PREFACE

■ Objectives and Intended Readers

The MB89890 Series was developed as one of the general-purpose products of the F²MC-8L family, ASIC (application specific IC) ready original eight-bit one-chip microcontrollers. The MB89890 Series can be used in many applications from portable devices and other consumer products to industrial products.

This manual provides information on the functions and operations of the series for engineers developing products for use with the MB89890 Series microcontroller. For details on commands, see "F²MC-8L MB89600 Series Programming Manual".

■ Structure of This Manual

This manual consists of the following 19 chapters.

CHAPTER 1 "OVERVIEW"
This chapter describes the features and basic specifications of the MB89890 series.

CHAPTER 2 "HANDLING OF ELECTRONIC DEVICES"
This chapter describes the precautions in handling the MB89890 series.

CHAPTER 3 "CPU"
This chapter describes the functions and operations of the CPU.

CHAPTER 4 "I/O PORTS"
This chapter describes the functions and operations of the I/O ports.

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

CHAPTER 6 "Watchdog Timer"
This chapter describes the functions and operations of the watchdog timer.

CHAPTER 7 "8-BIT PWM TIMER"
This chapter describes the functions and operations of the 8-bit PWM timer.

CHAPTER 8 "8/16-BIT TIMER/COUNTER"
This chapter describes the functions and operations of the 8/16-bit timer/counter.

CHAPTER 9 "8-BIT SERIAL I/O"
This chapter describes the functions and operations of the 8-bit serial I/O.

CHAPTER 10 "SERIAL I/O WITH 1-BYTE BUFFER"
This chapter describes the functions and operations of the serial I/O with 1-byte buffer.

CHAPTER 11 "SERIAL I/O PORT SWITCH CIRCUIT"
This chapter describes the functions and operations of the serial I/O port switching in the MB89890 Series.

CHAPTER 12 "BUZZER OUTPUT"
This chapter describes the functions and operations of the serial I/O port switch circuit.
CHAPTER 13 "EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)"
This chapter describes the functions and operations of the external interrupt circuit 1 (edge).

Chapter 14 "EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)"
This chapter describes the functions and operations of external interrupt circuit 2 (level).

CHAPTER 15 "A/D CONVERTER"
This chapter describes the functions and operations of the A/D converter.

CHAPTER 16 "WATCH PRESCALER"
This chapter describes the functions and operations of the watch prescaler.

CHAPTER 17 "DTMF GENERATOR"
This chapter describes the functions and operations of the DTMF generator.

CHAPTER 18 "MODEM TIMER"
This chapter describes the functions and operations of the modem timer.

CHAPTER 19 "MODEM SIGNAL OUTPUT CIRCUIT"
This chapter describes the functions and operations of the modem signal output circuit.

"APPENDIX"
The appendixes provide the I/O map, instruction lists, and other information.
• The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.
• The information and circuit diagrams in this document are presented as examples of semiconductor device applications, and are not intended to be incorporated in devices for actual use. Also, FUJITSU is unable to assume responsibility for infringement of any patent rights or other rights of third parties arising from the use of this information or circuit diagrams.
• The products described in this document are designed, 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.
READING THIS MANUAL

- Example of Notational Conventions of Register Names and Pin Names

  ○ Example of notational conventions of register names and pin names

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

    Bit name                  Register name                  Bit code

    Inhibit the interrupt request output of the timer base timer (TBTC: TBIE=0).

    Set data                  Bit code                     Register code

    If interrupts are enabled (CCR:I=1), the interrupt is accepted.

    Current status             Bit code                     Register code

  ○ Examples of notational conventions of dual-purpose pin

    P37/SO1 pin

    Some pins are dual-purpose pins that can be used for two functions by setting in a program. For the dual-purpose pins, the name corresponding to each function is represented by separating the name with a slash "/".
■ Documents and Development Tools Required for Development

The items required before developing this product are shown as follows.
For required documents and development tools, contact a Fujitsu sales representative.

