource output

- To use the port for peripheral function output, specify it with the peripheral function’s output enable bit.

- In spite of DDR7 setting, peripheral function’s output enable is prioritized.

- Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function’s output value to be detected.

#### Operation of common output

- When using as a common output pin, set the COM bit of common pin switching register corresponding to common output pin to "1". It disables the operation as general purpose I/O port and functions as the common output pin.

- Set the common output pin following the setting of display duty mode.

#### Notes

- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.

- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.
CHAPTER 4  I/O PORT

- **Operation at reset**
  - At CPU reset, value of the DDR7 register is initialized to "00H" (input port).
  - In a reset operation, the PDR7 register is not initialized. Therefore, if used as an output port, set the output data in the PDR7 register and then set DDR7 register corresponding to output to "1".

- **Operation in stop mode, timebase timer mode or watch mode**
  - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register's pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."

- **Operation of hardware standby mode**
  - When setting `HST` pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR7 register’s value, the output buffer is forcibly set to "OFF."
    However, even if setting `HST` pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.10-4 shows the state of the port 7 pins.

**Table 4.10-4 State of Port 7 pins**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>P70/TIN0/OUT4 to P73/TOT1/OUT7</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Input cutoff/output Hi-Z</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td>P74/COM0 to P77/COM3</td>
<td>Common output</td>
<td>Common output</td>
<td>Common output</td>
<td>Common output</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input cutoff/output Hi-Z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
4.11 Port 8

Port 8 is a general-purpose input/output port that is also used as segment output pin of LCD controller/driver. The use of each pin for a segment output pin or a general purpose I/O port is switched in units of 8 bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 8.

■ Configuration of Port 8

Port 8 consists of the following factors:

- General purpose I/O port / Segment output pin (P80/SEG16 to P87/SEG23)
- Port 8 data register (PDR8)
- Port 8 direction register (DDR8)
- LCDC control register 1 (LCR1)

■ Pins Assignment of Port 8

- Port 8 pins are used for switching segment output pin and general purpose I/O port.
- Port 8 pins are also used for segment output pin, therefore if using as segment output, they are not used for general purpose I/O port.
- When using port 8 as the segment output pin, set the output of the corresponding segment to enable. Port 8 functions as the segment output pin regardless of the settings of the DDR8.

Table 4.11-1 Pins Assignment of Port 8

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 8</td>
<td>P80/SEG16</td>
<td>P80</td>
<td>SEG16</td>
<td>CMOS Hysteresis</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>P81/SEG17</td>
<td>P81</td>
<td>SEG17</td>
<td>CMOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P82/SEG18</td>
<td>P82</td>
<td>SEG18</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P83/SEG19</td>
<td>P83</td>
<td>SEG19</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P84/SEG20</td>
<td>P84</td>
<td>SEG20</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P85/SEG21</td>
<td>P85</td>
<td>SEG21</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P86/SEG22</td>
<td>P86</td>
<td>SEG22</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P87/SEG23</td>
<td>P87</td>
<td>SEG23</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
### Block Diagram of Pins of Port 8

![Figure 4.11-1 Block Diagram of Pins of Port 8](image)

#### Registers for Port 8
- The register controlling port 8 is PDR8, DDR8, and LCR1.
- LCR1 is used for setting to enable/disable output of segment signal in segment output pin.
- The bits composing each register correspond to the pins of port 8 one-to-one.

Table 4.11-2 shows correspondence between port 8 registers and pins.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 8</td>
<td>PDR8, DDR8, LCR1 bit7, bit6, bit5, bit4, bit3, bit2, bit1, bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P87, P86, P85, P84, P83, P82, P81, P80</td>
</tr>
</tbody>
</table>
4.11.1 Registers for Port 8 (PDR8, DDR8, LCR1)

The registers for port 8 are explained.

- **Function of Registers for Port 8**

  - Port 8 data register (PDR8)
    - Port 8 data register indicates the state of the pins.

  - Port 8 direction register (DDR8)
    - DDR8 is used to set the I/O direction of pins.
    - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

  - LCDC control register 1 (LCR1: SEG1,SEG0)
    - LCD control register can be set the general purpose I/O port and segment output pin in units of 8 bits.
    - When the SEG bit corresponding to the segment output pin is set to "1", it functions as segment output pin. When the bit is set to "0", it functions as a general-purpose I/O port.

  Table 4.11-3 shows the function of register for port 8.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 8 data register (PDR8)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000008H</td>
<td>XXXXXXXXB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>00000000B</td>
<td></td>
</tr>
<tr>
<td>Port 8 direction register (DDR8)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000018H</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCDC control register 1 (LCR1)</td>
<td>0</td>
<td>SEG bit is &quot;0&quot;.</td>
<td>Sets SEG bit to &quot;0&quot; and functions as general purpose I/O port.</td>
<td>R/W</td>
<td>00006BH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>SEG bit is &quot;1&quot;.</td>
<td>Sets SEG bit to &quot;1&quot; and functions as segment output pin.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
CHAPTER 4  I/O PORT

4.11.2 Operation of Port 8

The operation of port 8 is explained.

■ Operation of Port 8

- Operation of output port
  - With the corresponding DDR8 register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDR8 register is retained in the PDR output latch and then output to the pins as is.
  - Reading the PDR8 register allows the pin values (output level) to be read.

Notes:
  - If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
  - To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.

- Operation of input port
  - With the corresponding DDR8 register bit set to "0", the port works as an input port.
  - If data is written to the PDR8 register, it is retained in the PDR8 output latch but not output to the pins.
  - If data is read the PDR8 register, it is read pin level ("L" or "H").

- Operation of segment output
  - When using as segment output pin, set "1" to SEG bit of LCR1.

- Operation at reset
  - At CPU reset, value of the DDR8 register is initialized to "00H" (input port).
  - In a reset operation, the PDR8 register is not initialized. Therefore, if used as an output port, set the output data in the PDR8 register and then set DDR8 register corresponding to output to "1".
● Operation in stop mode, timebase timer mode or watch mode

- When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."
- When functioning as a segment output pin, the segment output state is maintained.

● Operation of hardware standby mode

- When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR8 register’s value, the output buffer is forcibly set to "OFF". However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.11-4 shows the state of the port 8 pins.

**Table 4.11-4 State of Port 8 pins**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>P80 to P87</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td>SEG16 to SEG23</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Segment output</td>
</tr>
</tbody>
</table>
4.12 Port 9

Port 9 is an Nch open drain general-purpose input/output port that is also used as segment output pin of LCD controller/driver. The use of each pin for a segment output pin or a general purpose I/O port is switched in units of 8 bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port 9.

### Configuration of Port 9

Port 9 consists of the following factors:

- General purpose I/O port / Segment output pin (P90/SEG24 to P97/SEG31)
- Port 9 data register (PDR9)
- Port 9 direction register (DDR9)
- LCDC control register 1 (CR1)

### Pins Assignment of Port 9

- Port 9 pins are used for switching segment output pin and general purpose I/O port.
- Port 9 pins are also used for segment output pin, therefore if using as segment output, they are not used for general purpose I/O port.
- When using port 9 as the segment output pin, set the output of the corresponding segment to enable. Port 9 functions as the segment output pin regardless of the settings of the DDR9.

Table 4.12-1 shows pins assignment of port 9.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9</td>
<td>P90/SEG24</td>
<td>P90</td>
<td>SEG24</td>
<td>Input</td>
<td>CMOS Hysteresis</td>
</tr>
<tr>
<td></td>
<td>P91/SEG25</td>
<td>P91</td>
<td>SEG25</td>
<td>Input</td>
<td>N-ch Open drain</td>
</tr>
<tr>
<td></td>
<td>P92/SEG26</td>
<td>P92</td>
<td>SEG26</td>
<td>Input</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>P93/SEG27</td>
<td>P93</td>
<td>SEG27</td>
<td>Output</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>P94/SEG28</td>
<td>P94</td>
<td>SEG28</td>
<td>Output</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>P95/SEG29</td>
<td>P95</td>
<td>SEG29</td>
<td>Output</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>P96/SEG30</td>
<td>P96</td>
<td>SEG30</td>
<td>Output</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>P97/SEG31</td>
<td>P97</td>
<td>SEG31</td>
<td>Output</td>
<td>M</td>
</tr>
</tbody>
</table>

**Reference:** Refer to "1.7 I/O Circuit" for circuit type.
### Block Diagram of Pins of Port 9

Figure 4.12-1 Block Diagram of Pins of Port 9

- **Registers for Port 9**
  - The register controlling port 9 is PDR9, DDR9 and LCR1.
  - LCR1 is used for setting to enable/disable output of segment signal in segment output pin.
  - The bits composing each register correspond to the pins of port 9 one-to-one.
  
  Table 4.12-2 shows correspondence between port 9 registers and pins.

**Table 4.12-2 Correspondence between port 9 registers and pins**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9</td>
<td>PDR9, DDR9 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td></td>
<td>LCR1 SEG3, SEG2</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin P97 P96 P95 P94 P93 P92 P91 P90</td>
</tr>
</tbody>
</table>
4.12.1 Registers for Port 9 (PDR9, DDR9)

The registers for port 9 are explained.

## Function of Registers for Port 9

- **Port 9 data register (PDR9)**
  - Port 9 data register indicates the state of the pins.

- **Port 9 direction register (DDR9)**
  - DDR9 register is I/O direction of pins.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- **LCDC control register 1 (LCR1: SEG2,SEG3)**
  - LCD control register 1 can be set the general purpose I/O port and segment output pin in units of 8 bits.
  - When the SEG bit corresponding to the segment pin is set to "1" and "0", it functions as segment output and general purpose I/O port respectively.

Table 4.12-3 shows function of registers for port 9.

### Table 4.12-3 Function of Registers for Port 9

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9 data register (PDR9)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000009H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td>R/W</td>
<td>000009H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>Port 9 direction register (DDR9)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>000019H</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td>R/W</td>
<td>000019H</td>
<td>00000000_B</td>
</tr>
<tr>
<td>LCDC control register 1 (LCR1)</td>
<td>0</td>
<td>SEG bit is &quot;0&quot;.</td>
<td>Sets SEG bit to &quot;0&quot; to use it as a general purpose I/O port.</td>
<td>R/W</td>
<td>00006B_H</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>SEG bit is &quot;1&quot;.</td>
<td>Sets SEG bit to &quot;1&quot; to use it as a segment output pin.</td>
<td>R/W</td>
<td>00006B_H</td>
<td>00000000_B</td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
4.12.2 Operation of Port 9

The operation of port 9 is explained.

- **Operation of Port 9**
  
  - **Operation of output port**
    
    - With the corresponding DDR9 register bit set to "1", the port works as an output port.
    - When used as an output port, any data written to the PDR9 register is retained in the PDR output latch and then output to the pins as is.
    - Reading the PDR9 register allows the pin values (output level) to be read.

  - **Operation of input port**
    
    - When setting "0" to DDR9 corresponding to input pin, output transistor is "OFF" and pins are "high-impedance".
    - If data is read the PDR8 register, it is read pin level ("L" or "H").

  - **Operation of segment output**
    
    - When using as segment output pin, set "1" to SEG bit of LCR1. It disables the operation as general purpose I/O port is disabled and functions as segment output pin.

  - **Operation at reset**
    
    - At CPU reset, value of the DDR9 register is initialized to "00H" (input port).
    - In a reset operation, the PDR9 register is not initialized. Therefore, if used as an output port, set the output data in the PDR9 register and then set DDR9 register corresponding to output to "1".

  - **Operation in stop mode, timebase timer mode or watch mode**
    
    - When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."
    - When functioning as a segment output pin, the segment output state is maintained.

Notes:

- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.
Operation of hardware standby mode

- When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDR9 register's value, the output buffer is forcibly set to "OFF."

However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance".

Table 4.12-4 shows the state of the port 9 pins.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>P90 to P97</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z (Pull-up resistor disconnected)</td>
</tr>
<tr>
<td>SEG24 to SEG31</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Segment output</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
4.13 Port A

Port A is a general-purpose input/output port that is also used as segment output pin of LCD controller/driver. The use of each pin for a segment output pin or a general purpose I/O port is switched in units of 8 bits. This section mainly describes the configuration, pins, and block diagrams of pins and registers for Port A.

### Configuration of Port A

Port A consists of following factors.

- General purpose I/O port / Segment output pin (PA0/SEG8 to PA7/SEG15)
- Port A data register (PDRA)
- Port A direction register (DDRA)
- LCD control register 1 (LCR)

### Pins Assignment of Port A

- Port A pins are used for switching segment output pin and general purpose I/O port.
- Port A pins are also used for segment output pin, therefore if using as segment output, they are not used for general purpose I/O port.
- When using port A as the segment output pin, set the output of the corresponding segment to enable. Port A functions as the segment output pin regardless of the settings of the DDRA.

Table 4.13-1 shows pins assignment of port A.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Pin Name</th>
<th>Port Function</th>
<th>Resource</th>
<th>I/O Type</th>
<th>Circuit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PA0/SEG8</td>
<td>PA0</td>
<td>SEG8</td>
<td>Input</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>PA1/SEG9</td>
<td>PA1</td>
<td>SEG9</td>
<td>Output</td>
<td>Hysteresis</td>
</tr>
<tr>
<td></td>
<td>PA2/SEG10</td>
<td>PA2</td>
<td>SEG10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA3/SEG11</td>
<td>PA3</td>
<td>SEG11</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA4/SEG12</td>
<td>PA4</td>
<td>SEG12</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA5/SEG13</td>
<td>PA5</td>
<td>SEG13</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA6/SEG14</td>
<td>PA6</td>
<td>SEG14</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA7/SEG15</td>
<td>PA7</td>
<td>SEG15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference: Refer to "1.7 I/O Circuit" for circuit type.
Block Diagram of Pins of Port A

Figure 4.13-1  Block Diagram of Pins of Port A

Registers for Port A

- The register controlling port A is PDRA, DDRA and LCR1.
- LCR1 is used for setting to enable/disable output of segment signal in segment output pin.
- The bits composing each register correspond to the pins of port A one-to-one.

Table 4.13-2  Correspondence between port A registers and pins

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Bits of Related Registers and Corresponding Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port A</td>
<td>PDRA, DDRA bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>LCR1</td>
<td>SEG5, SEG4</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>PA7   PA6   PA5   PA4   PA3   PA2   PA1   PA0</td>
</tr>
</tbody>
</table>
4.13.1 Registers for Port A (PDRA, DDRA, LCR1)

The registers for port A are explained.

■ Function of Registers for Port A

- Port A data register (PDRA)
  - Port A data register indicates the state of the pins.

- Port A direction register (DDRA)
  - The port A direction register sets the input/output directions.
  - The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- LCDC control register (LCR1: SEG4,SEG5)
  - LCD control register 1 can be set the general purpose I/O port and segment output pin in units of 8 bits.
  - When the SEG bit corresponding to the segment output pin is set to "1", it functions as segment output pin. When the bit is set to "0", it functions as a general-purpose I/O port.

Table 4.13-3 shows function of registers for port A.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Register Address</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port A data register (PDRA)</td>
<td>0</td>
<td>The pin state is &quot;L&quot; level.</td>
<td>Outputs &quot;L&quot; level if used as an output port.</td>
<td>R/W</td>
<td>00000AH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin state is &quot;H&quot; level.</td>
<td>Outputs &quot;H&quot; level if used as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port A direction register (DDRA)</td>
<td>0</td>
<td>The direction latch is &quot;0&quot;.</td>
<td>Sets the output buffer to OFF to use it as an input port.</td>
<td>R/W</td>
<td>00001AH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The direction latch is &quot;1&quot;.</td>
<td>Sets the output buffer to ON to use it as an output port.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCDC control register 1 (LCR1)</td>
<td>0</td>
<td>SEG bit is &quot;0&quot;.</td>
<td>Sets SEG bit to &quot;0&quot; to use it as a general purpose I/O port.</td>
<td>R/W</td>
<td>00006BH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>SEG bit is &quot;1&quot;.</td>
<td>Sets SEG bit to &quot;1&quot; to use it as a segment output pin</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write
X: Indeterminate
4.13.2 Operation of Port A

The operation of port A is explained.

## Operation of Port A

- **Operation of output port**
  - With the corresponding DDRA register bit set to "1", the port works as an output port.
  - When used as an output port, any data written to the PDRA register is retained in the PDR output latch and then output to the pins as is.
  - Reading the PDRA register allows the pin values (output level) to be read.

### Notes:
- If the port data register uses a read-modify-write type instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected.
- To switch the input bit to the output bit, write the output data to the PDR register and then set to the DDR register by the PDR.

- **Operation of input port**
  - With the corresponding DDRA register bit set to "0", the port works as an input port.
  - If data is written to the PDRA register, it is retained in the PDRA output latch but not output to the pins.
  - If data is read the PDRA register, it is read pin level ("L" or "H").

- **Operation of segment output**
  - When using as segment output pin, set "1" to SEG bit of LCR1. It disables the operation as general purpose I/O port is disabled and functions as segment output pin.

- **Operation at reset**
  - At CPU reset, value of the DDRA register is initialized to "00H" (input port).
  - In a reset operation, the PDRA register is not initialized. Therefore, if used as an output port, set the output data in the PDRA register and then set DDRA register corresponding to output to "1".
● Operation in stop mode, timebase timer mode or watch mode

- When a transition to the stop mode, timebase timer mode or watch mode occurs, and the low-power consumption mode control register’s pin state specification bit (SPL in LPMCR) is "1", the pins are set to "high-impedance."
- When functioning as a segment output pin, the segment output state is maintained.

● Operation of hardware standby mode

- When setting HST pin to "L" level and a transition to the hardware standby mode, the pins are set to "high-impedance." This is because, irrespective of the DDRA register’s value, the output buffer is forcibly set to "OFF."
  However, even if setting HST pin to "L" level in the state oscillation stops, the pin is not set to "high-impedance."
  Table 4.13-4 shows the state of port A pins.

Table 4.13-4  State of port A pins

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Normal Operation</th>
<th>Sleep mode</th>
<th>Stop Mode, Timebase Timer Mode or Watch Mode</th>
<th>Hardware Standby Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SPL=0</td>
<td>SPL=1</td>
</tr>
<tr>
<td>PA0 to PA7</td>
<td>General-purpose</td>
<td>General-purpose</td>
<td>General-purpose I/O ports</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
<tr>
<td></td>
<td>I/O ports</td>
<td>I/O ports</td>
<td></td>
<td>Hi-Z</td>
</tr>
<tr>
<td>SEG8 to SEG15</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Segment output</td>
<td>Input cutoff/output Hi-Z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)
Hi-Z: High impedance
This chapter describes the functions and operations of the timebase timer.

5.1 Overview of Timebase Timer
5.2 Block Diagram of Timebase Timer
5.3 Configuration of Timebase Timer
5.4 Interrupt of Timebase Timer
5.5 Explanation of Operations of Timebase Timer
5.6 Precautions when Using Timebase Timer
5.7 Program Example of Timebase Timer
5.1 Overview of Timebase Timer

The timebase timer is an 18-bit free-running counter (timebase timer counter) that counts up in synchronization with the main clock (2-divided frequency of the oscillation clock HCLK).

- Interval times can be selected from 4 types \(2^{12}/HCLK, 2^{14}/HCLK, 2^{16}/HCLK, \text{and } 2^{19}/HCLK\), and interrupt requests can be generated for each interval time.
- Operating clock is supplied to peripheral functions, such as the watchdog timer and timer for the oscillation stabilization wait time for the main clock.

### Interval Timer Function

- When the timebase timer counter reaches the set interval time, overflow (carry) is generated on the bit that supports the interval time of the timebase timer counter, and an overflow flag will be set.
- An interrupt request is generated when interruption due to generation of overflow is enabled by the TBIE bit of the timebase timer control register (TBTC).
- Interval times of timebase timer can be selected from four types. Table 5.1-1 shows the interval times of the timebase timer.

#### Table 5.1-1 Interval Times of Timebase Timer

<table>
<thead>
<tr>
<th>Internal count clock cycle</th>
<th>Interval Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/HCLK (0.5µs)</td>
<td>(2^{12}/HCLK) (Approx. 1.024ms)</td>
</tr>
<tr>
<td></td>
<td>(2^{14}/HCLK) (Approx. 4.096ms)</td>
</tr>
<tr>
<td></td>
<td>(2^{16}/HCLK) (Approx. 16.384ms)</td>
</tr>
<tr>
<td></td>
<td>(2^{19}/HCLK) (Approx. 131.072ms)</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency

Descriptions within the ( ) indicate calculation examples when operated at an oscillation clock frequency of 4 MHz.
■ Clock Supply

- The timebase timer supplies an operating clock to the main clock oscillation stabilization wait time timer, watchdog timer, and peripheral functions. Cycle of the clock supplied to each peripheral function from the timebase timer is shown in Table 5.1-2.

<table>
<thead>
<tr>
<th>Where to Supply Clock</th>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Clock Oscillation Stabilization Wait Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2^{10}/HCLK$ (Approx. 0.256ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{13}/HCLK$ (Approx. 2.048ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{15}/HCLK$ (Approx. 8.192ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{17}/HCLK$ (Approx. 32.768ms)</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2^{12}/HCLK$ (Approx. 1.024ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{14}/HCLK$ (Approx. 4.096ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{16}/HCLK$ (Approx. 16.384ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{19}/HCLK$ (Approx. 131.072ms)</td>
</tr>
<tr>
<td>PPG timer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$2^{9}/HCLK$ (0.128ms)</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency
Descriptions within the () indicate calculation examples when operated at an oscillation clock frequency of 4 MHz.
5.2 Block Diagram of Timebase Timer

The timebase timer consists of the following blocks:
- Timebase timer counter
- Counter clear circuit
- Interval timer selector
- Timebase timer control register (TBTC)

Interrupt request number of the timebase timer is as follows.

Interrupt request number: #12 (0CH)
● Timebase timer counter

This is the 18-bit up counter that uses the main clock (2-divided frequency of the oscillation clock HCLK) as the count clock.

● Counter clear circuit

Circuit clears timebase timer counter

● Interval timer selector

The interval times that have been set to the timebase timer control register (TBTC) interval time selection bits (TBC1 and TBC0) are referred to, and when the counter value of the timebase timer reaches the interval time value, the TBTC overflow flag will be set.

● Timebase timer control register (TBTC)

Interval time selection, clearing of the counter for the timebase timer, enabling or disabling of interrupt requests, overflow (carry) status checks, and clearing of the overflow flag are performed.
5.3 Configuration of Timebase Timer

Interrupt causes of the timebase timer and the register details are described.

■ Generation of Interrupt Request from Timebase Timer

"1" will be set for the TBTC overflow flag bit (TBTC:TBOF) when the counter value of the timebase timer reaches the interval time that has been set by the interval time selection bits (TBC1 and TBC0) of the timebase timer control register (TBTC).

Interrupt request is generated when the overflow flag bit is set (TBTC:TBOF = 1) while interrupts are enabled (TBTC:TBIE = 1) for the timebase timer.
5.3.1 Timebase Timer Control Register (TBTC)

Timebase timer control register (TBTC) provides the following settings:
- Selecting the interval time of the timebase timer
- Clearing the count value of the timebase timer
- Interrupt request is enabled or disabled by generation of overflow (carry)
- Overflow flag status check and clear by generation of overflow (carry)

### Figure 5.3-1 Timebase timer control register (TBTC)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TBC0 TBC1</td>
<td>Interval time select bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HCLK : Oscillation clock&lt;br&gt;Descriptions within the ( ) indicate calculation examples when operated at an oscillation clock of 4 MHz.</td>
</tr>
<tr>
<td>15</td>
<td>TBIE</td>
<td>Interrupt enable bit&lt;br&gt;0 = interrupt disabled&lt;br&gt;1 = Interrupt enabled</td>
</tr>
<tr>
<td>14</td>
<td>TBR</td>
<td>Timebase timer clear bit&lt;br&gt;0 = No effect&lt;br&gt;1 = <em>1</em> is always read</td>
</tr>
<tr>
<td>13</td>
<td>TBOF</td>
<td>Overflow flag bit&lt;br&gt;0 = Not reach the interval time that has been set&lt;br&gt;1 = Reach the interval time that has been set</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>Reserved bit&lt;br&gt;1 = Always set this bit to <em>1</em></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>Reserved bit&lt;br&gt;1 = Always set this bit to <em>1</em></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>R/W : Readable and Writable&lt;br&gt;— : Unused&lt;br&gt;X : Undefined&lt;br&gt;: Reset value</td>
</tr>
</tbody>
</table>
### Table 5.3-1  Functions of Timebase Timer Control Register (TBTC)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit9  | TBC1, TBC0: Interval time selection bits| - These bits set interval times of timebase timer  
|       |                                        | - When the counter value of the timebase timer reaches the interval time value that has been set, overflow (carry) of the supported timebase timer counter bit is generated, and the overflow flag bit will be set (TBTC: TBOF=1). |
| bit10 | TBR: Timebase timer clear bit           | The counter value of the timebase timer is cleared.                                                                                                                                                    |
|       |                                        | **When the bit is set to "0"**: The counter value of timebase timer is cleared to "00000H" and also cleared overflow flag bit (TBOF).  
|       |                                        | **When the bit is set to "1"**: No effect.                                                                                                                                                    |
|       |                                        | **Read**: "1" is always read.                                                                                                                                                                         |
| bit11 | TBOF: Overflow flag bit                 | This is set to "1" when the counter value of the timebase timer reaches the value that has been set by the interval time selection bit.  
|       |                                        | When an overflow (carry) occurs (WTOF = 1) with interrupt request enabled (WTIE = 1), an interrupt request is generated.  
|       |                                        | **When the bit is set to "0"**: This TBOF bit will be cleared.  
|       |                                        | **When the bit is set to "1"**: No effect.  
|       |                                        | - When the overflow (carry) of the timebase timer counter bit that supports the interval time that has been set by the interval time selection bit (TBC1, TBC0) is generated, the overflow flag bit is set to "1". |
|       |                                        | **Note:**  
|       |                                        | 1) When this TBOF bit is cleared, execute it after setting disable for generation of interruption request (TBIE = 0) or setting disable for interruption using the ILM bit of the processor status.  
|       |                                        | 2) This TBOF bit is cleared when the timebase timer is cleared by the TBR bit, the reset and clock mode are switched, or on shifting to the stop mode or hardware standby are generated. |
| bit12 | TBIE: Interrupt enable bit              | Generation of interruption request is enabled or disabled through overflow (carry) of the timebase timer counter.  
|       |                                        | **When the bit is set to "0"**: Interrupt request is not generated even at overflow (WTOF = 1)  
|       |                                        | **When the bit is set to "1"**: Interrupt request is generated at overflow (WTOF = 1)  
| bit13 | Unused bits                             | **Read**: The value is undefined.  
|       |                                        | **Write**: No effect                                                                                                                                                                                |
| bit14 | Reserved: reserved bit                  | Always set this bit to "1".                                                                                                                                                                           |

---
5.4 Interrupt of Timebase Timer

Interrupt request is generated when the counter value of the timebase timer reaches the interval time, and the overflow flag bit is set to "1" while interruption of the timebase timer is enabled.

■ Interrupt of Timebase Timer

- The timebase timer continuously counts up only when the main clock (2-divided frequency of the oscillation clock HCLK) is operated.
- "1" is set to the TBTC overflow flag bit (TBTC: TBOF=1) when the counter value of the timebase timer reaches the value that has been set by the interval time selection bits (TBC1 and TBC0) of the timebase timer control register (TBTC).
- Interrupt request is generated when the overflow flag bit is set while interrupts are enabled (TBTC:TBIE = 1) for the timebase timer.

Notes: When the overflow flag bit is cleared, execute it after setting one of the following using interrupt processing.
- Interrupt of Timebase Timer is disabled. (TBTC: TBIE=0)
- Interruption of the timebase timer is disabled by the ILM bit of the processor status (PS).

■ Correspondence between Timebase Timer Interrupt and EI²OS

- The timebase timer does not correspond to EI²OS function.

Reference:
- Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.
5.5 Explanation of Operations of Timebase Timer

The timebase timer operates as the interval timer or main clock oscillation stabilization wait time timer. The operating clock is also supplied to peripheral functions.

### Interval Timer Function

The timebase timer can be used as an interval timer by generating an interrupt at each interval time.

Operating the timebase timer as an interval timer requires the settings shown in Figure 5.5-1.

#### Setting of timebase timer

![Figure 5.5-1 Setting of timebase timer](image)

- : Unused bit
- : Used bit
0 : Set to 0
1 : Set to 1

#### Operations of the Interval Timer Functions

The timebase timer can be used as an interval timer by generating an interrupt at every set interval time.

- The timebase timer continuously counts
- Up in synchronization with the main clock (2-divided frequency of the oscillation clock HCLK) while the oscillation clock (HCLK) is operated.
- "1" is set to the TBTC overflow flag bit (TBTC: TBOF=1) when the counter value of the timebase timer reaches the value that has been set by the interval time selection bits (TBC1 and TBC0) of the timebase timer control register (TBTC).
- Interrupt request is generated when the overflow flag bit is set (TBTC:TBOF = 1) while interrupts are enabled (TBTC:TBIE = 1) for the timebase timer.

**Reference:** When the counter value of the timebase timer is cleared, the interval time will be longer than the set value.
Example of operation of timebase timer

Figure 5.5-2 gives an example of the operation that the timebase timer performs under the following conditions:

- A power on reset occurs.
- The mode transits to the sleep mode during the operations of the interval timer functions.
- The mode transits to the stop mode during the operations of the interval timer functions.
- When clearing the counter value of the timebase timer is requested.

The timebase timer counter is cleared to "0" and counting is stopped when changing to the stop mode. The timebase timer counts the main clock oscillation stabilization wait time when returning from the stop mode.

**Figure 5.5-2 Example of operation of timebase timer**

- Counter value
- Oscillation stabilization
- Overflow
- 00000
- CPU ready for operation
- Power - on reset
- Interval cycle
- (TBTC : TBC1 : TBC0=11B)
- Clearing through interrupt processing
- Counter clear
- (TBTC : TBR=0)
- Clearing through shift to the stop mode
- TBOF bit
- TIE bit
- SLP bit
- (LPMCR register)
- STP bit
- (LPMCR register)
- When "11e" is set to Interval time select bit
- (TBTC : TBC1, TBC0) (2^16/HCLK)
- : Oscillation stabilization time
- HCLK : Oscillation clock
- Sleep
- Cancellation of sleep by Timebase timer interval interruption
- Stop
- Cancellation of stop by External interrupt circuit
Operation as Oscillation Stabilization Wait Time Timer

The timebase timer can function as the main clock and PLL clock oscillation stabilization wait time timer.

- The oscillation clock (HCLK) is stopped after the power is turned on or under the main stop mode, watch mode, sub-clock mode, hardware standby mode. When operation of the oscillation clock (HCLK) starts, main clock oscillation stabilization wait time is taken using the timebase timer.
- The main clock oscillation stabilization wait time is the time counting up from the counter value "0" of the timebase timer to when the time bit for the main clock oscillation stabilization wait time overflows.
- Set the main clock oscillation stabilization wait time using the clock selection register (CKSCR). Main clock oscillation stabilization wait time is fixed to $2^{18}$ or $2^{17}$ oscillation clock cycles ($2^{18}/HCLK$ or $2^{17}/HCLK$) for power on resets.

Oscillation stabilization wait time of PLL clock at recovering from timebase timer

- End the PLL clock oscillation stabilization wait time detects the time from the falling edge to rising edge ($2^{13}/HCLK$) of the timebase timer output. As a delay of up to 1 cycle ($2 \times 2^{13}/HCLK$) may occur before actually detecting the falling edge of the timebase timer output, a period equivalent to 2 to 3 $2^{13}/HCLK$ is required to change to the PLL clock operation.

Table 5.5-1 shows clearing conditions and oscillation stabilization wait time of timebase timer.

Table 5.5-1 Clearing Conditions and Oscillation Stabilization Wait Time of Timebase Timer (1/2)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Counter clear</th>
<th>Oscillation Stabilization Wait Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timebase timer counter Writing &quot;0&quot; to the clear bit (TBTC:TBR)</td>
<td>○</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power on reset</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>Cancellation of hardware standby mode</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>Watchdog reset</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>External reset</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Software reset</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Switching clock modes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Main clock $\rightarrow$ PLL clock (CKSCR: MCS = 1 $\rightarrow$ 0)</td>
<td>○</td>
<td>Transition to PLL clock mode after oscillation stabilization wait time of PLL clock completed</td>
</tr>
<tr>
<td>Main clock $\rightarrow$ sub-clock (CKSCR: SCS = 1 $\rightarrow$ 0)</td>
<td>○</td>
<td>None</td>
</tr>
<tr>
<td>Sub-clock $\rightarrow$ main clock (CKSCR: SCS = 0 $\rightarrow$ 1)</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>Sub-clock $\rightarrow$ PLL clock (CKSCR: MCS = 0, SCS = 0 $\rightarrow$ 1)</td>
<td>○</td>
<td>Transition to PLL clock mode after oscillation stabilization wait time of main clock and PLL clock completed</td>
</tr>
<tr>
<td>PLL clock $\rightarrow$ main clock (CKSCR: MCS = 0 $\rightarrow$ 1)</td>
<td>×</td>
<td>None</td>
</tr>
</tbody>
</table>
Table 5.5-1 Clearing Conditions and Oscillation Stabilization Wait Time of Timebase Timer (2 / 2)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Clear</th>
<th>Oscillation Stabilization Wait Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLL clock→sub clock (CKSCR: MCS = 0, SCS = 1→0)</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Cancellation of the stop mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cancellation of main stop mode</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>Cancellation of sub-stop mode</td>
<td>○</td>
<td>Transition to sub-clock mode after oscillation stabilization wait time of sub-clock completed</td>
</tr>
<tr>
<td>Cancellation of watch mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cancellation of main watch mode</td>
<td>○</td>
<td>Transition to main clock mode after oscillation stabilization wait time of main clock completed</td>
</tr>
<tr>
<td>Cancellation of sub watch mode</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Cancellation of PLL watch mode</td>
<td>○</td>
<td>Transition to PLL clock mode after oscillation stabilization wait time of main clock and PLL clock completed</td>
</tr>
<tr>
<td>Cancellation of timebase timer mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Return to PLL clock mode</td>
<td>×</td>
<td>Transition to PLL clock mode after oscillation stabilization wait time of PLL clock completed</td>
</tr>
<tr>
<td>Cancellation of sleep modes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cancellation of main sleep mode</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Cancellation of sub-sleep mode</td>
<td>×</td>
<td>None</td>
</tr>
<tr>
<td>Cancellation of PLL sleep mode</td>
<td>×</td>
<td>None</td>
</tr>
</tbody>
</table>

### Supply of Operation Clock

The timebase timer supplies an operation clock to the PPG timers and the watchdog timer.

**Note:** When the timebase counter is cleared, the count clock of the watchdog timer and PPG timer using the output from the timebase timer will be mismatched, so care must be taken.
5.6 Precautions when Using Timebase Timer

Care must be taken concerning the following points when the timebase timer is used.

- **Precautions when Using Timebase Timer**
  
  - **When the overflow flag is cleared**
    
    Clear the overflow flag bit of the timebase timer control register (TBTC:TBOF = 0) to "0" after disabling interrupt generation (TBTC:TBIE = 0), or disabling interruption of the timebase timer using the interrupt level mask register of the processor status.
  
  - **When the timebase timer counter value is cleared**
    
    Care must be taken when the timebase timer counter is cleared as the following operations are affected.
    
    - When the timebase timer is used as the interval timer (interruption at the interval time will be misaligned).
    - When the clock supplied from the timebase timer is used as the count clock of the watchdog timer, the interval time of the watchdog timer is misaligned.
    - When the clock supplied from the timebase timer is used as the operation clock of the PPG timer the count clock is misaligned and cannot output waveform.
  
  - **When the timebase timer as main clock oscillation stabilization wait time timer is used**
    
    - The oscillation clock (HCLK) is stopped after the power is turned on or under the main stop mode, watch mode, sub-clock mode, or hardware standby mode, so when operation of the oscillation clock (HCLK) starts, the main clock oscillation stabilization wait time will be taken by the timebase timer.

- **Reference:** Refer to "3.7.5 Oscillation Stabilization Wait Time" for details of the main clock oscillation stabilization wait time.

- **Effects to the supplied clock by Clearing timebase timer**
  
    - When the timebase timer counter is cleared, the clock supplied from the timebase timer will be the "L" level. When the timebase timer is cleared, the "L" level from the interval clock output may be up to 1/2 cycle longer.
    
    - The watchdog timer performs normal counting because the watchdog timer counter and timebase timer counter are cleared simultaneously.

Effects on the PPG timer by clearing the timebase timer are shown in Figure 5.6-1.
Figure 5.6-1 Effect to PPG by clearing timebase timer

When the clock supplied (oscillation clock is divided by 512) from the timebase timer is selected as the operation clock of the PPG timer.
5.7 Program Example of Timebase Timer

Programming examples for the timebase timer are shown below.

### Program Example of Timebase Timer

- **Processing specification**

  \(2^{12}/\text{HCLK}\) (HCLK: oscillation clock) interval interruption is repeatedly generated. Interval time is set to approx. 1.0 ms (for HCLK: 4 MHz operation).

- **Coding example**

  \[
  \begin{align*}
  \text{ICR00} & \equiv 0000B0H ; \text{Interrupt control register} \\
  \text{TBTC} & \equiv 0000A9H ; \text{Timebase timer control register} \\
  \text{TBOF} & \equiv \text{TBTC}:3 ; \text{Overflow flag bit} \\
  \end{align*}
  \]

  \[
  \begin{align*}
  \text{;---------Main Program-------------------------------------------} \\
  \text{CODE} & \text{ CSEG} \\
  \text{START:} & \text{ ;Stack pointer (SP)} \\
  & \text{;already initialized} \\
  \text{AND} & \text{ CCR,#0BFH} ; \text{Interruption is disabled} \\
  \text{MOV} & \text{ I:ICR00,#00H} ; \text{Interrupt levels 0 (strength)} \\
  \text{MOV} & \text{ I:TBTC,#10010000B} ; \text{upper 3 bits are reserved/undefined bits} \\
  & \text{Enables interruption, TBOF clear, counter clear} \\
  & \text{Interval Time Selection } 2^{12}/\text{HCLK} \\
  \text{MOV} & \text{ ILM,#07H} ; \text{Sets ILM in PS to level 7} \\
  \text{OR} & \text{ CCR,#40H} ; \text{Interruption is enabled} \\
  \text{LOOP:} & \text{ User processing} \\
  \text{;} & \text{ BRA LOOP} \\
  \text{;---------Interrupt Program-------------------------------------------} \\
  \text{WARI:} & \text{ Clear the overflow flag} \\
  \text{;} & \text{ User processing} \\
  \text{;} & \text{ RETI} ; \text{Returning from interrupt processing} \\
  \text{CODE} & \text{ ENDS} \\
  \text{;---------Vector Settings---------------------------------------------} \\
  \text{VECT} & \text{ CSEG ABS=0FFH} \\
  \text{ORG} & \text{ 00FFCCH} ; \text{vector set in interrupt number } \#12(0C_H) \\
  \text{DSL} & \text{ WARI} \\
  \text{ORG} & \text{ 00FFDCH} ; \text{Reset vector setting} \\
  \text{DSL} & \text{ START} \\
  \text{DB} & \text{ 00H} ; \text{Sets single-chip mode} \\
  \text{VECT} & \text{ ENDS} \\
  \text{END START}
  \end{align*}
  \]
CHAPTER 6
WATCHDOG TIMER

This chapter explains the functions and operation of the watchdog timer.

6.1 Overview of Watchdog Timer
6.2 Block Diagram of Watchdog Timer
6.3 Watchdog Timer Control Register (WDTC)
6.4 Explanation of Operations of Watchdog Timer
6.5 Precautions when Using Watchdog Timer
6.6 Program Examples of Watchdog Timer
6.1 Overview of Watchdog Timer

The watchdog timer is the timer counter that can be used to prevent program malfunctions.
- The watchdog timer is a 2-bit counter that uses the timebase timer or watch timer as a count clock.
- If this is not cleared until the watchdog timer overflows, reset the CPU.

Functions of Watchdog Timer

When the watchdog timer is activated, regularly clear the watchdog timer within the interval time that has been set. If the program malfunctions performs and is not cleared within the interval time, the CPU is reset by the watchdog function. This is called watchdog reset.
- Table 6.1-1 lists the interval times of the watchdog timer. If the set up time is exceeded without the watchdog timer being cleared, a watchdog reset will be generated. The reset generation time may be changed depending on the clearing timing for the timebase timer.

<p>| Table 6.1-1 Interval Time of Watchdog Timer |</p>
<table>
<thead>
<tr>
<th>Min.</th>
<th>Max.</th>
<th>Clock Cycle</th>
<th>Min.</th>
<th>Max.</th>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approx. 3.58ms</td>
<td>Approx. 4.61ms</td>
<td>$2^{14} \pm 2^{11}$/HCLK</td>
<td>Approx. 0.438s</td>
<td>Approx. 0.563s</td>
<td>$2^{12} \pm 2^{9}$/SCLK</td>
</tr>
<tr>
<td>Approx. 14.33ms</td>
<td>Approx. 18.3ms</td>
<td>$2^{16} \pm 2^{13}$/HCLK</td>
<td>Approx. 3.500s</td>
<td>Approx. 4.500s</td>
<td>$2^{15} \pm 2^{12}$/SCLK</td>
</tr>
<tr>
<td>Approx. 57.23ms</td>
<td>Approx. 73.73ms</td>
<td>$2^{18} \pm 2^{15}$/HCLK</td>
<td>Approx. 7.000s</td>
<td>Approx. 9.000s</td>
<td>$2^{16} \pm 2^{13}$/SCLK</td>
</tr>
<tr>
<td>Approx. 458.75ms</td>
<td>Approx. 589.82ms</td>
<td>$2^{21} \pm 2^{18}$/HCLK</td>
<td>Approx. 14.00s</td>
<td>Approx. 18.00s</td>
<td>$2^{17} \pm 2^{14}$/SCLK</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock, SLCK: sub-clock
Interval time is a calculation example when operated with HCLK at 4 MHz and SCLK at 8.192 kHz.

Maximum and minimum interval time for the watchdog timer are determined by clearing the timing for the watchdog timer.

Note: The watchdog timer comprises of a 2-bit counter using the carry signal of the timebase timer or watch timer as the count clock. When the timebase timer or watch timer is cleared, the watchdog reset generation time may differ to the setting.

Factor of Stopping watchdog timer

When the watchdog timer is activated, if the following factor is generated, it will be initialized, and stopped. Suspension is not possible by other factors.
- Power on reset
- Reset by cancelling hardware standby mode
- Watchdog reset
Clearing watchdog timer

Watchdog timer is cleared to "00B" only when the following factor is generated.

- External reset
- When the software reset and watchdog control bits (WTE) of the watchdog timer control register (WDTC) are set to "0" (from the 2nd time or backward)
- Shift to sleep mode (Counter is cleared, and counting is suspended.)
- Shift to timebase timer mode (Counter is cleared, and counting is suspended.)
- Shift to watch mode (Counter is cleared, and counting is suspended.)
- Shift to stop mode (Counter is cleared and counting is suspended.)

Count clock of Watchdog timer

<sub-clock mode>
- If the sub-clock mode, always set the watchdog clock selection bit (WDCS) in the watch timer control register (WTC) to "0" to select the watch timer output. If the mode transits to the sub-clock mode with the WDCS bit setting to "1", the watchdog timer stops.

<Main clock mode or PLL clock mode>
- Under the main clock mode or PLL clock mode, the timebase timer output or watch timer output can be selected as the count clock.
- The count clock selection can be set by the watchdog clock selection bit (WDCS) of the watch timer control register (WTC).

The relationship between the clock mode and settings using the watchdog clock selection bit (WDCS) is shown in Table 6.1-2.

Table 6.1-2 Count clock of watchdog timer

<table>
<thead>
<tr>
<th>WTC: WDCS</th>
<th>During operations with the sub-clock</th>
<th>Main clock or During operations with the PLL clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Set to &quot;0&quot; (Watch timer output)</td>
<td>Watch timer output</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Timebase timer output</td>
</tr>
</tbody>
</table>
6.2 Block Diagram of Watchdog Timer

The watchdog timer consists of the following blocks:
- Count clock selector
- Watchdog timer
- Watchdog reset generator circuit
- Counter clear control circuit
- Watchdog timer control register (WDTC)

Figure 6.2-1 Block Diagram of Watchdog Timer

HCLK : Oscillation clock frequency
Count clock selector

This circuit selects the count clock of the watchdog timer from the timebase timer or watch timer output. Interval time of watchdog reset is determined.

Watchdog timer

The watchdog timer is a 2-bit up counter that uses the timebase timer output or watch timer output as a count clock.

Watchdog reset generator circuit

This circuit generates an internal reset signal when the watchdog timer overflows.

Timer clear circuit

This circuit clears the watchdog timer and starts or stops the counter.

Watchdog timer control register (WDTC)

The watchdog timer control register activates and clears the watchdog timer, sets the interval time, and checks the reset generation factor.
6.3 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) activates and clears the watchdog timer, and checks the reset factor.

Watchdog timer control register (WDTC)

### Figure 6.3-1 Watchdog timer control register (WDTC)

<table>
<thead>
<tr>
<th>Reset value</th>
<th>X X X X X X X</th>
</tr>
</thead>
<tbody>
<tr>
<td>PONR</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>STBR</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>WRST</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>ERST</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>SRST</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>WTE</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>WT0</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>WT1</td>
<td>R R R R R W W</td>
</tr>
<tr>
<td>WTE</td>
<td>R R R R R W W</td>
</tr>
</tbody>
</table>

**Interval time select bits (Calculation example when operated with HCLK at 4 MHz)**

- **At select of timebase timer**
- **At select of watch timer**

<table>
<thead>
<tr>
<th>WT1</th>
<th>WT0</th>
<th>Interval time</th>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Approx. 3.58ms</td>
<td>Approx. 4.61ms</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Approx. 14.33ms</td>
<td>Approx. 18.3ms</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Approx. 57.23ms</td>
<td>Approx. 73.73ms</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Approx. 458.75ms</td>
<td>Approx. 589.82ms</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interval time select bits (Calculation example when operated with SCLK at 8.192 kHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>At select of timebase timer</strong></td>
</tr>
<tr>
<td><strong>At select of watch timer</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WT1</th>
<th>WT0</th>
<th>Interval time</th>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Approx. 0.438s</td>
<td>Approx. 0.563s</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Approx. 3.5s</td>
<td>Approx. 4.5s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Approx. 7.0s</td>
<td>Approx. 9.0s</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Approx. 14.0s</td>
<td>Approx. 18.0s</td>
</tr>
</tbody>
</table>

**Watchdog timer control bit (WTE)**

- **0**: Activating watchdog timer (when the first writing after reset)
- **0**: Clear the watchdog timer (when the second or more writing after reset)
- **1**: No effect

**Reset Factor bit**

<table>
<thead>
<tr>
<th>PONR</th>
<th>STBR</th>
<th>WRST</th>
<th>ERST</th>
<th>SRST</th>
<th>Reset Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Power-on</td>
</tr>
<tr>
<td>*</td>
<td>1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>hardware standby mode</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>1</td>
<td>*</td>
<td>*</td>
<td>Watchdog timer</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>1</td>
<td>*</td>
<td>External reset</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>1</td>
<td>RST bit (Software reset)</td>
</tr>
</tbody>
</table>

R : Read Only
W : Write Only
* : The previous state is held
X : Indeterminate
HCLK: Oscillation Clock Frequency
SCLK: Sub - clock frequency
## Table 6.3-1 Function of watchdog Timer Control Register (WDTC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit1</td>
<td>These bits set the interval time of the watchdog timer.</td>
</tr>
<tr>
<td>WT1, WT0:</td>
<td>• These bit set with the activation of the watchdog timer simultaneously.</td>
</tr>
<tr>
<td>bit0</td>
<td>• The setting after the watchdog timer is started is ignored.</td>
</tr>
<tr>
<td>bit2</td>
<td>WTE: Watchdog timer control bit</td>
</tr>
<tr>
<td></td>
<td>• When &quot;0&quot; is set for the 1st time after reset: Watchdog timer is activated.</td>
</tr>
<tr>
<td></td>
<td>• When &quot;0&quot; is set for the 2nd time after reset: Watchdog timer is cleared.</td>
</tr>
<tr>
<td></td>
<td>• When the bit is set to 1: No effect.</td>
</tr>
<tr>
<td>bit7 to</td>
<td>PONR, STBR, WRST, ERST, SRST: Reset Factor Bit</td>
</tr>
<tr>
<td>bit3</td>
<td>When reset is generated with the bit indicating the reset factor, &quot;1&quot; is set to the</td>
</tr>
<tr>
<td></td>
<td>supported reset factor bit.</td>
</tr>
<tr>
<td></td>
<td>• These bits are cleared to &quot;00000B&quot; after the watchdog timer control register (WDTC) is</td>
</tr>
<tr>
<td></td>
<td>read.</td>
</tr>
<tr>
<td></td>
<td>• Contents of bits other than this PONR bit are not guaranteed if power on reset is</td>
</tr>
<tr>
<td></td>
<td>generated. When &quot;1&quot; is set to this PONR bit, ignore contents of bits other than this PONR</td>
</tr>
<tr>
<td></td>
<td>bit.</td>
</tr>
</tbody>
</table>


6.4 Explanation of Operations of Watchdog Timer

Watchdog timer overflows and watchdog reset is generated.

Operations of Watchdog Timer

Settings of Figure 6.4-1 are required to operate the watchdog timer.

Figure 6.4-1 Setting of Watchdog Time

<table>
<thead>
<tr>
<th>Watchdog timer control register (WDTC)</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>PONR</td>
<td>STBR</td>
<td>WRST</td>
<td>ERST</td>
<td>SRST</td>
<td>WTE</td>
<td>WT1</td>
<td>WT0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>☰</td>
<td>☰</td>
<td>☰</td>
<td>☰</td>
<td>☰</td>
<td>☰</td>
<td>☰</td>
<td></td>
</tr>
</tbody>
</table>

- **Activating watchdog timer**
  - When the WTE bit of the watchdog timer control register (WDTC) is set to "0" for the first time after a reset, the watchdog timer is started. Simultaneously set the interval time using the WT1 and WT0 bits of the WDTC.
  - The watchdog timer can only be stopped by a power on reset, a reset by cancelling the hardware standby mode, or a watchdog reset.

- **Clearing watchdog timer**
  - When the WTE bit of the watchdog timer control register (WDTC) is set to "0", the watchdog timer can be cleared to "00B". When the counter is not cleared within the interval time that has been set, the watchdog timer overflows, and a watchdog reset is generated.
  - The watchdog counter can be cleared to "00B" by generation of a reset, or by changing to sleep mode, stop mode, watch mode, or timebase timer mode.

- **Interval Time of Watchdog Timer**
  - Figure 6.4-2 shows the relationship between the clear timing and the interval time of the watchdog timer.
  - The interval time varies with the timing of clearing the watchdog timer.
Checking reset factors

Reset causes can be checked by reading the PONR, STBR, WRST, ERST, and SRST bits of the watchdog timer control register (WDTC) after reset.
6.5 Precautions when Using Watchdog Timer

Take the following precautions when using the watchdog timer.

- **Precautions when Using Watchdog Timer**
  
  - **Stopping watchdog timer**
    
    Once the watchdog timer is activated, it cannot be stopped until a reset is generated by a power on reset, hardware standby, or watchdog timer.
    
    The watchdog timer can be cleared to "00B" by an external reset or reset using software, but the watchdog timer cannot be stopped.
  
  - **Interval time**
    
    As the carry signal for the timebase timer or watch timer is used as the count clock for interval time, the interval time of the watchdog timer may be longer than the set time by clearing the timebase timer or watch timer.
  
  - **Interval Time selection**
    
    Set the interval time concurrently with starting the watchdog timer. Resets will be ignored after start up.
  
  - **Precautions when creating program**
    
    When the watchdog timer is repeatedly cleared within the main program, set the interval time of the watchdog timer to be shorter than the processing time of the main program including interrupt processing.
6.6 Program Examples of Watchdog Timer

Program example of watchdog timer is given below:

- **Processing specification**
  - The watchdog timer is cleared each time in loop of the main program.
  - The main program must be executed once within the minimum interval time of the watchdog timer.

- **Coding example**

```assembly
WDTC EQU 0000A8H ;Watchdog timer control register
WTE EQU WDTC:2 ;Watchdog control bit

;---------Main Program------------------------------------------------
CODE CSEG
START: ;Stack pointer (SP)
        ;Already initialized
        MOV I:WDTC,#00000011B ;Activating watchdog timer
        ;Interval time of 2^{21}+2^{18} cycles is selected.
LOOP:
        CLR I:WTE ;Clearing watchdog timer
        • User processing
        •
        BRA LOOP

;---------Vector Settings---------------------------------------------
VECT CSEG ABS=0FFH
        ORG 00FFDCH ;Reset vector setting
        DSL START
        DB 00H ;Sets single-chip mode
VECT ENDS
END START
```
CHAPTER 7
WATCH TIMER

This section describes the functions and operations of the watch timer.

7.1 Overview of Watch Timer
7.2 Block Diagram of Watch Timer
7.3 Configuration of Watch Timer
7.4 Watch Timer Interrupt
7.5 Explanation of Operation of Watch Timer
7.6 Program Example of Watch Timer
CHAPTER 7  WATCH TIMER

7.1  Overview of Watch Timer

The watch timer is a 15-bit free-run counter that counts in synchronization with the sub-clock.

- Interval times can be selected from 7 types (2⁹/SCLK, 2¹⁰/SCLK, 2¹¹/SCLK, 2¹²/SCLK, 2¹³/SCLK, 2¹⁴/SCLK, or 2¹⁵/SCLK), and interrupt requests can be generated per interval time.
- An operation clock can be supplied to the oscillation stabilization wait time timer of the sub-clock and the watchdog timer.
- A sub-clock is used as the count clock regardless of the setting of the clock selection register (CKSCR).

### Interval Timer Function

- When the counter value of the watch timer counts the interval time that has been set, overflow (carry) of the watch timer is generated, and the overflow flag will be set to "1".
- Interrupt request is generated to the CPU when an interruption due to generation of overflow is enabled by the overflow interrupt enabling bit (WTIE) of the watch timer control register (WTC).
- The interval time of the watch timer can be selected from seven types shown in Table 7.1-1.

<table>
<thead>
<tr>
<th>Sub-clock Cycle</th>
<th>Interval Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCLK (122μs)</td>
<td>2⁹/SCLK (Approx. 62.5ms)</td>
</tr>
<tr>
<td></td>
<td>2¹⁰/SCLK (Approx. 125.0ms)</td>
</tr>
<tr>
<td></td>
<td>2¹¹/SCLK (Approx. 250.0ms)</td>
</tr>
<tr>
<td></td>
<td>2¹²/SCLK (Approx. 500.0ms)</td>
</tr>
<tr>
<td></td>
<td>2¹³/SCLK (Approx. 1.0s)</td>
</tr>
<tr>
<td></td>
<td>2¹⁴/SCLK (Approx. 2.0s)</td>
</tr>
<tr>
<td></td>
<td>2¹⁵/SCLK (Approx. 4.0s)</td>
</tr>
</tbody>
</table>

SCLK: Sub-clock frequency
Descriptions within the ( ) indicate calculation examples when operating at a sub oscillation clock frequency of 32.768 kHz. (SCLK: 8.192 kHz)
■ Cycle of Clock Supply

The watch timer supplies the timer for sub-clock oscillation stabilization wait time and operating clock to the watchdog timer. Table 7.1-2 shows the cycles of clocks supplied from the watch timer.

Table 7.1-2 Cycle of Clock Supply from Watch Timer

<table>
<thead>
<tr>
<th>Where to Supply Clock</th>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer for oscillation stabilization wait time of sub-clock</td>
<td>$2^{14}/SCLK$ (Approx. 2.000s)</td>
</tr>
<tr>
<td></td>
<td>$2^{10}/SCLK$ (Approx. 125.0 ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{13}/SCLK$ (Approx. 1.000s)</td>
</tr>
<tr>
<td></td>
<td>$2^{14}/SCLK$ (Approx. 2.000s)</td>
</tr>
<tr>
<td></td>
<td>$2^{15}/SCLK$ (Approx. 4.000s)</td>
</tr>
</tbody>
</table>

SCLK: sub-clock frequency
Descriptions within the () indicate calculation examples when operating at a sub oscillation clock frequency of 32.768 kHz. (SCLK: 8.192 kHz)
7.2 Block Diagram of Watch Timer

The watch timer consists of the following blocks:
- Watch timer counter
- Counter clear circuit
- Interval timer selector
- Watch timer control register (WTC)

Interrupt request number of the watch timer is as follows.
Interrupt request number: #26 (1AH)
Watch timer counter

The watch timer counter is a 15-bit up counter that uses the sub-clock (SCLK) as a count clock.

Counter clear circuit

The counter-clear circuit clears the watch timer counter.

Interval timer selector

The overflow flag of the WTC is set to "1" when the counter value of the watch timer reaches the interval time value by referring to the interval time set on the interval time selection bit (WTC2 to WTC1) of the watch timer control register (WTC).

Watch timer control register (WTC)

Selection of the interval time, clearing the counter value of the watch timer, enabling or disabling interruption, checking status of overflow (carry), and clearing the overflow flag can be executed.
7.3 Configuration of Watch Timer

Interrupt factors and register details for the watch timer are described.

- Generation of Interrupt Request from Watch Timer

  When the counter value of the watch timer reaches the interval time that has been set by the interval time selection bits (WTC2 to WTC1) of the watch timer control register (WTC), the overflow flag bit (WTC:WOF) of the WTC is set to "1".

  While interrupts are enabled on the watch timer (WTC:WTIE = 1), when the overflow flag bit is set (WTC:WTOF = 1), an interrupt request is generated to the CPU.
### 7.3.1 Watch Timer Control Register (WTC)

This section explains the functions of the watch timer control register (WTC).

#### Watch timer control register (WTC)

<table>
<thead>
<tr>
<th>R/W</th>
<th>WDCS</th>
<th>SCE</th>
<th>WTIE</th>
<th>WTOF</th>
<th>WTR</th>
<th>WTC2</th>
<th>WTC1</th>
<th>WTC0</th>
<th>WTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
</tbody>
</table>

**Reset value**: 10000000B

- **bit7**: Watchdog clock selection bit
  - 0: Main or PLL clock mode
  - 1: Under sub-clock mode
- **bit6**: Overflow interrupt enable bit
  - 0: This bit disables generation of an interrupt request
  - 1: This bit enables generation of an interrupt request
- **bit5**: Overflow flag bit
  - 0: This WTOF bit is cleared
  - 1: Overflow bit that supports the interval time that has been set by the interval time is not generated
- **bit4**: Watch timer clear bit
  - When reading is performed
  - When write is specified
  - 0: —
  - 1: "1" is always read
- **bit3**: Oscillation stabilization wait time end bit
  - 0: Subclock oscillation stabilization waiting state
  - 1: Expiration of the subclock oscillation stabilization time
- **bit2**: Interval time select bits
  - 0 0 0: 2^0/SCLK (62.5ms)
  - 0 0 1: 2^1/SCLK (125.0ms)
  - 0 1 0: 2^2/SCLK (250.0ms)
  - 0 1 1: 2^3/SCLK (500.0ms)
  - 1 0 0: 2^4/SCLK (1.0s)
  - 1 0 1: 2^5/SCLK (2.0s)
  - 1 1 0: 2^6/SCLK (4.0s)
  - 1 1 1: Setting disabled

Descriptions within the () indicate calculation examples when operated at an oscillation clock frequency of 32.768 kHz (Internal 8.192 kHz operation).
Table 7.3-1 Functions of Watch Timer Control Register (WTC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit2 to bit0</td>
<td>WTC2, WTC1, WTC0: Interval time selection bits&lt;br&gt;These bits set the interval time of the watch timer.&lt;br&gt;• When the counter value of the watch timer reaches the interval time value that has been set, overflow (carry) of the bit for the supported watch timer counter is generated, and the overflow flag bit is set. (WTC: WTOF=1)&lt;br&gt;• Simultaneously set the WTOF bit to &quot;0&quot; when setting these WTC2 to WTC0 bits.</td>
</tr>
<tr>
<td>bit3</td>
<td>WTR: Watch timer clear bit&lt;br&gt;The counter value of the watch timer is cleared.&lt;br&gt;<strong>When the bit is set to &quot;0&quot;:</strong> Counter value of watch timer is cleared to &quot;0000H&quot;.&lt;br&gt;<strong>When the bit is set to &quot;1&quot;:</strong> No effect.&lt;br&gt;<strong>Read:</strong> &quot;0&quot; is always read.</td>
</tr>
<tr>
<td>bit4</td>
<td>WTOF: Overflow flag bit&lt;br&gt;This bit is set to &quot;1&quot; when the counter value of the watch timer reaches the value set by the interval time selection bit.&lt;br&gt;When an overflow (carry) occurs (WTOF = 1) with interrupt request enabled (WTIE = 1), an interrupt request is generated.&lt;br&gt;<strong>When the bit is set to &quot;0&quot;:</strong> This WTOF bit is cleared.&lt;br&gt;<strong>When the bit is set to &quot;1&quot;:</strong> No effect.&lt;br&gt;• The overflow flag bit is set to &quot;1&quot; when the overflow (carry) of the watch timer counter bit that supports the interval time that has been set by the interval time selection bit (WTC2 to WTC0) is generated.</td>
</tr>
<tr>
<td>bit5</td>
<td>WTIE: Overflow interrupt enable bit&lt;br&gt;This bit enables or disables generation of an interrupt request when the watch timer counter overflows (carries).&lt;br&gt;<strong>When the bit is set to &quot;0&quot;:</strong> Even if an overflow is generated (WTOF = 1), no interruption request will be generated to the CPU.&lt;br&gt;<strong>When the bit is set to &quot;1&quot;:</strong> When an overflow is generated (WTOF = 1), an interruption request is generated to the CPU.</td>
</tr>
<tr>
<td>bit6</td>
<td>SCE: Oscillation stabilization wait time end bit&lt;br&gt;This bit indicates that the oscillation stabilization wait time of the sub-clock ends.&lt;br&gt;<strong>When the bit is cleared to &quot;0&quot;:</strong> Indicate the oscillation stabilization wait state&lt;br&gt;<strong>When the bit is set to &quot;1&quot;:</strong> Indicate the oscillation stabilization wait time ends&lt;br&gt;• The oscillation stabilization wait time of the sub-clock is fixed at $2^{14}$/SCLK (SCLK: sub-clock frequency).</td>
</tr>
<tr>
<td>bit7</td>
<td>WDCS: Watchdog clock selection bit&lt;br&gt;This bit selects the operation clock of the watchdog timer. This can be selected only when the main clock or PLL clock is used as the machine clock.&lt;br&gt;<strong>&lt;Main clock mode or PLL clock mode&gt;</strong>&lt;br&gt;<strong>When the bit is set to &quot;0&quot;:</strong> Selects output of watch timer as operation clock of watchdog timer.&lt;br&gt;<strong>When the bit is set to &quot;1&quot;:</strong> Selects output of timebase timer as operation clock of watchdog timer.&lt;br&gt;<strong>&lt;sub-clock mode&gt;</strong>&lt;br&gt;Always set this bit to &quot;0&quot; to set the output of the watch timer. If the mode transits to the sub-clock mode with setting to &quot;1&quot;, the watchdog timer stops.&lt;br&gt;<strong>Note:</strong>&lt;br&gt;As the watch timer and timebase timer are operated asynchronously, the watchdog timer proceeds when this WDCS bit is set from &quot;0&quot; to &quot;1&quot;. Clear the watchdog timer before and after setting.</td>
</tr>
</tbody>
</table>
7.4 Watch Timer Interrupt

While interruption of the watch timer is enabled, when the counter value of the watch timer reaches the interval time, and "1" is set to the overflow flag bit, an interrupt request is generated.

Watch Timer Interrupt

Table 7.4-1 shows the interrupt control bits and interrupt factors of the watch timer.

Table 7.4-1 Interrupt Control Bits of Watch Timer

<table>
<thead>
<tr>
<th>Interrupt factor</th>
<th>Watch timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interval time of watch timer counter</td>
<td>Interrupt request flag bit WTC: WTOF (overflow flag bit)</td>
</tr>
<tr>
<td>WTC: WTIE</td>
<td>WTC: WTOF (overflow flag bit)</td>
</tr>
</tbody>
</table>

• When the counter value of the watch timer has arrived the value set in advance to the interval time selection bits (WTC2 to WTC0) of the watch timer control register (WTC), set the WTC overflow flag bit to "1". (WTC: WTOF=1)
• While interruption is enabled for the watch timer (WTC:WTIE = 1), an interrupt request is generated to the CPU when the overflow flag bit is set to "1".
• Set the WTOF bit to "0" for interrupt processing, and cancel the interrupt request.

Watch Timer Interrupt and EI²OS Function

• The watch timer does not correspond to the EI²OS function.
• Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.
7.5 Explanation of Operation of Watch Timer

The watch timer operates as an interval timer or an oscillation stabilization wait time timer of sub-clock. Operating clock is also supplied to the watchdog timer.

■ Watch timer counter

The watch timer counter continues counting in synchronization with the sub-clock (SCLK) while the sub-clock (SCLK) is operating.

● Clearing watch timer counter

The watch timer counter will be cleared to "0000H" in the following cases.

- Power on reset
- Transition to stop mode
- Transition to hardware standby mode
- Set watch timer clear bit (WTR) of watch timer control register (WTC) to "0".

Note: When the watch timer counter is cleared, the watchdog timer and interval timer interruption using the output of the watch timer counter has an effect on the operation.

■ Interval Timer Function

The watch timer can be used as an interval timer by generating an interrupt at each interval time.

● Settings when using watch timer as interval timer

Operating the watch timer as an interval timer requires the settings shown in Figure 7.5-1.

Figure 7.5-1 Setting of Watch Timer

<table>
<thead>
<tr>
<th>Watch Timer Control Register (WTC)</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>WDCS</td>
<td>SCE</td>
<td>WTIE</td>
<td>WTOF</td>
<td>WTR</td>
<td>WTC2</td>
<td>WTC1</td>
<td>WTC0</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td></td>
</tr>
</tbody>
</table>

- When the counter value of the watch timer reaches the value that has been set by the interval time selection bits (WTC1 and WTC0) of the watch timer control register (WTC), "1" is set to the WTC overflow flag bit (WTC: WTOF=1).
- When the overflow flag bit is set (WTC: WTOF = 1) with the watch timer interrupt enabled (WTC: WTIE = 1), an interrupt request is generated.
- The overflow flag bit (WTC: WTOF) is set when the interval time is reached at the starting point of the timing at which the watch timer is finally cleared.
● Clearing overflow flag bit (WTC: WTOF)

In the following case, the watch timer is used as the timer for the sub-clock oscillation stabilization wait time, so WTOF is cleared at the same time that the mode is changed.

- Transition to stop mode
- Transition to hardware standby mode

■ Setting Operation Clock of Watchdog Timer

The clock source of the watchdog timer can be set using the watchdog clock selection bit (WDCS) of the watch timer control register (WTC).

If the sub-clock is used as a machine clock, always set the WDCS bit to "0" to select the watch timer output. If the mode transits to the sub-clock mode with the WDCS bit setting to "1", the watchdog timer stops.

■ Oscillation Stabilization Wait Time Timer of sub-clock

The watch timer functions as the sub-clock oscillation stabilization wait time timer in cases of a power on reset, returning from the stop mode and returning from hardware standby mode.

- Sub-clock oscillation stabilization wait time is fixed to $2^{14}/SCLK$ (SCLK: sub-clock frequency).
- Returning from the sub watch mode through external reset is not possible.

For details, see Section "3.8.7 Watch Mode".
7.6 Program Example of Watch Timer

This section gives a program example of the watch timer.

### Program Example of Watch Timer

#### Processing specification

An interval interrupt at $2^{13}/\text{SCLK}$ (SCLK: sub-clock) is generated repeatedly. The interval time is approximately 1.0s (when sub-clock operates at 8.192 kHz).

#### Coding example

```assembly
ICR07 EQU 0000B7H ;Interrupt control register
WTC EQU 0000AAH ;Watch timer control register
WTOF EQU WTC:4 ;Overflow flag bit

;---------Main Program--------------------------------------------------
CODE     CSEG
START:
; ;Stack pointer (SP)
;Already initialized
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR07,#00H ;Interrupt levels 0 (strength)
MOV I:WTC,#10100100B ;Enable interruption, clear the overflow flag
;Watch timer counter clear
;$2^{13}/\text{SCLK}$ (Approx. 1.0s)
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled

LOOP:
;User processing
;BRA LOOP

;---------Interrupt Program---------------------------------------------
WARI:
CLRB I:WTOF ;Clear the overflow flag
;User processing
;BRA LOOP
RETI ;Returning from interrupt processing

CODE   ENDS

;---------Vector Settings-----------------------------------------------
VECT    CSEG ABS=0FFH
ORG 00FF94H ;Vector set up to the interruption number
#26(1A_{16})
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode

VECT    ENDS
END START
```
CHAPTER 8
16-BIT I/O TIMER

This chapter explains the functions and operations of 16-bit input/output timer.

8.1 Overview of 16-bit Input/Output Timer
8.2 Block Diagram of 16-bit Input/Output Timer
8.3 Configuration of 16-bit Input/Output Timer
8.4 Interrupts of 16-bit Input/Output Timer
8.5 Explanation of Operation of 16-bit Free-run Timer
8.6 Explanation of Operation of Input Capture
8.7 Explanations of Operation of Output Compare
8.8 Precaution on Using 16-bit I/O Timer
8.9 Program Example of 16-bit Input/Output Timer
8.1 Overview of 16-bit Input/Output Timer

The 16-bit input/output timer is made up of two 16-bit free-run timers, two input captures, and eight output compares. 8 independent waveforms can be output, and the input pulse and external clock cycle can be measured based on the 16-bit free run timer.

Module configuration of 16-bit I/O timer

The 16-bit input/output timer consists of the following modules:

- **16-bit free-run timer x2**
  - 16-bit free-run timer 0 (channel 0)
  - 16-bit free-run timer 1 (channel 1)

- **Input capture x2**
  - Input capture (channel 0): The 16-bit free run timer 0 is captured.
  - Input capture (channel 1): The 16-bit free run timer 0 is captured.

- **Output compare x8**
  - **Output compare unit 0:** The 16-bit free run timer 0 is compared (compare/match detection)
  - Output compare 0 (Channel 0)
  - Output compare 1 (Channel 1)
  - Output compare 2 (Channel 2)
  - Output compare 3 (Channel 3)
  - **Output compare unit 1:** The 16-bit free run timer 1 is compared (compare/match detection)
  - Output compare 4 (Channel 4)
  - Output compare 5 (Channel 5)
  - Output compare 6 (Channel 6)
  - Output compare 7 (Channel 7)
Functions of 16-bit Input/Output Timer

- Functions of 16-bit free-run timer
  
  The 16-bit free run timer is made up of the 16-bit up counter, prescaler, and control register.
  The count value of the 16-bit free run timer is used as the base time for input capture and output compare.
  - The count clock cycle can be selected and set from four types.
  - Interrupt request can be generated by overflow of the counter.
  - EIO can be activated.
  - The 16-bit free run timer counter is cleared to "0000H" by a reset, timer clear (TCCS:CLR = 1), or compare match of the output compare.

- Functions of input capture
  
  The input capture is made up of two 16-bit capture registers and control registers that support the pin for external input, and the edge detection circuit.
  When the trigger edge is input to the pin for external input, the counter value of the 16-bit free run timer is retained, and interrupt requests are simultaneously generated.
  - Capture interrupts can be generated independently per channel.
  - EIO can be activated.
  - The trigger edge can be selected from rising edge, falling edge, or both edges.
  - Up to 2 inputs can be measured for each channel to be independently operated.

- Function of output compare
  
  The output compare function is made up of eight 16-bit compare registers, control registers, compare control circuits, and output control circuits.
  Output level of the supported output compare pin is reversed by a compare match between the counter value and output compare register value of the 16-bit free run timer, and an interrupt request is simultaneously generated.
  - EIO can be activated.
  - Each output compare register is provided with corresponding output pins and interrupt request flags and can operate eight output compare registers independently.
  - The detection of compare matches in both channels, ch0, 1 of output compare, makes it possible to reverse output levels. (OUT1, OUT3, OUT5 and OUT7 pin outputs only)
  - The output level of each pin can be set at startup.
8.2 Block Diagram of 16-bit Input/Output Timer

The 16-bit input/output timer consists of the following modules:
- 16-bit free-run timer
- Input capture
- Output compare

- 16-bit free-run timer
  The 16-bit free run timer counter value is used for input capture and output compare reference time.

- Input capture
  When a trigger edge is input to the input pins from outside, the 16-bit free run timer counter value is saved while an interrupt request is generated.

- Output compare
  When the 16-bit free run timer counter value and the output compare register value match, the corresponding pin output level is reversed and an interrupt request is issued.
### Details of Pins and Interrupt Numbers

Details on pins and interrupts using a 16-bit I/O timer are shown in Table 8.2-1.

**Table 8.2-1 Details of Pins and Interrupt Numbers**

<table>
<thead>
<tr>
<th>Channel</th>
<th>Specific pin</th>
<th>Pin Name</th>
<th>Interrupt Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input capture ch0 (16-bit free run timer ch0 is used)</td>
<td>IN00 pin</td>
<td>P20/IN00</td>
<td>#31(1FH)</td>
</tr>
<tr>
<td></td>
<td>IN01 pin</td>
<td>P21/IN01</td>
<td></td>
</tr>
<tr>
<td>Input capture ch1 (16-bit free run timer ch0 is used)</td>
<td>IN10 pin</td>
<td>P22/IN10</td>
<td>#32(20H)</td>
</tr>
<tr>
<td></td>
<td>IN11 pin</td>
<td>P23/IN11</td>
<td></td>
</tr>
<tr>
<td>Output compare ch0 (16-bit free run timer ch0 is used)</td>
<td>OUT0 pin</td>
<td>P32/OUT0</td>
<td>#33(21H)</td>
</tr>
<tr>
<td>Output compare ch1 (16-bit free run timer ch0 is used)</td>
<td>OUT1 pin</td>
<td>P33/OUT1</td>
<td>#34(22H)</td>
</tr>
<tr>
<td>Output compare ch2 (16-bit free run timer ch0 is used)</td>
<td>OUT2 pin</td>
<td>P34/OUT2</td>
<td>#35(23H)</td>
</tr>
<tr>
<td>Output compare ch3 (16-bit free run timer ch0 is used)</td>
<td>OUT3 pin</td>
<td>P35/OUT3</td>
<td>#36(24H)</td>
</tr>
<tr>
<td>Output compare ch4 (16-bit free run timer ch1 is used)</td>
<td>OUT4 pin</td>
<td>P70/OUT4</td>
<td>#25(19H)</td>
</tr>
<tr>
<td>Output compare ch5 (16-bit free run timer ch1 is used)</td>
<td>OUT5 pin</td>
<td>P71/OUT5</td>
<td></td>
</tr>
<tr>
<td>Output compare ch6 (16-bit free run timer ch1 is used)</td>
<td>OUT6 pin</td>
<td>P72/OUT6</td>
<td>#27(1BH)</td>
</tr>
<tr>
<td>Output compare ch7 (16-bit free run timer ch1 is used)</td>
<td>OUT7 pin</td>
<td>P73/OUT7</td>
<td></td>
</tr>
<tr>
<td>16-bit free run timer ch0 (overflow interruption, and interruption by comparing matches with the output compare ch0)</td>
<td>-</td>
<td>-</td>
<td>#14(0EH)</td>
</tr>
<tr>
<td>16-bit free run timer ch1 (overflow interruption, and interruption by comparing matches with the output compare ch4)</td>
<td>-</td>
<td>-</td>
<td>#28(1CH)</td>
</tr>
</tbody>
</table>
8.2.1 Block Diagram of 16-bit Free-run Timer

The 16-bit free run timer has two internal channels, each of which consists of the following blocks.

- **Prescaler**
  The machine clock is divided to provide the 16-bit counter with a count clock. Setting the timer control status register (TCCS:CLK1, CLK0) makes it possible to select one of four count clock cycle.

- **Timer Data Register (TCDT)**
  The timer data register can read 16-bit free run timer counter values. The counter value can be set by writing a counter value to TCDT when the 16-bit free run timer is halted.
  Timer data register 1 (TCDT1) is not equipped with an input capture module.

- **Timer control status register (TCCS)**
  The timer control status register makes it possible to select count clock and counter clear conditions, clear counter, enable count operations and interrupt requests as well as check overflow generation flags.
8.2.2 Block Diagram of Input Capture

The input capture consist of the following blocks:
- Input capture register 0, 1 (IPCP0, IPCP1)
- Input capture control status registers (ICS01)
- Edge detection circuit

- **Input capture register 0, 1 (IPCP0, IPCP1)**
  
  Input capture register stores counter values for 16-bit free-run timer 0 fetched using capture.

- **Input capture control status registers (ICS01)**
  
  The input capture control status register can set input capture ch0 and ch1 to select trigger edge, enable capture operation, enable capture interrupt request and check valid edge detection flag.

- **Edge detection circuit**
  
  The edge detection circuit detects the edge of the signal input to the external input pin. Edge detection can be selected to rising edge, falling edge, both edges or no detection (capture off).
8.2.3 Block Diagram of Output compare

Output compare consists of the following blocks.

- Block diagram of Output compare

![Block diagram of Output compare](image)

- The name of output compare unit 1

---

The name of output compare unit 1
Output compare register 0 to 7 (OCP0 to OCP7)

- The output compare register sets 16-bit free run timer counter values and performs compare operations.
- Values set in the output compare registers 0 to 3 are compared to 16-bit free run timer 0 counter values.
- Values set in the output compare registers 4 to 7 are compared to 16-bit free run timer 1 counter values.
- When values set in the output compare register 0 to 7 match values in the free run timer counter, the output level of the output compare output pin is reversed and an interrupt request is generated.

Output compare control status register 01 to 67 (OCS01 to OCS67)

- Output compare control status register sets and checks the output level of the output compare pin, enables output compare pin output, selects output level reverse mode, enables and checks compare match interrupts, enables output compare operations, etc.
- There are four output compare control status registers. As shown in Table 8.2-2, they control output compare operation of corresponding channels.

<table>
<thead>
<tr>
<th>Output compare unit 0</th>
<th>Register Name</th>
<th>Output to be controlled compare register</th>
<th>Output Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output compare control status register 01 (OCS01)</td>
<td>Output compare register 0</td>
<td>OUT0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output compare register 1</td>
<td>OUT1</td>
</tr>
<tr>
<td></td>
<td>Output compare control status register 23 (OCS23)</td>
<td>Output compare register 2</td>
<td>OUT2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output compare register 3</td>
<td>OUT3</td>
</tr>
<tr>
<td>Output compare unit 1</td>
<td>Output compare control status register 45 (OCS45)</td>
<td>Output compare register 4</td>
<td>OUT4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output compare register 5</td>
<td>OUT5</td>
</tr>
<tr>
<td></td>
<td>Output compare control status register 67 (OCS67)</td>
<td>Output compare register 6</td>
<td>OUT6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output compare register 7</td>
<td>OUT7</td>
</tr>
</tbody>
</table>

Compare control circuit 0 to 7

The compare control circuit compares 16-bit free run timer values with output compare register values and outputs a match signal to the output control circuit when a match is detected.

Output controller 0 to 7

The output control circuit reverses the output level of the output compare pin when the 16-bit free run timer value matches the output compare register value.
8.3 Configuration of 16-bit Input/Output Timer

This section describes the pins, interrupt factors, and registers of the 16-bit input/output timer.

■ Pins of 16-bit Input/Output Timer

The pins used in the 16-bit I/O timer are also used as general-purpose I/O ports. Settings using pin functions and the 16-bit I/O timer are shown in Table 8.3-1.

Table 8.3-1 Pins of 16-bit Input/Output Timer

<table>
<thead>
<tr>
<th>Channel</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>For use of pins setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input capture 0</td>
<td>P20/IN00</td>
<td>General-purpose I/O port/Capture input</td>
<td>Set up to the input port in Port direction register (DDR)</td>
</tr>
<tr>
<td></td>
<td>P21/IN01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input capture 1</td>
<td>P22/IN10</td>
<td>Set up to the input port in Port direction register (DDR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P23/IN11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output compare 0</td>
<td>P32/OUT0</td>
<td>General-purpose I/O port/Compare output</td>
<td>Output compare enables output (OCS01: OTE0=1)</td>
</tr>
<tr>
<td>Output compare 1</td>
<td>P33/OUT1</td>
<td></td>
<td>Output compare enables output (OCS01: OTE1=1)</td>
</tr>
<tr>
<td>Output compare 2</td>
<td>P34/OUT2</td>
<td></td>
<td>Output compare enables output (OCS23: OTE0=1)</td>
</tr>
<tr>
<td>Output compare 3</td>
<td>P35/OUT3</td>
<td></td>
<td>Output compare enables output (OCS23: OTE1=1)</td>
</tr>
<tr>
<td>Output compare 4</td>
<td>P70/OUT4</td>
<td></td>
<td>Output compare enables output (OCS45: OTE0=1)</td>
</tr>
<tr>
<td>Output compare 5</td>
<td>P71/OUT5</td>
<td>General-purpose I/O port/Compare output</td>
<td>Output compare enables output (OCS45: OTE1=1)</td>
</tr>
<tr>
<td>Output compare 6</td>
<td>P72/OUT6</td>
<td></td>
<td>Output compare enables output (OCS67: OTE0=1)</td>
</tr>
<tr>
<td>Output compare 7</td>
<td>P73/OUT7</td>
<td></td>
<td>Output compare enables output (OCS67: OTE1=1)</td>
</tr>
</tbody>
</table>

■ Block Diagram of Pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".
**Generation of Interrupt Request from 16-bit Input/Output Timer**

The following interrupt requests are generated in the 16-bit I/O timer.

- **Timer counter overflow interrupt**
  
  When overflow interrupt requests are enabled, (TCCS:IVFE=1) interrupt requests occur for the following reason.
  
  - Overflow in 16-bit free-run timer
  
  - Cleared when the 16-bit free run timer counter value matches output counter compare register value (TCCS:MODE=1).

- **Input capture interrupt**
  
  When input capture interrupt request are enabled (ICS01:ICE=1), interrupt requests are generated when a trigger edge is detected at the input capture pin.

- **Output compare interrupt**
  
  When output compare interrupt requests are enabled (OCS:IOE=1), an interrupt request is generated when a match between the 16-bit free run timer counter value the output compare register set value is detected.
8.3.1 Timer Control Status Register (TCCS)

The timer control status register makes it possible to select count clock and count clear conditions, clear counter, enable count operations and interrupt requests as well as check interrupt request flags.

### Timer control status register (TCCS)

**Figure 8.3-1 Timer control status register (TCCS)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>IVF</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>IVE</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>STOP</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MODE</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CLR</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>TCDT</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>TCDT</td>
<td></td>
</tr>
</tbody>
</table>

- **TCDT** (Timer Control Condition Selection) bits
  - **CLR** (Clear): Clear counter (TCDT = "0000H"
  - **Mode**: Reset, Clear by Clear bit
  - **Mode**: Reset, Clear bit, Clear by Compare register match

- **IVF** (Overflow Generation Flag): When reading is performed
  - **0**: No Timer overflow
  - **1**: With Timer overflow

- **Reserved** bit
  - **0**: Always set this bit to "0".

- **Count Clock Cycle Selection**
  - **CLK1**: 4/\phi
  - **CLK0**: 16/\phi, 64/\phi, 256/\phi

- **Machine clock frequency**

- **Reset value**: 0000 0000B
## Table 8.3-2 Timer Control status register (TCCS) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>Set the count clock cycle of 16-bit free-run timer</td>
</tr>
<tr>
<td>bit1</td>
<td>Count clock cycle selection bit</td>
</tr>
<tr>
<td>bit2</td>
<td>CLR: Timer clear bit Clear the counter (TCDT) of 16-bit free-run timer</td>
</tr>
<tr>
<td>bit3</td>
<td>MODE: Clear condition selection bits Set the conditions for the counter</td>
</tr>
<tr>
<td>bit4</td>
<td>STOP: Timer operation stop bit This bit enables or disables operation of</td>
</tr>
<tr>
<td>bit5</td>
<td>IVFE: Timer overflow interrupt enable bit Interrupt request when the</td>
</tr>
<tr>
<td>bit6</td>
<td>IVF: Timer overflow generation flag Timer overflow generation is</td>
</tr>
<tr>
<td>bit7</td>
<td>Reserved: reserved bit Always set this bit to &quot;0&quot;.</td>
</tr>
</tbody>
</table>

### Note:
- Set up selection of the count clock cycle while output compare operation is suspended (TCCS:STOP = 1) and input capture operation is also suspended (ICS01:EG01, EG00 = 00B or ICS01:EG11, EG10 = "00B").
- When clearing while the 16-bit free run timer is suspended (TCCS:STOP = 1), write "0000H" directly to the TCDT.
- When the bit is set to "0": The TCDT counter value will be cleared under the following conditions.
  - Reset
  - Set up "1" to the timer clear bit (TCCS:CLR = 1)
- When the bit is set to "1": The TCDT counter value will be cleared under the following conditions.
  - Reset
  - Set up "1" to the timer clear bit (TCCS:CLR = 1)
  - 16-bit free run timer 0 is cleared by comparing and matching between the counter value of the 16-bit free run timer 0 and the set value of the output compare register 0.
  - 16-bit free run timer 1 is cleared by comparing and matching between the counter value of the 16-bit free run timer 1 and the set value of the output compare register 4.
- When the bit is set to "0": The bit is cleared.
- When the bit is set to "1": The bit is cleared.
- When the bit is set to "1": No effect.
CHAPTER 8 16-BIT I/O TIMER

8.3.2 Timer Data Register (TCDT)

The timer data register (TCDT) is a 16-bit up counter.
- Counter value of 16-bit free-run timer can be read out.
- Counter value can be set during stop of 16-bit free-run timer.

## Timer Data Register (TCDT)

![Figure 8.3-2 Timer Data Register (TCDT)](image)

This register can read 16-bit free run timer counter values.

**Conditions when counter values are cleared**

Counter value is cleared to "0000H" when the following conditions apply.

- Overflow generation
- Compare match generation (only when TCCS:MODE=1)
- Setting "1" (TCCS:CLR=1) in the timer clear bit of the timer control status register
- Setting "0000H" in the timer data register when the 16-bit free run timer is paused
- Reset generation

**[Setting of counter value]**

Write counter values to the timer data register (TCDT) to set the timer when it is stopped (TCCS:STOP=1).

**Note:** The word command (MOVW) must be used to read/write in the timer data register.
8.3.3 Input Capture Control Status Registers (ICS01)

Input capture control status register functions are described below.

- Input capture control status registers (ICS01)

Figure 8.3-3 Input capture control status registers (ICS01)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP1</td>
<td>ICP0</td>
<td>ICE1</td>
<td>ICE0</td>
<td>EG11</td>
<td>EG10</td>
<td>EG01</td>
<td>EG00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mask</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W</td>
<td>Readable and Writable</td>
</tr>
<tr>
<td>Reset value</td>
<td>00000000 B</td>
</tr>
</tbody>
</table>

- **Edge selection bit 0 (EG00)**
  - 00: No detection of an edge (Operation stop status)
  - 01: Rising edge detection
  - 10: Falling edge detection
  - 11: Double-edge detection

- **Edge selection bit 1 (EG10)**
  - 00: No detection of an edge (Operation stop status)
  - 01: Rising edge detection
  - 10: Falling edge detection
  - 11: Double-edge detection

- **Capture interrupt enable bit 0 (ICE0)**
  - 0: Interruption of Input Capture 0 is disabled
  - 1: Interruption of Input Capture 0 is enabled

- **Capture interrupt enable bit 1 (ICE1)**
  - 0: Interruption of Input Capture 1 is disabled
  - 1: Interruption of Input Capture 1 is enabled

- **Valid edge detection flag bit 0 (ICP0)**
  - When reading is performed
    - 0: Without effective edge detection of input capture 0
    - 1: With effective edge detection of input capture 0
  - When write is specified
    - 0: Clear this ICP0 bit
    - 1: No effect

- **Valid edge detection flag bit 1 (ICP1)**
  - When reading is performed
    - 0: Without effective edge detection of input capture 1
    - 1: With effective edge detection of input capture 1
  - When write is specified
    - 0: Clear this ICP1 bit
    - 1: No effect
### Table 8.3-3 Input Capture Control Status Register (ICS01) function

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>EG01, EG00: Edge selection bits 0</td>
<td>Trigger edge of the capture operation is set for the input capture register 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set up for trigger edge is also used for enabling and stopping operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;00B&quot;:</strong> Edge detection is not executed, and capture operation is stopped.</td>
</tr>
<tr>
<td>bit1</td>
<td>EG01, EG00: Edge selection bits 0</td>
<td></td>
</tr>
<tr>
<td>bit2</td>
<td>EG11, EG10: Edge selection bits 1</td>
<td>Trigger edge of the capture operation is set for the input capture register 1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set up for trigger edge is also used for enabling and stopping operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;00B&quot;:</strong> Edge detection is not executed, and capture operation is stopped.</td>
</tr>
<tr>
<td>bit3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit4</td>
<td>ICE0:</td>
<td>Interrupt request for input capture 0 is enabled or disabled.</td>
</tr>
<tr>
<td></td>
<td>Capture interrupt enable bit 0</td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> When the valid edge detection flag bit 0 is set to &quot;1&quot; (ICS01:ICP0 = 1), an interrupt request is generated.</td>
</tr>
<tr>
<td>bit5</td>
<td>ICE1:</td>
<td>Interrupt request for input capture 1 is enabled or disabled.</td>
</tr>
<tr>
<td></td>
<td>Capture interrupt enable bit 1</td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> When the valid edge detection flag bit 1 is set to &quot;1&quot; (ICS01:ICP1 = 1), an interrupt request is generated.</td>
</tr>
<tr>
<td>bit6</td>
<td>ICP0:</td>
<td>&quot;1&quot; will be set when the valid edge is detected by the IN00 or IN01 pin.</td>
</tr>
<tr>
<td></td>
<td>Valid edge detection flag bit 0</td>
<td>• While interrupt requests for input capture 0 are enabled (ICS01:ICE0 = 1), interrupt requests are generated when this ICP0 bit is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> No effect.</td>
</tr>
<tr>
<td>bit7</td>
<td>ICP1:</td>
<td>&quot;1&quot; will be set when a valid edge is detected by the IN10 or IN11 pin.</td>
</tr>
<tr>
<td></td>
<td>Valid edge detection flag bit 1</td>
<td>While interrupt requests for input capture 1 are enabled (ICS01:ICE0 = 1), interrupt requests are generated when this ICP1 bit is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> No effect.</td>
</tr>
</tbody>
</table>
### 8.3.4 Input Capture Register (IPCP)

- The input capture register stores the counter value fetched from the 16-bit free run timer during capture operation.
- There is a 16-bit read-only register, input capture register 0 and 1 (IPCP0, IPCP1).

#### Input capture register (IPCP)

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>D15</td>
<td>D14</td>
<td>D13</td>
<td>D12</td>
<td>D11</td>
<td>D10</td>
<td>D9</td>
<td>D8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>D6</td>
<td>D5</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXXXXXXB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXXXXXXB</td>
</tr>
</tbody>
</table>

When capture operation trigger edge (set by ICS01:EG01, EG00 or EG11, EG10) is detected in the IN00 pin or IN01 pin, IN10 pin or IN11 pin, the 16-bit free run timer count value is stored in input capture register 0 and 1 in the corresponding each pin.

When the edge (ICS01:EG01, EG00 or EG11, EG10) set at the trigger input pin of IN00 or IN01, IN10 or IN11 pins is detected, the 16-bit free run timer count value is stored in the corresponding input capture registers.

- The input capture register is read-only and disables write operations.

### Note:
The word command (MOVW) must be used to read the input capture register.
8.3.5 High-order Output Compare Control Status Register (OCS)

The high-order in the output compare control status register (OCS) enables output compare pin output, sets and checks output levels, selects output level reverse mode. Figure 8.3-5 shows output pin OCS01.

Pair the following pin names and channel names other than OCS01.

OCS01: OUT0, OUT1 Output compare ch0, ch1

OCS23: OUT2, OUT3 Output compare ch2, ch3
OCS45: OUT4, OUT5 Output compare ch4, ch5
OCS67: OUT6, OUT7 Output compare ch6, ch7

---

High-order output compare control status register (OCS)

**Figure 8.3-5 High-order output compare control status register (OCS)**

<table>
<thead>
<tr>
<th>R/W</th>
<th>OTD0</th>
<th>OTD1</th>
<th>OTE0</th>
<th>OTE1</th>
<th>CMOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reset value**

XXX00000 B

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Output level set bit 0</td>
<td>When reading is performed: Output level of OUT0 terminal. When write is specified: Set OUT0 output to &quot;H&quot;.</td>
</tr>
<tr>
<td>6</td>
<td>Output level set bit 1</td>
<td>When reading is performed: Output level of OUT1 terminal. When write is specified: Set OUT1 output to &quot;H&quot;.</td>
</tr>
<tr>
<td>5</td>
<td>Compare output enable bit 0</td>
<td>General - purpose I/O port.</td>
</tr>
<tr>
<td>4</td>
<td>Output compare output (OUT0)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Compare output enable bit 1</td>
<td>General - purpose I/O port.</td>
</tr>
<tr>
<td>2</td>
<td>Output compare output (OUT1)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Output level reverse mode selection bit</td>
<td>Output level for the corresponding pin is reversed when compare match is detected with output compare ch0, and that of OUT1 pin is reserved when compare match is detected with output compare ch0 or ch1.</td>
</tr>
</tbody>
</table>

: Readable and Writable

: Undefined

: Unused

: Reset value
Table 8.3-4  Output compare control status register (OCS) high-order function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit8 OTD0: Output level setting bit 0 | Output level for the OUT0 pin is set up.  
  - When output of the output compare pin is enabled (OCS:OTE0 = 1), "L" or "H" that has been set to this OTD0 bit is output from the OUT0 pin.  
  - Set up this OTD0 bit while the output compare operation is stopped (OCS:CST0 = 0).  
  **Read:** Output level of the OUT0 pin is read. |
| bit9 OTD1: Output level setting bit 1 | Output level for the OUT1 pin is set up.  
  - When output of the output compare pin is enabled (OCS:OTE1 = 1), "L" or "H" that has been set to this OTD1 bit is output from the OUT1 pin.  
  - Set up this OTD1 bit while the output compare operation is stopped (OCS:CST1 = 0).  
  **Read:** Output level of the OUT1 pin is read. |
| bit10 OTE0: Compare output enable bit 0 | Output of the output compare pin OUT0 is enabled.  
  **When the bit is set to "1":** This will be the output compare output pin.  
  **When the bit is set to "0":** This will be general-purpose I/O port. |
| bit11 OTE1: Compare output enable bit 1 | Output of the output compare pin OUT1 is enabled.  
  **When the bit is set to "1":** This will be the output compare output pin.  
  **When the bit is set to "0":** This will be the general-purpose I/O port. |
| bit12 CMOD: Output level conversion mode selection bits | Detection factors for compare matches that will be the conditions for reversing the pin output level are set.  
  **When the bit is set to "0":** Output level for the corresponding pin is reversed when compare match is detected with output compare ch0 and ch1.  
  **When the bit is set to "1":** Output level of the OUT0 pin is reversed when compare match is detected with output compare ch0, and that of OUT1 pin is reserved when compare match is detected with output compare ch0 or ch1. If the output compare registers 0 and 1 set the same value, the operation will be the same as the channel of the output compare register 1.  
  - Table 8.3-5 shows how to detect compare match when the output level reverse mode is switched. |
| bit13 to bit15 Undefined bits | **Read:** The value is undefined.  
  **Write:** No effect |
### Table 8.3-5  Output level reverse mode and compare match detection

<table>
<thead>
<tr>
<th>Output level reverse mode</th>
<th>Control register name</th>
<th>Comparison is performed by the compare operation register name</th>
<th>Outputs reverse pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOD=0</td>
<td>OCS01</td>
<td>16-bit free-run timer 0: OCP0</td>
<td>OUT0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 0: OCP1</td>
<td>OUT1</td>
</tr>
<tr>
<td></td>
<td>OCS23</td>
<td>16-bit free-run timer 0: OCP2</td>
<td>OUT2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 0: OCP3</td>
<td>OUT3</td>
</tr>
<tr>
<td></td>
<td>OCS45</td>
<td>16-bit free-run timer 1: OCP4</td>
<td>OUT4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 1: OCP5</td>
<td>OUT5</td>
</tr>
<tr>
<td></td>
<td>OCS67</td>
<td>16-bit free-run timer 1: OCP6</td>
<td>OUT6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 1: OCP7</td>
<td>OUT7</td>
</tr>
<tr>
<td>CMOD=1</td>
<td>OCS01</td>
<td>16-bit free-run timer 0: OCP0</td>
<td>OUT0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 0: OCP0</td>
<td>OUT1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 0: OCP1</td>
<td>OUT1</td>
</tr>
<tr>
<td></td>
<td>OCS23</td>
<td>16-bit free-run timer 0: OCP2</td>
<td>OUT2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 0: OCP3</td>
<td>OUT3</td>
</tr>
<tr>
<td></td>
<td>OCS45</td>
<td>16-bit free-run timer 1: OCP4</td>
<td>OUT4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 1: OCP5</td>
<td>OUT5</td>
</tr>
<tr>
<td></td>
<td>OCS67</td>
<td>16-bit free-run timer 1: OCP6</td>
<td>OUT6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16-bit free-run timer 1: OCP7</td>
<td>OUT7</td>
</tr>
</tbody>
</table>
8.3.6 Low-order Output Compare Control Status Register (OCS)

Output compare control status register functions are described below. Figure 8.3-6 gives the output compare channel name for OCS01. Pair the following channel names other than OCS01.

- OCS01: Output compare ch0, ch1
- OCS23: Output compare ch2, ch3
- OCS45: Output compare ch4, ch5
- OCS67: Output compare ch6, ch7

Figure 8.3-6 Low-order output compare control status register (OCS)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Readable and Writable</th>
<th>When reading is performed</th>
<th>When write is specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>CST0: Compare operation Enable Bit 0</td>
<td>R/W</td>
<td>0: Output compare ch0 operation is disabled.</td>
<td>Clear this IOP0 bit</td>
</tr>
<tr>
<td>6</td>
<td>CST1: Compare operation Enable Bit 1</td>
<td>R/W</td>
<td>1: Output compare ch0 operation is enabled.</td>
<td>No effect</td>
</tr>
<tr>
<td>5</td>
<td>IOE0: Compare match Interrupt enable bit 0</td>
<td>R/W</td>
<td>0: Output compare ch0 interruption is disabled.</td>
<td>No effect</td>
</tr>
<tr>
<td>4</td>
<td>IOE1: Compare match Interrupt enable bit 1</td>
<td>R/W</td>
<td>1: Output compare ch0 interruption is enabled.</td>
<td>No effect</td>
</tr>
<tr>
<td>3</td>
<td>IOP0: Compare match flag bit 0</td>
<td>R/W</td>
<td>0: Without compare match detection of output compare ch0</td>
<td>No effect</td>
</tr>
<tr>
<td>2</td>
<td>IOP1: Compare match flag bit 1</td>
<td>R/W</td>
<td>1: With compare match detection of output compare ch0 (output compare 2)</td>
<td>Clear this IOP1 bit</td>
</tr>
<tr>
<td>1</td>
<td>IOP2: Compare match flag bit 2</td>
<td>R/W</td>
<td>0: Without compare match detection of output compare ch1</td>
<td>No effect</td>
</tr>
<tr>
<td>0</td>
<td>IOP3: Compare match flag bit 3</td>
<td>R/W</td>
<td>1: With compare match detection of output compare ch1 (output compare 3)</td>
<td>Clear this IOP3 bit</td>
</tr>
</tbody>
</table>
### Table 8.3-6 The output compare control status register (OCS) lower function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit0 CST0: Compare operation enable bit 0 | The compare operation of output compare ch0 is enabled or disabled.  
  When the bit is set to "0": Compare operation is disabled.  
  When the bit is set to "1": Compare operation is enabled.  
  Note: When the 16-bit free run timer is set to stop (TCCS0:STOP = 1), operation of the output compare ch0 is stopped. |
| bit1 CST1: Compare operation enable bit 1 | The compare operation of output compare ch1 is enabled or disabled.  
  When the bit is set to "0": Compare operation is disabled.  
  When the bit is set to "1": Compare operation is enabled.  
  Note: When the 16-bit free run timer is set to stop (TCCS0:STOP = 1), operation of the output compare ch1 is stopped. |
| bit2 bit3 Undefined bits | Read: The value is undefined.  
  Write: No effect |
| bit4 IOE0: Compare match interrupt enable bit0 | Interrupt request is enabled or disabled when the set value of the output compare register 0 and the counter value of the 16-bit free run timer 0 match.  
  When the bit is set to "1": Interrupt request is generated when the compare match flag bit 0 (OCS:IOP0) is set to "1". |
| bit5 IOE1: Compare match interrupt enable bit 1 | Interrupt request is enabled or disabled when the set value of the output compare register 1 and the counter value of the 16-bit free run timer 0 match.  
  When the bit is set to "1": Interrupt request is generated when the compare match interrupt request flag bit 1 (OCS:IOP1) is set to "1". |
| bit6 IOP0: Compare match flag bit 0 | "1" will be set when a match between the set value for the output compare register 0 and the counter value of the 16-bit free run timer 0 is detected.  
  • While the compare match interrupt request is enabled (OCS:IOE0 = 1), when this IOP0 bit is set to "1", an interrupt request is generated.  
  When the bit is set to "0": The bit is cleared.  
  When the bit is set to "1": No effect.  
  Note: When read is performed by the read-modify-write command, "1" is always read. |
| bit7 IOP1: Compare match flag bit 1 | "1" will be set when a match between the set value for the output compare register 1 and the counter value of the 16-bit free run timer 0 is detected.  
  • While the compare match interrupt request is enabled (OCS:IOE1 = 1), when this IOP1 bit is set to "1", an interrupt request is generated.  
  When the bit is set to "0": The bit is cleared.  
  When the bit is set to "1": No effect.  
  Note: When read is performed by the read-modify-write command, "1" is always read. |
8.3.7 Output Compare Register

The output compare register sets the value to be compared with the counter value of the 16-bit free run timer.
• The output compare register 0 to 7 (OCP0 to OCP7) has eight channels.

Output compare register (OCP)

The output compare register values are compared with free run timer counter values. When matches are detected, the level of the output compare output pin is reversed and an interrupt request is generated.
Value after reset and clearing is undefined. Set the value to be compared in the output compare register before enabling compare operation. (OCS: CST=1)

Note: The word command (MOVW) must be used to read/write the output compare register.
8.4 Interrupts of 16-bit Input/Output Timer

- The 16-bit I/O timer interrupt factors include 16-bit free run timer counter value overflow, trigger edge input to the input capture input pin and detection of output compare match.
- The EI^2OS can be started by any interrupt.

### Interrupts of 16-bit Input/Output Timer

Table 8.4-1 shows the interrupt control bits and interrupt factors of the 16-bit input/output timer.

<table>
<thead>
<tr>
<th>Interruption demand flag</th>
<th>Timer/counter overflow interrupt</th>
<th>Input capture interrupt</th>
<th>Output compare interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCCS: IVF</td>
<td>ICS01: ICP0, ICP1</td>
<td>OCS: IOP0, IOP1</td>
<td></td>
</tr>
<tr>
<td>Interruption demand output enable bit</td>
<td>TCCS: IVFE</td>
<td>ICS01: ICE0, ICE1</td>
<td>OCS: IOE0, IOE1</td>
</tr>
<tr>
<td>Interruption demand factor</td>
<td>16-bit free run timer counter overflow</td>
<td>Input capture, Valid edge input to the input pin</td>
<td>Output compare, Matching between the register value and counter value</td>
</tr>
</tbody>
</table>

- **Timer counter overflow interrupt**

  **When a timer overflow interrupt request flag is set**
  - In the following cases, the timer overflow generation flag of the timer control status register is set. (TCCS: IVF=1)
    - When an overflow ("FFFFH" → "0000H") occurs in the count up of the 16-bit free run timer
    - When clearing is enabled (TCCS: MODE=1) via compare matches, and the next compare match occurs.
      - 16-bit free run timer 0 set value and output compare register 0 value match
      - 16-bit free run timer 1 set value and output compare register 4 value match

  **When a timer overflow interrupt requests are generated**
  - When a timer overflow interrupt requests are enabled (TCCS:IVFE=1), interrupt requests occur when the timer overflow generation flag is set to "1" (TCCS:IVF=1).
- **Input Capture Interrupt**

  The interrupt that occurs when the set valid edge (ICS:EG) is detected at the input capture pin is described below.
  
  - The 16-bit free run timer counter value is stored in the input capture register when detected.
  - The valid edge detection flag of the input capture control status register is set to "1". (ICS01: ICP=1)
  - When the input capture interrupt request output is enabled (ICS01:ICE=1), an interrupt request is generated.

- **Output compare interrupt**

  Interrupts that occur when matches of the 16-bit free run timer count values and output compare register set values are detected are described below.
  
  - The output compare match flag of the output compare control status register is set to "1". (OCS:ICP=1)
  - When the output compare interrupt request is enabled (OCS:ICE=1), an interrupt request is generated.

- **Interrupts of 16-bit Input/Output Timer and EI²OS**

  **Reference:** Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

- **EI²OS Functions of 16-bit Input/Output Timer**

  The 16-bit I/O timer is compatible with EI²OS function.

  However, when the EI²OS function is used, other interrupts that share the interrupt control register (ICR) must be disabled.
8.5  Explanation of Operation of 16-bit Free-run Timer

The 16-bit free run timer counts up from counter value "0000\text{H}" after a reset. The 16-bit free run timer count value is set to the output compare and input compare reference time.

**Explanation of Operation of 16-bit Free-run Timer**

The settings listed in Figure 8.5-1 are required when 16-bit free run timer is operating.

**Figure 8.5-1 Setting of 16-bit Free-run Timer**

<table>
<thead>
<tr>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>bit8</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
<td>IVF</td>
<td>IVFE</td>
<td>STOP</td>
<td>MODE</td>
<td>CLR</td>
<td>CLK1</td>
<td>CLK0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Counts value of 16-bit free-run timer**

(Setting the counter value of 16-bit free-run timer)

- The 16-bit free run timer counts up from count value "0000\text{H}" since timer operation is enabled (TCCS:STOP=0) after a reset cancellation.
- To set the 16-bit free run timer counter value, disable the operation of the 16-bit free run timer (TCCS:STOP=1), set a value in the timer data register to start counting and enable timer operation (TCCS:STOP=0).

(Timer overflow and interrupt request generation)

- When an overflow ("FFFF\text{H}" → "0000\text{H}") occurs in the 16-bit free run timer, the timer overflow generation flag is set to "1" (TCCS:IVF) and count up starts from "0000\text{H}".
- When timer overflow interrupt requests are enabled (TCCS:IVFE=1), interrupt requests are generated.
(Counter value clear factor and clear timing)

- The 16-bit free run timer clear factor and clear timing are described in Table 8.5-1.

<table>
<thead>
<tr>
<th>Table 8.5-1 Counter value clear factor and clear timing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clear factor</strong></td>
</tr>
<tr>
<td>Writing &quot;1&quot; to the timer clear bit of the timer control status register (TCCS:CLR)</td>
</tr>
<tr>
<td>Writing &quot;0000H&quot; to the timer data register that is being stopped</td>
</tr>
<tr>
<td>Generation of reset</td>
</tr>
<tr>
<td>Generating timer overflow</td>
</tr>
<tr>
<td>Generation of compare match (when TCCS:MODE = 1)</td>
</tr>
</tbody>
</table>

- When clearing is enabled through compare matches (TCCS:MODE=1) and any of the following compare matches detects, the compare match flag is set to "1" (OCS:ICP), the counter value is cleared to "0000H" and counting up starts.
  - 16-bit free run timer 0 value and output compare register 0 value match
  - 16-bit free run timer 1 value and output compare register 4 value match

Counter clear timing due to compare register matches are shown in Figure 8.5-2.

**Figure 8.5-2 Clear timing of 16-bit free-run timer**

\[ \phi \]

\[ \phi : \text{Machine clock} \]

\[ \text{Count clear} \]
Figure 8.5-3 shows counter clearing at an overflow.

**Figure 8.5-3 Counter Clearing at an Overflow**

![Diagram showing counter clearing at an overflow]

Figure 8.5-4 shows counter clearing at compare match with compare register.

**Figure 8.5-4 Counter Clearing at compare match**

![Diagram showing counter clearing at compare match]
8.6  Explanation of Operation of Input Capture

The input capture stores the 16-bit free run timer counter value in the input capture register and an interrupt request is generated when the valid edge input signal from external input pin is detected.

■ Explanation of Operation of Input Capture

When input capture is used, the settings shown in Figure 8.6-1  are required.

*Figure 8.6-1  Input capture operation setting*

![Capture operation setting diagram]

- **ICS01**: Input capture control status register
- **IPCP**: Input capture pin
- **DDR Port direction register**: The bit corresponding to the pin as the capture input pin is set to "0".

*(Input capture operation)*

When the set valid edge (ICS:EG) is detected at the input capture pin, the following operation is performed:

- The 16-bit free run timer 0 counter value is stored in the input capture register when detected.
- The valid edge detection flag of the input capture control status register is set to "1" (ICS01:ICP=1).
- When the input capture interrupt request is enabled (ICS01:ICE=1), an interrupt request is generated.

Figure 8.6-2  shows the timing of input capture data retrieval. Figure 8.6-3  shows operation when rising edge/falling edge is set as valid edge. Figure 8.6-4  shows operation when both edges are set as valid edges.

*Figure 8.6-2  Timing of input capture data retrieval*

![Timing diagram]

- **Capture signal**: Valid edge
- **Capture register**: Data retrieval
- **Machine clock**: \( \phi \)
Figure 8.6-3 Operating of Input Capture (rising edge/falling edge)

<table>
<thead>
<tr>
<th>Count value</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFH</td>
<td></td>
</tr>
<tr>
<td>BFFFFH</td>
<td></td>
</tr>
<tr>
<td>7FFFFH</td>
<td></td>
</tr>
<tr>
<td>3FFFFH</td>
<td></td>
</tr>
<tr>
<td>0000H</td>
<td></td>
</tr>
</tbody>
</table>

Reset

IN00 (rising edge)
IN10 (falling edge)

Capture0: Undefined 3FFFFH
Capture1: Undefined 7FFFFH

Figure 8.6-4 Operating of Input Capture (Both edges)

<table>
<thead>
<tr>
<th>Count value</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFH</td>
<td></td>
</tr>
<tr>
<td>BFFFFH</td>
<td></td>
</tr>
<tr>
<td>7FFFFH</td>
<td></td>
</tr>
<tr>
<td>3FFFFH</td>
<td></td>
</tr>
<tr>
<td>0000H</td>
<td></td>
</tr>
</tbody>
</table>

Reset

IN00 (both edges)

Capture example: Undefined BFFFFH 3FFFFH
8.7 Explanations of Operation of Output Compare

The output compare compares set compare values and 16-bit free run timer counter values. When a match is detected, the output level of the output compare pin is reversed and an interrupt request is generated.

- Explanations of operation of output compare

When the output compare function is used, the settings shown in Figure 8.7-1 are required.

Figure 8.7-1 Output compare operation setting

<table>
<thead>
<tr>
<th>OCS0</th>
<th>bit15</th>
<th>bit14</th>
<th>bit13</th>
<th>bit12</th>
<th>bit11</th>
<th>bit10</th>
<th>bit9</th>
<th>bit8</th>
<th>bit7</th>
<th>bit6</th>
<th>bit5</th>
<th>bit4</th>
<th>bit3</th>
<th>bit2</th>
<th>bit1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>CMOD</td>
<td>OTE1</td>
<td>OTE0</td>
<td>OTD1</td>
<td>OTD0</td>
<td>IOP1</td>
<td>IOP0</td>
<td>IOE1</td>
</tr>
</tbody>
</table>

OCP

| Setting compare value |

- Ø : Used bit
- X : Undefined bit
- △ : Setting the bit corresponding to output compare pin for use to "1"

(Output compare operation)

- Output compare compares the output compare register value and the 16-bit free run timer count value. The following happens when a match is detected.
  - This reverses the level in the output pin of output compare.
  - The output compare match flag of the output compare control status register is set to "1". (OCS:IOP=1)
  - When the output compare interrupt request is enabled (ICS:ICE=1), an interrupt request is generated.

(Output level setting and reverse timing)

- The output level of the output compare pin is set by the output level set bit in the output compare control status register (OCS:OTD).
- Timing of output reversing when a compare match occurs is synchronized with 16-bit free run timer count timing.
- 16-bit free run timer counter values are not compared when the output compare register is being written to.
Figure 8.7-2, Figure 8.7-3, and Figure 8.7-4 show operations of output compare.

**Figure 8.7-2 Output compare timing**

<table>
<thead>
<tr>
<th>φ</th>
<th>Counter value</th>
<th>N</th>
<th>N+1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Compare register value</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare match</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

φ: Machine clock

**Figure 8.7-3 The output level of the output compare output pin is reversed**

<table>
<thead>
<tr>
<th>Counter value</th>
<th>N</th>
<th>N+1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>compare register value</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>compare match signal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 8.7-4 Compare operation of output compare register during writing**

<table>
<thead>
<tr>
<th>Count value</th>
<th>N</th>
<th>N+1</th>
<th>N+2</th>
<th>N+3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compare register0 value</td>
<td>M</td>
<td>N+1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compare register1 value</td>
<td>M</td>
<td>N+3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compare register0 Write</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compare register1 Write</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Not generate match signal

Compare0 stop

Compare1 stop
Reversing two compare register outputs

The OUT (n + 1) pin output level can be reversed when either of output compare ch (n), and output compare ch (n + 1) are compared and matched due to the output level reverse mode selection bit setting (OCS:CMOD=1).

OCS:CMOD=1 is set, the output compare pin output level is as follows.

- The OUT (n) pin is reversed the output level when matched in an output compare ch (n)
- The OUT (n + 1) is reversed the output level when output compare ch (n) and output compare ch (n + 1) are compared and matched

(n=0, 2, 4, 6)

Figure 8.7-5 shows an example of an output waveform for OUT (n + 1) pin output level reversal factor when a match between output compare ch (n) and output compare ch (n + 1) are detected.

**Figure 8.7-5  Example of output compare output waveform**

![Output Compare Waveform Diagram](image-url)
8.8 Precaution on Using 16-bit I/O Timer

Pay attention to the following when using the 16-bit I/O timer.

- **Precaution on using 16-bit I/O timer**
  - **When output compare operation is enabled**
    Since output compare operation is synchronized with the 16-bit free run timer clock, output compare operations stop when the 16-bit free run timer count stops.
    To enable output compare operation (OCS:CST=1), set (TCCS:STOP=0) after enabling 16-bit free run timer operation.
  - **Precautions when setting 16-bit free-run timer**
    - Do not change the count clock selection bit (TCCS:CLK1, CLK0) during 16-bit free run timer operation (TCCS:STOP=0).
    - The 16-bit free run timer count value is initialized to "0000H" by a reset.
    - To set up by directly writing counter values to the timer data register (TCDT), stop the 16-bit free run timer (TCCS:STOP=1).
    - The word command must be used to write to TCDT.
  - **Operation delay caused by synchronization**
    Operating time delays occur because the input capture and output compare synchronize with the operation clock. After detecting the trigger signal from the input capture pin, retrieval is performed in synchronization with the machine clock while output compare performs compare operations synchronized with the free run timer count clock.
  - **Delay in match detection due to write operations to the compare register in output compare**
    Detection of matches of 16-bit free run timer count values is invalid during output compare register write operations.
    Read the free run timer count value to check or clear the free run timer to "0000H" beforehand.
8.9 Program Example of 16-bit Input/Output Timer

This section gives a program example of the 16-bit input/output timer.

Program Example of 16-bit Input/Output Timer

- **Processing specification**
  - The cycle of a signal input to the IN00 pin is measured.
  - The 16-bit free-run timer 0 and input capture 0 are used.
  - Trigger detection is set to rising edge.
  - The machine clock (\(\phi\)) is 16 MHz and \(4/\phi\) (0.25 µs) is selected for the free run timer count clock.
  - The timer overflow interrupt and input capture 0 capture interrupt are used for interrupt.
  - Counts the free run timer overflow interrupts which are used in cycle calculations.
  - The cycle can be determined from the following equation:

    \[
    \text{Cycle} = (\text{overflow count} \times 10000 \text{ H} + \text{nth IPCP0 value} - (\text{N-1})\text{th IPCP0 value}) \times \text{count clock cycle}
    \]

    \[
    = (\text{overflow count} \times 10000 \text{ H} + \text{nth IPCP0 value} - (\text{N-1})\text{th IPCP0 value}) \times 0.25 \mu\text{s}
    \]

- **Coding example**

  ```assembly
  ICR01 EQU 0000B1H ;Interrupt control register
  ICR10 EQU 0000BAH ;Interrupt control register
  DDR2 EQU 000012H ;Port 2 direction register
  TCCS EQU 000058H ;Timer control status register
  TCDT EQU 000056H ;Timer data register
  ICS01 EQU 000054H ;Input capture control
  ;Status Register
  IPCP0 EQU 000050H ;Input capture register 0
  IVF0 EQU TCCS:6 ;Timer overflow generation flag bit
  ICP0 EQU ICS01:6 ;Valid edge detection flag bit
  DATA DSEG ABS=00H
  ORG 0100H
  OV_CNT RW 1H
  DATA ENDS ;Overflow counting counter
  
  ;----------Main Program---------------------------------------------
  CODE CSEG
  START:
  ;Stack pointer (SP)
  ;Already initialized
  AND CCR,#0BFH ;Interruption is disabled
  MOV I:ICR01,#00H ;Interrupt levels 0 (strength)
  MOV I:ICR10,#00H ;Interrupt levels 0 (strength)
  MOV I:DDR2,#00000000B ;Sets the Port 2 direction
  MOV I:TCCS,#00100100B ;Enables counting, clears counting
  ;Overflow, Interruption is enabled,
  ;Count clock 4/f selection, Counter clear
  MOV I:ICS01,#00010001B ;IN00 or IN01 pin selection, external trigger
  ;IPCP0 rising edge
  ```
MOV ILM,#07H ;Sets ILM in PS to level 7
OR  CCR,#40H ;Interruption is enabled

LOOP:
  •
  User processing
  •
  BRA LOOP

;---------Interrupt Program-----------------------------------------------

WAR10:
  CLRB I:ICP0 ;Clears the valid edge detection flag
  •
  ;Saves the OV-CNT and the input capture value
  User processing
  •
  MOV A,0 ;Overflow for measuring the next cycle
  MOV OV_CNT,A ;Clears the overflow counting counter
  RETI ;Returning from interrupt processing

WAR11:
  CLRB I:IVF0 ;Clear timer overflow generating flag
  INC OV_CNT ;Increments the overflow counter
  •
  User processing
  •
  RETI ;Returning from interrupt processing

CODE ENDS

;---------Vector Settings---------------------------------------------------

VECT  CSEG ABS=0FFH
ORG 00FF80H ;vector set in interrupt number #31 (1FH)
  ;(Input capture)
DSL WARI0
ORG 00FFC4H ;vector set in interrupt number #14 (0EH)
  ;(Overflow)
DSL WARI1
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode

VECT ENDS
END START
Programming example of the output compare

- **Processing specification**
  - Uses the 16-bit free-run timer 0 and the output compare 0.
  - Set the output compare value to "5555H" to use the OUT0 pin and the OCP0 register.
  - A compare match reverses the output from the pin and generates an interrupt at the same time.
  - Set the machine clock (φ) to 16 MHz and select 4/φ (0.25 µs) as the count clock of the 16-bit free-run timer.

- **Coding example**

```
ICR11 EQU 0000BBH ;Interrupt control register
TCCS EQU 000058H ;Timer control status register
TCDT EQU 000056H ;Timer data register
OCS0 EQU 000062H ;Output compare control status register
OCS1 EQU 000063H ;Output compare control status register
OCP0 EQU 00005AH ;Output compare register
IOP0 EQU OCS0:6 ;Compare match flag bit

;--------Main Program-----------------------------------------------
CODE   CSEG
START:
; ;Stack pointer (SP)
;Already initialized
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR11,#00H ;Interrupt levels 0 (strength)
MOV I:TCCS,#00000100B ;Enables counting, clears counting
;Overflow, Interruption is disabled
;Sets the pin to the "H" level.
;Count clock 4/φ selection
MOVW I:OCP0,#5555H ;Sets the compare register
MOV I:OCS0,#00010001B ;Clears the compare match flag,
;Output compare 0 enables output, enables operation
MOV I:OCS1,#00000100B ;Output compare 0 enables output,
;Sets the pin output to "L"
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled

LOOP:
• User processing
•
BRA LOOP

;--------Interrupt Program-----------------------------------------
WARI:
CLRB I:IOP0 ;Clears the compare match flag
•
User processing
•
RETI ;Returning from interrupt processing

CODE   ENDS
```
;------------------Vector Settings---------------------------------------------
VECT    CSEG ABS=0FFH
ORG 00FF78H ;vector set in interrupt number #33 (21H)
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START
This chapter explains the functions and operations of the 16-bit reload timer.

9.1 Overview of 16-bit Reload Timer
9.2 Block Diagram of 16-bit Reload Timer
9.3 Configuration of 16-bit Reload Timer
9.4 Interrupts of 16-bit Reload Timer
9.5 Explanation of Operation of 16-bit Reload Timer
9.6 Precautions when Using 16-bit Reload Timer
9.7 Program Example of 16-bit Reload Timer
9.1 Overview of 16-bit Reload Timer

The 16-bit reload timer has the following functions:
- The count clock can be selected from three internal clocks and external event clocks.
- Activation trigger selects software trigger or external trigger.
- Interruption can be generated when underflow of the 16-bit timer register is generated. The 16-bit reload timer can be used as an interval timer by using an interrupt.
- When underflow of the 16-bit timer register (TMR) is generated, the one-shot mode (which stops the counting of the TMR register) or reload mode (which continues the counting of the TMR register by reloading the 16-bit reload register value to the TMR register) can be selected.
- While the level of the set gate input inputs to the TIN pin, the external gate input operation mode which performs the count operation of 16-bit timer register (TMR) can be selected.
- The 16-bit reload timer corresponds to the E1²OS.
- MB90520A series has two internal channels of 16-bit reload timers.

### Operation Modes of 16-bit Reload Timer

Table 9.1-1 indicates the operation modes of the 16-bit reload timer.

**Table 9.1-1 Operation Modes of 16-bit Reload Timer**

<table>
<thead>
<tr>
<th>Count Clock</th>
<th>Start Trigger</th>
<th>Operation at Underflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal clock</td>
<td>Software trigger</td>
<td>One-shot mode</td>
</tr>
<tr>
<td></td>
<td>External trigger</td>
<td>Reload mode</td>
</tr>
<tr>
<td>Event count</td>
<td>Software trigger</td>
<td>One-shot mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reload mode</td>
</tr>
</tbody>
</table>

#### Internal clock mode

- 16-bit reload timer can be set to internal clock mode by setting the count clock selection bits of the timer control status register (TMCSR:CSL1 and CSL0) to "00B", "01B" or "10B".
- In the internal clock mode, the 16-bit reload timer count down in synchronization with the internal clock.
  - The count clock cycle can be selected from 3 types for the internal clock using the count clock selection bits (TMCSR:CSL1 and CSL0).
- Set the software trigger or edge detection of the external trigger for the activation trigger.

#### Event count mode

- When the count clock selection bits in the timer control status register (TMCSR: CSL1 and CSL0) are set to "11B", the 16-bit reload timer is set to the event count mode.
- Under event count mode, count down is performed in synchronization with the edge detection of the external event clock that has been input to the TIN pin.
• A software trigger is selected as the activation trigger.
• The 16-bit reload timer can be used as an interval timer by using a fixed cycle of the external clock.

## Operation at Underflow

- When an activation trigger is input, the value set to the 16-bit reload register will be reloaded to the 16-bit timer register, and count down starts in synchronization with the count clock. Underflow is generated when the 16-bit timer register is counted down from "0000H" to "FFFFH".
- Underflow interrupt request is generated to the CPU when underflow is generated while underflow interruption is enabled (TMCSR:INTE = 1).
- Operation of the 16-bit reload timer can be set using the reload selection bit of the timer control status register (TMCSR:RELD) when underflow is generated.

(One-shot mode (TMCSR: RELD = 0))

Counting of the 16-bit timer register (TMR) is stopped when underflow is generated. When the following activation trigger is input, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the TMR register, and counting of the TMR register starts.
- "H" or "L" level rectangular waves can be output from the TOT pin during counting of the 16-bit timer register under one-shot mode.
- The initial level ("H" or "L") of the rectangular wave to be output can be set using the TOT pin output level selection bit of the timer control status register (TMCSR:OUTL).

(Reload mode (TMCSR: RELD = 1))

When underflow is generated, the value that has been set to the 16-bit reload register is reload to the 16-bit timer register (TMR), and counting of the TMR register will continue.
- Under reload mode, the toggle wave that reverses the output level every time TMR underflow is generated is output from the TOT pin during counting of the 16-bit timer register (TMR).
- The level ("H" or "L") of the toggle wave to be output is set by the TOT pin output level selection bit of the timer control status register (TMCSR:OUTL).
- This can be used as the interval timer using underflow interruption.

### Table 9.1-2 Interval Time of 16-bit Reload Timer

<table>
<thead>
<tr>
<th>Count Clock</th>
<th>Count Clock Cycle</th>
<th>Interval Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal clock</td>
<td>$2^1T$ (0.125μs)</td>
<td>0.125μs to 8.192ms</td>
</tr>
<tr>
<td></td>
<td>$2^3T$ (0.5μs)</td>
<td>0.5μs to 32.768ms</td>
</tr>
<tr>
<td></td>
<td>$2^5T$ (2.0μs)</td>
<td>2.0μs to 131.1ms</td>
</tr>
<tr>
<td>Event count</td>
<td>$2^3T$ or more</td>
<td>0.5 μs or more</td>
</tr>
</tbody>
</table>

T: Machine cycle

Interval time example and description in the () indicates a calculation example when 16 MHz is selected as the machine clock frequency.

### Reference:
- The 16-bit reload timer 0 can be used as the clock input source for the UART.
- The 16-bit reload timer 1 can be used as the activation trigger of the A/D converter.
9.2 Block Diagram of 16-bit Reload Timer

A block diagram for the 16-bit reload timers 0 and 1 is shown as follows.

Block Diagram of 16-bit Reload Timer

Figure 9.2-1 Block Diagram of 16-bit Reload Timer

[Diagram of 16-bit Reload Timer block diagram]
● Pin, output to built-in peripheral functions, interrupt request number, within the block diagram

The 16-bit reload timer is equipped with 2 internal channels. The pin names per channel are as follows.

16-bit reload timer 0:
- TIN pin: P70/TIN0
- TOT pin: P71/TOT0
- Output to built-in peripheral function: UART clock input source
- Interrupt request number: #38 (26H)

16-bit reload timer 1:
- TIN pin: P72/TIN1
- TOT pin: P73/TOT1
- Output to built-in peripheral function: A/D converter activation trigger
- Interrupt request number: #40 (28H)

● Count clock generator circuit

The count clock to be supplied to the 16-bit timer register (TMR) is generated based on the machine clock or external event clock.

● Reload controller circuit

When the 16-bit reload timer starts counting or an underflow of the 16-bit timer register (TMR) is generated, the value that has been set to the 16-bit reload register is loaded to the TMR.

● Output controller circuit

Reverse output of the TOT pin and enabling or disabling of the TOT pin output can be set by the generation of underflow.

● Operation controller circuit

The operation controller circuit starts or stops the 16-bit reload timer.

● 16-bit timer register (TMR)

The 16-bit timer register (TMR) is a 16-bit down counter. When read is performed, the counter value can be read.

● 16-bit reload register (TMRLR)

Interval time of 16-bit reload timer can be set. When the 16-bit reload timer starts counting or underflow of the 16-bit timer register (TMR) is generated, the value that has been set to the 16-bit reload register is loaded to the TMR register.

● Timer control status register (TMCSR)

The 16-bit reload timer operation mode, operation condition, activation trigger, initiation using the software trigger, reload operation mode, enabling/disabling interrupt requests, TOT pin output level, and the TOT output pin can all be set up.
9.3 Configuration of 16-bit Reload Timer

Pins of 16-bit Reload Timer, interrupt factor and detail of register are described.

■ Pins of 16-bit Reload Timer

The pins of the 16-bit reload timer serve as general-purpose I/O ports. Table 9.3-1 shows the pin functions and the pin settings required to use the 16-bit reload timer.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Set up required for use of 16-bit reload timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>P70/TIN0</td>
<td>General-purpose I/O port, 16-bit reload timer 0 input</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P71/TOT0</td>
<td>General-purpose I/O port, 16-bit reload timer 0 output</td>
<td>Set timer output enable (TMCSR0:OUTE = 1).</td>
</tr>
<tr>
<td>P72/TIN1</td>
<td>General-purpose I/O port, 16-bit reload timer 1 input</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P73/TOT1</td>
<td>General-purpose I/O port, 16-bit reload timer 1 output</td>
<td>Set timer output enable (TMCSR1:OUTE = 1).</td>
</tr>
</tbody>
</table>

■ Block Diagram for Pins of 16-bit Reload Timer

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

■ Generation of Interrupt Request from 16-bit Reload Timer

When the 16-bit reload timer is activated, and if the count value of the 16-bit timer register is counted down from "0000H" to "FFFFH", underflow is generated. When underflow is generated, set "1" to the underflow generation flag bit of the timer control status register (TMCSR:UF = 1). When underflow interrupt output is enabled (TMCSR:INTE = 1), interrupt request is generated to the CPU.
9.3.1 Timer Control Status Registers (High)  
(TMCSR0: H, TMCSR1: H)

The timer control status register upper (TMCSR0:H and TMCSR1:H) bits and the bit 7 of timer control status register lower (TMCSR0:L and TMCSR1:L) can set up the operation mode and count clock. Bit 7 of the timer control status register upper and timer control status register lower

**Figure 9.3-1 Bit7 of the timer control status register upper and timer control status register lower**

<table>
<thead>
<tr>
<th>MOD2</th>
<th>MOD1</th>
<th>MOD0</th>
<th>MOD2</th>
<th>MOD1</th>
<th>MOD0</th>
<th>MOD2</th>
<th>MOD1</th>
<th>MOD0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

- **Operation mode select bit** (Internal clock mode: CSL1 and CSL0 = "01B" or "10B")
  - `0 0 0`: Trigger inhibit
  - `0 0 1`: Rising edge
  - `0 1 0`: Falling edge
  - `0 1 1`: both edges
  - `1 X 0`: "L" level
  - `1 X 1`: "H" level

- **Operation mode select bit** (Event count mode: CSL1 and CSL0 = "11B")
  - `X 0 0`: Valid edge, Level
  - `X 0 1`: Rising edge
  - `X 1 0`: Falling edge
  - `X 1 1`: both edges

- **Count clock selection bit**
  - `0 0`: Internal clock mode
  - `0 1`: 2^T
  - `1 0`: 2^T
  - `1 1`: External event clock

- **Count clock**
  - `2^T`
  - `2^T`

T: Machine clock frequency

---

R/W: Readable and Writable
X: Undefined
—: Unused
: Reset value

---

325
### Table 9.3-2 Bit7 function for the timer control status register upper and timer control status register lower

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit9 to bit7</td>
<td>These bits set the operation conditions of the 16-bit reload timer. (For internal clock mode (except when CSL1 and CSL0 are &quot;11B&quot;)) The MOD2 bit is used to select the function of the input pin.</td>
</tr>
</tbody>
</table>
|                   | **When MOD2 bit is set to "0":**  
|                   | • The TIN pin functions as a trigger input.  
|                   | • The edge to be detected is selected using these MOD1 and MOD0 bits.  
|                   | • When the edge is detected, the value set in the 16-bit reload register is reloaded in the 16-bit timer register (TMR), starting the count operation of the TMR.  |
|                   | **When MOD2 bit is set to "1":**  
|                   | • The TIN pin functions as a gate input.  
|                   | • The MOD1 bit is not used.  
|                   | • Signal level ("H" or "L") to be detected is selected using this MOD0 bit. 16-bit timer register is counted only while a signal level is input (For event count mode (CSL1 and CSL0 are "11B"))  |
|                   | **When MOD2 bit is not used:**  
|                   | • The MOD2 bit is not used.  
|                   | • An external event clock is input from the TIN pin.  
|                   | • The edge to be detected is selected using these MOD1 and MOD0 bits.  |
| bit11 to bit10    | These bits select the count clock of the 16-bit reload timer.  
|                   | **When other than "11B":** The internal clock is counted (internal clock mode).  
|                   | **When set to "11B":** The edge of the external event clock is counted (event count mode).  |
| bit12 to bit15    | **Read:** The value is undefined.  
|                   | **Write:** No effect  |
9.3.2 Timer Control Status Registers (Low) (TMCSR0: L, TMCSR1: L)

The bit 7 of the timer control status registers lower (TMCSR0: L, TMCSR1: L) can set enabling/disabling the timer operation, enabling the interrupt, and the pin output level.

- **Timer Control Status Registers (Low) (TMCSR0: L, TMCSR1: L)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Software trigger bit</td>
<td>No effect</td>
</tr>
<tr>
<td>6</td>
<td>Timer operation enable bit</td>
<td>Timer operation disabled</td>
</tr>
<tr>
<td>5</td>
<td>Underflow generation flag bit</td>
<td>No underflow</td>
</tr>
<tr>
<td>4</td>
<td>Underflow interrupt enable bit</td>
<td>Underflow interrupt disabled</td>
</tr>
<tr>
<td>3</td>
<td>Reload select bit</td>
<td>One-shot mode</td>
</tr>
<tr>
<td>2</td>
<td>TOT Pin output level select bit</td>
<td>The rectangular waveform &quot;H&quot; output at counting</td>
</tr>
<tr>
<td>1</td>
<td>TOT Pin output enable bit</td>
<td>Register and pin for channels</td>
</tr>
<tr>
<td></td>
<td>TMCSR0</td>
<td>general - purpose I/O port</td>
</tr>
<tr>
<td></td>
<td>TMCSR1</td>
<td>general - purpose I/O port</td>
</tr>
</tbody>
</table>

- **Reset value**: 00000000B
- **R/W**: Readable and Writable
- **: Refer to *Timer control status register upper* for bit 7 (MOD0).
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0 TRG: Software trigger bit</td>
<td>16-bit reload timer can be activated by the software. This TRG bit only functions when timer operation is enabled (CNTE = 1). When the bit is set to &quot;0&quot;: No effect. When the bit is set to &quot;1&quot;: Reloads value set in 16-bit reload register (TMRLR) to 16-bit timer register (TMR), starting TMR count operation When reading: &quot;0&quot; is always read.</td>
</tr>
<tr>
<td>bit1 CNTE: Timer operation enable bit</td>
<td>Operation of the 16-bit reload timer can be enabled or disabled. When the bit is set to &quot;0&quot;: Stops count operation. When the bit is set to &quot;1&quot;: 16-bit reload timer enters activation trigger wait state. When the activation trigger is input, the 16-bit timer register starts counting</td>
</tr>
<tr>
<td>bit2 UF: Underflow generation flag bit</td>
<td>This bit indicates that the TMR underflow. When the bit is set to &quot;0&quot;: This UF bit will be cleared. When the bit is set to &quot;1&quot;: No effect. &quot;1&quot; is read under the read-modify-write command.</td>
</tr>
<tr>
<td>bit3 INTE: Underflow interrupt enable bit</td>
<td>Underflow interruption can be enabled or disabled. When underflow is generated (TMCSR:UF = 1) while underflow interruption is enabled (TMCSR:INTE = 1), an interrupt request is generated.</td>
</tr>
<tr>
<td>bit4 RELD: Reload selection bit</td>
<td>Reload operation is set when underflow is generated. When the bit is set to &quot;0&quot;: When underflow is generated, counting is suspended. (one-shot mode) When the bit is set to &quot;1&quot;: When underflow is generated, the value that has been set to the 16-bit reload register is loaded to the 16-bit timer register, and counting will continue.</td>
</tr>
<tr>
<td>bit5 OUTL: TOT Pin output level selection bit</td>
<td>This bit sets the output level of the output pin of the 16-bit reload timer. &lt;One-shot mode (RELD = 0)&gt; When the bit is set to &quot;0&quot;: A rectangular wave of &quot;H&quot; is output while the 16-bit timer is counting. When the bit is set to &quot;1&quot;: A rectangular wave of &quot;L&quot; is output while the 16-bit timer is counting. &lt;Reload mode (RELD = 1)&gt; When the bit is set to &quot;0&quot;: When the 16-bit reload timer is activated, &quot;L&quot; level is output, or when underflow is generated, reversing toggle is output. When the bit is set to &quot;1&quot;: When the 16-bit reload timer is activated, &quot;H&quot; level is output, or when underflow is generated, reversing toggle is output.</td>
</tr>
<tr>
<td>bit6 OUTE: TOT Output enable bit</td>
<td>The TOT pin function of the 16-bit reload timer is set. When the bit is set to &quot;0&quot;: Functions as general-purpose I/O port When the bit is set to &quot;1&quot;: Functions TOT as pin of 16-bit reload timer</td>
</tr>
</tbody>
</table>
9.3.3 16-bit Timer Registers (TMR0, TMR1)

The 16-bit timer registers (TMR0, TMR1) are 16-bit down counters. When read is performed, the value being counted can be read.

16-bit Timer Registers (TMR0, TMR1)

While timer operation is enabled (TMCSR:CNTE = 1), when an activation trigger is input, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the 16-bit timer register (TMR), and counting of the TMR register starts.

When timer operation is disabled (TMCSR:CNTE = 0), the TMR register value is retained. Underflow is generated when the TMR register value is counted down from "0000H" to "FFFFH" while the TMR register is counting.

(Reload mode)
When underflow of the 16-bit timer register (TMR) is generated, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the TMR register, and counting the TMR register re-starts.

(One-shot mode)
When underflow of the 16-bit timer register (TMR) is generated, counting of the TMR register is stopped and activation trigger input wait state is adopted. The TMR register value will be retained as "FFFFH".

Notes:
- The word command (MOVW) must be used to read the 16-bit timer register (TMR).
- The TMR and the TMRLR are assigned to the same address. When writing was performed, the set value can be written to the TMRLR without affecting the TMR register. When reading is performed, the TMR register value can be read.
CHAPTER 9  16-BIT RELOAD TIMER

9.3.4  16-bit Reload Registers (TMRLR0, TMRLR1)

Set the reloaded value to the 16-bit timer register for the 16-bit reload register. The value that has been set to the 16-bit reload register is loaded to the 16-bit timer register when the activation trigger is input, and counting the 16-bit timer register starts.

■ 16-bit Reload Registers (TMRLR0, TMRLR1)

When the 16-bit reload register is set, disable the timer operation (TMCSR:CNTE = 0). Start up the timer after the 16-bit reload register is set (TMCSR:CNTE = 1).

When the activation trigger is input, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the 16-bit timer register (TMR), and counting of the TMR register starts.

| W : Read only | X : Undefined |

Notes:
- Disable the 16-bit reload timer operation (TMCSR:CNTE = 0) to write to the 16-bit timer register. Always use the word command (MOVW).
- The TMRLR and the TMR are assigned to the same address. When writing is performed, the set value can be written to the TMRLR register without affecting the TMR register. When reading is performed, the TMR register value can be read.
- Read-modify-write (RMW) commands, such as the INC/DEC command, cannot be used.
CHAPTER 9  16-BIT RELOAD TIMER

9.4  Intermits of 16-bit Reload Timer

The 16-bit reload timer generates interrupt requests when underflow of the 16-bit timer register is generated.

■ Interrupts of 16-bit Reload Timer

Underflow is generated when the TMR value is counted down from "0000H" to "FFFFH" while the 16-bit timer register (TMR) is counting. The underflow generation flag bit of the timer control status register (TMCSR:UF) is set to "1" when underflow is generated. When underflow interrupt is enabled (TMCSR:INTE = 1), interrupt request is output to the CPU.

Table 9.4-1  Interrupt Control Bits and Interrupt Factors of 16-bit Reload Timer

<table>
<thead>
<tr>
<th></th>
<th>16-bit reload timer 0</th>
<th>16-bit reload timer 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Underflow generation flag bit</td>
<td>TMCSR0: UF</td>
<td>TMCSR1: UF</td>
</tr>
<tr>
<td>Interrupt enable bit</td>
<td>TMCSR0: INTE</td>
<td>TMCSR1: INTE</td>
</tr>
<tr>
<td>Interrupt factor</td>
<td>Underflow in TMR0</td>
<td>Underflow in TMR1</td>
</tr>
</tbody>
</table>

■ Interrupts of 16-bit Reload Timer and EI²OS

Reference:  Refer to "3.5  Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

■ EI²OS Function of 16-bit Reload Timer

The 16-bit reload timer supports the EI²OS. The EI²OS can be activated by generating underflow of the 16-bit timer register.

EI²OS can be used only when peripheral functions that share the interrupt control register (ICR) do not use interruption. When the EI²OS is used by the 16-bit reload timers 0 and 1, disable interrupt request generation of the peripheral functions that share the 16-bit reload timers 0 and 1, and the interrupt control register (ICR).
### 9.5 Explanation of Operation of 16-bit Reload Timer

Setting of 16-bit Reload Timer and operating status are explained.

#### Setting of 16-bit Reload Timer

- **Setting of internal clock mode**

  The set bit for counting the internal clock is shown in Figure 9.5-1.

  ![Figure 9.5-1 Setting of internal clock mode](image)

  - TMCSR
    - bit15 14 13 12 11 10 9 bit8 bit7 6 5 4 3 2 1 bit0
    - CSL1 CSL0 MOD2 MOD1 MOD0 OUTE OUTL RELD INTE UF CNT E TRG
    - other than "11B"
    - TMRLR
      - Set the reloaded value to the 16-bit timer register
      - Used bit
      - 1: Set to "1"

- **Setting of event count mode**

  The set bit for operating through input of the external event is shown in Figure 9.5-2.

  ![Figure 9.5-2 Setting of Event Count Mode](image)

  - TMCSR
    - bit15 14 13 12 11 10 9 bit8 bit7 6 5 4 3 2 1 bit0
    - CSL1 CSL0 MOD2 MOD1 MOD0 OUTE OUTL RELD INTE UF CNT E TRG
    - 1 1
    - TMRLR
      - Set the reloaded value to the 16-bit timer register
      - Used bit
      - 1: Set to "1"

Set "0" to the DDR (Port Direction Register) bit corresponding for pin using as TIN pin.
### Operating State of 16-bit Timer Register

Operation status of the 16-bit timer register is decided by the timer operation enabling bit of the timer control status register (TMCSR:CNTE), and the WAIT signal. The timer control state include the stop state, start trigger input wait state (WAIT state), and RUN state.

Figure 9.5-3 shows the state transition diagram for the 16-bit timer registers.

**Figure 9.5-3 State Transition Diagram**

- **STOP status**: CNTE=0, WAIT=1
  - TIN pin: Input disabled
  - TOT pin: General-purpose I/O port
  - 16-bit timer register: Hold the value at stop
  - The value immediately after reset is undefined.

- **WAIT status**: CNTE=1, WAIT=1
  - TIN pin: valid only Trigger input
  - TOT pin: output 16-bit reload register
  - 16-bit timer register: Hold the value at stop
  - The value immediately after reset is undefined until loading

- **LOAD**: CNTE=1, WAIT=0
  - Load 16-bit reload register setting value to 16-bit timer register

- **RUN status**: CNTE=1, WAIT=0
  - TIN pin: function as input pin of 16-bit reload timer
  - TOT pin: function as output pin of 16-bit reload timer
  - 16-bit timer register: Count operation

- **Reset**: CNTE=0

**Legend**:
- **: State transition by Hardware**
- **: State transition by Register access**
- **WAIT**: WAIT signal (Internal signal)
- **TRG**: Software trigger bit (TMCSR)
- **CNTE**: Timer operation enable bit (TMCSR)
- **UF**: Underflow generation flag bit (TMCSR)
- **RELD**: Reload select bit (TMCSR)
9.5.1 Operation in Internal Clock Mode

Under internal clock mode, the operation mode can be selected from three types by setting the operation mode selection bits of the timer control status register (TMCSR:MOD2 to MOD0). The rectangular wave or toggle wave can be output from the TOT pin by setting the operating mode and reload mode.

■ Setting of internal clock mode

- The 16-bit reload timer is set to the internal clock mode by setting the count clock selection bits of the timer control status register (TMCSR:CSL1 and CSL0) to "00B", "01B" or "10B".
- When internal clock mode is set, the 16-bit timer register counts down in synchronization with the internal count clock.
- The internal count clock can be selected from 3 types by setting the count clock selection bits of the timer control status register (TMCSR:CSL1 and CSL0).

(Setting a reload value to 16-bit timer register)

After the 16-bit reload timer is activated, if a value is set to the 16-bit reload register (TMRLR), it is loaded to the 16-bit timer register (TMR) by underflow.

1. Disables the operation of the 16-bit reload timer (TMCSR: CNTE = 0).
2. Set the load value to the 16-bit timer register for the 16-bit reload register.
3. Enables the operation of the 16-bit reload timer (TMCSR: CNTE = 1).

Note:
It takes 1 machine cycle (abbreviated as "T") for the value that has been set in the 16-bit reload register (TMRLR) to be loaded to the 16-bit timer register (TMR) after an activation trigger is input.
Operation as 16-bit Timer Register Underflow

Underflow is generated when the TMR register value is counted down from "0000H" to "FFFFH" while the 16-bit timer register (TMR) is counting.

- When underflow is generated, the underflow generation flag bit (TMCSR:UF) of the timer control status register is set to "1".
- When the underflow interrupt enabling bit of the timer control status register (TMCSR:INTE) is set to "1", underflow interrupts are generated to the CPU.
- Reload operation when underflow is generated can be set by the reload selection bit of the timer control status register (TMCSR:RELD).

(One-shot mode (TMCSR: RELD = 0))

When underflow is generated, the 16-bit timer register (TMR) is stopped, and standby status will be adopted for the activation trigger input wait state. When the following activation trigger is input, counting of the TMR register re-starts.

Under the one-shot mode, rectangular waves can be output from the TOT pin while counting the TMR register. The rectangular wave level ("H" or "L") can be set by the TOT pin output level selection bit of the timer control status register (TMCSR:OUTL).

(Reload mode (TMCSR: RELD = 1))

When underflow is generated, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the 16-bit timer register (TMR), and counting of the TMR register continues.

Under reload mode, whenever underflow is generated while counting the TMR register, the toggle wave that reverses the output level can be output from the TOT pin. The toggle wave level ("H" or "L") when starting up the reload timer can be set by the TOT pin output level selection bit of the timer control status register (TMCSR:OUTL).

Operation in Internal Clock Mode

Under internal clock mode, the operation mode can be set by the operation mode selection bits of the timer control status register (TMCSR:MOD2 to MOD0).

When operation mode is set, stop the timer by setting the timer operation enabling bit of the timer control status register (TMCSR:CNTE) to "0".

(Software trigger mode (MOD2 to MOD0 =000B))

When the software trigger mode is set, the software trigger bit of the timer control status register (TMCSR:TRG) is set to "1", and the 16-bit reload timer is activated. When the 16-bit reload timer is activated, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the 16-bit timer register (TMR), and counting of the TMR register starts.

Note: When the timer operation enabling bit (TMCSR:CNTE) and software trigger bit (TMCSR:TRG) of the timer control status register are simultaneously set to "1", counting of the 16-bit timer register (TMR) starts at the same time as the 16-bit reload timer is activated.
Figure 9.5-4 Count Operation in Software Trigger Mode (One-shot Mode)

- **Count clock**: The clock signal is illustrated as a series of pulses.
- **Counter (TMR)**: The counter value is shown transitioning from 0000 to FFFF, back to 0000 again.
- **Data load signal**: This signal is active once, indicating the data load.
- **UF bit**: This bit remains constant, indicating no specific state change.
- **CNTE bit**: This bit also remains constant, indicating no specific state change.
- **TRG bit**: This bit changes state once, indicating the trigger event.
- **TOT pin**: The TOT pin remains in a wait state until the trigger event.

*T*: Machine cycle

* : It takes 1T time from trigger input to loading the data of reload register.

---

Figure 9.5-5 Count Operation in Software Trigger Mode (Reload Mode)

- **Count clock**: The clock signal is illustrated as a series of pulses.
- **Counter (TMR)**: The counter value is shown transitioning from 0000 to FFFF, back to 0000 again.
- **Data load signal**: This signal is active once, indicating the data load.
- **UF bit**: This bit remains constant, indicating no specific state change.
- **CNTE bit**: This bit also remains constant, indicating no specific state change.
- **TRG bit**: This bit changes state once, indicating the trigger event.
- **TOT pin**: The TOT pin remains in a wait state until the trigger event.

*T*: Machine cycle

* : It takes 1T time from trigger input to loading the data of reload register.
While the external trigger mode is set, the valid edge is input to the TIN pin from outside and the 16-bit reload timer can be activated. When the 16-bit reload timer is activated, the value that has been set to the 16-bit reload register (TMRLR) is loaded to the 16-bit timer register (TMR), and counting of the TMR starts.

- The clock edge to be detected can be selected from the rising edge, falling edge, and both edges by setting the operation mode selection bits of the timer control status register (TMCSR:MOD2 to MOD0).

**Figure 9.5-6  Count Operation in External Trigger Mode (One-shot Mode)**

*Note:* The trigger pulse width of the edge to be input to the TIN pin should be 2 machine cycles (abbreviated as "T") or more.

**Figure 9.5-7  Count Operation in External Trigger Mode (Reload Mode)**

*Note:* It takes 2T to 2.5T time from external trigger input to loading the data of reload register.
(External gate input mode (MOD2 to MOD0 = "1X0B", "1X1B"))

When the external gate input mode is set, start the 16-bit reload timer by setting the software trigger bit in the timer control status register (TMCSR: TRG) to "1". When the 16-bit reload timer is activated, the value that has been set to the 16-bit reload timer (TMRLR) is loaded to the 16-bit timer register (TMR).

- The gate input level ("H" or "L") can be set by setting the operation mode selection bits of the timer control status register (TMCSR:MOD2 to MOD0).
- Counting of the 16-bit timer register (TMR) is performed while the set gate input level is input to the TIN pin after starting up the 16-bit reload timer.

Figure 9.5-8  Count Operation in External Gate Input Mode (One-shot Mode)

* : It takes 1T time from trigger input to loading the data of reload register.

Figure 9.5-9  Count Operation in External Gate Input Mode (Reload Mode)

* : It takes 1T time from trigger input to loading the data of reload register.
### 9.5.2 Operation in Event Count Mode

In the event count mode, after the 16-bit reload timer is started, the edge of the signal input to the TIN pin is detected to perform the count operation of the 16-bit timer register (TMR). The rectangular wave or toggle wave can be output from the TOT pin by setting the operating mode and reload mode.

#### Setting of Event Count Mode

- The 16-bit reload timer will be set to the event count mode by setting the count clock selection bits of the timer control status register (TMCSR: CSL1 and CSL0) to "11B".
- In the event count mode, the TMR count down in synchronization with the edge detection of the external event clock input to the TIN pin.

**Setting initial value of counter**

After the 16-bit reload timer is activated, the value set to the 16-bit reload register (TMRLR) is reloaded to the 16-bit timer register (TMR).

1. Disables the operation of the 16-bit reload timer (TMCSR: CNTE = 0).
2. Sets a reload value to the TMR in the TMRLR.
3. Enables the operation of the 16-bit reload timer (TMCSR: CNTE = 1).

#### Note:

It takes 1 machine cycle (abbreviated as "T") to load the value set in the TMRLR to the TMR after the start trigger is input.
■ Operation as 16-bit Timer Register Underflow

When the 16-bit timer register TMR register value is counted down from "0000H" to "FFFFH" while counting the TMR, underflow is generated.

- When underflow is generated, the underflow generation flag bit of the timer control status register (TMCSR:UF) is set to "1".
- Underflow interrupt is generated when the underflow interrupt enabling bit of the timer control status register (TMCSR:INTE) is set to "1".
- The reload operation when an underflow occurs is set by the reload selection bit in the timer control status register (TMCSR: RELD).

   (One-shot mode (TMCSR: RELD = 0))

When underflow is generated, the 16-bit timer register (TMR) is stopped, and standby status will be adopted for the activation trigger input wait state. TMR register counter operation resumes when the next startup trigger is entered.

Rectangular waves are output from the TOT pin during TMR register counter operation in one-shot mode. The rectangular wave level ("H" or "L") can be set by the TOT pin output level selection bit of the timer control status register (TMCSR:OUTL).

   (Reload mode (TMCSR: RELD = 1))

When an underflow occurs, a set value in the 16-bit reload register (TMRLR) in the 16-bit timer register (TMR) is loaded to continue the TMR register count operation.

In the reload mode, a toggle wave is output from the TOT pin to reverse output levels whenever an underflow occurs during TMR register count operation. The output level selection bit of the TOT pin in the timer control status register (TMCSR:OUTL) can be used to set the toggle wave level ("H" or "L") at reload timer startup.

■ Operation in Event Count Mode

The 16-bit reload timer is enabled when the timer enable bit of the timer control status register (TMCSR:CNTE) is set to "1". The 16-bit reload timer can be started up when the software trigger bit of the timer control status register (TMCSR:TRG) is set to "1".

When the 16-bit reload timer is activated, a set value in the 16-bit reload register (TMRLR) in the 16-bit timer register (TMR) is loaded to continue the TMR register count operation.

After startup of the 16-bit reload timer, the edge of the external event clock input to the TIN pin is detected to perform TMR register count operation.

- The rising edge, falling edge or both can be selected for detection by setting the operation mode selection bit of the timer control status register (TMCSR:MOD2 to MOD0).

**Note:** The level width of the external event clock input to the TIN pin should be at least 4 T (abbreviated as "T").
Figure 9.5-10 Count Operation in Event Count Mode (One-shot Mode)

TIN pin
Counter (TMR)
Data load signal
UF bit
CNTE bit
TRG bit
TOT pin

Start trigger input wait

T : Machine cycle
* : It takes 1T time from trigger input to loading the data of reload register.

Figure 9.5-11 Count Operation in Event Count Mode (Reload Mode)

Count clock
Counter (TMR)
Data load signal
UF bit
CNTE bit
TRG bit
TOT pin

T : Machine cycle
* : It takes 1T time from trigger input to loading the data of reload register.
9.6 Precautions when Using 16-bit Reload Timer

This section explains the precautions when using the 16-bit reload timer.

**Precautions when Using 16-bit Reload Timer**

- **Notes on using programs to set**
  
  Set the 16-bit reload register (TMRLR) after disabling the timer operation (TMCSR: CNTE = 0)
  
  - Although 16-bit timer register (TMR) read operations can be performed during TMR count operation, the word command must be used.
  
  - Change the CSL1 and CSL0 bits in the timer control status register TMCSR after disabling the timer operation (TMCSR: CNTE = 0)

- **Precautions on interrupt**
  
  - When the timer control status register (TMCSR) UF bit is set to "1" and underflow interrupt output is enabled (TMCSR: INTE=1), it is not possible to return from interrupt processing. UF bits must be cleared in the interrupt routine. However, when EI²OS is used, the UF bit is automatically cleared to "0".
  
  - When using the EI²OS in the 16-bit reload timer, it is necessary to disable generation of interrupt by resources that share the interrupt control register (ICR).
9.7 Program Example of 16-bit Reload Timer

This section gives a program example of the 16-bit reload timer operated in the internal clock mode and the event count mode.

- Program Example in Internal Clock Mode

  ● Processing specification
  - The 25-ms interval timer interrupt is generated by the 16-bit reload timer 0.
  - The repeated interrupts are generated in the reload mode.
  - The timer is started using the software trigger instead of the external trigger input.
  - E1^OS is not used.
  - Machine clock is set to 16 MHz and counter clock is set to 2 µs.

  ● Coding example

```assembly
ICR13   EQU 0000BDH ;16-bit reload timer, interrupt control register
TMCSR0  EQU 000048H ;Timer control status register
TMR0    EQU 00004AH ;16-bit timer register
TMRLR0  EQU 00004AH ;16-bit reload register
UF0     EQU TMCSR0:2 ;Underflow generation flag bit
CNTE0   EQU TMCSR0:1 ;Timer operation enable bit
TRG0    EQU TMCSR0:0 ;software trigger bit

;--------Main program--------------------------------------------
CODE    CSEG
 ;stack pointer (SP), etc. are to be initialized
AND      CCR,#0BFH ;Interruption is disabled
MOV I:ICR13,#00H ;Interrupt levels 0 (strength)
CLRB I:CNTE0 ;counter paused
MOVW I:TMRLR0,#30D3H ;25ms timer data setting
MOVW I:TMCSR0,#0000100000011011B ;Interval timer operation, clock 2µs
 ;External trigger inhibit, external output inhibit,
 ;Reload mode selection, interrupt enabled
 ;Underflow generation flag clear, Count start
MOV ILM,#07H ;Sets ILM in PS to level 7
OR      CCR,#40H ;Interruption is enabled

LOOP:
 ;User processing
 ;
BRA     LOOP

;---------Interrupt Program----------------------------------------
WARI:
CLRB I:UF0 ;underflow generation flag is cleared
 ;User processing
```

343
CHAPTER 9 16-BIT RELOAD TIMER

RET ;Returning from interrupt processing

;----------Vector Settings---------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FF64H ;Vector set in interrupt number #38(26H)
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START

Program Example in Event Counter Mode

Processing specification

- An interrupt is generated when rising edges of the pulse input to the external event input pin are counted
  10,000 times by the 16-bit reload timer 0.
- Operation is performed in the one-shot mode.
- The rising edge is selected for the external trigger input.
- E12OS is not used.

Coding example

ICR13 EQU 0000BDH ;interrupt control register for 16-bit reload
timer
TMCSR0 EQU 000048H ;Timer control status register
TMR0  EQU 00004AH ;16-bit timer register
TMRLR0 EQU 00004AH ;16-bit reload register
DDR7  EQU 000017H ;Port 5 direction register
UF0   EQU TMCSR0:2 ;Underflow generation flag bit
CNTE0 EQU TMCSR0:1 ;Timer operation enable bit
TRG0  EQU TMCSR0:0 ;Software trigger bit

;----------Main program------------------------------------------------
CODE CSEG
; ;stack pointer (SP), etc. are to be initialized
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR13,#00H ;Interrupt levels 0 (strength)
MOV I:DDR7,#00H ;Sets the P70/INT0 pin for input.
CLRB I:CNTE0 ;counter paused
MOV I:TMRLR0,#270FH ;Reload value 10,000 times setting
MOV I:TMCSR0,#0000110010001011B ;Counter operation, external trigger, rising
edge, ;Disables external output and selects One-shot
mode ;Enables interrupts and clears underflow
;generation flags,
;Start count
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled
CHAPTER 9  16-BIT RELOAD TIMER

LOOP:

• User processing
• BRA LOOP

Interrupt Program;-----------------------------

WARI:

CLRB I:UFO ; underflow generation flag is cleared
• User processing
• RETI ; Returning from interrupt processing

CODE ENDS

Vector Settings;---------------------------------

VECT CSEG ABS=0FFH
ORG 00FF64H ; vector set in interrupt number #38 (26H)
DSL WARI
ORG 00FFDCH ; Reset vector setting
DSL START
DB 00H ; Sets single-chip mode

VECT ENDS

END START
CHAPTER 10
8-/16-BIT PPG TIMER

This chapter explains the functions and operations of the 8-/16-bit PPG timer.

10.1 Overview of 8-/16-bit PPG Timer
10.2 Block Diagram of 8-/16-bit PPG Timer
10.3 Configuration of 8-/16-bit PPG Timer
10.4 Interrupts of 8-/16-bit PPG Timer
10.5 Explanation of Operation of 8-/16-bit PPG Timer
10.6 Precautions when Using 8-/16-bit PPG Timer
10.7 Program Example for 8-/16-bit PPG Timer
10.1 Overview of 8-/16-bit PPG Timer

The 8-/16-bit PPG timer is the 2-channel reload timer module (PPG0 or PPG1) that can perform pulse output of the set cycle and duty ratio. The following operation can be performed by combining the 2-channel module.

- 8-bit PPG output 2-channel independent operation mode
- 16-bit PPG output operation mode
- 8 + 8-bit PPG output operation mode

Functions of 8-/16-bit PPG Timer

The 8-/16-bit PPG timer consists of four 8-bit reload registers (PRLH0/PRL0 and PRLH1/PRL1) and two PPG down counters (PCNT0 and PCNT1).

- The "H" width and "L" width can separately be set as the output pulse, so output pulse of cycle and duty ratio can be set randomly.
- The count clock can be selected from six internal clocks.
- The 8-/16-bit PPG timer can be used as an interval timer by generating an interrupt request at each interval time.
- An external circuit enables the 8-/16-bit PPG timer to be used as a D/A converter.

Operation Modes of 8-/16-bit PPG Timer

- 8-bit PPG output 2-channel independent operation mode

2-channel modules (PPG0 and PPG1) operate as independent 8-bit PPG timers.

Table 10.1-1 shows the interval times in 8-bit PPG Output 2-channel independent operation mode and Table 10.1-2 shows the relationship between the channel and output pin.

Table 10.1-1 Interval Times in 8-bit PPG Output 2-channel Independent Operation Mode

<table>
<thead>
<tr>
<th>Count Clock</th>
<th>PPG0, PPG1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Interval Time</td>
</tr>
<tr>
<td>φ/1 (62.5ns)</td>
<td>1/φ to 2(^3/φ)</td>
</tr>
<tr>
<td>φ/2 (125ns)</td>
<td>2/φ to 2(^3/φ)</td>
</tr>
<tr>
<td>2(^2/φ) (250ns)</td>
<td>2(^3/φ) to 2(^10/φ)</td>
</tr>
<tr>
<td>2(^3/φ) (500ns)</td>
<td>2(^3/φ) to 2(^11/φ)</td>
</tr>
<tr>
<td>2(^4/φ) (1µs)</td>
<td>2(^3/φ) to 2(^12/φ)</td>
</tr>
<tr>
<td>2(^9)/HCLK (128µs)</td>
<td>2(^9)/HCLK to 2(^17)/HCLK</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency
φ: Machine clock frequency
Descriptions within () indicate calculation example for count clock cycle when operations are carried out while HCLK = 4 MHz and φ = 16 MHz.
16-bit PPG output operation mode

This will operate as a 16-bit 1-channel PPG timer by linking the 2-channel modules (PPG0 and PPG1). Interval times for the 16-bit PPG output operation mode are shown in Table 10.1-3, and the support relationship between the channels and output pins is shown in Table 10.1-4.

### Table 10.1-2 Correspondence between channel and output pin in 8-bit PPG output 2-channel independent operation mode

<table>
<thead>
<tr>
<th>Channel</th>
<th>Output Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG ch0</td>
<td>PPG00 = PPG01</td>
</tr>
<tr>
<td>PPG ch1</td>
<td>PPG10 = PPG11</td>
</tr>
</tbody>
</table>

### Table 10.1-3 Interval Times in 16-bit PPG Output Operation Mode

<table>
<thead>
<tr>
<th>Count Clock</th>
<th>Interval Time</th>
<th>Output Pulse Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/φ (62.5ns)</td>
<td>1/φ to 2(^{16}/φ)</td>
<td>2/φ to 2(^{17}/φ)</td>
</tr>
<tr>
<td>2/φ (125ns)</td>
<td>2/φ to 2(^{17}/φ)</td>
<td>2(^2/φ) to 2(^{18}/φ)</td>
</tr>
<tr>
<td>2(^2/φ) (250ns)</td>
<td>2(^2/φ) to 2(^{18}/φ)</td>
<td>2(^3/φ) to 2(^{19}/φ)</td>
</tr>
<tr>
<td>2(^3/φ) (500ns)</td>
<td>2(^3/φ) to 2(^{19}/φ)</td>
<td>2(^4/φ) to 2(^{20}/φ)</td>
</tr>
<tr>
<td>2(^4/φ) (1µs)</td>
<td>2(^4/φ) to 2(^{20}/φ)</td>
<td>2(^5/φ) to 2(^{21}/φ)</td>
</tr>
<tr>
<td>2(^9)/HCLK (128µs)</td>
<td>2(^9)/HCLK to 2(^{25}/HCLK)</td>
<td>2(^10)/HCLK to 2(^{26}/HCLK)</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency
φ: Machine clock frequency
Description in the () indicates the count clock calculation example when operated while HCLK = 4 MHz and φ = 16 MHz.

### Table 10.1-4 16-bit PPG output operation mode

<table>
<thead>
<tr>
<th>Channel</th>
<th>Output Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG ch0, ch1</td>
<td>PPG00 = PPG01 = PPG10 = PPG11</td>
</tr>
</tbody>
</table>
8 + 8-bit PPG output operation mode

The PPG0 out of the 2-channel module is operated as an 8-bit prescaler, and the PPG0 underflow output is input and operated as the PPG1 count clock.

Interval times for the 8+8-bit PPG output operation mode are shown in Table 10.1-5, and the support relationship between the channels and output pins is shown in Table 10.1-6.

**Table 10.1-5 Interval Times in 8 + 8-bit PPG Output Operation Mode**

<table>
<thead>
<tr>
<th>Count Clock</th>
<th>PPG0 Interval Time</th>
<th>PPG0 Output Pulse Time</th>
<th>PPG1 Interval Time</th>
<th>PPG1 Output Pulse Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/φ (62.5ns)</td>
<td>1/φ to 28/φ</td>
<td>28/φ to 29/φ</td>
<td>1/φ to 216/φ</td>
<td>216/φ to 217/φ</td>
</tr>
<tr>
<td>2/φ (125ns)</td>
<td>2/φ to 29/φ</td>
<td>29/φ to 210/φ</td>
<td>2/φ to 217/φ</td>
<td>217/φ to 218/φ</td>
</tr>
<tr>
<td>22/φ (250ns)</td>
<td>22/φ to 210/φ</td>
<td>210/φ to 211/φ</td>
<td>22/φ to 218/φ</td>
<td>218/φ to 219/φ</td>
</tr>
<tr>
<td>23/φ (500ns)</td>
<td>23/φ to 211/φ</td>
<td>211/φ to 212/φ</td>
<td>23/φ to 219/φ</td>
<td>219/φ to 220/φ</td>
</tr>
<tr>
<td>24/φ (1μs)</td>
<td>24/φ to 212/φ</td>
<td>212/φ to 213/φ</td>
<td>24/φ to 220/φ</td>
<td>220/φ to 221/φ</td>
</tr>
<tr>
<td>29/HCLK (128μs)</td>
<td>29/HCLK to 217/HCLK</td>
<td>217/HCLK to 218/HCLK</td>
<td>29/HCLK to 225/HCLK</td>
<td>225/HCLK to 226/HCLK</td>
</tr>
</tbody>
</table>

HCLK: Oscillation clock frequency
φ: Machine clock frequency
Description in the () indicates the count clock calculation example when operated while HCLK = 4 MHz and φ = 16 MHz.

**Table 10.1-6 8 + 8-bit PPG output operation mode**

<table>
<thead>
<tr>
<th>Channel</th>
<th>Output Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG ch0</td>
<td>PPG00 = PPG01</td>
</tr>
<tr>
<td>PPG ch1</td>
<td>PPG10 = PPG11</td>
</tr>
</tbody>
</table>
10.2 Block Diagram of 8-/16-bit PPG Timer

The 8-/16-bit PPG timer is configured with 2-channel 8-bit PPG timer. Figure 10.2-1 shows block diagram of 8-/16-bit PPG timer 0 and 8-/16-bit PPG timer 1.

### Channels and PPG Pins of PPG Timers

Figure 10.2-1 shows the relationship between the channels and the PPG pins of the 8-/16-bit PPG timers in MB90520A series.

#### Figure 10.2-1 Channels and PPG Pins of PPG Timers

- **PPG0**
  - PPG00 output pin
  - Pin
  - PPG01 output pin

- **PPG1**
  - PPG10 output pin
  - Pin
  - PPG11 output pin
10.2.1 Block Diagram for 8-/16-bit PPG Timer 0

Figure 10.2-2 shows block diagram for 8-/16-bit PPG Timer 0.

---

**Block Diagram for 8-/16-bit PPG Timer 0**

![Block Diagram for 8-/16-bit PPG Timer 0](image-url)

**Legend:**
- **R:** Reset
- **S:** Set
- **Q:** Output
- **PEN0:** PPG0 Enable
- **PE00:** PPG0 Enable (to PPG1)
- **PIE0:** PPG0 Interrupt Enable
- **PUF0:** PPG0 Interrupt Flag
- **Reserved:**
- **HCLK:** Oscillation clock frequency
- **φ:** Machine clock frequency
- **Reserved bit:**
- **Undefined:**
The channel, output pin, and interrupt request number details within the block diagram

Table 10.2-1 shows pins name of 8-/16-bit PPG timer and interrupt request number.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Output Pin</th>
<th>Interrupt Request Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG0</td>
<td>P36/PPG00</td>
<td>#19 (13H)</td>
</tr>
<tr>
<td></td>
<td>P37/PPG01</td>
<td></td>
</tr>
</tbody>
</table>

- **PPG0 Operation Mode Control Register (PPGC0)**
  This register enables or disables operation of the 8-/16-bit PPG timer, the pin output, and an underflow interrupt. Underflow generation can also be checked.

- **PPG01 output control register (PPGOE)**
  The count clock of the 8-/16-bit PPG timer can be selected, and the pin output can be enabled or disabled.

- **PPG0 reload registers (PRLH0 and PRLL0)**
  "H" width or "L" width for output pulse can be set. The value set to the PPG0 reload register will be reloaded to the PPG0 down counter (PCNT0) when the 8-/16-bit PPG timer is activated.

- **PPG0 down counter (PCNT0)**
  This 8-bit down counter alternately reloads and counts down the values set by the PPG0 reload registers (PRLH0 and PRLL0). Pin output is reversed when underflow is generated.
  2-channel PPG down counter is concatenated for use as a single-channel 16-bit PPG down counter.

- **PPG0 temporary buffer (PRLBH0)**
  This buffer prevents misalignment of the output pulse width generated through the timing of writing to the PPG0 reload register (PRLH0 and PRLL0).

- **Reload register L/H selector**
  The reload register on the "L" level and "H" level sides (PRLL0, PRLH0) are referenced and which value shall reload in the PPG0 down counter is selected by detecting the current pin output level.

- **Count clock selector**
  This selector selects the count clock to be input to the PPG0 down counter from five frequency (ϕ/1, ϕ/2, ϕ/4, ϕ/8, ϕ/16)-divided clocks of the machine clock or the frequency-divided clocks of the timebase timer.

- **PPG output control circuit**
  Output reversal is performed depending on pin output level settings and underflow generation.
10.2.2 Block Diagram of 8-/16-bit PPG Timer 1

Figure 10.2-3 shows Block Diagram of 8-/16-bit PPG Timer 1.

- **Block Diagram of 8-/16-bit PPG Timer 1**

---

**Figure 10.2-3 Block Diagram of 8-/16-bit PPG Timer 1**

- PPG0 reload register
- Operation mode control signal
- PPG1 Temporary buffer (PRLBH1)
- Reload register "L" level/"H" level selector
- Count start value
- Reload
- PPG1 Down counter (PCNT1)
- PPG0 Underflow (from PPG0)
- Timebase timer output (HCLK/512)
- Peripheral clock (φ/1)
- Peripheral clock (φ/2)
- Peripheral clock (φ/4)
- Peripheral clock (φ/8)
- Peripheral clock (φ/16)
- Count clock selector
- Select signal
- PPG01 output control register (PPGOE)

---

- : Undefined
- : Reserved bit
- HCLK : Oscillation clock frequency
- : Machine clock frequency
The channel, output pin, and interrupt request number details within the block diagram

Table 10.2-2 shows pin name of 8-/16-bit PPG timer and interrupt request number.

Table 10.2-2 Pins and Interrupt Request Numbers in Block Diagram

<table>
<thead>
<tr>
<th>Channel</th>
<th>Output Pin</th>
<th>Interrupt Request Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG1</td>
<td>P40/PPG10</td>
<td>#23 (17H)</td>
</tr>
<tr>
<td></td>
<td>P41/PPG11</td>
<td></td>
</tr>
</tbody>
</table>

PPG1 Operation Mode Control Register (PPGC1)

Setting is enabled operation modes of 8-/16-bit PPG Timer. Enables/disables 8-/16-bit PPG timer operation, pin output and underflow interrupts. The generation of an underflow can be checked.

PPG01 output control register (PPGOE)

Selects 8-/16-bit PPG timer count clock selection and enables/disables pin output.

PPG1 reload registers (PRLH1 and PRLL1)

"H" width or "L" width for output pulse can be set. Values set in the PPG1 reload register are reloaded into the PPG1 down counter (PCNT1) when the 8/16 bit PPG timer 1 is started up.

PPG1 down counter (PCNT1)

This 8-bit down counter alternately reloads and counts down the values set by the PPG1 reload registers (PRLH1 and PRLL1). Pin output is reversed when underflow is generated.

2-channel PPG down counter (PPG0 and PPG1) is concatenated for use as a single-channel 16-bit PPG down counter.

PPG1 temporary buffer (PRLBH1)

This buffer prevents misalignment of the output pulse width generated through the timing of writing to the PPG1 reload register (PRLH1 and PRLL1).

Reload register L/H selector

The reload register on the "L" level and "H" level sides (PRLL1, PRLH1) are referenced and which value shall reload in the PPG1 down counter is selected by detecting the current pin output level.

Count clock selector

This selector selects the count clock to be input to the PPG1 down counter from five frequency (φ/1, φ/2, φ/4, φ/8, φ/16)-divided clocks of the machine clock or the frequency-divided clocks of the timebase timer.

In 16-bit PPG output operation mode and 8 + 8 bit PPG output operation mode, the PPG0 underflow output is selected as count clock.

PPG output control circuit

Output reversal is performed depending on pin output level and underflow generation.
10.3 Configuration of 8-/16-bit PPG Timer

Described Pins of 8-/16-bit PPG Timer, interrupt factor, detail of register.

■ Pins of 8-/16-bit PPG Timer

Pins of 8-/16-bit PPG Timer is used together with the general-purpose input/output port. Settings using the pin function, pin name and 8-/16-bit PPG timer are shown in Table 10.3-1.

Table 10.3-1 Pins of 8-/16-bit PPG Timer

<table>
<thead>
<tr>
<th>Channel</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting of the pins required for use of 8-/16-bit PPG timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG0</td>
<td>P36/PPG0 0</td>
<td>General-purpose I/O ports/PPG00 output pin</td>
<td>Enabling PPG00 pin output (PPGC0: PE00=1)</td>
</tr>
<tr>
<td></td>
<td>P37/PPG01</td>
<td>General-purpose I/O ports/PPG01 output pin</td>
<td>Enabling PPG00 pin output (PPGOE: PE01=1)</td>
</tr>
<tr>
<td>PPG1</td>
<td>P40/PPG10</td>
<td>General-purpose I/O ports/PPG10 output pin</td>
<td>Enabling PPG10 pin output (PPGC1: PE10=1)</td>
</tr>
<tr>
<td></td>
<td>P41/PPG11</td>
<td>General-purpose I/O ports/PPG11 output pin</td>
<td>Enabling PPG11 pin output (PPGOE: PE11=1)</td>
</tr>
</tbody>
</table>

■ Block Diagram of 8-/16-bit PPG Timer Pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

■ Generation of Interrupt from 8-/16-bit PPG Timer

The underflow generation flag bit of PPG operation mode control register (PPGC0:PUF0, PPGC1:PUF1) in the 8-/16-bit PPG timer is set to "1" when an underflow occurs. An underflow interrupt request is generated when an underflow occurs in a channel that is set to enable underflow interrupts (PPGC0:PIE0=1, PPGC1:PIE1=1).
### 10.3.1 PPG0 Operation Mode Control Register (PPGC0)

PPG0 operation mode control register provides the following settings:
- Operation of 8-/16-bit PPG Timer enabled
- Pin function switching (pulse output enabled)
- Underflow interrupt enabled
- Checks and clears the underflow generation flag

**Figure 10.3-1 PPG0 Operation Mode Control Register (PPGC0)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Reserved  bit</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Reserved  bit</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Reserved  bit</td>
<td></td>
</tr>
</tbody>
</table>

#### Underflow generation flag bit
- **PUF0**
  - When reading is performed:
    - 0: No underflow
    - 1: Detect underflow
  - When write is specified:
    - 0: Clear this PUF0 bit
    - 1: No effect

#### Underflow interrupt enable bit
- **PIE0**
  - 0: Underflow interrupt disabled
  - 1: Underflow interrupt enabled

#### PPG00 Pin output level select bit
- **PE00**
  - 0: General - purpose I/O port (Pulse output disabled)
  - 1: PPG00 output (Pulse output enabled)

#### PPG0 Operation enable bit
- **PEN0**
  - 0: Count operation disabled (Retention of "L" level output)
  - 1: Count operation enabled

---

357
Table 10.3-2 Functions of PPG0 Operation Mode Control Register (PPGC0)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>Reserved: reserved bit Always set this bit to &quot;1&quot;.</td>
</tr>
<tr>
<td>bit1 bit2</td>
<td>Undefined bits Read: The value is undefined. Write: No effect</td>
</tr>
<tr>
<td>bit3</td>
<td>PUF0: Underflow generation flag bit 8-bit PPG output 2-channel independent operation mode, 8 + 8-bit PPG output operation mode: When the value of the PPG0 down counter is counted down from &quot;00H&quot; to &quot;FFH&quot;, an underflow occurs (PUF0 = 1). 16-bit PPG output operation mode: When the values of the PPG0 and PPG1 down counters are counted down from &quot;0000H&quot; to &quot;FFFFH&quot;, an underflow occurs (PUF0 = 1). • While underflow interrupt is enabled (PIE0 = 1), if underflow is generated (PUF0 = 1), an interrupt request is generated. <strong>When the bit is set to &quot;0&quot;:</strong> The bit is cleared. <strong>When the bit is set to &quot;1&quot;:</strong> No effect. &quot;1&quot; is read under the read-modify-write command.</td>
</tr>
<tr>
<td>bit4</td>
<td>PIE0: Underflow interrupt enable bit This bit enables or disables an interrupt. <strong>When the bit is set to &quot;0&quot;:</strong> No interrupt request generated even at underflow (PUF0 = 1). <strong>When the bit is set to &quot;1&quot;:</strong> Interrupt request generated at underflow (PUF0 = 1)</td>
</tr>
<tr>
<td>bit5</td>
<td>PE00: PPG0 pin output enable bit The PPG0 pin function is switched and pulse output is enabled or disabled. <strong>When the bit is set to &quot;0&quot;:</strong> Functions as general-purpose I/O port. The pulse output is disabled. <strong>When the bit is set to &quot;1&quot;:</strong> Functions as PPG0 output pin. The pulse output is enabled.</td>
</tr>
<tr>
<td>bit6</td>
<td>Undefined bits Read: The value is undefined. Write: No effect</td>
</tr>
<tr>
<td>bit7</td>
<td>PEN0: PPG0 operation enable bit Enable/disable of count operation of 8-/16-bit PPG Timer 0 <strong>When the bit is set to &quot;0&quot;:</strong> Count operation disabled <strong>When the bit is set to &quot;1&quot;:</strong> Count operation enabled • When the count operation is disabled (PEN0 = 1), the output is held at a Low level.</td>
</tr>
</tbody>
</table>
### 10.3.2 PPG1 Operation Mode Control Register (PPGC1)

PPG1 operation mode control register can be set as follows.
- Operation of 8-/16-bit PPG Timer is enabled.
- Pin function switching (pulse output enabled)
- Enabling underflow interrupts and checking and clearing underflow generation flags
- Setting operation mode of 8-/16-bit PPG timer

#### PPG1 Operation Mode Control Register (PPGC1)

![Figure 10.3-2 PPG1 Operation Mode Control Register (PPGC1)](image)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Purpose</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>PEN1</td>
<td>Count operation enabled</td>
</tr>
<tr>
<td>14</td>
<td>PE10</td>
<td>PPG10 Pin output level select bit</td>
</tr>
<tr>
<td>13</td>
<td>PIE1</td>
<td>Underflow interrupt enable bit</td>
</tr>
<tr>
<td>12</td>
<td>PUF1</td>
<td>Underflow generation flag bit</td>
</tr>
<tr>
<td>11</td>
<td>MD1</td>
<td>Operation mode select bit</td>
</tr>
</tbody>
</table>
| 10  | MD0     | 0: 8-bit PPG output 2ch independent operation mode
|     | 1       | 8+8 bit PPG output operation mode |
| 9   | Reserved|      |
| 8   | Reserved|      |

**Figure 10.3-2 PPG1 Operation Mode Control Register (PPGC1)**

- **R/W**: Readable and Writable
- **X**: Undefined
- **-**: Unused
- **Reset value**: 0X0000018
Table 10.3-3 Functions of PPG1 Operation Mode Control Register (PPGC1)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8</td>
<td>Reserved: reserved bit</td>
</tr>
</tbody>
</table>
| bit10, bit9 | MD1, MD0: Operation mode selection bits | Setting of operation modes of 8-/16-bit PPG Timer is enabled. (Any mode other than 8-bit PPG output 2-channel independent operation mode)  
- Set 2 bits simultaneously using the word command for the PPG operation enabling bit (PEN0 and PEN1).  
- Do not set operation of only one of the two channels (PEN1 = 0/PEN0 = 1 or PEN1 = 1/PEN0 = 0).  
**Note:**  
Do not set the MD1 and MD0 bits to "10B". |
| bit11    | PUF1: Underflow generation flag bit | **8-bit PPG output 2-channel independent operation mode:** When the value of the PPG1 down counter is counted down from "00H" to "FFH", an underflow occurs (PUF1 = 1).  
**16-bit PPG output operation mode:** When the values of the PPG0 and PPG1 down counters are counted down from "0000H" to "FFFFH", an underflow occurs (PUF1 = 1).  
- While underflow interrupt is enabled (PIE0 = 1), if underflow is generated (PUF0 = 1), an interrupt request is generated.  
**When the bit is set to "0":** The bit is cleared.  
**When the bit is set to "1":** No effect.  
"1" is read under the read-modify-write command. |
| bit12    | PIE1: Underflow interrupt enable bit | Enable or disable interruption.  
**When the bit is set to "0":** No interrupt request is generated even at underflow (PUF1 = 1)  
**When the bit is set to "1":** Interrupt request is generated at underflow (PUF1 = 1) |
| bit13    | PE10: PPG10 Pin output enable bit | The PPG10 pin function is switched and pulse output is enabled or disabled.  
**When the bit is set to "0":** Functions as general-purpose I/O port. The pulse output is disabled.  
**When the bit is set to "1":** Functions as PPG10 output pin. The pulse output is enabled. |
| bit14    | Undefined bits | **Read:** The value is undefined.  
**Write:** No effect |
| bit15    | PEN1: PPG1 operation enable bit | Count operation of 8-/16-bit PPG Timer 1 is enabled/disabled.  
**When the bit is set to "0":** Count operation disabled  
**When the bit is set to "1":** Count operation enabled  
- When the count operation is disabled (PEN1 = 1), the output is held at a Low level. |
10.3.3 PPG01 Output Control Register (PPGOE)

PPG01 output control register selects the 8-/16-bit PPG timer count clock, and enables or disables pin output.

**Figure 10.3-3 PPG01 output control register (PPGOE)**

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Reset Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCS2</td>
<td>PCS1</td>
<td>PCS0</td>
<td>PCM2</td>
<td>PCM1</td>
<td>PCM0</td>
<td>PE11</td>
<td>PE01</td>
<td>00000000 B</td>
<td></td>
</tr>
</tbody>
</table>

- **PE01**: PPG01 Pin output enable bit
  - 0: general-purpose I/O port (pulse output disabled)
  - 1: PPG01 output (pulse output enabled)

- **PE11**: PPG11 Pin output enable bit
  - 0: general-purpose I/O port (pulse output disabled)
  - 1: PPG11 output (pulse output enabled)

- **PCM2, PCM1, PCM0**: PPG0 Count clock select bit
  - 0 0 0: φ/1 (62.5ns)
  - 0 0 1: φ/2 (125ns)
  - 0 1 0: φ/4 (250ns)
  - 0 1 1: φ/8 (500ns)
  - 1 0 0: φ/16 (1μs)
  - 1 0 1: Setting enabled
  - 1 1 0: Setting disabled
  - 1 1 1: 2^5/HCLK (128μs)

- **PCS2, PCS1, PCS0**: PPG1 Count clock select bit
  - 0 0 0: φ/1 (62.5ns)
  - 0 0 1: φ/2 (125ns)
  - 0 1 0: φ/4 (250ns)
  - 0 1 1: φ/8 (500ns)
  - 1 0 0: φ/16 (1μs)
  - 1 0 1: Setting enabled
  - 1 1 0: Setting disabled
  - 1 1 1: 2^5/HCLK (128μs)

Values in parentheses ( ) are the calculation example of count clock cycle operating at HCLK = 4 MHz and $\phi = 16$ MHz.
### Table 10.3-4 Function of the PPG01 output control selection register (PPGOE)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>The PPG01 pin function is switched and pulse output is enabled or disabled. When the bit is set to &quot;0&quot;: Functions as general-purpose I/O port. The pulse output is disabled. When the bit is set to &quot;1&quot;: Functions as PPG01 output pin. The pulse output is enabled.</td>
</tr>
<tr>
<td>bit1</td>
<td>The PPG11 pin function is switched and pulse output is enabled or disabled. When the bit is set to &quot;0&quot;: Functions as general-purpose I/O port. The pulse output is disabled. When the bit is set to &quot;1&quot;: Functions as PPG11 output pin. The pulse output is enabled.</td>
</tr>
<tr>
<td>bit4 to bit2</td>
<td>Count clock of the 8-/16-bit PPG timer 0 can be set.</td>
</tr>
<tr>
<td>bit5</td>
<td>Count clock of the 8-/16-bit PPG timer 1 can be set.</td>
</tr>
<tr>
<td>bit5</td>
<td>The count clock can be selected from five frequency-divided clocks of the machine clock, or the frequency-divided clocks of the timebase timer.</td>
</tr>
<tr>
<td>bit7 to bit5</td>
<td>PPG1 count clock selection bits (PCS2 to PCS0) can be set only under the 8-bit PPG output 2-channel independent operation mode (PPG1:MD1 and MD0 = &quot;00B&quot;).</td>
</tr>
</tbody>
</table>
10.3.4 PPG Reload Registers (PRL0/PRLH0, PRL1/PRLH1)

The PPG reload register makes it possible to set the value where a PPG down count starts counting (reload value). They are an 8-bit register at Low level and an 8-bit register at High level.

PPG Reload Registers (PRL0/PRLH0, PRL1/PRLH1)

---

**Table 10.3-5 Functions of PPG Reload Registers**

<table>
<thead>
<tr>
<th>Functions</th>
<th>8-/16-bit PPG Timer 0</th>
<th>8-/16-bit PPG Timer 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retains reload value on &quot;L&quot; level side</td>
<td>PRL0</td>
<td>PRL1</td>
</tr>
<tr>
<td>Retains reload value on &quot;H&quot; level side</td>
<td>PRLH0</td>
<td>PRLH1</td>
</tr>
</tbody>
</table>

**Notes:**
- In the 16-bit PPG output operation mode (PPGC1: MD1, MD0 = "11B"), use a long-word instruction to set the reload registers or the word instruction to set the PPG0 and PPG1 in this order.
- In the 8 + 8-bit PPG output operation mode (PPGC1: MD1, MD0 = "01B"), set the same value in both the Low-level and High-level PPG reload registers (PRTL0/PRLH0) of the 8-/16-bit PPG timer 0. When different values are set to the "L" level side and "H" level side, generated the misalignment of the duty.
10.4 Interrupts of 8-/16-bit PPG Timer

The 8-/16-bit PPG timer can generate an interrupt request when the PPG down counter underflow.

Interrupts of 8-/16-bit PPG Timer

Table 10.4-1 shows the interrupt control bits and interrupt factor of the 8-/16-bit PPG timer.

Table 10.4-1 The interrupt control bits of the 8-/16-bit PPG timer

<table>
<thead>
<tr>
<th></th>
<th>PPG0</th>
<th>PPG1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Underflow generation flag bit</td>
<td>PPGC0: PUF0</td>
<td>PPGC1: PUF1</td>
</tr>
<tr>
<td>Interrupt enable bit</td>
<td>PPGC0: PIE0</td>
<td>PPGC1: PIE1</td>
</tr>
<tr>
<td>Interrupt factor</td>
<td>Underflow generation of PPG0 down counter</td>
<td>Underflow generation of PPG1 down counter</td>
</tr>
</tbody>
</table>

(8-bit PPG output 2-channel independent operation mode or 8 + 8-bit PPG output operation mode)

- In the 8-bit PPG 2ch independent operating mode or 8 + 8-bit PPG output operating mode, PPG0 and PPG1 generate interrupts independently.
- When the value of the PPG0 or PPG1 down counter is counted down from "00H" to "FFH", an underflow occurs. When an underflow occurs, the underflow generation flag bit in the channel causing an underflow is set (PPGC0: PUF0 = 1 or PPGC1: PUF1 = 1).
- When interrupts are enabled in a channel where an underflow has occurred (PPGC0: PIE0=1 or PPGC1: PIE1=1), an interrupt request is generated.

(16-bit PPG output operation mode)

- In the 16-bit PPG output operation mode, when the values of the PPG0 + PPG1 down counters are counted down from "0000H" to "FFFFH", an underflow occurs. When an underflow occurs, the underflow generation flag bits in the two channels are set at one time (PPGC0: PUF0 = 1 and PPGC1: PUF1 = 1).
- When an interrupt is enabled in one of two channels (PPGC0: PIE0=0, PPGC1: PIE1=1 or PPGC0: PIE0=1, PPGC1: PIE1=0), an interrupt request is generated when an underflow occurs.
- Disable one of the underflow interrupt enabling bits on the 2 channels to prevent simultaneous generation of interrupt requests (PPGC0: PIE0 = 0 and PPGC1: PIE1 = 1, or PPGC0: PIE0 = 1 and PPGC1: PIE1 = 0).
- When the two channels of the underflow generation flag bits are set (PPGC0: PUF0 = 1 and PPGC1: PUF1 = 1), clear the two channels at the same time.

Detail of Interrupts of 8-/16-bit PPG Timer

Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.
10.5 Explanation of Operation of 8-/16-bit PPG Timer

The 8-/16-bit PPG timer outputs a pulse width at any frequency and at any duty ratio continuously.

Operation of 8-/16-bit PPG Timer

- Output operation of 8-/16-bit PPG timer
  - There are two 8-/16-mode PPG timers for each channel ("L" level and "H" level) each of which has an 8-bit reload timer (PRLL0/PRLH0, PRLL1/PRLH1).
  - Values set in the reload register (PRLL0/PRLH0, PRLL1/PRLH1) are reloaded into the PPG counters (PCNT0, PCNT1) alternately.
  - After reloading to the PPG down counter and count down starts synchronized with the count clock set with the PPG count clock selection bit (PPGOE:PCM2 to PCM0, PCS2 to PCS0).
  - Pin output is reversed when the value set in the reload register is reloaded into the PPG down counter due to an underflow.

Figure 10.5-1 shows the output waveform of the 8-/16-bit PPG timer.

Figure 10.5-1 Output Waveform of 8-/16-bit PPG Timer

<table>
<thead>
<tr>
<th>Start operation</th>
<th>Stop operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG operation enable bit (PEN)</td>
<td></td>
</tr>
<tr>
<td>PPG output pin</td>
<td></td>
</tr>
<tr>
<td>L: PPG reload register (PRLL) value</td>
<td>T x (L+1)</td>
</tr>
<tr>
<td>H: PPG reload register (PRLH) value</td>
<td>T x (H+1)</td>
</tr>
<tr>
<td>T: Count Clock Cycle</td>
<td>Generating interruption timing</td>
</tr>
</tbody>
</table>

Operation Modes of 8-/16-bit PPG Timer

As long as the operation of the 8-/16-bit PPG timer is enabled (PPGC0: PEN0 = 1, PPGC1: PEN1 = 1), a pulse waveform is output continuously from the PPG output pin. Set any pulse waveform frequency and duty ratio.

The 8-/16-bit PPG timer pulse output does not stop until the 8-/16-bit PPG timer operation stops (PPGC0: PEN0=0, PPGC1: PEN1=0).

8-/16-bit PPG timer has following three types of operation mode.

- 8-bit PPG output 2-channel independent operation mode
- 16-bit PPG output operation mode
- 8 + 8-bit PPG output operation mode
10.5.1 8-bit PPG Output 2-channel Independent Operation Mode

In the 8-bit PPG output 2-channel independent operation mode, the 8-/16-bit PPG timer is set as an 8-bit PPG timer with two independent channels. PPG output operation, interrupt request generation and other operations can be performed independently for each channel.

Setting for 8-bit PPG Output 2-channel Independent Operation Mode

Operating the 8-/16-bit PPG timer in the 8-bit PPG output 2-channel independent operation mode requires the setting shown in Figure 10.5-2.

**Figure 10.5-2 Setting for 8-bit PPG Output 2-channel Independent Operation Mode**

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEN1</td>
<td>−</td>
<td>PE10</td>
<td>PIE1</td>
<td>PUF1</td>
<td>MD1</td>
<td>MD0</td>
<td>Reserved</td>
<td>PEN0</td>
<td>−</td>
<td>PE00</td>
<td>PIE0</td>
<td>PUF0</td>
<td>−</td>
<td>−</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

PPGOE : (Reserved area) PPS2, PPS1, PPS0, PCSM, PCS0, PCS1, PCS2, PCS3, PCM0, PCM1, PCM2, PCM3, PE11, PE10

PRLH0/PRHL0

PPG0 Set reload value on "H" level PPG0 Set reload value on "L" level

PRLH1/PRHL1

PPG1 Set reload value on "H" level PPG1 Set reload value on "L" level

☐ : Used bit
- : Undefined bit
1 : Set to "1"
0 : Set to "0"
X : Unused bit
△ : PPG01 pin. To enable the output of PPG11 pin, this pin should be set.

**Note:** Use the word instruction to set both High-level and "L" level PPG reload registers (PRLL0/PRHL0 and PRLL1/PRHL1) at the same time.
Operation in 8-bit PPG output 2-channel independent operation mode

- The 8-bit PPG timer with two channels performs an independent PPG operation.
- Table 10.5-1 shows pin output when PPG pin output is set to enabled.

**Table 10.5-1 Setting to enable output of the PPG terminal**

<table>
<thead>
<tr>
<th>PPG terminal</th>
<th>Setting to enable output of the PPG pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG0</td>
<td>PPGC0: PE00=1</td>
<td>Pulse wave of the PPG0</td>
</tr>
<tr>
<td></td>
<td>PPG00 terminal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPG01 terminal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPG0E: PE01=1</td>
<td></td>
</tr>
<tr>
<td>PPG1</td>
<td>PPGC1: PE10=1</td>
<td>Pulse wave of the PPG1</td>
</tr>
<tr>
<td></td>
<td>PPG10 terminal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPG11 terminal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPGO: PE11=1</td>
<td></td>
</tr>
</tbody>
</table>

PPGC0: PPG0 operation mode control register
PPGC1: PPG1 operation mode control register
PPGOE: PPG01 output control register

- When a reload value is set in the PPG reload register (PRLL0/PRLH0, PRLL1/PRLH1) and PPG timer operation is enabled (PPGC0: PEN0=1, PPGC1: PEN1=1), the PPG down counter starts count operation in the channel that enables operation.
- To stop the count operation of the PPG down counter, disable the operation of the PPG timer of the channel to be stopped (PPGC0: PEN0 = 0, PPGC1: PEN1 = 0). The count operation of the PPG down counter is stopped and the output of the PPG output pin is held at a Low level.
- When an underflow occurs in the PPG down counter of each channel, the reload value set in the PPG reload register (PRLL0/PRLH0, PRLL1/PRLH1) is reloaded to the PPG down counter where the underflow occurred.
- When an underflow occurs, the underflow generation flag bit (PPGC0:PUF0=1, PPGC1:PUF1=1) sets in the channel where the underflow occurred. An interrupt request is generated when interrupts are enabled (PPGC0:PIE0=1, PPGC1:PIE1=1) in the channel where an underflow occurs.
Output waveform in 8-bit PPG output 2-channel independent operation mode

- Both the "L" and "H" output pulse width adds 1 to the PPG reload register value of each channel and becomes a value multiplied by the count clock cycle. When the PPG reload register value is "00H", the count clock is set to 1-cycle pulse width and to 256-cycle pulse width when "FFH".

The equations for calculating the pulse width are shown below:

\[
\begin{align*}
PL &= T \times (L+1) \\
PH &= T \times (H+1)
\end{align*}
\]

- PL: Low width of output pulse
- PH: High width of output pulse
- L: Values of 8 bits in PPG reload register (PRLL0 or PRLL1)
- H: Values of 8 bits in PPG reload register (PRLH0 or PRLH1)
- T: Count clock cycle

Figure 10.5-3 shows the output waveform in the 8-bit PPG output 2-channel independent operation mode.
10.5.2 16-bit PPG Output Operation Mode

In the 16-bit PPG output operation mode, the 8-/16-bit PPG timer is set as a 16-bit PPG timer with one channel.

Setting for 16-bit PPG Output Operation Mode

Operating the 8-/16-bit PPG timer in the 16-bit PPG output operation mode requires the setting shown in Figure 10.5-4.

![Figure 10.5-4 Setting for 16-bit PPG Output Operation Mode](image)

**Note:** Set up the PPG reload register value using long word commands, or set up using word commands in PPG0 → PPG1 order.
CHAPTER 10  8-/16-BIT PPG TIMER

- Operation in 16-bit PPG output operation mode
  - Table 10.5-2 shows pin output when PPG pin output is set to enabled.

**Table 10.5-2 Setting to enable output of the PPG terminal**

<table>
<thead>
<tr>
<th>PPG terminal</th>
<th>Setting to enable output of the PPG pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG00</td>
<td>PPGC0: PE00=1</td>
<td>Pulse wave of 16-bit PPG timer</td>
</tr>
<tr>
<td>PPG01</td>
<td>PPGOE: PE01=1</td>
<td>PPG00=PPG01=PPG10=PPG11</td>
</tr>
<tr>
<td>PPG10</td>
<td>PPGC1: PE10=1</td>
<td></td>
</tr>
<tr>
<td>PPG11</td>
<td>PPGOE: PE11=1</td>
<td></td>
</tr>
</tbody>
</table>

PPGC0: PPG0 operation mode control register
PPGC1: PPG1 operation mode control register
PPGOE: PPG01 output control register

- When a reload value is set in the PPG reload register (PRLL0/PRLH0, PRLL1/PRLH1) and a PPG timer operation is also enabled (PPGC0:PEN0=1 and PPGC1:PEN1=1), the PPG down counter starts count operation as a 16-bit down counter (PCNT0+PCNT1).

- To stop the count operation of the PPG down counter, disable PPG timer operation for both channels (PPGC0:PEN0=0 and PPGC1:PEN1=0). The count operation of the PPG down counter is stopped and the output of the PPG output pin is held at a Low level.

- When a underflow occurs in the PPG down counter of each channel, the reload value set in the PPG0 and PPG1 reload registers (PRLL0/PRLH0, PRLL1/PRLH1) is simultaneously reloaded into PPG down counters (PCNT0 + PCNT1).

- When an underflow occurs, the underflow generation flag bit (PPGC0:PUF0=1, PPGC1:PUF1=1) in simultaneously set in both channels. When interrupts are enabled in one of the 2 channels (PPGC0:PIE0=1, PPGC1:PIE1=1), an interrupt request is generated.

**Notes:**
- In the 16-bit PPG output operation mode, the underflow generation flag bits in the 2 channels are set simultaneously when an underflow occurs (PPGC0: PU0 = 1 and PPGC1: PU1 = 1). Disable one of the underflow interrupt enabling bits on the 2 channels to prevent simultaneous generation of interrupt requests (PPGC0: PIE0 = 0 and PPGC1: PIE1 = 1 or PPGC0: PIE0 = 1 and PPGC1: PIE1 = 0).
- When the underflow generation flag bit is set, clears 2 channels simultaneously (PPGC0: PU0 = 0 and PPGC1: PU1 = 0).
Output waveform in 16-bit PPG output operation mode

Both the "L" and "H" output pulse width adds "1" to the PPG reload register value of each channel and becomes a value multiplied by the count clock cycle. When the PPG reload register value is "0000H", the count clock is set to 1-cycle pulse width and to 65,536-cycle pulse width when "FFFFH".

The equations for calculating the pulse width are shown below:

\[ PL = T \times (L + 1) \]
\[ PH = T \times (H + 1) \]

- \( PL \): Low width of output pulse
- \( PH \): High width of output pulse
- \( L \): Values of 16 bits in PPG reload register (PRLL0 + PRLL1)
- \( H \): Values of 16 bits in PPG reload register (PRLH0 + PRLH1)
- \( T \): Count clock cycle

Figure 10.5-5 shows the output waveform in the 16-bit PPG output operation mode.

**Figure 10.5-5 Output waveform in 16-bit PPG output operation mode**

<table>
<thead>
<tr>
<th>PPG operation enable bit (PEN)</th>
<th>Start operation</th>
<th>Stop operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG output pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- \( L \): 16-bit value of PPG reload register (PRLL1+PRLL0)
- \( H \): 16-bit value of PPG reload register (PRLH1+PRLH0)
- \( T \): Count Clock Cycle
10.5.3 8 + 8-bit PPG Output Operation Mode

In 8 + 8-bit PPG output operation mode, PPG0 operates as an 8-bit prescaler, PPG1 operates as an 8-bit PPG timer, the clock source for the PPG0 PPG output.

Setting for 8 + 8-bit PPG Output Operation Mode

Operating the 8-/16-bit PPG timer in the 8 + 8-bit PPG output operation mode requires the setting shown in Figure 10.5-6.

![Figure 10.5-6 Setting for 8 + 8-bit PPG Output Operation Mode](image)

**Note:** Use the word instruction to set both High-level and "L" level PPG reload registers (PRL0/PRLH0 and PRL1/PRLH1) at the same time.
Operation in 8 + 8-bit PPG output operation mode

- It operates as a 8-bit prescaler + 8-bit PPG timer.
- Table 10.5-3 shows pin output when PPG pin output is set to enabled.

**Table 10.5-3 Setting to enable output of the PPG terminal**

<table>
<thead>
<tr>
<th>PPG terminal</th>
<th>Setting of Enabling output of the PPG pin</th>
<th>Description</th>
</tr>
</thead>
</table>
| PPG0          | PPGC0: PE00=1                            | Pulse wave of PPG0
               |                                          | (PPG1 count clock) |
| PPG00 terminal| PPGC0: PE01=1                            | PPG 00=PPG01 |
| PPG01 terminal| PPGOE: PE01=1                            |             |
| PPG1          | PPGC1: PE10=1                            | Generated waveform at 8+8-bit
               |                                          | PPG timer     |
| PPG10 terminal| PPGOE: PE11=1                            | PPG10=PPG11 |
| PPG11 terminal|                                          |             |

PPGC0: PPG0 operation mode control register
PPGC1: PPG1 operation mode control register
PPGOE: PPG01 output control register

- When a reload value is set in the PPG reload register (PRL00/PRLH0, PRL10/PRLH1) and PPG timer operation is enabled (PPGC0:PEN0=1 and PPGC1:PEN1=1), the PPG down counter starts counting.
- To stop the count operation of the PPG down counter, disable PPG timer operation for both channels (PPGC0:PEN0=0 and PPGC1:PEN1=0). The count operation of the PPG down counter is stopped and the output of the PPG output pin is held at a Low level.
- When an underflow occurs in the PPG down counter of each channel, the reload value set in the PPG reload register (PRL00/PRLH0, PRL10/PRLH1) is reloaded to the PPG down counter where the underflow occurred.
- When an underflow occurs, the underflow generation flag bit (PPGC0:PUF0=1, PPGC1:PUF1=1) sets in the channel where the underflow occurred. When interrupts are enabled in a channel where an underflow has occurred (PPGC0:PIE0=1, PPGC1:PIE1=1), an interrupt request is generated.

**Notes:**
- Do not operate PPG1 (PPGC1:PEN1 = 1) when PPG0 is stopped (PPGC0:PEN0 = 0).
- Set the same value in both Low-level and High-level PPG reload registers (PRL00/PRLH0, PRL10/PRLH1).
Output waveform in 8-bit PPG output 2-channel independent operation mode

- Both the "L" and "H" output pulse width adds "1" to the PPG reload register value of each channel and becomes a value multiplied by the count clock cycle.

The equations for calculating the pulse width are shown below:

\[ P_L = T \times (L_0 + 1) \times (L_1 + 1) \]
\[ P_H = T \times (H_0 + 1) \times (H_1 + 1) \]

- \( P_L \): "L" width of PPG1 output pin pulse
- \( P_H \): "H" width of PPG1 output pin pulse
- \( L_0 \): Values of 8 bits in PPG reload register (PRL0)
- \( L_0 \): Values of 8 bits in PPG reload register (PRLH0)
- \( L_1 \): Values of 8 bits in PPG reload register (PRL1)
- \( H_1 \): Values of 8 bits in PPG reload register (PRLH1)
- \( T \): Count clock cycle

Figure 10.5-7 shows the output waveform in the 8 + 8-bit PPG output operation mode.

**Figure 10.5-7 Output Waveform in 8 + 8-bit PPG Output Operation Mode**

<table>
<thead>
<tr>
<th>Start operation</th>
<th>Stop operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPG operation enable bit (PEN0, PEN1)</td>
<td></td>
</tr>
<tr>
<td>PPG0 output pin</td>
<td></td>
</tr>
<tr>
<td>PPG1 output pin</td>
<td></td>
</tr>
<tr>
<td>( T \times (L_0 + 1) \times (L_1 + 1) )</td>
<td>( T \times (H_0 + 1) \times (H_1 + 1) )</td>
</tr>
</tbody>
</table>

- \( L_0 \): 8-bit value of PPG reload register (PRL0)
- \( H_0 \): 8-bit value of PPG reload register (PRLH0)
- \( H_1 \): 8-bit value of PPG reload register (PRLH1)
- \( L_1 \): 8-bit value of PPG reload register (PRL1)
- \( T \): Count Clock Cycle

**Note:** When PPG1 is started up while PPG0 is operating under the 8 + 8-bit output operation mode, the first count cycle is misaligned.
10.6 Precautions when Using 8-/16-bit PPG Timer

Note the following regarding the use of the 8-/16-bit PPG timer.

■ Precautions when Using 8-/16-bit PPG Timer

● Effect on 8-/16-bit PPG timer when using timebase timer output
  - Timebase timer output is used for 8-/16-bit PPG timer count clock input (PPG0E:PCM2 to PCM0 = "111B", PCS2 to PCS0 = "111B"), the trigger input causes a misalignment in the first count cycle that the PPG timer is activated or count cycle directly after a stop.
  - Clearing the timebase timer counter (TBTC:TBR=0) during counting by the PPG down counter, causes count cycle misalignment.