☑ Manuals required for development

Check box

☐ "F₂MC-8L MB89890 Series data sheet"
   (The electric characteristics table and example of individual characteristics are described.)

☐ "F₂MC-8L MB89600 Series Programming Manual"
   (Manuals concerning instructions in F₂MC-8L family)

* "F₂MC-8L MB89600 Series C Compiler Manual"
   (Required only when you develop the product in C.)
   (Manual concerning the program development in C and the starting method)

* "F₂MC-8L MB89600 Series Assembler Manual"
   (Manual concerning the program development in assembler)

* "F₂MC-8L MB89600 Series Support System Manual"
   (Manual concerning the starting method of macro assembler, linker, and library manager)

* "F₂MC-8L MB89600 Series Software Simulator Manual"
   (Required only when you make evaluation with a simulator.)
   (Manual concerning the operation method of the software simulator)

The manuals marked with an asterisk "*" are provided with the products.
The manuals of the development tools are provided with the products.

☑ Software required for development

Check box

☐ C compiler (Required only when developing the product in C.)

☐ Assembler, linker, and librarian

☐ Software simulator (Required only when making an evaluation with a simulator.)

☐ Emulator debugger (Required only when making an evaluation with the MB2140A series.)

The type of software differs depending on the OS used.
For details, see "F₂MC Development Tool Catalog" or "Product Guide."
Items required when evaluating with one-time PROM microcomputer (As stated by the company)

Check box
- MB89P899
- ROM writer (A writer into which the MBM27C1001 can be written) For the recommended writers, see "Data Sheet."
- Package conversion adapter for writing (Purchase from Sun Hayato Co.)

<table>
<thead>
<tr>
<th>Package</th>
<th>Socket model</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-100P-M06 (0.65mm pitch)</td>
<td>ROM-100QF-32DP-8LA</td>
</tr>
</tbody>
</table>

Development tools

Check box
- MB89PV890 (piggyback model/evaluation chip)
- Evaluation tool (Main unit) (Pod) (Probe) MB2141A + MB2144-505 + MB2144-202

When you want to use a third party’s development environment, check with respective manufacturers.

References
- F²MC Development Tool Catalog
- Microcomputer Product Guide
# CONTENTS

## CHAPTER 1 OVERVIEW

- 1.1 Features of the MB89890 Series ................................................................. 2
- 1.2 MB89890 Series Configuration ................................................................. 4
- 1.3 Differences among Models ....................................................................... 6
- 1.4 Block Diagram .......................................................................................... 7
- 1.5 Pin Assignments ........................................................................................ 8
- 1.6 External Dimensions ................................................................................ 9
- 1.7 Pin Functions ............................................................................................ 11
- 1.8 Input Circuit Types .................................................................................. 15

## CHAPTER 2 HANDLING OF ELECTRONIC DEVICES

- 2.1 Precautions in Handling Electronic Devices ........................................... 20

## CHAPTER 3 CPU

- 3.1 Memory ..................................................................................................... 22
  - 3.1.1 Area for Specific Applications ............................................................. 24
  - 3.1.2 Allocation of 16-bit Data in Memory .................................................... 26
- 3.2 Dedicated Registers ................................................................................ 28
  - 3.2.1 Condition Code Register (CCR) ........................................................... 30
  - 3.2.2 Register Bank Pointer (RP) ................................................................. 33
- 3.3 General-purpose Registers ..................................................................... 34
- 3.4 Interrupts .................................................................................................. 36
  - 3.4.1 Interrupt Level Setting Registers 1, 2, 3 (ILR1, 2, 3) ................................. 37
  - 3.4.2 Flow of Interrupt Operation ................................................................. 38
  - 3.4.3 Multi-interrupts .................................................................................... 40
  - 3.4.4 Interrupt Processing Time ..................................................................... 41
  - 3.4.5 Stack Operation during Interrupt Processing ....................................... 42
  - 3.4.6 Stack Area for Interrupt Processing .................................................... 43
- 3.5 Reset ......................................................................................................... 44
  - 3.5.1 External Reset Pin ............................................................................... 46
  - 3.5.2 Reset Operation ................................................................................... 47
  - 3.5.3 States of Pins at the Occurrence of a Reset .......................................... 49
- 3.6 Clock ........................................................................................................ 50
  - 3.6.1 Clock Generator ................................................................................ 52
  - 3.6.2 Clock Controller ................................................................................ 54
  - 3.6.3 System Clock Control Register (SYCC) .............................................. 56
  - 3.6.4 Clock Mode ......................................................................................... 59
  - 3.6.5 Oscillation Stabilization Wait Time ..................................................... 62
- 3.7 Standby Mode ......................................................................................... 64
  - 3.7.1 Operating states in standby mode ....................................................... 66
  - 3.7.2 Sleep Mode ......................................................................................... 68
  - 3.7.3 Stop Mode ........................................................................................ 69
  - 3.7.4 Watch Mode ...................................................................................... 71
  - 3.7.5 Standby Control Register (STBC) ...................................................... 72
CHAPTER 1  OVERVIEW

This chapter describes the features and basic specifications of the MB89890 series.

1.1 "Features of the MB89890 Series"
1.2 "MB89890 Series Configuration"
1.3 "Differences among Models"
1.4 "Block Diagram"
1.5 "Pin Assignments"
1.6 "External Dimensions"
1.7 "Pin Functions"
1.8 "Input Circuit Types"
CHAPTER 1 OVERVIEW

1.1 Features of the MB89890 Series

The MB89890 Series are general-purpose one-chip microcontrollers that feature a compact instruction set, dual clock operation clock controls, and a wide selection of peripheral functions including A/D converter, timers, serial interface, PWM timer, modem, DTMF signal generation, and external interrupt.

■ Product Features

○ High-speed operation at low power consumption
  Minimum instruction execution time 0.5 μs (at 8 MHz source oscillation frequency)

○ F2MC-8L CPU core
  Instruction system suited for the controller
  • Addition and subtraction instructions
  • 16-bit manipulation
  • Branch instruction with bit test
  • Bit manipulation instruction

○ Dual clock operation control
  • Main clock: 8 MHz max.
    (Four types of speed switching, oscillation is stopped in sub-clock mode)
  • Sub-clock: 32.768 kHz (operating in sub-clock mode)

○ Five timer operation
  • 8-bit PWM timer (serving as an interval timer)
  • 8/16-bit timer/counter (8 bits x 2 channels or 16 bits x 1 channel)
  • 21-bit timebase timer
  • Watch prescaler (15 bits)

○ Two serial interface operation
  • Serial I/O
    Since the transfer direction (MSB first or LSB first) can be selected, communications with various equipment are possible.
  • Serial I/O with one-byte buffer
    Since the transfer direction (MSB first or LSB first) can be selected, communications with various equipment are possible. Signals can also be output as modem signals with this interface.
1.1 Features of the MB89890 Series

- **A/D converter (8 channels)**
  The A/D converter has the sense function with which voltages can be compared in 6.3 μs (at 8 MHz source oscillation frequency)

- **External interrupt**
  - External interrupt 1 (4 input 1 channel)
    Four inputs are independent and it can also be used for releasing power save mode (with edge detection function)
  - External interrupt 2 (12 input 1 channel)
    12 inputs are independent and it can also be used for releasing power save mode (with "L" level detection function)

- **Modem**
  - Modem timer (pulse width counter)
  - Modem signal output
  - DTMF modulator

- **Power save mode (stand-by mode)**
  - Stop mode (Since oscillation is stopped, there is almost no power consumption)
  - Sleep mode (Since the CPU is stopped, the power consumption is reduced to 1/3 the normal operation)
  - Watch mode (Since the operation of functions other than the watch prescaler is stopped, power consumption is very small.)
  - Sub-mode

- **I/O ports (85 max.)**
  - Output ports (Nch open-drain): 21
  - General-purpose I/O ports (Nch open-drain): 8
  - Output ports (CMOS): 8
  - General-purpose I/O ports (CMOS): 48

- **Package**
  QFP-100
1.2 MB89890 Series Configuration

The MB89890 Series includes four models. Table 1.2-1 "MB89890 Series Product Configuration" shows the product configurations for the MB89890 Series.