● Setting of PPG reload registers when using 8-bit PPG timer
  - The "L" and "H" level pulse widths are determined by the timing when the "L" level PPG reload register (PRLL0, PRLL1) value is reloaded into the PPG down counter.
  - When the 8-bit PPG timer is used in 8-bit PPG output 2 ch independent operation mode or 8+8 bit PPG output operation mode, use word commands to simultaneously set both the "H" level and "L" level sides of the PPG reload register (PRLL0/PRLH0, PRLL1/PRLH1). Do not use byte instructions.

● Setting of PPG reload registers when using 16-bit PPG timer
  - Use long word commands to set the PPG reload register (PRLL0/PRLH0, PRLL1/PRLH1) or set them in PPG0 → PPG1 order using word commands.

(Reload timing in 16-bit PPG output operation mode)

In the 16-bit PPG output operation mode, the reload value set in the PPG0 reload register is temporarily set in a temporary launch and transferred to the PPG0 reload register when the reload value is written to the PPG1 reload register. To set a reload value in PPG1, set also a reload value in PPG0 simultaneously or set the reload value in PPG0 before setting the reload value in PPG1.

Figure 10.6-1 shows the reload timing in the 16-bit PPG output operation mode.
Figure 10.6-1 Reload Timing in 16-bit PPG Output Operation Mode

- **PPG0 reload value**
- **PPG1 reload value**
- **Temporary latch**
- **PPG Reload register (PRLL0, PRLH0)**
- **PPG Reload register (PRLL1, PRLH1)**

- **Write to PPG0 outside 16-bit PPG output operation mode**
- **16-bit PPG output operation mode only**
- **Transfer in synchronization with the writing to PPG1**
- **Write to PPG1**
10.7  Program Example for 8-/16-bit PPG Timer

This section describes program examples of an 8-/16-bit PPG timer.

Program Example for 8-/16-bit PPG Timer

- **Processing specification**
  
  "L" width outputs a 100 μs and "H" width outputs a 200 μs pulse waveform in the 8-bit PPG output 2 ch independent operation mode.
  
  - Use PPG0 to output a pulse to PPG00 and PPG01 pins.
  - The count clock is set to 16 division (φ/16) of 16 MHz. (φ: Machine clock)
  - The PPG0 reload registers (PRLLO, PRLH0) setup value appears as follows.
    
    \[
    \text{PRLH0} = 200 \text{μs} \times 1\text{μs}^{-1} = 199 \text{ (C7H)}
    \]
    
    \[
    \text{PRLL0} = 100 \text{μs} \times 1\text{μs}^{-1} = 99 \text{ (63H)}
    \]

- **Coding example**
  
  ```
  ICRO4  EQU 0000B4H ;PPG0 Interrupt control register
  PPGC0  EQU 000044H ;PPG0 operation mode control register
  PPGC1  EQU 000045H ;PPG1 operation mode control register
  PPGE0  EQU 000046H ;PPG01 output control register
  PRL0   EQU 000040H ;PPG0 reload registers
  PENO   EQU PPGC0:7 ;PPG0 operation enable bit
  PUF0   EQU PPGC0:3 ;PPG0 underflow generation flag bit
  ;----------Main program-----------------------------------------------
  CODE   CSEG
  START:
  ; : ;Stack pointer (SP)
  ;Already initialized
  AND CCR,#0BFH ;Interrupt is disabled
  MOV I:ICR04,#00H ;Interrupt levels 0 (strength)
  MOVW I:PRL0,#C763H ;Sets L width=100μs, H width=200μs
  MOV I:PPGC0,#00110001B;Flag area, interrupt enabled,
  ;PPG00 terminal output enabled, PPG0 operation
  ;disabled
  MOV I:PPGC1,#00000001B;8-bit PPG output 2-channel independent
  ;operation mode selected,
  ;PPG1 operation disabled
  MOV I:PPGOE,#00010001B;Count clock=1μs,
  ;Set PPG01 terminal output to enabled
  SETB I:PENO ;PPG0 enabling operations
  MOV ILM,#07H ;Sets ILM in PS to level 7
  OR CCR,#40H ;Interruption is enabled
  ;---------- Interruption program--------------------------------------
  WARI:
  CLR I:PUF0 ;underflow generation flag is cleared
  •
  User processing
  •
  ```
CHAPTER 10 8-/16-BIT PPG TIMER

RETI ;Returning from interrupt processing
CODE ENDS

;----------Vector Settings--------------------------------------------------
VECT CSEG ABS=0FFH
ORG 0FFB0H ;vector set in interrupt number #19(13H)
DSL WARI
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START
CHAPTER 11
8-/16-BIT UP/DOWN COUNTER/TIMER

This chapter describes 8-/16-bit up/down counter/timer functions and operations.

11.1 8-/16-bit Up/Down Counter/Timer Overview
11.2 8-/16-bit Up/Down Counter/Timer 0 Block Diagram
11.3 8-/16-bit Up/Down Counter/Timer 1 Block Diagram
11.4 8-/16-bit Up/Down Counter/Timer Configuration
11.5 8-/16-bit Up/Down Counter/Timer Interrupt
11.6 8-/16-bit Up/Down Counter/Timer Operating Modes
11.7 8-/16-bit Up/Down Counter/Timer Count Mode
11.8 Operation of Timer Mode
11.9 Up/Down Count Mode Operation
11.10 Phase Differential Count Mode Operation
11.11 Reload/Compare Function Operation
11.12 Reload and Clear Timing
11.13 ZIN Pin Clear/Gate Function Operation
11.14 Precautions for 8-/16-bit Up/Down Counter/Timer
11.15 8-/16-bit Up/Down Counter/Timer Program Example
11.1 8-/16-bit Up/Down Counter/Timer Overview

- The 8-/16-bit up/down counter/timer has the following count modes: timer mode, up/down count mode and phase differential count mode.
- It can be used as an 8-bit, 2-channel or 16-bit 1-channel up/down counter/timer.

### 8-/16-bit up/down counter / timer functions

8-16-bit up/down counter/timer functions are shown in Table 11.1-1.

#### Table 11.1-1 8-/16-bit up/down counter / timer functions

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating mode</td>
<td>• 8-bit 2-channel mode</td>
</tr>
<tr>
<td></td>
<td>• 16-bit 1-channel mode</td>
</tr>
<tr>
<td>Count mode</td>
<td>4 types of count mode can be selected.</td>
</tr>
<tr>
<td></td>
<td>• Timer mode</td>
</tr>
<tr>
<td></td>
<td>• Up/down count mode</td>
</tr>
<tr>
<td></td>
<td>• Phase differential count mode (2 multiplier)</td>
</tr>
<tr>
<td></td>
<td>• Phase differential count mode (4 multiplier)</td>
</tr>
<tr>
<td>Count Clock (timer mode)</td>
<td>2 types of count clock can be selected.</td>
</tr>
<tr>
<td></td>
<td>• $\phi/2$ ($\phi$: machine clock frequency)</td>
</tr>
<tr>
<td></td>
<td>• $\phi/8$</td>
</tr>
<tr>
<td>Selection of detection edges (for up/down count mode)</td>
<td>Detection edge for external pin input signals can be selected.</td>
</tr>
<tr>
<td></td>
<td>• Edge detection is disabled.</td>
</tr>
<tr>
<td></td>
<td>• Falling edge detection</td>
</tr>
<tr>
<td></td>
<td>• Rising edge detection</td>
</tr>
<tr>
<td></td>
<td>• Both edge detection</td>
</tr>
<tr>
<td>ZIN pin function</td>
<td>2 types of functions can be selected.</td>
</tr>
<tr>
<td></td>
<td>• Counter Clear function</td>
</tr>
<tr>
<td></td>
<td>• Gate function</td>
</tr>
<tr>
<td>Compare/reload function</td>
<td>Operation is possible by combining the compare and reload functions.</td>
</tr>
<tr>
<td></td>
<td>• Compare function (interrupt request is generated and counter is cleared by comparing matches.)</td>
</tr>
<tr>
<td></td>
<td>• Reload function (interrupt request is generated and reloaded by underflow)</td>
</tr>
<tr>
<td></td>
<td>• Compare/reload function (interrupt request is generated and the counter is cleared by comparing matches, and interrupt request is generated and reloaded by underflow)</td>
</tr>
<tr>
<td></td>
<td>• Compare/reload is disabled.</td>
</tr>
<tr>
<td>Count direction</td>
<td>The count direction just prior can be identified using the up/down flag.</td>
</tr>
<tr>
<td>Interrupt request</td>
<td>Interrupt requests can be generated under the following conditions.</td>
</tr>
<tr>
<td></td>
<td>• Compare match</td>
</tr>
<tr>
<td></td>
<td>• Underflow or overflow</td>
</tr>
<tr>
<td></td>
<td>• Count direction change</td>
</tr>
</tbody>
</table>

**Reference:** The phase differential count mode is suitable for counting encoders, such as motors, and accurate detection of the rotation angle and counting of the rotation number are possible by inputting the A phase, B phase, Z-phase output of the encoder to the AIN, BIN, and ZIN pins respectively, and the rotation direction can easily be detected.
11.2 8-/16-bit Up/Down Counter/Timer 0 Block Diagram

A block diagram of the 8-16-bit up/down counter/timer 0 is shown in Figure 11.2-1.

**Details of Pins and Interrupt Numbers**

*(8-/16-bit up/down counter/timer 0)*

- AIN0 pin: P24/AIN0
- BIN0 pin: P25/BIN0
- ZIN0 pin: P26/ZIN0
- Compare match interrupt number: #21 (15H)
- Underflow/overflow interrupts, interrupt number of count direction change interrupt: #22 (16H)
CHAPTER 11  8-/16-BIT UP/DOWN COUNTER/TIMER

- **Up/down count register 0 (UDCR0)**

  This 8-bit up/down counter counts down using the count clock, counts up using the signal input to the AIN0 pin or counts down using the signal input to the BIN0 pin. Channel 0 and channel 1 can jointly be used as one 16-bit channel.

- **Reload compare register 0 (RCR0)**

  An 8-bit register for setting reload values or compare values in up/down count register 0. Channel 0 and channel 1 can jointly be used as one 16-bit channel.

- **Counter control register 0 (CCR0)**

  It can also select 8-bit or 16-bit operating mode, count mode, count clock and count edge, enable/disable count direction change interrupts, check count direction change generation, transfer data from the RCR0 register to the UDCR0 register, enable/disable compare match clearing, enable/disable the reload function, select the ZIN0 pin function and detection conditions.

- **Counter status register 0 (CSR0)**

  It makes it possible to enable/disable overflow/underflow and compare match interrupts, display previous count operation status and start/stop up/down counter/timer.

- **Reload controller circuit**

  It loads and reloads values set in the reload compare register 0 (RCR0) into the up/down count register 0 (UDCR0).

- **Prescaler**

  Machine clock frequency is divided to supply the count clock used in the timer mode. Select one of two division ratios.

- **Edge/level detection circuit**

  Detects the valid edge/level in signal input to the ZIN0 pin.

- **Up/down count selector**

  This circuit selects prescaler output clock or the timing of AIN0/BIN0 pin input signal count. It also handles count up and count down switching.

- **Compare control circuit**

  The up/down count register 0 (UDCR0) values and reload compare register 0 (RCR0) set values are compared to detect matches.

- **Counter clear circuit**

  It clears the up/down count register using software and the up/down count register by valid edge inputs to the ZIN0 pin and compare matches.

- **Edge detection circuit**

  In up/down count mode, the valid edge of signals input to AIN0/BIN0 pins are detected.
11.3 8-/16-bit Up/Down Counter/Timer 1 Block Diagram

A block diagram of the 8-16-bit up/down counter/timer 1 is shown in Figure 11.3-1.

- 8-/16-bit up/down counter/timer 1 block diagram

Figure 11.3-1 8-/16-bit up/down counter/timer 1 block diagram

- Details of Pins and Interrupt Numbers

(8-/16-bit up/down counter/timer 1)

- AIN1 pin: P50/AIN1
- BIN1 pin: P51/BIN1
- ZIN1 pin: P52/ZIN1
- Compare match interrupt number: #29 (1D_H)
- Underflow/overflow interrupts, interrupt number of count direction change interrupt: #30 (1E_H)
CHAPTER 11 8-/16-BIT UP/DOWN COUNTER/TIMER

- **Up/down count register 1 (UDCR1)**
  
  This 8-bit up/down counter counts down using the count clock, counts up using the signal input to the AIN1 pin or counts down using the signal input to the BIN1 pin. Channel 0 and channel 1 can jointly be used as one 16-bit channel.

- **Reload compare register 1 (RCR1)**
  
  An 8-bit register for setting reload values or compare values in up/down count register 1. Channel 0 and channel 1 can jointly be used as one 16-bit channel.

- **Counter control register 1 (CCR1)**
  
  It can also select count mode, count clock and count edge, enable/disable count direction change interrupts, check count direction change generation, transfer data from the RCR1 register to the UDCR1 register, enable/disable compare match clearing, enable/disable the reload function, select the ZIN1 pin function and detection conditions.

- **Counter status register 1 (CSR1)**
  
  It makes it possible to enable/disable overflow/underflow and compare match interrupts, display previous count operation status and start/stop up/down counter/timer.

- **Reload controller circuit**
  
  It loads and reloads set values of the reload compare register 1 (RCR1) into the up/down count register 1 (UDCR1).

- **Prescaler**
  
  Machine clock frequency is divided to provide the count clock used in the timer mode. Select one of two division ratios.

- **Edge/level detection circuit**
  
  Detects the valid edge/level in signal input to the ZIN1 pin.

- **Up/down count selector**
  
  This circuit selects prescaler output clock or the timing of AIN1/BIN1 pin input signal count. It also handles count up and count down switching.

- **Compare control circuit**
  
  The up/down count register 1 (UDCR1) value and reload compare register 1 (RCR1) set values are compared to detect matches.

- **Counter clear circuit**
  
  It clears the up/down count register using software and the up/down count register by valid edge inputs to the ZIN1 pin and compare matches.

- **Edge detection circuit**
  
  In up/down count mode, the valid edge of signals input to AIN1/BIN1 pins are detected.
11.4 8-/16-bit Up/Down Counter/Timer Configuration

Detailed description of 8-/16-bit up/down counter/timer pins, interrupt factors and registers are provided.

- **8-/16-bit up/down counter/timer pin**
  Pins that use the 8-/16-bit up/down counter/timer double as general-purpose I/O port.
  Table 11.4-1 shows pin functions and settings required for using the 8-/16-bit up/down counter/timer.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting of the pins required for use of up/down counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 0</td>
<td>P24/AIN0</td>
<td>General-purpose I/O port, A phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
<tr>
<td></td>
<td>P25/BIN0</td>
<td>General-purpose I/O port, B phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
<tr>
<td></td>
<td>P26/ZIN0</td>
<td>General-purpose I/O port, Z phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
<tr>
<td>Channel 1</td>
<td>P50/AIN1</td>
<td>General-purpose I/O port, A phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
<tr>
<td></td>
<td>P51/BIN1</td>
<td>General-purpose I/O port, B phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
<tr>
<td></td>
<td>P52/ZIN1</td>
<td>General-purpose I/O port, Z phase input</td>
<td>Set to the dedicated input port at port direction register (DDR).</td>
</tr>
</tbody>
</table>

- **Block Diagram of Pins**

  Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

- **Interrupt requests are generated by the 8-/16-bit up/down counter/timer**
  The following interrupt requests are generated in the 8-/16-bit up/down counter/timer.

  - **Overflow (underflow) interrupt**
    When overflow/underflow interrupt requests are enabled (CSR: UDIE=1), an overflow (underflow) interrupt request is generated when an overflow/underflow occurs in the up/down count register (UDCR).

  - **Compare match interrupt**
    When the compare interrupt requests are enabled (CSR:CITE=1), a compare match interrupt request occurs when the value of the up/down count register (UDCR) matches the value set in the reload compare register (RCR) during count up.

  - **Count direction change interrupt**
    When count direction change interrupt requests are enabled (CCR:CFIE=1), count direction changes from up count to down count, or from down count to up count during counting in the up/down count register, the count direction change interrupt request is generated.
11.4.1 Counter Status Register 0 (CSR0)

Counter status register 0 (CSR0) sets up/down count register 0 (UDCR0) overflow and underflow, checks compare match generation, starts/stops up/down counter/timer, etc.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Reset Value</th>
<th>Readable and writable</th>
<th>Read only</th>
<th>Description</th>
</tr>
</thead>
</table>
| 0   | UDFF Up/down flag                                | 00000000 B   | Yes                   | No        | Underflow happened
|     |                                                  |              |                       | Yes       | Clearing of this UDFF bit                                                  |
| 1   | OVFF Overflow happened flag                      |              | Yes                   | No        | Overflow happened
|     |                                                  |              |                       | Yes       | Clearing of this OVFF bit                                                  |
| 2   | CMPF Compare match happened flag                 |              | Yes                   | No        | Compare match
|     |                                                  |              |                       | Yes       | Clearing of this CMPF bit                                                  |
| 3   | UDIE Overflow/underflow interrupt enable bit      |              | Yes                   | No        | Overflow/underflow interrupt enable
|     |                                                  |              |                       | Yes       | Overflow/underflow interrupt is disabled                                   |
| 4   | CITE Compare interrupt enabling bit              |              | Yes                   | No        | Compare interrupt enabling
|     |                                                  |              |                       | Yes       | Compare match interrupts is disabled                                       |
| 5   | CSTR Up/down counter/timer start bit             |              | Yes                   | No        | Up/down counter/timer start
|     |                                                  |              |                       | Yes       | Up/down counter/timer operation is stopped                                 |

Figure 11.4-1 Counter status register 0 (CSR0)
### Table 11.4-2 Counter status register 0 (CSR0) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit1 bit0 UDF1, UDF0: Up/down flag | The counting direction (count up or down) just before the up/down count register 0 (UDCR0) is indicated.  
• Read is only possible.  
• This will be re-written every time it is counted.  
• This UDF bit is valid under all count modes.  
• If the current rotation direction for the motor control needs to be known, the rotation direction can be found by checking this UDF bit.  
**Note:**  
If the direction diversion period is short and is continuously generated, the direction indicated by the flag after direction diversion may return to the one before direction diversion, and may be the same direction. |
| bit2 UDFF: Underflow happened flag | This indicates that underflow of the up/down count register 0 (UDCR0) is generated.  
**When the bit is set to “1”:** When overflow/underflow interrupts enabling bit is set to “1” (CSR0:UDIE=1), an interrupt request is generated.  
When the reload enabling bit is set to "1" (CCR0:RLDE = 1), the set value for the reload compare register 0 (RCR0) is transferred to the up/down count register 0.  
**When the bit is set to “0”:** The bit is cleared.  
"1" cannot be set. |
| bit3 OVFF: Overflow generating flag | This indicates that overflow of the up/down count register 0 (UDCR0) is generated.  
**When the bit is set to “1”:** When the overflow/underflow interrupt enabling bit is set to "1" (CSR0:UDIE = 1), an interrupt request is generated.  
**When the bit is set to “0”:** The bit is cleared.  
"1" cannot be set. |
| bit4 CMPF: Compare match generating flag | This indicates that the up/down count register 0 (UDCR0) value matches the set value of the reload compare register 0 through count up (generation of compare match).  
**When the bit is set to “1”:** When the compare interrupt enabling bit is set to "1" (CSR0:CITE = 1), an interrupt request is generated. When the compare match counter clear enabling bit is set to "1" (CCR0:UCRE = 1), the up/down count register 0 will be cleared.  
**When the bit is set to “0”:** The bit is cleared.  
"1" cannot be set.  
• This CMPF bit does not generate compare match interrupts under timer mode.  
• For compare match through reload, "1" is set in cases where it has already been matched when counting was initiated. |
| bit5 UDIE: Overflow/Underflow interrupt enable bit | Interruption due to generation of overflow or underflow is enabled or disabled.  
• Overflow will not be generated under timer mode. |
| bit6 CITE: Compare interrupt enable bit | Interruption due to generation of compare match is enabled or disabled.  
• This CITE bit is invalid under timer mode. |
| bit7 CSTR: Up/down counter/timer activation bit | Counting of up/down count register 0 (UDCR0) is started or stopped. |

**Note:**  
When 8-bit 2-channel mode is set (CCR0:M16E = 0): The counter status register 0 (CSR0) functions as the status register of the up/down counter/timer channel 0.  
When 16-bit 1-channel mode is set (CCR0:M16E = 1): Only counter status register 0 (CSR0) functions as the status register of the 16-bit up/down count register that combines channels 0 and 1.
### 11.4.2 High-order Counter Control Register 0 (CCR0:H)

High-order counter control register 0 (CCR0:H) selects up/down counter/timer 0 operating mode and count mode, count clock and count edge, enables/disables count direction change interrupts and checks count direction change generation.

#### Counter control register 0 (CCR0)

![Counter control register 0 (CCR0)](image_url)

- **bit9 bit8**: Count edge selection bit
  - 0 0: Edge detection is disabled.
  - 0 1: Falling edge detection
  - 1 0: Rising edge detection
  - 1 1: Double-edge detection

- **bit11 bit10**: Count mode selection bit
  - 0 0: Timer mode
  - 0 1: Up/down count mode
  - 1 0: Phase differential mode multiply by 2
  - 1 1: Phase differential mode multiply by 4

- **bit12**: Count clock selection bit
  - 0: \( \phi/2 \)
  - 1: \( \phi/8 \)
  \( \phi \): machine clock frequency

- **bit13**: Count direction conversion interrupt enabling bit
  - 0: Count direction change interrupt requests are disabled
  - 1: Count direction change interrupt requests are enabled

- **bit14**: Count direction change flag
  - When reading is performed
    - 0: Have not been turn direction
    - 1: Have been turn direction twice or more
  - When write is specified
    - 0: Clearing of this CDCF bit
    - 1: No effect

- **bit15**: Operation mode select bit
  - 0: 8-bit 2-channel mode
  - 1: 16-bit 1-channel mode
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit9 bit8        | CES1, CES0: Count edge selection bits When the up/down count mode is set (CCR0:CMS1 and CMS0 = "01B"), the edges counted by the AIN0 and BIN0 pins are set.  
• This CES bit is valid only when up/down count mode is set.                                                                                   |
| bit11 bit10      | CMS1, CMS0: Count mode selection bits Setting the count mode.  
• Timer mode, up/down count mode, and phase differential count mode (2 multiplier/4 multiplier) can be set.                                        |
| bit12            | CLKS: Count clock selection bits Count clock using prescaler is set when the timer mode is set (CCR0:CMS1 and CMS0 = "00B").  
This CLKS bit is valid only under timer mode.                                                                                                   |
| bit13            | CFIE: Count direction change interrupt enable bit Interruption by generation of the count direction change is enabled or disabled.  
When the bit is set to "1": Interrupt request is generated when conversion in the count direction is generated (CCR0:CDCF = 1).  
• This CFIE bit is invalid under timer mode.                                                                                                   |
| bit14            | CDCF: Count direction change flag Whether the count direction is converted or not is indicated.  
• "1" will be set when the count direction is changed from up to down (or from down to up) while counting the up/down count register.  
When the bit is set to "1": When the count direction change interrupt enabling bit is set to "1", an interrupt request is generated.  
When the bit is set to "0": The bit is cleared.  
"1" cannot be set.  
This CDCF bit is valid under all count modes.                                                                                                   |
| bit15            | M16E: Operation mode selection bits This bit selects the operating mode.  
When the bit is set to "0": 8-bit 2-channel mode will be set.  
When the bit is set to "1": 16-bit 1-channel mode will be set.  
When 16-bit 1-channel mode is set: Only the counter status register (CSR0) and counter control register (CCR0) at the channel 0 side are valid.  
Note: Set up this M16E bit after stop the up/down counter/timer operation (CSR0 and CSR1:CSTR = 0).                                      |

Note: If set to 8-bit, 2-channel mode (CCR0:M16E=0), counter control register 0 (CCR0) functions as a control register for up/down count register 0 (UDCR0).  
If set to 16-bit, 1-channel mode (CCR0:M16E=0), counter control register 0 (CCR0) functions as a control register for channel 0 and 1 and their 16-bit up/down count registers.
11.4.3 Low-order Counter Control Register 0 (CCR0:L)

The low-order counter control register 0 (CCR0:L) is used to select the edge/level that operates the ZIN0 pin of up/down counter/timer 0.

### Low-order counter control register 0 (CCR0:L)

#### Figure 11.4-3  Low-order counter control register 0 (CCR0: L)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>CTUT Counter write bit</td>
<td>0: No effect</td>
</tr>
<tr>
<td>6</td>
<td>UCRE Compare match counter clear enabling bit</td>
<td>0: No effect</td>
</tr>
<tr>
<td>5</td>
<td>RLDE Reload enabling bit</td>
<td>0: Reload operation is disabled</td>
</tr>
<tr>
<td>4</td>
<td>UDCR Software counter clear bit</td>
<td>0: up/down count register 0 (UDCR0) is cleared</td>
</tr>
<tr>
<td>3</td>
<td>CGE2 Counter clear function/count level</td>
<td>0: Counter clear function</td>
</tr>
<tr>
<td>2</td>
<td>CGSC Counter clear function/gate function selection bit (ZIN0 terminal function)</td>
<td>0: Counter clear function</td>
</tr>
<tr>
<td>1</td>
<td>RLDE Reload enabling bit</td>
<td>0: Reload operation is disabled</td>
</tr>
<tr>
<td>0</td>
<td>Bit 0</td>
<td>Reset value X 0 0 0 0 0 0 0 B</td>
</tr>
</tbody>
</table>

**Counter clear function/count level**

<table>
<thead>
<tr>
<th>GE1</th>
<th>GE0</th>
<th>When counter clear function is selected</th>
<th>When gate function is selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Edge detection is disabled.</td>
<td>Level detection inhibited</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Cleared when the falling edge is detected</td>
<td>&quot;L&quot; level interval count enabled</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Cleared when the rising edge is detected</td>
<td>&quot;H&quot; level interval count enabled</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Setting disabled</td>
<td>Setting disabled</td>
</tr>
</tbody>
</table>

**Compare match counter clear enabling bit**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Clearing of up/down count register 0 (UDCR0) through compare match is disabled</td>
</tr>
<tr>
<td>1</td>
<td>Clearing of up/down count register 0 (UDCR0) through compare match is enabled</td>
</tr>
</tbody>
</table>

**Counter write bit**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No effect</td>
</tr>
<tr>
<td>1</td>
<td>Transfer data from the RCR0 register to the UDCR0 register</td>
</tr>
</tbody>
</table>

**Legend**

- **R/W**: Readable and Writable
- **X**: Undefined
- **-**: Unused
- ****: Reset value

---

390
Table 11.4-4 Function of the counter control register 0 lower (CCR0:L)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0 CGE1, CGE0:</td>
<td>Counter clear edge/count level selection bit</td>
</tr>
<tr>
<td>bit1</td>
<td>Functions</td>
</tr>
<tr>
<td>bit2 CGSC:</td>
<td>Counter clear function/gate function selection bit</td>
</tr>
<tr>
<td>bit3 UDCC:</td>
<td>Software counter clear bit</td>
</tr>
<tr>
<td>bit4 RLDE:</td>
<td>Reload enabling bit</td>
</tr>
<tr>
<td>bit5 UCRE:</td>
<td>Compare match counter clear enabling bit</td>
</tr>
<tr>
<td>bit6 CTUT:</td>
<td>Counter write bit</td>
</tr>
<tr>
<td>bit7 Undefined bits</td>
<td></td>
</tr>
</tbody>
</table>

**Bit0**
- **CGE1, CGE0:** Counter clear edge/count level selection bit
- Functions:
  - Edge or level to be detected by the ZIN0 pin is set.
  - When the edge/level that has been set is detected, the up/down count register 0 (UDCR0) is cleared or counting is enabled/disabled in accordance with function of the ZIN0 pin that has been set by the counter clear function/gate function selection bit (CCR0:CGSC).
  - **When counter clear function is selected (CCR0:CGSC = 0):** When the set edge is detected by the ZIN0 pin, the up/down count register 0 will be cleared.
  - **When gate function is selected (CCR0:CGSC = 1):** While the set level signal is detected by the ZIN0 pin, counting of the up/down count register 0 is enabled.

**Bit2**
- **CGSC:** Counter clear function/gate function selection bit
- Function of the ZIN0 pin is set.
  - **When the bit is set to "0":** When the edge that has been set by the CGE0 and CGE1 bits are detected by the ZIN0 pin, the up/down count register 0 (UDCR0) will be cleared.
  - **When the bit is set to "1":** Counting for the up/down count register 0 is only enabled when the level signal that has been set by the CGE0 and CGE1 bits is detected by the ZIN0 pin.

**Bit3**
- **UDCC:** Software counter clear bit
- Up/down count register 0 (UDCR 0) will be cleared.
  - **When the bit is set to "0":** Up/down count register will be cleared to "00H".
  - **When the bit is set to "1":** No effect.
  - **Read:** "0" is always read.

**Bit4**
- **RLDE:** Reload enabling bit
- Operation of the reload function is enabled or disabled.
  - **When the bit is set to "1":** When the underflow generation flag bit is set (CSR0:UDFF = 1), the value set for the reload compare register 0 (RCR0) is transferred to the up/down count register 0 (UDCR0).

**Bit5**
- **UCRE:** Compare match counter clear enabling bit
- Clearing of up/down count register 0 (UDCR0) through compare match is enabled or disabled.
  - This will not affect the following functions other than clearing through compare generation.
    - Clear by reset input
    - Clearing through edge input from the ZIN0 pin (Set up when CCR0:CGSC = 0; CCR0:CGE1 and CGE0 = "01B" and "10B")
    - Cleared by setting the software counter clear bit of the counter control register 0 to "0" (CCR0:UDCC = 0)

**Bit6**
- **CTUT:** Counter write bit
- The value set in the reload compare register 0 (RCR0) is transferred to the up/down count register 0 (UDCR0).
  - **When the bit is set to "1":** The value set in the reload compare register 0 is transferred to the up/down count register 0.
  - **When the bit is set to "0":** No effect.
  - **Read:** "0" is always read.
  - **Note:** Do not set this CTUT bit to "1" while operating the up/down counter/timer.

**Bit7**
- **Undefined bits**
- **Read:** The value is undefined.
- **Write:** No effect

**Note:** If set to 8-bit, 2-channel mode (CCR0:M16E=0), counter control register 0 (CCRO) functions as a control register for up/down count register.
If set to 16-bit, 1-channel mode (CCR0:M16E=1), counter control register 0 (CCRO) functions as a control register for channel 0 and 1 and their 16-bit up/down count registers.
11.4.4 Counter Status Register 1 (CSR1)

Counter status register 1 (CSR1) sets up/down count register 1 (UDCR1) overflow and underflow, checks compare match generation, starts/stops up/down counter/timer, etc.

■ Counter status register 1 (CSR1)

Figure 11.4-4 Counter status register 1 (CSR1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>CSTR</td>
<td>Readable and writable</td>
<td>Up/down counter/timer start bit</td>
</tr>
<tr>
<td>6</td>
<td>CITE</td>
<td>Readable and writable</td>
<td>Compare interrupt enabling bit</td>
</tr>
<tr>
<td>5</td>
<td>UDIE</td>
<td>Readable and writable</td>
<td>Overflow/underflow interrupt enable bit</td>
</tr>
<tr>
<td>4</td>
<td>CMPF</td>
<td>Readable and writable</td>
<td>Compare match happened flag</td>
</tr>
<tr>
<td>3</td>
<td>OVFF</td>
<td>Readable and writable</td>
<td>Overflow happened flag</td>
</tr>
<tr>
<td>2</td>
<td>UDFF</td>
<td>Readable and writable</td>
<td>Underflow happened flag</td>
</tr>
<tr>
<td>1</td>
<td>UDF0</td>
<td>Readable and writable</td>
<td>Up/down flag</td>
</tr>
<tr>
<td>0</td>
<td>UDF1</td>
<td>Readable and writable</td>
<td>Up/down flag</td>
</tr>
</tbody>
</table>

Reset value: 00000000

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0</td>
<td>Up/down counter/timer operation is stopped.</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>Up/down counter/timer operation is started</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>Compare match interrupts is disabled</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>Compare match interrupts is enabled</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>Clearing of this CMPF bit</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Clearing of this OVFF bit</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Clearing of this UDFF bit</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Underflow is generated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No effect</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No effect</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No effect</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No effect</td>
</tr>
</tbody>
</table>

R/W: Readable and writable
R: Read only
\( R: \) Reset value
Table 11.4-5 Counter status register 1 (CSR1) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit1, bit0</td>
<td><strong>UDF1, UDF0:</strong> Up/down flag</td>
</tr>
<tr>
<td></td>
<td>The counting direction (count up or down) just before the up/down count register 1 (UDCR1) is indicated.</td>
</tr>
<tr>
<td></td>
<td>• Read is only possible.</td>
</tr>
<tr>
<td></td>
<td>• This will be re-written every time it is counted.</td>
</tr>
<tr>
<td></td>
<td>• This UDF bit is valid under all count modes.</td>
</tr>
<tr>
<td></td>
<td>• If the current rotation direction for the motor control needs to be known, the rotation direction can be found by checking this UDF bit.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td></td>
<td>If the direction diversion period is short and is continuously generated, the direction indicated by the flag after direction diversion may return to the one before direction diversion, and may be the same direction.</td>
</tr>
<tr>
<td>bit2</td>
<td><strong>UDFF:</strong> Underflow happened flag</td>
</tr>
<tr>
<td></td>
<td>This indicates that underflow of the up/down count register 1 (UDCR1) is generated.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”1″:</strong> When the overflow/underflow interrupt enabling bit is set to ”1″ (CSR0:UDIE = 1), an interrupt request is generated.</td>
</tr>
<tr>
<td></td>
<td>When the reload enabling bit is set to ”1″ (CCR0:RLDE = 1), the set value for the reload compare register 1 (RCR1) is transferred to the up/down count register 1.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”0″:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td>”1″ cannot be set.</td>
</tr>
<tr>
<td>bit3</td>
<td><strong>OVFF:</strong> Overflow happened flag</td>
</tr>
<tr>
<td></td>
<td>This indicates that overflow of the up/down count register 1 (UDCR1) is generated.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”1″:</strong> When the overflow/underflow interrupt enabling bit is set to ”1″ (CSR1:UDIE = 1), an interrupt request is generated.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”0″:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td>”1″ cannot be set.</td>
</tr>
<tr>
<td></td>
<td>• Overflow will not be generated under timer mode.</td>
</tr>
<tr>
<td>bit4</td>
<td><strong>CMPF:</strong> Compare match happened flag</td>
</tr>
<tr>
<td></td>
<td>This indicates that the up/down count register 1 (UDCR1) value matches the set value of the reload compare register 1 through count up (generation of compare match).</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”1″:</strong> When the compare interrupt enabling bit is set to ”1″ (CSR1:CITE = 1), an interrupt request is generated. When the compare match counter clear enabling bit is set to ”1″ (CCR1: UCRE = 1), the up/down count register 1 will be cleared.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to ”0″:</strong> The bit is cleared. ”1″ cannot be set.</td>
</tr>
<tr>
<td></td>
<td>• This CMPF bit does not generate compare match interrupts under timer mode.</td>
</tr>
<tr>
<td></td>
<td>• For compare match through reload, ”1″ is set in cases where it has already been matched when counting was initiated.</td>
</tr>
<tr>
<td>bit5</td>
<td><strong>UDIE:</strong> Overflow/underflow interrupt enable bit</td>
</tr>
<tr>
<td></td>
<td>Interruption due to generation of overflow or underflow is enabled or disabled.</td>
</tr>
<tr>
<td></td>
<td>• Overflow will not be generated under timer mode.</td>
</tr>
<tr>
<td>bit6</td>
<td><strong>CITE:</strong> Compare interrupt enable bit</td>
</tr>
<tr>
<td></td>
<td>Interruption due to generation of compare match is enabled or disabled.</td>
</tr>
<tr>
<td></td>
<td>• This CITE bit is invalid under timer mode.</td>
</tr>
<tr>
<td>bit7</td>
<td><strong>CSTR:</strong> Up/down counter/timer activation bit</td>
</tr>
<tr>
<td></td>
<td>Counting of the up/down count register 1 (UDCR1) is started or stopped.</td>
</tr>
</tbody>
</table>

**Note:** Counter status register 1 is enabled if set to 8-bit 2-channel mode (CCR0:M16E=0) when it functions as a status register for channel 1 up/down counter/timer.
11.4.5 High-order Counter Control Register 1 (CCR1:H)

High-order counter control register 1 (CCR1:H) selects up/down counter/timer 1 count clock and count mode and count edge, enables/disables count direction change interrupts and checks count direction change generation.

### High-order counter control register 1 (CCR1: H)

*Figure 11.4-5 High-order counter control register 1 (CCR1: H)*

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>CDCF</td>
<td>Count direction change flag bit</td>
</tr>
<tr>
<td>14</td>
<td>CFIE</td>
<td>Count direction conversion interrupt enabling bit</td>
</tr>
<tr>
<td>13</td>
<td>CLKS</td>
<td>Count clock selection bit</td>
</tr>
<tr>
<td>12</td>
<td>CMS1</td>
<td>Count mode selection bit</td>
</tr>
<tr>
<td>11</td>
<td>CMS0</td>
<td>Count edge selection bit</td>
</tr>
<tr>
<td>9</td>
<td>CES1</td>
<td>Reset value</td>
</tr>
<tr>
<td>8</td>
<td>CES0</td>
<td>Reset value</td>
</tr>
</tbody>
</table>

- **R/W**: Readable and Writable
- **X**: Undefined
- **-**: Unused
- **□**: Reset value

### Figure 11.4-5 High-order counter control register 1 (CCR1: H)

- **CDCF**: Count direction change flag bit
  - 0: Have not been turn direction
  - 1: Have been turn direction twice or more
- **CFIE**: Count direction conversion interrupt enabling bit
  - 0: Count direction change interrupt requests are disabled
  - 1: Count direction change interrupt requests are enabled
- **CLKS**: Count clock selection bit
  - 0: \( \phi/2 \)
  - 1: \( \phi/8 \)
- **CMS1** and **CMS0**: Count mode selection bit
  - 00: Timer mode
  - 01: Up/down count mode
  - 10: Phase differential mode multiply by 2
  - 11: Phase differential mode multiply by 4
- **CES1** and **CES0**: Count edge selection bit
  - 00: Edge detection is disabled
  - 01: Falling edge detection
  - 10: Rising edge detection
  - 11: Double - edge detection
### Table 11.4-6 Function of the counter control register 1 upper (CCR1:H)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit9, bit8 | CES1, CES0: Count edge selection bits | When the up/down count mode is set (CCR1:CMS1 and CMS0 = "01B"), the edges counted by the AIN1 and BIN1 pins are set.  
  - This CES bit is valid only when up/down count mode is set. |
| bit11, bit10 | CMS1, CMS0: Count mode selection bits | Setting the count mode.  
  - Timer mode, up/down count mode, and phase differential count mode (2 multiplier/4 multiplier) can be set. |
| bit12 | CLKS: Count clock selection bits | Count clock using prescaler is set when the timer mode is set (CCR1:CMS1 and CMS0 = "00B").  
  - This CLKS bit is valid only under timer mode.  
  **Note:**  
  Set up count clock selection after stop up/down counter/timer operation (CSR0:CSTR = 0). Do not switch during operation. |
| bit13 | CFIE: Count direction change interrupt enable bit | Interruption by generation of the count direction change is enabled or disabled.  
  **When the bit is set to "1":** Interrupt request is generated when conversion in the count direction is generated (CCR1:CDCF = 1).  
  - This CFIE bit is invalid under timer mode. |
| bit14 | CDCF: Count direction change flag bit | Whether the count direction of the up/down count register is changed or not is indicated.  
  "1" will be set when the count direction is changed from up to down (or from down to up) while counting the up/down count register.  
  **When the bit is set to "1":** When the count direction change interrupt enabling bit is set to "1", an interrupt request is generated.  
  **When the bit is set to "0":** The bit is cleared.  
  "1" cannot be set.  
  This CDCF bit is valid under all count modes. |
| bit15 | Undefined bits | Read: The value is undefined.  
  Write: No effect |

**Note:** Counter control register 1 is enabled if set to 8-bit 2-channel mode (CCR0:M16E=0) when it functions as a control register for the channel 1 up/down counter/timer.
11.4.6 Low-order Counter Control Register 1 (CCR1: L)

The low-order counter control register 1 (CCR1:L) is used to select the edge/level that operates the ZIN1 pin of up/down counter/timer 1.

- **Low-order counter control register 1 (CCR1:L)**

  ![Figure 11.4-6 Low-order counter control register 1 (CCR1:L)](image)

  - **CGSC**: Counter clear function/gate function select bit (select the ZIN1 terminal function and detection conditions)
    - **bit2**: Counter clear function
      - 0: Counter clear function disabled
      - 1: Counter clear function enabled
    - **bit3**: Gate function
      - 0: Gate function disabled
      - 1: Gate function enabled

  - **UDCC**: Software counter clear bit
    - 0: Clearing the up/down count register 1 (UDCR1) execute
    - 1: No effect

  - **RLDE**: Reload enabling bit
    - 0: Reload operation is disabled
    - 1: Reload operation is enabled

  - **UCRE**: Compare match counter clear enabling bit
    - 0: Clearing of up/down count register 1 (UDCR1) through compare match is disabled
    - 1: Clearing of up/down count register 1 (UDCR1) through compare match is enabled

  - **CTUT**: Counter write bit
    - 0: No effect
    - 1: Transfer data from the RCR1 register to the UDCR1 register

---

**Reset value**: X 0 0 0 0 0 0 0 a

---

**Legend**

- **R/W**: Readable and Writable
- **X**: Undefined
- **—**: Unused
- **: Reset value**

---

396
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit1, bit0</td>
<td>CGE1, CGE0: Counter clear edge/count level selection bit</td>
</tr>
<tr>
<td></td>
<td>Edge or level to be detected by the ZIN1 pin is set. When the edge/level that has been set is detected, the up/down count register 1 (UDCR1) is cleared or counting is enabled/disabled in accordance with function of the ZIN1 pin that has been set by the counter clear function/gate function selection bit (CCR1:CGSC).</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;When counter clear function is selected (CCR1:CGSC = 0)&gt;</strong> When the set edge is detected by the ZIN1 pin, the up/down count register 1 will be cleared. <strong>&lt;When gate function is selected (CCR1:CGSC = 1)&gt;</strong> While the set level signal is detected by the ZIN1 pin, counting of the up/down count register 1 is enabled.</td>
</tr>
<tr>
<td>bit2</td>
<td>CGSC: Counter clear function/gate function selection bit</td>
</tr>
<tr>
<td></td>
<td>Function of the ZIN1 pin is set. <strong>When the bit is set to &quot;0&quot;:</strong> When the edge that has been set by the CGE0 and CGE1 bits are detected by the ZIN0 pin, the up/down count register 1 (UDCR1) will be cleared. <strong>When the bit is set to &quot;1&quot;:</strong> Counting for the up/down count register 1 is only enabled when the level signal that has been set by the CGE0 and CGE1 bits is detected by the ZIN1 pin.</td>
</tr>
<tr>
<td>bit3</td>
<td>UDCC: Software counter clear bit</td>
</tr>
<tr>
<td></td>
<td>Up/down count register 1 (UDCR 1) will be cleared. <strong>When the bit is set to &quot;0&quot;:</strong> Up/down count register 1 will be cleared to &quot;00H&quot;. <strong>When the bit is set to &quot;1&quot;:</strong> No effect. <strong>Read:</strong> &quot;0&quot; is always read.</td>
</tr>
<tr>
<td>bit4</td>
<td>RLDE: Reload enabling bit</td>
</tr>
<tr>
<td></td>
<td>Operation of the reload function is enabled or disabled. <strong>When the bit is set to &quot;1&quot;:</strong> When the underflow generation flag bit is set (CSR1:UDFF = 1), the value set for the reload compare register 1 (RCR1) is transferred to the up/down count register 1 (UDCR1).</td>
</tr>
<tr>
<td>bit5</td>
<td>UCRE: Compare match software clear enabling bit</td>
</tr>
<tr>
<td></td>
<td>Clearing of up/down count register 1 (UDCR1) through compare match is enabled or disabled. <strong>This will not affect the following functions other than clearing through compare generation.</strong> - Clear by reset input - Clearing through edge input from the ZIN1 pin (Set up when CCR1:CGSC = 0; CCR1:CGE1 and CGE0 = &quot;01B&quot; and &quot;10B&quot;) - Cleared by setting the software counter clear bit of the counter control register 1 to &quot;0&quot; (CCR1:UDCC = 0)</td>
</tr>
<tr>
<td>bit6</td>
<td>CTUT: Counter write bit</td>
</tr>
<tr>
<td></td>
<td>The value set in the reload compare register 1 is transferred to the up/down count register 1 (UDCR1). <strong>When the bit is set to &quot;1&quot;:</strong> The value set in the reload compare register 1 is transferred to the up/down count register 1. <strong>When the bit is set to &quot;0&quot;:</strong> No effect. <strong>Read:</strong> &quot;0&quot; is always read. <strong>Note:</strong> Do not set this CTUT bit to &quot;1&quot; while operating the up/down counter/timer.</td>
</tr>
<tr>
<td>bit7</td>
<td>Undefined bits</td>
</tr>
<tr>
<td></td>
<td><strong>Read:</strong> The value is undefined. <strong>Write:</strong> No effect</td>
</tr>
</tbody>
</table>

**Note:** Counter control register 1 (CCR1) is enabled if set to 8-bit 2-channel mode (CCR0:M16E=0) when it functions as a control register for the channel 1 up/down counter/timer.
11.4.7 Up/Down Count Register (UDCR0, UDCR1)

Up/down count register (UDCR0, UDCR1) can be used either as an 8-bit, 2-channel or 16-bit, 1-channel up/down counter. Count up/down is started by the AIN pin, BIN pin or prescaler input.

- Up/down count register set method (software reload)

  The up/down count registers (UDCR0, UDCR1) are read-only and cannot be written to. Set in the following order.

  - Set the selected value in the reload compare register (RCR0, RCR1).
  - When the up/down count register write bit in the counter control register is set to "1" (CCR:CTUT=1), the value set in the reload compare register is transferred to the up/down counter register.

- Up/down count register clear method

  Up/down count register can be cleared as follows. The up/down count register can be cleared to "0000H" during count operation or when paused.

  - Clear by reset input
  - Clearing through edge input from the ZIN pin (Set up when CCR:CGSC = 1; CCR:CGE1 and CGE0 = "01B" and "10B")
  - Counter control register is cleared when the UDCR clear bit is set to "0" (CCR:UDCC=1)
  - Clearing caused by compare match (CCR:UCRE=1)

**Note:** The word command (MOVW) must be used to read the up/down counter registers (UDCR0, UDCR1) in the 16-bit, 1-channel mode (CCR0:M16E = 1).
11.4.8  Reload Compare Register (RCR0, RCR1)

The reload compare registers (RCR0, RCR1) are used to set reload values in the up/down count registers (UDCR0, UDCR1) and the compared compare value.

■ Reload compare register (RCR0, RCR1)

When the reload function operation is enabled (CCR:RLDE=1), transfers the value set in the reload compare register to the up/down count register by an underflow in the up/down count register (CSR1:UDFF=1).

When the up/down count register (UDCR0, UDCR1) clear is enabled by a compare match (CCR:UCRE=1), and the up/down count register value matches the reload compare register set value (CSR:CMPF=1) during a count up, the up/down count register is cleared to "00H".

The reload function is enabled, setting a compare match to enable clear, enables the up/down count register to count up/down between "00H" and the value set in the reload compare register.

When the counter write bit in the counter control register is set to "1" (CCR:CTUT=1), the value set in the reload compare register is transferred to the up/down count register.

When the 16-bit 1-channel mode is set (CCR0:M16E = 1), the word command (MOVW) must be used to write the reload compare register.
11.5 8-/16-bit Up/Down Counter/Timer Interrupt

- 8-/16-bit up/down counter/timer generates interrupt requests when overflows, underflows, compare matches or count direction changes occur.
- The 8-/16-bit up/down counter/timer interrupt corresponds to the El²OS.

Table 11.5-1 shows 8-/16-bit up/down counter/timer interrupt control bits and interrupt factors.

<table>
<thead>
<tr>
<th>Interrupt flag</th>
<th>Overflow Interrupt</th>
<th>Underflow Interrupt</th>
<th>Comparison Interrupt</th>
<th>Count direction change interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt enable bit</td>
<td>CSR : OVFF</td>
<td>CSR : UDFF</td>
<td>CSR : CMPF</td>
<td>CCR : CDCF</td>
</tr>
<tr>
<td>Interrupt enable bit</td>
<td>CSR : UDIE</td>
<td>CSR : UDIE</td>
<td>CSR : CITE</td>
<td>CCR : CFIE</td>
</tr>
<tr>
<td>Interrupt factor</td>
<td>Counter overflow</td>
<td>Counter underflow</td>
<td>A match between the UDCR value and RCR set value</td>
<td>Change of count direction</td>
</tr>
</tbody>
</table>

- **Overflow interrupt**
  
  When an overflow occurs in the up/down count registers (UDCR0, UDCR1) during a count up, the overflow generation flag in the count status register is set to "1". (CSR: OVFF=1)
  
  When overflow/underflow interrupts are enabled (CSR:UDIE=1), an interrupt request is generated.

- **Underflow interrupt**
  
  When an underflow occurs in the up/down count registers (UDCR0, UDCR1) during a count down, the underflow generation flag in the count status register is set to "1". (CSR:UDFF=1)
  
  When overflow/underflow interrupts are enabled (CSR:UDIE=1), an interrupt request is generated.

- **Compare match interrupt**
  
  When the up/down count register value matches the reload compare register (RCR0, RCR1) set values during a count up, the counter status register compare match generation flag is set to "1". (CSR: CMPF=1)
  
  When the compare interrupt request is enabled (CSR:CITE=1), an interrupt request is generated.

- **Count direction change interrupt**
  
  When count direction changes from up to down or down to up during up/down count registers (UDCR0, UDCR1) counting, the counter control register count direction change flag is set to "1" (CCR:CFIE=1).
  
  When the count direction change interrupt is enabled (CCR:CFIE=1), an interrupt request is generated.
8-/16-bit up/down counter/timer interrupt and EI²OS

Reference: Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

Note: Both channels 0 and 1 use the same interrupts for underflow interrupts, overflow interrupts, and count direction change interrupts. Check each flag (CSR:OVFF, UDFF and CCR:CDCF) by interrupt processing.

8-/16-bit up/down counter/timer EI²OS

The 8-/16-bit up/down counter/timer supports EI²OS.

However, when EI²OS is used, interrupt of other interrupt factors that share the interrupt control register (ICR) must be disabled.
11.6 8-/16-bit Up/Down Counter/Timer Operating Modes

The 8-/16-bit up/down counter/timer has two operating modes: an 8-bit 2-channel mode and a 16-bit 1-channel mode.

■ Operating mode

The 8-/16-bit up/down counter/timer has two operating modes: an 8-bit 2-channel mode and a 16-bit 1-channel mode.

Table 11.6-1 shows details on operating mode, register and pin compatibility.

Table 11.6-1 Support between the operation mode and the register/pin

<table>
<thead>
<tr>
<th>Operation mode</th>
<th>Count Registers</th>
<th>Compare Reload Registers</th>
<th>Count Control Registers</th>
<th>Count Status Registers</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UDCR0</td>
<td>UDCR1</td>
<td>RCR0</td>
<td>RCR1</td>
<td>CCR0</td>
</tr>
<tr>
<td>2 channels</td>
<td>Channel 0</td>
<td>Channel 1</td>
<td>Channel 0</td>
<td>Channel 1</td>
<td>Channel 0</td>
</tr>
<tr>
<td>1 channel</td>
<td>Channels 0,1</td>
<td>Channels 0,1</td>
<td>Channels 0,1</td>
<td>Not provided</td>
<td>Channels 0,1</td>
</tr>
</tbody>
</table>

● 8-bit 2-channel mode

The high-order side of the up/down count register uses channel 1 (UDCR1) and the low-order side of that uses channel 0 (UDCR0) in 8-bit, 2-channel mode.

Use of the 8-bit, 2-channel mode is shown in Figure 11.6-1.
● 16-bit 1-channel mode

In 16-bit, 1-channel mode, use both the high-order side (UDCR1) and low-order side (UDCR0) of the up/down count register as a 16-bit up/down count register.

Use of the 16-bit, 1-channel mode is shown in Figure 11.6-2.

**Figure 11.6-2 16-bit, 1-channel mode setup**

<table>
<thead>
<tr>
<th>UDCR1, UDCR0</th>
<th>Counter value is displayed (16-bit data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCR1, RCR0</td>
<td>Setting reload values or compare values (16-bit data)</td>
</tr>
<tr>
<td>CSR0</td>
<td>CSTA, CITE, UDIE, CMPF, OVFF, UDFF, UDFO</td>
</tr>
<tr>
<td>CCR0</td>
<td>M16E, CDCF, CFIE, CLKS, CMS1, CMS0, CES1, CES0, CTU, UCRE, RLDE, UDCC, CGSC, CGE1, CGE0</td>
</tr>
<tr>
<td>CSR1</td>
<td>CSTR, CITE, UDIE, CMPF, OVFF, UDFF, UDFO</td>
</tr>
<tr>
<td>CCR1</td>
<td>CDCF, CFIE, CLKS, CMS1, CMS0, CES1, CES0, CTU, UCRE, RLDE, UDCC, CGSC, CGE1, CGE0</td>
</tr>
</tbody>
</table>

○: Used bit
X: Disabled bit
1: Set to "1"
11.7 8-/16-bit Up/Down Counter/Timer Count Mode

The 8-/16-bit up/down counter/timer provides four count modes: timer mode, up/down count mode, phase differential count mode (2 multiplier) and phase differential count mode (multiply by 4).

■ Count mode

As shown in Table 11.7-1, the 8-/16-bit up/down counter/timer provides four count modes. The count mode is set using the count mode selection bit in the count control register (CCR: CMS1, CMS0).

Table 11.7-1  Count mode

<table>
<thead>
<tr>
<th>CMS1</th>
<th>CMS0</th>
<th>Count mode selection bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Timer mode (count down)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Up/down count mode</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Phase difference count mode (2 multiplier)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Phase difference count mode (4 multiplier)</td>
</tr>
</tbody>
</table>

● Timer mode

- The timer mode counts down using the prescaler count clock.
- The count clock can be set to $\phi/2$ or $\phi/8$ ($\phi$:machine clock frequency) using the count clock selection bit in the counter control register (CCR: CLKS).

● Up/down count mode

- The up/down count mode starts count up operation when the edge is input to the AIN pin and starts count down operation when the edge is input to the BIN pin.
- The count edge selection bit in the count control register (CCR: CES1, CES0) can be used to set the edge to be detected.

● Phase differential count mode

Phase differential count mode (multiply by 2/4) counts the phase differentials in the AIN pin input (encoder output signal phase A) and BIN pin input (encoder output phase B).

- Phase differential count mode (multiply by 2): BIN pin rising/falling edge is used to detect the input level in the AIN pin to count up or count down.
- Phase differential count mode (multiply by 4): BIN pin rising/falling edge is used to detect the input level in the AIN pin to count up or count down while AIN pin rising/falling edge is used to detect the input level in the BIN pin to count up or count down.

- AIN and BIN pin input phase differential: Count up starts when the AIN pin input is first and count down starts when the BIN input is first.
11.8 Operation of Timer Mode

The timer mode counts down using the count clock from prescaler.

Figure 11.8-1 Timer mode setting

![Figure 11.8-1 Timer mode setting](image)

In the timer mode, a count down occurs using the divided clock (\(\phi/2\) or \(\phi/8\)) of the machine clock frequency (\(\phi\)) as the count clock.

(When an underflow is generated by a count down)

- When set to overflow/underflow interrupt enable (CSR:UDIE=1), an interrupt request is generated.
- If the reload function is set to the enabled state (CCR:RLDE=1), the reload compare register (RCR0, RCR1) set values are reloaded to the up/down count registers (UDCR0, UDCR1) to continue the count down.

(Up/down count register setting)

- The reset value of the up/down count register is "00H".
- After setting the up/down counter/timer operation to the disabled state (CSR:CSTR=0) and setting the up/down count register values (UDCR0, UDCR1) to the reload compare registers (RCR0, RCR1), set the counter write bit in the counter control register to "1" (CCR:CTUT=1) and transfer the set reload compare register value to the up/down count register.
11.9 Up/Down Count Mode Operation

The up/down count mode starts count up operation when the edge is input to the AIN pin and starts count down operation when the edge is input to the BIN pin.

■ Up/down count mode operation

![Figure 11.9-1 Up/down count mode setting](image)

In the up/down counter mode, a count up occurs with an edge input to the AIN pin and a count down occurs with an edge input to the BIN pin.

(Selection of a valid edge)

A setting for the valid edge to the AIN pin and the BIN pin can be selected from rising edge, falling edge or both edges by setting the counter control register (CCR:CES1, CES0).

(Up/down count register setting)

- The reset value of the up/down count register is "00H".
- After setting the up/down counter/timer operation to the disabled state (CSR:CSTR=0) and setting the up/down count register values (UDCR0) to the reload compare registers (RCR0, RCR1), set the counter write bit in the counter control register to "1" (CCR:CTUT=1) and transfer the set reload compare register value to the up/down count register.

(Start of the up/down count register counting operation)

By setting the operation mode, count edge, enable/disable state of each interrupt, reload/compare function, counter clear function and gate function and setting the up/down counter/timer start bit in the counter status register to "1" (CSR:CSTR=1), the counting operation of the up/down count register can be started.

(Occurrence of an interrupt)

When the counting operation of the up/down count register causes an overflow/underflow, compare match and direction change when each corresponding interrupt is enabled, an interrupt request is generated.
11.10 Phase Differential Count Mode Operation

Phase differential mode (multiplier 2/4) counts the phase differentials in the AIN pin input (encoder output signal phase A) and BIN pin input (encoder output phase B). Phase differential count mode operation

Figure 11.10-1 Setting in the phase difference count mode

<table>
<thead>
<tr>
<th>UDCR1, UDCR0</th>
<th>Counter value display</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCR1, RCR0</td>
<td>Setting reload values or compare value</td>
</tr>
<tr>
<td>CSR0</td>
<td></td>
</tr>
<tr>
<td>CCR0</td>
<td>M16E</td>
</tr>
<tr>
<td></td>
<td>O</td>
</tr>
<tr>
<td>CSR1</td>
<td></td>
</tr>
<tr>
<td>CCR1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>

Ø : Used bit
○ : Can only be used 8-bit 2-channel mode
(The setting at using channel 1 as the differential count is sure to follow the channel 0 setting.)
△ : at 2-divided: set "10B", at 4-divided: set "11B"
X : Unused bit

Phase difference count mode (2 multiplier)
To count phase differences between phase A and phase B of the encoder output signal, a count up or a count down is initiated by the detection of the level of input to the AIN pin with the timing of the rising/falling edge of BIN pin.

Phase difference count mode (4 multiplier)
To count phase differences between phase A and phase B of the encoder output signal, a count up or a count down is initiated by the detection of the level of input to the AIN pin with the timing of the rising/falling edge of the BIN pin or a count up or a count down is initiated by the detection of the level of input to the BIN pin with the timing of the rising/falling edge of the AIN pin.

(Phase difference between the AIN pin input and the BIN pin input)
A count up is initiated when the input to the AIN pin precedes and 9 count down is initiated when the input to the BIN pin precedes.
(Up/down count register setting)

- The reset value of the up/down count register is "00H".
- After setting the up/down counter/timer operation to the disabled state (CSR:CSTR=0) and setting the up/down count register values (UDCR0, UDCR1) to the reload compare registers (RCR0, RCR1), set the counter write bit in the counter control register to "1" and transfer the set reload compare register value to the up/down count register.

(Start of the counting operation)

By setting the operation mode, enable/disable state of each interrupt, reload/compare function, counter clear function and gate function and set the up/down counter/timer start bit in the counter status register to "1" (CSR:CSTR=1), the counting operation can be started.

(Occurrence of an interrupt)

When the counting operation of the up/down count register causes an overflow/underflow, compare match and direction change, and when each corresponding interrupt is enabled, an interrupt request is generated.

● Phase difference count mode (2 multiplier)

In 2 multiplier mode, a count up or a count down is initiated by the detection of the level of input to the AIN pin when the rising/falling edge of the BIN pin.

- A count up is initiated if the BIN pin level detected when the rising edge is input to the AIN pin is "H".
- A count down is initiated if the BIN pin level detected when the rising edge is input to the AIN pin is "L".
- A count down is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "H".
- A count up is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "L".

Figure 11.10-2 shows the operation in the phase difference count mode (2 multiplier).

**Figure 11.10-2 Phase difference (2 multiplier) count mode operation**

<table>
<thead>
<tr>
<th>AIN pin</th>
<th>BIN pin</th>
<th>Count value</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1 +1 +1 +1 +1 -1 +1 -1 -1 -1 -1 -1</td>
<td>+1 +1 +1 +1 +1 -1 +1 -1 -1 -1 -1 -1</td>
<td>0 1 2 3 4 5 4 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>
Phase difference (4 multiplier) count mode

In 4 multiplier mode, a count up or a count down is initiated by the detection of the level of input to the AIN pin with the timing of the rising/falling edge of the BIN pin or a count up or a count down is initiated by the detection of the level of input to the BIN pin with the timing of the rising/falling edge of the AIN pin.

- A count up is initiated if the BIN pin level detected when the rising edge is input to the AIN pin is "H".
- A count down is initiated if the BIN pin level detected when the rising edge is input to the AIN pin is "L".
- A count down is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "H".
- A count up is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "L".
- A count up is initiated if the BIN pin level detected when the rising edge is input to the AIN pin is "L".
- A count up is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "H".
- A count down is initiated if the BIN pin level detected when the falling edge is input to the AIN pin is "L".

Figure 11.10-3 shows the operation in the phase difference count mode (4 multiplier).

**Figure 11.10-3 Phase difference (4 multiplier) count mode operation**

<table>
<thead>
<tr>
<th>AIN pin</th>
<th>BIN pin</th>
<th>Count value</th>
</tr>
</thead>
<tbody>
<tr>
<td>↓</td>
<td>+1 +1 +1 +1 +1 +1 +1 +1 +1 +1 -1 +1 -1 -1 -1 -1 -1 -1 -1 -1</td>
<td>0 1 2 3 4 5 6 7 8 9 10 9 10 9 8 7 6 5 4 3 2 1</td>
</tr>
</tbody>
</table>
11.11 Reload/Compare Function Operation

The 8-/16-bit up/down counter/timer has the reload function and the compare function, both of them can be combined for use.

### Reload/Compare function

The reload/compare function can be set by the counter control registers (CCR0, CCR1), as shown in Table 11.11-1.

#### Table 11.11-1 Selecting Reload/Compare Function

<table>
<thead>
<tr>
<th>RLDE bit</th>
<th>UCRE bit</th>
<th>Reload/Compare function</th>
<th>Counter by Compare match</th>
<th>Reload operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Reload/compare inhibit (reset value)</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Compare enabled</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Reload is enabled</td>
<td>×</td>
<td>○</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Reload/Compare enabled</td>
<td></td>
<td>○</td>
</tr>
</tbody>
</table>

Note: The reload function does not work if a count down is not performed.
Compare function

- If the clearing of the up/down count registers (UDCR0, UDCR1) when a compare match occurs is set to the enabled state (CCR:UCRE=1), when the up/down count register values match the set value of the reload compare register (RCR1) as a result of a count up, the up/down count registers will be cleared when the next count up occurs.
- If the interrupt request is set to the enabled state (CSR:CITE=1) when a compare match occurred, the compare match happened flag bit is set to "1" (CCR:MDF=1), an interrupt request will be generated.
- The compare function operation is shown in Figure 11.11-2.

**Figure 11.11-2 Compare function operation**

**Note:** The compare function is invalid if a count up is not performed.
Concurrent use of the reload function and the compare function

- When the reload function and the compare function are in concurrent use, a count up or count down from "00H" to the set reload compare register values (RCR0, RCR1) occurs.
  - When an underflow is caused by the reload function, the set reload compare register values are transferred to the up/down count register (UDCR0, UDCR1).
  - When the compare function matches between the set reload compare register value and the up/down count register value, the up/down count register is cleared to "0000H".

- If a compare match or a reload (underflow) occurs, an interrupt request can be generated. Whether to enable or disable an interrupt to be caused by a compare match or an underflow can be specified by the register.

The operation when the reload function and the compare function are in concurrent use is shown in Figure 11.11-3.

Figure 11.11-3 Operation when the reload function and the compare function are in concurrent use
11.12 Reload and Clear Timing

The reload and counter clear timing for the up/down count registers (UDCR0, UDCR1) vary when the up/down count registers are performing the counting operation and when they are stopped.

### Reload/counter clear timing

The reload and counter clear timing for the up/down count registers vary when the up/down count registers are performing the counting operation and when they are stopped.

Figure 11.12-1 to Figure 11.12-3 shows reload and clear timing.

- **During count operation**

  A reload and counter clear during the counting operation of the up/down count register are synchronous with the count clock.

  **Figure 11.12-1 Reload and counter clear timing during the counting operation**

- **Immediately prior to the counting stop**

  If a reload or counter clear event that is generated during the counting operation of the up/down count register causes a stop during the count timing synchronization wait state (waiting for a synchronizing input), a reload or a counter clear occurs when the stop occurs.

  **Figure 11.12-2 Reload and counter clear timing immediately prior to the counting stop**
During stopping count

If a reload or counter clear event is generated during the counting stop of the up/down count register, a reload or a counter clear occurs when the event is generated.

**Figure 11.12-3 Reload and counter clear timing during the counting stop**

<table>
<thead>
<tr>
<th>UDCR register</th>
<th>65H</th>
<th>80H</th>
<th>(Reload example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reload/counter event</td>
<td>Reload</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Counter clear caused by a compare match

When the up/down count register values (UDCR0, UDCR1) and the set reload compare register values (RCR0, RCR1) match and when a count up also occurs, a counter clear occurs. Even when the up/down count register value matches the set reload compare register value, no counter clear does not occur when the up/down count register performs a count down or stops counting.

**Note:** When a counter clear event and a reload event occur simultaneously, the counter clear event takes precedence and the reload is not performed.
11.13 ZIN Pin Clear/Gate Function Operation

The 8-/16-bit up/down counter/timer has the counter clear function to clear the up/down count register with an edge input from the ZIN pin and the gate function to perform counting while valid level signals are input.

- **Up/down count register clear/gate function**

  The ZIN pin can perform the counter clear function or the gate function, depending on how the counter clear function/gate function selection bit (CCR:CGSC) is set in the counter control register. Table 11.13-1 shows the ZIN pin function setting.

  **Table 11.13-1 ZIN pin function selection**

<table>
<thead>
<tr>
<th>CGSC</th>
<th>ZIN pin function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Counter clear function</td>
</tr>
<tr>
<td>1</td>
<td>Gate function</td>
</tr>
</tbody>
</table>

  **Table 11.13-2 Counter clear/gate function edge/level selection**

<table>
<thead>
<tr>
<th>CGE1,CGE0</th>
<th>Counter clear function</th>
<th>Gate function</th>
</tr>
</thead>
<tbody>
<tr>
<td>00B</td>
<td>Edge detection is disabled.</td>
<td>Level detection inhibited</td>
</tr>
<tr>
<td>01B</td>
<td>Cleared when the rising edge is detected</td>
<td>&quot;L&quot; level interval count enabled</td>
</tr>
<tr>
<td>10B</td>
<td>Cleared when the falling edge is detected</td>
<td>&quot;H&quot; level interval count enabled</td>
</tr>
</tbody>
</table>

- **Up/down count register clear function**
  - When the counter clear function is selected, the up/down count register is cleared to "0000H" with an edge input from the ZIN pin.
  - A valid edge is specified by the counter clear function/gate function selection bit in the counter control register (CCR:CGE1, CGE0).
  - By inputting the Z phase output from the encoder to the ZIN pin, the up/down count register can be cleared with the start of encoder counting.
  - The counter clear function is enabled in all the count modes.

- **Function**
  - When the gate function is selected, the counting operation of the up/down count register is enabled with an input level of the ZIN pin.
  - The input level of permission for counting by the up/down count register is specified by the counter clear edge/count level selection bit in the counter control register (CCR:CGE1, CGE0).
  - The gate function is enabled in all the count modes.
11.14 Precautions for 8-/16-bit Up/Down Counter/Timer

Pay attention to the following points when using the 8-/16-bit up/down counter/timer.

- **Precautions for 8-/16-bit up/down counter/timer**

  - **Count timing**
    
    The count timing varies by each count mode. The prescaler output clock determines the count timing in the timer mode, and the AIN/BIN pin determines the count timing in other modes.

  - **Precautions for use in the timer mode**
    
    Since the timer mode only allows a count down, the compare function does not work.

  - **Precautions for use in the phase differential count mode**
    
    The phase differential count mode does not allow the selection of the count timing edge. To be detected at both edges.

  - **Precautions for use in the 16-bit one channel mode**
    
    - Be sure to use word commands (MOVW) to read out the up/down count registers (UDCR0, UDCR1) in the 16-bit mode.
    
    - Be sure to use word commands (MOVW) to write into the reload compare registers (RCR0, RCR1) in the 16-bit mode.
11.15 8-/16-bit Up/Down Counter/Timer Program Example

Shows a program example of the 8-/16-bit up/down counter/timer.

Program example of the 8-/16-bit timer mode

- Processing specification
  - Set up the 16-bit timer mode, enable the reload and the underflow interrupt and set the counter reset value to 1024 (400H).
  - Set the machine clock frequency (φ) to 16 MHz and the count clock to φ/8 (2 MHz).

- Coding example

```assembly
ICR05 EQU 0000B5H ;Interrupt control register
UDCR0 EQU 000080H ;Up/down count register 0
RCR0 EQU 000082H ;Reload compare register 0
CSR0 EQU 000084H ;Counter status register 0
CCRL0 EQU 000086H ;Count control register 0 Low-order
CCRH0 EQU 000087H ;Count control register 0 High-order
CSTR EQU CSR0:7 ;Timer start bit
UDFF EQU CSR0:2 ;Underflow happened flag

;---------Main Program------------------------------------------------
CODE CSEG
START:
    ; Stack pointer (SP)
    ; Already initialized
    AND CCR,#0BFH ; Interruption is disabled
    MOV I:ICR05,#00H ; Interrupt levels 0 (strength)
    MOVW I:RCR0,#0400H ; Sets up the reload value
    MOV I:CCRH0,#10010000B ; 16-bit mode, 8/φ clock, Operation mode
    MOV I:CCRL0,#01111000B ; Enables a reload, Disabled the ZIN0 pin, Clears UDCR
    MOV I:CSR0,#00100000B ; Clears each interrupt request flag
                          ; Enables the underflow interrupt
    MOV ILM,#07H ; Sets ILM in PS to level 7
    OR CCR,#40H ; Interruption is enabled
    SETB CSTR ; Starts the timer operation.
LOOP:
    User processing

    BRA LOOP

;---------Interrupt Program-------------------------------------------
WARI:
    CLR B I:UDFF ; Clears the interrupt request flag
    User processing

    RETI ; Returning from interrupt processing
CODE ENDS
```

417
CHAPTER 11 8-/16-BIT UP/DOWN COUNTER/TIMER

;---------Vector Settings---------------------------------------------
VECT   CSEG ABS=0FFH
    ORG 00FF64H ;vector set in interrupt number #22(16H)
    DSL WARI
    ORG 00FFDCCH ;Reset vector setting
    DSL START
    DB 00H ;Sets single-chip mode
VECT    ENDS
    END  START

Program example of the program mode of the 8-bit up/down count mode

● Processing specification
  • Use channel 0 in the 8-bit up/down count mode.
  • Enable the compare, the reload, the overflow interrupt and the underflow interrupt and set the counter reset value to 100 (64H).
  • AIN0 pin is the count up input and the BIN0 pin is the count down input.

● Coding example

ICR05 EQU 0000B5H ;Interrupt control register
DDR2 EQU 000012H ;Port 2 direction register
RCR0 EQU 000082H ;Reload compare register 0
CSR0 EQU 000084H ;Counter status register 0
CCRL0 EQU 000086H ;Count control register 0 Low-order
CCRH0 EQU 000087H ;Count control register 0 High-order
UDFF EQU CSR0:2 ;Underflow happened flag
OVFF EQU CSR0:3 ;Overflow happened flag
CMPF EQU CSR0:4 ;Compare match happened flag
CSTR EQU CSR0:7 ;Timer start bit

;---------Main program------------------------------------------------
CODE       CSEG
START:
   ;Stack pointer (SP)
   ;Already initialized
   AND CCR,#0BFH ;Interruption is disabled
   MOV I:ICR05,#00H ;Interrupt levels 0 (strength)
   MOV I:DDR2,#00H ;Sets the 24/AIN0 and P25/BIN0 pin for input.
   MOVW I:RCR0,#64H ;Sets up the reload compare value
   MOV I:CCRH0,#00000110B;8-bit mode
   ;Count up/down mode,
   ;Disables the count direction change interrupt.,
   ;Selects the rising edge detection.
   MOV I:CCRL0,#00110000B;Clears the compare and enables the reload.
   ;Clears UDCR and disabled the ZIN0 pin.
   MOV I:CSR0,#01100000B;Clears each interrupt request flag
   ;Enables the compare and the underflow interrupt.
   MOV I:ILM,#07H ;Sets ILM in PS to level 7
   OR CCR,#40H ;Interruption is disabled
   SETB CSTR ;Count start
LOOP:

User processing
BRA LOOP

;---------Interrupt Program-------------------------------------------
WARI0: ;Compare interrupt
  CLR B I: CMPF ;Clears the interrupt request flag.

User processing
  RETI ;Returning from interrupt processing

WARI1: ;Overflow and underflow interrupt
  CLR B I: UDFF ;Clears the interrupt request flag.

User processing
  RETI ;Returning from interrupt processing

CODE ENDS

;---------Vector Settings---------------------------------------------
VECT CSEG ABS=0FFH
  ORG 00FFA4H ;vector set in interrupt number #22(16H)
  ;(Overflow)
  DSL WARI1

ORG 00FFA8H ;vector set in interrupt number #21(15H)
  ;(Compare)
  DSL WARI0

ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode

VECT ENDS
END START

Program example of the 8-bit phase difference count mode

● Processing specification
  
  - Uses the 8-bit phase difference count mode (2 multiplier) and channel 0.
  - The AIN0, BIN0 and ZIN0 pins are used for the phase A, phase B and phase Z encoder input, respectively.
  - Clears the counter with the phase Z input.
  - Enables the compare, the reload, the overflow interrupt and the underflow interrupt.
  - Sets the counter reload value to 100(64H).
  - (The encoder resolution is assumed to be 100 pitches per rotation.)

● Coding example

ICR05 EQU 0000B5H ;Interrupt control register
DDR2 EQU 000012H ;Port 2 direction register
RCR0 EQU 000082H ;Reload compare register 0
CSR0 EQU 000084H ;Counter status register 0
CCRL0 EQU 000086H ;Count control register 0 Low-order
CCRH0 EQU 000087H ;Count control register 0 High-order
UDFF EQU CSR0:2 ;Underflow happened flag
OVFF EQU CSR0:3 ;Overflow happened flag
CHAPTER 11  8-/16-BIT UP/DOWN COUNTER/TIMER

CMPF EQU CSR0:4 ;Compare match happened flag
CDCF EQU CCRH0:6 ;Count direction change flag
CSTR EQU CSR0:7 ;Timer start bit

;-------------Main Program---------------------------------------------
CODE CSEG
START:

; ;Stack pointer (SP)
;Already initialized
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR05,#00H ;Interrupt levels 0 (strength)
MOV I:DDR2,#00H ;Configure P24/AIN0, P25/BIN0, and P26/ZIN0 pins
;Sets input pins.
MOVW I:RCR0,#64H ;Sets up the reload compare value
MOV I:CCRH0,#00101000B;8-bit mode,
;Phase difference count mode (2 multiplier)
;Clears the count direction change flag.
;Enables the count direction change and interrupt.
MOV I:CCRL0,#00110001B;Clears the compare, enables the reload and clears UDCR.
;Counter Clear function select,
;Clears with the ZIN0 pin rising edge.
MOV I:CSR0,#01100000B ;Clears each interrupt request flag,
;Compare, underflow and
;overflow interrupt enabled
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled
SETB CSTR ;Count start
LOOP:

;User processing
BRA LOOP ;

;------------------------------Interrupt Program---------------------
WAR10:

;Compare interrupt
CLRB I:CMPF ;Clears the interrupt request flag.

;User processing

;Returning from interrupt processing
WARI1:

;Overflow interrupt
CLRB I:UDFF ;Clears the interrupt request flag.
CLRB I:OVFF ;Clears the interrupt request flag.
CLRB I:CDCF ;Clears the interrupt request flag.

;User processing

;Returning from interrupt processing
CODE ENDS

;----------------Vector Settings-------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFA8H ;vector set in interrupt number #22 (16H)
; (Overflow)
DSL WARI1

420
ORG 00FFA0H ;vector set in interrupt number #21(15H)
; (Compare)
DSL WARI0
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START
CHAPTER 12
DELA YED INTERRUPT
GENERATION MODULE

This chapter explains the functions and operations of the delayed interrupt generation module.

12.1 Overview of Delayed Interrupt Generation Module
12.2 Block Diagram of Delayed Interrupt Generation Module
12.3 Delayed Interrupt Request Generate/Cancel Register (DIRR)
12.4 Explanation of Operation of Delayed Interrupt Generation Module
12.5 Precautions when Using Delayed Interrupt Generation Module
12.6 Program Example of Delayed Interrupt Generation Module
## 12.1 Overview of Delayed Interrupt Generation Module

The delayed interrupt generation module generates the interrupt for task switching. Hardware interrupt requests can be generated or canceled by the software.

### ■ Overview of Delayed Interrupt Generation Module

By using the delayed interrupt generation module, a hardware interrupt request can be generated or cancelled by software.

Table 12.1-1 shows overview of delayed interrupt generation module.

### Table 12.1-1 Overview of Delayed Interrupt Generation Module

<table>
<thead>
<tr>
<th>Function and Control</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interrupt Factor</strong></td>
<td>An interrupt request is generated by setting the R0 bit in the delayed interrupt request generate/cancel register to &quot;1&quot; (DIRR: R0 = 1). An interrupt request is cancelled by setting the R0 bit in the delayed interrupt request generate/cancel register to &quot;0&quot; (DIRR: R0 = 0).</td>
</tr>
<tr>
<td><strong>Interrupt Number</strong></td>
<td>#42 (2A_H)</td>
</tr>
<tr>
<td><strong>Interrupt Control</strong></td>
<td>An interrupt is not enabled by the register.</td>
</tr>
<tr>
<td><strong>Interrupt flag</strong></td>
<td>The interrupt flag is held in the R0 bit in the DIRR register.</td>
</tr>
<tr>
<td><strong>EI^2OS</strong></td>
<td>The DIRR register does not correspond to the EI^2OS.</td>
</tr>
</tbody>
</table>
12.2 Block Diagram of Delayed Interrupt Generation Module

The delayed interrupt generation module consists of the following blocks:
- Interrupt request latch
- Delayed interrupt request generate/cancel register (DIRR)

- **Block Diagram of Delayed Interrupt Generation Module**

![Figure 12.2-1 Block Diagram of Delayed Interrupt Generation Module](image)

- **Interrupt request latch**
  This latch the setting of the delayed interrupt request generate/cancel register

- **Delayed interrupt request generate/cancel register (DIRR)**
  This bit generates or cancels a delayed interrupt request.

- **Interrupt Number**
  The interrupt number used in the delayed interrupt generation module is as follows:
  Interrupt number #42 (2AH)
12.3 Delayed Interrupt Request Generate/Cancel Register (DIRR)

Delayed interrupt request generate/cancel register (DIRR) generates or cancels a delayed interrupt request.

Figure 12.3-1 Delayed interrupt request generate/cancel register (DIRR)

Table 12.3-1 Delayed interrupt request output cancellation register (DIRR) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8</td>
<td>R0: Delayed interrupt request generate bit</td>
</tr>
<tr>
<td></td>
<td>This bit generates or cancels a delayed interrupt request. When set to &quot;0&quot;: Cancels delayed interrupt request When set to &quot;1&quot;: Generates delayed interrupt request</td>
</tr>
<tr>
<td>bit9 to bit15</td>
<td>Undefined bits</td>
</tr>
<tr>
<td></td>
<td>Read: The value is undefined. Write: No effect</td>
</tr>
</tbody>
</table>
12.4 Explanation of Operation of Delayed Interrupt Generation Module

The delayed interrupt generation module has a function that generates or cancels interruption requests using the software.

■ Explanation of Operation of Delayed Interrupt Generation Module

Using the delayed interrupt generation module requires the setting shown in Figure 12.4-1.

![Figure 12.4-1 Setting for Delayed Interrupt Generation Module](image)

When the R0 bit in the delayed interrupt request generate/cancel register (DIRR) is set to "1" (DIRR: R0 = 1), an interrupt request is generated. There is no interrupt request enable bit.

● Operation of delayed interrupt generation module

- When "1" is set to the delayed interrupt request generate/cancel register R0 bit, the interrupt request latch will be set to "1", and an interrupt request will be issued to the interrupt controller.
- For the interrupt controller, if it is determined that the interrupt priority is higher than other interrupt requests, an interrupt request will be generated to the CPU.
- The interrupt level mask bit (CCR:ILM) and interrupt request level (ICR:IL) of the condition code register are compared at the CPU side, and if the interrupt request level is higher than the ILM, a delay interrupt processing will be executed after the command that is currently executing ends.
- Interrupt request is cancelled and the task is switched by setting "0" to the R0 bit of the user program within interrupt processing.

Figure 12.4-2 shows operation of delayed interrupt generation module.

![Figure 12.4-2 Operation of delayed interrupt generation module](image)
12.5 Precautions when Using Delayed Interrupt Generation Module

This section explains the precautions when using the delayed interrupt generation module.

- **Precautions when Using Delayed Interrupt Generation Module**
  - The interrupt processing is restarted at return from interrupt processing without setting the R0 bit in the delayed interrupt request generate/cancel register (DIRR) to "0" within the interrupt processing routine.
  - Delay is involved with interruption using the delayed interrupt generation module in a different manner from a software interruption.
12.6 Program Example of Delayed Interrupt Generation Module

This section gives a program example of the delayed interrupt generation module.

### Program Example of Delayed Interrupt Generation Module

#### Processing specification

Delay interrupt request is generated by writing "1" to the R0 bit of the delay interrupt request generate/cancel register (DIRR) in the main program, and task switching is carried out.

#### Coding example

```assembly
ICR15 EQU 0000BFH ;Interrupt control register
DIRR EQU 00009FH ;Delayed interrupt request generate/cancel register
DIRR_R0 EQU DIRR:0 ;Delayed interrupt request generate bit

;--------Main Program----------------------------------
CODE CSEG
START:
    ; Stack pointer (SP)
    ; Already initialized
    AND CCR,#0BFH ; Interruption is disabled
    MOV I:ICR15,#00H ; Interrupt levels 0 (strength)
    MOV ILM,#07H ; Sets ILM in PS to level 7
    OR CCR,#40H ; Interruption is enabled
    SETB I:DIRR_R0 ; Delay interrupt request generation
LOOP:
    BRA LOOP

;--------Interrupt Program-------------------------------
WARI:
    CLR I:DIRR_R0 ; Clears the interrupt request flag.
    ; User processing
    RETI ; Returning from interrupt processing

;--------Vector Settings-------------------------------
VECT CSEG ABS=0FFH
ORG 0FF54H ; Vector set up to the interruption number #42(2Ah)
DSL WARI
ORG 0FFDCH ; Reset vector setting
DSL START
DB 00H ; Sets single-chip mode

VECT ENDS

END START
```

---

429
This chapter explains the functions and operations of DTP/external interrupt.

13.1 Overview of DTP/External Interrupt
13.2 Block Diagram of DTP/External Interrupt
13.3 Configuration of DTP/External Interrupt
13.4 Explanation of Operation of DTP/External Interrupt
13.5 Precautions when Using DTP/External Interrupt
13.6 Program Example of DTP/External Interrupt Circuit
13.1 Overview of DTP/External Interrupt

DTP/external interrupts detect interrupt requests that generate external peripheral device or data transfer requests, and generate interrupt requests to the CPU. Extended intelligent I/O service EI²OS can also be activated.

DTP/External Interrupt Function

An interrupt request that generates external peripheral device is detected, and an interrupt request is output to the CPU using the same procedure as interrupts for the peripheral function, which generates an interrupt, and also can be activated extended intelligent I/O service (EI²OS).

When extended intelligent I/O service (EI²OS) under the interrupt control register has been disabled (ICR:ISE = 0), external interrupt function will be valid, and branches off to interrupt processing.

When EI²OS has been enabled (ICR:ISE = 1), the DTP function will be valid, auto data transfer will be carried out using EI²OS, and will branch off to interrupt processing after data transfer for the specified number of times ends.

Table 13.1-1 shows overview of DTP/external interrupt.

<table>
<thead>
<tr>
<th>Input pin</th>
<th>External interrupt</th>
<th>DTP Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 pins (INT0 to INT7)</td>
<td>The interrupt factor is in unit of pins using the detection level setting registers (ELVR).</td>
<td>The EI²OS is disabled. (ICR:ISE=0)</td>
</tr>
<tr>
<td>Interrupt factor</td>
<td>Input of &quot;H&quot; level, &quot;L&quot; level, rising edge, or falling edge</td>
<td>The EI²OS is enabled. (ICR:ISE=1)</td>
</tr>
<tr>
<td>Interrupt control</td>
<td>The interrupt request is enabled/disabled using the DTP/external interrupt enable register (ENIR).</td>
<td>Auto data transfer by EI²OS. Branching to interrupt processing after processing for the specified numbers.</td>
</tr>
<tr>
<td>Interrupt flag</td>
<td>The interrupt factor is held using the DTP/external interrupt factor register (EIRR).</td>
<td></td>
</tr>
<tr>
<td>Processing selection</td>
<td>The EI²OS is disabled. (ICR:ISE=0)</td>
<td></td>
</tr>
<tr>
<td>Processing contents</td>
<td>Branching to interrupt processing</td>
<td></td>
</tr>
</tbody>
</table>

Table 13.1-1 Overview of DTP/External Interrupt
13.2 Block Diagram of DTP/External Interrupt

Figure 13.2-1 shows the block diagram of DTP/external interrupt.

![Block Diagram of DTP/External Interrupt](image-url)
CHAPTER 13  DTP/EXTERNAL INTERRUPT

● DTP/external interrupt input detector

This circuit detects interrupt requests or data transfer requests generated from external peripheral devices. Set "1" for the interrupt request flag bit that supports the pin in which the level or edge that has been set by the detection level setting register was detected. (EIRR: ER=1)

● Detection level setting register (ELVR)

Detection level setting register sets the level or edge of the input cause detection signal from the external peripheral device which will be the condition for generating causes for DTP/external interrupts.

● DTP/external interrupt factor register (EIRR)

This register retains the cause of the DTP/external interrupt. When a signal is input to the DTP/external interrupt pin, "1" is set to the supported DTP/external interrupt request flag bit.

● DTP/external interrupt enable register (ENIR)

This register enables or disables DTP/external interrupt requests from external peripheral devices.

Details of Pins and Interrupt Numbers

Table 13.2-1 shows the pins and interrupt numbers used by DTP/external interrupt

Table 13.2-1  Pins and Interrupt Numbers Used by DTP/External Interrupt

<table>
<thead>
<tr>
<th>Pin</th>
<th>Channel</th>
<th>Interrupt Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00</td>
<td>INT0</td>
<td>#13 (0Dh)</td>
</tr>
<tr>
<td>P01</td>
<td>INT1</td>
<td></td>
</tr>
<tr>
<td>P02</td>
<td>INT2</td>
<td>#18 (12h)</td>
</tr>
<tr>
<td>P03</td>
<td>INT3</td>
<td></td>
</tr>
<tr>
<td>P04</td>
<td>INT4</td>
<td>#20 (14h)</td>
</tr>
<tr>
<td>P05</td>
<td>INT5</td>
<td></td>
</tr>
<tr>
<td>P06</td>
<td>INT6</td>
<td>#24 (18h)</td>
</tr>
<tr>
<td>P26</td>
<td>INT7</td>
<td></td>
</tr>
</tbody>
</table>
13.3 Configuration of DTP/External Interrupt

This section details the pins and registers in the DTP/external Interrupt.

■ Pins of DTP/External Interrupt

The pin used for DTP/external interrupts is simultaneously used as a general-purpose input/output port. Settings to be used for the pin functions and DTP/external interrupts are shown in Table 13.3-1.

Table 13.3-1 Pins of DTP/External Interrupt

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Pin settings required for use in DTP/external interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00/INT0</td>
<td>General-purpose I/O port/ DTP External interrupt input</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P01/INT1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P02/INT2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P03/INT3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P04/INT4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P05/INT5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P06/INT6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P26/INT7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

■ Block Diagram of Pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".
13.3.1 DTP/External Interrupt Factor Register (EIRR)

This register retains the cause of the DTP/external interrupt. When a signal is input to the DTP/external interrupt pin, "1" is set to the supported interrupt request flag bit.

### DTP/external interrupt factor register (EIRR)

This register retains the cause of the DTP/external interrupt. When a signal is input to the DTP/external interrupt pin, "1" is set to the supported interrupt request flag bit.

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit15 to bit8</td>
<td>&quot;1&quot; is set when the edge or level signal that has been set by the detection condition selection bit of the detection level setting register (ELVR; LB, LA) is input to the DTP/external interrupt pin.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> When the DTP/external interrupt request enabling bit (ENIR: EN) is set to &quot;1&quot;, an interrupt request for the supported DTP/external interrupt channel is generated.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> No effect.</td>
</tr>
</tbody>
</table>

**Note:**

The bit returns "1" when read by a read modify write instruction. When a number of DTP/external interrupt requests are enabled, (ENIR: EN = 1), clear only the bits of the channels that accepted interrupt to "0". (EIRR: ER=0) No other bits must be cleared unconditionally.

**Reference:**

While the extended intelligent I/O service (EI²OS) is activated, when 1 data transfer ends, the supported interrupt request flag bit is automatically cleared to "0". (EIRR: ER=0)
13.3.2 DTP/External Interrupt Enable Register (ENIR)

This register enables or disables DTP/external interrupt requests from external peripheral devices.

- DTP/external interrupt enable register (ENIR)

**Figure 13.3-2 DTP/external interrupt enable register (ENIR)**

<table>
<thead>
<tr>
<th>bit7 to bit0</th>
<th>EN7 to EN0: DTP/external interrupt request enable bits</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W R/W R/W R/W R/W R/W R/W R/W</td>
<td></td>
<td>Set up enable or disable for the DTP/external interrupt request on the DTP/external interrupt channel. When &quot;1&quot; is set to the DTP/external interrupt request enable bit (ENIR:EN) and DTP/external interrupt request flag bit (EIRR:ER), an interrupt request is generated for the supported DTP/external interrupt pin. In the standby mode, do not cut off the input.</td>
</tr>
<tr>
<td>R/W : Readable and Writable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reference:**
The DTP/external interrupt pin status can be read using the port data register regardless of the settings of the DTP/external interrupt request enabling bit.

**Table 13.3-4 Correspondence between DTP/External Interrupt Pins, DTP/External Interrupt Request Flag Bits, and DTP/External Interrupt Enable Bits**

<table>
<thead>
<tr>
<th>DTP/External Interrupt Pins</th>
<th>DTP/interrupt request flag bit</th>
<th>DTP/interrupt request enable bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT0</td>
<td>ER0</td>
<td>EN0</td>
</tr>
<tr>
<td>INT1</td>
<td>ER1</td>
<td>EN1</td>
</tr>
<tr>
<td>INT2</td>
<td>ER2</td>
<td>EN2</td>
</tr>
<tr>
<td>INT3</td>
<td>ER3</td>
<td>EN3</td>
</tr>
<tr>
<td>INT4</td>
<td>ER4</td>
<td>EN4</td>
</tr>
<tr>
<td>INT5</td>
<td>ER5</td>
<td>EN5</td>
</tr>
<tr>
<td>INT6</td>
<td>ER6</td>
<td>EN6</td>
</tr>
<tr>
<td>INT7</td>
<td>ER7</td>
<td>EN7</td>
</tr>
</tbody>
</table>
13.3.3 Detection Level Setting Register (ELVR) (High)

The detection level setting register (High) sets the levels or edges of input signals that cause interrupt factors in INT7 to INT4 of the DTP/external interrupt pins.

---

**Detection Level Setting Register (ELVR) (High)**

**Figure 13.3-3 Detection Level Setting Register (ELVR) (High)**

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 to bit15</td>
<td>LB7, LA7 to LB4, LA4: Detection condition selection bits</td>
</tr>
<tr>
<td>bit15</td>
<td>These bits set the levels or edges of input signals from external peripheral devices that cause interrupt factors in the DTP/external interrupt pins.</td>
</tr>
<tr>
<td></td>
<td>• Select the external interrupt from 2 level types and 2 edge types, or the EI²OS from 2 level types.</td>
</tr>
<tr>
<td></td>
<td><strong>Reference:</strong></td>
</tr>
<tr>
<td></td>
<td>If the detection signal that has been set is input to the DTP/external interrupt pin, &quot;1&quot; will be set to the DTP/external interrupt request flag bit, even though the DTP/external interrupt request has been disabled (ENIR:EN = 0).</td>
</tr>
</tbody>
</table>

---

**Table 13.3-5 Functions of Detection Level Setting Register (ELVR) (High)**

<table>
<thead>
<tr>
<th>DTP/External Interrupt Pins</th>
<th>Bit name</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT4</td>
<td>LB4, LA4</td>
</tr>
<tr>
<td>INT5</td>
<td>LB5, LA5</td>
</tr>
<tr>
<td>INT6</td>
<td>LB6, LA6</td>
</tr>
<tr>
<td>INT7</td>
<td>LB7, LA7</td>
</tr>
</tbody>
</table>
13.3.4 Detection Level Setting Register (ELVR) (Low)

The detection level setting register (lower) sets the levels or edges of input signals that cause interrupt factor in INT3 to INT0 of the DTP/external interrupt pins.

**Detection Level Setting Register (ELVR) (Low)**

![Detection Level Setting Register (ELVR) (Low)](image)

Table 13.3-7 Functions of Detection Level Setting Register (ELVR) (Low)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0 to bit7</td>
<td>LB3, LA3 to LB0, LA0: Detection condition selection bits</td>
</tr>
</tbody>
</table>

These bits set the levels or edges of input signals from external peripheral devices that cause interrupt factors in the DTP/external interrupt pins.

- Select the external interrupt from 2 level types and 2 edge types, or the EI2OS from 2 level types.

**Reference:**

If the detection signal that has been set is input to the DTP/external interrupt pin, "1" will be set to the DTP/external interrupt request flag bit, even though the DTP/external interrupt request has been disabled (ENIR:EN = 0).

Table 13.3-8 Correspondence between Detection Level Setting Register (ELVR) (Low) and Channels

<table>
<thead>
<tr>
<th>DTP/External Interrupt Pins</th>
<th>Bit name</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT0</td>
<td>LB0, LA0</td>
</tr>
<tr>
<td>INT1</td>
<td>LB1, LA1</td>
</tr>
<tr>
<td>INT2</td>
<td>LB2, LA2</td>
</tr>
<tr>
<td>INT3</td>
<td>LB3, LA3</td>
</tr>
</tbody>
</table>
13.4 **Explanation of Operation of DTP/External Interrupt**

The DTP/external interrupt has an external interrupt function and a DTP function. The setting and operation of each function are explained.

### Setting of DTP/External Interrupt

Using the DTP/external interrupt requires, the setting shown in Figure 13.4-1.

**Figure 13.4-1 Setting of DTP/External Interrupt**

<table>
<thead>
<tr>
<th>ICR Interrupt control register</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>bit8</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>External interrupt DTP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>EIRR/ENIR</td>
<td>ER7</td>
<td>ER6</td>
<td>ER5</td>
<td>ER4</td>
<td>ER3</td>
<td>ER2</td>
<td>ER1</td>
<td>ER0</td>
<td>EN7</td>
<td>EN6</td>
<td>EN5</td>
<td>EN4</td>
<td>EN3</td>
<td>EN2</td>
<td>EN1</td>
<td>EN0</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ELVR</td>
<td>LB7</td>
<td>LA7</td>
<td>LB6</td>
<td>LA6</td>
<td>LB5</td>
<td>LB4</td>
<td>LB3</td>
<td>LB2</td>
<td>LB1</td>
<td>LA1</td>
<td>LB0</td>
<td>LA0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DDR Port direction register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- : Undefined bit
○ : Used bit
○ : Set the bit corresponding to the using pin to "1".
0 : Set to "0"
1 : Set to "1"

#### Setting procedure

To use the DTP/external interrupt, set each register by using the following procedure:

1. Set the interrupt request enable bit that supports the DTP/external interrupt channel to be used to "0". (ENIR: EN)
2. Set the edge or level to be detected using the detection condition selection bit that supports the DTP/external pin to be used. (ELVR: LA,LB)
3. Set the interrupt request flag bit that supports the DTP/external interrupt channel to be used to "0". (EIRR: ER)
4. Set the interrupt request enable bit that supports the DTP/external interrupt channel to be used to "1". (ENIR: EN)

- Set the register for the DTP/external interrupt after setting the external interrupt request to disable (ENIR:EN = 0).
- When the DTP/external interrupt request is enabled (ENIR:EN = 1), first clear the supported DTP/external interrupt request flag bit (EIRR:ER = 0). This is to prevent erroneous generation of interrupt requests when the register is set up.
Selecting of DTP or external interrupt function

Whether the external interrupt function or DTP function will be executed is determined by the setting of the EI²OS enabling bit of the supported interrupt control register (ICR:ISE).

If the ISE bit is set to "1", the EI²OS is enabled and the DTP function is executed.

If the ISE bit is set to "0", the EI²OS is disabled and the external interrupt function is executed.

Notes:

- Interrupt levels (IL2 to IL0) for all interrupt requests that have been allocated to one interrupt control register will be the same.
- When a number of interrupt requests are allocated to one interrupt control register, if EI²OS is used on one of them, other interrupt requests cannot be used.

DTP/External Interrupt Operation

The control bits and the interrupt factors for the DTP/external interrupt are shown in Table 13.4-1.

Table 13.4-1 Control Bits and Interrupt Factors for DTP/External Interrupt

<table>
<thead>
<tr>
<th>Interrupt request flag bit</th>
<th>EIRR: ER7 to ER0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt request enable bit</td>
<td>ENIR: EN7 to EN0</td>
</tr>
<tr>
<td>Interrupt factor</td>
<td>Input of valid edge/level to INT7 to INT0</td>
</tr>
</tbody>
</table>

When an interrupt request of the DTP/external interrupt is output to the interrupt controller, interrupt processing is executed while the EI²OS enabling bit in the interrupt control register (ICR:ISE) is set to "0". The extended intelligent I/O service (EI²OS) is executed while it is set to "1".
Figure 13.4-2 shows operation of DTP/external interrupt.

**Figure 13.4-2 Operation of DTP/External Interrupt**

- DTP/External interrupt circuit
- Interrupt controller
- CPU
- Memory Peripheral data transfer
- Update descriptor
- Interrupt processing
- Interrupt flag clear
- Returning from External interrupt
- Re-setting or Stop
- Return from EI2OS processing (DTP processing)
- Processing and interrupt flag clear
- Activate External interruption processing
- Descriptors data counter
- =0
- ≠0
- ICR : ISE
- ICR YY
- ICR xx
- IL
- ILM
- CMP
- EIRR
- ELVR
- ENIR
- Factor
- DTP/External interrupt request generation
- Judging interrupt controller acceptance
- Judging CPU interrupt acceptance
- Interrupt processing microcoprogram activate
- EI2OS start
- =0
- ≠0
- =1
- Returning from DTP processing
- Returning from EI2OS processing (DTP processing)
13.4.1 External Interrupt Function

There is an external interrupt function that generates interrupt requests by detecting signals (edge or level) using the DTP/external interrupt pin for the DTP/external interrupt function.

External Interrupt Function

- "1" is set to the interrupt request flag bit of the DTP/external interrupt cause register (EIRR:ER) when the signal (edge or level) that has been set to the DTP/external interrupt pin by the detection level setting register is detected.
- When "1" is set to the interrupt request flag bit, generation of interrupt request is notified to the interrupt controller while the interrupt request enable bit of the DTP/external interrupt enable register is enabled (ENIR:EN = 1).
- Interrupt request is generated when the interrupt controller determines that the interrupt priority rank is higher than other interrupt requests.
- The condition code register's interrupt level mask bit (CCR:ILM) and interrupt request level (ICR:IL) are compared on the CPU, and when the interrupt request level is higher than ILM while the interrupt enabling bit is enabled (PS:CCR:I = 1), interrupt processing is executed after the command that is being executed ends, and branches off to interrupt processing.
- Set the supported DTP/external interrupt request flag bit to "0" for interrupt processing, and clear the DTP/external interrupt request to "0".

Notes:

- "1" will be set to the DTP/external interrupt request flag bit (EIRR:ER) when a DTP/external interrupt activation factor is generated regardless of the setting of the supported DTP/external interrupt request enabling bit (ENIR:EN).
- When interrupt processing is activated, clear the DTP/external interrupt request flag bit that caused activation to "0". It cannot be returned from the interrupt while the DTP/external interrupt request flag bit is set to "1". When clearing to "0", do not clear flag bits other than the DTP/external interrupt factor that has been accepted.
13.4.2 DTP Function

The DTP/external interrupt has a DTP function that activates the extended intelligent I/O service by detecting signals from the external peripheral device using the DTP/external interrupt pin.

■ DTP Function

The DTP function detects the signal level set by the detection level setting register of the DTP/external interrupt function to start the EI²OS.

- While operation of the EI²OS is enabled (ICR:ISE = 1), the EI²OS is activated and data transfer starts when an interrupt request is generated.
- When 1 data transfer ends, the descriptor is updated, the DTP/external interrupt request flag bit is cleared and prepared for the next request from the DTP/external interrupt pin.
- When all transfers from the EI²OS end, will branch off to interrupt processing.

Figure 13.4-3 Example of Interface with External Peripheral Device

*1: After starting transfer, cancel within 3-machine clock
*2: When the extended intelligent I/O service is "peripheral -> internal memory transfer".
13.5 Precautions when Using DTP/External Interrupt

Care must be taken concerning the following points when DTP/external interrupt is used.

- **Precautions when Using**
  - **Condition of external-connected peripheral device when DTP function is used**
    - When the DTP function is used, it must be the peripheral device that automatically clears data transfer requests by performing the transfer.
    - Change the transfer request signal to inactive within 3 machine clock after transfer is started. This will handle as if the following transfer request has been generated for the DTP/external interrupt when this is active.

- **External interrupt input polarity**
  - At least 3 machine clock are required for the pulse width to detect the edge when edge detection is set by the detection level setting register.
  - While level detection is set by the detection level setting register, when the level to be the interrupt cause is input, "1" is set to the factor F/F within the DTP/external interrupt cause register, and the factor will be retained as per Figure 13.5-1.

When the factor is retained by the factor F/F, even if the DTP/external interrupt request is withdrawn, the request to the interrupt controller remains active under the interrupt request enabled status (ENIR:EN = 1). Clear the interrupt request flag bit (EIRR:ER), to withdraw the request to the interrupt controller, and clear the factor F/F as shown in Figure 13.5-2.

**Figure 13.5-1 Clearing Factor Hold Circuit when Level Set**

**Figure 13.5-2 DTP/External Interrupt Factor and Interrupt Request Generated when Interrupt Request Enabled**
● Return of Standby Mode
  • The standby mode can be canceled by inputting the interrupt request of the external interrupt that has been set before entering the standby mode. "H" level, "L" level, rising edge, and falling edge can be selected as interrupt requests.

● Precautions on interrupts
  • When this is used as the external interrupt function, "1" is set to the DTP/external interrupt request flag bit (EIRR:ER), and if the DTP/external interrupt request is enabled (ENIR:EN = 1), returning from interrupt processing is impossible. Always set the DTP/external interrupt request flag bit to "0" (EIRR:ER) at interrupt processing.
  • When level detection is set by the detection level setting register, while the status under which the level to be the interrupt factor has been input is retained, even if the DTP/external interrupt request flag bit is cleared (EIRR:ER = 0), reset will be carried out immediately. Disable the DTP/external interrupt request (ENIR:EN = 0) on demand, or cancel the interrupt cause itself.
13.6 Program Example of DTP/External Interrupt Circuit

This section gives a program example of the DTP/external interrupt function.

Program Example of DTP/External Interrupt Function

- Processing specification
  An external interrupt is generated by detecting the rising edge of the pulse input to the INT0 pin.

- Coding example

  ICR01 EQU 0000B1H ; DTP/external interrupt control register
  DDR0  EQU 000010H ; Port 0 direction register
  ENIR  EQU 000030H ; DTP/external interrupt enable register
  EIRR  EQU 000031H ; DTP/external interrupt factor register
  ELVRL EQU 000032H ; Detection level setting register: L
  ELVRH EQU 000033H ; Detection level setting register: H
  ER0   EQU EIRR:0  ; INT0 interrupt request flag bit
  EN0   EQU ENIR:0  ; INT0 interrupt request enable bit

;---------Main Program-------------------------------------------------
CODE CSEG
START:

;Stack pointer (SP)
;Already initialized

  MOV  I:DDR0,#00000000B ; DDR0 is set to the input port.
  AND  CCR,#0BFH ; Interrupt is disabled
  MOV  I:ICR01,#00H ; Interrupt levels 0 (strength)
  CLRB I:EN0 ; INT0 is disabled by the ENIR.
  MOV  I:ELVRL,#00000010B; INT0 indicates rising edge selection.
  CLRB I:ER0 ; INT0 interrupt request flag by the EIRR.
  ;Clear
  SETB I:EN0 ; INT0 interrupt request is enabled by the ENIR.
  MOV  ILM,#07H ; ILM within the PS is set to level 7
  OR   CCR,#40H ; Interrupt is enabled

LOOP:

  • User processing
  •
  BRA LOO%

;---------Interrupt Program---------------------------------------------
WARI:

  CLRB I:ER0 ; Clears the interrupt request flag.
  • User processing
  •
  RETI ; Returning from interrupt processing

CODE ENDS

;---------Vector Settings-----------------------------------------------
VECT CSEG ABS=0FFH

ORG 00FFC8H ; vector set in interrupt number #13(0Dh)
DSL WARI
CHAPTER 13  DTP/EXTERNAL INTERRUPT

ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START

Program sample of DTP Function

Processing specification

- Channel 0 of extended intelligent I/O service (EI²OS) is started upon detection of the High level of the signal input to the INT0 pin.
- RAM data is output to port 1 by DTP processing (EI²OS).

Coding example

```
ICR01 EQU 0000B1H ;DTP/external interrupt control register
DDR0 EQU 000010H ;Port 0 direction register
DDR1 EQU 000011H ;Port 1 direction register
ENIR EQU 0000030H ;DTP/external interrupt enable register
EIRR EQU 0000031H ;DTP/external interrupt factor register
ELVRL EQU 0000032H ;Detection level setting register: L
ELVRH EQU 0000033H ;Detection level setting register: H
ER0 EQU EIRR:0 ;INT0 interrupt request flag bit
EN0 EQU ENIR:0 ;INT0 interrupt request enable bit
BAPL EQU 000100H ;Buffer address pointer lower
BAPM EQU 000101H ;Buffer address pointer middle
BAPH EQU 000102H ;Buffer address pointer upper
ISCS EQU 000103H ;EI²OS status register
IOAL EQU 000104H ;I/O address register lower
IOAH EQU 000105H ;I/O address register upper
DCTL EQU 000106H ;Data counter lower
DCTH EQU 000107H ;Data counter upper

;---------Main program-------------------------------------------------
CODE CSEG
START:

; Stack pointer (SP)
; Already initialized
MOV I:DDR0,#00000000B ;Set up to the input port using the DDR0
MOV I:DDR1,#11111111B ;Set up to the output port by the DDR1,
AND CCR,#0BFH ;Interrupt is disabled
MOV I:ICR01,#08H ;interrupt level 0 (strength) EI²OS channel 0
;Data bank register (DTB) = 00H
MOV BAPL,#00H ;Set up of the storage destination address for output data
MOV BAPM,#06H ;(uses 60H to 60AH)
MOV BAPH,#00H
MOV ISCS,#00010010B ;byte transfer, buffer address + 1,
;Fix the I/O address, memory→transfer to the I/O
MOV IOAL,#01H ;as the transfer destination address pointer
MOV IOAH,#00H ;Port 1 is set.
```
MOV DCTL,#0AH ;The number of transfers is set to 10 times.
MOV DCTH,#00H
CLR I:EN0 ;INT0 is disabled by the ENIR.
MOV I:ELVRL,#00000001B ;INT0 sets the "H" level detection.
CLR I:ER0 ;Clear the interrupt request flag of the INT0 by
SET I:EN0 ;INT0 interrupt request is enabled by the ENIR.
MOV ILM,#07H ;ILM within the PS is set to level 7
OR CCR,#40H ;Interrupt is enabled

LOOP:
• User processing
•
BRA LOOP

;---------Interrupt Program----------------------------------------------

WARI:
CLR I:ER0 ;INT0 clears the interrupt request flag.
• User processing
•
RETI ;Returning from interrupt processing

;---------Vector Settings------------------------------------------------

VECT CSEG ABS=OFFH
ORG 00FFC8H ;vector set in interrupt number #13(0Dh)
DSL WAR1
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode

VECT ENDS
END START
CHAPTER 14
WAKE-UP INTERRUPT

This chapter describes the functions and operation of the wake-up Interrupt

14.1 Wake-up Interrupt Outline
14.2 Block Diagram of Wake-up Interrupt
14.3 Wake-up Interrupt Configuration
14.4 Explanations for Wake-up Interrupt Operation
14.5 Precautions when Using Wake-up Interrupt
14.6 Program Examples for Wake-up Interrupt
14.1 Wake-up Interrupt Outline

Under wake-up interrupt, an interrupt request that generates external peripheral equipment ("L" level input to the wake-up interrupt input pin) is detected, and interrupts processing are activated by transfer to the CPU.
- Standby mode can be cancelled using a wake-up interrupt.
- The wake-up interrupt does not correspond to the Ei²OS.

### Wake-up interrupt function

The wake-up interrupt function generates an interrupt request by defining input of the "L" level signal to the wake-up interrupt input pin as an interrupt factor.

Standby mode can be cancelled using the wake-up interrupt function.

Table 14.1-1 shows wake-up interrupt outline.

<table>
<thead>
<tr>
<th>Function and Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input Pin</strong></td>
</tr>
<tr>
<td>8 pins (WI0 to WI7)</td>
</tr>
<tr>
<td><strong>Interrupt Generation Factor</strong></td>
</tr>
<tr>
<td>&quot;L&quot; level signal input, but OR connection for 8 input</td>
</tr>
<tr>
<td><strong>Interrupt Control</strong></td>
</tr>
<tr>
<td>Interrupt requests are enabled or disabled using the wake-up interrupt control register (EICR).</td>
</tr>
<tr>
<td><strong>Interrupt flag</strong></td>
</tr>
<tr>
<td>Retention of interruption factors using the wake-up interrupt flag register (EIFR)</td>
</tr>
<tr>
<td><strong>Ei²OS</strong></td>
</tr>
<tr>
<td>The wake-up interrupt does not correspond to the Ei²OS.</td>
</tr>
</tbody>
</table>
14.2 Block Diagram of Wake-up Interrupt

A wake-up interrupt is made up of the following blocks.
- Interrupt request detection circuit
- Wake-up interrupt control register (EICR)
- Wake-up interrupt flag register (EIFR)

- Block diagram of wake-up interrupt

![Figure 14.2-1 Block diagram of wake-up interrupt]

- Interrupt request detection circuit
  When the "L" level signal is input from the pin on which a wake-up interrupt request has been enabled (EICR:EN = 1), the interrupt request flag bit of the wake-up interrupt flag register will be set to "1". (EIFR: WIF=1)

- Wake-up interrupt control register (EICR)
  Enabling or disabling wake-up interrupt requests can be set per wake-up interrupt input pin.

- Wake-up interrupt flag register (EIFR)
  For the wake-up interrupt flag register, check generation status of wake-up interrupt request using the interrupt request flag bit, and set the flag bit (WIF) to "0" to cancel interrupt request.
Details of Pins and Interrupt Numbers

The pin to be used for wake-up interrupt and interruption number are indicated next.

WI0 pin: P10/WI0
WI1 pin: P11/WI1
WI2 pin: P12/WI2
WI3 pin: P13/WI3
WI4 pin: P14/WI4
WI5 pin: P15/WI5
WI6 pin: P16/WI6
WI7 pin: P17/WI7
Interrupt number: #16 (10H)
14.3 Wake-up Interrupt Configuration

Pin and register details for wake-up interrupt are described.

- **Pins of Wake-up Interrupt**
  
The pin to be used for wake-up interrupts is also used for the general-purpose input/output port. The pin functions and settings to be used for wake-up interrupts are shown in Table 14.3-1.

  **Table 14.3-1 Pins of Wake-up Interrupt**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting required for using the wake-up interrupt pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>P10/WI0</td>
<td>General-purpose I/O port/Wake-up interrupt input</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P11/WI1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P12/WI2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P13/WI3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P14/WI4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P15/WI5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P16/WI6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P17/WI7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Block Diagram for Pins of Wake-up Interrupt**

  Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".
14.3.1 Wake-up Interrupt Flag Register (EIFR)

The wake-up interrupt flag register can check the generation status of wake-up interrupt requests using the interrupt request flag bit. Set the interrupt request flag bit to "0" to cancel the interrupt request.

![Figure 14.3-1 Wake-up interrupt flag register (EIFR)](image)

### Table 14.3-2 Function of Wake-up interrupt flag register (EIFR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8</td>
<td>WIF: Interrupt request flag bit</td>
</tr>
<tr>
<td></td>
<td>When the &quot;L&quot; level signal is input to any of the wake-up interrupt input pins (WI0 to WI7) for which wake-up interrupt requests have been enabled (EICR:EN = 1), will be set to &quot;1&quot;. <strong>Write:</strong> This WIF bit will be cleared to &quot;0&quot; in each case when setting to &quot;0&quot; or &quot;1&quot;.</td>
</tr>
<tr>
<td>bit9 to bit15</td>
<td>Undefined bits</td>
</tr>
<tr>
<td></td>
<td><strong>Read:</strong> The value is undefined. <strong>Write:</strong> No effect</td>
</tr>
</tbody>
</table>
14.3.2 Wake-up Interrupt Control Register (EICR)

Wake-up interrupt control register (EICR) sets enabling or disabling of the wake-up interrupt request.

- Wake-up interrupt control register (EICR)

Figure 14.3-2 Wake-up interrupt control register (EICR)

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN7</td>
<td>EN6</td>
<td>EN5</td>
<td>EN4</td>
<td>EN3</td>
<td>EN2</td>
<td>EN1</td>
<td>EN0</td>
</tr>
<tr>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>

Reset Value: 00000000b

bit8 to bit15

<table>
<thead>
<tr>
<th>EN0 to EN7</th>
<th>Interrupt Request Flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Wake-up interruption request to disable</td>
</tr>
<tr>
<td>1</td>
<td>Wake-up interrupt request has been enabled</td>
</tr>
</tbody>
</table>

W: Writ Only

Table 14.3-3 Support between the wake-up interrupt input pin and each bit of the wake-up interrupt control register (EICR)

<table>
<thead>
<tr>
<th>No.</th>
<th>Bit name</th>
<th>Wake-up interrupt input pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8</td>
<td>EN0</td>
<td>W10</td>
</tr>
<tr>
<td>bit9</td>
<td>EN1</td>
<td>W11</td>
</tr>
<tr>
<td>bit10</td>
<td>EN2</td>
<td>W12</td>
</tr>
<tr>
<td>bit11</td>
<td>EN3</td>
<td>W13</td>
</tr>
<tr>
<td>bit12</td>
<td>EN4</td>
<td>W14</td>
</tr>
<tr>
<td>bit13</td>
<td>EN5</td>
<td>W15</td>
</tr>
<tr>
<td>bit14</td>
<td>EN6</td>
<td>W16</td>
</tr>
<tr>
<td>bit15</td>
<td>EN7</td>
<td>W17</td>
</tr>
</tbody>
</table>
### Table 14.3-4  Function of wake-up interrupt control register (EICR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 to bit15</td>
<td>EN7 to EN0: Interrupt request enable bit</td>
</tr>
</tbody>
</table>

Enabling or disabling interrupt requests can be set up for the wake-up interrupt input pins WI0 to WI7.

- **The bit is set to "0":** The supported pin functions as a general-purpose input/output port, and wake-up interrupt requests will not be accepted.
- **The bit is set to "1":** The supported pin functions as an input pin for wake-up interrupts, and wake-up interrupt requests are accepted.

**Note:**

Set "0" to the WIF bit to cancel the interrupt request.

**Reference:**

1) Set "0" to the bit supported by the port direction register, and set the input pin to use the wake-up interrupt input pin.
2) The wake-up interrupt pin status can be read directly using the port data register regardless of the interrupt request enabling bit status (EICR:EN).
14.4 Explanations for Wake-up Interrupt Operation

Wake-up interrupt generates an interrupt request to the CPU when "L" level is detected using the wake-up interrupt input pin. The extended intelligent I/O service (EI²OS) cannot be activated.

Explanations for wake-up interrupt operation

Setting of Figure 14.4-1 is required to operate wake-up interrupts.

Figure 14.4-1 Setting of the wake-up interrupt function

<table>
<thead>
<tr>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>bit8</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIFR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WIF</td>
</tr>
<tr>
<td>EICR</td>
<td>EN7</td>
<td>EN6</td>
<td>EN5</td>
<td>EN4</td>
<td>EN3</td>
<td>EN2</td>
<td>EN1</td>
</tr>
<tr>
<td>DDR port direction register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
- : Undefined bit
\(\odot\) : Used bit
1 : register to "1"

The bit corresponding with the pin as using the wake-up interrupt input is set to "0".

Outline of the wake-up interrupt operation

Interrupt factors and control bits for wake-up interrupts are shown in Table 14.4-1.

Table 14.4-1 Control bit and interrupt factors for wake-up interrupts

<table>
<thead>
<tr>
<th>Control bit and factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt request flag bit</td>
</tr>
<tr>
<td>Interrupt request enable bit</td>
</tr>
<tr>
<td>Interrupt factor</td>
</tr>
</tbody>
</table>

- When wake-up interrupt request is enabled (EICR:EN7 to EN0 = 1), and an "L" level signal is input to one of the wake-up interrupt input pins, the interrupt request flag bit is set to "1" (EIFR:WIF = 1), and an interrupt request is output to the CPU.
- When the interrupt controller determines that an interrupt priority rank is higher than other interrupt requests, an interrupt request is generated.
Figure 14.4-2 shows wake-up interrupt operation.

Figure 14.4-2 Wake-up interrupt operation

Note: When the wake-up interrupt input pin retains "L" level, set the wake-up interrupt request to disable (EICR:EN = 0), and then set the WIF bit to "0". Even if the WIF bit is set to "0" without setting the wake-up interrupt request to disable, the WIF bit will be set to "1" immediately.
14.5 Precautions when Using Wake-up Interrupt

Care must be taken with the use of wake-up interrupts.

■ Precautions when using wake-up interrupt

● Setting Input Pin

Input pins for wake-up interrupts (WI0 to WI7) are also used for the general-purpose input/output port. When a wake-up interrupt request is detected, set the relevant pin as the input pin.

● Checking interrupt factors

Interrupt request flag sets "1" to the interrupt request flag bit if a "L" level signal is input to one or more wake-up interrupt input pins for which interrupt requests have been enabled. (EIFR: WIF=1)

Check which pin was inputted interrupt request by retaining the interrupt input level that is input to the pin by adding an external circuit, and reading the port data register.
14.6 Program Examples for Wake-up Interrupt

Program examples for wake-up interrupts are shown.

### Program examples for wake-up interrupt

#### Processing specification

A "L" level signal to be input to the WI0 pin is detected, and an interrupt is generated.

#### Coding example

```
ICR02  EQU 0000B2H ;Interrupt control register for wake-up
                   ;interrupts
DDR1   EQU 000011H ;Port 1 direction register
EICR   EQU 00001FH ;Wake-up interrupt control register
EIFR   EQU 00000FH ;Wake-up interrupt flag register
WIF    EQU EIFR:0 ;Wake-up interrupt request flag bit

;---------Main Program-------------------------------------------------
CODE CSEG
START:

; Stack pointer (SP)
; Already initialized
MOV I:DDR1,#00000000B ;DDR1 is set to input.
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR02,#00H ;Interrupt levels 0 (strength)
CLRB I:WIF ;Wake-up interrupt request flag is cleared.
MOV ILM,#07H ;ILM within the PS is set to level 7
OR CCR,#40H ;Interruption is enabled
MOV I:EICR,#01H ;Enables wake-up interrupts

LOOP:

; User processing
BRA LOOP

;---------Interrupt Program---------------------------------------------
WARI:

CLRB I:WIF ;Wake-up interrupt request flag is cleared.

; User processing

RETI ;Returning from interrupt processing

;---------Vector Settings----------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFBCH ;vector set in interrupt number #16(10H)
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Setting to Single-chip mode

VECT ENDS
END START
```
CHAPTER 15

8/10-BIT A/D CONVERTER

This chapter explains the functions and operation of 8-/10-bit A/D converter.

15.1 Overview of 8-/10-bit A/D Converter
15.2 Block Diagram of 8-/10-bit A/D Converter
15.3 Configuration of 8-/10-bit A/D Converter
15.4 Interrupt of 8-/10-bit A/D Converter
15.5 Explanation of Operation of 8-/10-bit A/D Converter
15.6 Precautions when Using 8-/10-bit A/D Converter
15.7 Example of program in A/D converter-1
   (Example of EI^2OS start program in single mode)
15.8 Example of program in A/D converter-2
   (Example of EI^2OS start in continuos mode)
15.9 Example of program in A/D converter-3
   (Example of EI^2OS start in stop mode)
15.1 Overview of 8-/10-bit A/D Converter

The 8-/10-bit A/D converter converts the analog input voltage to a 8-or 10-bit digital value by using the RC sequential-comparison converter system.
- An input signal can be selected from the input signals of the analog input pins for 8 channels.
- Activation triggers for A/D conversion can be selected from the software trigger, internal timer output, and external pin trigger.

Functions of the 8-/10-bit A/D converter

The 8-/10-bit A/D converter converts the analog voltage (input voltage) input to the analog input pin into an 8-or 10-bit digital value (A/D conversion).

The 8-/10-bit A/D converter has the following functions:
- A/D conversion time is at least 99/φ + 64/φ including the sampling time.
  Example of calculation: When the machine clock frequency is 16MHz, 99/φ + 64/φ = 10.18μs
- Sampling time is at least 64/φ.
  Example of calculation: When the machine clock frequency is 16MHz, 64/φ = 4μs
- RC sequential-comparison converter system with sample & hold circuit
- Setting of 8-bit or 10-bit resolution enabled
- Up to 8 channels can be used for the analog input pin.
- Interrupt request can be generated with the timing at which the A/D conversion results were stored to the A/D data register.
- When interrupt requests are generated, the extended intelligent I/O service function (EI²OS) can be activated. When the EI²OS is used, data is not missed even if A/D conversion is continuously carried out.
- The activation trigger can be selected from the software trigger, internal timer output, and external pin trigger.

Conversion Modes of 8-/10-bit A/D Converter

There are conversion modes of 8-/10-bit A/D converter as shown below:

<table>
<thead>
<tr>
<th>Conversion Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-shot conversion mode</td>
<td>When a start-up trigger is input, A/D conversion is carried out continuously for analog input from the start to the end channels. When A/D conversion for the end channel is terminated, it stops.</td>
</tr>
<tr>
<td>Continuous conversion mode</td>
<td>When a start-up trigger is input, A/D conversion is carried out continuously for analog input from the start to the end channels. When A/D conversion for the end channel is terminated, it is continued after returning to the analog input for the start channel.</td>
</tr>
<tr>
<td>Pause-conversion mode</td>
<td>When a start-up trigger is input, A/D conversion is carried out suspending per channel. When A/D conversion of the end channel ends, A/D conversion is suspended, while A/D conversion continues by returning to analog input of the start channel and suspending per channel.</td>
</tr>
</tbody>
</table>
15.2 Block Diagram of 8-/10-bit A/D Converter

This section shows Block Diagram of 8-/10-bit A/D Converter

**Figure 15.2-1 Block Diagram of 8-/10-bit A/D Converter**

- **A/D control status register (ADCS)**: Interrupt request output
- **Start up selector**: Analog channel selector
- **Comparator**: Control circuit
- **D/A converter**: Analog input
- **AN0 to AN7**: Analog input
- **AVRH, AVRL**: Vref+ input pin
- **AVcc, AVss**: Vcc input pin
- **ADTG**: External trigger input
- **TO**: Internal timer output
- **Reserved**: Be sure to set to "0".
- **Machine clock**: φ
- **Unused**: –

Legend:
- BUSY
- INT
- INTE
- PAUS
- STS1
- STS0
- STRT
- MD1
- MD0
- ANS2
- ANS1
- ANS0
- ANE2
- ANE1
- ANE0
- **Internal data bus**
• Details of pins in block diagram

Table 15.2-1 shows pin name of 8-/10-bit A/D converter and interrupt request number.

<table>
<thead>
<tr>
<th>Pin Name/Interrupt Request Number in Block Diagram</th>
<th>Actual Pin Name/Interrupt Request Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADTG</td>
<td>P27/ADTG</td>
</tr>
<tr>
<td>TO</td>
<td>TO 16-bit reload timer 1</td>
</tr>
<tr>
<td>AN0</td>
<td>P60/AN0</td>
</tr>
<tr>
<td>AN1</td>
<td>P61/AN1</td>
</tr>
<tr>
<td>AN2</td>
<td>P62/AN2</td>
</tr>
<tr>
<td>AN3</td>
<td>P63/AN3</td>
</tr>
<tr>
<td>AN4</td>
<td>P64/AN4</td>
</tr>
<tr>
<td>AN5</td>
<td>P65/AN5</td>
</tr>
<tr>
<td>AN6</td>
<td>P66/AN6</td>
</tr>
<tr>
<td>AN7</td>
<td>P67/AN7</td>
</tr>
<tr>
<td>AVRH</td>
<td>AVRH</td>
</tr>
<tr>
<td>AVRL</td>
<td>AVRL</td>
</tr>
<tr>
<td>AVcc</td>
<td>AVCC</td>
</tr>
<tr>
<td>AVss</td>
<td>AVSS</td>
</tr>
<tr>
<td>Interrupt Request Number</td>
<td>#11(0B_H)</td>
</tr>
</tbody>
</table>

• A/D control status registers (ADCS)

Performs initiation of A/D conversion using the software, selection of the activation trigger for A/D conversion, enabling or disabling interrupt request, checking and clearing the interrupt request flag, suspending the A/D conversion operation and checking status during conversion, selection of the conversion mode, and setting up of the start and end channels for A/D conversion.

• A/D data registers (ADCR)

This register stores the A/D conversion results. Select the compare time (comparison time) for A/D conversion, sampling time, and resolution as well.

• Start selector

Activation trigger for A/D conversion is selected. The activation trigger can be selected from the internal timer output, external pin trigger, or software trigger.

• Decoder

The analog input pin to be used for A/D conversion can be selected from the A/D conversion start channel selection bit (ADCS:ANS2 to ANS0) and the A/D conversion end channel selection bit (ADCS:ANE2 to ANE0) settings of the A/D control status register.
● Analog channel selector

The pin to be used for A/D conversion can be selected from the 8-channel analog input pins (AN7 to AN0) by receiving the signal from the decoder.

● Sample & hold circuit

The analog input voltage selected by the analog channel selector is retained. Conversion without being affected by the input voltage is possible during A/D conversion by retaining the input voltage just after starting A/D conversion.

● D/A converter

This converter generates the reference voltage (Vcc, Vss, Vref+, Vref-) which is compared with the input voltage held in the sample & hold circuit.

● Comparator

Voltage value is determined by comparing the input voltage retained by the sample & hold circuit and the standard voltage of the D/A converter (Vcc, Vss, Vref+, Vref-).

● Control circuit

A/D conversion value is decided by receiving the large/small signal for voltage from the converter. When the conversion results are decided, conversion results data is stored in the A/D data register. When interrupt is enabled, an interrupt request is generated.
15.3 Configuration of 8-/10-bit A/D Converter

The pin, interrupt factor, and register details of the A/D converter are described.

■ Pins of 8-/10-bit A/D Converter

The pins of the 8-/10-bit A/D converter serve as general-purpose I/O ports. Settings when the 8/10-bit A/D converter pin function and 8/10-bit A/D converter are used are shown in Table 15.3-1.

Table 15.3-1  Pins of 8-/10-bit A/D Converter

<table>
<thead>
<tr>
<th>Function Used</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Set up required for use. of 8/10-bit A/D converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger input</td>
<td>P27/ADTG</td>
<td>General-purpose I/O port, external trigger input</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>Channel 0</td>
<td>P60/AN0</td>
<td>General-purpose I/O ports, analog inputs</td>
<td>Set as input port in port direction register (DDR). Input of analog signal enabled (ADER: ADE7 to ADE0 = 11111111 B)</td>
</tr>
<tr>
<td>Channel 1</td>
<td>P61/AN1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 2</td>
<td>P62/AN2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 3</td>
<td>P63/AN3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 4</td>
<td>P64/AN4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 5</td>
<td>P65/AN5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 6</td>
<td>P66/AN6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel 7</td>
<td>P67/AN7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

■ Diagram of Pins of 8-/10-bit A/D Converter

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

■ Generation of Interrupt from 8-/10-bit A/D Converter

In the 8-/10-bit A/D converter, when the A/D conversion results are stored in the A/D data register (ADCR), the interrupt request flag bit in the A/D control status register (ADCS: INT) is set to 1. Interrupt request is generated when interrupt is enabled (ADCS:INTE = 1).
15.3.1 A/D Control Status Register (High) (ADCS: H)

A/D control status registers (ADCS) upper 8-bit is enable to set following:
- Activation of A/D conversion by the software
- Selection of the activation trigger for A/D conversion
- Enabling or disabling interruption by storing the A/D conversion results in the A/D data register, and check and clear of the interrupt request flag
- Pausing A/D conversion and checking state during conversion

### A/D Control Status Register (High) (ADCS: H)

#### Figure 15.3-1 A/D Control Status Register (High) (ADCS:H)

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSY</td>
<td>INT</td>
<td>INTE</td>
<td>PAUS</td>
<td>STS1</td>
<td>STS0</td>
<td>STRT</td>
<td>Reserved bit</td>
<td></td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>W</td>
<td>00000000B</td>
<td></td>
</tr>
</tbody>
</table>

- **bit8**: Reserved bit
  - **Reserved**: Be sure to set to "0".

- **bit9**: A/D conversion software start up bit
  - **STRT**: Start up the A/D conversion
  - **0**: Not start up of the A/D conversion
  - **1**: Start up the A/D conversion

- **bit10**: A/D conversion start up trigger selection bit
  - **STS1**: Software start up
  - **STS0**: Software start up or external trigger start up
  - **PAUS**: Software start up or internal timer start up
  - **1**: Software start up, external trigger start up or internal timer start up

- **bit11**: Temporarily stop flag bit
  - **PAUS**: Temporarily stop flag bit
    - *(Only available for using EI2OS)*
  - **0**: A/D conversion is not in temporary stop.
  - **1**: A/D conversion is in temporary stop.

- **bit12**: Interrupt enable bit
  - **INTE**: Interrupt enabled
    - **0**: Interrupt disabled
    - **1**: Interrupt enabled

- **bit13**: Interrupt request flag
  - **INT**: Interrupt request flag
    - **0**: In A/D conversion operating
      - Clear to "0"
    - **1**: A/D conversion completion of 1 channel
      - No effect

- **bit14**: Flag bit in operating of A/D conversion
  - **BUSY**: Flag bit in operating of A/D conversion
    - **0**: A/D conversion completed
      - A/D conversion forcibly completed
    - **1**: In operating of A/D conversion
      - No effect
### Table 15.3-2 Function of A/D Control Status Register (High) (ADCS: H) (1 / 2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bit8</strong></td>
<td>Reserved: reserved bit</td>
</tr>
</tbody>
</table>
| **bit9** | STRT: A/D conversion software start bit | This bit starts the 8-/10-bit A/D converter by software.  
* **When the bit is set to "1":** Starts 8-/10-bit A/D converter  
  - When A/D conversion is suspended under the suspension conversion mode, A/D conversion will be re-started by setting this STRT bit to "1".  
  - **When the bit is set to "0":** A/D conversion is not activated.  
* **Note:**  
  - Do not perform forcible termination (BUSY = 0) and software start (STRT = 1) of the 8-/10-bit A/D converter simultaneously.  
  - Do not use commands carrying out read-modify-write (RMW) operations.  
  - The byte/word command reads "1". |
| **bit10** | STS1, STS0: A/D conversion start trigger selection bits | These bits select the trigger to start the 8-/10-bit A/D converter.  
  - When a number of activation triggers are set (except when STS1 and STS0 = "00b"), the 8/10-bit A/D converter is activated by the first activation trigger generated.  
  - **Note:**  
    - Start trigger setting should be changed when the operation of resource generating a start trigger is stopped. |
| **bit11** | PAUS: Pause flag bit | This is only valid when the EF2.OS function is used, and indicates the operating status of the A/D conversion.  
  - The PAUS bit is enabled only when the EF2.OS function is used.  
  - A/D conversion is suspended while the A/D conversion results are transferred from the A/D data register (ADCR) to the memory after 1-channel conversion. When A/D conversion is suspended, "1" will be set to this PAUS bit.  
  - When transfer of the A/D conversion results to the memory ends, A/D conversion is re-started.  
  - When A/D conversion is re-started, this PAUS bit will be cleared to "0". |
| **bit12** | INT: Interrupt request flag bit | This bit indicates that an interrupt request is generated.  
  - "1" will be set to this INT bit after A/D conversion of 1 channel ends, and the A/D conversion results are stored in the A/D data register (ADCR).  
  - While interruption is enabled (INTE = 1), when the interrupt request flag bit is set (INT = 1), an interrupt request is generated.  
  - **When the bit is set to "0":** The bit is cleared.  
  - **When the bit is set to "1":** No effect.  
  - **When EF2.OS function started:** Cleared  
  - **Note:**  
    - Write "0" while operation of the 8/10-bit A/D converter is suspended (ADCS:BUSY = 0) to clear this INT bit. |
Table 15.3-2  Function of A/D Control Status Register (High) (ADCS: H) (2 / 2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit15</td>
<td>BUSY: A/D conversion-on flag bit</td>
</tr>
</tbody>
</table>

This bit forcibly terminates the 8-/10-bit A/D converter. When read, this bit indicates whether the 8-/10-bit A/D converter is operating or stopped.  
**When the bit is set to "0":** Forcibly terminates 8-/10-bit A/D converter  
**When the bit is set to "1":** No effect.  
**Read:** 1 is read when the 8-/10-bit A/D converter is operating and "0" is written when the converter is stopped.  
**Note:**  
Do not perform forcible termination (BUSY = 0) and software start (STRT = 1) of the 8-/10-bit A/D converter simultaneously.
A/D control status registers (ADCS) lower 8-bit is enable to set following.
- Selecting A/D conversion mode
- Selecting start channel and end channel of A/D conversion

## A/D Control Status Register (Low) (ADCS: L)

### Figure 15.3-2 A/D Control Status Register (Low) (ADCS: L)

<table>
<thead>
<tr>
<th>MD1</th>
<th>MD0</th>
<th>ANS2</th>
<th>ANS1</th>
<th>ANS0</th>
<th>ANE2</th>
<th>ANE1</th>
<th>ANE0</th>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>00000000B</td>
</tr>
</tbody>
</table>

- **bit7** bit6: A/D conversion mode selection bit
  - 0 0: Single conversion mode 1 (Be able to restart during operation)
  - 0 1: Single conversion mode 2 (Not be able to restart during operation)
  - 1 0: Sequence conversion mode (Not be able to restart during operation)
  - 1 1: Stop conversion mode (Not be able to restart during operation)

- **bit5** bit4: A/D conversion start channel selection bit
  - 0 0: AN0 pin
  - 0 1: AN1 pin
  - 1 0: AN2 pin
  - 1 1: AN3 pin

- **bit3** bit2: A/D conversion end channel selection bit
  - 0 0: AN0 pin
  - 0 1: AN1 pin
  - 1 0: AN2 pin
  - 1 1: AN3 pin

- **MD1 MD0** ANS2 ANS1 ANS0 ANE2 ANE1 ANE0: A/D conversion mode selection bit
  - No activating state (ADCS.BUSY=0)
  - Read during converting
  - Read during halted in the stop conversion mode
  - Converted channel number immediately before

- **ANS2 ANS1 ANS0** ANE2 ANE1 ANE0: A/D conversion end channel selection bit
  - AN0 pin
  - AN1 pin
  - AN2 pin
  - AN3 pin

- **ANE2 ANE1 ANE0** A/D conversion end channel selection bit
  - AN0 pin
  - AN1 pin
  - AN2 pin
  - AN3 pin

- **bit1** bit0: A/D conversion start channel selection bit
  - 0 0: AN0 pin
  - 0 1: AN1 pin
  - 1 0: AN2 pin
  - 1 1: AN3 pin

- **bit5** bit4 bit3: A/D conversion start channel selection bit
  - 0 0 0: AN0 pin
  - 0 0 1: AN1 pin
  - 0 1 0: AN2 pin
  - 0 1 1: AN3 pin
  - 1 0 0: AN4 pin
  - 1 0 1: AN5 pin
  - 1 1 0: AN6 pin
  - 1 1 1: AN7 pin

- **bit2** R/W: Read/write
  - R/W: Read/write
  - : Reset value

- **R/W**: Read/write
  - : Reset value
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit2 to bit0</td>
<td>ANE2 to ANE0: A/D conversion end channel selection bits</td>
</tr>
<tr>
<td></td>
<td>These bits set the channel at which A/D conversion terminated.</td>
</tr>
<tr>
<td></td>
<td><strong>Start channel &lt; End channel:</strong> A/D Conversion starts from the channel that has been set by the A/D conversion start channel selection bit (ANS2 to ANS0), and A/D conversion ends at the channel that has been set by the A/D conversion end channel selection bit (ANE2 to ANE0).</td>
</tr>
<tr>
<td></td>
<td><strong>Start channel = End channel:</strong> A/D conversion is only carried out for 1 channel that has been set by the A/D conversion end (= start) channel selection bit.</td>
</tr>
<tr>
<td></td>
<td><strong>Start channel &gt; End channel:</strong> A/D conversion is carried out from the channel that has been set by the A/D conversion start channel selection bit to AN7, and moreover, A/D conversion is carried out from AN0 to the channel that has been set by the A/D conversion end channel selection bit.</td>
</tr>
<tr>
<td></td>
<td><strong>In case of continuous conversion mode and suspension conversion mode:</strong> When A/D conversion ends on the channel that has been set by the A/D conversion end channel selection bit, A/D conversion will be performed continuously after returning to the channel that has been set by the A/D conversion start channel selection bit.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Do not set the A/D conversion end channel bits during A/D conversion.</td>
</tr>
<tr>
<td></td>
<td>After setting the start channel to the A/D conversion start channel selection bit (ANS2, ANS1, ANS0), please set neither the A/D conversion mode selection bit (MD1, MD0) nor the A/D conversion end channel selection bit (ANE2, ANE1, ANE0) by the read-modify-write type instruction.</td>
</tr>
<tr>
<td></td>
<td>The last conversion channel is read from the ANS2, ANS1, ANS0 bits until the A/D conversion operating starts. Therefore, when the MD1, MD0 bits and ANE2, ANE1, ANE0 bits are set by the read-modify-write type instruction after setting the start channel to the ANS2, ANS1, ANS0 bits, the value of ANE2, ANE1, ANE0 bits may be re-written.</td>
</tr>
<tr>
<td>bit5 to bit3</td>
<td>ANS2 to ANS0: A/D conversion start channel selection bits</td>
</tr>
<tr>
<td></td>
<td>These bits set the channel at which A/D conversion start. At read, the channel number under A/D conversion or A/D-converted immediately before A/D conversion pauses can be checked. And before A/D conversion starts, the previous conversion channel will be read even if these bits have already been set to the new value. These bits are initialized to “000B” at reset.</td>
</tr>
<tr>
<td></td>
<td><strong>Read (During A/D conversion):</strong> The channel numbers (7 to 0) under A/D conversion are read.</td>
</tr>
<tr>
<td></td>
<td><strong>Read (During Pause-conversion mode and temporary stop):</strong> At read during a pause, the channel number A/D-converted immediately before a pause is read.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Do not set the A/D conversion start channel bits during A/D conversion.</td>
</tr>
</tbody>
</table>
These bits set the A/D conversion mode.

**Single-shot conversion mode 1:**
- A/D conversion will be performed continuously from the start channels (ADCS:ANS2 to ANS0) to the end channels (ADCS:ANE2 to ANE0).
- When A/D conversion of the end channel ends, A/D conversion is suspended (ADCS:BUSY = 0).
- This mode can be restarted during A/D conversion.

**Single-shot conversion mode 2:**
- A/D conversion will be performed continuously from the start channels to the end channels.
- When A/D conversion of the end channel ends, A/D conversion is suspended (ADCS:BUSY = 0).
- This mode cannot be restarted during A/D conversion.

**Continuous conversion mode:**
- A/D conversion will be performed continuously from the start channels to the end channels.
- When A/D conversion of the end channel ends, A/D conversion will be continuously performed by returning to the analog input of the start channel.
- Set the A/D conversion flag bit (ADCS:BUSY) to "0" to forcibly end A/D conversion.
- This mode cannot be restarted during A/D conversion.

**Pause conversion mode:**
- A/D conversion for the start channel starts. A/D conversion will be suspended when A/D conversion for 1 channel ends. A/D conversion of the following channel will be carried out if an activation trigger is input while A/D conversion is suspended.
- When A/D conversion of the end channel ends, A/D conversion is suspended. When the activation trigger is input while A/D conversion is suspended, A/D conversion is continued by returning to the start channel.
- Set the A/D conversion flag bit (ADCS:BUSY) to "0" to forcibly end A/D conversion.
- This mode cannot be restarted during A/D conversion.

**Note:**
- When unable to set conversion mode to re-activation (MD1 and MD0 = other than "00B"), re-initiation during conversion cannot be executed for all activated triggers (software, internal timer, and external trigger).

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit7 bit6</td>
<td>MD1, MD0: A/D conversion mode selection bits</td>
</tr>
<tr>
<td></td>
<td>These bits set the A/D conversion mode.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Single-shot conversion mode 1:</strong></td>
</tr>
<tr>
<td></td>
<td>- A/D conversion will be performed continuously from the start channels (ADCS:ANS2 to ANS0) to the end channels (ADCS:ANE2 to ANE0).</td>
</tr>
<tr>
<td></td>
<td>- When A/D conversion of the end channel ends, A/D conversion is suspended (ADCS:BUSY = 0).</td>
</tr>
<tr>
<td></td>
<td>- This mode can be restarted during A/D conversion.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Single-shot conversion mode 2:</strong></td>
</tr>
<tr>
<td></td>
<td>- A/D conversion will be performed continuously from the start channels to the end channels.</td>
</tr>
<tr>
<td></td>
<td>- When A/D conversion of the end channel ends, A/D conversion is suspended (ADCS:BUSY = 0).</td>
</tr>
<tr>
<td></td>
<td>- This mode cannot be restarted during A/D conversion.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Continuous conversion mode:</strong></td>
</tr>
<tr>
<td></td>
<td>- A/D conversion will be performed continuously from the start channels to the end channels.</td>
</tr>
<tr>
<td></td>
<td>- When A/D conversion of the end channel ends, A/D conversion will be continuously performed by returning to the analog input of the start channel.</td>
</tr>
<tr>
<td></td>
<td>- Set the A/D conversion flag bit (ADCS:BUSY) to &quot;0&quot; to forcibly end A/D conversion.</td>
</tr>
<tr>
<td></td>
<td>- This mode cannot be restarted during A/D conversion.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Pause conversion mode:</strong></td>
</tr>
<tr>
<td></td>
<td>- A/D conversion for the start channel starts. A/D conversion will be suspended when A/D conversion for 1 channel ends. A/D conversion of the following channel will be carried out if an activation trigger is input while A/D conversion is suspended.</td>
</tr>
<tr>
<td></td>
<td>- When A/D conversion of the end channel ends, A/D conversion is suspended. When the activation trigger is input while A/D conversion is suspended, A/D conversion is continued by returning to the start channel.</td>
</tr>
<tr>
<td></td>
<td>- Set the A/D conversion flag bit (ADCS:BUSY) to &quot;0&quot; to forcibly end A/D conversion.</td>
</tr>
<tr>
<td></td>
<td>- This mode cannot be restarted during A/D conversion.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td></td>
<td>- When unable to set conversion mode to re-activation (MD1 and MD0 = other than &quot;00B&quot;), re-initiation during conversion cannot be executed for all activated triggers (software, internal timer, and external trigger).</td>
</tr>
</tbody>
</table>
15.3.3 A/D Data Register (High) (ADCR: H)

The upper 5 bits of the A/D data register (ADCR) select the compare time (comparison time) for A/D conversion, sampling time, and resolution.

A/D Data Register (High) (ADCR: H)

Figure 15.3-3 A/D Data Register (High) (ADCR: H)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>SELB</td>
</tr>
<tr>
<td>14</td>
<td>ST1</td>
</tr>
<tr>
<td>13</td>
<td>ST0</td>
</tr>
<tr>
<td>12</td>
<td>CT1</td>
</tr>
<tr>
<td>11</td>
<td>CT0</td>
</tr>
<tr>
<td>10</td>
<td>+3</td>
</tr>
<tr>
<td>9</td>
<td>+3</td>
</tr>
<tr>
<td>8</td>
<td>00001XXXb</td>
</tr>
</tbody>
</table>

Reset value

- **SELB**: Machine clock
- **ST1/ST0**: Unused
- **CT1/CT0**: Compare time selection bit
- **bit14/bit13**: Sampling time selection bit
- **bit15**: Resolution selection bit
- **bit9/bit8**: A/D conversion data bit

**Compare time selection bit**
- 00: 99/φ (6.18 µs)
- 01: 176/φ (11 µs)
- 10: 44/φ (4.4 µs)
- 11: Setting disabled

**Sampling time selection bit**
- 00: 64/φ (4 µs)
- 01: 128/φ (8 µs)
- 10: Setting disabled
- 11: 4096/φ (256 µs)

**Resolution selection bit**
- 00: 10 bits (D9 to D0)
- 01: 8 bits (D7 to D0)

*1: Values in parentheses ( ) are the calculation example operating at φ = 16 MHz.
*2: Values in parentheses ( ) are the calculation example operating at φ = 10 MHz.
*3: bit9 and bit8 are described in A/D register lower (ADCR: L).
### Table 15.3-4 Functions of A/D Data Register (High) (ADCR: H)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit12, bit11 | CT1, CT0: Compare time selection bits  
These bits set the A/D conversion compare time.  
- The time from when an analog input is A/D converted to when it is stored as an A/D conversion data bit (D9 to D0) is set.  
**Note:**  
CT1 and CT0 = "10B" are set up when operations are performed at 10 MHz or less.  
When operated with a machine clock of 10 MHz or higher, conversion accuracy cannot be guaranteed. |
| bit14, bit13 | ST1, ST0: Sampling time selection bits  
These bits set the A/D conversion sampling time.  
- These bits set the time required from when A/D conversion starts until the input analog voltage is sampled and held by the sample & hold circuit. |
| bit15 | SELB: Resolution selection bit  
This bit selects the A/D conversion resolution.  
**When the bit is set to 0**: Resolution of the A/D conversion is set to the 10 bits of the A/D conversion data bits D9 to D0.  
**When the bit is set to 1**: Resolution of the A/D conversion is set to the 8 bits of the A/D conversion data bits D7 to D0.  
**Note:**  
When this SELB bit is modified, execute it under stop status before starting A/D conversion. When this SELB bit is modified after A/D conversion, conversion results stored in the A/D conversion data bits (D9 to D0) will be invalid. |

**Note:** When data is written to the A/D data register, execute it after the A/D conversion function ends (ADCS:BUSY = 0).
15.3.4 A/D Data Register (Low) (ADCR: L)

A/D conversion result is stored in A/D data register (Low) (ADCR)

Figure 15.3-4 A/D Data Register (Low) (ADCR:L) and Bit 9, 8 of A/D Data Register (High) (ADCR: H)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit9 to bit0</td>
<td>D9 to D0: A/D conversion data bits</td>
</tr>
<tr>
<td></td>
<td>This register stores the A/D conversion results.</td>
</tr>
<tr>
<td></td>
<td><em>When resolution is set to &quot;10&quot; bits (SELB = 0):</em></td>
</tr>
<tr>
<td></td>
<td>Conversion data will be stored on the 10 bits from bits D9 to D0.</td>
</tr>
<tr>
<td></td>
<td><em>When resolution is set to &quot;10&quot; bits (SELB = 1):</em></td>
</tr>
<tr>
<td></td>
<td>Conversion data will be stored on the 8 bits from bits D7 to D0.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td></td>
<td>Use a word instruction (MOVW) to read the conversion results stored in</td>
</tr>
<tr>
<td></td>
<td>the A/D conversion data bits (D9 to D0).</td>
</tr>
</tbody>
</table>

Table 15.3-5 Functions of A/D Data Register (Low) (ADCR: L)
15.3.5 Analog Input Enable Register (ADER)

Analog input of the pin to be used is enabled or disabled using the 8/10-bit A/D converter.

■ Analog input enable register (ADER)

![Figure 15.3-5 Analog input enable register (ADER)](image-url)

- **ADE0**: Analog input enable bit 0 (AN0)
  - 0: Disabling of analog input (AN0 pin)
  - 1: Enabling of analog input (AN0 pin)

- **ADE1**: Analog input enable bit 1 (AN1)
  - 0: Disabling of analog input (AN1 pin)
  - 1: Enabling of analog input (AN1 pin)

- **ADE2**: Analog input enable bit 2 (AN2)
  - 0: Disabling of analog input (AN2 pin)
  - 1: Enabling of analog input (AN2 pin)

- **ADE3**: Analog input enable bit 3 (AN3)
  - 0: Disabling of analog input (AN3 pin)
  - 1: Enabling of analog input (AN3 pin)

- **ADE4**: Analog input enable bit 4 (AN4)
  - 0: Disabling of analog input (AN4 pin)
  - 1: Enabling of analog input (AN4 pin)

- **ADE5**: Analog input enable bit 5 (AN5)
  - 0: Disabling of analog input (AN5 pin)
  - 1: Enabling of analog input (AN5 pin)

- **ADE6**: Analog input enable bit 6 (AN6)
  - 0: Disabling of analog input (AN6 pin)
  - 1: Enabling of analog input (AN6 pin)

- **ADE7**: Analog input enable bit 7 (AN7)
  - 0: Disabling of analog input (AN7 pin)
  - 1: Enabling of analog input (AN7 pin)
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit7 to bit0: ADE7 to ADE0: Analog input enable bits | These bits enable or disable the analog input of the pin to be used for A/D conversion.  
When the bit is set to "0": Disables analog input  
When the bit is set to "1": Enables analog input |

**Notes:**
- The analog input pin is simultaneously used for the general-purpose input/output port. When using the pin as an analog input pin, switch the pin to analog input pin according to the setting of the port direction register (DDR) and the analog input enable register (ADER).
- Set the port direction register that supports the pin to be used to "0" and turn off the output transistor when this is used as the analog input pin. Enable analog input by setting "1" to the analog input enabling register (ADER) bit that supports the pin to be used.
15.4 Interrupt of 8-/10-bit A/D Converter

Interrupt request is generated for the 8/10-bit A/D converter when 1-channel A/D conversion ends and the conversion results are stored in the A/D data register (ADCR). The Ei²OS function can be used.

■ Interrupt of A/D Converter

When A/D conversion of the analog input voltage is terminated and its results are stored in the A/D data register (ADCR), the interrupt request flag bit in the A/D control status register (ADCS: INT) is set to "1". Interrupt request is generated when the interrupt request flag bit (ADCS:INT = 1) is set while interrupt is enabled (ADCS:INTE = 1).

■ 8-/10-bit A/D Converter Interrupt and Ei²OS

Reference: Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

■ Ei²OS Function of 8-/10-bit A/D Converter

In the 8-/10-bit A/D converter, the Ei²OS function can be used to transfer the A/D conversion results from the A/D data register (ADCR) to memory. If the Ei²OS function is used, the A/D-converted data protection function is activated to cause A/D conversion to pause during memory transfer. The A/D-converted data protection function is activated to prevent data loss as A/D conversion is performed continuously.
15.5 Explanation of Operation of 8-/10-bit A/D Converter

Conversion of 8/10-bit A/D converter has following conversion modes.
- Single-shot conversion mode (restartable/not-restartable during A/D conversion)
- Continuous conversion mode (not-restartable during A/D conversion)
- Pause conversion mode (not-restartable during A/D conversion)

Set each mode according to the setting of the A/D conversion mode selection bits in the A/D control status register (ADCS: MD1, MD0).

■ Single-shot Conversion Mode (ADCS: MD1, MD0 = "00B" or "01B")
  - When the startup trigger is entered, A/D conversion of analog inputs is performed continuously from start channels (ADCS: ANS2 to ANS0) to end channels (ADCS: ANE2 to ANE0).
  - When A/D conversion of the end channel ends, A/D conversion is suspended (ADCS: BUSY = 0).
  - To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS: BUSY) in the A/D control status register to "0".
  - When the A/D conversion mode selection bits (MD1, MD0) are set to "00B", this mode can be restarted during A/D conversion. If the bits are set to "01B", this mode cannot be restarted during A/D conversion.

■ Continuous Conversion Mode (ADCS: MD1, MD0 = "10B")
  - When the startup trigger is entered, A/D conversion of analog inputs is performed continuously from start channels (ADCS: ANS2 to ANS0) to end channels (ADCS: ANE2 to ANE0).
  - When A/D conversion for the end channel is terminated, it is continued after returning to the analog input for the start channel.
  - To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS: BUSY) in the A/D control status register to "0".
  - This mode cannot be restarted during A/D conversion.

■ Pause-conversion Mode (ADCS: MD1, MD0= "11B")
  - When the start trigger is input, A/D conversion starts for the start channel (ADCS: ANS2 to ANS0). A/D conversion will be suspended when A/D conversion for 1 channel ends. A/D conversion of the following channel will be carried out if an activation trigger is input while A/D conversion is suspended.
  - When A/D conversion of the end channel ends, A/D conversion is suspended. When the startup trigger is input during an A/D conversion pause, a return is made to the start channel analog input to resume A/D conversion.
  - To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS: BUSY) in the A/D control status register to "0".
  - This mode cannot be restarted during A/D conversion.
15.5.1 Single-shot Conversion Mode

In single conversion mode, A/D conversion of analog input is performed continuously from the start channels to the end channels. When A/D conversion of the end channel ends, A/D conversion is suspended.

Setting of Single-shot Conversion Mode

Operating the 8-/10-bit A/D converter in the single conversion mode requires the setting shown in Figure 15.5-1.

![Figure 15.5-1 Setting of Single-shot Conversion Mode](image)

- : Unused
- : Used bit
- : Set the bit corresponding to the pin using as analog input pin to "1”.
- : Set to "0”.

Operation of Single-shot Conversion Mode

- Entering the startup trigger starts A/D conversion from channel set using the A/D conversion start channel selection bit (ANS2 to ANS0) continuously until the channel set using A/D conversion end channel selection bit (ANE2 to ANE0).
- When A/D conversion of channel set using the A/D conversion end channel selection bit (ANE2 to ANE0) ends, A/D conversion stops (ADCS:BUSY=0).
- To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS:BUSY) in the A/D control status register to "0”.
- When the A/D conversion mode selection bits (MD1, MD0) are set to 00B, this mode can be restarted during A/D conversion. If the bits are set to 01B, this mode cannot be restarted during A/D conversion.

(When start and end channels are the same)

- When the start channel and end channel are set to the same channel number (ADCS:ANS2 to ANS0=ADCS:ANE2 to ANE0), A/D conversion is performed for only one channel set as the start channel after which processing stops (ADCS:BUSY=0).
Table 15.5-1 gives an example of the conversion order in the single-shot conversion mode.

<table>
<thead>
<tr>
<th>Start Channel</th>
<th>End Channel</th>
<th>Conversion Order in Single-shot Conversion Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN0 pin (ADCS: ANS=&quot;000B&quot;)</td>
<td>AN3 pin (ADCS: ANE=&quot;011B&quot;)</td>
<td>AN0 → AN1 → AN2 → AN3 → End</td>
</tr>
<tr>
<td>AN6 pin (ADCS: ANS=&quot;110B&quot;)</td>
<td>AN2 pin (ADCS: ANE=&quot;010B&quot;)</td>
<td>AN6 → AN7 → AN0 → AN1 → AN2 → End</td>
</tr>
<tr>
<td>AN3 pin (ADCS: ANS=&quot;011B&quot;)</td>
<td>AN3 pin (ADCS: ANE=&quot;011B&quot;)</td>
<td>AN3 → End</td>
</tr>
</tbody>
</table>
15.5.2 Continuous Conversion Mode

In continuous conversion mode, A/D conversion of analog input is performed continuously from the start channels to the end channels. When the end channel A/D conversion ends, A/D conversion continues from the analog input of the start channel.

### Setting of Continuous Conversion mode

Operating the 8-/10-bit A/D converter in the continuous conversion mode requires the setting shown in Figure 15.5-2.

**Figure 15.5-2 Setting of Continuous Conversion mode**

<table>
<thead>
<tr>
<th>bit15</th>
<th>bit14</th>
<th>bit13</th>
<th>bit12</th>
<th>bit11</th>
<th>bit10</th>
<th>bit9</th>
<th>bit8</th>
<th>bit7</th>
<th>bit6</th>
<th>bit5</th>
<th>bit4</th>
<th>bit3</th>
<th>bit2</th>
<th>bit1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCS</td>
<td>BUSY</td>
<td>INT</td>
<td>INTE</td>
<td>PAUS</td>
<td>STS1</td>
<td>STS0</td>
<td>STRT</td>
<td>Re-</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ADCR</td>
<td>SELB</td>
<td>ST1</td>
<td>ST0</td>
<td>CT1</td>
<td>CT0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADER</td>
<td>ADE7</td>
<td>ADE6</td>
<td>ADE5</td>
<td>ADE4</td>
<td>ADE3</td>
<td>ADE2</td>
<td>ADE1</td>
<td>ADE0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- : Unused

○ : Used bit

○ : Set the bit corresponding to the pin using as analog input pin to "1".

1 : Set to "1".

0 : Set to "0".

### Operation of Continuous Conversion Mode

- Entering the startup trigger starts A/D conversion from channel set using the A/D conversion start channel selection bit (ANS2 to ANS0) continuously until the channel set using A/D conversion end channel selection bit (ANE2 to ANE0).

- When A/D conversion of channels set using the A/D conversion end channel selection bit (ANE2 to ANE0) ends A/D conversion continues from channels set using the A/D conversion start channel selection bit (ANS2 to ANS0).

- To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS:BUSY) in the A/D control status register to "0".

- This mode cannot be restarted during A/D conversion.

*(When start and end channels are the same)*

- When the start channel and end channel are set to the same channel number (ADCS:ANS2 to ANS0=ADCS:ANE2 to ANE0), A/D conversion is repeatedly performed for the channel set as the start channel.
(Conversion order in continuous conversion mode)

Table 15.5-2 gives an example of the conversion order in the continuous conversion mode.

<table>
<thead>
<tr>
<th>Start Channel</th>
<th>End Channel</th>
<th>Conversion Order in Continuous Conversion Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN0 pin (ADCS: ANSI=&quot;000B&quot;)</td>
<td>AN3 pin (ADCS: ANE=&quot;011B&quot;)</td>
<td>AN0 → AN1 → AN2 → AN3 → AN0 → Repeat</td>
</tr>
<tr>
<td>AN6 pin (ADCS: ANSI=&quot;110B&quot;)</td>
<td>AN2 pin (ADCS: ANE=&quot;010B&quot;)</td>
<td>AN6 → AN7 → AN0 → AN1 → AN2 → AN6 → Repeat</td>
</tr>
<tr>
<td>AN3 pin (ADCS: ANSI=&quot;011B&quot;)</td>
<td>AN3 pin (ADCS: ANE=&quot;011B&quot;)</td>
<td>AN3 → AN3 → Repeat</td>
</tr>
</tbody>
</table>
15.5.3  

**Pause-conversion Mode**

In the conversion stop mode, A/D conversion starts and stops repeatedly for each channel. When the start trigger is entered after A/D conversion of the last channel stops, A/D conversion returns to the analog input of the start channel to continue A/D conversion with stops between each channel.

### Setting of Pause-conversion Mode

Operating the 8-/10-bit A/D converter in the pause-conversion mode requires the setting shown in Figure 15.5-3.

![Figure 15.5-3 Setting of Pause-conversion Mode](image)

### Operation of Pause-conversion Mode

- When the start trigger is entered, A/D conversion starts from channel set using the A/D conversion start channel selection bit (ANS2 to ANS0). A/D conversion will be suspended when A/D conversion for 1 channel ends. A/D conversion of the following channel will be carried out if an activation trigger is input while A/D conversion is suspended.
- When A/D conversion of channel set using the A/D conversion end channel selection bit (ANE2 to ANE0) ends, A/D conversion is paused. When the start trigger is entered while A/D conversion is paused, A/D conversion resumes from the channel set using the A/D conversion start channel selection bit (ANS2 to ANS0).
- A restart when A/D conversion is paused is performed by entering the start trigger set by the A/D start trigger selection bits (ADCS:STS1, STS0) in the A/D control status register.
- To terminate A/D conversion forcibly, set the A/D conversion operating flag bit (ADCS:BUSY) in the A/D control status register to "0".
- This mode cannot be restarted during A/D conversion.
(When start and end channels are the same)
When the start channel and end channel are set to the same channel number (ADCS:ANS2 to ANS0=ADCS:ANE2 to ANE0), A/D conversion and pausing is repeatedly performed for the channel set as the start channel.

(Conversion order in pause-conversion mode)
Table 15.5-3 gives an example of the conversion order in the pause-conversion mode.

<table>
<thead>
<tr>
<th>Start Channel (ADCS: ANS=“000B”)</th>
<th>End Channel (ADCS: ANE=“011B”)</th>
<th>Conversion Order in Single-shot Conversion Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN0 pin</td>
<td>AN3 pin</td>
<td>AN0 → Suspend/Start → AN1 → Suspend/Start → AN2 → Suspend/Start → AN3 → Suspend/Start → AN0 → Repeat</td>
</tr>
<tr>
<td>AN6 pin (ADCS: ANS=“110B”)</td>
<td>AN2 pin (ADCS: ANE=“010B”)</td>
<td>AN6 → Suspend/Start → AN7 → Suspend/Start → AN0 → Suspend/Start → AN1 → Suspend/Start → AN2 → Suspend/Start → AN6 → Repeat</td>
</tr>
<tr>
<td>AN3 pin (ADCS: ANS=“011B”)</td>
<td>AN3 pin (ADCS: ANE=“011B”)</td>
<td>AN3 → Suspend/Start → AN3 → Suspend/Start → Repeat</td>
</tr>
</tbody>
</table>
15.5.4 Conversion Using El²OS Function

The 8-/10-bit A/D converter can transfer the A/D conversion result to memory by using the El²OS function.

■ Conversion Using El²OS

When the El²OS function is used, the conversion data protection function transfers multiple data to memory without data loss even during continuous A/D conversion.

The flow of conversion when El²OS is used is shown in Figure 15.5-4.

* Determined by setting of El²OS.
15.5.5 A/D Conversion Data Protection Function

The data protection function is triggered when A/D conversion is performed while interrupts are enabled.

A/D conversion Data Protection Function in 8-/10-bit A/D Converter

The 8/10-bit A/D converter has only one A/D data register (ADCR) for storing A/D conversion data so the data in this register is rewritten when A/D conversion ends and the A/D conversion result is determined. If the conversion result cannot be loaded before the A/D data register is overwritten, the conversion result may be lost. 8/10-bit A/D converter has a data protection function that pauses A/D conversion when interrupts are enabled and an interrupt request is generated (ADCS:INT=1) to prevent data loss.

● A/D conversion data protection function when EI²OS is not used

  • When the A/D conversion results are stored in the A/D data register (ADCR) after the analog input is A/D-converted, the interrupt request flag bit in the A/D control status register (ADCS: INT) is set to "1".
  • When an interrupt request flag bit (ADCS:INT) is set, A/D conversion is paused to protect data.
  • When A/D control status register interrupts are enabled (ADCS:INTE=1), "1" is set in the INT bit and an interrupt request is generated in the CPU. When the INT bit is cleared after interrupt processing, the pause of A/D conversion is cancelled.

● A/D conversion data protection function when EI²OS is used

  • When the EI²OS function is used, A/D conversion is stopped for the data protection after A/D conversion while it is transferred the A/D conversion results from A/D data register to memory. When A/D conversion pauses, "1" is set in the pause flag bit of the A/D control status register (ADCS:PAUS).
  • When the EI²OS function is used, and A/D conversion results have been transferred to memory, A/D conversion pause status is canceled and the pause flag bit (ADCS:PAUS) is cleared to "0". A/D conversion is restarted when A/D conversion is performed continuously.

● Processing flow of A/D conversion data protection function when EI²OS is used

Figure 15.5-5 shows the processing flow of the A/D conversion data protection function when the EI²OS is used.
The A/D conversion data protection function operates only when interrupt is enabled. Set the interrupt enabling bit of the A/D control status register (ADCS:INTE) to "1".

Do not disable interruption while the A/D conversion results are transferred to the memory using the EI^2OS function. If interrupt is disabled (ADCS:INTE = 0) while A/D conversion is suspended, A/D conversion will be started, and data during transfer may be rewritten.

Do not restart while the A/D conversion results are being transferred to the memory using the EI^2OS function. Restarting while A/D conversion is stopped the conversion results are destroyed.
15.6 Precautions when Using 8-/10-bit A/D Converter

Precautions when using the 8-/10-bit A/D converter are given below:

● Precautions when Using 8-/10-bit A/D Converter

- Analog input pin
  - The analog input pin is simultaneously used for the general-purpose input/output port. When using the pin as an analog input pin, switch the pin to analog input pin according to the setting of the port direction register (DDR) and the analog input enable register (ADER).
  - When used as an analog input pin, set the port direction register bit for the corresponding pin to be used to "0" to turn off the output transistor. Set the analog input enable register (ADER) bit for the corresponding pin to be used to "1" to enable analog inputs.
  - When an intermediate-level signal is input with the pin set as a general-purpose I/O port, the input leakage current flows in the gate. When used as an analog input pin, analog input must be enabled.

- Precaution when starting by internal timer or external trigger
  The input value at which the 8-/10-bit A/D converter is started by the internal timer output or external trigger should be set to inactive (High for external trigger). When the start trigger input value is set to active, operation may start concurrently with the A/D control status register A/D start trigger selection bit (ADCS:STS1, STS0) setting.

- Procedure of 8-/10-bit A/D converter and analog input power on
  - Voltage to the 8/10-bit A/D converter power supply (AVcc, AVRH, AVRL) and analog inputs (AN0 to AN7) must be applied after application to the digital power supply (Vcc).
  - To turn off the power supply, shut down the 8/10-bit A/D converter power supply and analog inputs before shutting down the digital power supply (Vcc). Apply a voltage to AVRH that does not exceed AVcc or shut down. (The analog and digital power supplies can be apply and shut down simultaneously.)

- Power supply voltage of 8-/10-bit A/D converter
  For latch-up prevention, the 8/10-bit A/D converter power supply (AVcc) must not exceed the voltage of the digital power supply (Vcc).
15.7 Program Example 1 for A/D Converter
(Example of EI\(^2\)OS Start Program in Single Mode)

It shows the A/D conversion program started EI\(^2\)OS in single mode.

Example of EI\(^2\)OS start program in single mode

- Processing specification
  - Conversion is performed up to analog inputs AN1 to AN3.
  - The converted data is transferred to addresses 200\(_H\) to 205\(_H\).
  - Startup is performed by software.

EI\(^2\)OS startup program (single mode) flow is described.

![Figure 15.7-1 EI\(^2\)OS startup program (single mode) flow](image)

Coding example

```assembly
BAPL EQU 000100H ;Buffer address pointer lower
BAPM EQU 000101H ;Buffer address pointer middle
BAPH EQU 000102H ;Buffer address pointer upper
ISCS EQU 000103H ;EI\(^2\)OS status register
IOAL EQU 000104H ;I/O address register lower
IOAH EQU 000105H ;I/O address register upper
DCTL EQU 000106H ;Data counter lower
DCTH EQU 000107H ;Data counter upper
ADBR EQU 00001BH ;Analog input enable register
ICR00 EQU 0000B0H ;A/D interrupt control register
ADCS1 EQU 000036H ;A/D control status register
ADCS2 EQU 000037H
ADCR EQU 000038H ;A/D data Register

;---------Main program---------------------------
CODE CSEG
;
AND CCR,#0BFH ; Interruption is disabled
MOV I:ICR00,#08H ; Interrupt level 0 (highest) EI\(^2\)OS channel 0
```

Parallel process

Completed

Interrupt sequence

AN1 \(\rightarrow\) AN2 \(\rightarrow\) AN3 \(\rightarrow\) EI\(^2\)OS transmission

---
MOV BAPL,#00H ;Setting the converted data storage address
MOV BAPM,#02H ;(Uses 200H to 205H)
MOV BAPH,#00H
MOV ISCS,#00011000B ;word data transfers, after transfer address + 1,
I/O → transfer to memory
MOV IOAL,#38H ;transfer destination address pointer
MOV IOAH,#00H ;Sets analog data register address
MOV DCTL,#03H ;EI2OS send is set to 3, same as the number of conversion channels
MOV DCTH,#00H

; MOV ADER,#00001110B ;analog input setting
MOV I:ADCS1,#00001011B ;Single start, changing AN1 to AN3 channels
MOV I:ADCS2,#10100010B ;Software startup, A/D conversion start, interrupt enabled
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled

LOOP: User processing
BRA LOOP

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
WARI:
MOV I:ADCS2,#10000000B ;A/D conversion stop, interrupt flag clear/disable
User processing
RETI ;Returning from interrupt processing

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
VECT CSEG ABS=0FFH
ORG 00FFD0H ;vector set in interrupt number #11(0BH)
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode

VECT ENDS
END START
15.8 Program Example 2 for A/D Converter
(Example of EI²OS Start Program in Continuous Mode)

It shows the A/D conversion program started EI²OS in continuous mode.

Example of EI²OS start program in continuous mode

- Processing specification
  - Convert the analog inputs AN3 to AN5 twice to acquire two converted data for each of channels.
  - The converted data is transferred to addresses 600H to 60BH.
  - An external edge input is used to start operation.

EI²OS startup program (continuous mode) flow is described.

Figure 15.8-1 EI²OS startup program (continuous mode) flow

- Coding example

```
BAPL EQU 000100H ; Buffer address pointer lower
BAPM EQU 000101H ; Buffer address pointer middle
BAPH EQU 000102H ; Buffer address pointer upper
ISCS EQU 000103H ; EI²OS status register
IOAL EQU 000104H ; I/O address register lower
IOAH EQU 000105H ; I/O address register upper
DCTL EQU 000106H ; Data counter lower
DCTH EQU 000107H ; Data counter upper
ADER EQU 000108H ; Analog input enable register
ICROO EQU 0000B0H ; A/D interrupt control register
ADCS1 EQU 000036H ; A/D control status register
ADCS2 EQU 000037H
ADCR EQU 000038H ; A/D data register

;----------------Main program----------------------------------------
CODE CSEG

; stack pointer (SP), etc. are to be initialized;
AND CCR,#0BFH ; Interruption is disabled
MOV I:ICR00,#08H ; Interrupt level 0 (strength) EI²OS channel 0
```
MOV BAPL,#00H ;Setting the converted data storage address
MOV BAPM,#06H ;(Uses 600H to 60BH)
MOV BAPH,#00H
MOV ISCS,#00011000B ;word data transfers, after transfer address + 1,
    ;I/O→ transfer to memory
MOV IOAL,#38H ;transfer destination address pointer
MOV IOAH,#00H ;Setting analog data register address
MOV DCTL,#06H ;EI2OS transfer is set to 6 times
MOV DCTH,#00H

MOV ADER,#00111000B ;analog input setting
MOV I:ADCS1,#10011101B;Continuous startup, changing AN3 to AN5
    ;channels
MOV I:ADCS2,#10100100B;External edge start, interrupt enabled
MOV ILM,#07H ;Sets ILM in PS to level 7
OR CCR,#40H ;Interruption is enabled

LOOP:
    User processing
    •
    BRA LOOP

;----------------Interrupt Program------------------------------------------
WARI:
    MOV I:ADCS2,#10000000B;A/D conversion stop, interrupt flag clear/
    disable
    •
    User processing
RET ;Returning from interrupt processing

;----------------Vector Settings-----------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFD0H ;vector set in interrupt number #11(0B H)
DSL WARI
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Sets single-chip mode
VECT ENDS
END START
15.9 Program Example 3 for A/D Converter
(Example of EI²OS Start Program in Stop Mode)

It shows the A/D conversion program started EI²OS in stop mode.

Example of EI²OS start program in stop mode

- Processing specification
  - Analog input AN3 is converted 12 times in a set period.
  - The converted data is transferred to addresses 600H to 617H.
  - An external edge input is used to start operation.

EI²OS startup program (stop mode) flow is described.

**Figure 15.9-1 EI²OS startup program (stop mode) flow**

![Diagram of EI²OS startup program flow]

- **Coding example**

```assembly
BAPL EQU 000100H ;Buffer address pointer lower
BAPM EQU 000101H ;Buffer address pointer middle
BAPH EQU 000102H ;Buffer address pointer upper
ISCS EQU 000103H ;EI²OS status register
IOAL EQU 000104H ;I/O address register lower
IOAH EQU 000105H ;I/O address register upper
DCTL EQU 000106H ;Data counter lower
DCTH EQU 000107H ;Data counter upper
ADER EQU 00001BH ;Analog input enable register
ICROO EQU 0000B0H ;A/D interrupt control register
ADCS1 EQU 000036H ;A/D control status register
ADCS2 EQU 000037H
ADCR EQU 000038H ;A/D data register

;---------Main program-----------------------------------------------------
CODE CSEG
; ;stack pointer (SP), etc. are to be initialized
AND CCR,#0BFH ;Interruption is disabled
MOV I;ICROO,#08H ;interrupt level 0 (strength) EI²OS channel 0
MOV BAPL,#00H ;Setting the converted data storage address
MOV BAPM,#06H ;(Uses 600H to 617H)
```
MOV BAPH,#00H
MOV ISCS,#00011001B ;Transferring the word data, Transferred address + 1, ;I/O → transfer to memory
MOV IOAL,#38H ; transfer destination address pointer
MOV IOAH,#00H ; Sets analog data register address
MOV DCTL,#0CH ;EI^2OS transfer is performed 12 times for 3 channels only
MOV DCTH,#00H
MOV ADER,#00001000B ; analog input setting
MOV I:ADCS1,#11011101B ; Stop mode, changing AN3 channel
MOV I:ADCS2,#10100100B ; External edge start, interrupt enabled
MOV ILM,#0FH ; Sets ILM in PS to level 7
OR CCR,#40H ; Interruption is enabled

LOOP:

• User processing

• BRA LOOP

;--------- Interrupt Program---------------------------------------------

WARI:

MOV I:ADCS2,#10000000B ; A/D conversion stop, interrupt flag clear/ disable

• User processing

• RETI ; Returning from interrupt processing

;--------- Vector Settings------------------------------------------------------

VECT CSEG ABS=0FFH
ORG 00FFD0H ; vector set in interrupt number #11(0B_H)
DSL WARI
ORG 00FFDCH ; Reset vector setting
DSL START
DB 00H ; Sets single-chip mode

VECT ENDS

END START
CHAPTER 16

D/A CONVERTER

This chapter describes the function and operation of the D/A converter.

16.1 Overview of D/A Converter
16.2 Block Diagram of the D/A Converter
16.3 Configuration of D/A Converter
16.4 Explanation of Operations of D/A Converter Functions
16.5 Notes on Using the D/A Converter
16.6 Program Example for D/A Converter
16.1 Overview of D/A Converter

D/A converter carries out D/A conversion at 8-bit resolution based on the R-2R method. The D/A converter has 2 built-in channels, with independent control possible per channel.

■ Function of D/A converter

D/A converter is made up of the D/A control register, D/A data register, and D/A conversion circuit. The value set up for the data register is converted to an analog voltage, and output to the D/A output pin. The theoretical value of the D/A converter output voltage is shown in Table 16.1-1.

Table 16.1-1 Theoretical values of output voltage of the D/A converter

<table>
<thead>
<tr>
<th>D/A data register setting value</th>
<th>Theoretical value of output voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>0/256×DVR (=0V)</td>
</tr>
<tr>
<td>01H</td>
<td>1/256×DVR</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>FEH</td>
<td>254/256×DVR</td>
</tr>
<tr>
<td>FFH</td>
<td>255/256×DVR</td>
</tr>
</tbody>
</table>

DVR: Reference voltage of D/A converter
16.2 Block Diagram of the D/A Converter

D/A converter is made up of the following blocks, and 2 channels are built in.
- D/A control register (DACR)
- D/A Data Register (DADR)
- D/A conversion circuit

Block Diagram of the D/A Converter

Figure 16.2-1 Block Diagram of the D/A Converter
D/A control register

Enabling or disabling output of the analog voltage generated through D/A conversion can be set.

D/A Data Register

Output value for D/A conversion is set. D/A conversion is carried out on the value that has been set to the D/A data register, and output to the D/A pin.

D/A conversion circuit

The analog voltage that supports the value set to the D/A data register is generated.

Details of pin

2 channels are built into the D/A converter. The pins to be used for each channel are shown as follows.

Channel 0

DA0 pin: P53/DA0

Channel 1

DA1 pin: P54/DA1
16.3 Configuration of D/A Converter

The pin and register details of the D/A converter are described.

- **Pin of D/A Converter**
  The pin to be used for the D/A converter is also used for the general-purpose input/output port. The pin functions and settings when used for the D/A converter are shown in Table 16.3-1.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting required using for D/A converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 0</td>
<td>DA0</td>
<td>General-purpose I/O port, D/A output</td>
<td>D/A output is enabled. (DACR0: DAE0=1)</td>
</tr>
<tr>
<td>Channel 1</td>
<td>DA1</td>
<td>D/A output</td>
<td>D/A output is enabled. (DACR1: DAE1=1)</td>
</tr>
</tbody>
</table>

- **Reference voltage pin of D/A converter**
  - DVcc, DVss

  The DVcc pin and DVss pin are the standard power pins for the D/A converter. The voltage between DVcc and DVss will be the standard voltage (DVR) for the D/A converter.
D/A control register (DACR) can set enabling or disabling output of analog voltage generated through D/A conversion of the D/A converter.

Table 16.3-2  D/A control register (DACR) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>DAE: D/A Output enable bit</td>
</tr>
<tr>
<td></td>
<td>Output of D/A output pin is set to enable or disable.</td>
</tr>
<tr>
<td></td>
<td>When the bit is set to &quot;0&quot;: This will be D/A output pin after D/A output is enabled.</td>
</tr>
<tr>
<td></td>
<td>When the bit is set to &quot;1&quot;: D/A output is disabled, and will be a general-purpose input/output port.</td>
</tr>
<tr>
<td>bit1 to</td>
<td>Unused bits</td>
</tr>
<tr>
<td>bit7</td>
<td>Read: The value is undefined.</td>
</tr>
<tr>
<td></td>
<td>Write: No effect</td>
</tr>
</tbody>
</table>
16.3.2 D/A Data Register (DADR)

This register sets up output value for D/A conversion.

- D/A Data Register (DADR)

Analog voltage to be output to the D/A pin is set.

D/A conversion is carried out on the value that has been set to the D/A data register, and output to the D/A pin.
16.4 Explanation of Operations of D/A Converter Functions

Analog output is started using the D/A converter by setting a value to the D/A data register, and setting the D/A output enabling bit of the D/A control register to "1".

■ Explanation of Operations of D/A Converter Functions

The setting as per Figure 16.4-1 is required to use the D/A converter.

![Figure 16.4-1 Setting of D/A converter](image)

When a value is set to the D/A data register and D/A output is enabled (DACR:DAE = 1), output by the D/A converter will be started.

(When D/A output is disabled)

When the D/A converter output is disabled (DACR:DAE = 0), the status will be as follows.

- The analog switch that is serially inserted to the output area of the D/A converted is turned off.
- Clear is carried out on the "L" level output within the D/A converter.
- The flow route for direct current electricity is cut off.

(Output voltage)

- Output voltage range of the D/A converter is 0 V to 255/256 × DVR.
- Output voltage range can be changed by controlling the DVR voltage externally.

The theoretical value of the D/A converter output voltage is shown in Table 16.4-1.

<table>
<thead>
<tr>
<th>Table 16.4-1 Theoretical values of output voltage of the D/A converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>D/A data register setting value</td>
</tr>
<tr>
<td>00H</td>
</tr>
<tr>
<td>01H</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>FEH</td>
</tr>
<tr>
<td>FFH</td>
</tr>
</tbody>
</table>

DVR: reference voltage of D/A converter
16.5 Notes on Using the D/A Converter

Care must be taken regarding the following points when the D/A converter is used.

- **Notes on Using D/A Converter**
  
  - **Settling time when an external load is connected**
    
    - Buffer amplifier is not built-in for the D/A converter output. As an analog switch (\(\approx 100\Omega\)) is built-in series in the output area, carefully consider the required settling time for external output loads.
  
  - **Power supply voltage of D/A converter**
    
    - Reference voltage (DVcc) of the D/A converter should not exceed Vcc.
    - The grounded power supply voltage (DVss) for the D/A converter should have the same potential as Vss.
16.6 Program Example for D/A Converter

An example of the D/A converter program is shown.

Program Example for D/A Converter

**Processing specification**

- Channel 0 is used and the reference voltage will be 5 V (DVR = 5 V).
- Voltage of approx. 3 V is generated from the DA0 output pin. The D/A data register value will be "9AH" as shown below.
- DADR value = (Output voltage × 256)/DVR = (3V×256)/5V =153.6 ≈ 9AH

**Coding example**

```
DACR0 EQU 00003CH ;D/A control register
DADR0 EQU 00003AH ;D/A Data Register
DAE0 EQU DACR0:0 ;D/A Output enable bit ;(Channel 0) definition

;----------Main Program---------------------------------------------
CODE CSEG
START
    MOV I:DADR0,#09AH ;D/A data register is set.
                     ;(Channel 0 is set to approx. 3 V.)
    SETB I:DAE0      ;D/A output is enabled.
;
CODE ENDS

;----------Vector Settings------------------------------------------
VECT CSEG ABS=0FFH
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START
```
Functions and operations of the communication prescaler are explained.

17.1 Overview of Communication Prescaler
17.2 Communication Prescaler Register (CDCR)
17.1 Overview of Communication Prescaler

- Clock is provided to the dedicated baud rate generator for the UART and extended serial I/O functions.
- The machine clock is divided to provide the dedicated baud rate generator.

### Block Diagram

![Figure 17.1-1 Block Diagram](image)

- **Communication prescaler register (CDCR)**
  
  Set up the machine clock division rate to acquire a fixed communication rate. Set up enable or disable operation for the communication prescaler as well.

- **Prescaler**
  
  Machine clock \( \phi \) is divided, and clock is supplied to the UART and I/O extended serial. Division rate can be selected from 7 types.
17.2 Communication Prescaler Register (CDCR)

The communication prescaler register enables or disables operation of the communication prescaler, and sets the machine clock division ratio.

Communication prescaler register

![Figure 17.2-1 Communication prescaler register (CDCR)]

**Notes:**
- When the division rate setting is changed, communications should continue only after 
  
  \( \frac{\text{div}}{\Phi} \times 2 \)
  
  cycles of the clock stabilization time have expired.
- A communication prescaler is used for both the UART and I/O extended serial interface. When the communication prescaler register settings are changed, care must be taken as the UART and I/O extended serial interface operations may be affected.
- When the exclusive baud rate generator is used at the synchronous transmission, the following settings are the prohibitions:
  1. \( \text{CS2 - CS0 = 000}_B \)
  2. \( \text{CS2 - CS0 = 001}_B \) and \( \text{DIV3 - DIV0 = 0000}_B \)
Settable range for the division ratio

Set up so that the machine clock division results will not exceed 4.25 MHz.
Set the division ratio as per Table 17.2-2 in accordance with the machine clock $\phi$ to be used.

Table 17.2-2 Division ratio set up using the communication prescaler

<table>
<thead>
<tr>
<th>Machine clock $\phi$ (MHz)</th>
<th>Bits setting the division ratio</th>
<th>Divide ratio div</th>
<th>Division result $\phi$/div (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1 1 0 0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>1 0 1 0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>1 0 0 0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>1 1 0 1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1 1 0 0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>1 0 1 1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>1 0 1 1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>1 0 0 1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>16</td>
<td>1 0 0 0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>1 1 1 0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>1 1 0 1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>1 1 0 0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Set up so that the division results of the machine clock will not exceed the maximum 4.25 MHz.

Reference: Refer to "CHAPTER 18 UART" and "CHAPTER 19 I/O EXTENDED SERIAL INTERFACE" for details.
CHAPTER 18
UART

This chapter describes UART function operation.

18.1 Overview of UART
18.2 UART Block Diagram
18.3 Configuration of UART
18.4 LIN-UART Interrupt
18.5 UART Baud Rate
18.6 Explanation of Operation of UART
18.7 Notes on Using UART
18.8 Example of UART Programming
CHAPTER 18 UART

18.1 Overview of UART

UART is a general-purpose serial data communication interface for synchronous and asynchronous communications with external devices.
- Incorporates a bidirectional communication function (clock synchronous and asynchronous modes)
- The master/slave communication function (multiprocessor mode) is incorporated.
- Interrupt requests can be output when reception ends, a reception error is detected or sending ends. Ei^2OS starts whichever interrupt is generated.

UART function

The functions of UART, a general-purpose serial data communications interface for sending and receiving general-purpose serial data, are listed in Table 18.1-1.

Table 18.1-1 UART function

| Data buffer | Full-duplicate double-buffer |
| Transfer mode | 
| • Synchronous to clock (without start bit/stop bit and parity bit)  
| • Clock Asynchronous (start-stop synchronization) |
| Baud rate | 
| • Dedicated baud rate generator  
| • The external clock signal can be input.  
| • Enable to use the clock supplied from 16-bit reload timer 0 |
| Data length | 
| • 7 bits (for asynchronous normal mode only)  
| • 8 bits |
| Signal type | NRZ (Non Return to Zero) type |
| Detection of receive error | 
| • Framing error (Clock synchronization mode is disabled.)  
| • Overrun error  
| • Parity error (Clock synchronization mode and multi processor mode are disabled.) |
| Interrupt request | 
| • Receive interrupt (receive, detection of receive error)  
| • Transmit interrupt (transmit)  
| • Both the transmission and reception support Ei^2OS. |
| Master/slave type communication function (multi processor mode) | This function enables communications between 1 (only use master) and n (slave)  
(This function is used only as the master side) |
### Table 18.1-2 UART operation modes

<table>
<thead>
<tr>
<th>Operating mode</th>
<th>Data length</th>
<th>Parity bit</th>
<th>Stop Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7 bit</td>
<td>8 bits</td>
<td>None</td>
</tr>
<tr>
<td>0 Asynchronous</td>
<td>Normal mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1 vs. 1)</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>1 Asynchronous</td>
<td>Multiprocessor mode</td>
<td></td>
<td>(1 vs. n)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>○ (+1)</td>
<td>○</td>
</tr>
<tr>
<td>2 Clock synchronous</td>
<td>Clock synchronous mode</td>
<td></td>
<td>(1 vs. 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

○: Can be set  
-: Setting disabled  
+ 1: This is the address/data bit (A/D) to be used for communication control.

**Notes:**  
- Set data length to 8 bits in the multiprocessor mode and clock synchronization mode.  
- A parity bit cannot be added in multiprocessor mode and clock synchronization mode.  
- In the clock synchronization mode, only data is transferred and no start bit or stop bit is added.
18.2 UART Block Diagram

The block diagram is shown below.

**UART block diagram**

![UART block diagram](image-url)
Details of Pins and Interrupt Numbers

- SIN pin: P42/SIN0
- SOT pin: P43/SOT0
- SCK pin: P44/SCK0
- Transmit interrupt number 1: #39 (27H)
- Receive interrupt number 1: #37 (25H)

Clock selector

The dedicated baud rate generator, external input clock or internal timer output clock (clock provided by 16-bit reload timer 0) are selected as synchronous clocks for sending and receiving.

Reception Control Circuit

The receive controller is composed of receive bit counter, start bit detector and receive parity counter.
The reception bit counter counts reception data and generates a reception interrupt request when one data frame has been received.
The start bit detection circuit detects start bits from the serial input signal.
The received parity counter calculates the parity of the received data.

Transmission Control Circuit

The transmit controller is composed of the transmit bit counter, transmit start circuit, and transmit parity counter.
The send bit counter counts send data and generates a send interrupt request when one data frame has been sent.
The send start circuit starts sending when send data is placed in the serial output data register.
The send parity counter generates the parity bit when the serial control register is set to "Parity".

Reception shift register

Received data input from outside the SIN0 pin is fetched bit by bit while shifting, the received data is transferred to the serial input data register when reception ends.

Transmit shift register

Send data set in the serial output data register is transferred to the send shift register, and then it is shifted by each bit and output from the SOT0 pin.

Serial mode register (SMR)

Operating mode and clock input source (baud rate) can be selected, and serial data pin output enable/disable and clock pin output enable/disable can be set.
● Serial control register (SCR)
  Parity settings, parity type selection, stop bit length setting, data length setting, frame data format selection in operating mode 1, error flag clear, send enable/disable setting, reception enable/disable setting can be set.

● Serial status register (SSR)
  Send/reception status and error status can be checked and send/reception interrupt requests can be enabled or disabled.

● Serial input data register (SIDR)
  The register retains the receive data. Reception data transferred from the reception shift register is stored.

● Serial output data register (SODR)
  The register sets the transmit data. Send data set in the serial output data register is transferred to the send shift register after which each bit is shifted and output.

● Communication prescaler register (CDCR)
  Communication pre-scaler startup/stop and machine clock division ratio can be selected. The divided clock is supplied to a baud rate generator.
18.3 Configuration of UART

Detailed description of UART pins, interrupt factors and registers are provided.

UART pins

Pins used by UART are combined with general-purpose input/output ports.

Table 18.3-1 shows pin functions and required settings for using UART functions.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting Necessary for Use in UART</th>
</tr>
</thead>
<tbody>
<tr>
<td>P42/SIN0</td>
<td>General purpose I/O port/serial data input</td>
<td>Setting to input port in port direction register (DDR)</td>
</tr>
<tr>
<td>P43/SOT0</td>
<td>General purpose I/O port/serial data output</td>
<td>Set to output enable. (SMR: SOE=1)</td>
</tr>
<tr>
<td>P44/SCK0</td>
<td>General purpose I/O port/serial clock output input</td>
<td>Set pin as input port in port direction register (DDR).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In clock output, set to output enable. (SMR: SCKE=1)</td>
</tr>
</tbody>
</table>

Block Diagram of UART Pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

Interrupt Request Generation by UART

● Reception Interrupt

A reception interrupt request is generated when one of the following is detected and reception interrupts are enabled (SSR:RIE=1).

• Data receive completed (SSR: RDRF=1)
• Overrun error detected (SSR:0 RE=1) (SSR: ORE=1)
• Frame error detected (SSR:FRE=1) (SSR: FRE=1)
• Parity error detected (SSR:PE=1) (SSR: PE=1)

● Transmission Interrupt

It disables the send interrupt (SSR:TIE=1) and sets the send data empty flag to "1" (SSR:TDRE=1) to generate a send request when send data has been transferred from the serial output data register to the send shift register.
18.3.1 Serial Control Register (SCR)

The serial control register (SCR) sets the parity bit, selects stop bit length and data length, selects frame data format in operating mode 1, clears reception error flags and enables and disables sending and reception.

Serial control register (SCR)

Figure 18.3-1 Serial control register (SCR)
## Table 18.3-2 Function of Serial Control Register (SCR)

<table>
<thead>
<tr>
<th>bit name</th>
<th>UART Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 TXE:</td>
<td>The bit enables or disables the UART for transmission. <strong>Note:</strong> When the transmitting in process is disabled, the device stops it after transmitting the current data from the serial output data register. When transmission/reception is disabled, set up after pausing 1/16 of the baud rate time for the clock asynchronous transfer mode, or the same length of time as the baud rate for the clock synchronous transfer mode, after writing the data to the serial output data register.</td>
</tr>
<tr>
<td>bit9 RXE:</td>
<td>The bit enables or disables the UART for reception. <strong>Note:</strong> When disabled reception is set during reception, reception will be suspended after the data being received is stored to serial input data register. Reception counter will not be cleared.</td>
</tr>
<tr>
<td>bit10 REC:</td>
<td>Clear the reception error flags (SSR0: FRE, ORE, PE) in the serial status register to 0. <strong>Note:</strong> While reception interruption is enabled (SSR:RIE = 1), set this REC bit to &quot;0&quot; only when the FRE, ORE, or PE flag is set to &quot;1&quot;.</td>
</tr>
<tr>
<td>bit11 A/D:</td>
<td>Frame data format to transmit/receive under operation mode 1 is set. <strong>Note:</strong> 7-bit can be selected under operating mode 0 only. 8-bit length must be set for operation modes 1 and 2.</td>
</tr>
<tr>
<td>bit12 CL:</td>
<td>Specify the length of send and receive data. <strong>Note:</strong> Only the 1st bit of the stop bit is always detected during reception.</td>
</tr>
<tr>
<td>bit13 SBL:</td>
<td>The bit lengths for the stop bit (frame end mark of the transmission data) for operation modes 0 and 1 are set. <strong>Note:</strong> In operation modes 1 and 2, no parity bit can be added. Always set this bit to &quot;1&quot;.</td>
</tr>
<tr>
<td>bit14 P:</td>
<td>When Yes is set to the parity bit (PEN = 1), whether it has odd or even parity will be set.</td>
</tr>
<tr>
<td>bit15 PEN:</td>
<td>Whether or not addition (for transmission) and detection (for reception) of the parity bit is executed. <strong>Note:</strong> In operation modes 1 and 2, no parity bit can be added. Always set this bit to &quot;1&quot;.</td>
</tr>
</tbody>
</table>
18.3.2 Serial Mode Register (SMR)

The serial mode register (SMR) selects operating mode, baud rate clock and enables/disables output to the serial data and clock pin.

### Serial mode register (SMR)

**Figure 18.3-2 Serial mode register (SMR)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Operation mode select bits</td>
<td>R/W</td>
</tr>
<tr>
<td>1</td>
<td>Baud rate by dedicated baud rate generator</td>
<td>&quot;000a&quot; to &quot;100a&quot;</td>
</tr>
<tr>
<td>0</td>
<td>Baud rate by Internal timer</td>
<td>&quot;101a&quot;</td>
</tr>
<tr>
<td></td>
<td>(16-bit reload timer 0)</td>
<td>&quot;110a&quot;</td>
</tr>
<tr>
<td></td>
<td>Baud rate by External clock</td>
<td>&quot;111a&quot;</td>
</tr>
<tr>
<td></td>
<td>Clock input source select bits</td>
<td>CS2 CS1 CS0</td>
</tr>
<tr>
<td></td>
<td>Mode No.</td>
<td>Operation mode</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Asynchronous normal mode</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Asynchronous multiprocessor mode</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Clock synchronous mode</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Setting disabled</td>
</tr>
</tbody>
</table>

**Reset value**: 00000000B

### Table 18.3-2 Serial mode register (SMR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Serial - data output enable bit (SOT0 pin)</td>
<td>0 Set as a general-purpose I/O port</td>
</tr>
<tr>
<td>1</td>
<td>Serial clock I/O enable bit (SCK0 pin)</td>
<td>1 Set as a serial clock output pin</td>
</tr>
<tr>
<td>2</td>
<td>Always set this bit to &quot;0&quot;.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Always set this bit to &quot;0&quot;.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Always set this bit to &quot;0&quot;.</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Always set this bit to &quot;0&quot;.</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Always set this bit to &quot;0&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and Writable

Reset value: 00000000B
<table>
<thead>
<tr>
<th>bit name</th>
<th>UART Functions</th>
</tr>
</thead>
</table>
| bit0            | **SOE:** Serial-data output enable bit  
  **Serial data output to the SOT0 pin is enabled or disabled.**  
  **When the bit is set to "0":** The pin is set as a general-purpose I/O port.  
  **When the bit is set to "1":** The pin is set as a serial data output pin. (SOT0)  
  **Reference:**  
  When the serial data output pin is set (SOE = 1), this functions as the SOT0 pin regardless of the setting of the general-purpose input/output port (DDR).                                                                                                                                 |
| bit1            | **SCKE:** Serial clock I/O enable bit  
  **Control the serial clock output/input of SCK0 pin.**  
  **When the bit is set to "0":** The pin is set as a general-purpose I/O port or serial clock input pin.  
  **When the bit is set to "1":** The pin is set as a serial clock output pin.  
  **Note:**  
  1) When the SCK0 pin is used as the serial clock input pin (SCKE = 0), set the pin to the input port using the port direction register (DDR). Set the clock selection bits to the external clock (SMR: CS2 to CS0 = "111B").  
  2) Set the clock selection bit to the internal timer or dedicated baud rate generator when it is used as the serial clock output pin (SCKE = 1). (SMR: CS2 to CS0 = other than "111B") (SMR: other than CS2 to CS0 = "111B")  
  **Reference:**  
  When the SCK0 pin is set for the serial clock output pin, it functions as the serial clock output pin regardless of the setting of the general-purpose input/output port.                                                                                   |
| bit2            | **Reserved:** reserved bit  
  Always set this bit to "0".                                                                                                                                                                                                                                                                                                                                       |
| bit5 to bit3    | **CS2 to CS0:** Clock input source selection bits  
  **Set the clock input source for the baud rate.**  
  • The clock input source is selected from the external clock (SCK0 pin), 16-bit reload timer 0 output, or dedicated baud rate generator.  
  • Set the baud rate when selecting the dedicated baud rate generator.  
  (Refer to "18.5.1 Baud Rate by Dedicated Baud Rate Generator".)  
  **Note:**  
  When the exclusive baud rate generator is used at the synchronous transmission, the following setting are the prohibitions.  
  1) CS2 - CS0 = 000B  
  2) CS2 - CS0 = 001B and DIV3 - DIV0 = 0000B                                                                                                                                                                                                                                                                 |
| bit7 to bit6    | **MD1, MD0:** Operation mode selection bits  
  **Select the UART operation mode.**  
  **Note:**  
  Operation mode 1 of the master slave type communication can only be used as a master. UART cannot be used as a slave, as the 9th address/data bit cannot receive.  
  **Reference:**  
  ts the external clock (SCK0 pin), 16-bit reload timer 0 output, or dedicated baud rate generator.  
  • Set the baud rate when selecting the dedicated baud rate generator.  
  (Refer to "18.5.1 Baud Rate by Dedicated Baud Rate Generator".)  
  **Note:**  
  When the exclusive baud rate generator is used at the synchronous transmission, the following setting are the prohibitions.  
  1) CS2 - CS0 = 000B  
  2) CS2 - CS0 = 001B and DIV3 - DIV0 = 0000B
18.3.3 Serial Status Register (SSR)

The serial status register (SSR) checks send/reception status and error status and enables/disables send/reception interrupt requests.

■ Serial status register (SSR)

![Figure 18.3-3 Serial status register (SSR)](image-url)

- **TIE**
  - Transmission interrupt enable bit
  - 0: Disable send interrupt.
  - 1: Enable send interrupt.

- **RIE**
  - Interrupt enable bit
  - 0: Disable send interrupt.
  - 1: Enable send interrupt.

- **TDRE**
  - Transmission data empty flag
  - 0: With Transmission data (Writing Transmission data disabled)
  - 1: No Transmission data (Writing Transmission data enabled)

- **RDRF**
  - Receive data load flag bit
  - 0: No Receive data
  - 1: With Receive data

- **FRE**
  - Framing error flag
  - 0: No Framing error
  - 1: With Framing error

- **ORE**
  - Overrun error flag
  - 0: No Overrun error
  - 1: With Overrun error

- **PE**
  - Parity error flag bit
  - 0: No Parity error
  - 1: With Parity error
### Table 18.3-4 Function of Serial Status Register (SSR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>UART Functions</th>
</tr>
</thead>
</table>
| bit8 TIE:         | **Transmission interrupt enable bit**  
|                   | Enable or disable send interrupt.  
|                   | • When transmission interruption is enabled (SSR:TIE = 1), and data written in the serial output data register is loaded to the shift register for transmission (SSR:TDRE = 1), a transmission interrupt request is generated.  
| bit9 RIE:         | **Reception interrupt enable bit**  
|                   | Enable or disable receive data.  
|                   | • Reception interrupt request is generated when reception data is loaded to the serial input data register (SSR:RDRF = 1) or a reception error is generated (one or more error flags for PE, ORE, or FRE are "1"), while reception interruption is enabled (SSR:RIE = 1).  
| bit10 Unused bits |  
|                   | **Read:** The value is undefined.  
|                   | **Write:** No effect                                                                                                                                   |
| bit11 TDRE:       | **Transmission data empty flag**  
|                   | This flag indicates the status of the serial output data register.  
|                   | • This will be cleared to "0" when transmission data is set in the serial output data register.  
|                   | • "1" will be set when the data of the serial output data register is loaded to the shift register for transmission, and transmission starts.  
|                   | • Transmission interrupt request is generated when data that has been set to the serial output data register is sent while transmission interrupt is enabled (SSR:TIE = 1).  
|                   | **Note:**  
|                   | This TDRE bit will be set to "1" (no transmission data) if reset.                                                                                     |
| bit12 RDRF:       | **Receive data load flag**  
|                   | This flag indicates the status of the serial input data register.  
|                   | • "1" will be set when reception data is loaded to the serial input data register.  
|                   | • This will be cleared to "0" when data of the serial input data register is read.  
|                   | • Reception interrupt request is generated when reception data is loaded to the serial input data register while reception interruption is enabled (SSR:RIE = 1).  
| bit13 FRE:        | **Framing error flag**  
|                   | This flag indicates whether or not a framing error exists during reception.  
|                   | • This will be cleared to "0" when the reception error flag clear bit is set to "0" (SCR:REC = 0).  
|                   | • When a framing error is generated while reception interrupts are enabled (SSR:RIE = 1), a reception interrupt request is generated.  
|                   | • Data in the serial input data register will be invalid when this FRE bit is set.  
| bit14 ORE:        | **Overrun error flag**  
|                   | This flag indicates whether or not an overrun error exists during reception.  
|                   | • This will be cleared to "0" when the reception error flag clear bit is set to "0" (SCR:REC = 0).  
|                   | • Reception interrupt request is generated when an overrun error is generated while reception interrupt is enabled (SSR:RIE = 1).  
|                   | • Data in the serial input data register will be invalid when this ORE bit is set.  
| bit15 PE:         | **Parity error flag**  
|                   | This flag indicates whether or not a parity error exists during reception.  
|                   | • This will be cleared to "0" when the reception error flag clear bit is set to "0" (SCR:REC = 0).  
|                   | • Reception interrupt request is generated when a parity error is generated while reception interrupt is enabled (SSR:RIE = 1).  
|                   | • Data in the serial input data register will be invalid when this PE bit is set.  

18.3.4 Serial Input Data Register (SIDR) and Serial Output Data Register (SODR)

Serial input data register (SIDR) is a serial data reception register and the serial output data register (SODR) is a serial data send register.
- SIDR register and SODR register are stored at the same address.
- It functions as SIDR during reading of data.
- It functions as SODR during writing of data.

■ Serial input data register (SIDR)

The bit configuration of the serial input data register is shown in Figure 18.3-4.

SIDR1 is a data buffer register for receiving serial data.
- Serial data signals input to the SIN0 pin are converted and stored in the serial input data register.
- Data becomes invalid when the data length of the high-order 1-bit (D7) of the serial input data register is 7 bits.
- The reception data load flag is set to "1" (SSR:RDRF=1) when reception data is stored in the serial input data register. When a reception interrupt request is enabled (SSR:RIE=1), a reception interrupt request is generated.
- Read the reception data of the serial input data register when the reception data load flag is set to "1" (SSR:RDRF=1). When data in the serial input data register is read, the reception data load flag is cleared to "0".
- When a reception error occurs (when one or more for SSR:PE, ORE, FRE error flag is "1"), the data in the serial input data register becomes invalid.
Serial output data register (SODR)

The serial output data register is shown in Figure 18.3-5.

### Figure 18.3-5 Serial output data register (SODR)

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>XXXXXXXXXb</td>
</tr>
</tbody>
</table>

W : Write only  
X : Undefined

The serial output data register 1 (SODR1) is a data buffer register for transmitting serial data.

- When reception data is placed in the serial output data register and sending is enabled, send data is transferred to the transfer shift register, shifted 1 bit at a time and output from the serial data output pin (SOT0 pin).
- Data becomes invalid when the data length of the high-order 1-bit (D7) of the serial output data register is 7 bits.
- When send data is placed in the serial output data register, the send data empty flag is cleared to "0" (SSR:TDRE=0). When the transfer from the serial output data register to the send shift register ends, the send data empty flag is set to "1" (SSR:TDRE=1) and send data can be set.
- When the send interrupt is enabled (SSR:TIE=1) and send data written to the serial output data register is loaded into the send shift register (SSR:TDRE=1), a send interrupt request is generated.

**Note:** Serial output data registers are write-only, while serial input data registers read-only. Since it is located at the same address, the write value and read value differ. The command that operates read-modify-write (RMW) cannot be used.
18.4 LIN-UART Interrupt

UART has reception and send interrupts and interrupt requests are generated by the following factors.
- When reception data is loaded into the serial input data register or a reception error occurs.
- When data to transmit is transferred from serial output data register to the transmission shift register.

All interrupts support functions of extended intelligent I/O service (EIO2OS).

### UART interrupt

The UART interrupt control bits and interrupt factors are shown in Figure 18.4-1.

**Table 18.4-1 UART Interrupt Control Bit and Interrupt Factor**

<table>
<thead>
<tr>
<th>Trans-mission/Reception</th>
<th>Interruption demand flag</th>
<th>Operating mode</th>
<th>Interrupt Factor</th>
<th>Output enable bits</th>
<th>Clear of the Interrupt-request Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reception</td>
<td>RDRF</td>
<td>0 1 2</td>
<td>Receive data loaded into serial input data register (SIDR)</td>
<td></td>
<td>Reading receive data</td>
</tr>
<tr>
<td></td>
<td>O           O       O</td>
<td></td>
<td>Generating Overrun error</td>
<td>SSR: RIE</td>
<td>Writing 0 to the reception error flag clear bit (SSR: REC)</td>
</tr>
<tr>
<td></td>
<td>O           O       X</td>
<td></td>
<td>Generating Framing error</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>O           X       X</td>
<td></td>
<td>Generating parity error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transmission</td>
<td>TDRE</td>
<td>0 1 2</td>
<td>Serial output data register (SODR) is empty.</td>
<td>SSR: TIE</td>
<td>Writing transmit data</td>
</tr>
</tbody>
</table>

- **Reception Interrupt**

  An interrupt request is generated when one of the following is detected and reception interrupts are enabled (SSR: RIE=1).
  - Data receive completed (SSR: RDRF=1)
  - Overrun error generated (SSR: ORE=1)
  - Frame error generated (SSR: FRE=1)
  - Generating parity error (SSR: PE=1)

- **Transmission Interrupt**

  When send data is transferred from the serial output data register to the send shift register, the send data empty flag is set to "1" (SSR: TDRE=1).