### MB89890 Series Product Configuration

#### Table 1.2-1 MB89890 Series Product Configuration

<table>
<thead>
<tr>
<th>Item</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MB89898</td>
</tr>
<tr>
<td>Type</td>
<td>Mass production models (mask ROM)</td>
</tr>
<tr>
<td>ROM capacity</td>
<td>48K x 8-bit (internal ROM)</td>
</tr>
<tr>
<td>RAM capacity</td>
<td>1.5K x 8-bit</td>
</tr>
<tr>
<td>Power save mode (standby mode)</td>
<td>Sleep mode, stop mode, watch mode, sub-mode</td>
</tr>
<tr>
<td>Process</td>
<td>CMOS</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>2.2 V to 6.0 V *1</td>
</tr>
</tbody>
</table>

*1: The voltage differs depending on the condition of the operation frequency.
*2: The voltage differs depending on the operating voltage of the EPROM used.
For the EPROM, use the MBM27C512-20TV.

#### Table 1.2-2 MB89890 Series CPU and Peripheral Functions

<table>
<thead>
<tr>
<th>Item</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MB89898</td>
</tr>
<tr>
<td>CPU function</td>
<td>Number of basic instructions : 136</td>
</tr>
<tr>
<td></td>
<td>Instruction bit length          : 8 bits</td>
</tr>
<tr>
<td></td>
<td>Instruction length              : 1 to 3 bytes</td>
</tr>
<tr>
<td></td>
<td>Data bit length                 : 1, 8, or 16 bits</td>
</tr>
<tr>
<td></td>
<td>Minimum instruction execution time : 0.5 μs to 8 μs (8 MHz), 61.0 μs (32.768 kHz)</td>
</tr>
<tr>
<td></td>
<td>Interrupt processing time       : 4.5 μs to 72 μs (8 MHz), 549.3 μs (32.768 kHz)</td>
</tr>
</tbody>
</table>
### Table 1.2-2 MB89890 Series CPU and Peripheral Functions (Continued)