  When send interrupts are enabled (SSR: TIE=1), the send data empty flag is set to "1" (SSR: TDRE) and a send interrupt request is generated.
Interrupt Related to UART and EI²OS

Reference: Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

EI²OS Function of UART

UART supports the EI²OS function and starts EI²OS when reception/send interrupt factors apply.

The EI²OS is available only when other resources sharing the interrupt control register (ICR) do not use interrupts.

When UART send interrupts (or receive interrupts) are to be used in EI²OS, disable interrupts to peripheral devices that share send interrupts (or receive interrupts) and interrupt control registers.

When a reception error occurs, EI²OS is automatically terminated.
18.4.1 Timing of Reception Interrupt Request Generation and Flag Setting

Interrupts that occur during reception are of two kinds: reception completed (SSR:RDRF) and reception error generation (SSR:PE, ORE, FRE).

- **Timing of reception interrupt request generation and flag setting**
  - Receive data load flag and each receive error flag sets
    - Reception data is stored in the serial input data register (SIDR) when the stop bit is detected in operating mode 0 and 1 or when the last bit (D7) is detected in operating mode 2.
    - When a reception error occurs during storage of reception data, the error flag (SSR:PE, ORE, FRE) for the error that occurs is set to "1" after which the reception data load flag is set to "1".
    - If one of the error flags is set to "1" in any operating mode, the reception data in the serial input data register becomes invalid.

**Operation mode 0 (Asynchronous normal mode)**
- When the stop bit is detected, the reception data load flag is set to "1". (SSR: RDRF=1)
- When the reception error is generated, the error flag is set to "1". (SSR: PE, ORE, FRE)

**Operating mode 1 (asynchronous multiprocessor mode)**
- When the stop bit is detected, the reception data load flag is set to "1". (SSR: RDRF=1)
- When the reception error is generated, the error flag is set to "1". (SSR:ORE, FRE) parity errors are not detected.

**Operating mode 2 (clock synchronous mode)**
- When the last reception bit is detected, the reception data load flag is set to "1". (SSR: RDRF=1)
- When a reception error is generated, the error flag is set to "1". (SSR:ORE) parity errors and framing errors are not detected.
Reception and timing of flag set are shown in Figure 18.4-1.

**Figure 18.4-1 Reception operation and flag set timing**

- **Clears of the reception data load flag and each reception error flag**
  - When the reception data load flag (SSR:RDRF) is cleared to "0" when the serial input data register (SIDR) is read.
  - The reception error flags (SSR:PE, ORE, FRE) are cleared to "0" when the reception error flag clear bit is set to "0".

**Note:** When a reception error is generated, set the reception error flag clear bit to "0" and clear each reception error flag to "0" after performing required error processing.

- **Timing of receive interrupt request generation**
  - When the reception interrupt enable bit is set to "1" (SSR:RIE=1), serial data register reception data load flag or error flag bit is set to "1" (SSR:RDRF, PE, ORE, FRE), reception interrupt requests are generated.
18.4.2 Timing of Send Interrupt Request Generation and Flag Setting

An interrupt request during a sending is generated when the serial output data register is ready to accept subsequent send data.

- **Timing of send interrupt request generation and flag setting**
  - **Setting and clearing send data empty flags**
    - The send data empty flag is set to "1" when the send data in the serial output data register is transferred to the send shift register and the next send data can be sent. The (SSR:TDRE=1) send data empty flag is cleared to "0" when the send data is written to the serial output data register.
    - Transmission and timing of flag set are shown in Figure 18.4-2.

**Figure 18.4-2 Transmission operation and flag set timing**

![Diagram showing timing of send interrupt request generation](image)

- **Timing of transmit interrupt request generating**
  - When the send interrupt enable bit is set to "1" (SSR:TIE=1) and the send data empty flag is set to "1", a send interrupt request is generated.

**Note:** Set so that sent data is written to the serial output data register when the send interrupt enable bit is set to "1" and cleared when the send data empty flag becomes "0".
18.5 UART Baud Rate

The UART transmission/reception clock is selected from among the following options:
- Dedicated baud rate generator
- Internal clock (16-bit reload timer 0 output)
- External clock (clock input to SCK pin)

Select of UART Baud Rate

The configuration of the UART baud rate selection circuit is shown in Figure 18.5-1. Set the baud rate according to the clock used as shown below.

- **Baud rate by dedicated baud rate generator**
  
  When using a dedicated generator as the clock input source, select one of five baud rates using the clock selection bit in the serial mode register. (SMR: CS2 to CS0)

- **Baud rate by internal timer**
  
  - Set the serial mode register clock selection bit to "110B" when internal clock supplied by the 16-bit reload timer 0 is used as the clock input source. (SMR: CS2 to CS0 = "110B")
  
  - The baud rate divides the clock provided by the internal timer by 2 in synchronous mode and by 32 in asynchronous mode.
  
  - Setting the 16-bit reload timer 0 reload register makes it possible to set any baud rate.

- **Baud rate by external clock**
  
  - Set the serial mode register clock selection bit to "111B" when an external clock input by the UART clock input pin (SCK0) is used as the clock input source. (SMR: CS2 to CS0 = "111B")
  
  - The baud rate uses the input clock in synchronous mode and divides the input clock by 16 in asynchronous mode.
  
  - The external clock can be set to any baud rate if below 2 MHz.
 CHAPTER 18 UART

Figure 18.5-1 UART Baud Rate Selector

SMR : CS2 to CS0
(Clock input source select bit)

Clock selector

[dedicated baud rate generator selection]
When setting to "000s" to "100s":

Division circuit
[Clock synchronization]
Select any divide ratio from 1/2, 1/4, 1/8, 1/16, 1/32.
[Clock asynchronous]
Select internal fixed divide ratio

[Internal timer]
TMCSR0 : CSL1, CSL0

[External clock]
SCK0

Pin

Clock selector

Down counter

UF

16-bit reload timer 0

[dedicated baud rate generator]

\( \phi \) \( \phi/2, \phi/3, \phi/4, \phi/5, \phi/6, \phi/7, \phi/8 \)

Communication prescaler
(CDCR : MD0, DIV3 to DIV0)

[internal timer selection]
When setting to "110s"

1/2 [Clock synchronous]
1/32 [Clock asynchronous]

[External clock selection]
When setting to "111s"

1/1 [Clock synchronous]
1/16 [Clock asynchronous]

UF : Underflow

: Machine clock
18.5.1 Baud Rate by Dedicated Baud Rate Generator

A dedicated baud rate generator output clock selected as a UART send/receive clock is described below.

Baud rate by dedicated baud rate generator

When the dedicated baud rate generator is used for generating the send/receive clock, use (1) or (2) to set the baud rate.

1. The machine clock division is set by the communication pre-scalar control register (CDCR) division ratio selection bit. (CDCR: DIV3 to DIV0)

2. The clock input to the dedicated baud rate generator from the communication pre-scalar is divided by the value in the clock selection bit in the serial mode register. (SMR: CS2 to CS0)

Although the machine clock divide ratio (1) of the communication prescaler is the same in the asynchronous and synchronous mode, the divide ratio (2) that determines the baud rate is different between the asynchronous and synchronous mode.

Figure 18.5-2 shows the baud rate selector based on the dedicated baud rate generator.

![Figure 18.5-2 Baud Rate Selector Based on Dedicated Baud Rate Generator](attachment:image)

Calculation expression for baud rate

The synchronous and asynchronous baud rate can be calculated using the following formula.

Asynchronous baud rate = \( \phi \times (\text{Prescaler divide ratio}) \times (\text{Asynchronous transfer clock divide ratio}) \)

Synchronous baud rate = \( \phi \times (\text{Prescaler divide ratio}) \times (\text{Synchronous transfer clock divide ratio}) \)

\( \phi \): Machine clock frequency
**Division ratio based on communication prescaler (common between asynchronous and clock synchronous modes)**

The machine clock divide ratio can be specified by the divide ratio selection bits (CDCR:DIV3 to DIV0) in the communication prescaler register, as shown in Table 18.5-1.

**Table 18.5-1 Output frequency of the communication prescaler (for 4 MHz and 10 MHz machine clocks)**

<table>
<thead>
<tr>
<th>Communication prescaler register (CDCR)</th>
<th>Machine clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD DIV3 DIV2 DIV1 DIV0</td>
<td>Divide ratio</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>Setting disabled</td>
</tr>
<tr>
<td>1 1 1 1 0</td>
<td>2</td>
</tr>
<tr>
<td>1 1 1 0 1</td>
<td>3</td>
</tr>
<tr>
<td>1 1 1 0 0</td>
<td>4</td>
</tr>
<tr>
<td>1 1 0 1 1</td>
<td>5</td>
</tr>
<tr>
<td>1 1 0 1 0</td>
<td>6</td>
</tr>
<tr>
<td>1 1 0 0 1</td>
<td>7</td>
</tr>
<tr>
<td>1 1 0 0 0</td>
<td>8</td>
</tr>
</tbody>
</table>

Cannot be set in excess of 4.25 MHz.
- Setting disabled

**Clock synchronous baud rate divide ratio**

The synchronous baud rate is generated by dividing the frequency of the output clock of the communication prescaler by 2, 4, 8, 16 or 32.

The divide ratio can be specified using the clock input source selection bits (SMR:CS2 to CS0) in the serial mode register, as shown in Table 18.5-2.

**Table 18.5-2 When output frequency of the communication prescaler is 2 MHz (clock cycle)**

<table>
<thead>
<tr>
<th>CS2</th>
<th>CS1</th>
<th>CS0</th>
<th>Divide ratio</th>
<th>Baud Rate (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2-frequency division</td>
<td>1M</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4-frequency division</td>
<td>500K</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>8-frequency division</td>
<td>250K</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>16-frequency division</td>
<td>125K</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>32-frequency division</td>
<td>62.6K</td>
</tr>
</tbody>
</table>
Asynchronous baud rate divide ratio

The asynchronous baud rate is generated by dividing the frequency of the output clock of the communication prescaler by the internal divide ratios.

The divide ratio can be specified using the clock input source selection bits (SMR:CS2 to CS0) in the serial mode register, as shown in Table 18.5-3.

Table 18.5-3 When the output frequency of the communication prescaler is 2 MHz (asynchronous)

<table>
<thead>
<tr>
<th>CS2</th>
<th>CS1</th>
<th>CS0</th>
<th>Divide ratio</th>
<th>Baud Rate (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8×13×2-frequency division</td>
<td>9,615</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8×13×4-frequency division</td>
<td>4,808</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>8×13×8-frequency division</td>
<td>2,404</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>8×13×16-frequency division</td>
<td>1,202</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>64-frequency division</td>
<td>31,250</td>
</tr>
</tbody>
</table>
18.5.2 Baud Rate by Internal Timer (16-bit Reload Timer 0)

This section explains the setting when selecting the internal clock supplied from the 16-bit reload timer 0 as the UART transmit/receive clock.

Baud Rate by Internal Timer (16-bit Reload Timer 0)

When selecting the baud rate from the internal timer, set the clock input source selection bits (SMR:CS2 to CS0= "110B") in the serial mode register to "110B".

The baud rate can be specified freely using the count clock divide ratio and the set reload register value by the 16-bit reload timer 0.

Figure 18.5-3 shows the baud rate selector based on the internal timer.

Figure 18.5-3 Baud Rate Selector by Internal Timer (16-bit Reload Timer 0 Output)

Calculation expression for baud rate

Asynchronous baud rate = \( \frac{\phi}{X(n + 1) \times 2 \times 16} \) bps

Clock synchronous baud rate = \( \frac{\phi}{X(n + 1) \times 2} \) bps

\( \phi \): Machine clock frequency

X: Count clock frequency divide ratio (2\(^1\), 2\(^3\), 2\(^5\)) for 16-bit reload timer 0

n: Set reload register value of the 16-bit reload timer 0 (0 to 65, 535)
An example of setting the baud rate and the reload register value (machine clock: 7.3728 MHz)

Table 18.5-4  Baud rate and reload register setting value

<table>
<thead>
<tr>
<th>Baud Rate (bps)</th>
<th>Reload register setting value</th>
<th>Clock Asynchronous (start-stop synchronization)</th>
<th>Clock synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X = 2^1 (machine clock 2-divided)</td>
<td>X = 2^3 (machine clock 8-divided)</td>
<td>X = 2^5 (machine clock 32-divided)</td>
</tr>
<tr>
<td>38,400</td>
<td>2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>19,200</td>
<td>5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9,600</td>
<td>11</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>4,800</td>
<td>23</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>2,400</td>
<td>47</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>1,200</td>
<td>95</td>
<td>23</td>
<td>5</td>
</tr>
<tr>
<td>600</td>
<td>191</td>
<td>47</td>
<td>11</td>
</tr>
<tr>
<td>300</td>
<td>383</td>
<td>95</td>
<td>23</td>
</tr>
</tbody>
</table>

X: Count clock frequency divide ratio for 16-bit reload timer 0
-: Setting disabled
18.5.3 Baud Rate by External Clock

This section explains the setting when selecting the external clock as the transmit/receive clock of the UART.

**Baud rate by external clock**

To select a baud rate based on the external clock input, make the following setting.

- Set the clock input source selection bits (SMR0: CS2 to CS0) in the serial mode register to 111B. (SMR: CS2 to CS0= "111B")
- Set the SCK0 pin as the input port in the port direction register (DDR).
- Set the serial clock I/O switching bit to "0" to use the SCK0 pin as the external clock input pin. (SMR: SCKE=0)

Set the baud rate based on the external clock input from SCK0 pin as shown in Figure 18.5-4. The internal divide ratios are fixed. To change the baud rate, change the external input clock cycle.

**Calculation expression for baud rate**

- Asynchronous baud rate = f/16
- Clock synchronous baud rate = f

f : External clock frequency (2 MHz max.)
18.6 Explanation of Operation of UART

The UART functions are comprised of the two-way serial communication function (operation mode 0, 2) and the master/slave type communication function (operation mode 1).

■ Operation of UART

● Operating mode

The UART operation mode has three types, mode 0 to 2. As shown in Table 18.6-1, one that is compatible with the connection system and the data communication system can be selected.

Table 18.6-1 UART operation modes

<table>
<thead>
<tr>
<th>Operating mode</th>
<th>Data length</th>
<th>Parity bit</th>
<th>Stop Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7 bit</td>
<td>8 bit</td>
<td>None</td>
</tr>
<tr>
<td>0 Asynchronous</td>
<td>Normal mode</td>
<td>-</td>
<td>o</td>
</tr>
<tr>
<td></td>
<td>(1-to-1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 Asynchronous</td>
<td>Multiprocessor mode</td>
<td>-</td>
<td>O(+1)</td>
</tr>
<tr>
<td></td>
<td>(1-to-n)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Clock synchronous</td>
<td>Clock synchronous mode</td>
<td>-</td>
<td>o</td>
</tr>
<tr>
<td></td>
<td>(1-to-1)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

○: Can be set
-: Setting disabled
+1: This is the address/data bit to be used for communication control.

Note: Only the master in a master/slave type connection can be used in UART operating mode 1.

● Connection system

As the connection system, select either 1-to-1 connection or master/slave type connection.

- To use 1-to-1 connection, set the operation mode to "0" in the asynchronous mode and set it to "2" in the synchronous mode. The CPU to be connected must be using the same operation mode.
- Can be used as the master by setting the operation mode to "1" when using master-slave type connection.

● Synchronization method

Set the asynchronous (start-stop synchronization) system or the clock synchronous system.

● Start of transmission/reception

- When the transmit data is set in the serial output data register and the transmit operation enable bit in the serial control register is set to "1" (SCR:TXE=1), the transmit operation will start.
- When the receive enable bit in the serial control register is set to "1" (SCR:RXE=1), the receiving operation will start.
● Stop of transmission/reception
  • When the transmit operation enable bit in the serial control register is set to "0" (SCR:TXE=0), the transmit operation will stop.
  • When the receive enable bit in the serial control register is set to "0" (SCR:RXE=0), the receiving operation will stop.

● Stop during transmission/reception
  • If the transmit operation enable bit is set to "0" (SCR:TXE=0) during the transmit operation, the transmit operation will stop after a transmission from the serial output data register is completed.
  • If the receive enable bit is set to "0" (SCR:RXE=0) during the receiving operation, the receiving operation will stop after one frame data being received is stored in the serial input data register.
18.6.1 Operation in Asynchronous Mode (Operation Mode 0 or 1)

When using UART in the operation mode 0 (normal mode) or operation mode 1 (multiprocessor mode), asynchronous communication can be implemented.

### Operation in Asynchronous Mode

- **Format of transmit/receive data**
  - The transmit and receive operations always start with the start bit ("L" level).
  - Data transmitting and receiving start from the LSB. (The least significant bit will be transmitted or received first.)
  - One frame transmitting and receiving end with the stop bit (1 or 2 bits) ("H" level).
  - When the operation mode is set to "0", the data length can be selected from 7 or 8 bits. You can select whether to use the parity bit.
  - In operation mode 1, the data length is fixed at 8 bits. Parity bit can not be added. The address/data bit (A/D) is added as bit 9.

Figure 18.6-1 shows the transmit/receive data format in the asynchronous mode.

**Figure 18.6-1 Format of Transmit/Receive Data (Operation Mode 0 or 1)**

<table>
<thead>
<tr>
<th>Operation mode 0</th>
<th>ST</th>
<th>D0</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ST</td>
<td>D0</td>
<td>D1</td>
<td>D2</td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
<td>D7</td>
<td>P</td>
</tr>
<tr>
<td>Operation mode 1</td>
<td>ST</td>
<td>D0</td>
<td>D1</td>
<td>D2</td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
<td>D7</td>
<td>A/D</td>
</tr>
</tbody>
</table>

ST : Start bit
SP : Stop bit
P : Parity bit
A/D : Operation mode 1 (multiprocessor mode) address/Data bit
CHAPTER 18 UART

● Transmission Operation

- The transmit data can be written to the serial output data register when the transmit data empty flag is set to "1" (SSR:TDRE=1).
- When the transmit data is written to the serial output data register, the transmit data empty flag is cleared to "0" (SSR:TDRE=0).
- When the transmit data is written and the transmit operation enable bit in the serial control register is set to "1" (SCR:TXE=1), the transmit operation will start.
- After the transfer of the transmit data from the serial output data register to the transmit shift register is completed, the transmit data empty flag is set to "1". (SSR: TDRE=1)
- If the transmit data empty flag is set to "1" when the transmit interrupt enable bit is set to "1" (SSR:TIE=1), a transmit interrupt request is generated. During an interrupt processing routine, the next transmit data can be written to the serial output data register.

● Reception Operation

- When the receive operation is set to the enabled state (SCR:RXE=1), the receive operation is always performed.
- When UART detects the start bit of the receive data, it receives one frame data into the serial input data register in accordance with the transfer format set in the serial control register.
- When the receiving of one frame data is completed, the receive data load flag is set to "1" (SSR:RDRF=1).
- When a receive error is generated, an error flag that corresponds with the error is set (SSR: PE, ORE or FRE is set to "1").
- When the receive interrupt enable bit is set to "1" (SSR:RIE=1) after the receiving of one frame data is completed, a receive interrupt request is generated.
- To read out the receive data, after the receiving of one frame data is completed, check the state of each error flag of the serial status register. If receiving is normally being done, read out the receive data from the serial input data register. When a receive error is being generated, perform error processing.
- When the receive data is read out, the receive data load flag is cleared to "0" (SSR:RDRF=0).

● Start bit detection method

Specify setting as follows for start bit detection:
- Immediately before the start of the communication period, be sure to set the communication line to "H" (mark level added).
- Set receive ready (RXE = "H") while the communication line is at "H" (mark level).
- Do not set receive ready (RXE = "H") during the non-communication period (mark level removed). Otherwise, data is not received correctly.
- After the stop bit is detected (the RDRF flag is set to "1"), set receive not ready (RXE = "L") while the communication line is at "H" (mark level).
Note that if receive ready is set at the timing represented in the following example, input data (SIN) is not recognized correctly by the microcontroller:

- Example of operation where receive ready (RXE = H) is set while the communication line is at "L".

**Stop Bit**

Selectable 1 bit or 2 bits. However, the receive side always detects only the first bit.

**Error detection**

- In operation mode 0, parity, overrun, and frame errors can be detected.
- In operation mode 1, overrun and frame errors can be detected.

**Parity bit**

A parity bit can be added only when the operation mode is set to "0".

Specifies the presence or absence of parity using the parity add enable bit (SCR:PEN) and specifies even or odd parity using the parity selection bit (SCR:P).

An example of transmit and receive data when a parity bit is added is shown in Figure 18.6-4.
Figure 18.6-4 Transmit and receive data when a parity bit is added

Parity error is generated with reception in even parity. (SCR : P=0)

Transmission by even parity (SCR : P=0)

Transmission by odd parity (SCR : P=1)

ST : Start bit
SP : Stop bit
Note: Parity can not be set in operating mode 1 and 2.
18.6.2 Operation at Clock Synchronous Mode (Operating Mode 2)

When the UART is used in operation mode 2, the transfer mode is clock synchronous.

■ Operation in Clock Synchronous Mode

- Format of transmit/receive data

In clock synchronous mode, 8-bit data is transmitted and received on an LSB-first basis. A start bit or stop bit cannot be added to the transmit and receive data.

Figure 18.6-5 shows the data format for the clock synchronous mode.

Figure 18.6-5 Format of Transmit/Receive Data (Operation Mode 2)
Clock Supply

When using the clock synchronization system, the supply of an 8-bit clock (the number of transmit and receive bits) is required.

- When an internal clock is output from the synchronous clock output side (exclusive baud rate generator or internal timer) (SMR:SCKE), the synchronous clock for data receiving is automatically generated when data is transmitted.
- When the synchronous clock output side is so set up to output an external clock (SMR:SCKE), check that the transmit data is in the serial output data register (SSR:TDRE=0) and supply a 1-byte clock from the outside. Make sure to have it brought back to the marked level ("H") before transmission and after reception.

Error detection

Only overrun errors can be detected. Parity and framing errors cannot be detected.

Setting of register

To transmit serial data from the transmit side to the receive side using the clock synchronous mode (operation mode 2), set up the control register as shown in Table 18.6-2.

<table>
<thead>
<tr>
<th>Table 18.6-2 Setting of Control Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register Name</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Serial mode register (SMR)</td>
</tr>
<tr>
<td>CS2, CS1, CS0</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SCKE</td>
</tr>
<tr>
<td>SOE</td>
</tr>
<tr>
<td>Serial controlled register (SCR)</td>
</tr>
<tr>
<td>CL</td>
</tr>
<tr>
<td>REC</td>
</tr>
<tr>
<td>TXE</td>
</tr>
<tr>
<td>RXE</td>
</tr>
<tr>
<td>Serial status register (SSR)</td>
</tr>
<tr>
<td>RIE</td>
</tr>
</tbody>
</table>

Notes:

- When an internal timer is used, set the baud rate of 16-bit reload timer 0 and supply the data to UART.
- Set the serial data input pin (SIN0) on the receiving side to the input pin on the port direction register (DDR).
● Starting communications

When the transmit data is written to the serial output data register, the transmit operation will start. To start a communication only when receiving is performed, write provisional transmit data to the serial output data register.

● Terminating communications

When the transmission and receiving of one frame data is completed, the receive data load flag is set to "1". (SSR: RDRF=1)

To read out the receive data, check the overrun error flag (SSR:ORE) and confirm whether communication was executed correctly.
18.6.3 Bidirectional Communication Function (Operation Modes 0 and 2)

In operation mode 0 and 2, usual two-way serial communication (1-to-1 communication) is enabled.

In operation mode 0, asynchronous communication is performed and in operation mode 2, clock synchronous communication is performed.

- **Bidirectional Communication Function**

  To use UART in operation mode 0 and 2, make a setting as shown in Figure 18.6-6.

  **Figure 18.6-6 UART operation mode 0, 2 setting**

<table>
<thead>
<tr>
<th>SCR, SMR</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>bit8</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode 0 →</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Mode 2 →</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSR, SIDR/SODR</th>
<th>PE</th>
<th>ORE</th>
<th>FRE</th>
<th>RDRF</th>
<th>TORE</th>
<th>–</th>
<th>RIE</th>
<th>TIE</th>
<th>Set transmit data (When write is specified) /Retain transmit data (When read is specified)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode 0 →</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>–</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Mode 2 →</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>–</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

  **DDR Port direction register**

  - **: Used bit
  - X: Unused bit
  - 1: Set to "1"
  - 0: Set to "0"

- **Inter-CPU connect**

  Connect the two CPUs as shown in Figure 18.6-7.

  **Figure 18.6-7 Example of Bidirectional Communication Connect for UART**

  ![Diagram of bidirectional communication connect for UART]
Communication procedure

The communication starts from the transmit side. The transmit side must write the transmit data to the serial output data register and set the transmit operation enable bit in the serial control register to "1" (SCR:TXE=1) before starting a transmission.

Figure 18.6-8 shows the example of transferring the receiving data to the sender to indicate that the data was successfully received.

**Figure 18.6-8 Flowchart for Bidirectional Communication**

```
(Transmit End)
Start

Setting Operating mode
(any of 0, 1 or 2)

Set 1 byte data to SODR
and communicate.

Receive data

YES
Transmitting data
(ANS)

NO

Read receive data
and execute

(Receive End)
Start

Setting Operating mode
(matching with the sending side)

Receive data

YES
Read receive data
and execute

NO
Transmit 1 byte data
```
18.6.4 Master/Slave Type Communication Function (Multi Processor Mode)

In operation mode 1, multiple CPUs can be connected in a master-slave configuration for communication. However, it should be noted that only the master side can be operated.

---

Master/Slave Mode Communication Function

To use UART in operation mode 1, make a setting as shown in Figure 18.6-9.

<table>
<thead>
<tr>
<th>SCR, SMR</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>bit8</th>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PEN</td>
<td>P</td>
<td>SBL</td>
<td>CL</td>
<td>AD</td>
<td>REC</td>
<td>RXE</td>
<td>TYE</td>
<td>MD1</td>
<td>MD0</td>
<td>CS2</td>
<td>CS1</td>
<td>CS0</td>
<td>BCH</td>
<td>SCKE</td>
<td>SOE</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSR, SIDR/SODR</th>
<th>PE</th>
<th>ORE</th>
<th>FRE</th>
<th>RDRF</th>
<th>TDRE</th>
<th>RIE</th>
<th>TIE</th>
<th>Set transmit data (When write is specified) /reatintransmit data (When read is specified)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
<td>🟪</td>
<td>🟪</td>
<td>🟪</td>
<td>🟪</td>
<td>🟪</td>
<td>🟪</td>
<td>Set the bit corresponding to the pin using as serial data input pin and serial clock input pin to &quot;0&quot;.</td>
</tr>
</tbody>
</table>

DDR Port direction register

- 🟪: Used bit
- X: Unused bit
- 1: Set to "1"
- 0: Set to "0"
Inter-CPU connect

Connects a single master CPU and multiple slave CPUs over two shared communication lines. UART can only be used on the master side. Figure 18.6-10 shows a connection example.

Figure 18.6-10 Connection example for UART master-slave communications

Function Selection

The operation mode and the communication system using a master-slave configuration are shown in Table 18.6-3.

Table 18.6-3 Select of Master/Slave Communication Function

<table>
<thead>
<tr>
<th>Operating mode</th>
<th>Master CPU</th>
<th>Slave CPU</th>
<th>Data</th>
<th>Parity</th>
<th>Synchronization method</th>
<th>Stop Bit</th>
</tr>
</thead>
</table>
| Address transmit/receive | Operation Mode 1 | -         | A/D=
"1"
+ 8-bit address | Not provided | Asynchronous | 1 bit or 2 bits |
| Data transmit/receive | -         | A/D=
"0"
+ 8-bit data | | | | |
● Communication procedure

The communication starts with the transmission of address data from the master CPU side.

- The master CPU side sets the address/data selection bit in the serial control register to "1" (SCR:A/D) and transmits the address data whose A/D bit is set to "1".
- The slave CPU side compares the received address data and the assigned address using the program and performs communication with the master CPU if they match.

Figure 18.6-11 shows the flowchart for master/slave communications.

**Figure 18.6-11 Flowchart for Master/Slave Communications**
18.7 Notes on Using UART

Pay attention to the following points when using UART.

---

## Notes on using UART

- **Enabling Operations**
  
  UART has transmit/receive enable bits. (Transmit SCR: TXE, Receive SCR: RXE)
  
  - Since the transmit and receive functions are both disabled in the initial state after a reset, the operation enable bit must be set to "1" before transmitting or receiving. (SCR: TXE = "1", SCR: RXE = "1")
  
  - If the transmit/receive operation enable bits are set to the transmit/receive disabled state, transmitting and receiving can be cancelled.

- **Setting operation mode**
  
  - Before setting the operation mode, set the operation enable bit to the disabled state to stop the operation. (SCR: TXE = "0", SCR: RXE = "0")
  
  - When the operation mode is switched during transmitting or receiving, the validity of the transmit and receive data cannot be guaranteed.

- **Synchronization mode**
  
  The operation mode 2 performs clock synchronous communication. Transmit/receive data is associated with no start and stop bits.

- **Timing of enabling send interrupt**
  
  Since the transmit data empty flag is set to "1" (transmit data is absent) by a reset (SSR: TDRE), when the transmit interrupt is set to the enabled state (SSR: TIE = 1), a transmit interrupt request is immediately generated. Be sure to first write the transmit data to the serial output data register and then set the transmit interrupt to the enabled state in the state where the transmit data empty flag is cleared to "0".

- **Clock setting at the clock synchronous mode**
  
  When the dedicated baud rate generator is used at the clock synchronous mode, the following settings are prohibited.
  
  - CS2 - CS0 = 000B
  
  - CS2 - CS0 = 001B and DIV3 to DIV0 = 0000B
18.8 Example of UART Programming

This section provides program example for UART.

### Example of UART programming

- **Processing specification**

  Perform serial transmission and reception using the two-way communication function of UART.
  - Make the following settings: Operation mode to 0, asynchronous, data length 8 bits, stop bit length 2 bits and no parity.
  - Use the P42/SIN0 and P43/SOT0 pins for communication.
  - Set the baud rate to approx. 9,600 bps using the exclusive baud rate generator.
  - Transmits the character "13H" from the SOT0 pin and receives it by an interrupt.
  - Set the machine clock (φ) to 16 MHz.

- **Coding example**

```assembly
ICR14   EQU  0000BEH    ;UART transmit interrupt control register
ICR13   EQU  0000BDH    ;UART receive interrupt control register
CDCR    EQU  000027H    ;Communication prescaler register
SMR     EQU  0000020H   ;Serial mode register
SCR     EQU  0000021H   ;Serial control register
SIDR    EQU  0000022H   ;Serial input data register
SODR    EQU  0000022H   ;Serial output data register
SSR     EQU  0000023H   ;Serial Status Register
REC     EQU  SCR:2      ;Receive error flag clear bit

;---------Main Program-----------------------------------------------
CODE    CSEG
START:  ;Stack pointer (SP)
         ;Already initialized
AND     CCR,#0BFH      ;Disables the interrupt.
MOV     I:ICR13,#00H   ;Interrupt levels 0 (strength)
MOV     I:ICR14,#00H   ;Interrupt levels 0 (strength)
MOV     I:CDCR,#10001000B ;f/8 operation
         ;When changing the divide ratio of the
         ;communication prescaler,
         ;The stabilization time is required.
MOV     I:SMR,#0000001B ;Operation mode 0 (Asynchronous)
         ;Selects the use of the exclusive generator at
         ;9615 bps.
         ;Disables the clock output and enables the data
         ;output.
MOV     I:SCR,#00110011B ;No Parity,
         ;Stop bit 2 bits, data length 8 bits,
         ;Clears the error flag and enables the transmit
         ;and receive operation.
MOV     I:SSR,#0000010B ;Disables the transmit interrupt and enables the
```


receive interrupt.

MOV I:SODR,#13H ; Writes the transmit data.
MOV ILM,#07H ; Sets ILM in PS to level 7
OR CCR,#40H ; Interruption is enabled

LOOP:
  •
  User processing
  •
  BRA LOOP

;--------- Interrupt Program -----------------------------------------------
WARI:
  ; Checks the receive error and fetches the receive data.
  MOV A, S IDR
  •
  User processing
  •
  RETI ; Returning from interrupt processing

CODE ENDS

;--------- Vector Settings -----------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FF68H ; Vector set in interrupt number #37(25H)
DSL WARI
ORG 00FFDCH ; Reset vector setting
DSL START
DB 00H ; Sets single-chip mode

VECT ENDS
END START
CHAPTER 19

I/O EXTENDED
SERIAL INTERFACE

In this chapter, the function and operation of the I/O extended serial interface are explained.

19.1 I/O Extended Serial Interface Outline
19.2 Block Diagram for the I/O Extended Serial Interface
19.3 Configuration of the I/O Extended Serial Interface
19.4 Interruption of the I/O Extended Serial Interface
19.5 Shift Clock
19.6 Explanations for the I/O Extended Serial Interface Operation
19.7 Instructions for Use of the I/O Extended Serial Interface
19.8 Program Example for I/O Extended Serial Interface
19.1 I/O Extended Serial Interface Outline

I/O extended serial interface performs data transfer using the clock synchronous method.
- The I/O extended serial interface is equipped with 2 channels in the MB90520A series.
- LSB first or MSB first can be selected for data transfer.

<table>
<thead>
<tr>
<th>I/O extended serial interface outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>The I/O extended serial interface functions as shown in Table 19.1-1.</td>
</tr>
</tbody>
</table>

Table 19.1-1 Function of the I/O extended serial interface

<table>
<thead>
<tr>
<th>Communication direction</th>
<th>USART Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simultaneous transmission and receipt (User decision relating to transmission or receipt is required.)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transfer mode</th>
<th>Clock synchronization method (transfer of data only)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transfer (shift) clock</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Intern 6 shift clock mode (The output clock of the communication prescaler is used as a shift clock by internal division.)</td>
<td></td>
</tr>
<tr>
<td>External shift clock mode (The input clock from the SCK pin is used as a shift clock. Shift clock generation is possible using the program.)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transfer rate</th>
<th>For the internal shift clock:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When the division rate between a machine clock frequency of 16 MHz and communication prescaler is set to 8, up to 1 MHz can be generated.</td>
</tr>
<tr>
<td></td>
<td>For external shift clock:</td>
</tr>
<tr>
<td></td>
<td>8 machine cycles or more (When the machine clock frequency is 16 MHz, up to 2 MHz can be input as the external shift clock frequency.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data transfer format</th>
<th>Selectable LSB first or MSB first</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Transfer of data only</td>
</tr>
<tr>
<td></td>
<td>Fixed 8-bit data length</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interrupt request generation</th>
<th>Interruption by the end of transfer</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Inter-CPU connection</th>
<th>2-way communication only</th>
</tr>
</thead>
</table>

Note: The I/O extended serial interface has no data buffer. Reception error cannot be detected.
19.2 Block Diagram for the I/O Extended Serial Interface

A block diagram of the I/O extended serial interface is shown in Figure 19.2-1.

**Figure 19.2-1** Block diagram for the I/O extended serial interface
CHAPTER 19 I/O EXTENDED SERIAL INTERFACE

● Details of Pins and Interrupt Numbers

The I/O extended serial interface is equipped with 2 channels.

(For I/O extended serial interface 1)

SIN1 pin: P45/SIN1
SOT1 pin: P46/SOT1
SCK1 pin: P47/SCK1
Serial I/O interrupt number: #15 (0FH)

(For I/O extended serial interface 2)

SIN2 pin: P50/SIN2
SOT2 pin: P51/SOT2
SCK2 pin: P52/SCK2
Serial I/O interrupt number: #17 (11H)

● Control circuit

Selection of the input clock and control of transmission/reception operation, etc. are performed.

● Shift clock counter

Bit number of the transfer data is counted.

● Serial mode control status register (SMCS)

Sets the selection of the shift clock, enabling serial I/O interruption, starting/stopping serial transfer, displaying transmission/reception status, selection of the transmission/reception bit direction, enabling the SOT pin output, and enabling the SCK pin output, etc.

● Serial data register (SDR)

The register retains the receive data.

● Communication prescaler register

It sets the start/stop of the communication prescaler and the division rate of machine clock. The clock used to divide the machine clock is supplied to the I/O extended serial interface.
19.3  Configuration of the I/O Extended Serial Interface

The I/O extended serial interface pin, interrupt factor, and register details are described.

- **I/O extended serial interface pins**
  The pin to be used for the I/O extended serial interface is simultaneously used for the general-purpose input/output port.
  The pin function and settings to be used for the I/O extended serial interface are shown in Table 19.3-1.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting required to use the extended serial I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O Extended serial interface 1</td>
<td>P45/SIN1</td>
<td>General-purpose I/O port, serial data input</td>
<td>Set to the dedicated input port with the port direction register (DDR)</td>
</tr>
<tr>
<td></td>
<td>P46/SOT1</td>
<td>General-purpose I/O port, serial data output</td>
<td>Set to output enable. (SMCS1: SOE=1)</td>
</tr>
<tr>
<td></td>
<td>P47/SCK1</td>
<td>General-purpose I/O port, serial Clock input</td>
<td>Set to the dedicated input port with the port direction register (DDR)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial clock output</td>
<td>Set to output enable. (SMCS1: SCOE=1)</td>
</tr>
<tr>
<td>I/O Extended serial interface 2</td>
<td>P50/SIN2</td>
<td>General-purpose I/O port, serial data input</td>
<td>Set to the dedicated input port with the port direction register (DDR)</td>
</tr>
<tr>
<td></td>
<td>P51/SOT2</td>
<td>General-purpose I/O port, serial data output</td>
<td>Set to output enable. (SMCS2: SOE=1)</td>
</tr>
<tr>
<td></td>
<td>P52/SCK2</td>
<td>General-purpose I/O port, Serial Clock input</td>
<td>Set to the dedicated input port with the port direction register (DDR)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial clock output</td>
<td>Set to output enable. (SMCS2: SCOE=1)</td>
</tr>
</tbody>
</table>

- **Block diagram of I/O extended serial interface pins**

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".

- **Generation of interrupt request with the I/O extended serial interface**
  The I/O extended serial interface can generate interrupt requests when data transmission/reception ends.
  When data transmission/reception ends, the serial I/O transmission/reception end flag of the serial mode control status register is set to "1" (SMCS:SIR = 1), and an interrupt request is generated when interruption is enabled (SMCS:SIE = 1).
19.3.1 Serial Mode Control Status Register Upper (SMCS: H)

The upper 8 bits (SMCS:H) of the serial mode control status register perform selection of the shift clock, enable serial I/O interrupt requests, check for the end of transmission/reception, start/stop serial transfers, and show the transfer status.

Serial mode control status register upper (SMCS:H)

Figure 19.3-1 Serial mode control status register upper (SMCS:H)

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMD2</td>
<td>SMD1</td>
<td>SMD0</td>
<td>SIE</td>
<td>SIR</td>
<td>BUSY</td>
<td>STOP</td>
<td>STRT</td>
<td>00000010B</td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
</tr>
</tbody>
</table>

### Bit Descriptions

- **Start bit (STRT)**
  - 0: No effect
  - 1: Start of transmission/reception

- **Stop bit (STOP)**
  - 0: Transmission/reception is enabled (Stopped status due)
  - 1: Transmission/reception is forcibly suspended (STOP status)

- **Bit displayed during transmission/reception (BUSY)**
  - 0: Suspended status, SDR read/write standby status
  - 1: Transmission/reception is being executed

- **Serial I/O send/receive end flag bit (SIR)**
  - When reading is performed:
    - 0: Serial I/O stop or during send/receive
    - 1: Serial I/O send/receive end
  - When write is specified:
    - 0: This SIR bit is cleared
    - 1: No effect

- **Serial I/O interrupt enable bit (SIE)**
  - 0: Disabling serial I/O interruption
  - 1: Enabling serial I/O interruption

- **Shift clock selection bit**
  - 0 0 0 2: Internal shift clock mode
  - 0 0 1 4: Shift clock frequency = Output of the communication prescaler/Division ratio
  - 0 1 0 16: Setting disabled
  - 0 1 1 32: Setting disabled
  - 1 0 0 64: Setting disabled
  - 1 0 1 External shift clock mode
  - 1 1 0 Setting disabled
  - 1 1 1 Setting disabled
Table 19.3-2 Functions of the serial mode control status register upper (SMCS:H)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 STRT: Start bit</td>
<td>The I/O extended serial interface is activated.</td>
</tr>
<tr>
<td>bit9 STOP: Stop Bit</td>
<td>Transmission/reception is forcibly suspended.</td>
</tr>
<tr>
<td>bit10 BUSY: Bit displayed during transmission/reception</td>
<td>Execution status of transmission/reception is indicated.</td>
</tr>
<tr>
<td>bit11 SIR: SIREQUSMC:11; serial I/O send/receive end flag bit</td>
<td>&quot;1&quot; will be set when transmission/reception ends.</td>
</tr>
<tr>
<td>bit12 SIE: Interrupt enable bit</td>
<td>Generation of serial I/O interrupt request is enabled or disabled.</td>
</tr>
<tr>
<td>bit15 to bit13 SMD2, SMD1, SMD0: Clock selection bit</td>
<td>The shift clock to be used is set.</td>
</tr>
</tbody>
</table>

**Note:**
1) "110B" and "111B" settings are disabled for these SMD2 to SMD0 bits.
2) Re-writing of these SMD2 to SMD0 bits is disabled during transmission/reception.
19.3.2 Serial Mode Control Status Register Lower (SMCS:L)

The lower 8 bits (SMCS:L) of the serial mode control status register perform selection of activation conditions, selection of transmission/reception bit direction, enable output of the SOT pin, and enable output of the shift clock to the pin.

### Serial mode control status register lower (SMCS:L)

![Figure 19.3-2 Serial mode control status register lower (SMCS:L)](image-url)
### Table 19.3-3  Function of the serial mode control status register lower (SMCS:L)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| bit0 SCOE: Shift clock output enable bit | Shift clock input/output of the SCK pin is switched.  
**When the bit is set to "0":** The pin is set as a general-purpose I/O port.  
**When the bit is set to "1":** This will be set to the shift clock output pin.  
  - When "0" is set to this SCOE bit, the shift clock can be supplied through the program by operating the general-purpose input/output port that is also used as the SCK pin.  
  **Reference:**  
  When shift clock output is enabled (SCOE = 1), it functions as the shift clock output pin regardless of the setting of the general-purpose input/output port. |
| bit1 SOE: Serial Output enable bit | Serial data output from the SOT pin is enabled or disabled.  
**When the bit is set to "0":** The pin is set as a general-purpose I/O port.  
**When the bit is set to "1":** The pin is set as a serial data output pin.  
  **Reference:**  
  When serial data output enable (SOE = 1) is set, it functions as the serial data output pin regardless of the setting of the general-purpose input/output port. |
| bit2 BDS: Transmission/reception bit Direction selection bit | This bit selects the transfer direction for serial data input/output.  
**When the bit is set to "0":** Transfer is carried out from the lowest bit (LSB) side.  
**When the bit is set to "1":** Transfer is carried out from the highest bit (MSB).  
  **Note:**  
  Set this BDS bit before writing data to the serial data register. |
| bit3 MODE: Activating condition selection bit | The condition to activate the I/O extended serial interface is set from the stop status once transmission/reception ends, or from the SDR read/write standby status.  
**When the bit is set to "0":** When "1" is set to the start bit of the serial mode control status register (SMCS:STRT), the I/O extended serial interface is initiated.  
**When the bit is set to "1":** I/O extended serial interface is activated by reading/writing on the serial data register (SDR).  
  **Note:**  
  1) Do not rewrite this MODE bit during transmission/reception.  
  2) Set this MODE bit to "1" to activate the extended intelligent I/O service. |
| bit7 to bit4 Unused bits | **Read:** The value is undefined.  
**Write:** No effect |
19.3.3 Serial Data Register (SDR)

Serial data register (SDR) retains the serial I/O transmission/reception data.

Serial data register (SDR)

Figure 19.3-3 shows bit configuration of serial data register.

![Figure 19.3-3 Serial data register (SDR)](image)

<table>
<thead>
<tr>
<th>bit7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
<th>Reset value</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>D6</td>
<td>D5</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
</tbody>
</table>

R/W : Readable and Writable
X : Undefined

- **Serial output operation**
  
  Functions as transmission Data Register. When transmission starts, data written to the serial data register is sent.

- **Serial input operation**
  
  Functions as reception Data Register. When reception starts, data received is stored in the serial data register.

**Note:**

- Do not read or write to the serial data register (SDR) during data transmission/reception (SMCS:BUSY = 1).
- When the serial data register (SDR) was read, it functions as the reception data register, and when the SDR was written on, it functions as the transmission data register. The command that operates read-modify-write (RMW) cannot be used.
19.4 Interruption of the I/O Extended Serial Interface

The I/O extended serial interface can generate interrupt requests when data transmission/reception ends. The hardware interrupt corresponds to the EI²OS.

- **Interruption of the I/O extended serial interface**
  
  The bit that controls interruption of the I/O extended serial interface and the interrupt factor are shown in Table 19.4-1.

<table>
<thead>
<tr>
<th>Serial I/O send/receive end flag bit</th>
<th>SMCS: SIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt enable bit</td>
<td>SMCS: SIE</td>
</tr>
<tr>
<td>Interrupt Factor</td>
<td>Completion of Transmission/Reception</td>
</tr>
</tbody>
</table>

When data transmission/reception ends, the serial I/O transmission/reception end flag of the serial mode control status register is set to "1" (SMCS:SIR = 1), and an interrupt request is generated when interruption is enabled (SMCS:SIE = 1).

(Clear serial I/O transmission/reception end flag bit (SMCS:SIR))

This will be cleared under the following conditions.

- When the activation condition selection bit is set to "0" (SMCS:MODE = 0) and the serial I/O transmission/reception end flag bit is set to "0". (SMCS:SIR=0)
- When the activation condition selection bit is set to "1" and the serial data register is in read/write mode.
- When "1" is set to the reset or stop bits. (SMCS:STOP=1)

- **Interruption of the I/O extended serial interface and EI²OS**

  Reference: Refer to "3.5 Interrupt" for the interrupt number, interrupt control register, and interrupt vector address.

- **EI²OS function of the I/O extended serial interface**

  The I/O extended serial interface supports the EI²OS function. The EI²OS can be activated by generating interrupt requests through completion of transmission/reception.

  Interruption of other peripheral functions that share the interrupt control register (ICR) needs to be disabled when the EI²OS function is used.
19.5 Shift Clock

The shift clock can select the internal shift clock mode and external shift clock mode.
- Under the internal shift clock mode, output of the communication prescaler can be divided internally and used as the shift clock.
- Under the external shift clock mode, the clock input from the shift clock input pin (SCK) is used as the shift clock.

![Shift clock generating circuit](image)

**Figure 19.5-1** Shift clock generating circuit

- **Pin**
  - SCK1
  - SCK2
- **DDR** (Port direction register)
- **PDR** (Port data register)
- **Shift clock output enable (SMCS : SCOE)**
- **Clock selector area**
  - When "000b" to "100b"
  - When "101b"
- **SMCS : SMD2 to SMD0**
- **CDCR : DIV3 to DIV0** (division selection)
- **Share UART**
- **Communication prescaler**
- **Internal clock division circuit**
- **Sift clock**

- Machine clock : \( \phi \)
- External shift clock : \( \rightarrow \)
- Internal shift clock : \( \rightarrow \)
- Shift clock generated by the software : \( \cdots \rightarrow \)

*When "000b" to "100b"
*When "101b"
Selection of shift clock

The shift clock can be selected from the internal shift clock and external shift clock.

- When the internal shift clock mode is selected, set the shift clock using the shift clock selection bits (SMCS:SMD2 to SMD0) of the serial mode control status register.
- Under the external shift clock mode, the shift clock is input from outside, and transmission/reception can also be performed using the clock generated by the software by writing "1" or "0" to the general-purpose input/output port that is also used as the shift clock input pin (SCK) using the program.

**Note:** Switch of shift clock mode while transmission/reception operation is not executed (SMCS:BUSY = 0).
19.5.1 Internal Shift Clock Mode

Under internal shift clock mode, output of the communication prescaler can be divided and used as the shift clock.

Internal shift clock mode

When internal shift clock mode is selected (SMCS:SMD2 to SMD0), set the division ratio for the communication prescaler output.

- The output clock (ϕ/2 to ϕ/8 and ϕ: machine clock frequency) of the communication prescaler is divided into 1/2 to 1/64 by the internal clock division circuit, and a 50% duty shift clock is generated.
- The selected shift clock can be output from the SCK pin through the pin control area.

Transmission/reception transfer rate can be calculated using the following formula.

Transfer rate (bit/s) = \( \frac{\phi}{(\text{div}\times M)} \)

ϕ: Machine clock frequency
div: Division ratio set by the DIV3 to DIV0 bits of the CDCR register (Table 19.5-1)
M: Division ratio set by the SMD2 to SMD0 bits of the SMCS register (Table 19.5-2)

Table 19.5-1 Division ratio set up using the communication prescaler (CDCR)

<table>
<thead>
<tr>
<th>CDCR set value</th>
<th>Division ratio (div)</th>
<th>Recommended machine cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD DIV3 DIV2 DIV1 DIV0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>2</td>
<td>4MHz</td>
</tr>
<tr>
<td>1 1 1 1 0</td>
<td>3</td>
<td>6MHz</td>
</tr>
<tr>
<td>1 1 1 0 0</td>
<td>4</td>
<td>8MHz</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>5</td>
<td>10MHz</td>
</tr>
<tr>
<td>1 1 1 0 0</td>
<td>6</td>
<td>12MHz</td>
</tr>
<tr>
<td>1 1 1 0 1</td>
<td>7</td>
<td>14MHz</td>
</tr>
<tr>
<td>1 1 1 0 0</td>
<td>8</td>
<td>16MHz</td>
</tr>
</tbody>
</table>
### Table 19.5-2 Division ratio set up using the SMD bit of the SMCS register

<table>
<thead>
<tr>
<th>Bit set value</th>
<th>Division ratio (M)</th>
<th>Internal shift clock</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Calculation formula</td>
<td>φ=16MHz, Calculation example when div=8.</td>
</tr>
<tr>
<td>SMD2</td>
<td>SMD1</td>
<td>SMD0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

φ: Machine clock frequency  
div: Division ratio of the machine clock using the communication prescaler  

**Note:** Output of the communication prescaler is also supplied as the UART serial clock, so consider any effects on the UART when the setting is modified.
19.5.2 External Shift Clock Mode

Under the external shift clock mode, the clock input from the shift clock input pin (SCK) is used as the shift clock. "1" or "0" are written to the general-purpose input/output port that is also used as the SCK pin by the program, and transmission/reception operation can be performed by the generated shift clock.

■ External shift clock mode

- External shift clock mode
  
  When the external shift clock mode is selected (SMCS:SMD2 to SMD0 = "101B"), the external clock to be input to the SCK pin is used as the shift clock.
  1-bit data will be transferred per clock in synchronization with the external shift clock to be input from the SCK pin.
  When the transfer rate is the 16 MHz machine clock frequency, the external shift clock frequency can be input to the SCK pin at up to 2 MHz (8 machine cycles or more).

- When the shift clock is supplied by the program
  
  The shift clock can be generated in accordance with the program using the general-purpose input/output port that is also used as the SCK pin.

  **(Set up when the shift clock is supplied by the program)**
  
  - External shift clock mode is set. (SMCR: SMD0 to 2 = "101B")
  - The SCK pin is set to the general-purpose input/output port. (SMCS: SCOE=0)
  - "1" is set to the bit supported by the port direction register (DDR) of the general-purpose input/output port that is also used as the SCK pin, and the output port is set.

  **(Generation of the shift clock by the program)**
  
  When "1" and "0" are set alternately to the port data register (PDR) that supports the SCK pin, the general-purpose input/output port value to be output to the SCK pin is loaded as the external shift clock, and transmission/reception operation is performed.
  
  - Since the SCK pin operates as the shift clock output pin, start writing the shift clock from "1" ("H" level).
19.6 Explanations for the I/O Extended Serial Interface Operation

The I/O extended serial interface transmits/receives 8-bit data in synchronization with the shift clock.

■ Transmission/reception operation of the I/O extended serial interface

- I/O timing of serial data
  - Data from the serial data register (SDR) is output to the serial output pin (SOT) in synchronization with the falling edge of the serial shift clock (external shift clock or internal shift clock).
  - The input signal for the serial input pin (SIN) is input to the serial data register (SDR) in synchronization with the rising edge of the serial shift clock.

- Data transfer (shift) direction
  The bit shift direction (transfer from the MSB or LSB) can be set by the transmission/reception bit direction selection bit (SMCS:BDS) of the serial mode control status register.

- Operating clock
  - The operation clock is set to internal or external shift clock mode by the shift clock selection bits (SMCS:SMD2 to SMD0) of the serial mode control status register.
  - When the external clock mode is set, the shift clock can be generated by the program.

■ Start and stop transmission/reception operation of the I/O extended serial interface

- Activation of the I/O extended serial interface and start of transmission/reception
  When the stop bit of the serial mode control status register is set to "0" and the start bit is set to "1", the I/O extended serial interface is activated, and transmission/reception starts. (SMCS: STOP=0, STRT=1)

- Completion of 1-byte transmission/reception
  When 1-byte transmission/reception is complete, the serial I/O transmission/reception end flag is set (SMCS:SIR = 1), and if "0" is set for the activation condition selection bit (SMCS:MODE = 0), will move to stop status, and if "1" is set to the activation condition selection bit (SMCS:MODE = 1), will move to the SDR read/write standby status.

- Transmission/Reception
  When transmission/reception is forcibly suspended by setting the stop bit to "1" (SMCS:STOP = 1), the serial I/O transmission/reception end flag retains clear status (SMCS:SIR = 0), and will move to stop status.
● Re-start after completing transmission/reception

[When the activation condition selection bit is set to "0" (SMCS: MODE=0)]
- When the start bit of the serial mode control status register is set to "1" (SMCS:STRT = 1), transmission/reception is re-started by activating the I/O extended serial interface.

[When the activation condition selection bit is set to "1" (SMCS: MODE=1)]
- When read/write was performed on the serial data register (SDR), the I/O extended serial interface is activated and transmission/reception is re-started.

● Re-start after forced suspension

[When the activation condition selection bit is set to "0" (SMCS: MODE=0)]
- When the stop bit of the serial mode control status register is set to "0" and the start bit is set to "1" (SMCS:STOP = 0 and STRT = 1), the I/O extended serial interface is activated, and transmission/reception re-starts.

[When the activation condition selection bit is set to "1" (SMCS: MODE=1)]
- While the stop bit of the serial mode control status register is set to "0" and the start bit is set to "1" (SMCS:STOP=0, STRT=1), and when read/write is performed on the serial data register (SDR), the I/O extended serial interface is activated, and transmission/reception re-starts.

■ I/O extended serial interface set up

● Internal shift clock mode

Setting as per Figure 19.6-1 is required to use the internal shift clock.

Figure 19.6-1 Setting of transmission/reception operation (Internal shift clock)

<table>
<thead>
<tr>
<th>SMCS</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SMD2</td>
<td>SMD1</td>
<td>SMD0</td>
<td>SIE</td>
<td>SIR</td>
<td>BUSY</td>
<td>STOP</td>
<td>STRT</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>000B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"000B" to "100B"

SDR Read/Write transfer data

DDR (Port direction register)

**: Used bit
1 : set to "1"
− : Undefined bit

Corresponding bit to SIN pin and SCK pin is set to "0"
● External shift clock mode

Setting as per Figure 19.6-2 is required when the external shift clock is used.

**Figure 19.6-2 Set of transmission/reception operation (External shift clock)**

<table>
<thead>
<tr>
<th>SMCS</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMC1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

SDR (Port direction register)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Read/Write transfer data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Used bit
- 0: Set "0"
- 1: set to "1"
- #: Undefined bit

- Corresponding bit to SIN pin and SCK pin is set to "0"

● When the shift clock is supplied by the program under external shift clock mode

Setting as per Figure 19.6-3 is required to supply the shift clock using the program.

**Figure 19.6-3 Setting for transmission/reception operation (Supply of the shift clock by the program)**

<table>
<thead>
<tr>
<th>SMCS</th>
<th>bit15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMC1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

SDR (Port direction register)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Read/Write transfer data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Used bit
- 0: Set "0"
- 1: set to "1"
- #: Undefined bit

- Corresponding bit to SIN pin is set to "0", and corresponding bit to SCK pin is set to "1"
19.6.1 Operating Status of the I/O Extended Serial Interface

There are four operating status types for the I/O extended serial interface, namely STOP, suspend, SDR read/write standby, and transmission/reception execution statuses.

Figure 19.6-4 shows I/O extended serial interface operation transition diagram.

Stop state

This will change to STOP status after reset. This will also change to STOP status when "1" is set to the stop bit (SMCS:STOP = 1) of the serial mode control status register.

- When changing to STOP status, the shift clock counter is initialized, and the serial I/O transmission/reception end flag bit of the serial mode control status register is cleared to "0". (SMCS: SIR=0)
• Returning from STOP status to transmission/reception operation status is performed as follows in accordance with the set up for the activation condition selection bit (SMCS:MODE) of the serial mode control status register.

  - When MODE = "0" is set, "0" is set to the stop bit of the serial mode control status register, and "1" is set to the start bit. (SMCS: STOP=1, SMCS: STRT=1)
  - When MODE = "1" is set, read/write is performed on the serial data register (SDR) by setting "0" to the stop bit and "1" to the start bit.

Note: As the stop bit is handled as a higher priority than the start bit, when the stop bit is "1", transmission/reception operation is not performed even if "1" is set for the start bit.

● Stop state

  This will move to stop status when the transfer ends while the activation condition selection bit is set to (SMCS: MODE=0) activation start bit.

  • When transmission/reception ends, the during transmission/reception display bit is cleared (SMCS:BUSY = 0), the serial I/O transmission/reception end flag is set (SMCS:SIR = 1), and the shift clock counter is initialized, and it changes to stop status.
  • Interrupt requests are generated when serial I/O interrupts are enabled (SMCS:SIE = 1).
  • Set "1" to the start bit to return from the stop status to the transmission/reception execution status. (SMCS: STRT=1)

● Read/Write waiting state to SDR

  This will change to SDR read/write standby status when the transfer ends while the activation condition selection bit is set to (Activate by SDR read/write) (SMCS:MODE = 1).

  • When the transfer ends, the send/receive display bit (SMCS:BUSY = 0) is cleared, the serial I/O send/receive end flag is set (SMCS:SIR=1) and SDR read/write standby status is invoked.
  • Interrupt requests are output if serial I/O interrupt is set to enabled (SMCS:SIE=1).
  • A read or write operation of the serial data register (SDR) enables the register to return from SDR read/write standby status to send/receive operation.

● Transmission/Reception executing state

  Reception status (SMCS:BUSY=1).

  Stopped state or SDR read/write status is invoked after sending/receiving ends depending on how the startup status selection bit is set.
19.6.2 Timing of I/O Extension Serial Interface Operations

Timing of start and end of data send/reception are described below.

■ Timing of I/O extension serial interface operations

- Check of transfer status

  The send/receive display bit (SMCS:BUSY) of the serial mode control status register is set to "1" while sending/receiving and set to "0" in the stop state or in SDR read/write standby status.

  To check send/receive status, read the send/receive display bit (SMCS:BUSY) of the serial mode control status register.

  Figure 19.6-5 to Figure 19.6-8 show start/finish timing of transmission/reception operation in each mode.

---

**Figure 19.6-5 Internal Shift Clock Mode (LSB first)**

- SCK pin
  - (Transmit/Receive start)
  - (Transmit/Receive end)

- STRT pin
  - When MOD set to 0

- BUSY bit
  - Read value is always "0" Writing "1"

- SOT pin
  - DO7 to DO0: shows Output data bit

- DO7 to DO0: shows Output data bit

---

**Figure 19.6-6 External Shift Clock Mode (LSB first)**

- SCK pin
  - (Transmit/Receive start)
  - (Transmit/Receive end)

- STRT bit
  - When MOD set to 0

- BUSY bit
  - Read value is always "0" Writing "1"

- SOT pin
  - DO7 to DO0: shows Output data bit

- DO7 to DO0: shows Output data bit
**Figure 19.6-7  When a program shift is performed in the external shift clock mode (LSB first)**

![Diagram](image)

- SCK pin: "0" is written to SCK bit of PDR, "1" is written to SCK bit of PDR, "0" is written to SCK bit of PDR.
  - (Transmit/Receive end)
- STRT bit: When MOD set to "0".
  - Read value is always "0".
- BUSY bit: (Transmit/Receive start)
- SOT pin: DO6 to DO7 (Data retention)

**Note:** In Shift clock by the program, when "1" is written to the bit corresponding to SCK of PDR, "H" is output, and when "0" is written, "L" is output (where external shift clock mode is selected, and SCOE=0).

**Figure 19.6-8 Using STOP bit for forcible termination**

![Diagram](image)

- SCK pin: (Transmit/Receive start) (Forcible termination)
- STRT bit: When MOD set to "0".
  - Reading "1"
- BUSY bit: (Forcible termination)
- STOP bit: (Forcible termination)
- SOT pin: DO3 to DO5 (Data retention)

**Note:** In Shift clock by the program, when "1" is written to the bit corresponding to SCK of PDR, "H" is output, and when "0" is written, "L" is output (where external shift clock mode is selected, and SCOE=0).
19.6.3 Serial Data I/O Shift Timing

During serial data transfers, data is output from the serial data output terminal (SOT) when the shift clock drops while data is input from the serial data input pin (SIN) when the shift clock rises.

- **Shift timing of serial data I/O**
  
  Figure 19.6-9 shows serial data I/O shift timing in LSB first and MSB first.

![Figure 19.6-9 Serial data I/O shift timing](image)

- **LSB - first (if the BDS bit is 0)**
  - Latch SIN input at rising
  - SOT output at rising

- **MSB - first (if the BDS bit is 0)**
  - Latch SIN input at rising
  - SOT output at falling

\[\text{DI7 to DI0} : \text{shows Input data bit}\]
\[\text{DO7 to DO0} : \text{shows Output data bit}\]
19.7 Instructions for Use of the I/O Extended Serial Interface

Note the following when using the I/O expansion serial interface.

- **Instructions for use of the I/O extended serial interface**

  - Deviation at the starting of sending/receiving
    
    Since the shift clock is not synchronized with the startup (SMCS:STRT=1) by the send/receive start bit, there is a shift clock delay of at most 1 cycle until the first serial data is input/output.

  - Malfunction due to noise
    
    When the shift clock is affected by external noise pulses (pulses exceeding hysteresis) during sending or receiving, the extended I/O serial interface may not function.

  - Precaution on program setting
    
    Write to bits other than the start bit (STRT) and stop bit (STOP) of the serial data register (SDR) and serial mode control status register (SMCS) only when the extended I/O serial interface is halted.
    
    - When stopping sending/receiving (SMCS:STOP=1) and end of sending/receiving occurs simultaneously, the serial I/O send/receive end flag bit (SMCS:SIR) is not set.
    - The extended I/O serial interface is not provided with an error-detection function. Use timer, check data and take other measures.
    - The I/O extended serial interface has no data buffer. Measures to set the extended I/O serial interface interrupt priority to high-order must be taken when the shift clock is fast or used for multiple interrupts.

  - Precaution on internal clock mode
    
    The communication prescaler is not used with UART. When the telecommunication pre-scalar setting is changed, the extended I/O serial interface and UART may have to be adjusted.

  - Shift clock idle state
    
    - In the external clock mode, set the external shift clock to "H" level during the standby time (idle state) between 8-data transfers.
    - The internal shift clock outputs "H" level signals in idle state when the shift clock output is set (SMCS:SCOE=1).

    Figure 19.7-1 shows shift clock idle state.

    - **Figure 19.7-1 Shift clock idle state**

      ![Shift clock idle state diagram]

      - Idle state
      - 8-bit Data Transfer
      - Idle state
19.8 Program Example for I/O Extended Serial Interface

The following is an example of the extended I/O serial interface program.

### Program example for I/O extended serial interface

#### Processing specification

- Communication function of the extended I/O serial interface is used to transfer serial data.
- Use the P45/SIN1, P46/SOT1 and P47/SCK1 pins for communication.
- The MODE=0 STRT bit starts sending and reception.
- The SOT1 pin sends character "13H" and performs reception during interrupts.
- The machine clock (φ) 16 MHz is divided so that the communication pre-scalar frequency ratio is set to 8 and the internal clock division ratio is set to 2.
- The transfer rate is as follows.
  Transfer rate (bit/s) = φ / (div M) = 16MHz / (8x2) = 1MHz

#### Coding example

```assembly
;DDR4    EQU 000014H ;Port 4 direction register
SMCS   EQU 000024H ;serial mode control status
;Register
CDCR   EQU 000027H ;Communication prescaler register
SDR    EQU 000026H ;Serial data register
ICR02  EQU 0000B2H ;Interrupt control register
STRT   EQU SMCS:8 ;start bit
SIR    EQU SMCS:11 ;end of serial I/O send/receive
;flag bit

;----------Main Program-----------------------------------------------
CODE  CSEG
START:
; ;Stack pointer (SP)
AND CCR,#0BFH ;Interruption is disabled
MOV I:ICR02,#00H ;Interrupt levels 0 (strength)
MOV I:CDCR,#10001000B ;communication pre-scalar setting (8/f)
MOV I:SMCS,#0001000000000011B ;Internal clock selection, interruption is enabled,
;Serial I/O send/receive end flag clear
;Normal operation status, MODE=0,
;LSB first, clock output enabled
;serial data output
MOV ILM,#07 ;Interrupt mask level is set,
OR CCR,#40H ;Interruption is enabled
MOV I:SDR,#13H ;Transfer data (13H)writing
SETB I:STRT ;Serial data send start
;:
;----------Interrupt Program------------------------------------------
```


WARI: MOV A,I:SDR0 ;Outputting transfer data  
CLRB I:SIR ;Serial I/O send/receive end  
;flag clear  
User processing  
RETI  
CODE ENDS

;----------Vector Settings--------------------------------------------
VECT CSEG ABS=0FFH
ORG 0FFC0H ;Vector set up to the interruption number #15 (0FH)
DSL WARI
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START

Program example 2 for I/O extended serial interface

● Processing specification
- Communication function of the extended I/O serial interface is used to transfer serial data.
- Use the P45/SIN1, P46/SOT1 and P47/SCK1 pins for communication.
- Send/receive starts when data is written to the SDR register in MODE=1.
- The SOT1 pin sends character "13H" and performs sending and reception during interrupts.
- The shift clock is generated and is output by the program.

● Coding example
PDR4 EQU 000004H ;Port 4 data register
DDR4 EQU 000014H ;Port 4 direction register
SMCS EQU 000024H ;serial mode control status register
SDR EQU 000026H ;Serial data register
CDCR EQU 000027H ;Communication prescaler register
ICR02 EQU 0000B2H ;Interrupt control register
STRT EQU SMCS:8 ;start bit
SIR EQU SMCS:11 ;serial I/O send/receive end flag bit
PSCK EQU PDR4:7 ;port 4 bit 7

;---------------Main Program------------------------------------------
CODE CSEG
START:
;Stack pointer (SP)
;Shall be initialized
AND CCR,#0BFH ;Interruptation is disabled
MOV I:ICR02,#00H ;Interrupt levels 0 (strength)
MOV I:DDR4,#10000000B ;P45/SIN1 pin input,
;Output P47/SCK1 pin
MOV I:CDCR,#10001000B ;communication pre-scalar setting (8/f)
MOV I:SMCS,#10110010001010B ;external clock selection, interrupt enabled
;Serial I/O send/receive end flag clear
;Stop status, MODE=1, ;LSB first, output for each instruction ;Serial data output
MOV ILM,#07 ;Interrupt mask level is set, 
OR CCR,#40H ;Interruption is enabled 
MOV I:SDR,#13H ;Transfer data (13H)writing, ;Serial data transmission start
MOV A,8 ;"L"→"H" eight times 
           ;Shift clock
LP1:  CLRB I:PSCK ;"L" output
       SETB I:PSCK ;"H" output
       DEC A
       BNZ LP1 ;"H" is output upon exiting the LP1 loop

;----------------Interrupt Program-----------------------------
WARI: MOV A,I:SDR ;Outputting transfer data
       CLRB I:SIR ;serial I/O send/receive end flag clearing
       •
       User processing
       •
       RETI
CODE ENDS

;----------------Vector Settings-------------------------------
VECT CSEG ABS=0FFH
ORG 0FFC0H ;Vector set up to the interruption number #15 (0FH)
DSL WARI
ORG 0FFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START
In this chapter, the functions and operation of the LCD controller/driver are explained.

20.1 LCD Controller/Driver Outline
20.2 Block Diagram for the LCD Controller/Driver
20.3 Configuration of the LCD Controller/Driver
20.4 Explanations for the LCD Controller/Driver Operation
20.5 Program Example for the LCD Controller/Driver
20.1 LCD Controller/Driver Outline

The LCD controller/driver directly drives the liquid crystal display (LCD)
- The LCD is driven by 4 common outputs and 32 segment outputs.
- Data memory for 16-byte display is built in.
- 3 types of duty output mode can be selected.

■ LCD controller/driver outline

The LCD controller/driver directly shows details of the data memory for display on the LCD via the common output and segment output.
- Although divided resistance of the LCD activation power source is built in, it can also be connected externally.
- Up to four common outputs (COM0 to COM3 pins) and 32 segment outputs (SEG0 to SEG31 pins) can be used.
- Data memory for 16-byte display is built in.
- Duty setting is selected from 1/2, 1/3, 1/4. Bias is fixed on the 1/3 bias setting.
- Oscillation clock (HCLK) or sub-clock (SCLK) is used as the drive clock.
- The LCD can directly be driven.
- The COM0 to COM3 pins and SEG8 to SEG31 pins can be used as general-purpose input/output ports.
  (Switching between the LCDCMR register and LCR1 register)

The bias and duty settings that can be used and the common output to be used are as per Table 20.1-1.

<table>
<thead>
<tr>
<th>Bias</th>
<th>1/2 duty output mode</th>
<th>1/3 duty output mode</th>
<th>1/4 duty output mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/3 bias setting</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>COM1 output and COM0 output</td>
<td>COM2 output to COM0 output</td>
<td>COM3 output to COM0 output</td>
<td></td>
</tr>
</tbody>
</table>

○: Usable mode
20.2 Block Diagram for the LCD Controller/Driver

The LCD controller/driver is made up of the following blocks. The function can be separated into the controller part that generates the segment signal and common signal in accordance with the content of the display data memory, and the driver part that drives the LCD.

- Control area (common pin switching register, LCDC control register 0/1, prescaler, timing controller, data memory for display)
- Driver area (divided resistance, common driver, segment driver, alternating circuit)

■ Block diagram for the LCD controller/driver

![Diagram of LCD controller/driver](image-url)
LCDC control register 0 (LCR0)

Selects the clock for frame cycle generation, control of the power source driving the LCD, selection of whether the LCD is to display or not (blanking), selection of the display mode, and cycle selection of the LCD frame.

LCDC control register 1 (LCR1)

The input/output pin function can be switched to the segment output or general-purpose input/output port.

Common pin switch register (LCDCMR)

Switch the input/output pin function to the common output or general-purpose input/output port.

Data memory for display

This is 16-byte RAM for segment output signal generation. The RAM contents are automatically read in synchronization with the common signal timing, and output from the segment output pins (SEG0 to SEG31).

Prescaler

The clock for frame cycle generation (oscillation clock or sub-clock) is divided, and frame cycle is output.

Timing controller

Output timing for the common signal and segment signal are synchronized based on the setting of the frame cycle and LCR0.

Circuit of making to exchange

Generates AC waveform for LCD driving from the timing controller signals.

Common driver

This is the LCD common pin driver.

Segment driver

This is the LCD segment pin driver.

Internal divided resistance

This resistance divides the power source for driving the LCD, and generates power voltage for driving. The internal divided resistance is cut off, and divided resistance can be connected externally. The V0 to V3 pins are the divided resistance connection pins.

Details of pin

Actual Pin Name is as following.
COM0 to COM3 pin: P74/COM0 to P77/COM3
SEG0 to SEG7 pin: Dedicated pins for segment output
SEG8 to SEG15 pin: PA0/SEG8 to PA7/SEG15
SEG16 to SEG23 pin: P80/SEG16 to P87/SEG23
SEG24 to SEG31 pin: P90/SEG24 to P97/SEG31
20.3 Configuration of the LCD Controller/Driver

The pin and register details of the LCD controller/driver are described.

■ LCD controller/driver pins

The pin used for the LCD controller/driver is also used as a general-purpose input/output port. The pin function and necessary settings to be used for the LCD controller/driver are shown in Table 20.3-1.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting of pins for using of LCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>P74/COM0 to P77COM3</td>
<td>General-purpose I/O port, LCD common power output</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>SEG0 to SEG7</td>
<td>Segment output</td>
<td>Not required for setting</td>
</tr>
<tr>
<td>PA0/SEG8 to PA3/SEG11</td>
<td>General-purpose I/O port, Segment output</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>PA4/SEG12 to PA7/SEG15</td>
<td>Segment output</td>
<td></td>
</tr>
<tr>
<td>P80/SEG16 to P83/SEG19</td>
<td>General-purpose I/O port, Segment output</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P84/SEG20 to P87/SEG23</td>
<td>Segment output</td>
<td></td>
</tr>
<tr>
<td>P90/SEG24 to P93/SEG27</td>
<td>General-purpose I/O port, Segment output</td>
<td>Set as input port in port direction register (DDR).</td>
</tr>
<tr>
<td>P94/SEG28 to P97/SEG31</td>
<td>Segment output</td>
<td></td>
</tr>
<tr>
<td>V0 to V3</td>
<td>Power supply for LCD driving</td>
<td>Not required for setting</td>
</tr>
</tbody>
</table>

Note: When SEG8 to SEG11 or SEG12 to SEG15 are used as the segment outputs, if pins that are not used as segment outputs are simultaneously used as ports, input will be cut off, so they can only be used as output ports. Combinations of SEG16 to SEG19 and SEG20 to SEG23, and SEG24 to SEG27 and SEG28 to SEG31 will be the same. In the same way, when COM0 is used as the COM pin while COM1 to COM3 are used as ports, it can only be used as an output port.

■ Block diagram of LCD controller/driver pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".
20.3.1 Common Pin Switching Register (LCDCMR)

The common pin switching register switches the input/output pin function to common output or general-purpose input/output port.

- **Common pin switching register (LCDCMR)**

![Common pin switching register (LCDCMR) diagram](image_url)

- **COM0**: Common terminal (COM0) switching bit
  - 0: Functions as general-purpose I/O port
  - This functions as a general-purpose output port if any of bit9 to bit11 are set to "1".
  - 1: Functions as Common output terminal

- **COM1**: Common terminal (COM1) switching bit
  - 0: Functions as general-purpose I/O port
  - This functions as a general-purpose output port if any of bit8, bit10 to bit11 are set to "1".
  - 1: Functions as Common output terminal

- **COM2**: Common terminal (COM2) switching bit
  - 0: Functions as general-purpose I/O port
  - This functions as a general-purpose output port if any of bit8 to bit9, bit11 are set to "1".
  - 1: Functions as Common output terminal

- **COM3**: Common terminal (COM3) switching bit
  - 0: Functions as general-purpose I/O port
  - This functions as a general-purpose output port if any of bit8 to bit10 are set to "1".
  - 1: Functions as Common output terminal

- **Reset value**: XXXX0000b
### Table 20.3-2  Common pin switching register (LCDCMR) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 COM0: Common pin (COM0) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The pin is set as a general-purpose I/O port. This functions as a general-purpose output port if any of bit 11 to bit 9 are set to &quot;1&quot;. <strong>When the bit is set to &quot;1&quot;:</strong> The common pin will be set.</td>
</tr>
<tr>
<td>bit9 COM1: Common pin (COM1) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The pin is set as a general-purpose I/O port. This functions as a general-purpose output port if any of bit 11, bit 10, bit 8 are set to &quot;1&quot;. <strong>When the bit is set to &quot;1&quot;:</strong> The common pin will be set.</td>
</tr>
<tr>
<td>bit10 COM2: Common pin (COM2) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The pin is set as a general-purpose I/O port. This functions as a general-purpose output port if any of bit 11, bit 9, bit 8 are set to &quot;1&quot;. <strong>When the bit is set to &quot;1&quot;:</strong> The common pin will be set.</td>
</tr>
<tr>
<td>bit11 COM3: Common pin (COM3) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The pin is set as a general-purpose I/O port. This functions as a general-purpose output port if any of bit 10 to bit 8 are set to &quot;1&quot;. <strong>When the bit is set to &quot;1&quot;:</strong> The common pin will be set.</td>
</tr>
<tr>
<td>bit12 to bit15 Unused bits</td>
<td><strong>Read:</strong> The value is undefined. <strong>Write:</strong> No effect</td>
</tr>
</tbody>
</table>
### 20.3.2 LCDC Control Register 0 (LCR0)

The LCDC control register 0 (LCR0) selects the frame cycle and the clock for frame cycle generation, selects the display mode and whether to display or not, sets enabling operation of the LCD controller/driver under the clock mode, and controls driving power source.

#### LCDC control register 0 (LCR0)

![Figure 20.3-2 LCDC control register 0 (LCR0)]

- **Reset value**: 00010000B
- **bit0**: Frame cycle selection bit
  - FP1: When Oscillation clock (HCLK) is selected. (CSS=0)
  - FP0: When Sub clock (HCLK) is selected. (CSS=1)
  - N: time division number (by setting of display mode select bit)
  - HCLK: Oscillation clock frequency
  - SCLK: Sub clock frequency

- **bit1**: Display mode selection bit
  - MS1 MS0: LCD operation stop
  - 0 0: LCD operation stop
  - 0 1: 1/2 Duty output mode (time division number = 2)
  - 1 0: 1/3 Duty output mode (time division number = 3)
  - 1 1: 1/4 Duty output mode (time division number = 4)

- **bit2**: Blanking selection bit
  - BK: Show
  - 0: Show
  - 1: Hide (Blanking)

- **bit3**: LCD driving power control bit
  - VSEL: Internal division resistance is disconnected.
  - 0: Internal division resistance is disconnected.
  - 1: Internal division resistance is connected.

- **bit4**: Operation enabling bit under clock mode
  - LCEN: Stop in clock mode
  - 0: Stop in clock mode
  - 1: Also operating in clock mode

- **bit5**: clock select bit for frame frequency generation
  - CSS: Oscillation clock (HCLK)
  - 0: Oscillation clock (HCLK)
  - 1: Sub clock

R/W : Readable and Writable

- : Reset value
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0 bit1</td>
<td>FP1, FP0: Frame cycle selection bits</td>
</tr>
<tr>
<td></td>
<td>The frame cycle for the LCD display is selected and set from 4 types.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Set up the optimum frame cycle that supports the frame frequency of the LCD panel to be used.</td>
</tr>
<tr>
<td>bit2 bit3</td>
<td>MS1, MS0: Display mode selection bits</td>
</tr>
<tr>
<td></td>
<td>Output waveform duty setting is selected from 3 types.</td>
</tr>
<tr>
<td></td>
<td>• The common pin to be used is decided by supporting the duty output mode that has been set.</td>
</tr>
<tr>
<td></td>
<td>When &quot;00B&quot; is set, the LCD controller/driver suspends the display operation, and the common/segment pin outputs the &quot;L&quot; level.</td>
</tr>
<tr>
<td>bit4</td>
<td>BK: Blanking selection bits</td>
</tr>
<tr>
<td></td>
<td>Show/Hide LCD is selected.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> Segment output will be non-selected waveforms (the waveform not selected as the display condition) and will not be shown.</td>
</tr>
<tr>
<td>bit5</td>
<td>VSEL: LCD driving power control bit</td>
</tr>
<tr>
<td></td>
<td>Whether internal division resistance is to be connected or disconnected is selected.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> Internal division resistance is disconnected.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> Internal division resistance is connected.</td>
</tr>
<tr>
<td></td>
<td>&quot;0&quot; must be set to connect to the external division resistance.</td>
</tr>
<tr>
<td>bit6</td>
<td>LCEN: Operation enabling bit under clock mode</td>
</tr>
<tr>
<td></td>
<td>Enabling or disabling operation of the LCD controller/driver is set when shifting to clock mode.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> LCD is shown even when shifting to clock mode.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> LCD display is suspended when shifting to clock mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The sub-clock needs to be set to the clock for frame frequency generation to operate even under clock mode. (LCR0: CSS=1)</td>
</tr>
<tr>
<td>bit7</td>
<td>CSS: Clock selection bit for frame cycle generating</td>
</tr>
<tr>
<td></td>
<td>The clock to generate frame cycle for the LVD display is selected.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The LCD controller/driver operates on the oscillation clock (HCLK).</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> The LCD controller/driver operates on the sub-clock.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Oscillation of the oscillation clock (HCLK) is stopped under clock mode and sub-clock mode, so this will not be operated under the oscillation clock setting.</td>
</tr>
</tbody>
</table>
20.3.3 LCDC Control Register 1 (LCR1)

The input/output pin is switched to the segment output or general-purpose input/output port for the LCDC control register 1 (LCR1).

### LCDC control register 1 (LCR1)

The LCDC control register 1 (LCR1) is used to switch the input/output pin between the segment output or general-purpose input/output port. The register is 16 bits wide and contains bits that control the switching of specific segments (SEG0 to SEG15) between output and input states.

#### Figure 20.3-3 LCDC control register 1 (LCR1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SEG0 Segment terminal (SEG19 to SEG16) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as general-purpose I/O port if SEG1 is set to &quot;1&quot;, it can only be used as a general-purpose output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG1 Segment terminal (SEG23 to SEG20) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as general-purpose I/O port if SEG0 is set to &quot;1&quot;, it can only be used as a general-purpose output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG2 Segment terminal (SEG27 to SEG24) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as general-purpose I/O port if SEG3 is set to &quot;1&quot;, it can only be used as a general-purpose output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG3 Segment terminal (SEG31 to SEG28) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as general-purpose I/O port if SEG2 is set to &quot;1&quot;, it can only be used as a general-purpose output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG4 Segment terminal (SEG11 to SEG8) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as general-purpose I/O port if SEG5 is set to &quot;1&quot;, it can only be used as a general-purpose output port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG5 Segment terminal (SEG15 to SEG12) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as a segment output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SEG6 Segment terminal (SEG19 to SEG16) switching bit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Functions as a segment output pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reset value**: 00000000B

- **R**: Readable
- **W**: Writable
- **R/W**: Readable and Writable
- **Reserved**: Always set this bit to "0".
### Table 20.3-4 LCDC control register 1 (LCR1) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 SEG0: Segment pins (SEG19 to SEG16) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG1 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit9 SEG1: Segment pins (SEG23 to SEG20) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG0 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit10 SEG2: Segment pins (SEG27 to SEG24) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG3 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit11 SEG3: Segment pins (SEG31 to SEG28) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG2 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit12 Reserved: reserved bit</td>
<td>Always set this bit to &quot;0&quot;.</td>
</tr>
<tr>
<td>bit13 SEG4: Segment pins (SEG11 to SEG8) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG5 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit14 SEG5: Segment pins (SEG15 to SEG12) switching bit</td>
<td><strong>When the bit is set to &quot;0&quot;</strong>: The pin is set as a general-purpose I/O port. If SEG4 is set to &quot;1&quot;, it can only be used as a general-purpose output port. <strong>When the bit is set to &quot;1&quot;</strong>: The segment pin will be set.</td>
</tr>
<tr>
<td>bit15 Reserved: reserved bit</td>
<td>Always set this bit to &quot;0&quot;.</td>
</tr>
</tbody>
</table>
20.3.4 Data Memory for the Display (VRAM)

Data memory for display is the 16-byte data RAM. This will be used for the data setting for displays.

■ Data memory for the LCD

Data memory contents for display are automatically read in synchronization with the common signal timing, and output from the segment output pins (SEG0 to SEG31).

When the bit supporting the common output and segment output is set to "1", selected waveform (liquid crystal element is lit) is output from the segment output pin. When "0" is set, non-selected waveform (liquid crystal element is not lit) is output from the segment output pin.

Accessing data memory for display can read/write at arbitrary timing regardless the LCD controller/driver operation.

The SEG8 to SEG31 pins can be switched for use as general-purpose input/output ports by setting the segment pin switching bit of the LCDC control register 1. (LCR1:SEG5, SEG4, SEG3 to SEG0) If this is used as the SEG output, partial port input is impossible.

The data memory for display that supports the common pin and segment pin set with the general-purpose input/output port can be used as the normal general-purpose memory. Partial port input is impossible when COM0 is used.

Figure 20.3-5 shows support between the data memory for display and the common output pin and segment output pin.

Table 20.3-5 shows duty output mode and relationship between common output and data memory for display.
Figure 20.3-5 Support between the data memory for display and the common output pin and segment output pin

<table>
<thead>
<tr>
<th>bit7</th>
<th>bit6</th>
<th>bit5</th>
<th>bit4</th>
<th>SEG0</th>
<th>SEG1</th>
<th>SEG2</th>
<th>SEG3</th>
<th>SEG4</th>
<th>SEG5</th>
<th>SEG6</th>
<th>SEG7</th>
<th>SEG8</th>
<th>SEG9</th>
<th>SEG10</th>
<th>SEG11</th>
<th>SEG12</th>
<th>SEG13</th>
<th>SEG14</th>
<th>SEG15</th>
<th>SEG16</th>
<th>SEG17</th>
<th>SEG18</th>
<th>SEG19</th>
<th>SEG20</th>
<th>SEG21</th>
<th>SEG22</th>
<th>SEG23</th>
<th>SEG24</th>
<th>SEG25</th>
<th>SEG26</th>
<th>SEG27</th>
<th>SEG28</th>
<th>SEG29</th>
<th>SEG30</th>
<th>SEG31</th>
</tr>
</thead>
</table>

**SEG0 to SEG7 terminals:**
Dedicated terminals for segment output

**SEG8 to SEG31 terminals:**
Used as a general-purpose input/output port.

When using as general purpose I/O port, valid only for port unit (by every 8 lines). When switching by every 4 lines, valid for using as general purpose I/O port.

---

**Note:** Any RAM area not used as data memory for display purposes can be used as the normal general-purpose memory.
### Table 20.3-5  Duty output mode and relationship between common output and data memory for display

<table>
<thead>
<tr>
<th>Duty output mode selection</th>
<th>LCD common output</th>
<th>The display bit of the data memory to be used</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>bit7</td>
</tr>
<tr>
<td>1/2</td>
<td>COM0, COM1</td>
<td>-</td>
</tr>
<tr>
<td>1/3</td>
<td>COM0 to COM2</td>
<td>-</td>
</tr>
<tr>
<td>1/4</td>
<td>COM0 to COM3</td>
<td>○</td>
</tr>
</tbody>
</table>

○: EPROM for use  
-: Unused
20.4 Explanations for the LCD Controller/Driver Operation

The LCD controller/driver performs the control and drive required for the LCD display.

Explanations for the LCD controller/driver operation

Setting as per Figure 20.4-1 is required when display is carried out with the LCD.

![Figure 20.4-1 The LCD controller/driver setup](image)

When the clock for frame cycle generation oscillates after the setup as per Figure 20.4-1 is carried out, the waveform that drives the LCD is output to the common/segment output pins (COM0 to COM3 and SEG0 to SEG31) in accordance with the details that have been set to the data memory for display.

- The LCD is driven by the 2-frame alternating waveform in accordance with the duty settings of the LCDC control register 0 (LCR0:MS1 and MS0).
- The non-selected level waveform is output from the COM2 and COM3 pins when the display mode is on the 1/2 duty setting, and from the COM3 pin when it is on the 1/3 duty setting.
- When display of the LCD is stopped (LCR0:MS1 and MS0 = "00B"), both the common and segment output pins will be "L" level.
- The SEG0 to SEG7 pins of the common/segment output pins will be "L" level output, and the COM0 to COM3 pins and SEG8 to SEG31 pins will be set as pins for the general-purpose input/output ports by a reset.

Switching the clock for the frame cycle generation

The clock for the frame cycle generation can be switched even during display operation of the LCD (LCR0: CSS)

Even if it is switched to a frame cycle that can be displayed by the LCD, the display may flicker due to switching.

In order to prevent a flickering display due to switching, set the blanking selection bit of the LCD control register 0 to "1" (LCR0:BK = 1) to change the state to "Not Show", and then switch the clock for frame cycle generation.
**Drive waveform for the LCD**

Drive the LCD panel with a 2-frame alternating waveform as the elements are degraded if driven by a direct current. Output waveform is selected from the following three types by the display mode selection bits (LCR0:MS1 and MS0) of the LCDC control register 0.

- 1/2 duty output mode (time division number = 2)
- 1/3 duty output mode (time division number = 3)
- 1/4 duty output mode (time division number = 4)
20.4.1 Power Supply Voltage Set Up for the LCD Controller/Driver (LCDC)

Supply the 1/3 bias LCD-driving power supply voltage to the power pins (V0 to V3) to drive the LCD.

- **Power supply for LCD driving**
  LCD drive power supply voltage is set either by the internal division resistor or an external division resistor.
  
  Table 20.4-1 shows the LCD drive voltage.

  **Table 20.4-1 LCD driving voltage setup**

<table>
<thead>
<tr>
<th></th>
<th>V3</th>
<th>V2</th>
<th>V1</th>
<th>V0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/3 bias setting</td>
<td>V_{LCD}</td>
<td>2/3V_{LCD}</td>
<td>1/3V_{LCD}</td>
<td>GND</td>
</tr>
</tbody>
</table>

- V0 to V3: Voltage of the V0 to V3 pins
- V_{LCD}: LCD operating voltage

- **Note:**
  The LCDC operation-enabling transistor will be inactive in the following cases.
  - When stop LCD operation is set (LCR0: MS1, MS0= "00B").
  - When changing to clock mode (LPMCR:TMD = 0) by disabling operation under the clock mode (LCR0:LCEN = 0)
When internal division resistors are used

The internal division resistors are connected to the LCD drive power supply pins (V0 to V3).

When internal division resistors are used, set the LCDC control register 0 LCD drive power supply control bit to "1" (LCR0:VSEL=1) to connect to the internal division resistors.

Figure 20.4-3 shows status when internal division resistors are used.

Brightness level when internal divisions resistors are used

Connect a variable resistor externally (between the LCD drive power supply and V3 pin) and adjust the voltage of the V3 pin. Figure 20.4-4 shows the example of variable resistor connections.

Figure 20.4-4 Brightness level when internal divisions resistors are used
When external division resistors are used

Set the LCD drive voltage by connecting external division resistors to the LCD drive power supply pins (V0 to V3).

To connect to an external division resistor without affecting the internal division resistor, set the LCDC control register 0 LCD drive power supply control bit to "0" (LCR0:VS EL=0) and disconnect the internal division resistor.

Figure 20.4-5 shows external division resistor connection.

Figure 20.4-5 External divisions resistor connection

- V0 pin is connected to Vss (GND) via an internal transistor. When external resistors are used, connect the Vss side of the resistors to the V0 pin to turn off the current when the LCD controller is halted.
- When the duty output mode is set using the LCDC control register 0 (LCR0) display mode selection bit, LCDC operation enable transistor becomes active and current goes to the external division resistor.

Note: The external segment resistance to be connected externally differs depending on the LCD used, so connect the appropriate resistance value.

Brightness level when external division resistors are used

Connect a variable resistor between the LCD drive power supply and V3 pin and adjust the voltage of the V3 pin.
20.4.2 Output Waveform (1/2 Duty Output Mode) During LCD Controller Drive Operation

The LCD display is driven by a multiplexed drive system 2-frame AC waveform. Only the COM0 and COM1 outputs are used for the LCD display at 1/2 duty output mode. COM2 and COM3 outputs are not used.

■ 1/3 bias, 1/2 duty output waveform example

In the LCD liquid crystal elements light when their common output and segment output reaches maximum potential.

When the content of the display data memory is as shown in Table 20.4-2, the output waveform becomes as shown in Figure 20.4-6.

Table 20.4-2 Example of data memory contents for display

<table>
<thead>
<tr>
<th>Segment</th>
<th>Contents of data memory for display</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COM3 output</td>
</tr>
<tr>
<td>SEGn output</td>
<td>-</td>
</tr>
<tr>
<td>SEGn+1 output</td>
<td>-</td>
</tr>
</tbody>
</table>

1: On
0: Off
-: Not used
Figure 20.4-6 1/3 bias, 1/2 duty output waveform example

V3 to V1: Voltage setting to power supply pin for driving

1 frame cycle
Example of LCD panel wiring and display data (1/2 duty drive system)

Figure 20.4-7 LCD panel display data example

- Address
  - \( n \) and \( n+1 \): Indicates correspondence with data memory for display. bit2, 3, 6 and 7 are unused

- Example) Displaying "5"
  - 0: OFF
  - 1: ON

- Example of Display data 0 to 9
  - Display of LCD
  - [Data memory for display]
20.4.3 Output Waveform (1/3 Duty Output Mode) During LCD Controller Drive Operation

The COM0, COM1 and COM2 outputs are used for the LCD display at 1/3 duty output mode. The COM3 output is not used.

■ 1/3 bias, 1/2 duty output waveform example

In the LCD liquid crystal elements light when their common output and segment output reaches maximum potential.

When the content of the display data memory is as shown in Table 20.4-3, the output waveform becomes as shown in Figure 20.4-8.

Table 20.4-3 Example of data memory contents for display

<table>
<thead>
<tr>
<th>Segment</th>
<th>Contents of data memory for display</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COM3 output</td>
</tr>
<tr>
<td>SEGn output</td>
<td>-</td>
</tr>
<tr>
<td>SEGn+1 output</td>
<td>-</td>
</tr>
</tbody>
</table>

1: On
0: Off
- : Not used
Figure 20.4-8 1/3 bias, 1/2 duty output waveform example

- COM0 output
- COM1 output
- COM2 output
- COM3 output
- SEGn output
- SEGn+1 output

1 frame cycle

V3 to V1: Voltage setting to power supply for driving
Example of LCD panel wiring and display data (1/3 duty drive system)

Figure 20.4-9 LCD panel display data example

Example) Displaying "5"

<table>
<thead>
<tr>
<th>Address</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>nH</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n+1H</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

* 0 to 8: Indicates correspondence with data memory for display. bit3, bit7 and 2 are unused.

Example) Displaying "5"

<table>
<thead>
<tr>
<th>Display of LCD</th>
<th>Example of Display data 0 to 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEG0</td>
<td>0 0 1 1 1 0 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>SEG1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>SEG2</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>SEG3</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>SEG4</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>SEG5</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>SEG6</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>SEG7</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>SEG8</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

As 2 digits are indicated by 3 bytes in 1/3 duty operation, two types of data assignment are allowed: from bit 0 of byte 1 and from bit 4 of byte 2.
20.4.4  Output Waveform (1/4 Duty Output Mode) During LCD Controller Drive Operation

The COM0, COM1, COM2 and COM3 outputs are all used for the LCD display at 1/4 duty output mode.

### 1/3 bias, 1/4 duty output waveform example

Liquid crystal elements light when their common output and segment output reaches maximum potential.

When the content of the display data memory is as shown in Table 20.4-4, the output waveform becomes as shown in Figure 20.4-10.

<table>
<thead>
<tr>
<th>Segment</th>
<th>Contents of data memory for display</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COM3 output</td>
</tr>
<tr>
<td>SEGn output</td>
<td>0</td>
</tr>
<tr>
<td>SEGn+1 output</td>
<td>0</td>
</tr>
</tbody>
</table>

1: On  
0: Off  
-: Not used
**Figure 20.4-10  1/3 bias, 1/4 duty output waveform example**

| COM0 output | V3 | V2 | V1 | V0=Vss |
| COM1 output | V3 | V2 | V1 | V0=Vss |
| COM2 output | V3 | V2 | V1 | V0=Vss |
| COM3 output | V3 | V2 | V1 | V0=Vss |
| SEGn output | ON | OFF | ON | ON |
| SEGn+1 output | ON | OFF | ON | ON |

1 frame cycle

V3 to V1: Voltage setting to power supply for driving
Example of LCD panel wiring and display data (1/4 duty drive system)

Figure 20.4-11 LCD panel display data example

- Example) displaying "5"

<table>
<thead>
<tr>
<th>Address</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
<th>SEGn</th>
<th>SEGn+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>nH</td>
<td>bit3</td>
<td>bit2</td>
<td>bit1</td>
<td>bit0</td>
<td>bit7</td>
<td>bit6</td>
</tr>
</tbody>
</table>

* 0 to 7: Indicates correspondence with data memory for display.

Example of display data 0 to 9

<table>
<thead>
<tr>
<th>Display of LCD</th>
<th>Example of Display data 0 to 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 1 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1 1 0 0 0 1 0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>1 1 1 1 0 1 1 0 0</td>
</tr>
<tr>
<td>3</td>
<td>1 1 1 1 1 1 0 0 0</td>
</tr>
<tr>
<td>4</td>
<td>1 1 1 0 1 0 0 1 1</td>
</tr>
<tr>
<td>5</td>
<td>1 1 1 0 1 0 1 0 1</td>
</tr>
<tr>
<td>6</td>
<td>0 1 1 1 1 1 0 1 0</td>
</tr>
<tr>
<td>7</td>
<td>0 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>8</td>
<td>1 1 0 1 1 0 0 1 1</td>
</tr>
<tr>
<td>9</td>
<td>1 1 1 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>
20.5 Program Example for the LCD Controller/Driver

The following is an example of an LCD controller driver program.

Program example for the LCD controller/driver

● Processing specification

Display data is set in the LCD display data memory. "0" to "9" of the LCD panel connected in the Figure 20.4-11 connection example is shown here. Each setting is described below.

- Selects internal division resistor (LCR1:VSEL=1).
- 1/3 bias, 1/4 duty (LCR0:MS1, MS0= "11B").
- COM0-COM3 and SEG-SEG31 pins are used.
- Frame cycle generation clock is set to sub oscillation clock (LCR0:CSS=1).
- Frame frequency is set to 32 Hz (LCR0:FP1, FP0= "11B").

● Coding example

VRAM EQU 000070H ;First LCD display data memory address
LCDCMR EQU 00000BH ;Common pin switch register
LCR0 EQU 00006AH ;LCDC control register 0 (LCR0)
LCR1 EQU 00006BH ;Control register 1 (LCR1)

;----------Main Program-----------------------------------------------
CODE CSEG
START:

; ;Stack pointer (SP)
;Shall be initialized
;
; ;Data bank register (DTB)=00H
;
MOVW RW0,#VRAM ;LCD display data memory address setting
MOVL A,#LCD_TBL ;LCD display data table address setting
MOVL RL1,A
LCDSET:

MOV A,@RL1+00H
BZ LCDEND ;Data end (00H) is output
MOV @RW0,A
INCW RW0
INCL RL1
BRA LCDSET ;Continue

LCDEND:

MOV LCDCMR,#00001111B ;LCDCMR is set and COM0 to COM3 pins are used
MOV LCR1,#01101111B ;LCR1 is set and SEG0 to SEG31 pins are used
MOV LCR0,#10101111B ;LCR0 is set, LCD is displayed

;
LOOP:

; User processing

BRA LOOP
CHAPTER 20  LCD CONTROLLER DRIVER

LCD_TBL:
    DB 11011111B ;"0"
    DB 11001000B ;"1"
    DB 11101110B ;"2"
    DB 11111011B ;"3"
    DB 11101000B ;"4"
    DB 01111101B ;"5"
    DB 01111111B ;"6"
    DB 11011001B ;"7"
    DB 11111111B ;"8"
    DB 11111011B ;"9"
    DB 00000000B ;END

; CODE ENDS

;-------------------Vector Settings-------------------------------------------
VECT CSEG ABS=OFFH
ORG OFFDCH ;Reset vector setting
DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START
This chapter explains the address match detection function and its operation.

21.1 Overview of Address Match Detection Function
21.2 Block Diagram of Address Match Detection Function
21.3 Program Address Detection Control Status Register (PACSR)
21.4 Program Address Detect Register (PADR)
21.5 Explanation of Operation of Address Match Detection Function
21.6 Program Example of Address Match Detection Function
21.1 Overview of Address Match Detection Function

- The address match detection function replaces the command executed by the CPU with the INT9 command when the address value that executes the program matches the set value of the program address detection register, and the interrupt program is executed.
- The address match detection function can be used to correct (patch application) the program mistake.

### Overview of Address Match Detection Function

- When the value set to the program address detection register and value retained for the address latch (address value to which the next program will be executed) match, the command executed by the CPU is replaced with the INT9 command, and the interrupt program is executed using the address match detection function.
- The program address detection register has 2 channels (PADR0 and PADR1), and the generation of interrupt requests on condition of match detection between the program address detection register value and address latch value can be enabled or disabled per PADR0 and PADR1 register.
21.2 Block Diagram of Address Match Detection Function

The address match detection module consists of the following blocks:
- Address latch
- Address detection control register (PACSR)
- Detect address setting registers

Figure 21.2-1  Block Diagram of Address Match Detection Function

- Address latch
  The address value that was output to the internal data bus is retained.

- Program address detection control status register (PACSR)
  Enabling interruption request is set for address match detection.

- Program address detect register (PADR0, PADR1)
  The address that suspends execution of the program and executes the correction program is set.
21.3 Program Address Detection Control Status Register (PACSR)

The program address detection control status register (PACSR) controls interrupts within the address match detection function.

Figure 21.3-1 Program address detection control status register (PACSR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Address match detection enable bit 0</td>
<td>0: PADR0 disabled</td>
</tr>
<tr>
<td>1</td>
<td>Address match detection enable bit 1</td>
<td>0: PADR1 disabled</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
<td>0: Always set to 0.</td>
</tr>
</tbody>
</table>

R/W: Readable and Writable

Reset value: 00000000
### Table 21.3-1 Program address detection control status register (PACSR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0</td>
<td>Reserved: reserved bit</td>
</tr>
</tbody>
</table>
| bit1     | ADE0: Address match detection enable bit 0 | The value set up in the program address detection register "0" enables or disables detection of matched addresses.  
**When the bit is set to "0":** Detection is disabled.  
**When the bit is set to "1":** Detection is enabled. When detecting, INT9 instruction is executed. |
| bit2     | Reserved: reserved bit | Always set this bit to "0". |
| bit3     | ADE1: Address match detection enable bit 1 | The value set up in the program address detection register "1" enables or disables detection of matched addresses.  
**When the bit is set to "0":** Detection is disabled.  
**When the bit is set to "1":** Detection is enabled. When detecting, INT9 instruction is executed. |
| bit4 to bit7 | Reserved: reserved bit | Always set this bit to "0". |
21.4 Program Address Detect Register (PADR)

The program address detection register (PADR) sets the address to be compared.

Program address detect register (PADR)

Figure 21.4-1 Program address detect register

There are 2 channels for the program address detection register, and 24-bit addresses can be set per register.

- The head address (1st byte) of the command code to be replaced with the INT9 command needs to be set for the program address detection register.

Note: When an address other than the 1st byte is set for the program address detection register, replacement with the INT9 command is not carried out, and the correction program is not executed. Command code of the address set up for other than the 1st byte is replaced with "01H" (INT9 command code), and causes to generate erroneous operation.
Address value examples that need to be set up in the program address detection register are shown in Figure 21.4-2.

**Figure 21.4-2 Setting example of Starting Address of Instruction Code to be Replaced by INT9**

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF001C</td>
<td>A8_00 00</td>
<td>MOVW RW0,#0000</td>
</tr>
<tr>
<td>FF001F</td>
<td>4A_00 00</td>
<td>MOVW A,#0000</td>
</tr>
<tr>
<td>FF0022</td>
<td>4A_80 08</td>
<td>MOVW A,#0880</td>
</tr>
</tbody>
</table>

Upper : FFH, Middle : 00H, Lower : 1FH

**Note:** When address is set to the program address detection register, set the address match detection to disable. (PACSR: ADE=0)
If setting is performed without being disabled, erroneous operation may result if an address match is detected while writing addresses.
21.5 Explanation of Operation of Address Match Detection Function

Under the address match detection function, the command executed by the CPU is executed by replacement with the INT9 command when the address value that executes the program matches the value set in the program address detection register. A correction program can be executed using an INT9 interrupt.

Explanation of Operation of Address Match Detection Function

Settings for the address match detection function and operation when address match is detected are shown in Figure 21.5-1.

- Address match detection is disabled. (PACSR: ADE0=0)
- "FF001FH" is set to the program address detection register (PADR0). (PADR0: Upper = "FFH", PADR0: Middle = "00H", PADR0: Lower = "1FH")
- Address match detection is enabled. (PACSR: ADE0=1)

Figure 21.5-1 Operation of Address Match Detection Function

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF001C</td>
<td>A8 00 00</td>
<td>MOVW RW0,#0000</td>
</tr>
<tr>
<td>FF001F</td>
<td>4A 00 00</td>
<td>MOVW A,#0000</td>
</tr>
<tr>
<td>FF0022</td>
<td>4A 80 08</td>
<td>MOVW A,#0880</td>
</tr>
</tbody>
</table>

- The command code of the matched address is replaced with the INT9 command (code "01H").
- An INT9 interrupt is generated by executing the INT9 command, and the interrupt program is executed.
21.5.1 Example of Using Address Match Detection Function

An example of the address match detection function being used for patch processing of the program correction is shown.

■ System Configuration and $E^2$PROM Memory Map configuration

- System configuration

  System configuration example is shown in Figure 21.5-2.

  Figure 21.5-2 System configuration

![System configuration diagram](image)
E\textsuperscript{2}PROM Memory Map

Figure 21.5-3 shows the allocation of the patch program and data at storing the patch program in E\textsuperscript{2}PROM.

**Figure 21.5-3 Allocation of E\textsuperscript{2}PROM Patch Program and Data**

- **Program Byte Number** (8bit)
  - Store the byte number of the correction program (source).
  - At "00H", display "No correction program".

- **Program Address** (24bit)
  - Store the address of the program miss point
  - The address is set to the program address detection register.

- **Correction Program (Main body)**
  - Store the executing program with the INT9 interrupt of the address match detection.

- **Correction Program Starting Address**
  - Correction program 0 is allocated from any predetermined address.
  - Correction program 1 is allocated from the address indicating <starting address of correction program 0 + total byte count of correction program 0>. 

---

- **E\textsuperscript{2}PROM Address**
  - 0000H: Correction program byte number
  - 0001H: detection address 0 (lower)
  - 0002H: detection address 0 (middle)
  - 0003H: detection address 0 (upper)
  - 0004H: correction program byte number
  - 0005H: detection address 1 (lower)
  - 0006H: detection address 1 (middle)
  - 0007H: detection address 1 (upper)
  - 0010H: Correction program 0 (main body)
  - 0020H: Correction program 1 (main body)
Setting and Operating State

● The Initial State
  - E²PROM data are all cleared to "00H".
  - Set up a status in which no correction program supporting program errors exists.

● Occurrence of program error
  - The E²PROM correction program and the correction program information in accordance with the data layout method are sent to the MCU (MB90520A) from outside using the UART (connector connection).
  - The MCU (MB90520A) side receives the correction program information, and stores it in E²PROM.

● Reset sequence
  - The MCU (MB90520A) side reads the byte number of the E²PROM correction program and checks whether the correction program exists or not after a reset.
  - When the byte number of the correction program is other than "00H", program addresses 0 to 2 are read, and set to the program address detection register. The correction program (main body) is read in accordance with the byte number indicating the program length, and written to the RAM of the MCU.
  - The correction program (main body) is allocated to the address to be executed by the INT9 interrupt process by the detecting address match.
  - Address match detection is enabled (PACSR: ADE=1)

● INT9 interrupt processing
  - Which address detection factor caused output of the interrupt request through interrupt processing is determined in accordance with the value of the program counter (PC) that has been saved in the stack, and branched to the program on which an interrupt request was output.

  When branching to the program, information stacked by interruption is disposed (returned to the stack pointer before generating interruption), and returned to the normal program after the correction program is executed.

Figure 21.5-4 shows operation of Address Match Detection Function.
Figure 21.5-4 Operation of Address Match Detection Function

1. Setting the program address detecting of reset sequence, executing normal program
2. Branch to the patch program that is expanded to RAM by INT9 interruption from address match detection.
3. Executing the patch program by branching of INT9 operation.
4. Executing the normal program that is branched by the patch program.
CHAPTER 21 ADDRESS MATCH DETECTING FUNCTION

### Processing flow of the program patching

Processing flow of the program patching is shown in Figure 21.5-5.

**Figure 21.5-5 Processing flow of the program patching**

- **Reset**
  - E\(^2\)PROM : 0000H
  - Read

- **YES**
  - E\(^2\)PROM : 0000H
  - Read Program address
  - E\(^2\)PROM : 0001H to 0003H
  - MCU : PADR0 setting

- **NO**
  - Program address = PADR0

- **YES**
  - Program address = PADR0
  - Normal program execution

- **NO**
  - Program address = PADR0
  - INT9

**Figure 21.5-5 Processing flow of the program patching**

- **ROM**
  - MB90522B
  - I/O area
  - Register/RAM area
  - Correction program
  - RAM area
  - Stack area
  - Program address detection register
  - FF0000H
  - Abnormal program
  - FF8000H
  - FF8050H
  - FFFFFH

- **RAM**
  - E\(^2\)PROM
  - Program address upper : FFH
  - Program address middle : 80H
  - Program address lower : 00H
  - Correction program
  - 0000H
  - 0001H
  - 0002H
  - 0003H
  - 0010H
  - 0090H
  - FFFFH

- **INT9**
  - Branching to Correction program
  - JMP 000400H

- **Correction program execution**
  - Correction program completion
  - JMP FF8050H
CHAPTER 21  ADDRESS MATCH DETECTING FUNCTION

21.6  Program Example of Address Match Detection Function

This section gives a program example for the address match detection function.

■ Program Example of Address Match Detection Function

● Processing specification

When the addresses set to PADR0 and the program address match, the INT9 command is forcibly executed.

● Coding example

```assembly
PACSR EQU 00009EH ;Program address detection control
;Status Register
PADRL EQU 001FF0H ;Program address detection register 0 lower
PADRM EQU 001FF1H ;Program address detection register 0 middle
PADRH EQU 001FF2H ;Program address detection register 0 upper

;---------Main Program-----------------------------------------------
CODE CSEG
START:
    ;Stack pointer (SP)
    ;Shall be initialized
    MOV PADRL,#00H ;Program address detection register 0 lower setting
    MOV PADRM,#00H ;Program address detection register 0 middle setting
    MOV PADRH,#00H ;Program address detection register 0 upper setting
    MOV I:PACSR,#00000010B;Enables address match detection
    •
    User processing
    •
    LOOP:
    •
    User processing
    •
    BRA LOOP

;---------Interrupt Program-------------------------------------------
WARI:
    •
    User processing
    •
    RETI ;Returning from interrupt processing
CODE ENDS

;---------Vector Settings---------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFD8H ;Vector set up to the interruption number #09 (09H)
    DSL WARI
ORG 00FFDCH ;Reset vector setting
    DSL START
DB 00H ;Single-chip mode
VECT ENDS
END START
```
CHAPTER 22
ROM MIRROR FUNCTION
SELECTION MODULE

This chapter describes the functions and operations of the ROM mirroring function select module.

22.1 Overview of ROM Mirroring Function Select Module
22.2 ROM Mirroring Function Select Register (ROMM)
22.1 Overview of ROM Mirroring Function Select Module

Set up so that data within the ROM that has been allocated to the FF bank can be read by accessing the 00 bank for the ROM mirror function selection module.

Block Diagram of ROM Mirroring Function Select Module

Access to FF Bank by ROM Mirroring Function

The physical relationship on the memory when the ROM data of the FF bank is read by accessing the 00 bank using the ROM mirror function is shown in Figure 22.1-2.
Memory Space when ROM Mirroring Function Enabled/Disabled

Figure 22.1-3 shows the availability of access to memory space when the ROM mirroring function is enabled or disabled.

**Figure 22.1-3 Memory Space when ROM Mirroring Function Enabled/Disabled**

<table>
<thead>
<tr>
<th>Addresses #1</th>
<th>Addresses #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>001000H</td>
<td>001000H</td>
</tr>
<tr>
<td>001100H</td>
<td>001100H</td>
</tr>
<tr>
<td>001FF0H</td>
<td>000000H</td>
</tr>
<tr>
<td>001FFFH</td>
<td>000000H</td>
</tr>
<tr>
<td>004000H</td>
<td>000000H</td>
</tr>
<tr>
<td>010000H</td>
<td>000000H</td>
</tr>
</tbody>
</table>

When the ROM mirroring function is enabled

When the ROM mirroring function is disabled

<table>
<thead>
<tr>
<th>Product</th>
<th>MB90522A</th>
<th>MB90522B</th>
<th>MB90523A</th>
<th>MB90523B</th>
<th>MB90F523B</th>
<th>MB90V520A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addresses 1</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001100H</td>
<td>001900H</td>
</tr>
<tr>
<td>Addresses 2</td>
<td>FF0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>FE0000H</td>
<td>—</td>
</tr>
</tbody>
</table>

Address #1 and #2 are different from depending on the kind of products.

* : Address match detection function register area

*: Access prohibance
22.2 ROM Mirroring Function Select Register (ROMM)

The ROM mirroring function select register enables or disables the ROM mirroring function. When valid is set, the ROM data of the FF bank can be read by reading the 00 bank.

ROM Mirroring Function Select Register (ROMM)

- Figure 22.2-1 ROM Mirroring Function Select Register (ROMM)

- Table 22.2-1 Functions of ROM Mirroring Function Select Register (ROMM)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit8 MI</td>
<td>This bit enables or disables the ROM mirroring function.</td>
</tr>
<tr>
<td></td>
<td><strong>When set to &quot;0&quot;:</strong> Disables ROM mirroring function</td>
</tr>
<tr>
<td></td>
<td><strong>When set to &quot;1&quot;:</strong> Enables ROM mirroring function</td>
</tr>
<tr>
<td></td>
<td>When setup is carried out effectively, the contents of the ROM addresses</td>
</tr>
<tr>
<td></td>
<td>&quot;FF4000H&quot; to &quot;FFFFFFFH&quot; can be read by accessing the addresses &quot;004000H&quot;</td>
</tr>
<tr>
<td></td>
<td>to &quot;00FFFFFFH&quot;.</td>
</tr>
<tr>
<td>bit9 to bit15</td>
<td>Unused bits</td>
</tr>
<tr>
<td></td>
<td><strong>Read:</strong> The value is undefined.</td>
</tr>
<tr>
<td></td>
<td><strong>Write:</strong> No effect.</td>
</tr>
</tbody>
</table>

Note: While the ROM area at addresses 004000H to 00FFFFFFH is being used, access to the ROM mirroring function select register (ROMM) is prohibited.
 CHAPTER 23
CLOCK MONITOR FUNCTION

The clock monitor function and operations are explained in this chapter.

23.1 Clock Monitor Function Outline
23.2 Clock Monitor Function Block Diagram
23.3 Clock Monitor Function Configuration
23.4 Program Examples of the Clock Monitor Function
23.1 Clock Monitor Function Outline

The clock monitor function outputs the division clock of the machine clock for the monitor from the clock monitor pin (CKOT).

Clock monitor function outline

- When the output enabling bit of the clock output enabling register is set to "1" (CLKR:CKEN = 1), the clock is output from the clock monitor pin (CKOT).
- The frequency of the clock to be output is set by the output frequency selection bit of the clock output enabling register. (CLKR: FRQ2 to FRQ0)

The frequency of the clock to be output using the clock monitor function is shown in Table 23.1-1.

<table>
<thead>
<tr>
<th>FRQ2 to 0 bit</th>
<th>Clock output frequency (Hz)</th>
<th>φ=16MHz</th>
<th></th>
<th>φ=8MHz</th>
<th></th>
<th>φ=4MHz</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Cycle</td>
<td>Frequency</td>
<td>Cycle</td>
<td>Frequency</td>
<td>Cycle</td>
<td>Frequency</td>
</tr>
<tr>
<td>000B</td>
<td>φ/2¹</td>
<td>125ns</td>
<td>8MHz</td>
<td>250ns</td>
<td>4MHz</td>
<td>500ns</td>
<td>2MHz</td>
</tr>
<tr>
<td>001B</td>
<td>φ/2²</td>
<td>250ns</td>
<td>4MHz</td>
<td>500ns</td>
<td>2MHz</td>
<td>1.0µs</td>
<td>1MHz</td>
</tr>
<tr>
<td>010B</td>
<td>φ/2³</td>
<td>500ns</td>
<td>2MHz</td>
<td>1.0µs</td>
<td>1MHz</td>
<td>2.0µs</td>
<td>500kHz</td>
</tr>
<tr>
<td>011B</td>
<td>φ/2⁴</td>
<td>1.0µs</td>
<td>1MHz</td>
<td>2.0µs</td>
<td>500kHz</td>
<td>4.0µs</td>
<td>250kHz</td>
</tr>
<tr>
<td>100B</td>
<td>φ/2⁵</td>
<td>2.0µs</td>
<td>500kHz</td>
<td>4.0µs</td>
<td>250kHz</td>
<td>8.0µs</td>
<td>125kHz</td>
</tr>
<tr>
<td>101B</td>
<td>φ/2⁶</td>
<td>4.0µs</td>
<td>250kHz</td>
<td>8.0µs</td>
<td>125kHz</td>
<td>16.0µs</td>
<td>62.5kHz</td>
</tr>
<tr>
<td>110B</td>
<td>φ/2⁷</td>
<td>8.0µs</td>
<td>125kHz</td>
<td>16.0µs</td>
<td>62.5kHz</td>
<td>32.0µs</td>
<td>31.25kHz</td>
</tr>
<tr>
<td>111B</td>
<td>φ/2⁸</td>
<td>16.0µs</td>
<td>62.5kHz</td>
<td>32.0µs</td>
<td>31.25kHz</td>
<td>64.0µs</td>
<td>15.625kHz</td>
</tr>
</tbody>
</table>

φ: Machine clock frequency
23.2 Clock Monitor Function Block Diagram

The module of the clock monitor function is made up of the following blocks.
- Prescaler
- Count clock selector
- Clock output enable register

Block Diagram of clock monitor function

Figure 23.2-1 Block Diagram of clock monitor function

- Prescaler
  The machine clock $\phi$ is divided and supplied to the count clock selector.

- Count clock selector
  The clock to be output is selected from 8 division clock types.

- Clock output enable register
  Clock output is enabled and output frequency is selected.

Details of the pin
Details of the pin to be used for the clock monitor function are described as follows.

CKOT pin: P31/CKOT
23.3 Clock Monitor Function Configuration

Pin and register details of the clock monitor function are described.

■ Pins for the clock monitor function

Clock monitor pin (CKOT) is used together with the general-purpose input/output port.

Settings to be used for the pin function and clock monitor function are shown in Table 23.3-1.

Table 23.3-1 Pins for the clock monitor function

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Function</th>
<th>Setting required to use the clock monitor function</th>
</tr>
</thead>
<tbody>
<tr>
<td>P31/CKOT</td>
<td>General-purpose I/O port, clock output</td>
<td>Clock output is enabled. (CLKR: CKEN=1)</td>
</tr>
</tbody>
</table>

■ Block Diagram of Pins

Reference: For the block diagram of the pins, see "CHAPTER 4 I/O PORT".
23.3.1 Clock Output Enabling Register (CLKR)

Clock output enabling register (CLKR) enables clock output and selects output frequency.

Clock output enabling register (CLKR)

![Figure 23.3-1 Clock output enabling register (CLKR)](image)

Table 23.3-2 Clock output enable register (CLKR) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit0 to bit2</td>
<td>FRQ0, FRQ1, FRQ2: Output frequency selection bits</td>
</tr>
<tr>
<td></td>
<td>Frequency of the clock to be output is set.</td>
</tr>
<tr>
<td></td>
<td>The division rate for the machine clock can be selected and set from 8 types.</td>
</tr>
<tr>
<td>bit3</td>
<td>CKEN: Output enable bit</td>
</tr>
<tr>
<td></td>
<td>Output of the clock monitor pin (CKOT) is enabled.</td>
</tr>
<tr>
<td></td>
<td>When the bit is set to &quot;1&quot;: Set to the clock monitor pin.</td>
</tr>
<tr>
<td></td>
<td>When the bit is set to &quot;0&quot;: Set to the general purpose I/O port.</td>
</tr>
<tr>
<td>bit4 to bit7</td>
<td>Unused bits</td>
</tr>
<tr>
<td></td>
<td>Read: The value is undefined.</td>
</tr>
<tr>
<td></td>
<td>Write: No effect</td>
</tr>
</tbody>
</table>
CHAPTER 23  CLOCK MONITOR FUNCTION

23.4  Program Examples of the Clock Monitor Function

The clock monitor program example is shown.

### Example of a clock monitor program

**Processing specification**
- Clock with 500 kHz frequency is output from the CKOT pin at the 16 MHz machine clock \( \phi \).
- The FRQ2 to 0 bits will be "100B" (clock: \( \phi/2^5 \)).

**Coding example**

```
CLKR EQU 00003EH ;Clock output control register
;
;----------Main Program-----------------------------------------------
CODE CSEG
START:
; ;Stack pointer (SP)
; :Shall be initialized
    MOV I:CLKR,#00001100B ;Enable clock output, \( \phi/2^5 \) set up
;
    User processing
    ;
CODE ENDS
END START
;----------Vector Settings---------------------------------------------
VECT CSEG ABS=0FFH
ORG 00FFDCH ;Reset vector setting
DSL START
DB 00H ;Set to single-chip mode
```
CHAPTER 24
1M-BIT FLASH MEMORY

In this chapter, the function and operation of the 1M-bit flash memory are explained.

24.1 Overview of 1M-bit Flash Memory
24.2 Sector Configuration for Flash Memory
24.3 Flash Memory Control Status Register (FMCS)
24.4 Flash Memory Auto Algorithm Start-up Method
24.5 Check the Execution State of Automatic Algorithm
24.6 Details of Programming/Erasing Flash Memory
24.7 Notes on Using Flash Memory
24.8 Program Example of 1 Mbit Flash Memory
24.1 **Overview of 1M-bit Flash Memory**

The following three methods are available for writing data to and deleting data from the flash memory.

1. Using parallel writer
2. Using serial dedicated writer
3. Write/delete operation by program execution

This chapter describes the above "3. Write/delete operation by program execution".

---

**Overview of 1M-bit Flash Memory**

1 Mbit flash memory is placed in the FE_H to FF_H banks on the CPU memory map. The function of the flash memory I/F circuit provides read access and program access from the CPU to flash memory.

Write/delete of the flash memory can be performed by the command from the CPU via the flash memory interface circuit, so re-writing while it is mounted is possible, and data can be updated efficiently.

**Features of 1Mbit Flash Memory**

- 128K words × 8 bits/64K words × 16 bits (16K + 8K + 8K + 32K + 64K) sector configuration
- Automatic program algorithm (same as the Embedded Algorithm™:MBM29F400TA)
- Erase pause/restart function
- Detection of completion of writing/erasing by data polling and toggle bit.
- Detects completion of writing/erasing by CPU interrupts
- Compatibility with the JEDEC standard-type command
- Sector erase function (any combination of sectors)
- The write/delete number is 10,000 (Min).

*Embedded Algorithm™* is a registered trademark of Advanced Micro Devices, Inc.

---

**Note:**

The function for reading the manufacture code and device code is unprovided. These codes cannot be accessed by any command.

---

**Programming and Erasing Flash Memory**

- The flash memory cannot simultaneously perform write/delete and read.
- When write/delete is performed on data in the flash memory, copy the program on the flash memory to the RAM and writing to the flash memory can be performed by executing the copied program to the RAM.
### 24.2 Sector Configuration for Flash Memory

Sector configuration of Flash Memory is shown.

#### Sector Configuration of 1Mbit Flash memory

"Figure 24.2-1 Sector Configuration of 1Mbit Flash Memory" shows sector configuration of 1Mbit Flash memory. The upper and lower addresses of each sector are given in the figure. For access from the CPU, SA0 is allocated in bank register FE and SA1 to SA4 are allocated in bank register FF.

#### Figure 24.2-1 Sector Configuration of 1Mbit Flash Memory

<table>
<thead>
<tr>
<th>Flash memory</th>
<th>CPU address</th>
<th>Writer address*</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA0 (64K byte)</td>
<td>FE0000H</td>
<td>60000H</td>
</tr>
<tr>
<td></td>
<td>FFFFFFH</td>
<td>6FFFFFH</td>
</tr>
<tr>
<td>SA1 (32K byte)</td>
<td>FF0000H</td>
<td>70000H</td>
</tr>
<tr>
<td></td>
<td>FF7FFFH</td>
<td>77FFFFH</td>
</tr>
<tr>
<td>SA2 (8K byte)</td>
<td>FF8000H</td>
<td>78000H</td>
</tr>
<tr>
<td></td>
<td>FF9FFFH</td>
<td>79FFFFH</td>
</tr>
<tr>
<td>SA3 (8K byte)</td>
<td>FFA000H</td>
<td>7A000H</td>
</tr>
<tr>
<td></td>
<td>FFBFFFH</td>
<td>7BFFFFH</td>
</tr>
<tr>
<td>SA4 (16K byte)</td>
<td>FCC000H</td>
<td>7C000H</td>
</tr>
<tr>
<td></td>
<td>FFFFFFH</td>
<td>7FFFFFH</td>
</tr>
</tbody>
</table>

*: The writer address is equivalent to the CPU address when writing the data to the flash memory using the parallel writer. If the write/delete operation is executed using the general-purpose writer, the write/delete operation is executed using this address.
### Flash Memory Control Status Register (FMCS)

Function of the flash memory control status register (FMCS) is shown in "Figure 24.3-1 Flash memory control status register (FMCS)".

#### Flash memory control status register (FMCS)

**Figure 24.3-1 Flash memory control status register (FMCS)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reset value</td>
<td>000X0000B</td>
</tr>
<tr>
<td>R/W</td>
<td>Read/write</td>
<td>Readable</td>
</tr>
<tr>
<td>R</td>
<td>Read only</td>
<td>Readable</td>
</tr>
<tr>
<td>W</td>
<td>Write only</td>
<td>Writable</td>
</tr>
<tr>
<td>X</td>
<td>Undefined</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Low power consumption mode selection bit</td>
<td>Low power consumption mode Condition of internal operating frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 0 Normal power consumption mode Operating enabled at 16MHz or less</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 1 Low power consumption mode Operating enabled at 4MHz or less</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 Operating enabled at 8MHz or less</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1 Operating enabled at 10MHz or less</td>
</tr>
<tr>
<td>R/W</td>
<td>Flash memory write/erase status bit</td>
<td>Executing of write/erase (disable of write/erase the next data)</td>
</tr>
<tr>
<td>R</td>
<td>Disable of write/erase for Flash memory area</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>Enable of write/erase for Flash memory area</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>Flash memory operating flag bit</td>
<td>Read</td>
</tr>
<tr>
<td></td>
<td>Executing of write/erase</td>
<td>Clear this RDYINT bit</td>
</tr>
<tr>
<td></td>
<td>Complete of write/erase</td>
<td>No effect</td>
</tr>
<tr>
<td>R/W</td>
<td>Write/erase interrupt enable bit for Flash memory</td>
<td>Interrupt disabled by complete of write/erase</td>
</tr>
<tr>
<td>R</td>
<td>Interrupt enabled by complete of write/erase</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>Interrupt enabled by complete of write/erase</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>Reset value</td>
<td></td>
</tr>
</tbody>
</table>

Be sure to set bit 0 to '0'.
Table 24.3-1 Flash memory control status register (FMCS) function

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit2, bit0</td>
<td>LPM1, LPM0: Low electricity consumption mode</td>
</tr>
<tr>
<td></td>
<td>Power consumption for main unit of flash memory is controlled.</td>
</tr>
<tr>
<td></td>
<td>• Access time to the flash memory differs significantly depending on the operating frequency, so set up with reference to conditions for the CPU internal operation frequency.</td>
</tr>
<tr>
<td>bit1, bit3</td>
<td>Reserved: reserved bit</td>
</tr>
<tr>
<td></td>
<td>Always set this bit to &quot;0&quot;.</td>
</tr>
<tr>
<td>bit4</td>
<td>RDY: Flash memory programming/erasing status bit</td>
</tr>
<tr>
<td></td>
<td>This bit shows the programming/erasing status of flash memory.</td>
</tr>
<tr>
<td></td>
<td>• If the RDY bit is &quot;0&quot;, programming/erasing flash memory is disabled.</td>
</tr>
<tr>
<td></td>
<td>• The commands, such as the read/reset command and sector erasing pause, can be accepted even if the RDY bit is &quot;0&quot;. The RDY bit is set to &quot;1&quot; when programming/erasing is completed.</td>
</tr>
<tr>
<td>bit5</td>
<td>WE: Flash memory programming/erasing enable bit</td>
</tr>
<tr>
<td></td>
<td>This bit enables or disables the programming/erasing of flash memory.</td>
</tr>
<tr>
<td></td>
<td>The WE bit should be set before starting the command to program/erase flash memory.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> No program/erase signal is generated even if the command to program/erase the FE and FF bank is input.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> Programming/erasing to flash memory is enabled after inputting program/erase command to the FE and FF bank.</td>
</tr>
<tr>
<td></td>
<td>• When not performing programming/erasing, the WE bit should be set to &quot;0&quot; so as not to accidentally program or erase flash memory.</td>
</tr>
<tr>
<td>bit6</td>
<td>RDYINT: Flash memory operation flag bit</td>
</tr>
<tr>
<td></td>
<td>This bit shows the operating state of flash memory.</td>
</tr>
<tr>
<td></td>
<td>If programming/erasing of flash memory is terminated, the RDYINT bit is set to &quot;1&quot; in timing of termination of the automatic flash memory algorithm.</td>
</tr>
<tr>
<td></td>
<td>• If the RDYINT bit is set to &quot;1&quot; when an interrupt as programming/erasing flash memory is terminated is enabled (FMCS: INTE = 1), an interrupt is requested.</td>
</tr>
<tr>
<td></td>
<td>• If the RDYINT bit is &quot;0&quot;, programming/erasing to flash memory is disabled.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;0&quot;:</strong> The bit is cleared.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> No effect.</td>
</tr>
<tr>
<td></td>
<td>If the read-modify-write (RMW) instructions are used, &quot;1&quot; is always read.</td>
</tr>
<tr>
<td>bit7</td>
<td>INTE: Programming and Erasing interrupt enable bit of Flash Memory</td>
</tr>
<tr>
<td></td>
<td>This bit enables or disables an interrupt as programming/erasing flash memory is terminated.</td>
</tr>
<tr>
<td></td>
<td><strong>When the bit is set to &quot;1&quot;:</strong> If the flash memory operation flag bit is set to &quot;1&quot; (FMCS: RDYINT = 1), an interrupt is requested.</td>
</tr>
</tbody>
</table>

Reference: Refer to "24.4 Flash Memory Auto Algorithm Start-up Method" for command sequences.
The flash memory operation flag bit (RDYINT) and flash memory programming/erasing status bit (RDY) do not change simultaneously. A program should be created so that either RDYINT bit or RDY bit can identify the termination of programming/erasing.
24.4 Flash Memory Auto Algorithm Start-up Method

There are four commands for starting the automatic algorithm of flash memory: read/reset, write, chip erase, and sector erase. The sector erase command controls suspension and resumption of sector erase.

■ Command Sequence Table

"Table 24.4-1 Command Sequence Table" shows command sequence table for using at writing/erasing of Flash memory. All data to be written to the command register is of byte length, but writing is performed by word access. When writing with word access, upper data bytes are ignored.

Table 24.4-1 Command Sequence Table

<table>
<thead>
<tr>
<th>Command sequence</th>
<th>Bus write access</th>
<th>1st bus write cycle</th>
<th>2nd bus write cycle</th>
<th>3rd bus write cycle</th>
<th>4th bus write cycle</th>
<th>5th bus write cycle</th>
<th>6th bus write cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read/ Reset*</td>
<td>1</td>
<td>FxxXXX</td>
<td>XXXF0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read/ Reset*</td>
<td>4</td>
<td>FxAAAA</td>
<td>XXAA</td>
<td>Fx5554</td>
<td>XX55</td>
<td>FxAAAA</td>
<td>XXXF0</td>
</tr>
<tr>
<td>Write program</td>
<td>4</td>
<td>FxAAAA</td>
<td>XXAA</td>
<td>Fx5554</td>
<td>XX55</td>
<td>FxAAAA</td>
<td>XXX0</td>
</tr>
<tr>
<td>Chip erase</td>
<td>6</td>
<td>FxAAAA</td>
<td>XXAA</td>
<td>Fx5554</td>
<td>XX55</td>
<td>FxAAAA</td>
<td>XX80</td>
</tr>
<tr>
<td>Sector erase</td>
<td>6</td>
<td>FxAAAA</td>
<td>XXAA</td>
<td>Fx5554</td>
<td>XX55</td>
<td>FxAAAA</td>
<td>XX80</td>
</tr>
</tbody>
</table>

Note:

- Fx which is the address in the table means FF and FE. When controlling addresses, set the value of bank for access.
- Addresses in the table are the values in the CPU memory map. All addresses and data are hexadecimal values. However, "x" is any value.
- PA: Program address. Only even addresses can be specified.
- SA: Sector address
- RD: Read data
- PD: Program data. Only word data can be specified.

*: 2 types of read/reset commands enable the flash memory to be reset to reading mode.
Auto-select in "Table 24.4-1 Command Sequence Table" is the command to indicate the state of sector protection. In practice, the address needs to be set together with the command in "Table 24.4-1 Command Sequence Table" as follows.

**Table 24.4-2  Address Setting for Auto Select**

<table>
<thead>
<tr>
<th>Sector protection</th>
<th>AQ13 to AQ16</th>
<th>AQ7</th>
<th>AQ2</th>
<th>AQ1</th>
<th>AQ0</th>
<th>DQ7 to DQ0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sector address</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td></td>
<td>CODE*</td>
</tr>
</tbody>
</table>

*: The output at the protected sector address is 01H.
   The output at the unprotected sector address is 00H.
24.5 Check the Execution State of Automatic Algorithm

As the flash memory performs the write/delete flow using the auto algorithm, the operating status inside the flash memory can be checked using the hardware sequence flag.

■ Hardware sequence flag

● Overview of hardware sequence flag

The hardware sequence flag consists of the following 5-bit outputs:
- Data polling flag (DQ7)
- Toggle bit flag (DQ6)
- Timing limit over flag (DQ5)
- Sector Deletion Timer Flag (DQ3)
- Toggle bit 2 flag (DQ2)

These flags can be used to check whether programming, chip and sector erasing, and erase code writing are enabled.

The hardware sequence flags can be referred by setting command sequences and performing read access to the address of a target sector in flash memory. "Table 24.5-1 Bit Allocation of Hardware Sequence Flags" gives the bit allocation of the hardware sequence flags.

<table>
<thead>
<tr>
<th>Bit No.</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware sequence flag</td>
<td>DQ7</td>
<td>DQ6</td>
<td>DQ5</td>
<td>-</td>
<td>DQ3</td>
<td>DQ2</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- To identify whether automatic programming/chip and sector erasing is in execution or terminated, check the hardware sequence flag or the flash memory programming/erasing status bit in the flash memory control status register (FMCS: RDY). After programming/erasing is terminated, the state returns to the read/reset state.
- To create a programming/erasing program, use the DQ7, DQ6, DQ5, DQ3 and DQ2 flags to check that automatic programming/erasing is terminated and read data.
- The hardware sequence flags can also be used to check whether the second and later sector erase code writing is enabled.
### Explanation of hardware sequence flag

Table 24.5-2 shows list of hardware sequence flag function.

**Table 24.5-2 List of Hardware Sequence Flag Functions**

<table>
<thead>
<tr>
<th>State change in normal operation</th>
<th>State</th>
<th>DQ7</th>
<th>DQ6</th>
<th>DQ5</th>
<th>DQ3</th>
<th>DQ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming → Write Completed (Writing address specified)</td>
<td>Programming → Write Completed (Writing address specified)</td>
<td>DQ7 → Toggle</td>
<td>Toggle → DATA: 7</td>
<td>DQ6 → Toggle</td>
<td>DATA: 6</td>
<td>0 → DATA: 5</td>
</tr>
<tr>
<td>Chip and sector erasing → Erase Completed</td>
<td>Chip and sector erasing → Erase Completed</td>
<td>0 → 1</td>
<td>Toggle → Toggle</td>
<td>0 → 1</td>
<td>Toggle → Toggle</td>
<td>0</td>
</tr>
<tr>
<td>Sector erasing wait → Erase Started</td>
<td>Sector erasing wait → Erase Started</td>
<td>0</td>
<td>Toggle</td>
<td>0</td>
<td>Toggle</td>
<td>0</td>
</tr>
<tr>
<td>Delete → Suspend sector deletion (Sector being erased)</td>
<td>Delete → Suspend sector deletion (Sector being erased)</td>
<td>0 → 1</td>
<td>Toggle → Toggle</td>
<td>0</td>
<td>Toggle</td>
<td>0</td>
</tr>
<tr>
<td>Sector Erasing being Suspended → Resume Erased (Sector being erased)</td>
<td>Sector Erasing being Suspended → Resume Erased (Sector being erased)</td>
<td>1 → 0</td>
<td>1 → Toggle</td>
<td>0</td>
<td>Toggle</td>
<td>0</td>
</tr>
<tr>
<td>Sector erasing being suspended (Sector not being deleted)</td>
<td>Sector erasing being suspended (Sector not being deleted)</td>
<td>DATA: 7</td>
<td>DATA: 6</td>
<td>DATA: 5</td>
<td>DATA: 3</td>
<td>DATA: 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Abnormal operation</th>
<th>Programming</th>
<th>DQ7</th>
<th>Toggle</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chip and sector erasing</td>
<td>Chip and sector erasing</td>
<td>0</td>
<td>Toggle</td>
<td>1</td>
<td>1</td>
<td>*</td>
</tr>
</tbody>
</table>

*: When DQ5 is "1" (timing limit exceeded), DQ2 carries out toggle operation to continuous reading of the sector during writing/deletion, and does not carry out toggle operation to reading other sectors.
24.5.1 Data Polling Flag (DQ7)

The data polling flag (DQ7) is mainly used to notify that the automatic algorithm is executing or has been completed using the data polling function.

Table 24.5-3 and Table 24.5-4 show state transition of data polling flag.

Table 24.5-3 State Transition of Data Polling Flag (State Change at Normal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming → Write completed</th>
<th>Chip Sector Erasing → Erase Completed</th>
<th>Wait for Sector Erasing → Erase Started</th>
<th>Sector Erasing → Sector Erasing Temporary Stop (Sector being erased)</th>
<th>Sector Erasing Temporary → Resume erasing (Sector being Erased)</th>
<th>Sector Erasing Temporary (Sector not being erased)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ7</td>
<td>DQ7 → DATA: 7</td>
<td>0 → 1</td>
<td>0</td>
<td>0 → 1</td>
<td>1 → 0</td>
<td>DATA: 7</td>
</tr>
</tbody>
</table>

Table 24.5-4 State Transition of Data Polling (State Change at Abnormal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming</th>
<th>Chip and sector erasing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ7</td>
<td>DQ7</td>
<td>0</td>
</tr>
</tbody>
</table>

- **At programming**
  - Read access during execution of the auto-programming algorithm causes flash memory to output the reversed data of bit 7 last written.
  - Read access at the end of the auto-programming algorithm causes flash memory to output the value of bit 7 at the address to which read access was performed.

- **At chip/sector erasing**
  During executing chip and sector erasing algorithms, when read access is made to the currently being erasing sector, bit 7 of flash memory outputs 0. When chip erasing/sector erasing is terminated, bit 7 of flash memory outputs 1.
At sector erasing suspension

- Read access during sector erasing suspension causes flash memory to output 1 if the address specified by the address signal belongs to the sector being erased. If a read access is made in the sector erasing suspension state, flash memory outputs bit 7 (DATA: 7) for the read value of the read address when the read address is not the sector being erased.
- Referring this flag together with the toggle bit flag (DQ6) permits a decision on whether flash memory is in the erase suspended state and which sector is being erased.

**Note:** Read access to the specified address while the automatic algorithm starts is ignored. At data reading, other bits can be output at the end of data polling flag (DQ7). Data reading after the end of the automatic algorithm should be performed following read access after completion of data polling has been checked.
24.5.2 Toggle Bit Flag (DQ6)

The toggle bit flag (DQ6) is a hardware sequence flag to notify that the automatic algorithm is being executed or in the end state using the toggle bit function.

Table 24.5-5 and Table 24.5-6 show the state transition of toggle bit flag.

Table 24.5-5 State Transition of Toggle Bit Flag (State Change at Normal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming → Write completed</th>
<th>Chip Sector Erasing → Erase Completed</th>
<th>Wait for Sector Erasing → Erase Started</th>
<th>Sector Erasing → Sector Erasing Temporary Sector (Sector being erased)</th>
<th>Sector Erasing Temporary → Resume erasing (Sector being erased)</th>
<th>Sector Erasing Temporary (Sector not being erased)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ6</td>
<td>Toggle → DATA:6</td>
<td>Toggle → Stop</td>
<td>Toggle</td>
<td>Toggle → 1</td>
<td>1 → Toggle</td>
<td>DATA:6</td>
</tr>
</tbody>
</table>

Table 24.5-6 State Transition of Toggle Bit Flag (State Change at Abnormal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming</th>
<th>Chip and sector erasing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ6</td>
<td>Toggle</td>
<td>Toggle</td>
</tr>
</tbody>
</table>

**At programming and chip/sector erasing**

- When read access is continuously performed during algorithm execution for auto-writing algorithm and chip/sector deletion, the flash memory outputs toggles alternately "1" and "0" each time read is performed.
- If a continuous read access is made after the completion of the automatic algorithm for programming and chip erasing/sector erasing, flash memory outputs bit 6 (DATA: 6) for the read value of the read address every reading.

**At sector erasing suspension**

If a read access is made in the sector erasing suspension state, flash memory outputs 1 when the read address is the sector being erased. Flash memory outputs bit 6 (DATA: 6) for the read value of the read address when the read address is not the sector being erased.

**Reference:**

If the sector for programming is reprogram-protected, the toggle bit flag (DQ6) produces a toggle output for approximately 2 µs, and then terminates it without reprogramming data.
If all sectors for erasing are reprogram-protected, the toggle bit flag (DQ6) produces a toggle output for approximately 100 µs, and then returns to the read/reset state without reprogramming data.
24.5.3 Timing Limit Over Flag (DQ5)

The timing limit over flag (DQ5) is a hardware sequence flag that notifies flash memory that the execution of the automatic algorithm has exceeded a prescribed time (the time required for programming/erasing).

Table 24.5-7 State Transition of Timing Limit Over Flag (State Change at Normal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming → Write completed</th>
<th>Chip Sector Erasing → Erase Completed</th>
<th>Wait for Sector Erasing → Erase Started</th>
<th>Sector Erasing → Sector Erasing Temporary (Sector being erased)</th>
<th>Sector Erasing Temporary → Resume erasing (Sector being Erased)</th>
<th>Sector Erasing Temporary (Sector not being erased)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ5</td>
<td>0 → DATA: 5</td>
<td>0 → 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>DATA: 5</td>
</tr>
</tbody>
</table>

Table 24.5-8 State Transition of Timing Limit Over Flag (State Change at Abnormal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming</th>
<th>Chip and sector erasing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ5</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

● At programming and chip/sector erasing

- If a read access made after starting the automatic algorithm for programming or chip erasing/sector erasing is within a prescribed time (the time required for programming/erasing), the timing limit over flag (DQ5) outputs 0. If it exceeds the prescribed time, the timing limit over flag (DQ5) outputs 1.
- The timing limit over flag (DQ5) can be used to identify the success or failure of programming/erasing, regardless of whether the automatic algorithm is in progress or terminated. If the automatic algorithm by the data polling or the toggle bit function is in execution when the timing limit over flag (DQ5) outputs 1, programming can be identified as a failure.
- For example, when "1" is set to the flash memory address with "0" set the flash memory, programming fails. In this case, the flash memory will be locked and the automatic algorithm will not complete. In rare cases normal termination may be seen as with the case where "1" can be written. Therefore, no valid data is output from the data polling flag (DQ7), the toggle bit flag (DQ6) does not stop the toggle operation and exceeds the time limit, causing the timing limit over flag (DQ5) to output 1. The state that timing limit over flag (DQ5) outputs 1 means that the flash memory is not being used correctly; it does not mean that the flash memory is faulty. When this state occurs, execute the reset command.
24.5.4 Sector Deletion Timer Flag (DQ3)

The sector erase timer flag (DQ3) is a hardware to notify during the period of waiting for sector erasing after the sector erase command has started.

### Sector Deletion Timer Flag (DQ3)

Table 24.5-9 and Table 24.5-10 show state transition of sector erase timer flag.

**Table 24.5-9 State Transition of Sector Erase Timer Flag (State Change at Normal Operation)**

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming → Write completed</th>
<th>Chip Sector Erasing → Erase Completed</th>
<th>Wait for Sector Erasing → Erase Started</th>
<th>Sector Erasing → Sector Erasing Temporary (Sector being erased)</th>
<th>Sector Erasing Temporary → Resume erasing (Sector being Erased)</th>
<th>Sector Erasing Temporary (Sector not being erased)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ3</td>
<td>0 → DATA: 3</td>
<td>1</td>
<td>0 → 1</td>
<td>1 → 0</td>
<td>0 → 1</td>
<td>DATA: 3</td>
</tr>
</tbody>
</table>

**Table 24.5-10 State Transition of Sector Erase Timer Flag (State Change at Abnormal Operation)**

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming</th>
<th>Chip and sector erasing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ3</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- **At sector erasing**
  - If a read access made after starting the sector erase command is within a sector erasing wait period, the sector erasing timer flag (DQ3) outputs 0. If it exceeds the period, the sector erasing timer flag (DQ3) outputs 1.
  - If the sector erasing timer flag (DQ3) is "1", indicating that the automatic algorithm for sector erasing by the data polling or toggle bit function is in progress (DQ7 = 0, DQ6 produces a toggle output), sector erasing is performed. If any command other than the sector erasing suspension is set, it is ignored until sector erasing is terminated.
  - If the sector erasing timer flag (DQ3) is "0", flash memory can accept the sector erase command. To program the sector erase command, check that the sector erasing timer flag (DQ3) is "0". If the flag is "1", flash memory may not accept the sector erase command of suspending.

- **At sector erasing suspension**
  - If a read access is made in the sector erasing suspension state, flash memory outputs 1 when the read address is the sector being erased. If a read access is made in the sector erasing suspension state, flash memory outputs bit 3 (DATA: 3) for the read value of the read address when the read address is not the sector being erased.
24.5.5 Toggle Bit 2 Flag (DQ2)

The toggle bit 2 flag (DQ2) notifies that sector deletion is temporarily paused via the toggle function.

**Toggle bit 2 flag (DQ2)**

Table 24.5-11 and Table 24.5-12 show state transition of toggle bit 2 flag.

### Table 24.5-11 State Transition of Toggle Bit 2 Flag (State Change at Normal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming → Write completed</th>
<th>Chip Sector Erasing → Erase Completed</th>
<th>Wait for Sector Erasing → Erase Started</th>
<th>Sector Erasing → Sector Erasing Temporary (Sector being erased)</th>
<th>Sector Erasing Temporary → Resume erasing (Sector being Erased)</th>
<th>Sector Erasing Temporary (Sector not being erased)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ2</td>
<td>1 → DATA:2</td>
<td>Toggle → Stop</td>
<td>Toggle</td>
<td>Toggle</td>
<td>Toggle</td>
<td>DATA: 2</td>
</tr>
</tbody>
</table>

### Table 24.5-12 State Transition of Toggle Bit Flag (State Change at Abnormal Operation)

<table>
<thead>
<tr>
<th>Operating State</th>
<th>Programming</th>
<th>Chip and sector erasing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ2</td>
<td>1</td>
<td>*</td>
</tr>
</tbody>
</table>

*: When DQ5 is "1" (timing limit exceeded), DQ2 carries out toggle operation to continuous reading of the sector during writing/deletion, and does not carry out toggle operation to reading other sectors.

- **Sector deletion**
  - If a continuous read access is made during the execution of the automatic algorithm for chip erasing/sector erasing, flash memory toggle-outputs 1 and 0 alternately every reading.
  - If read access is continued after chip deletion/sector deletion algorithm ends, the flash memory outputs read value bit 2 (DATA:2) for each read operation.

- **At sector erasing suspension**
  - If a read access is made in the sector erasing suspension state, flash memory outputs 1 and 0 alternately when the read address is the sector being erased. If a read access is made in the sector erasing suspension state, flash memory outputs bit 2 (DATA: 2) for the read value of the read address when the read address is not the sector being erased.
  - If programming is performed in the sector erasing suspension state, flash memory outputs 1 when a continuous read access is started with the sector that is not in the erasing suspension state.
  - Toggle flag 2 bit (DQ2) is used with the toggle bit flag (DQ6) to detect when deletion is paused. (the DQ2 flag performs the toggle operation but the DQ6 flag does not)
  - Since toggle flag 2 bit (DQ2) performs toggle operations when a read access is made to a sector that is being deleted, it can also be used to detect delete operations.
| Reference: | When all sectors that perform deletion are write protected, after the toggle bit2 flag (DQ2) performs toggle output for approx. 100µs, it returns to the read/reset status without rewriting any data. |
24.6 Details of Programming/Erasing Flash Memory

The following describes how to enter the command to start an auto algorithm, reading/resetting flash memory, writing, chip deletion, sector deletion, pausing sector deletion, restarting sector deletion.

Details of Programming/Erasing Flash Memory

Auto algorithms can be started up by writing the read/reset, write, chip delete, sector delete, pause sector delete, resume sector delete command sequence from CPU to flash memory. Programming flash memory from the CPU should always be performed continuously. The termination of the automatic algorithm can be checked by the data polling function. After normal termination, it returns to the read/reset state.

Each operation is explained in the following order.

- Read/reset state
- Programming data
- All data erasing (chip all erase)
- Any data erasing (sector erase)
- Sector erasing suspension
- Sector erasing resumption
24.6.1 Read/Reset State in Flash Memory

This section explains the procedure for inputting the read/reset command to place flash memory in the read/reset state.

Read/Reset State in Flash Memory

- To read and reset the flash memory, continuously transfer read/reset commands of the command sequence table from the CPU to flash memory.
- There are two kinds of read/reset commands: one is executed at one time bus operation, and the other is executed at three times bus operations; the command sequence of both is essentially the same.
- Since the read/reset state is the initial state for flash memory, flash memory always enters this state after power on and at the normal termination of command. The read/reset state is also described as the wait state for command input.
- In the read/reset state, a read access to flash memory enables data to be read. As is the case with mask ROM, a program access from the CPU can be made. A read access to flash memory does not require the read/reset command. If the command is not terminated normally, use the read/reset command to initialize the automatic algorithm.
24.6.2 Writing Data to Flash Memory

Procedures for entering write commands to write data to flash memory are described.

- **Writing data to flash memory**
  - In order to start the data programming automatic algorithm, continuously transmit the program command in the command sequence table from CPU to flash memory.
  - At completion of data programming to a target address in the fourth cycle, the automatic algorithm starts automatic programming.

- **How to Specify the Address**
  - The only even addresses can be specified for the programming address specified by programming data cycle. Specifying odd addresses prevents correct writing. Writing to even addresses must be performed in word data units.
  - Programming is possible in any address order or even beyond sector boundaries. However, execution of one programming command, permits programming of only one word for data.

- **Notes on data programming**
  - Data 0 cannot be returned to data 1 by programming. When data 0 is programmed to data 1, the data polling algorithm (DQ7) or toggling (DQ6) is not terminated and the flash memory is considered faulty; the timing limit over flag (DQ5) is determined as an error.
  - When data is read in the read/reset state, the bit data remains 0. To return the bit data to 1 from 0, erase flash memory data.
  - All commands are ignored during automatic programming. If a hardware reset occurs during programming, data being programmed to addresses are not assured.

- **Data Programming Procedure**
  - Figure 24.6-1 shows example of data programming procedure. The hardware sequence flags can be used to check the operating state of the automatic algorithm in flash memory. The data polling flag (DQ7) is used for checking the completion of programming to flash memory in this example.
  - Flag check data should be read from the address where data was last written.
  - Because the data polling flag (DQ7) and the timing limit over flag (DQ5) change at the same time, the data polling flag (DQ7) must be checked even when the timing limit over flag (DQ5) is "1".
  - Similarly, since the toggle bit flag (DQ6) stops toggling at the same time the timing limit over flag (DQ5) changes to "1", toggle bit flag (DQ6) must be checked.
Figure 24.6-1 Example of Data Programming Procedure

1. **Programming start**
   - FMCS : WE (bit5)
   - Flash memory program enabled

2. **Programming command sequence**
   - 1. FFAAAA ← XXAA
   - 2. FF5554 ← XX55
   - 3. FFAAAA ← XXA0
   - 4. Programming address ← Programming data

3. **Internal address read**
4. **Data poling (DQ7)**
5. **Timing limit (DQ5)**
6. **Internal address read**
7. **Data poling (DQ7)**

8. **Programming error**
9. **Last address**
   - YES: FMCS : WE (bit5)
   - Flash memory programming disabled
   - Programming completed

10. **Next address**

- Insertion: Check by hardware sequence flag

**Notes:**
- Check by hardware sequence flag.
24.6.3 Erasing All Data from Flash Memory (Chip Erase)

This section explains the procedure for inputting the chip erase command to erase all data from flash memory.

**Data Erase from Flash Memory (Chip Erase)**

- All data can be erased from flash memory by continuously transmitting the chip erase command in the command sequence table from CPU to flash memory.
- The chip erase command is executed in six bus operations. Chip erasing is started at completion of the sixth programming cycle.
- Before chip erasing, the user need not perform programming to flash memory. During execution of the automatic erasing algorithm, flash memory automatically programs 0 before erasing all cells.
24.6.4 Erasing Any Data in Flash Memory (Sector Erasing)

Procedures for entering a sector delete command to delete specific flash memory sectors (sector delete) are described. Sector-by-sector erasing is enabled and multiple sectors can be specified at a time.

- Erasing Any Data in Flash Memory (Sector Erasing)
  Any sector in flash memory can be erased by continuously transmitting the sector erase command in the command sequence table from CPU to flash memory.

  • Method of Specifying a Sector
    - The sector erase command is executed in six bus operations. By setting the address on the sixth cycle in the even address in the target sector and programming the sector erase code (30H) to data, a 50 μs sector erasing wait is started
    - When erasing more than one sector, the sector erase code (30H) is programmed to the sector address to be erased, following the above.

  • Notes on specifying multiple sectors
    - Sector erasing is started after a 50 μs period waiting for sector erasing is completed after the last sector erase code has been programmed.
    - That is, when erasing more than one sector simultaneously, the address of erase sector address and the sector code (the sixth cycle of command sequence) must be input within 50 μs. If the sector erase code is input 50 μs or later, it cannot be accepted.
    - Whether continuous programming of the sector erase code is enabled can be checked by the sector erase timer flag (DQ3).
    - In this case, the address from which the sector erase timer is flag (DQ3) read should correspond to the sector to be erased.

- Erasing Procedure for Flash Memory Sectors
  - The hardware sequence flags can be used to check the operating state of the automatic algorithm in flash memory. Figure 24.6-2 gives an example of the flash memory sector erase procedure. In this example, the toggle bit flag (DQ6) is used to check that erase ends.
  - DQ6 terminates toggling concurrently with the change of the timing limit over flag (DQ5) to 1. So the DQ6 must be checked even when DQ5 is 1.
  - Similarly, the data polling flag (DQ7) changes concurrently with the transition of the DQ5, so DQ7 must be checked.
Figure 24.6-2 Example of Sector Erasing Procedure

1. Erase start
2. FMCS : WE (bit5)
   Flash memory erase enabled
3. Erase command sequence
   - FxAAAA ← XXAA
   - Fx5554 ← XX55
   - FxAAAA ← XX80
   - FxAAAA ← XXAA
   - Fx5554 ← XX55
4. Code input to erase sector (30H)
5. Any other erase sector?
   - NO
   - YES
   Internal address read 1
6. Internal address read 2
7. Toggle bit (DQ6)
   data 1 = data
8. Timing limit (DQ5)
   - NO
   - YES
9. Internal read 1
10. Internal read 2
11. Toggle bit (DQ6)
    data 1 = data 2
12. Last sector
   - YES
   - NO
   FMCS : WE (bit5)
   Flash memory erase disabled
13. Erase completed
   - YES
   - NO
   Erasing error
14. Sector erase Completed
   - YES
   - NO

Note: Check by the hardware sequence flag
24.6.5 Sector Erase Suspension of Flash memory

This section explains the procedure for inputting the sector erase suspend command to suspend sector erasing flash memory. Data can be read from sectors that are not being deleted.

- **Sector Erase Suspension**

  - To cause flash memory sector erasing to suspend, transmit the sector erasing suspend command in the command sequence table from CPU to flash memory.
  
  - The sector erasing suspend command suspends the sector erase currently being performed, enabling data read from a sector that is currently not being erased. Only read can be performed when this command is suspended; programming cannot be performed.
  
  - This command is only enabled during the sector erasing period including the erasing wait time; it is ignored during the chip erasing period or during programming.
  
  - The sector erasing suspend command is executed when the sector erasing suspend code (B0H) is programmed. Arbitrary address in flash memory should be set for address. If the sector erasing suspend command is executed during sector erasing pause, the command input again is ignored.
  
  - When the sector erasing suspend command is input during the sector erasing wait period, the sector erase wait state ends immediately, the erasing is interrupted, and the erase stop state occurs.
  
  - When the erase suspend command is input during the sector erasing after the sector erase wait period, the erase suspend state occurs after 20 µs max.
  
  - The sector erasing suspend command should be executed after 20ms and more ms when the sector erasing command or the sector erase resume command have been issued.
24.6.6 Sector Erase Resumption of Flash Memory

This section explains the procedure for inputting the sector erase resume command to resume erasing of the suspended flash memory sector.

- **Sector Erase Resumption of Flash Memory**
  - Suspended sector erasing can be resumed by transmitting the sector erase resume command in the command sequence table from CPU to flash memory.
  - The sector erase resume command resumes sector erasing suspended by the sector erase suspend command. This command is executed by writing the erase resume code (30H). In this case, address in the flash memory area is specified.
  - Inputting the sector erase resume command during sector erasing is ignored.
24.7 Notes on Using Flash Memory

This section contains notes on using 1M-bit flash memory.

- **Notes on using flash memory**

  - **Input of a hardware reset (RST)**

    To input a hardware reset when the automatic algorithm has not been started and reading is in progress, a minimum low-level width of 500 ns must be maintained. In this case, a maximum of 500 ns is required until data can be read from the flash memory after a hardware reset has been activated.

    Similarly, to input a hardware reset when the automatic algorithm has been activated and writing or erasing is in progress, a minimum low-level width of 500 ns must be maintained. In this case, 20 µs are required until data can be read after the operation for initializing the flash memory has terminated.

    A hardware reset during writing the data being written to be undefined.

  - **Canceling of a software reset, watchdog timer reset, and hardware standby**

    When the flash memory is being written to or erased with CPU access and if reset conditions occur while the automatic algorithm is active, the CPU may run out of control. This occurs because these reset conditions cause the automatic algorithm to continue without initializing the flash memory unit, possibly preventing the flash memory unit from entering the read state when the CPU starts the sequence after the reset has been deasserted. These reset conditions must be disabled during writing to or erasing of the flash memory.

  - **Program access to flash memory**

    When the automatic algorithm is operating, read access to the flash memory is disabled. With the memory access mode of the CPU set to internal ROM mode, writing or erasing must be started after the program area is switched to another area such as RAM. In this case, when sectors (SA4) containing interrupt vectors are erased, writing or erasing interrupt processing cannot be executed. For the same reason, all interrupt sources other than the flash memory are disabled while the automatic algorithm is operating.

  - **Extended intelligent I/O service (EI²OS)**

    Because write and erase interrupts issued to the CPU from the flash memory interface circuit cannot be accepted by the EI²OS, they should not be used.

  - **Applying V_ID**

    Applying V_ID required for the sector protect operation should always be started and terminated when the supply voltage is on.
24.8 Program Example of 1 Mbit Flash Memory

A 1 Mbyte flash memory program example is shown below.

## 1 Mbit Flash Memory Program Example

```
<table>
<thead>
<tr>
<th>NAME</th>
<th>FLASHWE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>FLASHWE</td>
</tr>
</tbody>
</table>

;------------------------------------------------------------------
;1Mbit-FLASH sample program
;1:Transfer program in FLASH (address FFBC00H sector to SA4) to RAM (address 000700H)
;2:It is run in RAM.
;3: PDR1 value is written to Flash (address FE0000H sector SA0)
;4:Written values (address FE0000H sector SA0) are read and output to PDR2.
;5:Delete written sectors (SA0).
;6:Outputting check of deleted data
; Conditions
; • RAM transfer bytes: 100H (256 bytes)
; • Criteria for determining whether writing and deleting ended
;     DQ5 (timing limit excess flag criteria (timing limit excess flag) criteria
;     DQ6 (toggle bit flag) criteria
;     RDY (FMCS) criteria
; • Error processing
;     "H" is output by P00 to P07
;     Issuing reset commands
------------------------------------------------------------------
RESOUS IOSEGABS=00 ;"RESOUS" I/O segment definition
ORG 0000H
PDR0 RB 1
PDR1 RB 1
PDR2 RB 1
PDR3 RB 1
ORG 0010H
DDR0 RB 1
DDR1 RB 1
DDR2 RB 1
DDR3 RB 1
ORG 00A1H
CKSCR RB 1
ORG 00AEH
FMCS RB 1
ORG 006FH
ROMM RB 1
RESOUS ENDS
; SSTA SSEG
RW 0127H
STA_T RW 1
SSTA ENDS
```
CHAPTER 24  1M-BIT FLASH MEMORY

; DATA DSEG ABS=0FFH ;FLASH command address
ORG 5554H
COMADR2 RW 1
ORG 0AAAAH
COMADR1 RW 1
DATA ENDS

;Main Program (SA1)

---------------------------------------------------------------------
CODE CSEG
START:

;---------------------------------------------------------------------
;Initialization
;---------------------------------------------------------------------
MOV CKSCR,#0BAH ; set to multiply by 3
MOV RP,#0
MOV A,#!STA_T
MOV SSB,A
MOVW A,#STA_T
MOVW SP,A
MOV ROMM,#00H ; mirror OFF
MOV PDR0,#00H ; for error checking
MOV DDR0,#0FFH
MOV PDR1,#00H ; data I/O port
MOV DDR1,#00H
MOV PDR2,#00H ; data output port
MOV DDR2,#0FFH

;---------------------------------------------------------------------
; Transfer the "Flash write program (FFBC00H)" to RAM (address 700H)
;---------------------------------------------------------------------
MOVW A,#0700H ; transfer destination RAM area
MOVW A,#0BC00H ; transfer address (program position)
MOVW RW0,#0100H ; transfer no of bytes
MOVW A,#0000H ; transferring 100H from MOVSADB, PCB; FFBC00H to 000700H
CALLP 000700H ; Jump to address of transferred program

;---------------------------------------------------------------------
; data output
;---------------------------------------------------------------------
OUT MOV A,#0FEH
MOV ADB,A
MOVW RW2,#0000H
MOV A, @RW2+00
MOV PDR2,A
END JMP *

---------------------------------------------------------------------

; Flash write delete program (SA4)

---------------------------------------------------------------------
RAMPRG CSEG ABS=0FFH
ORG 0BC00H

; Initialization
;
MOVW RW0,#0500H ;RW0: RAM space 00: from 0500 for input data assurance
MOVW RW2,#0000H ;RW2: flash memory write address FD: from 0000
MOV A,#00H ; DTB conversion
MOV DTB,A ; @RW0 bank setting
MOV A,#0FEH ; ADB conversion 1
MOV ADB,A ; Bank specification for write mode selection address
MOV PDR3,#00H ; switch initialization
MOV DDR3,#00H

; WAIT1 BBC PDR3:0,WAIT1 ; PDR3:0 "H" starts writing
;
;---------------------------------------------------------------------
; Programming (SA0)
;---------------------------------------------------------------------

MOV A,PDR1
MOVW @RW0+00,A ; PDR1 data is stored in RAM
MOV FMCS,#20H ; write mode setting
MOVW ADB:COMADR1,#00AAH ; flash write command 1
MOVW ADB:COMADR2,#0055H ; flash write command 2
MOVW ADB:COMADR1,#00A0H ; flash write command 3

; MOVW A,@RW0+00 ; input data (RW0) is input to flash memory (RW2)
; Writing

; MOVW @RW2+00,A
WRITE ; wait time check
;---------------------------------------------------------------------
; When the time limit excess check flag goes on during toggle operation
; ERROR
;---------------------------------------------------------------------
MOVW A,@RW2+00
AND A,#20H ; DQ5 time limit check
BZ NTOW ; time limit over
MOVW A,@RW2+00 ; AH
MOVW A,@RW2+00 ; AL
XORW A ; XOR of AH, AL ("1" if different)
AND A,#40H ; DQ6 toggle bit differs
BNZ ERROR ; to ERROR if different
;---------------------------------------------------------------------
; Write end check (FMCS to RDY)
;---------------------------------------------------------------------
NTOW MOVW A,FMCS
AND A,#10H ; FMCS RDY bit (4-bit) is extracted
BZ WRITE ; end write?
MOV FMCS,#00H ; write mode is canceled
;---------------------------------------------------------------------
; Write data output
;
MOVW RW2,#0000H ; write data output
MOVW A,@RW2+00
MOV PDR2,A

; WAIT2 BBC PDR3:1,WAIT2 ; PDR3:1 "H" starts sector deletion
; Chapter 24  1M-Bit Flash Memory

; Sector Erasing (SA0)

; MOV @RW2+00,#0000H  ; address initialization
; MOV FMCS,#20H        ; delete mode setting
; MOVW ADB:COMADR1,#00AAH; flash delete command 1
; MOVW ADB:COMADR2,#0055H; flash delete command 2
; MOVW ADB:COMADR1,#0080H; flash delete command 3
; MOVW ADB:COMADR1,#00AAH; flash delete command 4
; MOVW ADB:COMADR2,#0055H; flash delete command 5
; MOV @RW2+00,#0030H  ; issue the delete command to the deleting sector

ELS ; wait time check

; When the time limit excess check flag goes on during toggle operation
; ERROR

; MOVW A,@RW2+00
; AND A,#20H           ; DQS time limit check
; BZ NTOE             ; time limit over
; MOVW A,@RW2+00      ; During AH writing operation from
; MOVW A,@RW2+00      ; Output "H/L" alternately for each AL read
; operation.
; XORW A               ; When AH and AL XOR (DQ6) values differ,
;                      ; the counter is cleared when a compare
;                      ; match occurs.
;                     ; "1" Writing)
; AND A,#40H           ; is the DQ6 toggle bit "H"?
; BNZ ERROR           ; If "H" to ERROR

; Write end check (FMCS to RDY)

; NTOE
; MOVW A,FMCS         ;
; AND A,#10H          ; FMCS RDY bit (4-bit) is extracted
; BZ ELS              ; end sector deletion?
; MOV FMCS,#00H       ; Flash delete mode canceled
; RETP ; Return to main program

; error

; ERROR
; MOV ADB:COMADR1,#0F0H ; Reset command (read is enabled)
; MOV FMCS,#00H        ; Flash mode canceled
; MOV PDR0,#0FFH       ; error processing check
; RETP ; Return to main program

RAMPGR ENDS

VECT CSEG ABS=0FFH
ORG 0FFDCH
DSL START
DB 00H
VECT ENDS
CHAPTER 25

CONNECTION EXAMPLE OF
FLASH SERIAL PROGRAMMING

Explanation is given for a serial writing connection example when the flash microcomputer programmer made by Yokogawa Digital Computer Corporation is used.

25.1 Basic Configuration of Serial Programming Connection for F2MC-16LX MB90F523B
25.2 Connection Example in Single-chip Mode (User Power Supply)
25.3 Connection Example in Single-chip Mode (Writer Power Supply)
25.4 Example of Minimum Connection to Flash Microcomputer Programmer (User Power Supply)
25.5 Example of Minimum Connection to Flash Microcomputer Programmer (Writer Power Supply)
25.1 Basic Configuration of Serial Programming Connection for F2MC-16LX MB90F523B

MB90F523B supports the Fujitsu standard for serial on-board writing of the flash ROM. Specifications for on-board writing are explained as follows.

- Basic Configuration of Serial Programming Connection for F2MC-16LX MB90F523B
  The flash microcomputer programmer made by Yokogawa Digital Computer Corporation is used for Fujitsu standard serial on-board writing.

Figure 25.1-1 Basic Configuration of Serial Programming Connection

<table>
<thead>
<tr>
<th>Pin</th>
<th>Pin Description</th>
<th>Supplementary Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD2, MD1,</td>
<td>Mode input pin</td>
<td>Flash serial writing mode is active when setting MD2 = 1, MD1 = 1, and MD0 = 0.</td>
</tr>
<tr>
<td>MD0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X0, X1</td>
<td>Oscillation clock pin</td>
<td>Under flash serial writing mode, the CPU internal operating clock will be 1 multiplication of the PLL clock. As the oscillation clock frequency is determined by the CPU internal operating clock, the 1 MHz to 16 MHz oscillators can be used for serial writing.</td>
</tr>
<tr>
<td>P00, P01</td>
<td>Programming program start pins</td>
<td>Input a &quot;L&quot; level to P00 and a &quot;H&quot; level to P01.</td>
</tr>
<tr>
<td>RST</td>
<td>Reset pin</td>
<td>-</td>
</tr>
<tr>
<td>HST</td>
<td>Hardware standby pin</td>
<td>Set &quot;1&quot; to the HST during the serial writing mode.</td>
</tr>
</tbody>
</table>
Table 25.1-1 Pins Used for Fujitsu Standard Serial On-board Programming (2 / 2)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Pin Description</th>
<th>Supplementary Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIN0</td>
<td>Serial data input pin</td>
<td></td>
</tr>
<tr>
<td>SOT0</td>
<td>Serial data output pin</td>
<td>UART is used under clock-synchronous mode (mode 0).</td>
</tr>
<tr>
<td>SCK0</td>
<td>Serial clock input pin</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>C pin</td>
<td>This pin is a capacitance pin for stabilizing supply voltage. Connect the ceramic capacitor approx. 0.1 µF externally</td>
</tr>
<tr>
<td>VCC</td>
<td>Supply voltage pin</td>
<td>If the program voltage (5 V ± 10%) is supplied from the user system, the flash microcomputer programmer need not be connected.</td>
</tr>
<tr>
<td>VSS</td>
<td>GND pin</td>
<td>Share GND with the flash microcomputer programmer.</td>
</tr>
</tbody>
</table>

Note: The control circuit as "Figure 25.1-2 Control circuit" is required when the P00, SIN0, SOT0 and SCK0 pins are used for the user system as well. The user circuit can be separated during serial writing by the TICS signal of the flash microcomputer programmer.

See the following serial programming connection examples given in "25.2 Connection Example in Single-chip Mode (User Power Supply)" to "25.5 Example of Minimum Connection to Flash Microcomputer Programmer (Writer Power Supply)".

- Connection example in single-chip mode (user power supply)
- Connection example in single-chip mode (write power supply)
- Example of Minimum Connection to Flash Microcomputer Programmer (User Power Supply)
- Example of Minimum Connection to Flash Microcomputer Programmer (Writer Power Supply)
CHAPTER 25  CONNECTION EXAMPLE OF FLASH SERIAL PROGRAMMING

■ Oscillation clock frequency and serial clock input frequency

Serial clock frequency that can input MB90F523B can be calculated using the following calculation formula, so set up the serial clock input frequency to the flash microcomputer programmer in accordance with the oscillation clock frequency used.

Imputable serial clock frequency = $0.125 \times$ oscillation clock frequency.

Table 25.1-2  Example of Input Enable Serial Clock Frequency

<table>
<thead>
<tr>
<th>Oscillation clock frequency</th>
<th>Possible to Input maximum serial clock frequency of microcomputer</th>
<th>Settable maximum serial clock frequency of AF220/AF210/AF120/AF110</th>
<th>Settable maximum serial clock frequency of AF200</th>
</tr>
</thead>
<tbody>
<tr>
<td>at 4 MHz</td>
<td>500kHz</td>
<td>500kHz</td>
<td>500kHz</td>
</tr>
<tr>
<td>at 8 MHz</td>
<td>1MHz</td>
<td>850kHz</td>
<td>500kHz</td>
</tr>
<tr>
<td>at 16 MHz</td>
<td>2MHz</td>
<td>1.25MHz</td>
<td>500kHz</td>
</tr>
</tbody>
</table>

■ System configuration of the flash microcomputer programmer (made by Yokogawa Digital Computer Corporation)

Table 25.1-3  System configuration of the flash microcontroller programmer (made by Yokogawa Digital Computer Corporation)

<table>
<thead>
<tr>
<th>Model</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF220/AC4P</td>
<td>Model with internal Ethernet interface/100V to 220V power adapter</td>
</tr>
<tr>
<td>AF210/AC4P</td>
<td>Standard model/100 V to 220 V power adapter</td>
</tr>
<tr>
<td>AF120/AC4P</td>
<td>Model with single key internal Ethernet interface/100V to 220V power adapter</td>
</tr>
<tr>
<td>AF110/AC4P</td>
<td>Single key model/100 V to 220 V power adapter</td>
</tr>
<tr>
<td>AZ221</td>
<td>PC/AT RS232C cable for writer</td>
</tr>
<tr>
<td>AZ210</td>
<td>Standard target probe (a) length: 1 m</td>
</tr>
<tr>
<td>FF201</td>
<td>Control module for Fujitsu F^2^MC-16LX flash microcomputer control module</td>
</tr>
<tr>
<td>/P2</td>
<td>2MB PC Card (Option) Flash memory to 128KB corresponding</td>
</tr>
<tr>
<td>/P4</td>
<td>4MB PC Card (Option) Flash memory to 512KB corresponding</td>
</tr>
</tbody>
</table>

Note: The AF200 flash microcomputer programmer is an end product but is made available using the control module FF201. Serial writing connection example is shown from “25.2  Connection Example in Single-chip Mode (User Power Supply)” onward.
25.2 Connection Example in Single-chip Mode
(User Power Supply)

MD2 = "1" and MD0 = "0" will be set for mode pins MD2 and MD0 of the user system that
has been set to single chip mode from the pins TAUX3 and TMODE of AF220/AF210/
AF210/AF120/AF110, and flash serial writing mode will be adopted. An connection
e example using the user power supply is given below.

■ Connection Example in Single-chip Mode (User Power Supply Used)

Figure 25.2-1 Example of Serial Programming Connection for MB90F523B (User Power Supply Used)
Note:

- Even if the SIN0, SOT0, and SCK0 pins are used for the user system, the controller shown in the figure above is required in the same as P00 "Figure 25.2-2 Control circuit". The /TICS signal of the flash microcomputer programmer can be used to disconnect the user circuit during serial programming.

- Connect the AF220/AF210/AF120/AF110 while the user power is off.

Figure 25.2-2 Control circuit
25.3 Connection Example in Single-chip Mode
(Writer Power Supply)

MD2 = "1" and MD0 = "0" will be set for mode pins MD2 and MD0 of the user system that has been set to single chip mode from the pins TAUX3 and TMODE of AF220/AF210/AF120/AF110, and flash serial writing mode will be adopted. An connection example using the writer power supply is given below.

- Connection Example in Single-chip Mode (Power Supplied from Flash microcomputer Programmer)

![Figure 25.3-1 Example of Serial Programming Connection for MB90F523B (Power Supplied from Flash Microcontroller Programmer)](image-url)
Note:  
- Even if the SIN0, SOT0, and SCK0 pins are used for the user system, the controller shown in the figure above is required in the same as P00 "Figure 25.3-2 Control circuit". The /TICS signal of the flash microcomputer programmer can be used to disconnect the user circuit during serial programming.
- Connect the AF220/AF210/AF120/AF110 while the user power is off.

Figure 25.3-2 Control circuit
25.4 Example of Minimum Connection to Flash Microcomputer Programmer (User Power Supply)

When each pin (MD2, MD0, and P00) is set as per Figure 25.4-1 during serial writing, MD2, MD0, and P00 and the flash microcomputer programmer do not need to be connected.

Example of Minimum Connection to Flash Microcontroller Programmer (User Power Supply Used)

**Figure 25.4-1 Example of Minimum Connection to Flash Microcontroller Programmer (User Power Supply Used)**

![Diagram of connection setup](image)

- **AF220/AF210/AF120/AF110 Flash microcontroller programmer**
- **User system**
- **Writing serial : "1"**
- **Writing serial : "0"**
- **User circuit**
- **Connector DX10-28S**
- **MB90F523B**
- **MD2**
- **MD1**
- **MD0**
- **X0**
- **X1**
- **P00**
- **P01**
- **RST**
- **SIN0**
- **SOT0**
- **SCK0**
- **Vcc**
- **Vss**
- **User power supply**
- **Pin Assignment of Connector (Hirose Electric)**

DX10-28S : light angle type

Pin Assignment of Connector (Hirose Electric)
Note:  
- Even if the SIN0, SOT0, and SCK0 pins are used for the user system, the controller shown in "Figure 25.4-2 Control circuit" above is required. The /TICS signal of the flash microcontroller programmer can be used to disconnect the user circuit during serial programming.
- Connect the AF220/AF210/AF120/AF110 while the user power is off.

Figure 25.4-2 Control circuit
25.5 Example of Minimum Connection to Flash Microcomputer Programmer (Writer Power Supply)

When each pin (MD2, MD0, and P00) is set as per Figure 25.5-1 during serial writing, MD2, MD0, and P00 and the flash microcomputer programmer do not need to be connected.

Example of Minimum Connection (Power Supplied from Flash Microcomputer Programmer)

Figure 25.5-1 Example of Minimum Connection (Power Supplied from Flash Microcomputer Programmer)
Note:  
- Even if the SIN0, SOT0, and SCK0 pins are used for the user system, the controller shown in "Figure 25.5-2 Control circuit" above is required. The/TICS signal of the flash microcomputer programmer can be used to disconnect the user circuit during serial programming.
- Connect the AF220/AF210/AF120/AF110 while the user power is off.
- Connect the AF220/AF210/AF120/AF110 while the user power is off.

![Figure 25.5-2 Control circuit](image)

### Supports serial writer

**Table 25.5-1 Supports serial writer**

<table>
<thead>
<tr>
<th>Serial writer</th>
<th>OVERVIEW</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF200 (made by Yokogawa Digital Computer Corporation)</td>
<td>Handy type serial writer for 1 unit</td>
</tr>
<tr>
<td>ITF2000 (made by Interface Co., Ltd.)</td>
<td>Serial gang writer that can simultaneously write up to 10 units</td>
</tr>
</tbody>
</table>
APPENDIX

APPENDIX A  Instruction
APPENDIX B  Register Index (address list)
APPENDIX C  Register Index (List by Peripheral Function)
APPENDIX D  Pin Function Index
APPENDIX E  Interrupt Vector Index
APPENDIX A Instruction

APPENDIX A describes the instructions used by the F² MC-16LX.

A.1 Instruction Types
A.2 Addressing
A.3 Direct Addressing
A.4 Indirect Addressing
A.5 Execution Cycle Count
A.6 Effective Address Field
A.7 How to Read the Instruction List
A.8 F²MC-16LX Instruction List
A.9 Instruction Map
The F^2MC-16LX supports 351 types of instructions.

### Instruction Types

- 41 transfer instructions (byte)
- 38 transfer instructions (word or long word)
- 42 Addition/subtraction Instructions (byte, word, long word)
- 12 Increment/decrement Instructions (byte, word, long word)
- 11 Compare Instructions (byte, word, long word)
- 11 Unsigned multiplication/division instructions (word, long word)
- 11 Signed Multiplication/Division Instructions (word, long word)
- 39 logic instructions (byte or word)
- 6 logic instructions (long word)
- Six Sign Inversion Instructions (byte, word)
- One Normalization Instruction (long word)
- 18 Shift Instructions (byte, word, long word)
- 50 branch instructions (branch command 1: 31 instructions, branch command 2: 19 instructions)
- Six Accumulator Operation Instructions (byte, word)
- 28 Other Control Instructions (byte, word, long word)
- 21 Bit Operand Instructions
- Ten String Instructions
A.2 Addressing

With the F²MC-16LX, the address format is determined by the instruction effective address field or the instruction code itself (implied). When the address format is determined by the instruction code itself, specify an address in accordance with the instruction code used. Some commands may be specified by several addressing methods.

### Addressing

The F²MC-16LX supports the following 23 types of addressing:

- Immediate (#imm)
- Register direct
- Direct branch address (addr16)
- Physical direct branch address (addr24)
- I/O direct (io)
- Abbreviated direct address (dir)
- Direct address (addr16)
- I/O direct bit address (io:bp)
- Abbreviated direct bit address (dir:bp)
- Direct bit address (addr16:bp)
- Vector address (#vct)
- Register indirect (@RWj  j = 0 to 3)
- Register indirect with post increment (@RWj +  j = 0 to 3)
- Register indirect with displacement (@RWi + disp8  i = 0 to 7, @RWj + disp16  j = 0 to 3)
- Long register indirect addressing with offset (@RLi + disp8  i = 0 to 3)
- Program counter indirect with displacement (@PC + disp16)
- Register indirect with base index (@RW0 + RW7, @RW1 + RW7)
- Program counter relative branch address (rel)
- Register list (rlst)
- Accumulator indirect (@A)
- Accumulator indirect branch address (@A)
- Indirectly-specified branch address (@ear)
- Indirectly-specified branch address (@eam)
## Effective Address Field

Table A.2-1 Effective Address Field lists the address formats specified by the effective address field.

Table A.2-1 Effective Address Field

<table>
<thead>
<tr>
<th>Code</th>
<th>Representation</th>
<th>Address format</th>
<th>Default bank</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>R0 RW0</td>
<td>RL0</td>
<td>None</td>
</tr>
<tr>
<td>01</td>
<td>R1 RW1</td>
<td>(RL0)</td>
<td>None</td>
</tr>
<tr>
<td>02</td>
<td>R2 RW2</td>
<td>RL1</td>
<td>None</td>
</tr>
<tr>
<td>03</td>
<td>R3 RW3</td>
<td>(RL1)</td>
<td>None</td>
</tr>
<tr>
<td>04</td>
<td>R4 RW4</td>
<td>RL2</td>
<td>None</td>
</tr>
<tr>
<td>05</td>
<td>R5 RW5</td>
<td>(RL2)</td>
<td>None</td>
</tr>
<tr>
<td>06</td>
<td>R6 RW6</td>
<td>RL3</td>
<td>None</td>
</tr>
<tr>
<td>07</td>
<td>R7 RW7</td>
<td>(RL3)</td>
<td>None</td>
</tr>
<tr>
<td>08</td>
<td>@RW0</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>09</td>
<td>@RW1</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>0A</td>
<td>@RW2</td>
<td></td>
<td>ADB</td>
</tr>
<tr>
<td>0B</td>
<td>@RW3</td>
<td></td>
<td>SPB</td>
</tr>
<tr>
<td>0C</td>
<td>@RW0+</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>0D</td>
<td>@RW1+</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>0E</td>
<td>@RW2+</td>
<td></td>
<td>ADB</td>
</tr>
<tr>
<td>0F</td>
<td>@RW3+</td>
<td></td>
<td>SPB</td>
</tr>
<tr>
<td>10</td>
<td>@RW0+disp8</td>
<td>With 8-bit displacement</td>
<td>DTB</td>
</tr>
<tr>
<td>11</td>
<td>@RW1+disp8</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>12</td>
<td>@RW2+disp8</td>
<td></td>
<td>ADB</td>
</tr>
<tr>
<td>13</td>
<td>@RW3+disp8</td>
<td></td>
<td>SPB</td>
</tr>
<tr>
<td>14</td>
<td>@RW4+disp8</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>15</td>
<td>@RW5+disp8</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>16</td>
<td>@RW6+disp8</td>
<td></td>
<td>ADB</td>
</tr>
<tr>
<td>17</td>
<td>@RW7+disp8</td>
<td></td>
<td>SPB</td>
</tr>
<tr>
<td>18</td>
<td>@RW0+disp16</td>
<td>With 16-bit displacement</td>
<td>DTB</td>
</tr>
<tr>
<td>19</td>
<td>@RW1+disp16</td>
<td></td>
<td>DTB</td>
</tr>
<tr>
<td>1A</td>
<td>@RW2+disp16</td>
<td></td>
<td>ADB</td>
</tr>
<tr>
<td>1B</td>
<td>@RW3+disp16</td>
<td></td>
<td>SPB</td>
</tr>
<tr>
<td>1C</td>
<td>@RW0+RW7</td>
<td>Register indirect with index</td>
<td>DTB</td>
</tr>
<tr>
<td>1D</td>
<td>@RW1+RW7</td>
<td>Register indirect with index</td>
<td>DTB</td>
</tr>
<tr>
<td>1E</td>
<td>@PC+disp16</td>
<td>PC indirect with 16-bit displacement</td>
<td>PCB</td>
</tr>
<tr>
<td>1F</td>
<td>addr16</td>
<td>Direct address</td>
<td>DTB</td>
</tr>
</tbody>
</table>
A.3 Direct Addressing

An operand value, register, or address is specified explicitly in direct addressing mode.

- Direct Addressing
  - Immediate (#imm)
    Specify an operand value explicitly. (#imm4/#imm8/#imm16/#imm32)
    Figure A.3-1 shows the an example.

  ![Figure A.3-1 Immediate example (#imm)]

    | MOVW   A, #01212H (Instruction storing value of operand to A) |
    |-------------------------------------------------------------|
    | Before execution   A 2 2 3 3 : 4 4 5 5                     |
    | After execution    A 4 4 5 5 : 1 2 1 2                     |
    | (AL→AH transmission executes by some instructions)         |

  - Register direct
    Specify a register explicitly as an operand. Table A.3-1 Direct Addressing Registers lists the registers that can be specified.

  ![Table A.3-1 Register direct]

<table>
<thead>
<tr>
<th>General-purpose Register</th>
<th>Byte</th>
<th>Word</th>
<th>Long word</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0, R1, R2, R3, R4, R5, R6, R7</td>
<td>RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7</td>
<td>RL0, RL1, RL2, RL3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dedicated Registers</th>
<th>Accumulator</th>
<th>Pointer</th>
<th>Bank</th>
<th>Page</th>
<th>control</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, AL</td>
<td>SP*</td>
<td>PCB, DTB, USB, SSB, ADB</td>
<td>DPR</td>
<td>PS, CCR, RP, ILM</td>
<td></td>
</tr>
</tbody>
</table>

*: Either the user stack pointer (USP) or system stack pointer (SSP) is selected and used depending on the value of the S flag bit in the condition code register (CCR). In branch instructions, the program counter (PC) is specified without being defined as an instruction operand.
Figure A.3-2 shows an example.

**Figure A.3-2 Example of Register Direct Addressing**

| MOV   R0, A (Instruction transmitting lower 8-bit of A to general purpose register R0) |
|-------|---------------------------------|
| Before execution | A | 0 7 1 6 : 2 5 3 4 |
| After execution | A | 0 7 1 6 : 2 5 6 4 |

Memory space

R0

Direct branch address (addr16)

Specify an offset explicitly for the branch destination address. The size of the offset is 16 bits, which indicates the branch destination in the logical address space. Direct branch addressing is used for an unconditional branch, subroutine call, or software interrupt instruction. Bits 23 to 16 of the address are specified by the program bank register (PCB). Figure A.3-3 shows an example.

**Figure A.3-3 Example of Direct Branch Addressing (addr16)**

<table>
<thead>
<tr>
<th>JMP 3B20H (Instruction executing unconditional branch with direct branch address specification in bank)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before execution</td>
</tr>
<tr>
<td>After execution</td>
</tr>
</tbody>
</table>

Memory space

JMP 3B20H

6F3B20H

4F3B20H

4F3C20H

4F3C21H

4F3C22H
Physical direct branch address (addr24)

Specify an offset explicitly for the branch destination address. The data length of the displacement (offset) is 24 bits. Direct branch addressing is used for an unconditional branch, subroutine call, or software interrupt instruction. Figure A.3-4 shows an example.

Figure A.3-4 Example of Direct Branch Addressing (addr24)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>PC</th>
<th>PCB</th>
<th>After execution</th>
<th>PC</th>
<th>PCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>JMPP 333B20H</td>
<td>3 C 2 0</td>
<td>4 F</td>
<td>JMPP 333B20H</td>
<td>3 B 2 0</td>
<td>3 3</td>
</tr>
<tr>
<td>Memory space</td>
<td>333B20H</td>
<td>Next instruction</td>
<td>Memory space</td>
<td>4F3C20H</td>
<td>6 3</td>
</tr>
<tr>
<td></td>
<td>4F3C21H</td>
<td>2 0</td>
<td></td>
<td>4F3C21H</td>
<td>3 B</td>
</tr>
<tr>
<td></td>
<td>4F3C22H</td>
<td>3 3</td>
<td></td>
<td>4F3C23H</td>
<td>3 3</td>
</tr>
</tbody>
</table>

I/O direct (io)

Specify an 8-bit offset explicitly for the memory address in an operand. The I/O address space in the physical address space from 000000H to 0000FFH is accessed regardless of the data bank register (DTB) and direct page register (DPR). A bank select prefix for bank addressing is invalid if specified before an instruction using I/O direct addressing. Figure A.3-5 shows an example.

Figure A.3-5 Example of I/O Direct Addressing (io)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>A</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVW A, i : 0C0H</td>
<td>0 7 1 6 : 2 5 3 4</td>
<td>0000C0H E E</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0000C1H F F</td>
</tr>
<tr>
<td>After execution</td>
<td>A</td>
<td>Memory space</td>
</tr>
<tr>
<td></td>
<td>2 5 3 4 : F F E E</td>
<td>0000C0H E E</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0000C1H F F</td>
</tr>
</tbody>
</table>
Abbreviated direct address (dir)

Specify the eight low-order bits of a memory address explicitly in an operand. Address bits 8 to 15 are specified by the direct page register (DPR). Address bits 16 to 23 are specified by the data bank register (DTB). Figure A.3-6 shows an example.

**Figure A.3-6 Example of Abbreviated Direct Addressing (dir)**

![Example of Abbreviated Direct Addressing (dir)]

Direct address (addr16)

Specify the 16 low-order bits of a memory address explicitly in an operand. Address bits 16 to 23 are specified by the data bank register (DTB). A prefix instruction for access space addressing is invalid for this mode of addressing. Figure A.3-7 shows an example.

**Figure A.3-7 Example of Direct Addressing (addr16)**

![Example of Direct Addressing (addr16)]
I/O direct bit address (io:bp)

Specify bits in physical addresses 000000_H to 0000FF_H explicitly. Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB). Figure A.3-8 shows an example.

**Figure A.3-8 Example of I/O Direct Bit Addressing (io:bp)**

| SETB i : 0C1H : 0 (Instruction executing set bit by I/O direct bit address specification) |
| Memory space |
| Before execution | 0000C1H | 0 | 0 |
| After execution | 0000C1H | 0 | 1 |

Abbreviated direct bit address (dir:bp)

Specify the eight low-order bits of a memory address explicitly in an operand. Address bits 8 to 15 are specified by the direct page register (DPR). Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB). Figure A.3-9 shows an example.

**Figure A.3-9 Example of Abbreviated Direct Bit Addressing (dir:bp)**

| SETB S:10H:0 (This instruction sets bits by abbreviated direct bit addressing.) |
| Memory space |
| Before execution | DTB 5 5 | DPR 6 6 | 556610H | 0 | 0 |
| After execution | DTB 5 5 | DPR 6 6 | 556610H | 0 | 1 |
● Direct bit address (addr16:bp)

Specify arbitrary bits in 64 Kbytes explicitly. Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB). Figure A.3-10 shows an example.

**Figure A.3-10 Example of Direct Bit addressing (addr16:bp)**

| SETB 2222H : 0 (Instruction executing set bit by direct bit address specification) |
| Memory space | Before execution | DTB | 5 5 |
| | | 552222H | 0 0 |
| Memory space | After execution | DTB | 5 5 |
| | | 552222H | 0 1 |

● Vector address (#vct)

Specify vector data in an operand to indicate the branch destination address. There are two sizes for vector numbers: 4 bits and 8 bits. Vector addressing is used for a subroutine call or software interrupt instruction. Figure A.3-11 shows an example.

**Figure A.3-11 Example of Vector Addressing (#vct)**

| CALLV #15 (This instruction causes a branch to the address indicated by the interrupt vector specified in an operand.) |
| Memory space | Before execution | PC | 0 0 0 0 |
| | PCB | FF |
| After execution | PC | D 0 0 0 |
| | PCB | FF |
### Table A.3-2 CALLV Vector List

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Vector address L</th>
<th>Vector address H</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLV #0</td>
<td>XXFFFE_{H}</td>
<td>XXFFFFFF_{H}</td>
</tr>
<tr>
<td>CALLV #1</td>
<td>XXFFFFC_{H}</td>
<td>XXFFFFD_{H}</td>
</tr>
<tr>
<td>CALLV #2</td>
<td>XXFFFFA_{H}</td>
<td>XXFFFFB_{H}</td>
</tr>
<tr>
<td>CALLV #3</td>
<td>XXFFFF8_{H}</td>
<td>XXFFFF9_{H}</td>
</tr>
<tr>
<td>CALLV #4</td>
<td>XXFFFF6_{H}</td>
<td>XXFFFF7_{H}</td>
</tr>
<tr>
<td>CALLV #5</td>
<td>XXFFFF4_{H}</td>
<td>XXFFFF5_{H}</td>
</tr>
<tr>
<td>CALLV #6</td>
<td>XXFFFF2_{H}</td>
<td>XXFFFF3_{H}</td>
</tr>
<tr>
<td>CALLV #7</td>
<td>XXFFFF0_{H}</td>
<td>XXFFFF1_{H}</td>
</tr>
<tr>
<td>CALLV #8</td>
<td>XXFFEE{H}</td>
<td>XXFFEF{H}</td>
</tr>
<tr>
<td>CALLV #9</td>
<td>XXFFEC{H}</td>
<td>XXFFED{H}</td>
</tr>
<tr>
<td>CALLV #10</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
<tr>
<td>CALLV #11</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
<tr>
<td>CALLV #12</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
<tr>
<td>CALLV #13</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
<tr>
<td>CALLV #14</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
<tr>
<td>CALLV #15</td>
<td>XXFFE{AH}</td>
<td>XXFFE{BH}</td>
</tr>
</tbody>
</table>

Note: A PCB register value is set in XX.

---

**Note:** When the program bank register (PCB) contains FF_{H}, the vector area overlaps the vector area of INT #vct8 (#0 to #7). Use vector addressing carefully. See Table A.3-2.
A.4 Indirect Addressing

In indirect addressing mode, an address is specified indirectly by the address data of an operand.

Indirect Addressing

- Register indirect (@RWj  j = 0 to 3)

Memory is accessed using the contents of general-purpose register RWj as an address. When using RW0 or RW1, bit23 to bit16 of address are shown data bank register (DTB). When using RW3, bit23 to bit16 of address are shown system stack bank register (SSB) or user stack bank register (USB). And when using RW2, bit23 to bit16 of address are shown additional data bank register (ADB). Figure A.4-1 shows an example.

Figure A.4-1 Example of Register Indirect Addressing (@RWj j = 0 to 3)

- Register indirect with post increment (@RWj +  j = 0 to 3)

Addressing for access with memory using the contents of the general-purpose register RWj as the address. After the operation of the operand, RWj has the operand data length (1 for a byte, 2 for a word and 4 for a long word) added.

When using RW0 or RW1, bit23 to bit16 of address are shown data bank register (DTB). When using RW3, bit23 to bit16 of address are shown system stack bank register (SSB) or user stack bank register (USB). And when using RW2, bit23 to bit16 of address are shown additional data bank register (ADB).

If the post increment results in the address of the register that specifies the increment, the incremented value is referenced after that. In this case, if the next instruction is a write instruction, priority is given to writing by an instruction and, the
Figure A.4-2 shows an example.

**Figure A.4-2 Example of Register Indirect Addressing with Post Increment (@RWj + j = 0 to 3)**

- **Register indirect with displacement (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)**

  Addressing for access with memory using the contents of the general-purpose register RWj added with the displacement as the address. Two types of offset, byte and word offsets, are used. They are added as signed numeric values. When using RW0, RW1, RW4 or RW5, bit23 to bit16 of address are shown data bank register (DTB). When using RW3 or RW7, bit23 to bit16 of address are shown system stack bank register (SSB) or user stack bank register (USB). And when using RW2 or RW6, bit23 to bit16 of address are shown additional data bank register (ADB). Figure A.4-3 shows an example.

**Figure A.4-3 Example of Register Indirect Addressing with Offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)**

MOVW A, @RW1+ (Instruction reading by register indirection with post increment and storing to A)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>RW1</td>
</tr>
<tr>
<td>0716:2534</td>
<td>D311</td>
</tr>
<tr>
<td>DTB</td>
<td>DTB</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>78D30FH</td>
<td>78D310H</td>
</tr>
<tr>
<td>78D31FH</td>
<td>78D320H</td>
</tr>
</tbody>
</table>

Memory space

MOVW A, @RW1+10H (Instruction reading by register indirection with displacement and storing to A)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW1</td>
<td>RW1</td>
</tr>
<tr>
<td>0716:2534</td>
<td>D30F</td>
</tr>
<tr>
<td>DTB</td>
<td>DTB</td>
</tr>
<tr>
<td>(+10H)</td>
<td>E</td>
</tr>
<tr>
<td>78D31FH</td>
<td>78D320H</td>
</tr>
<tr>
<td>78D310H</td>
<td>78D321H</td>
</tr>
</tbody>
</table>

Memory space
Long register indirect addressing with offset (@RLi + disp8 \( i = 0 \text{ to } 3 \))

Addressing for access by memory using the low 24 bits? the result of adding the displacement to the contents of the general-purpose register RLi as the address. The offset is 8-bits long and is added as a signed numeric value. Figure A.4-4 shows an example.

**Figure A.4-4 Example of Long Register Indirect Addressing with Offset (@RLi + disp8 \( i = 0 \text{ to } 3 \))**

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A ( 0 7 1 6 : 2 5 3 4 )</td>
<td>A ( 2 5 3 4 : F F F F )</td>
</tr>
<tr>
<td>RL2 ( F 3 8 2 : 4 B 0 2 )</td>
<td>RL2 ( F 3 8 2 : 4 B 0 2 )</td>
</tr>
<tr>
<td>Memory space</td>
<td>Memory space</td>
</tr>
<tr>
<td>824B27H E E</td>
<td>824B28H F F</td>
</tr>
</tbody>
</table>

Program counter indirect with displacement (@PC + disp16)

Memory is accessed using the address indicated by (instruction address + 4 + disp16). The displacement is one word long. Address bits 16 to 23 are specified by the program bank register (PCB). Note that the operand address of each of the following instructions is not deemed to be (next instruction address + disp16):
- DBNZ eam, rel
- CBNE eam, #imm8, rel
- MOV eam, #imm8
- DWBNZ eam, rel
- CWBNE eam, #imm16, rel
- MOVW eam, #imm16

Figure A.4-5 shows an example.

**Figure A.4-5 Example of Program Counter Indirect Addressing with Offset (@PC + disp16)**

<table>
<thead>
<tr>
<th>Before execution</th>
<th>After execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A ( 0 7 1 6 : 2 5 3 4 )</td>
<td>A ( 2 5 3 4 : F F F F )</td>
</tr>
<tr>
<td>PCB C5</td>
<td>PCB C5</td>
</tr>
<tr>
<td>Memory space</td>
<td>Memory space</td>
</tr>
<tr>
<td>54556H 7 3</td>
<td>54557H 9 E</td>
</tr>
<tr>
<td>54558H 2 0</td>
<td>54559H 0 0</td>
</tr>
<tr>
<td>MOVW A, @PC+20H</td>
<td>MOVW A, @PC+20H</td>
</tr>
<tr>
<td>5455AH</td>
<td></td>
</tr>
<tr>
<td>+20H</td>
<td>+20H</td>
</tr>
<tr>
<td>5457AH E E</td>
<td>5457BH F F</td>
</tr>
</tbody>
</table>
Register indirect with base index (@RW0 + RW7, @RW1 + RW7)

Addressing for access by memory using the contents of general-purpose register RW7 added with RW0 or RW1 as an address. Address bits 16 to 23 are indicated by the data bank register (DTB). Figure A.4-6 shows an example.

Figure A.4-6 Example of Register Indirect Addressing with Base Index (@RW0 + RW7, @RW1 + RW7)

Program counter relative branch address (rel)

The address of the branch destination is a value determined by adding an 8-bit offset to the program counter (PC) value. If the result of addition exceeds 16 bits, bank register incrementing or decrementing is not performed and the excess part is ignored, and therefore the address is contained within a 64-Kbyte bank. This addressing is used for both conditional and unconditional branch instructions. Address bits 16 to 23 are indicated by the program bank register (PCB). Figure A.4-7 shows an example.

Figure A.4-7 Example of Program Counter Relative Branch Addressing (rel)
Register list (rlst)

Specify a register to be pushed onto or popped from a stack. Figure A.4-8 shows the configuration of the register list; Figure A.4-9 gives an example of the register list.

**Figure A.4-8 Configuration of the Register List**

<table>
<thead>
<tr>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW7</td>
<td>RW6</td>
</tr>
<tr>
<td>RW5</td>
<td>RW4</td>
</tr>
<tr>
<td>RW3</td>
<td>RW2</td>
</tr>
<tr>
<td>RW1</td>
<td>RW0</td>
</tr>
</tbody>
</table>

A register is selected when the corresponding bit is 1 and deselected when the bit is 0.

MSB: The uppermost bit
LSB: The lowest bit

**Figure A.4-9 Example of Register List (rlist)**

POPW RW0, RW4 (This instruction transfers memory data indicated by the SP to multiple word registers indicated by the register list.)

<table>
<thead>
<tr>
<th>SP</th>
<th>Memory space</th>
<th>SP</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 4 F A</td>
<td>RW0</td>
<td>0 2</td>
<td>0 1</td>
</tr>
<tr>
<td>RW0</td>
<td>x x; x x</td>
<td>RW0</td>
<td>0 2</td>
</tr>
<tr>
<td>RW1</td>
<td>x x; x x</td>
<td>RW1</td>
<td>x x; x x</td>
</tr>
<tr>
<td>RW2</td>
<td>x x; x x</td>
<td>RW2</td>
<td>x x; x x</td>
</tr>
<tr>
<td>RW3</td>
<td>x x; x x</td>
<td>RW3</td>
<td>x x; x x</td>
</tr>
<tr>
<td>RW4</td>
<td>x x; x x</td>
<td>RW4</td>
<td>0 4</td>
</tr>
<tr>
<td>RW5</td>
<td>x x; x x</td>
<td>RW5</td>
<td>x x; x x</td>
</tr>
<tr>
<td>RW6</td>
<td>x x; x x</td>
<td>RW6</td>
<td>x x; x x</td>
</tr>
<tr>
<td>RW7</td>
<td>x x; x x</td>
<td>RW7</td>
<td>x x; x x</td>
</tr>
</tbody>
</table>

Before execution

<table>
<thead>
<tr>
<th>SP</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>34FAH</td>
<td>34FBH</td>
</tr>
<tr>
<td>34FCH</td>
<td>34FDH</td>
</tr>
</tbody>
</table>

After execution

<table>
<thead>
<tr>
<th>SP</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>34FAH</td>
<td>34FBH</td>
</tr>
<tr>
<td>34FCH</td>
<td>34FDH</td>
</tr>
<tr>
<td>34FEH</td>
<td>SP</td>
</tr>
</tbody>
</table>
● Accumulator indirect (@A)

Memory is accessed using the address indicated by the contents of the low-order bytes (16 bits) of the accumulator (AL). Address bits 16 to 23 are specified by a mnemonic in the data bank register (DTB). Figure A.3-1 shows an example. Figure A.4-10 shows an example.

**Figure A.4-10 Example of Accumulator Indirect Addressing (@A)**

![Diagram showing Accumulator Indirect Addressing (@A)]

- MOVW A, @A (Instruction read by accumulator indirection and storing to A)
  - Before execution:
    - A: 0716:2534
    - DTB: BB
  - After execution:
    - A: 0716:FFFE
    - DTB: BB

● Accumulator indirect branch address (@A)

The address of the branch destination is the content (16 bits) of the low-order bytes (AL) of the accumulator. It indicates the branch destination in the bank address space. Address bits 16 to 23 are specified by the program bank register (PCB). For the Jump Context (JCTX) instruction, however, address bits 16 to 23 are specified by the data bank register (DTB). This addressing is used for unconditional branch instructions. Figure A.4-11 shows an example.

**Figure A.4-11 Example of Accumulator Indirect Branch Addressing (@A)**

![Diagram showing Accumulator Indirect Branch Addressing (@A)]

- JMP @A (Instruction executing unconditional branch with accumulator indirection branch address)
  - Before execution:
    - PC: 3C20
    - PCB: 4F
    - A: 6677:3B20
  - After execution:
    - PC: 3B20
    - PCB: 4F
    - A: 6677:3B20
    - Memory space: 4F3B20H
    - Next instruction: 4F3C20H
• Indirectly-specified branch address (@ear)

The address of the branch destination is the word data at the address indicated by ear. Figure A.4-12 shows an example.

**Figure A.4-12 Example of Indirect Specification Branch Addressing (@ear)**

JMP @@RW0 (Instruction executing unconditional branch with indirection of register indirection)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>3 C 2 0</td>
</tr>
<tr>
<td>RW0</td>
<td>7 F 4 8</td>
</tr>
<tr>
<td></td>
<td>217F48H</td>
</tr>
<tr>
<td></td>
<td>4F3B20H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>After execution</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>3 B 2 0</td>
</tr>
<tr>
<td>RW0</td>
<td>7 F 4 8</td>
</tr>
<tr>
<td></td>
<td>4F3C20H</td>
</tr>
</tbody>
</table>

• Indirectly-specified branch address (@eam)

The address of the branch destination is the word data at the address indicated by eam. Figure A.4-13 shows an example.

**Figure A.4-13 Example of Indirect Specification Branch Addressing (@eam)**

JMP @RW0 (This instruction causes an unconditional branch by register indirect addressing.)

<table>
<thead>
<tr>
<th>Before execution</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>3 C 2 0</td>
</tr>
<tr>
<td>RW0</td>
<td>3 B 2 0</td>
</tr>
<tr>
<td></td>
<td>4F3B20H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>After execution</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>3 B 2 0</td>
</tr>
<tr>
<td>RW0</td>
<td>3 B 2 0</td>
</tr>
<tr>
<td></td>
<td>4F3C20H</td>
</tr>
<tr>
<td></td>
<td>7 3</td>
</tr>
</tbody>
</table>
A.5 Execution Cycle Count

The number of cycles required for instruction execution (execution cycle count) is obtained by adding the number of cycles required for each instruction, correction value determined by the condition, and the number of cycles for instruction fetch.