<table>
<thead>
<tr>
<th>Item</th>
<th>MB89898</th>
<th>MB89899</th>
<th>MB89P899</th>
<th>MB89PV890</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ports</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output ports (Nch open-drain)</td>
<td>: 21 (8 ports also serve as analog inputs)</td>
<td>: 8 (8 ports also serve as resources)</td>
<td>: 8</td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (Nch open-drain)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output ports (CMOS)</td>
<td>: 8</td>
<td>: 8</td>
<td>: 48</td>
<td>: 85</td>
</tr>
<tr>
<td>General-purpose I/O ports (CMOS)</td>
<td></td>
<td></td>
<td>(16 ports also serve as external interrupt pins, 8 ports also serve as resources)</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>: 85</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Timebase timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>21 bits</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle:</td>
<td>(1.02 ms, 4.10 ms, 32.77 ms, 524.29 ms) for main clock 8 MHz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Watchdog timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset generation cycle:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(524.3 ms min.) for main clock 8 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(500 ms min.) for sub-clock 32.768 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8/16-bit timer/counter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During 2-channel 8-bit timer/counter operation (Timers 1 and 2 operate independently) or 16-bit timer/counter operation (operating clock cycle is 1.0μs to 64.0μs): square wave output possible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During Timer 1 or 16-bit timer/counter operation: event counting by external clock input possible.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Serial I/O</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-bit length</td>
<td>8-bit length</td>
<td></td>
<td>8-bit length</td>
<td></td>
</tr>
<tr>
<td>LSB first or MSB first selectable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer clock (3 types of external or internal clocks: 1μs, 4μs, 16μs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Serial I/O with one-byte buffer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-bit length</td>
<td>8-bit length</td>
<td></td>
<td>8-bit length</td>
<td></td>
</tr>
<tr>
<td>LSB first or MSB first selectable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer clock (3 types of external or internal clocks: 1μs, 4μs, 16μs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Buzzer output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output frequency:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(977 Hz, 1953 Hz, 3906 Hz, 7813 Hz) for main clock 8 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1024 Hz, 2048 Hz, 4096 Hz) for sub-clock 32.768 kHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>External interrupt 1 (wake-up)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 input 1 channel (request flag independent, request output permission independent)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edge selectable (rising/falling)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Also used for releasing from stop/sleep (Edge can be detected in stop mode)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>External interrupt 2 (wake-up)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 input 1 channel (&quot;L&quot; level interrupt, input permission independent)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Also used for releasing from stop/sleep (Level can be detected in stop mode)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>A/D converter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-bit precision 8 channels</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D conversion function (conversion time 22 μs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sense function (comparison time 6.3 μs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>With reference voltage input (AVR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Watch prescaler</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle: (31.25 ms, 0.25s, 0.50s, 1.00s) for sub-clock 32.768 kHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DTMF</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>All CCITT tones selectable as output</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source oscillation frequency (4 MHz/8 MHz ready)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Modem timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5-bit noise reduction circuit (digital low-pass filter)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse measurement operation (&quot;L,&quot; &quot;H&quot; level independent data register)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Modem signal output circuit</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rectangular output frequency (approx. 1208bps, approx. 2415bps) selectable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source oscillation frequency (4/8 MHz selectable)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The clock cycle and conversion time without the specification of the operating clock are the values used when the highest speed (8 MHz) is selected for the main clock.
1.3 Differences among Models

This section describes the difference among 4 types of the MB89890 Series models and precautions in selecting models.

Differences among Models and Precautions in Selecting Models

Table 1.3-1 Correspondence between Packages and Models

<table>
<thead>
<tr>
<th>Package</th>
<th>MB89898</th>
<th>MB89899</th>
<th>MB89P899</th>
<th>MB89PV890</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-100P-M06</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>MQP-100C-P01</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

Y: Available  
N: Not available

Memory

When evaluating with piggyback models, check for differences in the model to be actually used before making an evaluation. In the following case, particular care should be taken. (See 3.1 "Memory.")

- When the stack area is set to the upper limit of the RAM

Current consumption

- In the MB89PV890, the current consumed by the EPROM connected to the top socket is added.
- When operating at low speed, the current consumption of one-time PROM, EPROM mounted devices is greater than that of mask ROM mounted devices. However, the current consumption in sleep/stop mode is almost the same.

Reference

For details on each package, see Section 1.6 "External Dimensions."

For details on current consumption, see the electric characteristics in "Data Sheet."

Mask options

The functions that can be selected as options and the method of specifying options differ from model to model.

See Appendix C "Mask Option," to use the mask options.

In particular, exercise caution with the following.

- In the MB89PV890, the options are fixed.
- In the MB89P899, the pull-up resistor options are specified in two bits for P00-07, P10-17, P60-67, P90-97, and PA0-PA7 and specified in bits for P40-44, P70-77, and P80-87.
1.4 Block Diagram

The whole block diagram of the MB89890 Series is shown in Figure 1.4-1 "Whole Block Diagram of the MB89890 Series".

Whole Block Diagram of the MB89890 Series

![Block Diagram Image]
1.5 Pin Assignments

The pin assignments in the MB89890 Series are shown in Figure 1.5-1 "Pin Assignments of FPT-100P-M06, MQP-100C-P01".