■ Execution Cycle Count

In the mode of fetching an instruction from memory such as internal ROM connected to a 16-bit bus, the program fetches the instruction being executed in word increments. Therefore, intervening in data access increases the execution cycle count.

Similarly, in the mode of fetching an instruction from memory connected to an 8-bit external bus, the program fetches every byte of an instruction being executed. Therefore, intervening in data access increases the execution cycle count.

In CPU intermittent operation mode, access to a general-purpose register, internal ROM, internal RAM, internal I/O, or external data bus causes the clock to the CPU to halt for the cycle count specified by the CG0 and CG1 bits of the low power consumption mode control register. Therefore, for the cycle count required for instruction execution in CPU intermittent operation mode, add the “access count × cycle count for the halt” as a correction value to the normal execution count.

■ Calculating the Execution Cycle Count

Table A.5-1, Table A.5-2 and Table A.5-3 show the data of execution cycle counts and correction values.

Table A.5-1 Execution Cycle Counts in Each Addressing Mode

<table>
<thead>
<tr>
<th>Code</th>
<th>Operand</th>
<th>(a)*</th>
<th>Register access count to each addressing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Execution Cycle Count to each addressing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00 to 07</td>
<td>Ri, RWi, RLi</td>
<td>See the instruction list.</td>
<td>See the instruction list.</td>
</tr>
<tr>
<td>08 to 0B</td>
<td>@RWj</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>0C to 0F</td>
<td>@RWj+</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>10 to 17</td>
<td>@RWi+disp8</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>18 to 1B</td>
<td>@RWi+disp16</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>@RW0+RW7</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1D</td>
<td>@RW1+RW7</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1E</td>
<td>@PC+disp16</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1F</td>
<td>addr16</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*: (a) is used for (cycle count) and B (correction value) in "A.8 F2MC-16LX Instruction List".
Table A.5-2  Cycle Count Correction Values for Counting Execution Cycles

<table>
<thead>
<tr>
<th>Operand</th>
<th>(b) byte *1</th>
<th>(c) word *1</th>
<th>(d) long *1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cycle count</td>
<td>Access count</td>
<td>Cycle count</td>
</tr>
<tr>
<td>Internal register</td>
<td>+0</td>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>Internal memory even-numbered address</td>
<td>+0</td>
<td>1</td>
<td>+0</td>
</tr>
<tr>
<td>Internal memory odd-numbered address</td>
<td>+0</td>
<td>1</td>
<td>+2</td>
</tr>
<tr>
<td>External data bus*2 16-bit even-numbered address</td>
<td>+1</td>
<td>1</td>
<td>+1</td>
</tr>
<tr>
<td>External data bus*2 16-bit odd-numbered address</td>
<td>+1</td>
<td>1</td>
<td>+4</td>
</tr>
<tr>
<td>External data bus*2 8bit</td>
<td>+1</td>
<td>1</td>
<td>+4</td>
</tr>
</tbody>
</table>

*1: (b), (c), and (d) are used for (cycle count) and B (correction value) in A.8 F2MC-16LX Instruction List.
*2: When an external data bus is used, the number of cycles during which an instruction is made to wait by ready-signal input or automatic ready must also be added.

Table A.5-3  Cycle Count Correction Values for Counting Instruction Fetch Cycles

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Byte boundary</th>
<th>Word boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal memory</td>
<td>-</td>
<td>+2</td>
</tr>
<tr>
<td>External data bus 16-bits</td>
<td>-</td>
<td>+3</td>
</tr>
<tr>
<td>External data bus 8-bits</td>
<td>+3</td>
<td>-</td>
</tr>
</tbody>
</table>

Notes:
- When an external data bus is used, the number of cycles during which an instruction is made to wait by ready-signal input or automatic ready must also be added.
- As every instruction fetch does not delay instruction execution, in practice, the correction values should be used to calculate the worst case.
A.6 Effective Address Field

Table A.6-1 shows effective address field.

### Effective Address Field

<table>
<thead>
<tr>
<th>Code</th>
<th>Representation</th>
<th>Address format</th>
<th>Number of byte in address extended part</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>R0 RW0 RL0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>R1 RW1 RL0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>R2 RW2 RL1</td>
<td>Register direct</td>
<td>In sequential order from the left, &quot;ea&quot; indicates</td>
</tr>
<tr>
<td>03</td>
<td>R3 RW3 RL1</td>
<td>• Byte</td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>R4 RW4 RL2</td>
<td>• Word</td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>R5 RW5 RL2</td>
<td>• Long word</td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>R6 RW6 RL3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>R7 RW7 RL3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>@RW0</td>
<td>Register indirect</td>
<td>0</td>
</tr>
<tr>
<td>09</td>
<td>@RW1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0A</td>
<td>@RW2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0B</td>
<td>@RW3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0C</td>
<td>@RW0+</td>
<td>With post increment</td>
<td>0</td>
</tr>
<tr>
<td>0D</td>
<td>@RW1+</td>
<td>Register indirect</td>
<td></td>
</tr>
<tr>
<td>0E</td>
<td>@RW2+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0F</td>
<td>@RW3+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>@RW0+disp8</td>
<td>With 8-bit displacement</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>@RW1+disp8</td>
<td>Register indirect</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>@RW2+disp8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>@RW3+disp8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>@RW4+disp8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>@RW5+disp8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>@RW6+disp8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>@RW7+disp8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table A.6-1  Effective Address Field (2 / 2)

<table>
<thead>
<tr>
<th>Code</th>
<th>Representation</th>
<th>Address format</th>
<th>Number of byte in address extended part*</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>@RW0+disp16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>@RW1+disp16</td>
<td>With 16-bit displacement Register indirect</td>
<td>2</td>
</tr>
<tr>
<td>1A</td>
<td>@RW2+disp16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IB</td>
<td>@RW3+disp16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1C</td>
<td>@RW0+RW7</td>
<td>Register indirect with index</td>
<td>0</td>
</tr>
<tr>
<td>1D</td>
<td>@RW1+RW7</td>
<td>Register indirect with index</td>
<td>0</td>
</tr>
<tr>
<td>1E</td>
<td>@PC+disp16</td>
<td>PC indirect with 16-bit displacement</td>
<td>2</td>
</tr>
<tr>
<td>1F</td>
<td>addr16</td>
<td>Direct address</td>
<td>2</td>
</tr>
</tbody>
</table>

*: Each byte count of the extended address part applies to + in the # (byte count) column in A.8 F2MC-16LX Instruction List
A.7 How to Read the Instruction List

Table A.7-1 Description of Items in the Instruction List describes the items used in the F^2MC-16LX Instruction List, and Table A.7-2 Explanation on Symbols in the Instruction List describes the symbols used in the same list.

Description of instruction presentation items and symbols

Table A.7-1 Description of Items in the Instruction List

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mnemonic</td>
<td>Uppercase, symbol: Represented as is in the assembler. Lowercase: Rewritten in the assembler. Number of following lowercase: Indicates bit length in the instruction.</td>
</tr>
<tr>
<td>#</td>
<td>Indicates the number of bytes.</td>
</tr>
<tr>
<td>~</td>
<td>Indicates the number of cycles.</td>
</tr>
<tr>
<td>RG</td>
<td>Indicates the number of times a register access is performed during instruction execution. The number is used to calculate the correction value for CPU intermittent operation.</td>
</tr>
<tr>
<td>B</td>
<td>Indicates the correction value used to calculate the actual number of cycles during instruction execution. The actual number of cycles during instruction execution can be determined by adding the value in the ~ column to this value.</td>
</tr>
<tr>
<td>Operation</td>
<td>Indicates the instruction operation.</td>
</tr>
<tr>
<td>LH</td>
<td>Indicates the special operation for bits 15 to 08 of the accumulator. Z: Transfers 0. X: Transfers after sign extension. --: No transfer</td>
</tr>
<tr>
<td>AH</td>
<td>Indicates the special operation for the 16 high-order bits of the accumulator. *: Transfers from AL to AH. --: No transfer Z: Transfers 00H to AH. X: Transfers 00H or FFH to AH after AL sign extension.</td>
</tr>
<tr>
<td>I</td>
<td>I (interrupt enable), S (stack), T (sticky bit).</td>
</tr>
<tr>
<td>S</td>
<td>N (negative), Z (zero), V (overflow), C (carry).</td>
</tr>
<tr>
<td>T</td>
<td>Each indicates the state of each flag:</td>
</tr>
<tr>
<td>N</td>
<td>*: Changes upon instruction execution.</td>
</tr>
<tr>
<td>Z</td>
<td>--: No change</td>
</tr>
<tr>
<td>V</td>
<td>Z: Set upon instruction execution.</td>
</tr>
<tr>
<td>C</td>
<td>X: Reset upon instruction execution.</td>
</tr>
<tr>
<td>RMW</td>
<td>Indicates whether the instruction is a Read-modify-write instruction (reading data from memory by the I instruction and writing the result to memory). *: Read-Modify-Write instruction --: Not Read-Modify-Write instruction</td>
</tr>
</tbody>
</table>

Note: Cannot be used for an address that has different meanings between read and write operations.
Table A.7-2 Explanation on Symbols in the Instruction List (1 / 2)

<table>
<thead>
<tr>
<th>Representation</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| A              | 32 bits accumulator  
The bit length used varies depending on each instruction.  
Byte : 8 low-order bit of AL  
Word : 16-bit of AL  
Long : 32bit of AL: AH |
| AH             | 16 high-order bits of A |
| AL             | 16 low-order bits of A |
| SP             | Stack pointer (USP or SSP) |
| PC             | Program counter |
| PCB            | Program bank register |
| DTB            | Data bank register |
| ADB            | Additional data bank register |
| SSB            | System stack bank register |
| USB            | User stack bank register |
| SPB            | Current stack bank register (SSB or USB) |
| DPR            | Direct page register |
| brg1           | DTB, ADB, SSB, USB, DPR, PCB, SPB |
| brg2           | DTB, ADB, SSB, USB, DPR, SPB |
| Ri             | R0, R1, R2, R3, R4, R5, R6, R7 |
| RWi            | RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7 |
| RWj            | RW0, RW1, RW2, RW3 |
| RLi            | RL0, RL1, RL2, RL3 |
| dir            | Specifying of Abbreviated direct addressing |
| addr16         | Specifying of Direct addressing |
| addr24         | Specifying of Physical direct addressing |
| ad24 0-15      | Bits 0 to 15 of addr24 |
| ad24 16-23     | Bits 16 to 23 of addr24 |
| io             | I/O area (000000H to 0000FFH) |
| #imm4          | 4-bit immediate data |
| #imm8          | 8-bit immediate data |
| #imm16         | 16-bit immediate data |
| #imm32         | 32-bit immediate data |
| ext (imm8)     | 16-bit data obtained by sign extension of 8-bit immediate data |
| disp8          | 8-bit displacement |
| disp16         | 16-bit displacement |
| bp             | Bit offset value |
| vct4           | Vector number (0 to 15) |
Table A.7-2  Explanation on Symbols in the Instruction List  (2 / 2)

<table>
<thead>
<tr>
<th>Representation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>vct8</td>
<td>Vector number (0 to 255)</td>
</tr>
<tr>
<td>( )b</td>
<td>Bit address</td>
</tr>
<tr>
<td>rel</td>
<td>Specifying of PC relative branch</td>
</tr>
<tr>
<td>ear</td>
<td>Specifying of Effective addressing (code 00 to 07)</td>
</tr>
<tr>
<td>eam</td>
<td>Specifying of Effective addressing (code 08 to 1F)</td>
</tr>
<tr>
<td>rlst</td>
<td>Register list</td>
</tr>
</tbody>
</table>
# F\(^2\)MC-16LX Instruction List

The table lists the instructions used by the F\(^2\)MC-16LX family.

## F\(^2\)MC-16LX Instruction List

### Table A.8-1  41 Transfer instructions (byte)

| Mnemonic          | #  | ~   | RG | B  | Operation | L H | A H | I   | S   | T   | N   | Z   | V   | C   | R   | M | W   |
|-------------------|----|-----|----|----|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|
| MOV A,dir         | 2  | 3   | 0  | (b) byte (A) <- (dir) | Z  | *   | *   | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,addr16      | 3  | 4   | 0  | (b) byte (A) <- (addr16) | Z  | *   | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,Ri          | 1  | 2   | 1  | 0   byte (A) <- (Ri) | Z  | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,ear         | 2  | 2   | 1  | 0   byte (A) <- (ear) | Z  | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,eam         | 2+ | 3 + (a) | 0 | (b) byte (A) <- (eam) | Z  | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,io          | 2  | 3   | 0  | (b) byte (A) <- (io) | Z  | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,#imm8       | 2  | 2   | 0  | 0   byte (A) <- (imm8) | X  | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOV A,Ri+disp8    | 2  | 3   | 0  | (b) byte (A) <- (Ri)+disp8 | X  | *   | *   | *   | *   | *   | *   | *   | *   | *   |
| MOVX A,dir        | 2  | 3   | 0  | (b) byte (A) <- (dir) | X  | *   | *   | *   | *   | *   | *   | *   | *   |
| MOVX A,addr16     | 3  | 4   | 0  | (b) byte (A) <- (addr16) | X  | *   | *   | *   | *   | *   | *   | *   | *   |
| MOVX A,Ri         | 2  | 2   | 1  | 0   byte (A) <- (Ri) | X  | *   | *   | *   | *   | *   | *   | *   |
| MOVX A,ear        | 2  | 2   | 1  | 0   byte (ear) <- (A)  | X  | *   | *   | *   | *   | *   | *   |
| MOVX A,ear+disp8  | 2+ | 3 + (a) | 0 | (b) byte (ear) <- (A)| X  | *   | *   | *   | *   | *   | *   |
| MOVX A,#imm8      | 2  | 2   | 0  | 0   byte (A) <- (imm8) | X  | *   | *   | *   | *   | *   | *   |
| MOVX A,@RWi+disp8 | 2  | 5   | 1  | (b) byte (A) <- (RWi)+disp8 | X  | *   | *   | *   | *   | *   | *   |
| MOVX @RLi+disp8,A | 3  | 10  | 2  | (b) byte (A) <- (RLi)+disp8 | X  | *   | *   | *   | *   | *   | *   |
| MOV dir,A         | 2  | 3   | 0  | (b) byte (dir) <- (A)  | -  | -   | -   | -   | -   | -   | -   | -   | -   | -   | -   |
| MOV addr16,A      | 3  | 4   | 0  | (b) byte (addr16) <- (A) | -  | -   | -   | -   | -   | -   | -   | -   | -   | -   | -   |
| MOV Ri,A          | 1  | 2   | 1  | 0   byte (Ri) <- (A)  | -  | -   | -   | -   | -   | -   | -   | -   | -   | -   |
| MOV ear,A         | 2  | 2   | 1  | 0   byte (ear) <- (A)  | -  | -   | -   | -   | -   | -   | -   | -   | -   |
| MOV ear,Ri        | 2  | 4   | 2  | 0   byte (ear) <- (Ri) | -  | -   | -   | -   | -   | -   | -   | -   |
| MOV ear,imm8      | 3  | 2   | 1  | 0   byte (ear) <- (imm8) | -  | -   | -   | -   | -   | -   |
| MOV ear,#imm8     | 3  | 2   | 1  | 0   byte (ear) <- (imm8) | -  | -   | -   | -   | -   | -   |
| MOV ear,#imm8     | 3  | 5   | 0  | (b) byte (imm8) <- (ear) | -  | -   | -   | -   | -   | -   |
| MOV ear,#imm8     | 3  | 5   | 0  | (b) byte (imm8) <- (ear) | -  | -   | -   | -   | -   | -   |
| MOV @AL,AH / MOV @A,T | 2  | 3   | 0  | (b) byte ((A) <- (AH)) | -  | -   | -   | -   | -   | -   |
| MOVX A,ear        | 2+ | 3 + (a) | 2 | 0   byte (A) <- (ear) | Z  | -   | -   | -   | -   | -   | -   |
| MOVX A,eam        | 2+ | 5 + (a) | 0 | 2 x (b) byte (A) <- (eam) | Z  | -   | -   | -   | -   | -   |
| MOVX Ri,ear       | 2  | 7   | 4  | 0   byte (Ri) <- (ear) | -  | -   | -   | -   | -   |
| MOVX Ri,eam       | 2+ | 9 + (a) | 2 | 2 x (b) byte (Ri) <- (eam) | -  | -   | -   | -   | -   |

### Note:

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
## Table A.8-2 38 Transfer instructions (byte)

| Mnemonic          | # | ~ | RG | B | Operation                              | L | A | H | I | S | T | N | Z | V | C | R | M | W |
|-------------------|---|---|----|---|----------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOVW A,dir       | 2 | 3 | 0  | (c) | word (A) <-- (dir)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,addr16    | 3 | 4 | 0  | (c) | word (A) <-- (addr16)                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,SP        | 1 | 1 | 0  | (c) | word (A) <-- (SP)                      |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,RWi       | 1 | 2 | 1  | 0  | word (A) <-- (RWi)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,ear       | 2 | 2 | 1  | 0  | word (A) <-- (ear)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,eam       | 2+| 3+a| 0  | (c) | word (A) <-- (eam)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,io        | 2 | 3 | 0  | (c) | word (A) <-- (io)                      |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,@A        | 2 | 3 | 0  | (c) | word (A) <-- (A)                       |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,@RWi+disp8| 2 | 5 | 1  | (c) | word (A) <-- ((RWi)+disp8)            |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW A,@RLi+disp8| 3 | 10| 2  | (c) | word (A) <-- ((RLi)+disp8)            |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW dir,A       | 2 | 3 | 0  | (c) | word (dir) <-- (A)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW addr16,A    | 3 | 4 | 0  | (c) | word (addr16) <-- (A)                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW SP,A        | 1 | 1 | 0  | (c) | word (SP) <-- (A)                      |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW RWi,A       | 1 | 2 | 1  | 0  | word (RWi) <-- (A)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW ear,A       | 2 | 2 | 1  | 0  | word (ear) <-- (A)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW eam,A       | 2+| 3+a| 0  | (c) | word (eam) <-- (A)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW io,A        | 2 | 3 | 0  | (c) | word (io) <-- (A)                      |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW @RWi+disp8,A| 2 | 5 | 1  | (c) | word (@RWi)+disp8 <-- (A)             |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW @RLi+disp8,A| 3 | 10| 2  | (c) | word (@RLi)+disp8 <-- (A)             |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW RWi,ear     | 2 | 3 | 2  | 0  | word (RWi) <-- (ear)                   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW RWi,eam     | 2+| 4+a| 1  | (c) | word (RWi) <-- (eam)                   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW ear,Rwi     | 2 | 4 | 2  | 0  | word (ear) <-- (Rwi)                   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW eam,Rwi     | 2+| 5+a| 1  | (c) | word (eam) <-- (Rwi)                   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW RWi,#imm16  | 3 | 2 | 1  | 0  | word (RWi) <-- #imm16                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW io,#imm16   | 4 | 5 | 0  | (c) | word (io) <-- #imm16                   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW ear,#imm16  | 4 | 2 | 1  | 0  | word (ear) <-- #imm16                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW eam,#imm16  | 4+| 4+a| 0  | (c) | word (eam) <-- #imm16                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVW @AL,AH / MOVW @A,T | 2 | 3 | 0  | (c) | word (A) <-- (AH)                        |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| XCHW ear         | 2 | 4 | 2  | 0  | word (A) <--→ (ear)                    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| XCHW,eam         | 2+| 5+a| 0  | 2×(c) | word (A) <--→ (eam)                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| XCHW RWi,ear     | 2 | 7 | 4  | 0  | word (RWi) <--→ (ear)                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| XCHW RWi,eam     | 2+| 9+a| 2  | 2×(c) | word (RWi) <--→ (eam)                  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVIL A,ear      | 2 | 4 | 2  | 0  | long (A) <-- (ear)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVIL A,eam      | 2+| 5+a| 0  | (d) | long (A) <-- (eam)                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVIL A,#imm32   | 5 | 3 | 0  | 0  | long (A) <-- imm32                     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVIL ear,A      | 2 | 4 | 2  | 0  | long (ear1) <-- (A)                    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| MOVIL eam,A      | 2+| 5+a| 0  | (d) | long (eam1) <-- (A)                    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

**Note:**

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
### Table A.8-3  42 Addition/subtraction instructions (byte, word, long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD A,#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (A) + imm8</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD A,dir</td>
<td>2</td>
<td>5</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) + (dir)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) &lt;- (A) + (ear)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) + (eam)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) &lt;- (ear) + (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) &lt;- (eam) + (A)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDC A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (AH) + (AL) + (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDC A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) &lt;- (A) + (ear) + (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDC A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) + (eam) + (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDDC A</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (AH) + (AL) + (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB A,#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (A) - imm8</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB A,dir</td>
<td>2</td>
<td>5</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) - (dir)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) &lt;- (A) - (ear)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) - (eam)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) &lt;- (ear) - (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) &lt;- (eam) - (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBC A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (AH) - (AL) - (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBC A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) &lt;- (A) - (ear) - (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBC A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) &lt;- (A) - (eam) - (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBDC A</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (A) &lt;- (AH) - (AL) - (C)</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW A</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (AH) + (AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (A) + (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) &lt;- (A) + (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW A,#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) &lt;- (A) + imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) &lt;- (ear) + (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) &lt;- (eam) + (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDCW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (A) + (ear) + (C)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDCW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) &lt;- (A) + (eam) + (C)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW A</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (AH) - (AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (A) - (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) &lt;- (A) - (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW A,#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) &lt;- (A) - imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) &lt;- (ear) - (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) &lt;- (eam) - (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBWC A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) &lt;- (A) - (ear) - (C)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBWC A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) &lt;- (A) - (eam) - (C)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDL A,ear</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>long (A) &lt;- (A) + (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDL A,eam</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>(d)</td>
<td>long (A) &lt;- (A) + (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDL A,#imm32</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>long (A) &lt;- (A) + imm32</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBL A,ear</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>long (A) &lt;- (A) - (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBL A,eam</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>(d)</td>
<td>long (A) &lt;- (A) - (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBL A,#imm32</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>long (A) &lt;- (A) - imm32</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
### Table A.8-4  12 Increment/decrement instructions (byte, word, long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>INC</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>byte (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INC</td>
<td>ear</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>byte (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEC</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>byte (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEC</td>
<td>ear</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>byte (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCW</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>word (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCW</td>
<td>ear</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>word (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECW</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>word (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECW</td>
<td>ear</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>word (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCL</td>
<td>ear</td>
<td>2</td>
<td>7</td>
<td>4</td>
<td>long (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCL</td>
<td>ear</td>
<td>2+</td>
<td>9+(a)</td>
<td>0</td>
<td>long (ear) &lt;- (ear) + 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECL</td>
<td>ear</td>
<td>2</td>
<td>7</td>
<td>4</td>
<td>long (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECL</td>
<td>ear</td>
<td>2+</td>
<td>9+(a)</td>
<td>0</td>
<td>long (ear) &lt;- (ear) - 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.

### Table A.8-5  11 Compare instructions (byte, word, long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP</td>
<td>A</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>byte (AH) - (AL)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMP</td>
<td>ear</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>byte (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMP</td>
<td>ear</td>
<td>2+</td>
<td>3+(a)</td>
<td>0</td>
<td>byte (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMP</td>
<td>#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>byte (A) - imm8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPW</td>
<td>A</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>word (AH) - (AL)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPW</td>
<td>ear</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>word (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPW</td>
<td>ear</td>
<td>2+</td>
<td>3+(a)</td>
<td>0</td>
<td>word (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPW</td>
<td>#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (A) - imm16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPL</td>
<td>A</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>long (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPL</td>
<td>ear</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>long (A) - (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMPL</td>
<td>#imm32</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>long (A) - imm32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
### Table A.8-6 11 Unsigned multiplication/division instructions (word, long word)

| Mnemonic | #  | ~  | RG | B  | Operation                                      | L | A | H | I | S | T | N | Z | V | C | R | M | W |
|----------|----|----|----|----|-----------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DIVU A   | 1  | *1 | 0  | 0  | word (AH) / byte (AL) quotient --> byte (AL) remainder --> byte (AH) | - | - | - | - | - | - | * | * | * | * | * | * | * | * | * | * | * |
| DIVU A,ear | 2  | *2 | 1  | 0  | word (A) / byte (ear) quotient --> byte (A) remainder --> byte (ear) | - | - | - | - | - | - | * | * | * | * | * | * | * | * | * | * | * |
| DIVU A,eam | 2+ | *3 | 0  | *6 | word (A) / byte (eam) quotient --> byte (A) remainder --> byte (eam) | - | - | - | - | - | - | * | * | * | * | * | * | * | * | * | * | * |
| DIVUW A,ear | 2  | *4 | 1  | 0  | long (A) / word (ear) quotient --> word (A) remainder --> word (eam) | - | - | - | - | - | - | * | * | * | * | * | * | * | * | * | * | * |
| DIVUW A,eam | 2+ | *5 | 0  | *7 | long (A) / word (eam) quotient --> word (A) remainder --> word (eam) | - | - | - | - | - | - | * | * | * | * | * | * | * | * | * | * | * |
| MULU A   | 1  | *8 | 0  | 0  | byte (AH) * byte (AL) --> word (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MULU A,ear | 2  | *9 | 1  | 0  | byte (A) * byte (ear) --> word (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MULU A,eam | 2+ | *10| 0  | (b) | byte (A) * byte (eam) --> word (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MULUW A   | 1  | *11| 0  | 0  | word (AH) * word (AL) --> long (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MULUW A,ear | 2  | *12| 1  | 0  | word (A) * word (ear) --> long (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MULUW A,eam | 2+ | *13| 0  | (c) | word (A) * word (eam) --> long (A) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

*1: 3: Division by 0 7: Overflow 15: Normal
*2: 4: Division by 0 8: Overflow 16: Normal
*3: 6+(a): Division by 0 9+(a): Overflow 19+(a): Normal
*4: 4: Division by 0 7: Overflow 22: Normal
*5: 6+(a): Division by 0 8+(a): Overflow 26+(a): Normal
*6: (b): Division by 0 or overflow 2 x (b): Normal
*7: (c): Division by 0 or overflow 2 x (c): Normal
*8: 3: Byte (AH) is 0 7: Byte (AH) is not 0.
*9: 4: Byte (ear) is 0 8: Byte (ear) is not 0.
*10: 5+(a): Byte (eam) is 0 9+(a): Byte (eam) is not 0.
*11: 3: Word (AH) is 0 11: Word (AH) is not 0.
*12: 4: Word (ear) is 0 12: Word (ear) is not 0.
*13: 5+(a): Word (eam) is 0 13+(a): Word (eam) is not 0.

---

**Note:**

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (c) in the table.
Notes:

- The execution cycle count found when an overflow occurs in a DIV or DIVW instruction may be a pre-operation count or a post-operation count depending on the detection timing.

- When an overflow occurs with DIV or DIVW instruction, the contents of the AL are destroyed.

- See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (c) in the table.
### Table A.8-8 39 Logic 1 instructions (byte, word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND A,#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← (A) and imm8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>AND A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← (A) and (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>AND A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ← (A) and (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>AND ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← (ear) and (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>AND eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) ← (eam) and (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR A,#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← (A) or imm8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← (A) or (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ← (A) or (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← (ear) or (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) ← (eam) or (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XOR A,#imm8</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← (A) xor imm8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XOR A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← (A) xor (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XOR A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ← (A) xor (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XOR ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← (ear) xor (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XOR eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) ← (eam) xor (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOT A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← not (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOT ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← not (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOT eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) ← not (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (AH) and (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW A,#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (A) and imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) ← (A) and (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) ← (A) and (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) ← (ear) and (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDW eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) ← (eam) and (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (AH) or (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW A,#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (A) or imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) ← (A) or (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) ← (A) or (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) ← (ear) or (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORW eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) ← (eam) or (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (AH) xor (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW A,#imm16</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← (A) xor imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW A,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (A) ← (A) xor (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW A,eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (A) ← (A) xor (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW ear,A</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) ← (ear) xor (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORW eam,A</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) ← (eam) xor (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOTW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← not (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOTW ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) ← not (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NOTW eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) ← not (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (c) in the table.
APPENDIX A  Instruction

Table A.8-9  6 Logic 2 instructions (long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L H</th>
<th>A H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANDL</td>
<td>A,ear</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>long (A) ←→ (A) and (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ANDL</td>
<td>A,eam</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>(d)</td>
<td>long (A) ←→ (A) and (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORL</td>
<td>A,ear</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>long (A) ←→ (A) or (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ORL</td>
<td>A,eam</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>(d)</td>
<td>long (A) ←→ (A) or (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORL</td>
<td>A,ear</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>long (A) ←→ (A) xor (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XORL</td>
<td>A,eam</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>(d)</td>
<td>long (A) ←→ (A) xor (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Note:
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.

Table A.8-10  6 Sign inversion instructions (byte, word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L H</th>
<th>A H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEG</td>
<td>A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ←→ 0 - (A)</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NEG</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ←→ 0 - (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NEG</td>
<td>eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (b)</td>
<td>byte (eam) ←→ 0 - (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NEGW</td>
<td>A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ←→ 0 - (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NEGW</td>
<td>ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>word (ear) ←→ 0 - (ear)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>NEGW</td>
<td>eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (eam) ←→ 0 - (eam)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Note:
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (c) in the table.

Table A.8-11  1 Normalization instruction (long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L H</th>
<th>A H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRML</td>
<td>A,R0</td>
<td>2</td>
<td>*</td>
<td>1</td>
<td>0</td>
<td>long (A) ←→ Shifts to the position where ‘1’ is set for the first time. byte (RD) ←→ Shift count at that time</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

* : 4 when all accumulators have a value of 0; otherwise, 6+(R0)
### Table A.8-12 18 Shift instructions (byte, word, long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>RORC A</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← With right rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ROLC A</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← With left rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RORC ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← With right rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RORC eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>2</td>
<td>0</td>
<td>byte (eam) ← With right rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROLC ear</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>byte (ear) ← With left rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROLC eam</td>
<td>2+</td>
<td>5+(a)</td>
<td>2</td>
<td>0</td>
<td>byte (eam) ← With left rotation carry</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASR A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← Arithmetic right shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSR A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← Logical right barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSL A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>byte (A) ← Logical left barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASRW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← Arithmetic right shift (A, 1 bit)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSRW A/SHRW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← Logical right shift (A, 1 bit)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>R</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSLW A/SHLW A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (A) ← Logical left shift (A, 1 bit)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASRW A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>word (A) ← Arithmetic right barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS RW A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>word (A) ← Logical right barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSLW A,R0</td>
<td>2</td>
<td>*1</td>
<td>1</td>
<td>0</td>
<td>word (A) ← Logical left barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASRL A,R0</td>
<td>2</td>
<td>*2</td>
<td>1</td>
<td>0</td>
<td>long (A) ← Arithmetic right barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSRL A,R0</td>
<td>2</td>
<td>*2</td>
<td>1</td>
<td>0</td>
<td>long (A) ← Logical right barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSLL A,R0</td>
<td>2</td>
<td>*2</td>
<td>1</td>
<td>0</td>
<td>long (A) ← Logical left barrel shift (A, R0)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1: 6 when R0 is 0; otherwise, 5 + (R0)  
*2: 6 when R0 is 0; otherwise, 6 + (R0)

### Note:

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (b) in the table.
APPENDIX A  Instruction

### Table A.8-13 31 Branch 1 instructions

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>BZ/BEQ</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (Z) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNZ/BNE</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (Z) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BC/BLO</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (C) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNC/BHS</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (C) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BN</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (N) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BP</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (N) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BV</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (V) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNV</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (V) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BT</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (T) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNT</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (T) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLT</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (V) nor (N) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BGE</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (V) nor (N) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLE</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on ((V) xor (N)) or (Z) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BGT</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on ((V) xor (N)) or (Z) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLS</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (C) or (Z) = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BHI</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Branch on (C) or (Z) = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRA</td>
<td>rel</td>
<td>2</td>
<td>+1</td>
<td>0</td>
<td>Unconditional branch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMP</td>
<td>@A</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>word (PC) &lt;= (A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMP</td>
<td>addr16</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>word (PC) &lt;= addr16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMP</td>
<td>@ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>word (PC) &lt;= (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMP</td>
<td>@eam</td>
<td>2+</td>
<td>4+(a)</td>
<td>0</td>
<td>(c)</td>
<td>word (PC) &lt;= (eam)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMPP</td>
<td>@eam +3</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>word (PC) &lt;= (eam), (PCB) &lt;= (eam+2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMPP</td>
<td>@eam +3</td>
<td>2+</td>
<td>6+(b)</td>
<td>0</td>
<td>(d)</td>
<td>word (PC) &lt;= (eam), (PCB) &lt;= (eam+2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JMPP</td>
<td>addr24</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>word (PC) &lt;= ad24 0-15, (PCB) &lt;= ad24 16-23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>@ear +4</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>word (PC) &lt;= (ear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>@eam +4</td>
<td>2+</td>
<td>7+(a)</td>
<td>0</td>
<td>2 x (c)</td>
<td>word (PC) &lt;= (eam)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>addr16 +4</td>
<td>3</td>
<td>6</td>
<td>0</td>
<td>word (PC) &lt;= addr16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALLV</td>
<td>#vec4 +5</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>2 x (c) Vector call instruction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALLP</td>
<td>@ear +6</td>
<td>2</td>
<td>10</td>
<td>2</td>
<td>2 x (c) word (PC) &lt;= (ear)0-15, (PCB) &lt;= (ear)16-23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALLP</td>
<td>@eam +6</td>
<td>2+</td>
<td>11+(a)</td>
<td>0</td>
<td>*2</td>
<td>word (PC) &lt;= (eam)0-15, (PCB) &lt;= (eam)16-23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALLP</td>
<td>addr24 +7</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>2 x (c) word (PC) &lt;= addr0-15, (PCB) &lt;= addr16-23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1: 4 when a branch is made; otherwise, 3  
*2: 3 x (c) + (b)  
*3: Read (word) of branch destination address  
*4: W: Save to stack (word)  
R: Read (word) of branch destination address  
*5: Save to stack (word)  
*6: W: Save to stack (long word)  
R: Read (long word) of branch destination address  
*7: Save to stack (long word)

**Note:**  
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 'Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
### Table A.8-14 19 Branch 2 instructions

| Mnemonic   | #    | RG | B | Operation                                      | L | H | I | S | T | N | Z | V | C | R | M | W |
|------------|------|----|---|-----------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CBNE       | 3    | +1 | 0 | Branch on byte (A) not equal to imm8         |   |   | * | * | * | * | * | * |   |   |   |   |   |
| CBNE       | 4    | +1 | 0 | Branch on word (A) not equal to imm16        |   |   | * | * | * | * | * | * |   |   |   |   |   |
| CBNE       | 4    | +2 | 1 | Branch on byte (ear) not equal to imm8      |   |   | * | * | * | * | * | * |   |   |   |   |   |
| CBNE       | 4    | +3 | 0 | Branch on byte (eam) not equal to imm8      |   |   | * | * | * | * | * | * |   |   |   |   |   |
| CWBNE      | 5    | +4 | 1 | Branch on word (ear) not equal to imm16     |   |   | * | * | * | * | * | * |   |   |   |   |   |
| CWBNE      | 5    | +3 | 0 | Branch on word (eam) not equal to imm16     |   |   | * | * | * | * | * | * |   |   |   |   |   |
| DBNZ       | 3    | +5 | 2 | Branch on byte (ear) = (ear) - 1, (ear) not equal to 0 |   |   | * | * | * | * | * | * |   |   |   |   |   |
| DBNZ       | 3    | +6 | 2 | Branch on byte (eam) = (eam) - 1, (eam) not equal to 0 |   |   | * | * | * | * | * | * |   |   |   |   |   |
| DWBNZ      | 3    | +5 | 2 | Branch on word (ear) = (ear) - 1, (ear) not equal to 0 |   |   | * | * | * | * | * | * |   |   |   |   |   |
| DWBNZ      | 3    | +6 | 2 | Branch on word (eam) = (eam) - 1, (eam) not equal to 0 |   |   | * | * | * | * | * | * |   |   |   |   |   |
| INT        | 2    | 20 | 0 | Software interrupt                          |   |   | R | S |   |   |   |   |   |   |   |   |   |
| INT        | 3    | 16 | 0 | Software interrupt                          |   |   | R | S |   |   |   |   |   |   |   |   |   |
| INTP       | 4    | 17 | 0 | Software interrupt                          |   |   | R | S |   |   |   |   |   |   |   |   |   |
| INT9       | 1    | 20 | 0 | Software interrupt                          |   |   | R | S |   |   |   |   |   |   |   |   |   |
| RETI       | 1    | 11 | 0 | Return from interrupt                       |   |   |   |   |   |   |   |   |   |   |   |   |   |
| LINK       | 2    | 6  | 0 | Saves the old frame pointer in the stack upon entering the function, then sets the new frame pointer and reserves the local pointer area. |   |   |   |   |   |   |   |   |   |   |   |   |   |
| UNLINK     | 1    | 5  | 0 | Recovers the old frame pointer from the stack upon exiting the function. |   |   |   |   |   |   |   |   |   |   |   |   |   |
| RET        | 1    | 4  | 0 | Return from subroutine                       |   |   |   |   |   |   |   |   |   |   |   |   |   |
| RETP       | 1    | 6  | 0 | Return from subroutine                       |   |   |   |   |   |   |   |   |   |   |   |   |   |

*1: 5 when a branch is made; otherwise, 4  
*2: 13 when a branch is made; otherwise, 12  
*3: 7+(a) when a branch is made; otherwise, 6+(a)  
*4: 8 when a branch is made; otherwise, 7  
*5: 7 when a branch is made; otherwise, 6  
*6: 8+(a) when a branch is made; otherwise, 7+(a)  
*7: 3 x (b) + 2 x (c) when jumping to the next interruption request; 6 x (c) when returning from the current interruption  
*8: 15 when jumping to the next interruption request; 17 when returning from the current interruption  
*9: Do not use RW+j+ addressing mode with a CBNE or CWBNE instruction.  
*10: Return from stack (word)  
*11: Return from stack (long word)

### Note:
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
### Table A.8-15 28 Other control instructions (byte, word, long word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>H</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUSHW A</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>(c)</td>
<td>word (SP) ← (SP) - 2, ((SP)) ← (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PUSHW AH</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>(c)</td>
<td>word (SP) ← (SP) - 2, ((SP)) ← (AH)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PUSHW PS</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>(c)</td>
<td>word (SP) ← (SP) - 2, ((SP)) ← (PS)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PUSHW rlst</td>
<td>2</td>
<td>*3</td>
<td>*5</td>
<td>*4</td>
<td>(SP) ← (SP) - 2n, ((SP)) ← (rlst)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>POPW A</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>(c)</td>
<td>word (A) ← ((SP)), (SP) ← (SP) + 2</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>POPW AH</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>(c)</td>
<td>word (AH) ← ((SP)), (SP) ← (SP) + 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>POPW PS</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>(c)</td>
<td>word (PS) ← ((SP)), (SP) ← (SP) + 2</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>POPW rlst</td>
<td>2</td>
<td>*2</td>
<td>*5</td>
<td>*4</td>
<td>(rlst) ← ((SP)), (SP) ← (SP)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>JCTX</td>
<td>1</td>
<td>14</td>
<td>0</td>
<td>6 x (c)</td>
<td>Context switch instruction</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AND</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (CCR) ← (CCR) and imm8</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (CCR) ← (CCR) or imm8</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOV</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (RP) ← imm8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOV</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (ILM) ← imm8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVEA RW1,ear</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>word (RW1) ← ear</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVEA RW1,eam</td>
<td>2+</td>
<td>3+(a)</td>
<td>1</td>
<td>0</td>
<td>word (RW1) ← eam</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVEA A,ear</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>word (A) ← ear</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVEA A,eam</td>
<td>2+</td>
<td>1+(a)</td>
<td>0</td>
<td>0</td>
<td>word (A) ← eam</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ADDSP</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>word (SP) ← ext(imm8)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ADDSP</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>word (SP) ← imm16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOV</td>
<td>2</td>
<td>*1</td>
<td>0</td>
<td>0</td>
<td>byte (A) ← (brg1)</td>
<td>Z</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOV</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>byte (brg2) ← (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NOP</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>No operation</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ADB</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for AD space access</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DSB</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for DT space access</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PCB</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for PC space access</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SPB</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for SP space access</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NCC</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for flag no-change</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CMR</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Prefix code for common register bank</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

*1: PCB, ADB, SSB, USB, SPB: 1, DSB, DPR: 2
*2: 7 + 3 x (POP count) + 2 x (POP last register number), 7 when RLST = 0 (no transfer register)
*3: 29 + 3 x (PUSH count) - 3 x (PUSH last register number), 8 when RLST = 0 (no transfer register)
*4: (POP count) x (c) or (PUSH count) x (c)
*5: (POP count) or (PUSH count)

### Note:
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (c) in the table.
### Table A.8-16 21 Bit operand instructions

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVB dir:bp</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ⇐ (dir:bp)b</td>
<td>Z</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>MOVB addr16:bp</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ⇐ (addr16:bp)b</td>
<td>Z</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>MOVB io:bp:bp</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>(b)</td>
<td>byte (A) ⇐ (io:bp:bp)b</td>
<td>Z</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>MOVB dir:bp</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (dir:bp)b ⇐ (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVB addr16:bp</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (addr16:bp)b ⇐ (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MOVB io:bp:bp</td>
<td>3</td>
<td>6</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (io:bp:bp)b ⇐ (A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SETB dir:bp</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (dir:bp)b ⇐ 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SETB addr16:bp</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (addr16:bp)b ⇐ 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SETB io:bp</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (io:bp)b ⇐ 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CLRB dir:bp</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (dir:bp)b ⇐ 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CLRB addr16:bp</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (addr16:bp)b ⇐ 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CLRB io:bp</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>2 x (b)</td>
<td>bit (io:bp)b ⇐ 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBC dir:bp,rel</td>
<td>4</td>
<td>*1</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (dir:bp)b = 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBC addr16:bp,rel</td>
<td>5</td>
<td>*1</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (addr16:bp)b = 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBC io:bp,rel</td>
<td>4</td>
<td>*2</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (io:bp)b = 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBS dir:bp,rel</td>
<td>4</td>
<td>*1</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (dir:bp)b = 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBS addr16:bp,rel</td>
<td>5</td>
<td>*1</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (addr16:bp)b = 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BBS io:bp,rel</td>
<td>4</td>
<td>*1</td>
<td>0</td>
<td>(b)</td>
<td>Branch on (io:bp)b = 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SBBS addr16:bp,rel</td>
<td>5</td>
<td>*3</td>
<td>2 x (b)</td>
<td>Branch on (addr16:bp)b = 1, bit = 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>WBTS io:bp</td>
<td>3</td>
<td>*4</td>
<td>0</td>
<td>*5</td>
<td>Waits until (io:bp)b = 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WBTC io:bp</td>
<td>3</td>
<td>*4</td>
<td>0</td>
<td>*5</td>
<td>Waits until (io:bp)b = 0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

*1: 8 when a branch is made; otherwise, 7  
*2: 7 when a branch is made; otherwise, 6  
*3: 10 when the condition is met; otherwise, 9  
*4: Undefined count  
*5: Until the condition is met

**Note:**  
See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (b) in the table.

### Table A.8-17 6 Accumulator operation instructions (byte, word)

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>#</th>
<th>~</th>
<th>RG</th>
<th>B</th>
<th>Operation</th>
<th>L</th>
<th>H</th>
<th>A</th>
<th>I</th>
<th>S</th>
<th>T</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWAP</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>byte (A)0-7 ⇐ (A)8-15</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SWAPW / XCHW A.T</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>word (AH) ⇐ (AL)</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EXT</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Byte sign extension</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EXTW</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>Word sign extension</td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ZEXT</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Byte zero extension</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>R</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ZEXTW</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Word zero extension</td>
<td>Z</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>R</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
## APPENDIX A  Instruction

### Table A.8-18 10 String instructions

| Mnemonic     | # | ~  | RG | B | Operation | L | H | A | I | S | T | N | Z | V | C | R | M | W |
|--------------|---|----|----|---|-----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOVSI / MOVSI | 2 | *2 | *5 | *3 | byte transfer @AH+ <- @AL+, counter = RW0 | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MOVSD        | 2 | *2 | *5 | *3 | byte transfer @AH+ <- @AL+, counter = RW0 | - | - | - | - | - | - | - | - | - | - | - | - | - |
| SCEQ / SCEQI | 2 | *1 | *5 | *4 | byte search @AH+ <- @AL, counter RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |
| SCEQD        | 2 | *1 | *5 | *4 | byte search @AH+ <- @AL, counter RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |
| FILS / FILSI | 2 | 6m+6 | *5 | *3 | byte fill @AH+ <- @AL, counter RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |
| MOVSW / MOVSWI | 2 | *2 | *5 | *6 | word transfer @AH+ <- @AL+, counter = RW0 | - | - | - | - | - | - | - | - | - | - | - | - | - |
| MOVSWD       | 2 | *2 | *5 | *6 | word transfer @AH+ <- @AL+, counter = RW0 | - | - | - | - | - | - | - | - | - | - | - | - | - |
| SCWEQ / SCWEQI | 2 | *1 | *5 | *7 | word search @AH+ - @AL, counter = RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |
| SCWEQD       | 2 | *1 | *5 | *7 | word search @AH+ - @AL, counter = RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |
| FILSW / FILSWI | 2 | 6m+6 | *5 | *6 | word fill @AH+ <- @AL, counter = RW0 | - | - | - | - | - | - | * | * | * | - | - | - | - |

*1: 5 when RW0 is 0, 4 + 7 x (RW0) when the counter expires, or 7n + 5 when a match occurs
*2: 5 when RW0 is 0; otherwise, 4 + 8 x (RW0)
*3: (b) x (RW0) + (b) x (RW0) When the source and destination access different areas, calculate the (b) item individually.
*4: (b) x n
*5: 2 x (RW0)
*6: (c) x (RW0) + (c) x (RW0) When the source and destination access different areas, calculate the (c) item individually.
*7: (c) x n

### Note:

m: RW0 value (counter value), n: Loop count

See Table A.5-1 "Execution cycle counts in each addressing mode" and Table A.5-2 "Cycle count correction values for counting execution cycles" for information on (a) to (d) in the table.
A.9 Instruction Map

Each F\(^2\)MC-16LX instruction code consists of 1 to 2 bytes. Therefore, the instruction map consists of multiple pages. The F\(^2\)MC-16LX instruction map is given below.

### Configuration of Instruction Map

![Figure A.9-1 Configuration of Instruction Map](image)

An instruction such as the NOP instruction that ends in one byte is completed within the basic page. An instruction such as the MOVS instruction that requires two bytes recognizes the existence of byte 2 when it references byte 1, and can check the follow

Figure A.9-2 shows correspondence between actual instruction code and instruction map.
An example of an instruction code is shown in Table A.9-1 Example of an Instruction Code.

**Table A.9-1 Example of an instruction code**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Byte 1 (from basic page map)</th>
<th>Byte 2 (from extended page map)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOP</td>
<td>00 + 0=00</td>
<td>-</td>
</tr>
<tr>
<td>AND A,#8</td>
<td>30 + 4=34</td>
<td>-</td>
</tr>
<tr>
<td>MOV A,ADB</td>
<td>60 + F=6F</td>
<td>00 + 0=00</td>
</tr>
<tr>
<td>@RW2+d8,#8,rel</td>
<td>70 + 0=70</td>
<td>F0 + 2=F2</td>
</tr>
</tbody>
</table>

* The extended page map is a generic name of maps for bit operation instructions, character string operation instructions, 2-byte instructions, and ea instructions. Actually, there are multiple extended page maps for each type of instructions.
<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>NOP</td>
<td>CMR</td>
<td>ADD A,dir</td>
<td>ADD A,#8</td>
<td>MOV A,dir</td>
<td>MOV A,#0</td>
<td>BRA</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+1</td>
<td>INT9</td>
<td>NCC</td>
<td>SUB A,dir</td>
<td>SUB A,#8</td>
<td>MOV A,dir</td>
<td>MOV A,#0</td>
<td>JMP</td>
<td>@A</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+2</td>
<td>ADDC A</td>
<td>SUBDC A</td>
<td>ADDC A</td>
<td>SUBDC A</td>
<td>MOV A,dir</td>
<td>MOV A,#0</td>
<td>JMP</td>
<td>@A</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+3</td>
<td>NEG A</td>
<td>JCTX @A</td>
<td>CMP A</td>
<td>CMP A</td>
<td>MOVX A,dir</td>
<td>MOVX A,#0</td>
<td>JMP</td>
<td>@A</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+4</td>
<td>PCB</td>
<td>EXT</td>
<td>AND CCR,#8</td>
<td>AND dir,#8</td>
<td>MOV dr,#8</td>
<td>MOV io,#8</td>
<td>CALL adr16</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+5</td>
<td>DTB</td>
<td>ZEXT</td>
<td>OR CCR,#8</td>
<td>OR dir,#8</td>
<td>MOVX dr,#8</td>
<td>MOVX io,#8</td>
<td>CALL adr16</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+6</td>
<td>ADB</td>
<td>SWAP</td>
<td>DIVU A</td>
<td>XOR A,#8</td>
<td>MOVW A,dir #16</td>
<td>MOVW io,#16</td>
<td>RETP</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+7</td>
<td>SPB</td>
<td>ADDSP #8</td>
<td>MULU A</td>
<td>NOT A</td>
<td>MOVW SP,#8</td>
<td>MOVX adr16</td>
<td>RET</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+8</td>
<td>LINK</td>
<td>ADDL A,#32</td>
<td>ADDW A,adr16</td>
<td>MOVW A,dir</td>
<td>MOVW A,#0</td>
<td>INT #vct8</td>
<td>ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
<td>A,R1</td>
</tr>
<tr>
<td>+9</td>
<td>UNLINK</td>
<td>SUBL A,#32</td>
<td>SUBW A,adr16</td>
<td>MOVW A,dir</td>
<td>MOVW A,#0</td>
<td>INT adr16</td>
<td>MOVEA</td>
<td>RW,ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+A</td>
<td>MOV</td>
<td>RP,#6</td>
<td>CBNE A,#8</td>
<td>CBNE A,#16,rel</td>
<td>MOVW A,adr16</td>
<td>INTP adr24</td>
<td>MOV</td>
<td>RW,ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
<td>MOVX</td>
</tr>
<tr>
<td>+B</td>
<td>NEGW A</td>
<td>CMP</td>
<td>CMPW A</td>
<td>CMPW A</td>
<td>MOVL A,adr16</td>
<td>MOVX adr16</td>
<td>RETI</td>
<td>MOVW</td>
<td>RW,ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+C</td>
<td>LSLW A</td>
<td>EXTW</td>
<td>ANDW A</td>
<td>ANDW A</td>
<td>PUSHW A</td>
<td>POPW A</td>
<td>BIT operation instruction</td>
<td>MOV</td>
<td>ea,R1</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+D</td>
<td>ZEXTW</td>
<td>ORW A</td>
<td>ORW A</td>
<td>ORW A,adr16</td>
<td>PUSHW AH</td>
<td>POPW AH</td>
<td>MOVW ea,RW</td>
<td>MOV</td>
<td>ea,R1</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+E</td>
<td>ASRW A</td>
<td>SWAPW A</td>
<td>XORW A</td>
<td>XORW A,adr16</td>
<td>PUSHW PS</td>
<td>POPW PS</td>
<td>Character string operation instruction</td>
<td>XCH</td>
<td>ea,R1</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
<tr>
<td>+F</td>
<td>LS RW A</td>
<td>ADDSP #16</td>
<td>MULUW A</td>
<td>NOTW A</td>
<td>PUSHW rist</td>
<td>POPW rist</td>
<td>2-byte instruction</td>
<td>XCHW</td>
<td>RW,ea</td>
<td>MOV</td>
<td>A,R1</td>
<td>MOV</td>
<td>R1,A</td>
<td>MOV</td>
<td>R1,#8</td>
</tr>
</tbody>
</table>

Table A.9-2: Basic page map
Table A.9-3  Bit operation instruction map (first byte = 6C_H)

<table>
<thead>
<tr>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV8</td>
<td>A</td>
<td>MOV8</td>
<td>A</td>
<td>MOV8</td>
<td>A</td>
<td>MOV8</td>
<td>A</td>
<td>MOV8</td>
<td>A</td>
</tr>
<tr>
<td>0x00</td>
<td>0x01</td>
<td>0x02</td>
<td>0x03</td>
<td>0x04</td>
<td>0x05</td>
<td>0x06</td>
<td>0x07</td>
<td>0x08</td>
<td>0x09</td>
</tr>
</tbody>
</table>

### Bit Operation Instructions

- **MOVB**: Move byte to destination
- **CLRB**: Clear bit
- **SETB**: Set bit
- **BBC**: Branch on bit condition
- **BBS**: Bit select
- **WBTS**: Write back to source
- **WBTC**: Write back to destination

### Bit Operation Map

- **First Byte**: 6C_H
- **Second Byte**: Various values for different conditions
- **Third Byte**: Target address or condition code

### Example

- **MOVB**: Move byte to destination
- **CLRB**: Clear bit
- **SETB**: Set bit
- **BBC**: Branch on bit condition
- **BBS**: Bit select
- **WBTS**: Write back to source
- **WBTC**: Write back to destination
### Table A.9-4: Character string operation instruction map (first byte = 6Eh)

<table>
<thead>
<tr>
<th>+0</th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVSI</td>
<td>MOVSD</td>
<td>MOVSWI</td>
<td>MOVSWD</td>
<td>SCEQI</td>
<td>SCEQD</td>
<td>SCWEOQI</td>
<td>SCWEOQD</td>
<td>FILSI</td>
<td>FILSWI</td>
<td>PCB</td>
<td>PCB</td>
<td>PCB</td>
<td>PCB</td>
<td>PCB</td>
<td>PCB</td>
<td></td>
</tr>
<tr>
<td>PCB,PCB</td>
<td>PCB,DTB</td>
<td>PCB,ADB</td>
<td>PCB,SPB</td>
<td>DTB,PCB</td>
<td>DTB,DTB</td>
<td>DTB,DTB</td>
<td>DTB,DTB</td>
<td>DTB,DTB</td>
<td>DTB</td>
<td>DTB</td>
<td>DTB</td>
<td>DTB</td>
<td>DTB</td>
<td>DTB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+2</td>
<td>PCB,ADB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+3</td>
<td>PCB,SPB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+4</td>
<td>DTB,PCB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+5</td>
<td>DTB,DTB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+6</td>
<td>DTB,ADB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+7</td>
<td>DTB,SPB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+8</td>
<td>ADB,PCB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+9</td>
<td>ADB,DTB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+A</td>
<td>ADB,ADB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+B</td>
<td>ADB,SPB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+C</td>
<td>SPB,PCB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+D</td>
<td>SPB,DTB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+E</td>
<td>SPB,ADB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+F</td>
<td>SPB,SPB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**APPENDIX A Instruction**
Table A.9-5  2-byte instruction map (first byte = 6FH)

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>MOV A, DTB</td>
</tr>
<tr>
<td>01</td>
<td>MOV A, DB</td>
</tr>
<tr>
<td>02</td>
<td>MOV A, SB</td>
</tr>
<tr>
<td>03</td>
<td>MOV A, UB</td>
</tr>
<tr>
<td>04</td>
<td>MOV A, DPR</td>
</tr>
<tr>
<td>05</td>
<td>MOV A, @A</td>
</tr>
<tr>
<td>06</td>
<td>MOV A, PC</td>
</tr>
<tr>
<td>07</td>
<td>ROLC A</td>
</tr>
<tr>
<td>08</td>
<td>LSL W A</td>
</tr>
<tr>
<td>09</td>
<td>MOVX A, @R0</td>
</tr>
<tr>
<td>0A</td>
<td>MOVX A, @R1+8</td>
</tr>
<tr>
<td>0B</td>
<td>MOVX A, @R2+8</td>
</tr>
<tr>
<td>0C</td>
<td>MOVX A, @R3+8</td>
</tr>
<tr>
<td>0D</td>
<td>MOVX A, @R4+8</td>
</tr>
<tr>
<td>0E</td>
<td>MOVX A, @R5+8</td>
</tr>
<tr>
<td>0F</td>
<td>MOVX A, @R6+8</td>
</tr>
<tr>
<td>10</td>
<td>MOVX A, @R7+8</td>
</tr>
<tr>
<td>11</td>
<td>MOVX A, @R8+8</td>
</tr>
<tr>
<td>12</td>
<td>MOVX A, @R9+8</td>
</tr>
<tr>
<td>13</td>
<td>MOVX A, @RA</td>
</tr>
<tr>
<td>14</td>
<td>MOVX A, @RB</td>
</tr>
<tr>
<td>15</td>
<td>MOVX A, @RC</td>
</tr>
<tr>
<td>16</td>
<td>MOVX A, @RD</td>
</tr>
<tr>
<td>17</td>
<td>LSL W A</td>
</tr>
<tr>
<td>18</td>
<td>NRML A</td>
</tr>
<tr>
<td>19</td>
<td>ASR A</td>
</tr>
<tr>
<td>1A</td>
<td>LSR A</td>
</tr>
<tr>
<td>1B</td>
<td>MOV A, @R0</td>
</tr>
<tr>
<td>1C</td>
<td>MOV A, @R1</td>
</tr>
<tr>
<td>1D</td>
<td>MOV A, @R2</td>
</tr>
<tr>
<td>1E</td>
<td>MOV A, @R3</td>
</tr>
<tr>
<td>1F</td>
<td>MOV A, @R4</td>
</tr>
</tbody>
</table>

Notes:
- DTB: Direct Transfer Register
- DB: Data Register
- SB: Stack Pointer
- UB: User Register
- DPR: Data Pointer Register
- @A: Address Register
- PC: Program Counter
- R0-R9: General Purpose Register
- RA-RD: Address Pointer Register
- @R0-R9: Address Pointer Register
- @R0 @R1 @R2 @R3 @R4 @R5 @R6 @R7 @R8 @R9: Address Pointer Register
- @R0 @R1 @R2 @R3 @R4 @R5 @R6 @R7 @R8 @R9: Address Pointer Register
- LSL: Logical Shift Left
- NRML: Normal
- ASR: Arithmetic Shift Right
- LSR: Logical Shift Right
### Table A.9-6: ea instruction (first byte = 70H)

<table>
<thead>
<tr>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW0+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW1+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW2+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW3+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW4+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW5+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW6+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RW7+d8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+8</td>
<td>A</td>
<td>R8W: 70H</td>
<td>A</td>
<td>R8W: 70H</td>
<td>A</td>
<td>R8W: 70H</td>
<td>A</td>
<td>R8W: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+C</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td>A&amp;R0: 70H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>A0</td>
<td>B0</td>
<td>C0</td>
<td>D0</td>
<td>E0</td>
</tr>
<tr>
<td>----------</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>+0</td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
<td>@R0</td>
</tr>
<tr>
<td>+1</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+2</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+3</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+4</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+5</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+6</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+7</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW7</td>
</tr>
<tr>
<td>+8</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+9</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+A</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+B</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+C</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+D</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+E</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td></td>
<td>JMP</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>INCL</td>
<td>INCL</td>
<td>DECL</td>
<td>DECL</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
<tr>
<td></td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>+F</td>
<td>@R0</td>
<td>@R0</td>
<td>@R1</td>
<td>@R1</td>
<td>@R2</td>
<td>@R2</td>
<td>@R3</td>
<td>@R3</td>
<td>@R4</td>
<td>@R4</td>
<td>@R5</td>
<td>@R5</td>
<td>@R6</td>
<td>@R6</td>
<td>@R7</td>
</tr>
<tr>
<td>00</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td>08</td>
<td>09</td>
<td>A0</td>
<td>B0</td>
<td>C0</td>
<td>D0</td>
<td>E0</td>
<td>F0</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>+0</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+1</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+2</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+3</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+4</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+5</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+6</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+7</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+8</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+9</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+A</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+B</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+C</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+D</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+E</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
<tr>
<td>+F</td>
<td>ROLC</td>
<td>ROLC</td>
<td>RORC</td>
<td>RORC</td>
<td>INC</td>
<td>INC</td>
<td>DEC</td>
<td>DEC</td>
<td>MOV</td>
<td>MOV</td>
<td>A,MOV @R</td>
<td>MOVX</td>
<td>MOVX A,</td>
<td>XCH</td>
<td>XCH A.</td>
</tr>
</tbody>
</table>

**Table A.9-8: ea instruction 3 (first byte = 72H)**

- **ROLC**: Rotate Left C (carry flag)
- **RORC**: Rotate Right C (carry flag)
- **INC**: Increment
- **DEC**: Decrement
- **MOV**: Move
- **MOVX**: Move with index
- **XCH**: Exchange
- **@RW**: Pointer register
- **@RWX**: Pointer register with index
- **@PC**: Program counter
- **addr**: Absolute address

Note: The table represents the instruction set for the 8086 architecture.
<table>
<thead>
<tr>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>NCW</td>
<td>INCW</td>
<td>DECW</td>
<td>DECW</td>
<td>MOVW</td>
<td>MOVW A</td>
<td>MOVW</td>
<td>MOVW @R</td>
<td>MOVW</td>
<td>MOVW @RW</td>
<td>XOHW</td>
<td>XOHW A</td>
</tr>
<tr>
<td>+1</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW0</td>
<td>@RW0+dB</td>
<td>@RW0+dB</td>
<td>RW0</td>
<td>@RW0+dB</td>
<td>A@RW0+dB</td>
<td>BRW0+dB</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
</tr>
<tr>
<td>+2</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW1</td>
<td>@RW1+dB</td>
<td>@RW1+dB</td>
<td>RW1</td>
<td>@RW1+dB</td>
<td>A@RW1+dB</td>
<td>BRW1+dB</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
</tr>
<tr>
<td>+3</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW2</td>
<td>@RW2+dB</td>
<td>@RW2+dB</td>
<td>RW2</td>
<td>@RW2+dB</td>
<td>A@RW2+dB</td>
<td>BRW2+dB</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
</tr>
<tr>
<td>+4</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW3</td>
<td>@RW3+dB</td>
<td>@RW3+dB</td>
<td>RW3</td>
<td>@RW3+dB</td>
<td>A@RW3+dB</td>
<td>BRW3+dB</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
</tr>
<tr>
<td>+5</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW4</td>
<td>@RW4+dB</td>
<td>@RW4+dB</td>
<td>RW4</td>
<td>@RW4+dB</td>
<td>A@RW4+dB</td>
<td>BRW4+dB</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
</tr>
<tr>
<td>+6</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW5</td>
<td>@RW5+dB</td>
<td>@RW5+dB</td>
<td>RW5</td>
<td>@RW5+dB</td>
<td>A@RW5+dB</td>
<td>BRW5+dB</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
</tr>
<tr>
<td>+7</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW6</td>
<td>@RW6+dB</td>
<td>@RW6+dB</td>
<td>RW6</td>
<td>@RW6+dB</td>
<td>A@RW6+dB</td>
<td>BRW6+dB</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
</tr>
<tr>
<td>+8</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>RW7</td>
<td>@RW7+dB</td>
<td>@RW7+dB</td>
<td>RW7</td>
<td>@RW7+dB</td>
<td>A@RW7+dB</td>
<td>BRW7+dB</td>
<td>RW7, @W0+dB A</td>
<td>RW7, @W0+dB A</td>
<td>RW7, @W0+dB A</td>
<td>RW7, @W0+dB A</td>
<td>RW7, @W0+dB A</td>
</tr>
<tr>
<td>+9</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW0</td>
<td>@RW0+dB</td>
<td>@RW0+dB</td>
<td>@RW0</td>
<td>@RW0+dB</td>
<td>A@RW0+dB</td>
<td>BRW0+dB</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
<td>RW0, @W0+dB A</td>
</tr>
<tr>
<td>+A</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW1</td>
<td>@RW1+dB</td>
<td>@RW1+dB</td>
<td>@RW1</td>
<td>@RW1+dB</td>
<td>A@RW1+dB</td>
<td>BRW1+dB</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
<td>RW1, @W0+dB A</td>
</tr>
<tr>
<td>+B</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW2</td>
<td>@RW2+dB</td>
<td>@RW2+dB</td>
<td>@RW2</td>
<td>@RW2+dB</td>
<td>A@RW2+dB</td>
<td>BRW2+dB</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
<td>RW2, @W0+dB A</td>
</tr>
<tr>
<td>+C</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW3</td>
<td>@RW3+dB</td>
<td>@RW3+dB</td>
<td>@RW3</td>
<td>@RW3+dB</td>
<td>A@RW3+dB</td>
<td>BRW3+dB</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
<td>RW3, @W0+dB A</td>
</tr>
<tr>
<td>+D</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW4</td>
<td>@RW4+dB</td>
<td>@RW4+dB</td>
<td>@RW4</td>
<td>@RW4+dB</td>
<td>A@RW4+dB</td>
<td>BRW4+dB</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
<td>RW4, @W0+dB A</td>
</tr>
<tr>
<td>+E</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW5</td>
<td>@RW5+dB</td>
<td>@RW5+dB</td>
<td>@RW5</td>
<td>@RW5+dB</td>
<td>A@RW5+dB</td>
<td>BRW5+dB</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
<td>RW5, @W0+dB A</td>
</tr>
<tr>
<td>+F</td>
<td>JMP</td>
<td>CALL</td>
<td>CALL</td>
<td>@RW6</td>
<td>@RW6+dB</td>
<td>@RW6+dB</td>
<td>@RW6</td>
<td>@RW6+dB</td>
<td>A@RW6+dB</td>
<td>BRW6+dB</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
<td>RW6, @W0+dB A</td>
</tr>
</tbody>
</table>

Table A.9-9: ea instruction (first byte = 73H)
| +0 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW0+d8 | @RW0+d8 | DBNZ @ RW0+d8 | RW0+d8 | @RW0+d8 | DBNZ @ RW0+d8 |
| +1 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW1+d8 | @RW1+d8 | DBNZ @ RW1+d8 | RW1+d8 | @RW1+d8 | DBNZ @ RW1+d8 |
| +2 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW2+d8 | @RW2+d8 | DBNZ @ RW2+d8 | RW2+d8 | @RW2+d8 | DBNZ @ RW2+d8 |
| +3 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW3+d8 | @RW3+d8 | DBNZ @ RW3+d8 | RW3+d8 | @RW3+d8 | DBNZ @ RW3+d8 |
| +4 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW4+d8 | @RW4+d8 | DBNZ @ RW4+d8 | RW4+d8 | @RW4+d8 | DBNZ @ RW4+d8 |
| +5 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW5+d8 | @RW5+d8 | DBNZ @ RW5+d8 | RW5+d8 | @RW5+d8 | DBNZ @ RW5+d8 |
| +6 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW6+d8 | @RW6+d8 | DBNZ @ RW6+d8 | RW6+d8 | @RW6+d8 | DBNZ @ RW6+d8 |
| +7 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW7+d8 | @RW7+d8 | DBNZ @ RW7+d8 | RW7+d8 | @RW7+d8 | DBNZ @ RW7+d8 |
| +8 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW0+d16 | @RW0+d16 | DBNZ @ RW0+d16 | RW0+d16 | @RW0+d16 | DBNZ @ RW0+d16 |
| +9 | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW1+d16 | @RW1+d16 | DBNZ @ RW1+d16 | RW1+d16 | @RW1+d16 | DBNZ @ RW1+d16 |
| +A | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW2+d16 | @RW2+d16 | DBNZ @ RW2+d16 | RW2+d16 | @RW2+d16 | DBNZ @ RW2+d16 |
| +B | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW3+d16 | @RW3+d16 | DBNZ @ RW3+d16 | RW3+d16 | @RW3+d16 | DBNZ @ RW3+d16 |
| +C | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW4+d16 | @RW4+d16 | DBNZ @ RW4+d16 | RW4+d16 | @RW4+d16 | DBNZ @ RW4+d16 |
| +D | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW5+d16 | @RW5+d16 | DBNZ @ RW5+d16 | RW5+d16 | @RW5+d16 | DBNZ @ RW5+d16 |
| +E | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW6+d16 | @RW6+d16 | DBNZ @ RW6+d16 | RW6+d16 | @RW6+d16 | DBNZ @ RW6+d16 |
| +F | ADD | ADD | A | SUB | SUB | A | ADDC | ADDC | A | CMP | CMP | A | AND | AND | A | OR | OR | A | XOR | XOR | A | DBNZ | DBNZ | RW7+d16 | @RW7+d16 | DBNZ @ RW7+d16 | RW7+d16 | @RW7+d16 | DBNZ @ RW7+d16 |

**Table A.9-10** ea instruction (first byte = 74H)
<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+1</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+2</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+3</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+4</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+5</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+6</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+7</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+8</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+9</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+A</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+B</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+C</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+D</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+E</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>+F</td>
<td>ADD</td>
<td>@R</td>
<td>SUB</td>
<td>@R</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>A</td>
<td>NEG</td>
<td>R0</td>
<td>@R</td>
<td>W0</td>
<td>@R</td>
<td>W0</td>
<td>D8</td>
</tr>
<tr>
<td>00</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>A0</td>
<td>B0</td>
<td>C0</td>
<td>D0</td>
<td>E0</td>
<td>F0</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>+0</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+1</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+2</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+3</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+4</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+5</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+6</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+7</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+8</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>+9</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>A</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>B</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>C</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>D</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>E</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>F</td>
<td>ADNW</td>
<td>ADNW A.</td>
<td>SUBW</td>
<td>SUBW A.</td>
<td>ADNW</td>
<td>ADNW</td>
<td>CMPW</td>
<td>CMPW A.</td>
<td>ANDW</td>
<td>ANDW A.</td>
<td>ORW</td>
<td>ORW A.</td>
<td>XORW</td>
<td>XORW A.</td>
<td>DBWBNZ</td>
<td>DBWBNZ @</td>
</tr>
<tr>
<td>Instruction</td>
<td>RW0</td>
<td>RW1</td>
<td>RW2</td>
<td>RW3</td>
<td>RW4</td>
<td>RW5</td>
<td>RW6</td>
<td>RW7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDW</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBW</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBC</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBC</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEGW</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANDW</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORW</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XORW</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOTW</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- RW0 to RW7 represent registers 0 to 7.
- Each instruction is shown with a constant byte value for demonstration purposes.
- Addresses are not shown in this table.
- Instructions are grouped by the first byte of the instruction code.
Table A.9-14 ea instruction 9 (first byte = 78H)

<table>
<thead>
<tr>
<th>opcode</th>
<th>FO</th>
<th>D0</th>
<th>E0</th>
<th>Co</th>
<th>B0</th>
<th>AO</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MULU**

A,R0MUL  A,R1MUL  A,R2MUL  A,R3MUL  A,R4MUL  ...  A,@RW1MUL  A,@RW2MUL  A,@RW3MUL  A,@RW0+MUL  A,@RW1+MUL  A,@RW2+MUL  A,@RW3+

**MULU** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  ...  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16

**MUL**

A,R0MUL  A,R1MUL  A,R2MUL  A,R3MUL  A,R4MUL  ...  A,@RW1MUL  A,@RW2MUL  A,@RW3MUL  A,@RW0+MUL  A,@RW1+MUL  A,@RW2+MUL  A,@RW3+

**MUL** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  ...  A,@RW1+d16  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16

**DIVU**

A,R0DIVU  A,R1DIVU  A,R2DIVU  A,R3DIVU  A,R4DIVU  ...  A,@RW1DIVU  A,@RW2DIVU  A,@RW3DIVU  A,@RW0+DIVU  A,@RW1+DIVU  A,@RW2+DIVU  A,@RW3+

**DIVU** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  ...  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16

**DIVUW**

A,RW0DIVUW  A,RW1DIVUW  A,RW2DIVUW  A,RW3DIVUW  A,RW4DIVUW  A,RW5DIVUW  ...  A,@RW1DIVUW  A,@RW2DIVUW  A,@RW3DIVUW  A,@RW0+DIVUW  A,@RW1+DIVUW  A,@RW2+DIVUW  A,@RW3+

**DIVUW** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  A,@RW7+d8  ...  A,@RW1+d16  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16

**DIV**

A,R0DIV  A,R1DIV  A,R2DIV  A,R3DIV  A,R4DIV  ...  A,@RW1DIV  A,@RW2DIV  A,@RW3DIV  A,@RW0+DIV  A,@RW1+DIV  A,@RW2+DIV  A,@RW3+

**DIV** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  ...  A,@RW1+d16  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16

**DIVW**

A,RW0DIVW  A,RW1DIVW  A,RW2DIVW  A,RW3DIVW  A,RW4DIVW  A,RW5DIVW  ...  A,@RW1DIVW  A,@RW2DIVW  A,@RW3DIVW  A,@RW0+DIVW  A,@RW1+DIVW  A,@RW2+DIVW  A,@RW3+

**DIVW** A,@RW0+d8  A,@RW1+d8  A,@RW2+d8  A,@RW3+d8  A,@RW4+d8  A,@RW5+d8  A,@RW6+d8  A,@RW7+d8  ...  A,@RW1+d16  A,@RW2+d16  A,@RW3+d16  A,@RW0+RW7  A,@RW1+RW7  A,@PC+d16  A,addr16
Table A.9-15  MOVEA RWi, ea instruction (first byte = 79H)

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>MOVEA</td>
<td>MOVEA RW1</td>
<td>MOVEA</td>
<td>MOVEA RW2</td>
<td>MOVEA</td>
<td>MOVEA RW3</td>
<td>MOVEA</td>
<td>MOVEA RW4</td>
<td>MOVEA</td>
<td>MOVEA RW5</td>
<td>MOVEA</td>
<td>MOVEA RW6</td>
<td>MOVEA</td>
<td>MOVEA RW7</td>
</tr>
<tr>
<td>+1</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>@RW0+@d8</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
<td>RW1RW0</td>
<td>RW1</td>
</tr>
<tr>
<td>+2</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
<td>RW2</td>
<td>RW2RW0</td>
</tr>
<tr>
<td>+3</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
<td>RW3</td>
<td>RW3RW0</td>
</tr>
<tr>
<td>+4</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
<td>RW4</td>
<td>RW4RW0</td>
</tr>
<tr>
<td>+5</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
<td>RW5</td>
<td>RW5RW0</td>
</tr>
<tr>
<td>+6</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
<td>RW6</td>
<td>RW6RW0</td>
</tr>
<tr>
<td>+7</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
<td>RW7</td>
<td>RW7RW0</td>
</tr>
<tr>
<td>+8</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
<td>RW8</td>
<td>RW8RW0</td>
</tr>
<tr>
<td>+9</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
<td>RW9</td>
<td>RW9RW0</td>
</tr>
<tr>
<td>+A</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
<td>RW10</td>
<td>RW10RW0</td>
</tr>
<tr>
<td>+B</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
<td>RW11</td>
<td>RW11RW0</td>
</tr>
<tr>
<td>+C</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
<td>RW12</td>
<td>RW12RW0</td>
</tr>
<tr>
<td>+D</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
<td>RW13</td>
<td>RW13RW0</td>
</tr>
<tr>
<td>+E</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
<td>RW14</td>
<td>RW14RW0</td>
</tr>
<tr>
<td>+F</td>
<td>MOVEA</td>
<td>MOVEA RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
<td>RW15</td>
<td>RW15RW0</td>
</tr>
</tbody>
</table>

APPENDIX A  Instruction
<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+1</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+2</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+3</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+4</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+5</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+6</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+7</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+8</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+9</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+A</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+B</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+C</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+D</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+E</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
<tr>
<td>+F</td>
<td>MOV R0, R0</td>
<td>MOV R0, @RWO+d8</td>
<td>MOV R1, R1</td>
<td>MOV R1, @RWO+d8</td>
<td>MOV R2, R2</td>
<td>MOV R2, @RWO+d8</td>
<td>MOV R3, R3</td>
<td>MOV R3, @RWO+d8</td>
<td>MOV R4, R4</td>
<td>MOV R4, @RWO+d8</td>
<td>MOV R5, R5</td>
<td>MOV R5, @RWO+d8</td>
<td>MOV R6, R6</td>
<td>MOV R6, @RWO+d8</td>
<td>MOV R7, R7</td>
<td>MOV R7, @RWO+d8</td>
</tr>
</tbody>
</table>

Table A.9-16 MOV Ri, ea instruction (first byte = 7AH)
Table A.9-17 MOVW RWi, ea instruction (first byte = 7BH)

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
</tr>
<tr>
<td>+1</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
<td>MOVW</td>
<td>MOVW RW0,</td>
</tr>
<tr>
<td>+2</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+3</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+4</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+5</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+6</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+7</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+8</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+9</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+A</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+B</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+C</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+D</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+E</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>+F</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
<td>RW0,RW0</td>
<td>@RW0,@RW0</td>
</tr>
<tr>
<td>Address</td>
<td>MOV R0, R0</td>
<td>MOV R1, R0</td>
<td>MOV R2, R0</td>
<td>MOV R3, R0</td>
<td>MOV R4, R0</td>
<td>MOV R5, R0</td>
<td>MOV R6, R0</td>
<td>MOV R7, R0</td>
<td>MOV R8, R0</td>
<td>MOV R9, R0</td>
<td>MOV R10, R0</td>
<td>MOV R11, R0</td>
<td>MOV R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0A</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0B</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0C</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0D</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0E</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0F</td>
<td>MOV @R0, R0</td>
<td>MOV @R1, R0</td>
<td>MOV @R2, R0</td>
<td>MOV @R3, R0</td>
<td>MOV @R4, R0</td>
<td>MOV @R5, R0</td>
<td>MOV @R6, R0</td>
<td>MOV @R7, R0</td>
<td>MOV @R8, R0</td>
<td>MOV @R9, R0</td>
<td>MOV @R10, R0</td>
<td>MOV @R11, R0</td>
<td>MOV @R12, R0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>A0</td>
<td>B0</td>
<td>C0</td>
<td>D0</td>
<td>E0</td>
<td>F0</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>+0</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td>RW0 RW0</td>
<td></td>
</tr>
<tr>
<td>+1</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW1 RW0</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td>RW1 RW1</td>
<td></td>
</tr>
<tr>
<td>+2</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td>RW2 RW0</td>
<td></td>
</tr>
<tr>
<td>+3</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW3 RW0</td>
<td>RW3 RW1</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td>RW3 RW0</td>
<td></td>
</tr>
<tr>
<td>+4</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW4 RW0</td>
<td>RW4 RW1</td>
<td>RW4 RW2</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td>RW4 RW0</td>
<td></td>
</tr>
<tr>
<td>+5</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW5 RW0</td>
<td>RW5 RW1</td>
<td>RW5 RW2</td>
<td>RW5 RW3</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td>RW5 RW0</td>
<td></td>
</tr>
<tr>
<td>+6</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW6 RW0</td>
<td>RW6 RW1</td>
<td>RW6 RW2</td>
<td>RW6 RW3</td>
<td>RW6 RW4</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td>RW6 RW0</td>
<td></td>
</tr>
<tr>
<td>+7</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>RW7 RW0</td>
<td>RW7 RW1</td>
<td>RW7 RW2</td>
<td>RW7 RW3</td>
<td>RW7 RW4</td>
<td>RW7 RW5</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td>RW7 RW0</td>
<td></td>
</tr>
<tr>
<td>+8</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW0 RW0</td>
<td>@RW0 RW1</td>
<td>@RW0 RW2</td>
<td>@RW0 RW3</td>
<td>@RW0 RW4</td>
<td>@RW0 RW5</td>
<td>@RW0 RW6</td>
<td>@RW0 RW7</td>
<td>@RW0 RW8</td>
<td>@RW0 RW9</td>
<td>@RW0 RW10</td>
<td>@RW0 RW11</td>
<td>@RW0 RW12</td>
<td>@RW0 RW13</td>
<td>@RW0 RW14</td>
<td></td>
</tr>
<tr>
<td>+9</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW1 RW0</td>
<td>@RW1 RW1</td>
<td>@RW1 RW2</td>
<td>@RW1 RW3</td>
<td>@RW1 RW4</td>
<td>@RW1 RW5</td>
<td>@RW1 RW6</td>
<td>@RW1 RW7</td>
<td>@RW1 RW8</td>
<td>@RW1 RW9</td>
<td>@RW1 RW10</td>
<td>@RW1 RW11</td>
<td>@RW1 RW12</td>
<td>@RW1 RW13</td>
<td>@RW1 RW14</td>
<td></td>
</tr>
<tr>
<td>+A</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW2 RW0</td>
<td>@RW2 RW1</td>
<td>@RW2 RW2</td>
<td>@RW2 RW3</td>
<td>@RW2 RW4</td>
<td>@RW2 RW5</td>
<td>@RW2 RW6</td>
<td>@RW2 RW7</td>
<td>@RW2 RW8</td>
<td>@RW2 RW9</td>
<td>@RW2 RW10</td>
<td>@RW2 RW11</td>
<td>@RW2 RW12</td>
<td>@RW2 RW13</td>
<td>@RW2 RW14</td>
<td></td>
</tr>
<tr>
<td>+B</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW3 RW0</td>
<td>@RW3 RW1</td>
<td>@RW3 RW2</td>
<td>@RW3 RW3</td>
<td>@RW3 RW4</td>
<td>@RW3 RW5</td>
<td>@RW3 RW6</td>
<td>@RW3 RW7</td>
<td>@RW3 RW8</td>
<td>@RW3 RW9</td>
<td>@RW3 RW10</td>
<td>@RW3 RW11</td>
<td>@RW3 RW12</td>
<td>@RW3 RW13</td>
<td>@RW3 RW14</td>
<td></td>
</tr>
<tr>
<td>+C</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW4 RW0</td>
<td>@RW4 RW1</td>
<td>@RW4 RW2</td>
<td>@RW4 RW3</td>
<td>@RW4 RW4</td>
<td>@RW4 RW5</td>
<td>@RW4 RW6</td>
<td>@RW4 RW7</td>
<td>@RW4 RW8</td>
<td>@RW4 RW9</td>
<td>@RW4 RW10</td>
<td>@RW4 RW11</td>
<td>@RW4 RW12</td>
<td>@RW4 RW13</td>
<td>@RW4 RW14</td>
<td></td>
</tr>
<tr>
<td>+D</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW5 RW0</td>
<td>@RW5 RW1</td>
<td>@RW5 RW2</td>
<td>@RW5 RW3</td>
<td>@RW5 RW4</td>
<td>@RW5 RW5</td>
<td>@RW5 RW6</td>
<td>@RW5 RW7</td>
<td>@RW5 RW8</td>
<td>@RW5 RW9</td>
<td>@RW5 RW10</td>
<td>@RW5 RW11</td>
<td>@RW5 RW12</td>
<td>@RW5 RW13</td>
<td>@RW5 RW14</td>
<td></td>
</tr>
<tr>
<td>+E</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW6 RW0</td>
<td>@RW6 RW1</td>
<td>@RW6 RW2</td>
<td>@RW6 RW3</td>
<td>@RW6 RW4</td>
<td>@RW6 RW5</td>
<td>@RW6 RW6</td>
<td>@RW6 RW7</td>
<td>@RW6 RW8</td>
<td>@RW6 RW9</td>
<td>@RW6 RW10</td>
<td>@RW6 RW11</td>
<td>@RW6 RW12</td>
<td>@RW6 RW13</td>
<td>@RW6 RW14</td>
<td></td>
</tr>
<tr>
<td>+F</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
<td>MOVW</td>
</tr>
<tr>
<td></td>
<td>@RW7 RW0</td>
<td>@RW7 RW1</td>
<td>@RW7 RW2</td>
<td>@RW7 RW3</td>
<td>@RW7 RW4</td>
<td>@RW7 RW5</td>
<td>@RW7 RW6</td>
<td>@RW7 RW7</td>
<td>@RW7 RW8</td>
<td>@RW7 RW9</td>
<td>@RW7 RW10</td>
<td>@RW7 RW11</td>
<td>@RW7 RW12</td>
<td>@RW7 RW13</td>
<td>@RW7 RW14</td>
<td></td>
</tr>
<tr>
<td>00</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>A0</td>
<td>B0</td>
<td>C0</td>
<td>D0</td>
<td>E0</td>
<td>F0</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>XCH</td>
<td>XCH</td>
<td>R0</td>
<td>XCH</td>
<td>XCH</td>
<td>R1</td>
<td>XCH</td>
<td>R2</td>
<td>XCH</td>
<td>R3</td>
<td>XCH</td>
<td>R4</td>
<td>XCH</td>
<td>R5</td>
<td>XCH</td>
<td>R6</td>
<td></td>
</tr>
<tr>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td>R7</td>
<td></td>
</tr>
</tbody>
</table>

Table A.9-20: XCH Ri, ea instruction (first byte = 7EH)
<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>XCHW</td>
<td>XCHW</td>
<td>RWO</td>
<td>XCHW</td>
<td>RWI</td>
<td>XCHW</td>
<td>RW0</td>
<td>XCHW</td>
<td>RW2</td>
<td>XCHW</td>
<td>RW4</td>
<td>XCHW</td>
<td>RW6</td>
<td>XCHW</td>
<td>RW7</td>
<td>XCHW</td>
</tr>
<tr>
<td>+1</td>
<td>RWO</td>
<td>@RW0</td>
<td>@RW0</td>
<td>RWO</td>
<td>RW2</td>
<td>RWO</td>
<td>RW4</td>
<td>RWO</td>
<td>RW6</td>
<td>RWO</td>
<td>RW7</td>
<td>RWO</td>
<td>RW7</td>
<td>RWO</td>
<td>RW7</td>
<td>RWO</td>
</tr>
<tr>
<td>+2</td>
<td>RWI</td>
<td>@RW1</td>
<td>@RW1</td>
<td>RWI</td>
<td>RW3</td>
<td>RWI</td>
<td>RW5</td>
<td>RWI</td>
<td>RW7</td>
<td>RWI</td>
<td>RW7</td>
<td>RWI</td>
<td>RW7</td>
<td>RWI</td>
<td>RW7</td>
<td>RWI</td>
</tr>
<tr>
<td>+3</td>
<td>RW0</td>
<td>@RW2</td>
<td>@RW2</td>
<td>RW0</td>
<td>RW4</td>
<td>RW0</td>
<td>RW6</td>
<td>RW0</td>
<td>RW8</td>
<td>RW0</td>
<td>RW8</td>
<td>RW0</td>
<td>RW8</td>
<td>RW0</td>
<td>RW8</td>
<td>RW0</td>
</tr>
<tr>
<td>+4</td>
<td>RW2</td>
<td>@RW3</td>
<td>@RW3</td>
<td>RW2</td>
<td>RW5</td>
<td>RW2</td>
<td>RW7</td>
<td>RW2</td>
<td>RW9</td>
<td>RW2</td>
<td>RW9</td>
<td>RW2</td>
<td>RW9</td>
<td>RW2</td>
<td>RW9</td>
<td>RW2</td>
</tr>
<tr>
<td>+5</td>
<td>RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>RW4</td>
<td>RW6</td>
<td>RW4</td>
<td>RW8</td>
<td>RW4</td>
<td>RW10</td>
<td>RW4</td>
<td>RW10</td>
<td>RW4</td>
<td>RW10</td>
<td>RW4</td>
<td>RW10</td>
<td>RW4</td>
</tr>
<tr>
<td>+6</td>
<td>RW6</td>
<td>@RW5</td>
<td>@RW5</td>
<td>RW6</td>
<td>RW8</td>
<td>RW6</td>
<td>RW10</td>
<td>RW6</td>
<td>RW12</td>
<td>RW6</td>
<td>RW12</td>
<td>RW6</td>
<td>RW12</td>
<td>RW6</td>
<td>RW12</td>
<td>RW6</td>
</tr>
<tr>
<td>+7</td>
<td>RW8</td>
<td>@RW6</td>
<td>@RW6</td>
<td>RW8</td>
<td>RW10</td>
<td>RW8</td>
<td>RW12</td>
<td>RW8</td>
<td>RW14</td>
<td>RW8</td>
<td>RW14</td>
<td>RW8</td>
<td>RW14</td>
<td>RW8</td>
<td>RW14</td>
<td>RW8</td>
</tr>
<tr>
<td>+8</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
<td>@RW0</td>
</tr>
<tr>
<td>+9</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
<td>@RW1</td>
</tr>
<tr>
<td>+A</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
<td>@RW2</td>
</tr>
<tr>
<td>+B</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
<td>@RW3</td>
</tr>
<tr>
<td>+C</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
<td>@RW4</td>
</tr>
<tr>
<td>+D</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
<td>@RW5</td>
</tr>
<tr>
<td>+E</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
<td>@RW6</td>
</tr>
<tr>
<td>+F</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
<td>@RW7</td>
</tr>
</tbody>
</table>
## APPENDIX B Register Index (address list)
### Register Index (address list)

Table B-1 Register Index (address list)  

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000H</td>
<td>PDR0</td>
<td>Port 0 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 0</td>
<td>180</td>
</tr>
<tr>
<td>000001H</td>
<td>PDR1</td>
<td>Port 1 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 1</td>
<td>186</td>
</tr>
<tr>
<td>000002H</td>
<td>PDR2</td>
<td>Port 2 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 2</td>
<td>192</td>
</tr>
<tr>
<td>000003H</td>
<td>PDR3</td>
<td>Port 3 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 3</td>
<td>197</td>
</tr>
<tr>
<td>000004H</td>
<td>PDR4</td>
<td>Port 4 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 4</td>
<td>202</td>
</tr>
<tr>
<td>000005H</td>
<td>PDR5</td>
<td>Port 5 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 5</td>
<td>209</td>
</tr>
<tr>
<td>000006H</td>
<td>PDR6</td>
<td>Port 6 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 6</td>
<td>214</td>
</tr>
<tr>
<td>000007H</td>
<td>PDR7</td>
<td>Port 7 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 7</td>
<td>222</td>
</tr>
<tr>
<td>000008H</td>
<td>PDR8</td>
<td>Port 8 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 8</td>
<td>227</td>
</tr>
<tr>
<td>000009H</td>
<td>PDR9</td>
<td>Port 9 data register</td>
<td>XXXXXXXXX_B</td>
<td>Port 9</td>
<td>232</td>
</tr>
<tr>
<td>00000AH</td>
<td>PDRA</td>
<td>Port A data register</td>
<td>XXXXXXXXX_B</td>
<td>Port A</td>
<td>237</td>
</tr>
<tr>
<td>00000BH</td>
<td>LCDCMR</td>
<td>Common pin switch register</td>
<td>XXXX0000B</td>
<td>LCD controller/Driver</td>
<td>560</td>
</tr>
<tr>
<td>00000CH</td>
<td>OCP4</td>
<td>Output compare register 4 (ch4)</td>
<td>XXXXXXXXX_B</td>
<td>16-bit I/O timer (Output compare unit 1)</td>
<td>303</td>
</tr>
</tbody>
</table>

(Use prohibited)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000H</td>
<td>EIFR</td>
<td>Wake-up interrupted flag register</td>
<td>XXXXXXXX0_B</td>
<td>Wake-up interrupted</td>
<td>456</td>
</tr>
<tr>
<td>000001H</td>
<td>DDR0</td>
<td>Port 0 data direction register</td>
<td>00000000B</td>
<td>Port 0</td>
<td>180</td>
</tr>
<tr>
<td>000001H</td>
<td>DDR1</td>
<td>Port 1 direction register</td>
<td>00000000B</td>
<td>Port 1</td>
<td>186</td>
</tr>
<tr>
<td>000002H</td>
<td>DDR2</td>
<td>Port 2 direction register</td>
<td>00000000B</td>
<td>Port 2</td>
<td>192</td>
</tr>
<tr>
<td>000003H</td>
<td>DDR3</td>
<td>Port 3 direction register</td>
<td>00000000B</td>
<td>Port 3</td>
<td>197</td>
</tr>
<tr>
<td>000004H</td>
<td>DDR4</td>
<td>Port 4 direction register</td>
<td>00000000B</td>
<td>Port 4</td>
<td>202</td>
</tr>
<tr>
<td>000005H</td>
<td>DDR5</td>
<td>Port 5 direction register</td>
<td>XXX000000B</td>
<td>Port 5</td>
<td>209</td>
</tr>
<tr>
<td>000006H</td>
<td>DDR6</td>
<td>Port 6 direction register</td>
<td>00000000B</td>
<td>Port 6</td>
<td>214</td>
</tr>
<tr>
<td>000007H</td>
<td>DDR7</td>
<td>Port 7 direction register</td>
<td>00000000B</td>
<td>Port 7</td>
<td>222</td>
</tr>
<tr>
<td>000008H</td>
<td>DDR8</td>
<td>Port 8 direction register</td>
<td>00000000B</td>
<td>Port 8</td>
<td>227</td>
</tr>
<tr>
<td>000009H</td>
<td>DDR9</td>
<td>Port 9 direction register</td>
<td>00000000B</td>
<td>Port 9</td>
<td>232</td>
</tr>
</tbody>
</table>
### Table B-1 Register Index (address list) (2 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001AH</td>
<td>DDRA</td>
<td>Port A direction register</td>
<td>00000000B</td>
<td>Port A</td>
<td>237</td>
</tr>
<tr>
<td>00001BH</td>
<td>ADER</td>
<td>Analog input enable registers</td>
<td>11111111B</td>
<td>8/10-bit A/D converter</td>
<td>477</td>
</tr>
<tr>
<td>00001CH</td>
<td>OCP5</td>
<td>Output compare register 5 (ch5)</td>
<td>XXXXXXXXXB</td>
<td>16-bit I/O timer (Output compare unit 1)</td>
<td>303</td>
</tr>
<tr>
<td>00001DH</td>
<td></td>
<td></td>
<td>XXXXXXXXXB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001EH</td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001FH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000020H</td>
<td>SMR</td>
<td>Serial mode register</td>
<td>00000000B</td>
<td>Wake-up interrupted</td>
<td>457</td>
</tr>
<tr>
<td>000021H</td>
<td>SCR</td>
<td>Serial control register</td>
<td>00000100B</td>
<td>UART</td>
<td>520</td>
</tr>
<tr>
<td>000022H</td>
<td>SIDR/ SODR</td>
<td>Serial input data register/ Serial output data registers</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>518</td>
</tr>
<tr>
<td>000023H</td>
<td>SSR</td>
<td>Serial status registers</td>
<td>00001X00B</td>
<td></td>
<td>522</td>
</tr>
<tr>
<td>000024H</td>
<td>SMCS1</td>
<td>Serial mode control status register 1 (ch1)</td>
<td>XXXX0000B</td>
<td>I/O Extended serial interface 1</td>
<td>524/525</td>
</tr>
<tr>
<td>000025H</td>
<td></td>
<td></td>
<td>00000010B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000026H</td>
<td>SDR1</td>
<td>Serial data register 1 (ch1)</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>564</td>
</tr>
<tr>
<td>000027H</td>
<td>CDCR</td>
<td>Communication prescaler registers</td>
<td>0XXX1111B</td>
<td>Communication prescaler</td>
<td>562</td>
</tr>
<tr>
<td>000028H</td>
<td>SMCS2</td>
<td>Serial mode control status register 2 (ch2)</td>
<td>XXXX0000B</td>
<td>I/O Extended serial interface 2</td>
<td></td>
</tr>
<tr>
<td>000029H</td>
<td></td>
<td></td>
<td>00000010B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00002AH</td>
<td>SDR2</td>
<td>Serial data register 2 (ch2)</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>566</td>
</tr>
<tr>
<td>00002BH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00002CH</td>
<td>OCS45</td>
<td>Output compare control status register (ch4, ch5)</td>
<td>0000XX00B</td>
<td>16-bit I/O timer (Output compare unit 1)</td>
<td>301</td>
</tr>
<tr>
<td>00002DH</td>
<td></td>
<td></td>
<td>XXXX00000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00002EH</td>
<td>OCS67</td>
<td>Output compare control status register (ch6, ch7)</td>
<td>0000XX00B</td>
<td></td>
<td>301</td>
</tr>
<tr>
<td>00002FH</td>
<td></td>
<td></td>
<td>XXXX00000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000030H</td>
<td>ENIR</td>
<td>DTP/external interrupt enabled registers</td>
<td>00000000B</td>
<td>DTP/external interrupt</td>
<td>437</td>
</tr>
<tr>
<td>000031H</td>
<td>EIRR</td>
<td>DTP/interrupt factor registers</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>436</td>
</tr>
<tr>
<td>000032H</td>
<td>ELVR</td>
<td>Detection level setting register</td>
<td>00000000B</td>
<td></td>
<td>439</td>
</tr>
<tr>
<td>000033H</td>
<td></td>
<td></td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000034H</td>
<td>OCP6</td>
<td>Output compare register 6 (ch6)</td>
<td>XXXXXXXXXB</td>
<td>16-bit I/O timer (Output compare unit 1)</td>
<td>303</td>
</tr>
<tr>
<td>000035H</td>
<td></td>
<td></td>
<td>XXXXXXXXXB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table B-1 Register Index (address list) (3 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>000036H</td>
<td>ADCS</td>
<td>A/D control status registers</td>
<td>00000000B</td>
<td>8/10-bit A/D converter</td>
<td>471</td>
</tr>
<tr>
<td>000037H</td>
<td></td>
<td></td>
<td>00000000B</td>
<td></td>
<td>469</td>
</tr>
<tr>
<td>000038H</td>
<td>ADCR</td>
<td>A/D data register</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>476</td>
</tr>
<tr>
<td>000039H</td>
<td></td>
<td></td>
<td>00001XXXB</td>
<td></td>
<td>474</td>
</tr>
<tr>
<td>00003AH</td>
<td>DADR0</td>
<td>D/A data register 0 (ch0)</td>
<td>XXXXXXXXXB</td>
<td>D/A converter</td>
<td>503</td>
</tr>
<tr>
<td>00003BH</td>
<td>DADR1</td>
<td>D/A data register 1 (ch1)</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>503</td>
</tr>
<tr>
<td>00003CH</td>
<td>DACR0</td>
<td>D/A control register 0 (ch0)</td>
<td>XXXXXXXX0B</td>
<td></td>
<td>502</td>
</tr>
<tr>
<td>00003DH</td>
<td>DACR1</td>
<td>D/A control register 1 (ch1)</td>
<td>XXXXXXXX0B</td>
<td></td>
<td>502</td>
</tr>
<tr>
<td>00003EH</td>
<td>CLKR</td>
<td>Clock output enable register</td>
<td>XXXXX0000B</td>
<td>Clock monitor function</td>
<td>637</td>
</tr>
<tr>
<td>00003FH</td>
<td></td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000040H</td>
<td>PRLLO</td>
<td>PPG0 reload register L</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>363</td>
</tr>
<tr>
<td>000041H</td>
<td>PRLH0</td>
<td>PPG0 reload register H</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>363</td>
</tr>
<tr>
<td>000042H</td>
<td>PRLL1</td>
<td>PPG1 reload register L</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>363</td>
</tr>
<tr>
<td>000043H</td>
<td>PRLH1</td>
<td>PPG1 reload register H</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>363</td>
</tr>
<tr>
<td>000044H</td>
<td>PPGC0</td>
<td>PPG0 operating mode control register</td>
<td>0X000XX1B</td>
<td>8-/16-bit PPG timer</td>
<td>357</td>
</tr>
<tr>
<td>000045H</td>
<td>PPGC1</td>
<td>PPG1 operating mode control register</td>
<td>0X000001B</td>
<td></td>
<td>359</td>
</tr>
<tr>
<td>000046H</td>
<td>PPGOE</td>
<td>PPG01 output control register</td>
<td>00000000B</td>
<td></td>
<td>361</td>
</tr>
<tr>
<td>000047H</td>
<td></td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000048H</td>
<td>TMCSR0</td>
<td>Timer control status register 0 (ch0)</td>
<td>00000000B</td>
<td>16-bit 16bit reload timer 0</td>
<td>327</td>
</tr>
<tr>
<td>000049H</td>
<td></td>
<td></td>
<td>XXXX0000B</td>
<td></td>
<td>325</td>
</tr>
<tr>
<td>00004AH</td>
<td>TMR0/ TMRLR0</td>
<td>16-bit timer register 0 (ch0)/ 16-bit reloading register 0 (ch0)</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>329</td>
</tr>
<tr>
<td>00004BH</td>
<td></td>
<td></td>
<td>XXXXXXXX0B</td>
<td></td>
<td>330</td>
</tr>
<tr>
<td>00004CH</td>
<td>TMCSR1</td>
<td>Timer control status register 1 (ch1)</td>
<td>00000000B</td>
<td>16-bit 16bit reload timer 1</td>
<td>327</td>
</tr>
<tr>
<td>00004DH</td>
<td></td>
<td></td>
<td>XXXX0000B</td>
<td></td>
<td>325</td>
</tr>
<tr>
<td>00004EH</td>
<td>TMR1/ TMRLR1</td>
<td>16-bit timer register 1 (ch1)/ 16-bit reloading register 1 (ch1)</td>
<td>XXXXXXXXXB</td>
<td></td>
<td>329</td>
</tr>
<tr>
<td>00004FH</td>
<td></td>
<td></td>
<td>XXXXXXXXXB</td>
<td></td>
<td>330</td>
</tr>
</tbody>
</table>
### Table B-1  Register Index (address list)  (4 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>000050H</td>
<td>IPCP0</td>
<td>Input capture register 0 (ch0)</td>
<td>XXXXXXXXX0B</td>
<td>16-bit I/O timer (Input capture)</td>
<td>297</td>
</tr>
<tr>
<td>000051H</td>
<td>IPCP1</td>
<td>Input capture register 1 (ch1)</td>
<td>XXXXXXXXX0B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000052H</td>
<td>ICS01</td>
<td>Input capture control status registers</td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000053H</td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000054H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000055H</td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000056H</td>
<td>TCDT0</td>
<td>Free run timer data register 0 (ch0)</td>
<td>00000000B</td>
<td>16-bit I/O timer (16-bit free run timer 0)</td>
<td>294</td>
</tr>
<tr>
<td>000057H</td>
<td></td>
<td></td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000058H</td>
<td>TCCS0</td>
<td>Free run timer control status register 0 (ch0)</td>
<td>00000000B</td>
<td></td>
<td>292</td>
</tr>
<tr>
<td>000059H</td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00005AH</td>
<td>OCP0</td>
<td>Output compare register 0 (ch0)</td>
<td>XXXXXXXXX0B</td>
<td>16-bit I/O timer (Output compare unit 0)</td>
<td>303</td>
</tr>
<tr>
<td>00005BH</td>
<td>OCP1</td>
<td>Output compare register 1 (ch1)</td>
<td>XXXXXXXXX0B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00005CH</td>
<td>OCP2</td>
<td>Output compare register 2 (ch2)</td>
<td>XXXXXXXXX0B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00005DH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00005EH</td>
<td>OCP3</td>
<td>Output compare register 3 (ch3)</td>
<td>XXXXXXXXX0B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000060H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000061H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000062H</td>
<td>OCS01</td>
<td>Output compare control status register (ch0,ch1)</td>
<td>0000XX00B</td>
<td>16-bit I/O timer (16-bit free run timer 1)</td>
<td>294</td>
</tr>
<tr>
<td>000063H</td>
<td></td>
<td></td>
<td>XXX00000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000064H</td>
<td>OCS23</td>
<td>Output compare control status register (ch2,ch3)</td>
<td>0000XX00B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000065H</td>
<td></td>
<td></td>
<td>XXX00000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000066H</td>
<td>TCDT1</td>
<td>Free run timer data register 1 (ch1)</td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000067H</td>
<td></td>
<td></td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000068H</td>
<td>TCCS1</td>
<td>Free run timer control status register 1 (ch1)</td>
<td>00000000B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000069H</td>
<td></td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00006AH</td>
<td>LCR0</td>
<td>LCD control register 0</td>
<td>00010000B</td>
<td>LCD controller/Driver</td>
<td>592</td>
</tr>
<tr>
<td>00006BH</td>
<td>LCR1</td>
<td>LCD control register 1</td>
<td>00000000B</td>
<td></td>
<td>594</td>
</tr>
</tbody>
</table>
Table B-1  Register Index (address list) (5 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>00006C_H</td>
<td>OCP7</td>
<td>Output compare register 7 (ch7)</td>
<td>XXXXXXXXX_B</td>
<td>16-bit I/O timer (Output compare unit 1)</td>
<td>303</td>
</tr>
<tr>
<td>00006D_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00006E_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00006F_H</td>
<td>ROMM</td>
<td>ROM mirror function selection registers</td>
<td>XXXXXXXXX_B</td>
<td>ROM mirror function selection module</td>
<td>632</td>
</tr>
<tr>
<td>000070_H to 00007F_H</td>
<td>VRAM</td>
<td>Data memory for the LCD display</td>
<td>XXXXXXXXX_B</td>
<td>LCD controller/Driver</td>
<td>596</td>
</tr>
<tr>
<td>000080_H</td>
<td>UDCR0</td>
<td>Up/down count register 0</td>
<td>00000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>398</td>
</tr>
<tr>
<td>000081_H</td>
<td>UDCR1</td>
<td>Up/down count register 1</td>
<td>00000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>398</td>
</tr>
<tr>
<td>000082_H</td>
<td>RCR0</td>
<td>Reload compare register 0</td>
<td>00000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>399</td>
</tr>
<tr>
<td>000083_H</td>
<td>RCR1</td>
<td>Reload compare register 1</td>
<td>00000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>399</td>
</tr>
<tr>
<td>000084_H</td>
<td>CSR0</td>
<td>Counter status register 0</td>
<td>00000000_B</td>
<td></td>
<td>386</td>
</tr>
<tr>
<td>000085_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000086_H</td>
<td>CCR0</td>
<td>Counter control register 0</td>
<td>X0000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>390</td>
</tr>
<tr>
<td>000087_H</td>
<td></td>
<td></td>
<td>00000000_B</td>
<td>8-/16-bit up/down counter/timer</td>
<td>388</td>
</tr>
<tr>
<td>000088_H</td>
<td>CSR1</td>
<td>Counter status register 1</td>
<td>00000000_B</td>
<td></td>
<td>392</td>
</tr>
<tr>
<td>000089_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00008A_H</td>
<td>CCR1</td>
<td>Counter control register 1</td>
<td>X0000000_B</td>
<td>8-/16-bit up/down counter/timer 0,1</td>
<td>396</td>
</tr>
<tr>
<td>00008B_H</td>
<td></td>
<td></td>
<td>X0000000_B</td>
<td>8-/16-bit up/down counter/timer 0,1</td>
<td>394</td>
</tr>
<tr>
<td>00008C_H</td>
<td>RDR0</td>
<td>Port 0 Input resistance register</td>
<td>00000000_B</td>
<td>Port 0</td>
<td>180</td>
</tr>
<tr>
<td>00008D_H</td>
<td>RDR1</td>
<td>Port 1 Input resistance register</td>
<td>00000000_B</td>
<td>Port 1</td>
<td>186</td>
</tr>
<tr>
<td>00008E_H</td>
<td>RDR4</td>
<td>Port 4 Input resistance register</td>
<td>00000000_B</td>
<td>Port 4</td>
<td>202</td>
</tr>
<tr>
<td>00008F_H to 00009D_H</td>
<td>PACSR</td>
<td>Address detection control register</td>
<td>00000000_B</td>
<td>Address match detecting function</td>
<td>618</td>
</tr>
<tr>
<td>00009E_H</td>
<td>DIRR</td>
<td>Delayed interrupt request output/release register</td>
<td>XXXXXXXX0_B</td>
<td>Delayed interrupt generation module</td>
<td>426</td>
</tr>
<tr>
<td>00009F_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table B-1 Register Index (address list) (6 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000A0H</td>
<td>LPMCR</td>
<td>Low-power Consumption Mode control register</td>
<td>00011000B</td>
<td>Low-power consumption Mode</td>
<td>141</td>
</tr>
<tr>
<td>0000A1H</td>
<td>CKSCR</td>
<td>Clock selection register</td>
<td>11111100B</td>
<td>Clock</td>
<td>126</td>
</tr>
<tr>
<td>0000A2H to 0000A7H</td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000A8H</td>
<td>WDTC</td>
<td>Watchdog timer control register</td>
<td>XXXXXXXX_B</td>
<td>Watchdog timer</td>
<td>262</td>
</tr>
<tr>
<td>0000A9H</td>
<td>TBTC</td>
<td>Timebase timer control registers</td>
<td>1XX00000B</td>
<td>Timebase timer</td>
<td>247</td>
</tr>
<tr>
<td>0000AAH</td>
<td>WTC</td>
<td>Watch timer control register</td>
<td>1X000000B</td>
<td>Watch timer</td>
<td>275</td>
</tr>
<tr>
<td>0000ABH to 00004DH</td>
<td>(Use prohibited)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000AEH</td>
<td>FMCS</td>
<td>Flash memory control status register (FMCS)</td>
<td>000X0000B</td>
<td>1-Mbit flash Memory</td>
<td>642</td>
</tr>
<tr>
<td>0000AFH</td>
<td>ICR00</td>
<td>Interrupt control register 00</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B0H</td>
<td>ICR01</td>
<td>Interrupt control register 01</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B1H</td>
<td>ICR02</td>
<td>Interrupt control register 02</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B2H</td>
<td>ICR03</td>
<td>Interrupt control register 03</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B3H</td>
<td>ICR04</td>
<td>Interrupt control register 04</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B4H</td>
<td>ICR05</td>
<td>Interrupt control register 05</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B5H</td>
<td>ICR06</td>
<td>Interrupt control register 06</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B6H</td>
<td>ICR07</td>
<td>Interrupt control register 07</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B7H</td>
<td>ICR08</td>
<td>Interrupt control register 08</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B8H</td>
<td>ICR09</td>
<td>Interrupt control register 09</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000B9H</td>
<td>ICR10</td>
<td>Interrupt control register 10</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BAH</td>
<td>ICR11</td>
<td>Interrupt control register 11</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BBH</td>
<td>ICR12</td>
<td>Interrupt control register 12</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BCH</td>
<td>ICR13</td>
<td>Interrupt control register 13</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BDH</td>
<td>ICR14</td>
<td>Interrupt control register 14</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BEH</td>
<td>ICR15</td>
<td>Interrupt control register 15</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000BFH</td>
<td>ICR16</td>
<td>Interrupt control register 16</td>
<td>00000111B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Notes
- **Address**
- **Registers Abbreviation**
- **Register Name**
- **Reset Value**
- **Resource Name**
- **Page Number**
### Table B-1 Register Index (address list) (7 / 7)

<table>
<thead>
<tr>
<th>Address</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Resource Name</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000C0H to 0000FFH</td>
<td></td>
<td>(External data area) *1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>000100H to 001FFFFH</td>
<td></td>
<td>ROM Area *2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>001FFFFH to 01FEFH</td>
<td></td>
<td>(Reserved area) *3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>001FF0H</td>
<td>PADR0</td>
<td>Program address detection register 0 (lower)</td>
<td>XXXXXXXXXXXB</td>
<td>Address match detection function</td>
<td>620</td>
</tr>
<tr>
<td>001FF1H</td>
<td>PADR0</td>
<td>Program address detection register 0 (middle)</td>
<td>XXXXXXXXXXXB</td>
<td></td>
<td>620</td>
</tr>
<tr>
<td>001FF2H</td>
<td>PADR0</td>
<td>Program address detection register 0 (upper)</td>
<td>XXXXXXXXXXXB</td>
<td></td>
<td>620</td>
</tr>
<tr>
<td>001FF3H</td>
<td>PADR1</td>
<td>Program address detection register 1 (lower)</td>
<td>XXXXXXXXXXXB</td>
<td></td>
<td>620</td>
</tr>
<tr>
<td>001FF4H</td>
<td>PADR1</td>
<td>Program address detection register 1 (middle)</td>
<td>XXXXXXXXXXXB</td>
<td></td>
<td>620</td>
</tr>
<tr>
<td>001FF5H</td>
<td>PADR1</td>
<td>Program address detection register 1 (upper)</td>
<td>XXXXXXXXXXXB</td>
<td></td>
<td>620</td>
</tr>
<tr>
<td>001FF6H to 001FFFH</td>
<td></td>
<td>(Reserved area) *3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Explanation of reset value**

0: Reset value of this bit is 0.
1: Reset value of this bit is 1.
X: Reset value of this bit is undefined.