- Pin Assignments of FPT-100P-M06, MQP-100C-P01

Figure 1.5-1 Pin Assignments of FPT-100P-M06, MQP-100C-P01

*: Package top pin assignments (MB89PV890 only)

<table>
<thead>
<tr>
<th>No.</th>
<th>Pin name</th>
<th>No.</th>
<th>Pin name</th>
<th>No.</th>
<th>Pin name</th>
<th>No.</th>
<th>Pin name</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>N.C.</td>
<td>109</td>
<td>A2</td>
<td>117</td>
<td>N.C.</td>
<td>125</td>
<td>CE</td>
</tr>
<tr>
<td>102</td>
<td>A15</td>
<td>110</td>
<td>A1</td>
<td>118</td>
<td>O4</td>
<td>126</td>
<td>N.C.</td>
</tr>
<tr>
<td>103</td>
<td>A12</td>
<td>111</td>
<td>A0</td>
<td>119</td>
<td>O5</td>
<td>127</td>
<td>A11</td>
</tr>
<tr>
<td>104</td>
<td>A7</td>
<td>112</td>
<td>N.C.</td>
<td>120</td>
<td>O6</td>
<td>128</td>
<td>A9</td>
</tr>
<tr>
<td>105</td>
<td>A6</td>
<td>113</td>
<td>O1</td>
<td>121</td>
<td>O7</td>
<td>129</td>
<td>A8</td>
</tr>
<tr>
<td>106</td>
<td>A5</td>
<td>114</td>
<td>O2</td>
<td>122</td>
<td>O8</td>
<td>130</td>
<td>A13</td>
</tr>
<tr>
<td>107</td>
<td>A4</td>
<td>115</td>
<td>O3</td>
<td>123</td>
<td>CE</td>
<td>131</td>
<td>A14</td>
</tr>
<tr>
<td>108</td>
<td>A3</td>
<td>116</td>
<td>Vss</td>
<td>124</td>
<td>A10</td>
<td>132</td>
<td>Vcc</td>
</tr>
</tbody>
</table>

N.C.: No connection (Do not use as these pins are connected internally)
1.6 External Dimensions

The MB89890 Series provides two types of packages. Figure 1.6-1 "External Dimensions of FPT-100P-M06" to Figure 1.6-2 "External Dimensions of MQP-100C-P01" show external dimensions.

### External Dimensions of FPT-100P-M06

**Figure 1.6-1 External Dimensions of FPT-100P-M06**

<table>
<thead>
<tr>
<th>Package Width x Package Length</th>
<th>14.00 x 20.00 mm</th>
</tr>
</thead>
<tbody>
<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>3.35 mm MAX</td>
</tr>
</tbody>
</table>

100-pin plastic QFP

(FPT-100P-M06)

Note: Pins width and pins thickness include plating thickness.

Dimensions in mm (inches).
Figure 1.6-2 External Dimensions of MQP-100C-P01

- **Lead pitch**: 0.65 mm
- **Lead shape**: Straight
- **Mother board material**: Ceramic
- **Material of mounted socket**: Plastic

© 1994 FUJITSU LIMITED M10001SC-1-2

Unit: mm (inches)
## 1.7 Pin Functions

Table 1.7-1 "Pin Functions" and Table 1.7-2 "External EPROM Pin Functions (MB89PV890 Only)" show I/O pins and their functions. The letters in the "I/O Circuit Type" column correspond to those in the "Type" column in Table 1.8-1 "I/O Circuit Types".