*1: Only this area out of the area at the address following 0000FFH is used for external access. The access to this address is processed as the external I/O.

*2: For more information on "(RAM Area)", refer to "3.1.2 Memory Map".

*3: Do not write the data to (Reserved area).

*4: "(Used area of the system)" is where the evaluation tool registers are set up.
## APPENDIX C  Register Index (List by Peripheral Function)

### Table C-1  Register Index (List by Peripheral Function) (1 / 6)

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Address</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt controller</td>
<td>ICR00</td>
<td>Interrupt control register 00</td>
<td>00000111B</td>
<td>0000B0H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR01</td>
<td>Interrupt control register 01</td>
<td>00000111B</td>
<td>0000B1H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR02</td>
<td>Interrupt control register 02</td>
<td>00000111B</td>
<td>0000B2H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR03</td>
<td>Interrupt control register 03</td>
<td>00000111B</td>
<td>0000B3H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR04</td>
<td>Interrupt control register 04</td>
<td>00000111B</td>
<td>0000B4H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR05</td>
<td>Interrupt control register 05</td>
<td>00000111B</td>
<td>0000B5H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR06</td>
<td>Interrupt control register 06</td>
<td>00000111B</td>
<td>0000B6H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR07</td>
<td>Interrupt control register 07</td>
<td>00000111B</td>
<td>0000B7H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR08</td>
<td>Interrupt control register 08</td>
<td>00000111B</td>
<td>0000B8H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR09</td>
<td>Interrupt control register 09</td>
<td>00000111B</td>
<td>0000B9H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR10</td>
<td>Interrupt control register 10</td>
<td>00000111B</td>
<td>0000BAH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR11</td>
<td>Interrupt control register 11</td>
<td>00000111B</td>
<td>0000BBH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR12</td>
<td>Interrupt control register 12</td>
<td>00000111B</td>
<td>0000BCH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR13</td>
<td>Interrupt control register 13</td>
<td>00000111B</td>
<td>0000BDH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR14</td>
<td>Interrupt control register 14</td>
<td>00000111B</td>
<td>0000BEH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICR15</td>
<td>Interrupt control register 15</td>
<td>00000111B</td>
<td>0000BFH</td>
<td></td>
</tr>
<tr>
<td>Clock</td>
<td>CKSCR</td>
<td>Clock selection register</td>
<td>11111100B</td>
<td>0000A1H</td>
<td>126</td>
</tr>
<tr>
<td>Low-power Consumption Mode</td>
<td>LPMCR</td>
<td>Low-power Consumption Mode register</td>
<td>00110000B</td>
<td>0000A0H</td>
<td>141</td>
</tr>
<tr>
<td>Port 0</td>
<td>PDR0</td>
<td>Port 0 data register</td>
<td>XXXXXXXX_B</td>
<td>000000H</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>DDR0</td>
<td>Port 0 data direction register</td>
<td>00000000_B</td>
<td>0000010H</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>RDR0</td>
<td>Port 0 input resistance register</td>
<td>00000000_B</td>
<td>000008CH</td>
<td>180</td>
</tr>
<tr>
<td>Port 1</td>
<td>PDR1</td>
<td>Port 1 data register</td>
<td>XXXXXXXX_B</td>
<td>000001H</td>
<td>186</td>
</tr>
<tr>
<td></td>
<td>DDR1</td>
<td>Port 1 direction register</td>
<td>00000000_B</td>
<td>0000011H</td>
<td>186</td>
</tr>
<tr>
<td></td>
<td>RDR1</td>
<td>Port 1 input resistance register</td>
<td>00000000_B</td>
<td>000008DH</td>
<td>186</td>
</tr>
<tr>
<td>Port 2</td>
<td>PDR2</td>
<td>Port 2 data register</td>
<td>XXXXXXXX_B</td>
<td>000002H</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>DDR2</td>
<td>Port 2 direction register</td>
<td>00000000_B</td>
<td>0000012H</td>
<td>192</td>
</tr>
</tbody>
</table>
Table C-1 Register Index (List by Peripheral Function) (2 / 6)

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Address</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3</td>
<td>PDR3</td>
<td>Port 3 data register</td>
<td>XXXXXXXX_B</td>
<td>000003_H</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td>DDR3</td>
<td>Port 3 direction register</td>
<td>00000000_B</td>
<td>000013_H</td>
<td>197</td>
</tr>
<tr>
<td>Port 4</td>
<td>PDR4</td>
<td>Port 4 data register</td>
<td>XXXXXXXX_B</td>
<td>000004_H</td>
<td>202</td>
</tr>
<tr>
<td></td>
<td>DDR4</td>
<td>Port 4 direction register</td>
<td>00000000_B</td>
<td>000014_H</td>
<td>202</td>
</tr>
<tr>
<td></td>
<td>RDR4</td>
<td>Port 4 input resistance register</td>
<td>00000000_B</td>
<td>00008E_H</td>
<td>202</td>
</tr>
<tr>
<td>Port 5</td>
<td>PDR5</td>
<td>Port 5 data register</td>
<td>XXXXXXXX_B</td>
<td>000005_H</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td>DDR5</td>
<td>Port 5 direction register</td>
<td>XXX00000_B</td>
<td>000015_H</td>
<td>209</td>
</tr>
<tr>
<td>Port 6</td>
<td>PDR6</td>
<td>Port 6 data register</td>
<td>XXXXXXXX_B</td>
<td>000006_H</td>
<td>214</td>
</tr>
<tr>
<td></td>
<td>DDR6</td>
<td>Port 6 direction register</td>
<td>00000000_B</td>
<td>000016_H</td>
<td>214</td>
</tr>
<tr>
<td>Port 7</td>
<td>PDR7</td>
<td>Port 7 data register</td>
<td>XXXXXXXX_B</td>
<td>000007_H</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td>DDR7</td>
<td>Port 7 direction register</td>
<td>00000000_B</td>
<td>000017_H</td>
<td>222</td>
</tr>
<tr>
<td>Port 8</td>
<td>PDR8</td>
<td>Port 8 data register</td>
<td>XXXXXXXX_B</td>
<td>000008_H</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td>DDR8</td>
<td>Port 8 direction register</td>
<td>00000000_B</td>
<td>000018_H</td>
<td>227</td>
</tr>
<tr>
<td>Port 9</td>
<td>PDR9</td>
<td>Port 9 data register</td>
<td>XXXXXXXX_B</td>
<td>000009_H</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td>DDR9</td>
<td>Port 9 direction register</td>
<td>00000000_B</td>
<td>000019_H</td>
<td>232</td>
</tr>
<tr>
<td>Port A</td>
<td>PDRA</td>
<td>Port A data register</td>
<td>XXXXXXXX_B</td>
<td>00000A_H</td>
<td>237</td>
</tr>
<tr>
<td></td>
<td>DDRA</td>
<td>Port A direction register</td>
<td>00000000_B</td>
<td>00001A_H</td>
<td>237</td>
</tr>
<tr>
<td>Timebase timer</td>
<td>TBTC</td>
<td>Timebase timer control registers</td>
<td>1XX00000_B</td>
<td>0000A9_H</td>
<td>247</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>WDTC</td>
<td>Watchdog timer control register</td>
<td>XXXXXXXX_B</td>
<td>0000A8_H</td>
<td>262</td>
</tr>
<tr>
<td>Watch timer</td>
<td>WTC</td>
<td>Watch timer control register</td>
<td>1X000000_B</td>
<td>0000AA_H</td>
<td>275</td>
</tr>
<tr>
<td>16-bit I/O timer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(16-bit free run timer)</td>
<td>TCCS0</td>
<td>Free run timer control status register 0 (ch0)</td>
<td>00000000_B</td>
<td>000058_H</td>
<td>292</td>
</tr>
<tr>
<td></td>
<td>TCCS1</td>
<td>Free run timer control status register 1 (ch1)</td>
<td>00000000_B</td>
<td>000068_H</td>
<td>292</td>
</tr>
<tr>
<td></td>
<td>TCDT0</td>
<td>Free run timer data register 0 (ch0)</td>
<td>00000000_B</td>
<td>000056_H</td>
<td>294</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000_B</td>
<td>000057_H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TCDT1</td>
<td>Free run timer data register 1 (ch1)</td>
<td>00000000_B</td>
<td>000066_H</td>
<td>294</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000_B</td>
<td>000067_H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ICS01</td>
<td>Input capture control status registers</td>
<td>00000000_B</td>
<td>000054_H</td>
<td>295</td>
</tr>
<tr>
<td>Resource Name</td>
<td>Registers Abbreviation</td>
<td>Register Name</td>
<td>Reset Value</td>
<td>Address</td>
<td>Page Number</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------</td>
<td>---------------</td>
<td>-------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>IPCP0</td>
<td>Input capture register 0 (ch0)</td>
<td></td>
<td>XXXXXXXXX_B 000050_H</td>
<td>XXXXXXXXX_B 000051_H</td>
<td>297</td>
</tr>
<tr>
<td>IPCP1</td>
<td>Input capture register 1 (ch1)</td>
<td></td>
<td>XXXXXXXXX_B 000052_H</td>
<td>XXXXXXXXX_B 000053_H</td>
<td>297</td>
</tr>
<tr>
<td>OCS01</td>
<td>Output compare control status (ch0,ch1)</td>
<td></td>
<td>XXX000000_B 000063_H</td>
<td>0000XX00_B 000062_H</td>
<td>298</td>
</tr>
<tr>
<td>OCS23</td>
<td>Output compare control status (ch2,ch3)</td>
<td></td>
<td>XXX000000_B 000065_H</td>
<td>0000XX00_B 000064_H</td>
<td>301</td>
</tr>
<tr>
<td>OCS45</td>
<td>Output compare control status (ch4,ch5)</td>
<td></td>
<td>XXX000000_B 00002D_H</td>
<td>0000XX00_B 00002C_H</td>
<td>301</td>
</tr>
<tr>
<td>OCS67</td>
<td>Output compare control status (ch6,ch7)</td>
<td></td>
<td>XXX000000_B 00002F_H</td>
<td>0000XX00_B 00002E_H</td>
<td>301</td>
</tr>
<tr>
<td>OCP0</td>
<td>Output compare register 0 (ch0)</td>
<td></td>
<td>XXXXXXXXX_B 00005A_H</td>
<td>XXXXXXXXX_B 000055_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP1</td>
<td>Output compare register 1 (ch1)</td>
<td></td>
<td>XXXXXXXXX_B 00005C_H</td>
<td>XXXXXXXXX_B 00005D_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP2</td>
<td>Output compare register 2 (ch2)</td>
<td></td>
<td>XXXXXXXXX_B 00005E_H</td>
<td>XXXXXXXXX_B 00005F_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP3</td>
<td>Output compare register 3 (ch3)</td>
<td></td>
<td>XXXXXXXXX_B 000060_H</td>
<td>XXXXXXXXX_B 000061_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP4</td>
<td>Output compare register 4 (ch4)</td>
<td></td>
<td>XXXXXXXXX_B 00000C_H</td>
<td>XXXXXXXXX_B 00000D_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP5</td>
<td>Output compare register 5 (ch5)</td>
<td></td>
<td>XXXXXXXXX_B 00001C_H</td>
<td>XXXXXXXXX_B 00001D_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP6</td>
<td>Output compare register 6 (ch6)</td>
<td></td>
<td>XXXXXXXXX_B 000034_H</td>
<td>XXXXXXXXX_B 000035_H</td>
<td>303</td>
</tr>
<tr>
<td>OCP7</td>
<td>Output compare register 7 (ch7)</td>
<td></td>
<td>XXXXXXXXX_B 000006_H</td>
<td>XXXXXXXXX_B 00000D_H</td>
<td>303</td>
</tr>
</tbody>
</table>

*16-bit I/O timer (Output compare)*
<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Address</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>16-bit reload timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMCSR0 (ch0)</td>
<td>TMCSR0</td>
<td>Timer control status register 0</td>
<td>XXXX0000B</td>
<td>000049F</td>
<td>325</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000B</td>
<td>000048F</td>
<td>327</td>
</tr>
<tr>
<td>TMCSR1 (ch1)</td>
<td>TMCSR1</td>
<td>Timer control status register 1</td>
<td>XXXX0000B</td>
<td>00004DF</td>
<td>325</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000B</td>
<td>00004CF</td>
<td>327</td>
</tr>
<tr>
<td>TMR0/TMR1 (ch0)</td>
<td>TMR0/TMRLR0</td>
<td>16-bit timer register 0/16-bit</td>
<td>XXXXXXXXXB</td>
<td>00004AF</td>
<td>329</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reloading register 0/16-bit</td>
<td>XXXXXXXXXB</td>
<td>00004BF</td>
<td>330</td>
</tr>
<tr>
<td>TMR1/TMRLR1 (ch1)</td>
<td>TMR1/TMRLR1</td>
<td>16-bit timer register 1/16-bit</td>
<td>XXXXXXXXXB</td>
<td>00004EF</td>
<td>329</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reloading register 1/16-bit</td>
<td>XXXXXXXXXB</td>
<td>00004FF</td>
<td>330</td>
</tr>
<tr>
<td><strong>8-/16-bit PPG timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPGC0</td>
<td>PPGC0</td>
<td>PPG0 operating mode registers</td>
<td>0X000XX1B</td>
<td>000044F</td>
<td>357</td>
</tr>
<tr>
<td>PPGC1</td>
<td>PPGC1</td>
<td>PPG1 operating mode register</td>
<td>0X000001B</td>
<td>000045F</td>
<td>359</td>
</tr>
<tr>
<td>PPGOE</td>
<td>PPGOE</td>
<td>PPG01 output control register</td>
<td>00000000B</td>
<td>000046F</td>
<td>361</td>
</tr>
<tr>
<td>PRLLO</td>
<td>PRLLO</td>
<td>PPG0 reload register L</td>
<td>XXXXXXXXXB</td>
<td>000040F</td>
<td>363</td>
</tr>
<tr>
<td>PRLH0</td>
<td>PRLH0</td>
<td>PPG0 reload register H</td>
<td>XXXXXXXXXB</td>
<td>000041F</td>
<td>363</td>
</tr>
<tr>
<td>PRLL1</td>
<td>PRLL1</td>
<td>PPG1 reload register L</td>
<td>XXXXXXXXXB</td>
<td>000042F</td>
<td>363</td>
</tr>
<tr>
<td>PRLH1</td>
<td>PRLH1</td>
<td>PPG1 reload register H</td>
<td>XXXXXXXXXB</td>
<td>000043F</td>
<td>363</td>
</tr>
<tr>
<td><strong>8-/16-bit up/down counter/timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSR0</td>
<td>CSR0</td>
<td>Counter status register 0</td>
<td>00000000B</td>
<td>000084F</td>
<td>386</td>
</tr>
<tr>
<td>CCR0</td>
<td>CCR0</td>
<td>Counter control register 0</td>
<td>00000000B</td>
<td>000087F</td>
<td>388</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X000000B</td>
<td>000086F</td>
<td>390</td>
</tr>
<tr>
<td>CSR1</td>
<td>CSR1</td>
<td>Counter status register 1</td>
<td>00000000B</td>
<td>000088F</td>
<td>392</td>
</tr>
<tr>
<td>CCR1</td>
<td>CCR1</td>
<td>Counter control register 1</td>
<td>X0000000B</td>
<td>00008BF</td>
<td>394</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X0000000B</td>
<td>00008AF</td>
<td>396</td>
</tr>
<tr>
<td>UDCR0</td>
<td>UDCR0</td>
<td>Up/down count register 0</td>
<td>00000000B</td>
<td>000080F</td>
<td>398</td>
</tr>
<tr>
<td>UDCR1</td>
<td>UDCR1</td>
<td>Up/down count register 1</td>
<td>00000000B</td>
<td>000081F</td>
<td>398</td>
</tr>
<tr>
<td>RCR0</td>
<td>RCR0</td>
<td>Reload compare register 0</td>
<td>00000000B</td>
<td>000082F</td>
<td>399</td>
</tr>
<tr>
<td>RCR1</td>
<td>RCR1</td>
<td>Reload compare register 1</td>
<td>00000000B</td>
<td>000083F</td>
<td>399</td>
</tr>
<tr>
<td><strong>Delayed interrupt generation module</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIRR</td>
<td>DIRR</td>
<td>Delayed interrupt request output/register</td>
<td>XXXXXXXXX0B</td>
<td>00009F</td>
<td>426</td>
</tr>
</tbody>
</table>
Table C-1 Register Index (List by Peripheral Function) (5 / 6)

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Address</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTP/external interrupt</td>
<td>EIRR</td>
<td>DTP/External interrupt Factor register</td>
<td>XXXXXXXX_B</td>
<td>000031_H</td>
<td>436</td>
</tr>
<tr>
<td></td>
<td>ENIR</td>
<td>DTP/external interrupt enabled register</td>
<td>00000000_B</td>
<td>000030_H</td>
<td>437</td>
</tr>
<tr>
<td></td>
<td>ELVR</td>
<td>Detection level setting register</td>
<td>00000000_B</td>
<td>000033_H</td>
<td>438</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000_B</td>
<td>000032_H</td>
<td>439</td>
</tr>
<tr>
<td>Wake-up interrupted</td>
<td>EIFR</td>
<td>Wake-up interrupted flag register</td>
<td>XXXXXXXX0_B</td>
<td>00000F_H</td>
<td>456</td>
</tr>
<tr>
<td></td>
<td>EICR</td>
<td>Wake-up interrupt control Registers</td>
<td>00000000_B</td>
<td>00001F_H</td>
<td>457</td>
</tr>
<tr>
<td>8/10-bit A/D converter</td>
<td>ADCS</td>
<td>A/D control status register</td>
<td>00000000_B</td>
<td>000037_H</td>
<td>469</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00000000_B</td>
<td>000036_H</td>
<td>471</td>
</tr>
<tr>
<td></td>
<td>ADCR</td>
<td>A/D data register</td>
<td>00001XXX_B</td>
<td>000039_H</td>
<td>474</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXX_B</td>
<td>000038_H</td>
<td>476</td>
</tr>
<tr>
<td></td>
<td>ADER</td>
<td>Analog input enable register</td>
<td>11111111_B</td>
<td>00001B_H</td>
<td>477</td>
</tr>
<tr>
<td>D/A converter</td>
<td>DACR0</td>
<td>D/A control register 0 (ch0)</td>
<td>XXXXXXXX0_B</td>
<td>00003C_H</td>
<td>502</td>
</tr>
<tr>
<td></td>
<td>DACR1</td>
<td>D/A control register 1 (ch1)</td>
<td>XXXXXXXX0_B</td>
<td>00003D_H</td>
<td>502</td>
</tr>
<tr>
<td></td>
<td>DADR0</td>
<td>D/A data register 0 (ch0)</td>
<td>XXXXXXXX_B</td>
<td>00003A_H</td>
<td>503</td>
</tr>
<tr>
<td></td>
<td>DADR1</td>
<td>D/A data register 1 (ch1)</td>
<td>XXXXXXXX_B</td>
<td>00003B_H</td>
<td>503</td>
</tr>
<tr>
<td>Communication prescaler</td>
<td>CDCR</td>
<td>Communication prescaler register</td>
<td>0XXX1111_B</td>
<td>000027_H</td>
<td>509</td>
</tr>
<tr>
<td>UART</td>
<td>SCR</td>
<td>Serial control register</td>
<td>00000100_B</td>
<td>000021_H</td>
<td>518</td>
</tr>
<tr>
<td></td>
<td>SMR</td>
<td>Serial mode register</td>
<td>00000000_B</td>
<td>000020_H</td>
<td>520</td>
</tr>
<tr>
<td></td>
<td>SSR</td>
<td>Serial status register</td>
<td>00001X00_B</td>
<td>000023_H</td>
<td>522</td>
</tr>
<tr>
<td></td>
<td>SIDR/ SODR</td>
<td>Serial input data register/ Serial output data register</td>
<td>XXXXXXXX_B</td>
<td>000022_H</td>
<td>524/525</td>
</tr>
<tr>
<td>I/O Extended serial</td>
<td>SMCS1</td>
<td>Serial mode control status register 1 (ch1)</td>
<td>00000010_B</td>
<td>000025_H</td>
<td>560</td>
</tr>
<tr>
<td>interface</td>
<td></td>
<td></td>
<td>XXXX0000_B</td>
<td>000024_H</td>
<td>562</td>
</tr>
<tr>
<td></td>
<td>SMCS2</td>
<td>Serial mode control status register 2 (ch2)</td>
<td>00000010_B</td>
<td>000029_H</td>
<td>560</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>XXXX0000_B</td>
<td>000028_H</td>
<td>562</td>
</tr>
<tr>
<td></td>
<td>SDR1</td>
<td>Serial data register 1 (ch1)</td>
<td>XXXXXXXX_B</td>
<td>000026_H</td>
<td>564</td>
</tr>
<tr>
<td></td>
<td>SDR2</td>
<td>Serial data register 2 (ch2)</td>
<td>XXXXXXXX_B</td>
<td>00002A_H</td>
<td>564</td>
</tr>
</tbody>
</table>
## Table C-1  Register Index (List by Peripheral Function) (6 / 6)

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Registers Abbreviation</th>
<th>Register Name</th>
<th>Reset Value</th>
<th>Address</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD controller/Driver</td>
<td>LCDCMR</td>
<td>Common pin switch register</td>
<td>XXXX0000B</td>
<td>00000BH</td>
<td>588</td>
</tr>
<tr>
<td>LCD controller/Driver</td>
<td>LCR0</td>
<td>LCDC control register 0</td>
<td>00010000B</td>
<td>00006AH</td>
<td>590</td>
</tr>
<tr>
<td>LCD controller/Driver</td>
<td>LCR1</td>
<td>LCDC control register 1</td>
<td>00000000B</td>
<td>00006BH</td>
<td>592</td>
</tr>
<tr>
<td>VRAM</td>
<td></td>
<td>Data memory for the LCD display</td>
<td>XXXXXXXXBBBB</td>
<td>000070H to 00007FH</td>
<td>594</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PACSR</td>
<td>Address detection control status registers</td>
<td>0000000B</td>
<td>00009EH</td>
<td>616</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR0</td>
<td>Program address detection register 0 (lower)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF0H</td>
<td>618</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR0</td>
<td>Program address detection register 0 (middle)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF1H</td>
<td>618</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR0</td>
<td>Program address detection register 0 (upper)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF2H</td>
<td>618</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR1</td>
<td>Program address detection register 1 (lower)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF3H</td>
<td>618</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR1</td>
<td>Program address detection register 1 (middle)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF4H</td>
<td>618</td>
</tr>
<tr>
<td>Address match detecting function</td>
<td>PADR1</td>
<td>Program address detection register 1 (upper)</td>
<td>XXXXXXXXBBBB</td>
<td>001FF5H</td>
<td>618</td>
</tr>
<tr>
<td>ROM mirror function selection module</td>
<td>ROMM</td>
<td>ROM mirror function selection register</td>
<td>XXXXXXXXBBBB</td>
<td>00006FH</td>
<td>630</td>
</tr>
<tr>
<td>Clock monitor function</td>
<td>CLKR</td>
<td>Clock output enable register</td>
<td>XXXX0000B</td>
<td>00003EH</td>
<td>635</td>
</tr>
<tr>
<td>1 Mbit flash memory</td>
<td>FMCS</td>
<td>Flash memory control status register</td>
<td>000X0000B</td>
<td>0000AEH</td>
<td>640</td>
</tr>
</tbody>
</table>

Explanation of reset value
0: Reset value of this bit is 0.
1: Reset value of this bit is 1.
X: Reset value of this bit is undefined.
## APPENDIX D  Pin Function Index

### Pin Function Index

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Peripheral resource name and function name</th>
<th>Functional description</th>
<th>Block figure page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P31</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>195</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td>CKOT</td>
<td></td>
<td>Clock monitor function Output pin</td>
<td>634</td>
<td>633</td>
</tr>
<tr>
<td>2 to 5</td>
<td>P32 to P35</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>195</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td>OUT0 to OUT3</td>
<td></td>
<td>Output compare channel 0 to 3</td>
<td>290</td>
<td>288</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6/7</td>
<td>P36/P37</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>195</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td>PPG00/PPG01</td>
<td></td>
<td>Output pins for PPG timers 0</td>
<td>356</td>
<td>352</td>
</tr>
<tr>
<td>8,54,94</td>
<td>Vcc</td>
<td></td>
<td>Power (5 V) input pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/10</td>
<td>P40/P41</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>PPG10/PPG11</td>
<td></td>
<td>Output pins for PPG timers 1</td>
<td>356</td>
<td>352</td>
</tr>
<tr>
<td>11</td>
<td>P42</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>203</td>
</tr>
<tr>
<td></td>
<td>SIN0</td>
<td></td>
<td>UART serial data input pin</td>
<td>517</td>
<td>514</td>
</tr>
<tr>
<td>12</td>
<td>P43</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>SOT0</td>
<td></td>
<td>UART serial data output pin</td>
<td>517</td>
<td>514</td>
</tr>
<tr>
<td>13</td>
<td>P44</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>SCK0</td>
<td></td>
<td>Serial clock input/output pin for UART</td>
<td>517</td>
<td>514</td>
</tr>
<tr>
<td>14</td>
<td>P45</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>SIN1</td>
<td></td>
<td>I/O extended serial channel 1 Serial data input pin</td>
<td>559</td>
<td>557</td>
</tr>
<tr>
<td>15</td>
<td>P46</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>SOT1</td>
<td></td>
<td>I/O extended serial channel 1 Serial data output pin</td>
<td>559</td>
<td>557</td>
</tr>
<tr>
<td>16</td>
<td>P47</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>200</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>SCK1</td>
<td></td>
<td>I/O extended serial channel 1 Serial clock input/output pin</td>
<td>559</td>
<td>557</td>
</tr>
<tr>
<td>17 to 24</td>
<td>SEGO to SEG7</td>
<td>F</td>
<td>LCD controller/driver Segment output</td>
<td>587</td>
<td>585</td>
</tr>
<tr>
<td>25 to 32</td>
<td>PA0 to PA7</td>
<td>L</td>
<td>General-purpose I/O ports</td>
<td>235</td>
<td>236</td>
</tr>
<tr>
<td></td>
<td>SEG8 to SEG15</td>
<td></td>
<td>LCD controller/driver Segment output</td>
<td>587</td>
<td>585</td>
</tr>
<tr>
<td>33,63,91,</td>
<td>Vss</td>
<td></td>
<td>Power (0 V) input pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>119</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pin Number</td>
<td>Pin Name</td>
<td>Circuit Type</td>
<td>Peripheral resource name and function name</td>
<td>Functional description page</td>
<td>Block figure page</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>--------------</td>
<td>---------------------------------------------</td>
<td>----------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>34</td>
<td>C</td>
<td>G</td>
<td>Pin capacitance</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>35</td>
<td>P50</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>206</td>
<td>207</td>
</tr>
<tr>
<td></td>
<td>SIN2</td>
<td></td>
<td>I/O extended serial channel 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Serial data input pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AIN1</td>
<td></td>
<td>8-/16-bit up/down counter channel 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count timing input (A) pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>P51</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>206</td>
<td>207</td>
</tr>
<tr>
<td></td>
<td>SOT2</td>
<td></td>
<td>I/O extended serial channel 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Serial data output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BIN1</td>
<td></td>
<td>8-/16-bit up/down counter channel 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count timing input (B) pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>P52</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>206</td>
<td>207</td>
</tr>
<tr>
<td></td>
<td>SCK2</td>
<td></td>
<td>I/O extended serial channel 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Serial clock input/output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZIN1</td>
<td></td>
<td>8-/16-bit up/down counter channel 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count control input (Z) pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>DVCC</td>
<td>H</td>
<td>D/A converter standard power (Vref +) input pin</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>39</td>
<td>DVSS</td>
<td>H</td>
<td>D/A converter GND input pin</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>40/41</td>
<td>P53/P54</td>
<td>I</td>
<td>General-purpose I/O ports</td>
<td>206</td>
<td>207</td>
</tr>
<tr>
<td></td>
<td>DA0/DA1</td>
<td></td>
<td>D/A converter Analog output pin</td>
<td>501</td>
<td>499</td>
</tr>
<tr>
<td>42</td>
<td>AVcc</td>
<td>H</td>
<td>Vcc input pin for A/D converter</td>
<td>-</td>
<td>465</td>
</tr>
<tr>
<td>43</td>
<td>AVRH</td>
<td>J</td>
<td>Power supply (Vref+) input pin for A/D converter</td>
<td>-</td>
<td>465</td>
</tr>
<tr>
<td>44</td>
<td>AVRL</td>
<td>H</td>
<td>Power supply (Vref-) input pin for A/D converter</td>
<td>-</td>
<td>465</td>
</tr>
<tr>
<td>45</td>
<td>AVss</td>
<td>H</td>
<td>A/D converter Vss power supply input pin</td>
<td>-</td>
<td>465</td>
</tr>
<tr>
<td>46 to 53</td>
<td>P60 to P67</td>
<td>K</td>
<td>General-purpose I/O port</td>
<td>212</td>
<td>213</td>
</tr>
<tr>
<td></td>
<td>AN0 to AN7</td>
<td></td>
<td>Analog input pins for A/D converter</td>
<td>468</td>
<td>465</td>
</tr>
<tr>
<td>55</td>
<td>P70</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>218</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>TIN0</td>
<td></td>
<td>Event input pin for reload timer 0</td>
<td>324</td>
<td>322</td>
</tr>
<tr>
<td></td>
<td>OUT4</td>
<td></td>
<td>Output compare channel 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>P71</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>218</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>TOT0</td>
<td></td>
<td>Event output pin for reload timer 0</td>
<td>324</td>
<td>322</td>
</tr>
<tr>
<td></td>
<td>OUT5</td>
<td></td>
<td>Output compare channel 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event output pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table D-1 Pin Function Index (3/4)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Peripheral resource name and function name</th>
<th>Functional description page</th>
<th>Block figure page</th>
</tr>
</thead>
<tbody>
<tr>
<td>57</td>
<td>P72</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>218</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>TIN1</td>
<td>E</td>
<td>Event input pin for reload timer 1</td>
<td>324</td>
<td>322</td>
</tr>
<tr>
<td></td>
<td>OUT6</td>
<td>E</td>
<td>Output compare 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>P73</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>218</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>TOT1</td>
<td>E</td>
<td>Event output pin for reload timer 1</td>
<td>324</td>
<td>322</td>
</tr>
<tr>
<td></td>
<td>OUT7</td>
<td>E</td>
<td>Output compare 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event output pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59 to 62</td>
<td>JP74 to P77</td>
<td>L</td>
<td>General-purpose I/O ports</td>
<td>218</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>COM0 to COM3</td>
<td>L</td>
<td>LCD controller/driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Output Pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64 to 71</td>
<td>P80 to P87</td>
<td>L</td>
<td>General-purpose I/O port</td>
<td>225</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td>SEG16 to SEG23</td>
<td>L</td>
<td>LCD controller/driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Segment output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72, 75 to 81</td>
<td>P90 to P97</td>
<td>M</td>
<td>General-purpose I/O ports</td>
<td>231</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td>SEG24 to SEG31</td>
<td>M</td>
<td>LCD controller/driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Segment output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74/73</td>
<td>X0A/X1A</td>
<td>B</td>
<td>Low-speed oscillation pin</td>
<td>135</td>
<td>123</td>
</tr>
<tr>
<td>82 to 85</td>
<td>V0 to V3</td>
<td>N</td>
<td>LCD controller/driver Reference power pin</td>
<td>599</td>
<td>599</td>
</tr>
<tr>
<td>86</td>
<td>HST</td>
<td>C</td>
<td>Hardware standby input pin</td>
<td>158</td>
<td>139</td>
</tr>
<tr>
<td>87 to 89</td>
<td>MD2,MD1,MD0</td>
<td>C</td>
<td>Input pin for specifying CPU operation mode</td>
<td>168</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>RST</td>
<td>C</td>
<td>External reset input pin</td>
<td>108</td>
<td>112</td>
</tr>
<tr>
<td>93/92</td>
<td>X0/X1</td>
<td>A</td>
<td>High-speed oscillation pin</td>
<td>135</td>
<td>123</td>
</tr>
<tr>
<td>95 to 101</td>
<td>P00 to P06</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>176</td>
<td>179</td>
</tr>
<tr>
<td></td>
<td>INTO to INTO6</td>
<td>D</td>
<td>External interrupt input pins</td>
<td>435</td>
<td>433</td>
</tr>
<tr>
<td>102</td>
<td>P07</td>
<td>D</td>
<td>General-purpose I/O ports</td>
<td>178</td>
<td>179</td>
</tr>
<tr>
<td>103 to 110</td>
<td>P10 to P17</td>
<td>D</td>
<td>General-purpose I/O port</td>
<td>184</td>
<td>185</td>
</tr>
<tr>
<td></td>
<td>W10 to W17</td>
<td>D</td>
<td>Wake-up interrupt input pin</td>
<td>455</td>
<td>453</td>
</tr>
<tr>
<td>111/112</td>
<td>P20/P21</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>IN00/IN01</td>
<td>E</td>
<td>Input capture channel 0</td>
<td>290</td>
<td>289</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Trigger input pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>113/114</td>
<td>P22/P23</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>IN10/IN11</td>
<td>E</td>
<td>Input capture channel 1</td>
<td>290</td>
<td>288</td>
</tr>
</tbody>
</table>
## Table D-1  Pin Function Index  (4 / 4)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Pin Name</th>
<th>Circuit Type</th>
<th>Peripheral resource name and function name</th>
<th>Functional description page</th>
<th>Block figure page</th>
</tr>
</thead>
<tbody>
<tr>
<td>115</td>
<td>P24</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>AIN0</td>
<td></td>
<td>8-/16-bit up/down counter channel 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count timing input (A) pin</td>
<td>385</td>
<td>383</td>
</tr>
<tr>
<td>116</td>
<td>P25</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>BIN0</td>
<td></td>
<td>8-/16-bit up/down counter channel 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count timing input (B) pin</td>
<td>385</td>
<td>383</td>
</tr>
<tr>
<td>117</td>
<td>P26</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>ZIN0</td>
<td></td>
<td>8-/16-bit up/down counter channel 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Count control input (Z) pin</td>
<td>385</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>INT7</td>
<td></td>
<td>External interrupt input</td>
<td>435</td>
<td>433</td>
</tr>
<tr>
<td>118</td>
<td>P27</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>ADTG</td>
<td></td>
<td>External trigger input pin for A/D converter</td>
<td>468</td>
<td>465</td>
</tr>
<tr>
<td>120</td>
<td>P30</td>
<td>E</td>
<td>General-purpose I/O ports</td>
<td>195</td>
<td>196</td>
</tr>
</tbody>
</table>
### APPENDIX E Interrupt Vector Index

#### Interrupt vector index

<table>
<thead>
<tr>
<th>Interrupt Number</th>
<th>Interrupt Factor</th>
<th>Interrupt Control Address in Vector Table</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>#08</td>
<td>Reset</td>
<td>-</td>
<td>110</td>
</tr>
<tr>
<td>#09</td>
<td>INT9 instruction</td>
<td>-</td>
<td>89</td>
</tr>
<tr>
<td>#10</td>
<td>Exception processing</td>
<td>-</td>
<td>101</td>
</tr>
<tr>
<td>#11</td>
<td>8/10-bit A/D converter</td>
<td>-</td>
<td>478</td>
</tr>
<tr>
<td>#12</td>
<td>Timebase timer</td>
<td>ICR00</td>
<td>249</td>
</tr>
<tr>
<td>#13</td>
<td>External interrupt 0 (DTP0) External interrupt 1 (DTP1)</td>
<td>ICR01</td>
<td>432</td>
</tr>
<tr>
<td>#14</td>
<td>16-bit free-run timer 0 overflow</td>
<td>ICR02</td>
<td>304</td>
</tr>
<tr>
<td>#15</td>
<td>I/O Extended serial interface 1</td>
<td>ICR03</td>
<td>565</td>
</tr>
<tr>
<td>#16</td>
<td>Wake-up interrupted</td>
<td>ICR04</td>
<td>452</td>
</tr>
<tr>
<td>#17</td>
<td>I/O Extended serial interface 2</td>
<td>ICR05</td>
<td>565</td>
</tr>
<tr>
<td>#18</td>
<td>External interrupt 2 (DTP2) External interrupt 3 (DTP3)</td>
<td>ICR06</td>
<td>364</td>
</tr>
<tr>
<td>#19</td>
<td>8-/16-bit PPG Timer 0 counter borrow</td>
<td>ICR07</td>
<td>492</td>
</tr>
<tr>
<td>#20</td>
<td>External interrupt 4 (DTP4) External interrupt 5 (DTP5)</td>
<td>ICR08</td>
<td>400</td>
</tr>
<tr>
<td>#21</td>
<td>8-/16-bit up/down counter/timer 0 compare match</td>
<td>ICR09</td>
<td>400</td>
</tr>
<tr>
<td>#22</td>
<td>8-/16-bit up/down counter/timer 0 Overflow, Underflow, UP/down, Reverse</td>
<td>ICR10</td>
<td>364</td>
</tr>
<tr>
<td>#23</td>
<td>8-/16-bit PPG Timer 1 Counter borrow</td>
<td>ICR11</td>
<td>492</td>
</tr>
<tr>
<td>#24</td>
<td>External interrupt 6 (DTP6) External interrupt 7 (DTP7)</td>
<td>ICR12</td>
<td>304</td>
</tr>
<tr>
<td>#25</td>
<td>Output compare (Unit 1) ch4,ch5 match</td>
<td>ICR13</td>
<td>304</td>
</tr>
<tr>
<td>#26</td>
<td>Watch timer</td>
<td>ICR14</td>
<td>277</td>
</tr>
<tr>
<td>#27</td>
<td>Output compare (Unit 1) ch6,ch7 match</td>
<td>ICR15</td>
<td>304</td>
</tr>
<tr>
<td>#28</td>
<td>16-bit free-run timer 1 overflow</td>
<td>ICR16</td>
<td>304</td>
</tr>
<tr>
<td>Interrupt Number</td>
<td>Interrupt Factor</td>
<td>Interrupt Control</td>
<td>Address in Vector Table</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>#29</td>
<td>8-/16-bit up/down counter/timer 1 compare match</td>
<td>ICR09</td>
<td>0000B9H</td>
</tr>
<tr>
<td>#30</td>
<td>8-/16-bit up/down counter/timer 1 Overflow, Underflow, UP/down, Reverse</td>
<td>ICR09</td>
<td>0000B9H</td>
</tr>
<tr>
<td>#31</td>
<td>Input capture 0 fetched</td>
<td>ICR10</td>
<td>0000BAH</td>
</tr>
<tr>
<td>#32</td>
<td>Input capture 1 fetched</td>
<td>ICR10</td>
<td>0000BAH</td>
</tr>
<tr>
<td>#33</td>
<td>Output compare (Unit 0) ch0 match</td>
<td>ICR11</td>
<td>0000BBH</td>
</tr>
<tr>
<td>#34</td>
<td>Output compare (Unit 1) ch1 match</td>
<td>ICR11</td>
<td>0000BBH</td>
</tr>
<tr>
<td>#35</td>
<td>Output compare (Unit 0) ch2 match</td>
<td>ICR12</td>
<td>0000BCH</td>
</tr>
<tr>
<td>#36</td>
<td>Output compare (Unit 0) ch3 match</td>
<td>ICR12</td>
<td>0000BCH</td>
</tr>
<tr>
<td>#37</td>
<td>UART receive completion</td>
<td>ICR13</td>
<td>0000BDH</td>
</tr>
<tr>
<td>#38</td>
<td>16-bit reload timer 0</td>
<td>ICR13</td>
<td>0000BDH</td>
</tr>
<tr>
<td>#39</td>
<td>UART transmit completion</td>
<td>ICR14</td>
<td>0000BEH</td>
</tr>
<tr>
<td>#40</td>
<td>16-bit reload timer 1</td>
<td>ICR14</td>
<td>0000BEH</td>
</tr>
<tr>
<td>#41</td>
<td>Flash memory</td>
<td>ICR15</td>
<td>0000BFH</td>
</tr>
<tr>
<td>#42</td>
<td>Delayed interrupt generation module</td>
<td>ICR15</td>
<td>0000BFH</td>
</tr>
</tbody>
</table>
The index follows on the next page.
This is listed in alphabetic order.
INDEX

Numerics

1/2 duty

1/3 bias, 1/2 duty output waveform example ................................................. 606, 609

1/3 bias

1/3 bias, 1/2 duty output waveform example ................................................. 606, 609

1/3 bias, 1/4 duty output waveform example 612

1/4 duty

1/3 bias, 1/4 duty output waveform example ................................................. 612

16-bit Free-run Timer

Block Diagram of 16-bit Free-run Timer ...... 286

Explanation of Operation of 16-bit Free-run Timer ........................................ 306

16-bit I/O timer

Module configuration of 16-bit I/O timer....... 282

Precaution on using 16-bit I/O timer .......... 314

16-bit Input/Output Timer

Block Diagram of 16-bit Input/Output Timer . 284

EI²OS Functions of 16-bit Input/Output Timer ................................................. 305

Functions of 16-bit Input/Output Timer ..... 283

Generation of Interrupt Request from 16-bit Input/Output Timer ................... 291

Interrupts of 16-bit Input/Output Timer .... 304

Interrupts of 16-bit Input/Output Timer and EI²OS ........................................ 305

Pins of 16-bit Input/Output Timer .......... 290

Program Example of 16-bit Input/Output Timer ........ 315

16-bit PPG Output Operation Mode

Setting for 16-bit PPG Output Operation Mode ............................................. 369

16-bit Reload Registers

16-bit Reload Registers (TMRLR0, TMRLR1) ............................................. 330

16-bit Reload Timer

Baud Rate by Internal Timer

(16-bit Reload Timer 0) ....................... 538

Block Diagram for Pins of 16-bit Reload Timer ............................................. 324

Block Diagram of 16-bit Reload Timer .... 322

EI²OS Function of 16-bit Reload Timer .... 331

Generation of Interrupt Request from 16-bit Reload Timer ................................ 324

Interrupts of 16-bit Reload Timer .......... 331

Interrupts of 16-bit Reload Timer and EI²OS ............................................. 331

Operation Modes of 16-bit Reload Timer ..... 320

Pins of 16-bit Reload Timer ...................... 324

Precautions when Using 16-bit Reload Timer ............................................. 342

Setting of 16-bit Reload Timer ............. 332

16-bit Timer Register

16-bit Timer Registers (TMR0, TMR1) ........ 329

Operating State of 16-bit Timer Register .......... 333

Operation as 16-bit Timer Register Underflow ............................................. 335, 340

1Mbit Flash Memory

1 Mbit Flash Memory Program Example ...... 668

Features of 1Mbit Flash Memory ............ 642

Overview of 1M-bit Flash Memory .......... 642

Sector Configuration of 1Mbit Flash memory ............................................. 643

24-bit operand

Linear addressing specified by 24-bit operand ............................................. 36

2-channel Independent Operation Mode

Setting for 8-bit PPG Output 2-channel Independent Operation Mode ............... 366

8 + 8-bit PPG Output Operation Mode

Setting for 8 + 8-bit PPG Output Operation Mode ........................................ 372

8-/10-bit A/D Converter

8-/10-bit A/D Converter Interrupt and EI²OS ............................................. 480

A/D conversion Data Protection Function in 8-/10-bit A/D Converter ............... 489

Block Diagram of 8-/10-bit A/D Converter .... 465

Conversion Modes of 8-/10-bit A/D Converter ............................................. 464

Diagram of Pins of 8-/10-bit A/D Converter ............................................. 468

EI²OS Function of 8-/10-bit A/D Converter ............................................. 480

Functions of the 8-/10-bit A/D converter ............................................. 464

Generation of Interrupt from 8-/10-bit A/D Converter ................................ 468

Pins of 8-/10-bit A/D Converter ............... 468

Precautions when Using 8-/10-bit A/D Converter ........................................ 491

8-/16-bit PPG Timer

Block Diagram for 8-/16-bit PPG Timer 0 ...... 352

Blode Diagram of 8-/16-bit PPG Timer 1 .... 354

Block Diagram of 8-/16-bit PPG Timer Pins ............................................. 356

Detail of Interrupts of 8-/16-bit PPG Timer .... 364

Functions of 8-/16-bit PPG Timer ............... 348
INDEX

Generation of Interrupt from 8-/16-bit PPG Timer ........................................... 356
Interrupts of 8-/16-bit PPG Timer ................................................................. 364
Operation Modes of 8-/16-bit PPG Timer ......................................................... 348
Operation of 8-/16-bit PPG Timer ................................................................. 365
Pins of 8-/16-bit PPG Timer ........................................................................... 356
Precautions when Using 8-/16-bit PPG Timer .................................................. 375
Program Example for 8-/16-bit PPG Timer ...................................................... 377

8-/16-bit timer mode
Program example of the 8-/16-bit timer mode .................................................... 417

8-/16-bit up/down counter/timer
8-/16-bit up/down counter/timer 0 block diagram .............................................. 381
8-/16-bit up/down counter/timer 1 block diagram .............................................. 383
8-/16-bit up/down counter/timer EWS ............................................................. 401
8-/16-bit up/down counter/timer functions ....................................................... 380
8-/16-bit up/down counter/timer interrupt ......................................................... 400
8-/16-bit up/down counter/timer interrupt and EWS ......................................... 401
8-/16-bit up/down counter/timer pin ............................................................... 385
Interrupt requests are generated by the 8-/16-bit up/down counter/timer ........... 385
Precautions for 8-/16-bit up/down counter/timer ................................................ 416

8-bit phase difference count mode
Program example of the 8-bit phase difference count mode ................................ 419

8-bit PPG Output 2-channel Independent Operation Mode
Setting for 8-bit PPG Output 2-channel Independent Operation Mode ............... 366

8-bit up/down count mode
Program example of the program mode of the 8-bit up/down count mode ............ 418

A

A
Accumulator (A) ................................................................. 44

A/D Control Status Register
A/D Control Status Register (High) (ADCS: H) .............................................. 469
A/D Control Status Register (Low) (ADCS: L) .............................................. 472

A/D conversion Data Protection
A/D conversion Data Protection Function in 8-/10-bit A/D Converter ............... 489

A/D Converter
8-/10-bit A/D Converter Interrupt and EWS Function ..................................... 480
A/D conversion Data Protection Function in 8-/10-bit A/D Converter ............... 489
Block Diagram of 8-/10-bit A/D Converter ..................................................... 465
Conversion Modes of 8-/10-bit A/D Converter ................................................ 464
Diagram of Pins of 8-/10-bit A/D Converter ................................................... 468
EWS Function of 8-/10-bit A/D Converter ....................................................... 480
Functions of the 8-/10-bit A/D converter ......................................................... 464
Generation of Interrupt from 8-/10-bit A/D Converter ..................................... 468
Interrupt of A/D Converter ............................................................................ 480
Pins of 8-/10-bit A/D Converter ...................................................................... 468
Precautions when Using 8-/10-bit A/D Converter ............................................ 491

A/D Data Register
A/D Data Register (High) (ADCR: H) ......................................................... 475
A/D Data Register (Low) (ADCR: L) ......................................................... 477

Access
Setting of Each Bank and Data Access ............................................................ 57
Access Space
Bank Registers and Access Space ................................................................. 37
Accumulator
Accumulator (A) ................................................................. 44

ADB
Additional data bank register (ADB) ............................................................... 57
Bank select prefix (PCB, DTB, ADB, and SPB) .................................................. 61

ADCR
A/D Data Register (High) (ADCR: H) ......................................................... 475
A/D Data Register (Low) (ADCR: L) ......................................................... 477

ADCS
A/D Control Status Register (High) (ADCS: H) ............................................ 469
A/D Control Status Register (Low) (ADCS: L) ............................................. 472
Continuous Conversion Mode
(ADCS: MD1, MD0= “10B”) ................................................................. 481
Pause-conversion Mode
(ADCS: MD1, MD0= “11B”) ................................................................. 481
INDEX

Single-shot Conversion Mode (ADCS: MD1,MD0="00B" or "01B") ............... 481
Additional data bank register
  Additional data bank register (ADB) ............... 57
Address Match Detection
  Block Diagram of Address Match Detection Function ............................. 619
  Explanation of Operation of Address Match Detection Function .......... 624
  Overview of Address Match Detection Function .................................. 618
  Program Example of Address Match Detection Function ...................... 630
Addressing
  Addressing .................................................... 688
  Addressing by indirect specification with a 32-bit register ............... 36
  Bank Addressing and Default Space ........................................ 38
  Direct Addressing .............................................. 690
  Indirect Addressing ........................................... 697
  Linear Addressing and Bank Addressing ................................... 35
  Linear addressing specified by 24-bit operand .................................. 36
ADER
  Analog input enable register (ADER) .................................... 478
Analog input enable register
  Analog input enable register (ADER) .................................... 478
Asynchronous Mode
  Operation in Asynchronous Mode .......................................... 543

B
Bank
  Access to FF Bank by ROM Mirroring Function .................................. 632
  Setting of Each Bank and Data Access ............... 57
Bank Addressing
  Bank Addressing and Default Space ........................................ 38
  Linear Addressing and Bank Addressing ....................... 35
Bank Registers
  Bank Registers and Access Space ........................................ 37
Bank select prefix
  Bank select prefix (PCB,DTB,ADB,and SPB) .................. 61
BAP
  Buffer address pointer (BAP) ........................................... 96
Basic Configuration
  Basic Configuration of Serial Programming Connection for F2MC-16LX MB90F523B ......................................................... 674
Baud Rate
  Baud rate by dedicated baud rate generator ................................ 535
  Baud rate by external clock ........................................... 540
  Baud Rate by Internal Timer (16-bit Reload Timer 0) ...................... 538
  Select of UART Baud Rate .......................................... 533
  Baud rate by dedicated baud rate generator ................................ 535
Bias
  1/3 bias,1/2 duty output waveform example .................................... 606, 609
  1/3 bias,1/4 duty output waveform example .................................... 612
Bidirectional Communication
  Bidirectional Communication Function ..................................... 550
Block Diagram
  8-/16-bit up/down counter/timer 0 block diagram ........................................... 381
  8-/16-bit up/down counter/timer 1 block diagram ........................................... 383
  Block Diagram ......................................................... 510
  Block Diagram for 8-/16-bit PPG Timer 0 .................................. 352
  Block Diagram for Pins of 16-bit Reload Timer ...................................... 324
  Block Diagram for Pins of Wake-up Interrupt ...................................... 455
  Block diagram for the I/O extended serial interface .............................. 561
  Block diagram for the LCD controller/driver ..................................... 589
  Block Diagram of 16-bit Free-run Timer ...................................... 286
  Block Diagram of 16-bit Input/Output Timer ...................................... 284
  Block Diagram of 16-bit Reload Timer ...................................... 322
  Block Diagram of 8-/10-bit A/D Converter ...................................... 465
  Block Diagram of 8-/16-bit PPG Timer 1 ...................................... 354
  Block Diagram of 8-/16-bit PPG Timer Pins ...................................... 356
  Block Diagram of Address Match Detection Function .................................. 619
  Block diagram of all MB90520A series ........................................ 8
  Block Diagram of Clock Generation Section ...................................... 124
  Block Diagram of clock monitor function ........................................ 637
  Block Diagram of Delayed Interrupt Generation Module .......................... 425
  Block Diagram of DTP/External Interrupt ....................................... 433
  Block Diagram of External Reset Pin ........................................ 112
  Block diagram of I/O extended serial interface pins ........................................... 563
  Block Diagram of Input Capture ........................................... 287
  Block diagram of LCD controller/driver pins .................................... 591
  Block diagram of Output compare ........................................... 288
  Block Diagram of Pins ................................................. 290, 385, 435, 503, 638
  Block Diagram of Pins of Port 0 ........................................ 179
  Block Diagram of Pins of Port 1 ........................................ 185
  Block Diagram of Pins of Port 2 ........................................ 191
  Block Diagram of Pins of Port 3 ........................................ 196
INDEX

Command Sequence Table
Command Sequence Table .................................. 647

Common pin switching register
Common pin switching register (LCDCMR) .................. 592

Common Register Bank Prefix
Common Register Bank Prefix (CMR) ..................... 63

Communication
Bidirectional Communication Function ........ 550
Communication prescaler register .................. 511
Master/Slave Mode Communication Function ........ 552

Communication prescaler register
Communication prescaler register .................. 511

Compare
Reload/Compare function ............................ 410

Condition Code Register
Configuration of Condition Code Register (CCR) ........ 51

Connection Example
Connection Example in Single-chip Mode
(Power Supplied from Flash microcomputer Programmer) .... 679
Connection Example in Single-chip Mode
(User Power Supply Used) ................. 677

Continuous mode
Example of El2OS start program
in continuous mode .................................. 494

Continuous Conversion Mode
Continuous Conversion Mode
(ADCS: MD1,MD0="10B") .................. 481
Operation of Continuous Conversion Mode .......... 484
Setting of Continuous Conversion mode ........ 484

Conversion
Conversion Using El2OS ................................ 488

Conversion Mode
Continuous Conversion Mode
(ADCS: MD1,MD0="10B") .................. 481
Conversion Modes of 8-/10-bit A/D Converter ........ 464
Operation of Continuous Conversion Mode .......... 484
Operation of Pause-conversion Mode .......... 486
Operation of Single-shot Conversion Mode .......... 482
Pause-conversion Mode
(ADCS: MD1,MD0="11B") .................. 481
Setting of Continuous Conversion mode .......... 484
Setting of Pause-conversion Mode .......... 486
Setting of Single-shot Conversion Mode .......... 482

Single-shot Conversion Mode (ADCS: MD1,MD0="00B" or "01B") ..... 481

Corresponding products
MB90520A series package and corresponding products .......... 5

Count clock
Count clock of Watchdog timer .................. 259

Count mode
Count mode .................................. 404

Counter control register
Counter control register 0 (CCR0) ........ 388
Counter status register
Counter status register 0 (CSR0) .......... 386
Counter status register 1 (CSR1) .......... 392

CPU
CPU and peripheral functions of MB90520A series .......... 6
CPU Intermittent operation mode .......... 137, 144

CPU Intermittent operation mode
CPU Intermittent operation mode ........ 137

CPU Mode
CPU Mode Data .................................. 170
CPU Mode setting .................................. 168

CPU operation modes
CPU operation modes .......... 168

CSR
Counter status register 0 (CSR0) .......... 386
Counter status register 1 (CSR1) .......... 392

D

D/A control register
D/A control register (DACR) ........ 504

D/A Converter
Block Diagram of the D/A Converter ........ 501
Explanation of Operations of D/A Converter
Functions .................................. 506
Function of D/A converter .......... 500
Notes on Using D/A Converter .......... 507
Pin of D/A Converter .......... 503
Program Example for D/A Converter .......... 508
Reference voltage pin of D/A converter .......... 503

D/A Data Register
D/A Data Register (DADR) .......... 505

DACR
D/A control register (DACR) .......... 504

DADR
D/A Data Register (DADR) .......... 505

Data Access
Setting of Each Bank and Data Access ........ 57

Data bank register
Data bank register (DTB) .......... 57

Data counter
Data counter (DCT) .......... 94

Data polling flag
Data polling flag (DQ7) .......... 651
Data Programming Procedure

Data Programming Procedure........................................660

DCT

Data counter (DCT) .................................................................94

Dedicated baud rate generator

Baud rate by dedicated baud rate generator

..................................................................................535

Dedicated Registers

Configuration of Dedicated Registers.................................42

Dedicated registers and general-purpose registers..................41

Default Space

Bank Addressing and Default Space........................................38

Delayed Interrupt Generation Module

Block Diagram of Delayed Interrupt Generation Module ...........425

Explanation of Operation of Delayed Interrupt Generation Module ..........................................................427

Overview of Delayed Interrupt Generation Module ..................424

Precautions when Using Delayed Interrupt Generation Module ..........................................................424

Program Example of Delayed Interrupt Generation Module ...........428

Delayed interrupt request generate/cancel register

Delayed interrupt request generate/cancel register (DIRR) ..............426

Descriptor

Configuration of E²PROM Descriptor (ISD) ..........................92

Detection Level Setting Register

Detection Level Setting Register (ELVR) (High) ..........................................................438

Detection Level Setting Register (ELVR) (Low) .............................439

Direct Addressing

Direct Addressing .................................................................690

Direct page register

Direct page register (DPR) .....................................................56

DIRR

Delayed interrupt request generate/cancel register (DIRR) ..............426

DIV A,Ri

Precautions when using the "DIV A,Ri" and "DIVW A,RWi" commands ..........................................................67

Division resistors

When external division resistors are used..............................605

When internal division resistors are used..............................604

DIVW A,RWi

Precautions when using the "DIV A,Ri" and "DIVW A,RWi" commands ..........................................................67

DPR

Direct page register (DPR) .....................................................56

DQ2

Toggle bit 2 flag (DQ2) ...............................................................656

DQ3

Sector Deletion Timer Flag (DQ3) ........................................655

DQ5

Timing limit over flag (DQ5) ..................................................654

DQ6

Toggle bit flag (DQ6) ...............................................................653

DQ7

Data polling flag (DQ7) .........................................................651

DTP

DTP Function ........................................................................444

Program sample of DTP Function ........................................448

DTP/External Interrupt

Block Diagram of DTP/External Interrupt......................................433

DTP/External Interrupt Function ...........................................432

DTP/External Interrupt Operation ...........................................441

Pins of DTP/External Interrupt ...............................................435

Program Example of DTP/External Interrupt Function ...............447

Setting of DTP/External Interrupt ...........................................440

DTP/external interrupt enable register

DTP/external interrupt enable register (ENIR) ..........................................................437

DTP/external interrupt factor register

DTP/external interrupt factor register (EIRR) ............................436

Duty

1/3 bias, 1/2 duty output waveform example

.............................................................606, 609

1/3 bias, 1/4 duty output waveform example

.............................................................612

E

E²PROM

System Configuration and E²PROM Memory Map configuration ..........................................................625

Effective Address

Effective Address Field ..........................................................689, 706

E²OS

8-/10-bit A/D Converter Interrupt and E²OS Function .................480

8-/16-bit up/down counter/timer E²OS ........................................401

8-/16-bit up/down counter/timer interrupt and E²OS Function ..........401

Conversion Using E²OS...............................................................488

Correspondence between Timebase Timer Interrupt and E²OS ..........................................................249

E²OS Function of 16-bit Reload Timer ......................................331

E²OS Function of 8-/10-bit A/D Converter ..................................480

E²OS function of the I/O extended serial interface ..........................................................569
INDEX

EI²OS Function of UART .............................. 529
EI²OS Functions of 16-bit Input/Output Timer ......................................................... 305
EI²OS Processing Time (time for one transfer) ........................................................... 99
Example of EI²OS start program in continuous mode............................................. 494
Example of EI²OS start program in single mode ..................................................... 492
Example of EI²OS start program in stop mode ....................................................... 496
Extended intelligent I/O service (EI²OS) .............................................................. 90
Extended Intelligent I/O Service (EI²OS) program ..................................................... 106
Interrupt Related to UART and EI²OS ................................................................. 529
Interruption of the I/O extended serial interface and EI²OS .................................... 569
Interrupts of 16-bit Input/Output Timer and EI²OS ................................................... 305
Interrupts of 16-bit Reload Timer and EI²OS .......................................................... 331
Operation of EI²OS .................................................................................................. 91, 97
Procedure for Use of EI²OS ..................................................................................... 98
Watch Timer Interrupt and EI²OS Function ............................................................. 277
EI²OS Descriptor
    Configuration of EI²OS Descriptor (ISD) .............................................................. 92
EI²OS Status Register
    EI²OS Status Register (ISCS) .................................................................................. 95
EICR
    Wake-up interrupt control register (EICR) .......................................................... 457
EIFR
    Wake-up interrupt flag register (EIFR) .................................................................. 456
EIRR
    DTP/external interrupt factor register (EIRR) ....................................................... 436
ELVR
    Detection Level Setting Register (ELVR) (High) .................................................. 438
    Detection Level Setting Register (ELVR) (Low) .................................................. 439
ENIR
    DTP/external interrupt enable register (ENIR) ..................................................... 437
Erase
    Data Erase from Flash Memory (Chip Erase) ...................................................... 662
    Sector Erase Resumption of Flash Memory .......................................................... 666
    Sector Erase Suspension ....................................................................................... 665
Erasing
    Erasing Any Data in Flash Memory (Sector Erasing) .......................................... 663
    Erasing Procedure for Flash Memory Sectors ....................................................... 663
Event Count Mode
    Event count mode .................................................................................................. 320
    Operation in Event Count Mode ............................................................................. 340
    Setting of Event Count Mode ................................................................................ 339
Event Counter Mode
    Program Example in Event Counter Mode ........................................................... 344
Exception processing
    Exception processing ............................................................................................. 101
Execution Cycle
    Calculating the Execution Cycle Count ............................................................... 704
    Execution Cycle Count ......................................................................................... 704
Extended Intelligent I/O Service
    Extended Intelligent I/O Service (EI²OS) program .................................................. 106
Extended intelligent I/O service
    Extended intelligent I/O service (EI²OS) .............................................................. 90
External clock
    Baud rate by external clock .................................................................................... 540
    Connection of Oscillator and External clock .......................................................... 135
External division resistors
    When external division resistors are used ............................................................... 605
External Interrupt
    External Interrupt Function .................................................................................... 443
External Reset
    Block Diagram of External Reset Pin ..................................................................... 112
External shift clock mode
    External shift clock mode ...................................................................................... 574
F²MC-16LX
    Basic Configuration of Serial Programming
        Connection for F²MC-16LX MB90F523B .......................................................... 674
    F²MC-16LX Instruction List .................................................................................... 711
FF Bank
    Access to FF Bank by ROM Mirroring Function ..................................................... 632
Flag
    Data polling flag (DQ7) ......................................................................................... 651
    Hardware sequence flag ....................................................................................... 649
    Relationships between the reset factor flag bit and the reset factor ...................... 116
    Reset factor flag bit .............................................................................................. 115
    Sector Deletion Timer Flag (DQ3) ........................................................................ 655
    Setting/Clearing a reset factor flag bit ................................................................... 116
    Timing limit over flag (DQ5) ................................................................................ 654
    Timing of reception interrupt request generation and flag setting ....................... 530
    Timing of send interrupt request generation and flag setting .............................. 532
    Toggle bit flag (DQ6) ............................................................................................ 653
INDEX

Flag change inhibit prefix
  Flag change inhibit prefix (NCC) .................. 64

Flash Memory
  1 Mbit Flash Memory Program Example ........... 668
  Assignment of flash memory writing pin ........ 177
  Data Erase from Flash Memory (Chip Erase) ... 662
  Details of Programming/Erasing Flash Memory ... 658

Erasing Any Data in Flash Memory
  (Sector Erasing) .................................. 663

Features of 1Mbit Flash Memory .................. 642
Overview of 1M-bit Flash Memory .................. 642
Programming and Erasing Flash Memory .......... 642
Read/Reset State in Flash Memory ............... 659
Sector Configuration of 1Mbit Flash memory ... 643
Sector Erase Resumption of Flash Memory ....... 666
Writing data to flash memory ..................... 660
Notes on using flash memory ..................... 667

Flash memory control status register
  Flash memory control status register (FMCS) ... 644

Flash microcomputer programmer
  System configuration of the flash microcomputer
  programmer (made by Yokogawa Digital
  Computer Corporation) ......................... 676

Flash Microcontroller Programmer
  Example of Minimum Connection to Flash
  Microcontroller Programmer
  (User Power Supply Used) ...................... 681

FMCS
  Flash memory control status register (FMCS) ... 644

FPT-120P-M05
  Package dimensions of FPT-120P-M05 .......... 11

FPT-120P-M13
  Package dimensions of FPT-120P-M13 .......... 12

Free-run Timer
  Block Diagram of 16-bit Free-run Timer ....... 286
  Explanation of Operation of 16-bit Free-run Timer
  .................................................. 306

G

General-purpose Register
  Configuration of a general-purpose register
  ................................................... 58
  Dedicated registers
    and general-purpose registers .............. 41
  General-purpose Register Area and Register
    Bank Pointer .................................. 53

H

Handling Devices
  Precautions when Handling Devices ............ 24

Hardware Interrupt
  Hardware Interrupt ............................. 81, 82, 86
  Hardware Interrupt Inhibition ................. 82
  Operation of Hardware Interrupt .............. 85
  Return from Hardware Interrupt ............... 84
  Start of Hardware Interrupt ................... 84

Hardware sequence flag
  Hardware sequence flag ........................ 649

Hardware Standby Mode
  Cancellation of Hardware Standby Mode ....... 158
  Transition to Hardware Standby Mode .......... 158

High-order counter control register
  High-order counter control register 1 (CCR1: H)
  .................................................. 394

High-order output compare control status register
  High-order output compare control status register
  (OCS) .......................................... 298

I

I/O address pointer
  I/O address pointer (IOA) ....................... 94

I/O Area
  I/O Area ........................................... 31

I/O Circuit
  I/O Circuit ....................................... 18

I/O extended serial interface
  Block diagram for the I/O extended serial
  interface ........................................ 561
  Block diagram of I/O extended serial interface
  pins ............................................. 563
  EI^2OS function of the I/O extended serial
  interface ........................................ 569
  Generation of interrupt request with the I/O
  extended serial interface ..................... 563
  I/O extended serial interface outline ....... 560
  I/O extended serial interface pins .......... 563
  I/O extended serial interface set up .......... 576
  Instructions for use of the I/O extended serial
  interface ........................................ 583
  Interruption of the I/O extended serial interface
  .................................................. 583
  Interruption of the I/O extended serial interface
  and EI^2OS ...................................... 583
  Operating status of the I/O extended serial
  interface ........................................ 578
  Program example 2 for I/O extended serial
  interface ....................................... 585
  Program example for I/O extended serial
  interface ....................................... 584
Start and stop transmission/reception operation of the I/O extended serial interface ........................................ 575
Transmission/reception operation of the I/O extended serial interface ........................................ 575
I/O extension serial interface
Timing of I/O extension serial interface operations .......................................................... 580
I/O Pin
I/O Pin state ........................................................................................................ 164
I/O Port
I/O Port Function .................................................................................................. 176
Registers of I/O Ports ................................................................................................. 176
I/O timer
Module configuration of 16-bit I/O timer .......................................................... 282
Precaution on using 16-bit I/O timer ........................................................................ 314
ICR
Bit Configuration of Interrupt Control Register (ICR) .................................................. 78
Interrupt Control Register (ICR00 to ICR15) .......................................................... 76
ICS
Input capture control status registers (ICS00) .................................................. 295
ILM
Interrupt level mask register (ILM) ........................................................................ 54
Indirect Addressing
Indirect Addressing ..................................................................................................... 697
Indirect specification
Addressing by indirect specification with a 32-bit register ........................................ 36
Input Capture
Block Diagram of Input Capture .................................................................................. 287
Explanation of Operation of Input Capture .................................................................. 309
Input capture control status registers
Input capture control status registers (ICS00) .......................................................... 295
Input capture register
Input capture register (IPCP) .................................................................................. 297
Input/Output Timer
Block Diagram of 16-bit Input/Output Timer ................................................................ 284
EI²OS Functions of 16-bit Input/Output Timer .................................................................. 305
Factors of 16-bit Input/Output Timer .................................................................................. 305
Generation of Interrupt Request from 16-bit Input/Output Timer ................................ 283
Interrupts of 16-bit Input/Output Timer .......................................................................... 304
Interrupts of 16-bit Input/Output Timer and EI²OS .................................................. 305
Pins of 16-bit Input/Output Timer .................................................................................. 290
Program Example of 16-bit Input/Output Timer .......................................................... 315
Instruction
Configuration of Instruction Map .......................................................... 725
Description of instruction presentation items and symbols ........................................... 708
F²MC-16LX Instruction List ...................................................................................... 711
Instruction Types ........................................................................................................ 687
Instructions for use of the I/O extended serial interface .................................................. 583
Prefix Code and Interrupt Inhibit Instruction ............................................................ 65
Internal Clock Mode
Internal clock mode ............................................................................................... 320
Operation in Internal Clock Mode ........................................................................... 335
Program Example in Internal Clock Mode .............................................................. 343
Setting of internal clock mode ................................................................................. 334
Internal division resistors
When internal division resistors are used .......................................................... 604
Internal shift clock mode
Internal shift clock mode .......................................................................................... 572
Internal Timer
Baud Rate by Internal Timer (16-bit Reload Timer 0) ............................................. 538
Interrupt
8-/10-bit A/D Converter Interrupt and EI²OS .............................................................. 480
8-/16-bit up/down counter/timer interrupt ........................................................... 400
8-/16-bit up/down counter/timer interrupt and EI²OS .............................................. 401
Block Diagram of DTP/External Interrupt ............................................................... 433
Cancellation of Standby Mode by Interrupt .......................................................... 166
Correspondence between Timebase Timer Interrupt and EI²OS ......................... 249
Details of Interrupts of 8-/16-bit PPG Timer ............................................................ 364
Details of Pins and Interrupt Numbers ........................................................................ 285, 434, 454
DTP/External Interrupt Function ............................................................................. 432
DTP/External Interrupt Operation ............................................................................. 441
External Interrupt Function ....................................................................................... 443
Generation of Interrupt from 8-/10-bit A/D Converter ........................................... 468
Generation of Interrupt from 8-/16-bit PPG Timer ................................................. 356
Generation of Interrupt Request from 16-bit Input/Output Timer ......................... 291
Generation of Interrupt Request from 16-bit Reload Timer .................................. 324
Generation of Interrupt Request from Timebase Timer ........................................ 246
Generation of Interrupt Request from Watch Timer ................................................ 274
Generation of interrupt request with the I/O extended serial interface ................. 563
Hardware Interrupt .................................................................................................. 81, 82
Hardware Interrupt Inhibition .................................................................................... 82
Interrupt Number ..................................................................................................... 425
Interrupt of A/D Converter ........................................480
Interrupt of Timebase Timer .......................................249
Interrupt Operation ..................................................70
Interrupt Related to UART and EI²OS .........................529
Interrupt Request Generation by UART .......................519
Interrupt requests are generated by the 8-/16-bit up/down counter/timer ........................................385
Interrupt factors,interruption vectors, and interrupt control register ........................................72
Interrupts of 16-bit Input/Output Timer .........................304
Interrupts of 16-bit Input/Output Timer and EI²OS ........305
Interrupts of 16-bit Reload Timer ..................................331
Interrupts of 8-/16-bit PPG Timer ................................364
Multiple interrupts ....................................................87
Pins of DTP/External Interrupt ....................................435
Program Example of DTP/External Interrupt Function ..........447
Program Example of Interrupt Processing .......................105
Setting of DTP/External Interrupt .................................440
Stack Operation at Return from Interrupt Processing ........104
Stack Operation at Starting Interrupt Processing ..............104
Start and operation of software interrupt .......................89
Time Required to Start Interrupt Processing .....................102
Timing of reception interrupt request generation and flag setting ........................................530
Timing of send interrupt request generation and flag setting ........................................532
Type and Function of Interrupt .....................................69
UART interrupt .........................................................528
Watch Timer Interrupt ..............................................277
Watch Timer Interrupt and EI²OS Function .....................277
Interrupt Control Register
Bit Configuration of Interrupt Control Register (ICR) ........78
Function of Interrupt Control Register ..........................79
Interrupt Control Register (ICR00 to ICR15) ....................76
Interrupt Control Register List .....................................74
Interrupt factors,interruption vectors, and interrupt control register ........................................72
Interrupt Inhibit Instruction
Prefix Code and Interrupt Inhibit Instruction ....................65
Interrupt Inhibition
Hardware Interrupt Inhibition .....................................82
Interrupt level mask register
Interrupt level mask register (ILM) ..............................54
Interrupt vector
Interrupt vector index ..............................................764
Interrupt
Interrupt factors,interruption vectors, and interrupt control register ........................................72
Interrupt of the I/O extended serial interface and EI²OS ...569
Interrupt factors
Interrupt factors,interruption vectors, and interrupt control register ........................................72
Interrupt vector
Interrupt factors,interruption vectors, and interrupt control register ........................................72
Interrupt vector
Interrupt factors,interruption vectors, and interrupt control register ........................................71
Interval Timer
Interval Timer Function ........................................242, 250, 270, 278
IOA
I/O address pointer (IOA) ...........................................94
IPCP
Input capture register (IPCP) ..........................................297
ISCS
EI²OS Status Register (ISCS) ........................................95
ISD
Configuration of EI²OS Descriptor (ISD) .........................92
L
LCD
Data memory for the LCD ...........................................598
Drive waveform for the LCD ........................................602
Power supply for LCD driving ....................................603
LCD controller/driver
Block diagram for the LCD controller/driver ....................589
Block diagram of LCD controller/driver pins ....................591
Explanations for the LCD controller/driver operation ..........601
LCD controller/driver outline .......................................588
LCD controller/driver pins .........................................591
Program example for the LCD controller/driver .................615
LCDC control register
LCDC control register 0 (LCR0) ..................................594
LCDC control register 1 (LCR1) ..................................596
LCDCMR
Common pin switching register (LCDCMR) ......................592
LCR
LCDC control register 0 (LCR0) ..................................594
LCDC control register 1 (LCR1) ..................................596
Linear Addressing
Linear Addressing and Bank Addressing .......................35
Linear addressing specified by 24-bit operand .................36
INDEX

Low-order counter control register
Low-order counter control register 0
(CCR0:L) ........................................... 390
Low-order counter control register 1
(CCR1:L) ........................................... 396
Low-order output compare control status register
Low-order output compare control status register
(OCS) ............................................. 301
Low-power consumption control circuit
Block Diagram of the Low-power consumption
circuit ........................................... 139
Low-power Consumption Mode
Low-power Consumption Mode .................... 136
Low-power consumption mode control register
Low-power consumption mode control register
(LPMCR) ........................................... 141
LPMCR
Low-power consumption mode control register
(LPMCR) ........................................... 141

M

Machine clock
Machine clock .................................... 130
Machine clock selection ......................... 131
Main Clock
Main Clock Oscillation Stabilization Wait Time
.................................................... 134
Main Clock Mode, PLL Clock Mode, and sub-clock
Mode ............................................. 129
Master/Slave Mode Communication
Master/Slave Mode Communication Function
.................................................. 552
MB90520A
Block diagram of all MB90520A series ......... 8
CPU and peripheral functions of MB90520A
series ............................................. 6
Feature of MB90520A series ..................... 2
MB90520A series package
and corresponding products ..................... 5
Memory allocation of MB90520A ............... 32
Product lineup of the MB90520A series ....... 5

MB90F523B
Basic Configuration of Serial Programming
Connection for F2MC-16LX MB90F523B
................................................... 674
MD
Continuous Conversion Mode
(ADCS: MD1, MD0 = "10B") .................. 481
Mode Pin (MD2 to MD0) .......................... 169
Pause-conversion Mode
(ADCS: MD1, MD0 = "11B") .................. 481
Single-shot Conversion Mode (ADCS:
MD1, MD0 = "00B" or "01B") ............... 481

Memory
Data memory for the LCD .......................... 598
Memory Access Mode
Memory Access Mode ............................ 168
Memory allocation
Memory allocation of MB90520A .......... 32
Memory Map
Memory Map .................................... 34
System Configuration and E²PROM Memory
Map configuration ......................... 625
Memory Space
Memory Space .................................. 30
Memory Space when ROM Mirroring Function
Enabled/Disabled ......................... 633
Minimum Connection
Example of Minimum Connection
(Power Supplied from Flash
Microcomputer Programmer) ............. 683
Example of Minimum Connection to Flash
Microcontroller Programmer
(User Power Supply Used) ............ 681

Mode
Cancellation of Hardware Standby Mode
.................................................. 158
Cancellation of sleep modes ................... 148
Cancellation of Standby Mode by Interrupt
.................................................. 166
Cancellation of timebase timer mode ....... 151
Cancellation of watch mode ................... 153
Connection Example in Single-chip Mode
(Power Supplied from Flash
microcomputer Programmer) ............. 679
Connection Example in Single-chip Mode
(User Power Supply Used) ............ 677
Continuous Conversion Mode
(ADCS: MD1, MD0 = "10B") ............... 481
Conversion Modes of 8-/10-bit A/D Converter
.................................................. 464
Count mode .................................. 404
CPU Intermittent operation mode .......... 137, 144
CPU Mode Data ................................ 170
CPU Mode setting ............................. 168
CPU operation modes ......................... 168
Event count mode ............................. 320
Example of E²POS start program
in continuous mode ....................... 494
Example of E²POS start program in single mode
................................................... 492
Example of E²POS start program in stop mode
.................................................. 496
External shift clock mode ..................... 574
Internal clock mode .......................... 320
Internal shift clock mode .................... 572
Low-power Consumption Mode ............ 136
Main Clock Mode, PLL Clock Mode,
and sub-clock Mode ..................... 129
Operation of Port 4 ........................................204
Pin Assignment of Port 4..................................200
Registers for Port 4.......................................201

Port 5
Block Diagram of Pins of Port 5............................207
Configuration of Port 5....................................206
Function of Registers for Port 5..........................209
Operation of Port 5.........................................210
Pins Assignment of Port 5..................................206
Registers for Port 5.......................................208

Port 6
Block Diagram of Pins of Port 6............................213
Configuration of Port 6....................................212
Function of Registers for Port 6..........................214
Operation of Port 6.........................................216
Pins Assignment of Port 6..................................212
Registers for Port 6.......................................213

Port 7
Block Diagram of Pins of Port 7............................220
Configuration of Port 7....................................218
Function of Registers for Port 7..........................222
Operation of Port 7.........................................223
Pins Assignment of Port 7..................................218
Registers for Port 7.......................................221

Port 8
Block Diagram of Pins of Port 8............................226
Configuration of Port 8....................................225
Function of Registers for Port 8..........................227
Operation of Port 8.........................................228
Pins Assignment of Port 8..................................225
Registers for Port 8.......................................226

Port 9
Block Diagram of Pins of Port 9............................231
Configuration of Port 9....................................230
Function of Registers for Port 9..........................232
Operation of Port 9.........................................233
Pins Assignment of Port 9..................................230
Registers for Port 9.......................................231

Port A
Block Diagram of Pins of Port A............................236
Configuration of Port A....................................235
Function of Registers for Port A..........................237
Operation of Port A.........................................238
Pins Assignment of Port A..................................235
Registers for Port A.......................................236

Power Supply
Connection Example in Single-chip Mode
(Power Supplied from Flash microcomputer Programmer)......679
Connection Example in Single-chip Mode
(User Power Supply Used).................................677
Example of Minimum Connection
(Power Supplied from Flash Microcomputer Programmer)......683
Power supply for LCD driving...............................603

PPG Reload Registers
PPG Reload Registers
(PRLL0/PRLH0,PRLL1/PRLH1)..........................363

PPG Timer
Block Diagram for 8-/16-bit PPG Timer 0..........................352
Block Diagram of 8-/16-bit PPG Timer 1..........................354
Block Diagram of 8-/16-bit PPG Timer Pins
...............................................................356
Channels and PPG Pins of PPG Timers..........................351
Detail of Interrupts of 8-/16-bit PPG Timer
...............................................................364
Functions of 8-/16-bit PPG Timer..............................348
Generation of Interrupt from 8-/16-bit PPG Timer
...............................................................356
Interrupts of 8-/16-bit PPG Timer..............................364
Operation Modes of 8-/16-bit PPG Timer..........................348
Operation of 8-/16-bit PPG Timer..............................365
Pins of 8-/16-bit PPG Timer..................................356
Precautions when Using 8-/16-bit PPG Timer
...............................................................375
Program Example for 8-/16-bit PPG Timer
...............................................................377

PPG0 Operation Mode Control Register
PPG0 Operation Mode Control Register
(PPGC0).......................................................357
PPG0 output control register
PPG0 output control register (PPGOE)..........................361
PPG1 Operation Mode Control Register
PPG1 Operation Mode Control Register
(PPGC1).......................................................359

PPGC
PPG0 Operation Mode Control Register
(PPGC0).......................................................357
PPG1 Operation Mode Control Register
(PPGC1).......................................................359

PPGOE
PPG01 output control register (PPGOE)..........................361

Prefix
Bank select prefix
(PCI,DTB,ADB,and SPB)..................................61
Flag change inhibit prefix (NCC)............................64

Prefix Code
Array of Prefix Codes.......................................66
Prefix Code.....................................................60
Prefix Code and Interrupt Inhibit Instruction
...............................................................65

Prescaler
Communication prescaler register.............................511

PRLH
PPG Reload Registers
(PRLL0/PRLH0,PRLL1/PRLH1)..........................363

PRLL
PPG Reload Registers
(PRLL0/PRLH0,PRLL1/PRLH1)..........................363
INDEX

Processor Status
  Configuration of Processor Status (PS) ........ 50
Product lineup
  Product lineup of the MB90520A series .......... 5
Program address detect register
  Program address detect register (PADR) .............................................. 622
Program address detection control status register
  Program address detection control status register (PACSR) ......................... 620
Program counter
  Program counter (PC) .............................................. 55
Program counter bank register
  Program counter bank register (PCB) .............. 57

Program Example
  1 Mbit Flash Memory Program Example ...... 668
  Program example 2 for I/O extended serial interface ...................................... 585
  Program Example for 8-/16-bit PPG Timer ........................................................................... 377
  Program Example for D/A Converter .......... 508
  Program example for I/O extended serial interface ...................................................... 584
  Program example for the LCD controller/driver ......................................................... 615
  Program Example in Event Counter Mode ........................................................................... 344
  Program Example in Internal Clock Mode ........................................................................... 343
  Program Example of 16-bit Input/Output Timer ............................................................ 315
  Program Example of Address Match Detection Function ................................................... 630
  Program Example of Delayed Interrupt Generation Module ............................................ 429
  Program Example of DTP/External Interrupt Function .................................................... 447
  Program Example of Interrupt Processing ........................................................................... 105
  Program example of the 8-/16-bit timer mode ....................................................................... 417
  Program example of the 8-bit phase difference count mode ........................................... 419
  Program example of the program mode of the 8-bit up/down count mode ......................... 418
  Program mode
    Program example of the program mode of the 8-bit up/down count mode ......................... 418
  Program patching
    Processing flow of the program patching ........................................................................... 629
  Program sample
    Program sample of DTP Function .................................................................................. 448

Programming Procedure
  Data Programming Procedure .................................. 660

Protection
  A/D conversion Data Protection Function in 8-/10-bit A/D Converter ......................... 489

PS
  Configuration of Processor Status (PS) ........ 50

Pull-up resistance
  Pull-up resistance .............................................. 164

R

RAM
  Sequence of multi-byte data setting in RAM ................................................................. 39

RAM Area
  RAM Area ................................................................. 31

RCR
  Reload compare register (RCR0,RCR1) .......... 399

Reception
  Start and stop transmission/reception operation of the I/O extended serial interface .................. 575
  Transmission/reception operation of the I/O extended serial interface ................................. 575

Reception interrupt
  Timing of reception interrupt request generation and flag setting ...................................... 530

Reference voltage
  Reference voltage pin of D/A converter ................................................................. 503

Register Area
  Register Area ................................................................. 31

Register Bank
  Register Bank ................................................................. 59

Register Bank Pointer
  General-purpose Register Area
    and Register Bank Pointer ................................................................. 53
  Register bank pointer (RP) ................................................................. 53

Register Index
  Register Index (address list) ................................................................. 747
  Register Index (List by Peripheral Function) ................................................................. 754

Release
  Release of the stop mode ................................................................. 156

Reload
  Reload/Compare function ................................................................. 410
  Reload/counter clear timing ................................................................. 413

Reload compare register
  Reload compare register (RCR0,RCR1) ................................................................. 399

Reload Timer
  Baud Rate by Internal Timer
    (16-bit Reload Timer 0) ................................................................. 538
  Block Diagram for Pins of 16-bit Reload Timer ................................................................. 324
INDEX

Block Diagram of 16-bit Reload Timer...........322
EI2OS Function of 16-bit Reload Timer...........331
Generation of Interrupt Request
  from 16-bit Reload Timer ..................324
Interrupts of 16-bit Reload Timer ..............331
Operation Modes of 16-bit Reload Timer ...........320
Pins of 16-bit Reload Timer ....................324
Precautions when Using 16-bit Reload Timer .......342
Setting of 16-bit Reload Timer .................332

Reset
Block Diagram of External Reset Pin ..........112
Operation Status Changes during a Reset .....113
Overview of Reset Operation .................113
Registers that are initialized by reset factors ........................................118
Relationships between the reset factor flag bit
  Bits and the reset factor ..................116
Reset Factor ........................................108
Reset factor flag bit .........................115
Setting/Clearing a reset factor flag bit .....116
State of Pins at Reset .........................117
The register that can not be initialized
  by reset input ....................................119

Reset Factor
Registers that are initialized by reset factors ........................................118
Relationships between the reset factor flag bit
  Bits and the reset factor ..................116
Reset Factor ........................................108
Reset factor flag bit .........................115
Reset Factors and Oscillation Stabilization Wait Times ................................110

Reset factor flag bit
Relationships between the reset factor flag bit
  Bits and the reset factor ..................116
Reset factor flag bit .........................115
Setting/Clearing a reset factor flag bit .....116

ROM Area
ROM Area................................................31

ROM Mirroring
Access to FF Bank by ROM Mirroring Function ..................................632
Memory Space when ROM Mirroring Function
  Enabled/Disabled ..............................633
ROM mirroring function .......................33
ROM Mirroring Function Select Module
Block Diagram of ROM Mirroring Function Select
  Module...........................................632

ROM Mirroring Function Select Register
ROM Mirroring Function Select Register (ROMM)
..................................................634

ROMM
ROM Mirroring Function Select Register (ROMM)
..................................................634

RP
Register bank pointer (RP) ......................53

S
SCR
Serial control register (SCR) ....................520

SDR
Serial data register (SDR) ......................568

Sector
Erasing Procedure for Flash Memory Sectors ..................................663
Sector Configuration of 1Mbit Flash memory ..................................643
Sector Erase Resumption of Flash Memory ..................................666
Sector Erase Suspension ..................................665

Sector Deletion Timer Flag
Sector Deletion Timer Flag (DQ3) .................655

Sector Erasing
Erasing Any Data in Flash Memory
  (Sector Erasing) ...............................663

Send interrupt
Timing of send interrupt request generation
  and flag setting ................................532

Serial clock
Oscillation clock frequency
  and serial clock input frequency ........676

Serial control register
Serial control register (SCR) ....................520

Serial data I/O
Shift timing of serial data I/O ................582

Serial data register
Serial data register (SDR) ......................568

Serial input data register
Serial input data register (SIDR) ...............526

Serial mode control status register
Serial mode control status register lower
  (SMCS:L) .......................................566
Serial mode control status register upper
  (SMCS:H) .......................................564

Serial mode register
Serial mode register (SMR) .....................522

Serial output data register
Serial output data register (SODR) ...............527

783
INDEX

Serial Programming Connection
Basic Configuration of Serial Programming
Connection for F2MC-16LX MB90F523B .............................................. 674

Serial status register
Serial status register (SSR) ......................................................... 524

Serial writer
Supports serial writer ............................................................... 684

Shift clock
External shift clock mode .......................................................... 574
Internal shift clock mode ............................................................ 572
Selection of shift clock ............................................................... 571

Shift clock generating circuit
Shift clock generating circuit ..................................................... 570

SIDR
Serial input data register (SIDR) ................................................. 526

Single mode
Example of El2OS start program in single mode .......................... 492

Single-chip Mode
Connection Example in Single-chip Mode
(Power Supplied from Flash microcomputer Programmer) ............ 679
Connection Example in Single-chip Mode
(User Power Supply Used) ....................................................... 677
Single-chip mode ............................................................... 171

Single-shot Conversion Mode
Operation of Single-shot Conversion Mode .................................. 482
Setting of Single-shot Conversion Mode ......................................... 482
Single-shot Conversion Mode
(ADCS: MD1,MD0="00B" or "01B") ................................................. 481

Sleep Mode
Cancellation of sleep modes .................................................. 148
Transition to Sleep Mode ........................................................... 147

SMCS
Serial mode control status register lower (SMCS:L) ..................... 566
Serial mode control status register upper (SMCS:H) ...................... 564

SMR
Serial mode register (SMR) ...................................................... 522

SODR
Serial output data register (SODR) ........................................... 527

Software Interrupt
Return from Software Interrupt .................................................. 89
Start and operation of software interrupt ..................................... 89

SPB
Bank select prefix (PCB,DTB,ADB,and SPB) .................................. 61

SSB
User Stack Bank Register (USB) and System
Stack Bank Register (SSB) ....................................................... 57

SSP
System stack pointer (SSP) ...................................................... 48

SSR
Serial status register (SSR) ....................................................... 524

Stack
Sequence of multi-byte data setting in a stack ............................... 40
Stack Operation at Return from Interrupt Processing ..................... 104
Stack Operation at Starting Interrupt Processing .......................... 104
Stack Selection ................................................................. 47

Stack Area
Stack Area ............................................................................. 49

Standby Mode
Cancellation of Standby Mode by Interrupt ................................. 166
Notes on cancelling Standby Mode ............................................. 167
Notes on Setting Standby Mode .................................................. 166
Notes on the Transition to Standby Mode ...................................... 166
Operating State of Standby Mode ................................................. 145
Standby Mode ......................................................................... 137
Transition to Standby Mode ......................................................... 166

State Transition Diagram
State Transition Diagram ............................................................. 159

Stop mode
Example of El2OS start program in stop mode ................................ 496
Release of the stop mode ............................................................ 156
Transition to stop mode ............................................................. 155

Sub watch mode
Note on cancelling sub watch mode ............................................ 167

Sub-clock
Main Clock Mode, PLL Clock Mode, and sub-clock Mode ................ 129
Oscillation stabilization wait time of sub-clock .............................. 134
Oscillation Stabilization Wait Time Timer of sub-clock .................. 279

Synchronous Mode
Operation in Clock Synchronous Mode ......................................... 547

System configuration
System configuration of the flash microcomputer programmer (made by Yokogawa Digital Computer Corporation) ............................................. 676

System Stack Bank Register
User Stack Bank Register (USB) and System Stack Bank Register (SSB) ......................................................... 57

System stack pointer
System stack pointer (SSP) ........................................................ 48
INDEX

T

TBTC
Timebase timer control register (TBTC)..................247

TCCS
Timer control status register (TCCS)....................292

TCDT
Timer Data Register (TCDT)...............................294

Timebase Timer
Block Diagram of Timebase Timer.......................244
Correspondence between Timebase Timer
Interrupt and EI2OS........................................249
Generation of Interrupt Request
from Timebase Timer.......................................246
Interrupt of Timebase Timer................................249
Precautions when Using Timebase Timer.................254
Program Example of Timebase Timer.....................256

Timebase timer control register
Timebase timer control register (TBTC)................247

Timebase Timer Mode
Cancellation of timebase timer mode..................151
Notes on cancelling timebase timer mode...............167
Transition to timebase Timer Mode.....................150

Timer Control Status Register
Timer control status register (TCCS)..................292
Timer Control Status Registers (Low)
(TMCSR0: L, TMCSR1: L).................................327

Timer Data Register
Timer Data Register (TCDT)...............................294

Timer mode
Operation of timer mode..................................405

Timer Register
Operating State of 16-bit Timer Register..............333
Timing limit over flag
Timing limit over flag (DQ5).............................654

TMCSR
Timer Control Status Registers (Low)
(TMCSR0: L, TMCSR1: L).................................327

TMR
16-bit Timer Registers (TMR0, TMR1)...................329

TMRLR
16-bit Reload Registers (TMRLR0, TMRLR1)
..........................................................330

Toggle bit flag
Toggle bit flag (DQ6).....................................653
Toggle bit 2 flag (DQ2)..................................656

Transition
Notes on the Transition to Standby Mode.............166
State Transition Diagram.................................159
Transition of clock mode................................167
Transition to Hardware Standby Mode................158
Transition to Sleep Mode................................147

Transition to Standby Mode.............................166
Transition to stop mode..................................155
Transition to timebase Timer Mode....................150
Transition to Watch mode...............................152

Transmission
Start and stop transmission/reception operation of the I/O extended serial interface
......................................................................575
Transmission/reception operation of the I/O extended serial interface........575

U

UART
Block Diagram of UART Pins...............................519
EI2OS Function of UART..................................529
Example of UART programming.........................556
Interrupt Related to UART and EI2OS.................529
Interrupt Request Generation by UART................519
Notes on using UART....................................555
Operation of UART.......................................541
Select of UART Baud Rate...............................533
UART block diagram.....................................516
UART function............................................514
UART interrupt...........................................528
UART pins..................................................519

UART programming
Example of UART programming.........................556

UDCR
Up/down count register (UDCR0, UDCR1)............398

Underflow
Operation as 16-bit Timer Register Underflow
.....................................................................335, 340
Operation at Underflow...................................321

Up/down count mode
Up/down count mode operation.........................406

Up/down count register
Up/down count register (UDCR0, UDCR1)
.........................................................................398
Up/down count register clear/gate function
........................................................................415

Up/down counter/timer
8-/16-bit up/down counter/timer 0 block diagram
.....................................................................381
8-/16-bit up/down counter/timer 1 block diagram
.....................................................................383
8-/16-bit up/down counter/timer EI2OS.................401
8-/16-bit up/down counter/timer functions
........................................................................380
8-/16-bit up/down counter/timer interrupt
.........................................................................400
8-/16-bit up/down counter/timer interrupt
and EI2OS......................................................401
8-/16-bit up/down counter/timer pin....................385

785
## INDEX

Interrupt requests are generated by the 8-/16-bit up/down counter/timer .... 385  
Precautions for 8-/16-bit up/down counter/timer .............................................. 416

USB  
User Stack Bank Register (USB) and System Stack Bank Register (SSB) ........ 57

User Power Supply  
Connection Example in Single-chip Mode (User Power Supply Used) ........ 677

User Stack Bank Register  
User Stack Bank Register (USB) and System Stack Bank Register (SSB) ........ 57

User stack pointer  
User stack pointer (USP) ............................................. 48

USP  
User stack pointer (USP) ............................................. 48

### W

Wake-up Interrupt  
Block Diagram for Pins of Wake-up Interrupt ...................... 455  
Block diagram of wake-up interrupt ...................... 453  
Explanations for wake-up interrupt operation .................. 459  
Pins of Wake-up Interrupt ........................................ 455  
Precautions when using wake-up interrupt .................... 461  
Program examples for wake-up interrupt ...................... 462  
Wake-up interrupt function ........................................ 452

Wake-up interrupt control register  
Wake-up interrupt control register (EICR) .................. 457

Wake-up interrupt flag register  
Wake-up interrupt flag register (EIFR) ..................... 456

Watch mode  
Cancellation of watch mode ..................................... 153  
Transition to Watch mode ........................................ 152

Watch Timer  
Block Diagram of Watch Timer ......................... 272  
Generation of Interrupt Request from Watch Timer .................... 274  
Program Example of Watch Timer ......................... 280  
Watch timer counter ........................................... 278  
Watch Timer Interrupt ........................................ 277  
Watch Timer Interrupt and EI2OS Function ....................... 277

Watch timer control register  
Watch timer control register (WTC) ..................... 275

Watchdog Timer  
Block Diagram of Watchdog Timer ....................... 260  
Count clock of Watchdog timer ................................ 259  
Functions of Watchdog Timer ................................ 258  
Operations of Watchdog Timer ......................... 264

Precautions when Using Watchdog Timer ................................................. 266

Program Examples of Watchdog Timer ...................... 267  
Setting Operation Clock of Watchdog Timer ....................... 279

Watchdog timer control register  
Watchdog timer control register (WDTC) .................. 262

Waveform  
1/3 bias, 1/2 duty output waveform example .................. 606, 609  
1/3 bias, 1/4 duty output waveform example .................. 612

Drive waveform for the LCD .................................. 602

WDTC  
Watchdog timer control register (WDTC) .................. 262

WTC  
Watch timer control register (WTC) ..................... 275