### Pin Functions

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>I/O Circuit Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>X0</td>
<td>A</td>
<td>Crystal oscillator signal pins for the main clock. When using an external clock, enter signals to X0 and keep X1 open.</td>
</tr>
<tr>
<td>7</td>
<td>X1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>X1A</td>
<td>B</td>
<td>Crystal oscillator signal pins for the sub-clock. (For low speed: 32.768 kHz) When using an external clock, enter signals to X0A and keep X1A open.</td>
</tr>
<tr>
<td>3</td>
<td>X0A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MOD0</td>
<td>C</td>
<td>Memory access mode setting input pins. When in use, connect to the Vss (GND) pin.</td>
</tr>
<tr>
<td>5</td>
<td>MOD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>RST</td>
<td>D</td>
<td>Reset I/O pin. Nch open-drain output with pull-up resistor and hysteresis input. Upon internal reset request, &quot;L&quot; is output from the pin. The internal circuit is also initialized by entering &quot;L.&quot;</td>
</tr>
<tr>
<td>10 to 17</td>
<td>P00 to P07</td>
<td>E</td>
<td>General-purpose I/O ports</td>
</tr>
<tr>
<td>18 to 25</td>
<td>P10 to P17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26 to 33</td>
<td>P20 to P27</td>
<td>G</td>
<td>General-purpose output port</td>
</tr>
<tr>
<td>39</td>
<td>P30/PWM</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as 8-bit PWM timer square wave output or PWM wave output pin.</td>
</tr>
<tr>
<td>40</td>
<td>P31/BZ</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as a buzzer output pin.</td>
</tr>
<tr>
<td>41</td>
<td>P32/MSKI</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as a modem timer.</td>
</tr>
<tr>
<td>42</td>
<td>P33</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input</td>
</tr>
<tr>
<td>43</td>
<td>P34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 1.7-1 Pin Functions (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>I/O Circuit Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>P35/SK1</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as serial I/O clock I/O pin 1.</td>
</tr>
<tr>
<td>45</td>
<td>P36/SI1</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as serial I/O data input pin 1.</td>
</tr>
<tr>
<td>46</td>
<td>P37/SO1</td>
<td>F</td>
<td>General-purpose I/O port. It also serves as serial I/O data output pin 1.</td>
</tr>
<tr>
<td>34 to 38</td>
<td>P40 to P44</td>
<td>J</td>
<td>Nch open-drain output-only port.</td>
</tr>
<tr>
<td>85 to 92</td>
<td>P50/AN00 to P57/AN07</td>
<td>H</td>
<td>Nch open-drain general-purpose output port. They also serve as analog input pins.</td>
</tr>
<tr>
<td>47</td>
<td>P60/TMO1</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as 8/16-bit timer/counter output pin 1.</td>
</tr>
<tr>
<td>48</td>
<td>P61/TMO2</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as 8/16-bit timer/counter output pin 2.</td>
</tr>
<tr>
<td>49</td>
<td>P62/TCLK</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as 8/16-bit timer/counter clock output pin.</td>
</tr>
<tr>
<td>51</td>
<td>P63/MSKO</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as modem output.</td>
</tr>
<tr>
<td>52</td>
<td>P64</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input.</td>
</tr>
<tr>
<td>53</td>
<td>P65/BSK1</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as clock I/O pin 1 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>54</td>
<td>P66/BSI1</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also serves as data input pin 1 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>55</td>
<td>P67/BSO1</td>
<td>F</td>
<td>General-purpose I/O port, hysteresis input. It also functions as data output pin 1 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>56</td>
<td>P70/SK2</td>
<td>I</td>
<td>Nch open-drain general-purpose I/O port. It also serves as serial I/O clock I/O pin 2.</td>
</tr>
<tr>
<td>57</td>
<td>P71/SI2</td>
<td>I</td>
<td>Nch open-drain general-purpose I/O port. It also serves as serial I/O data input pin 2.</td>
</tr>
<tr>
<td>58</td>
<td>P72/SO2</td>
<td>I</td>
<td>Nch open-drain general-purpose I/O port. It also serves as serial I/O data output pin 2.</td>
</tr>
</tbody>
</table>
Table 1.7-1 Pin Functions (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>I/O Circuit Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>P73/BSK2</td>
<td></td>
<td>Nch open-drain general-purpose I/O port. It also serves as clock I/O pin 2 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>61</td>
<td>P74/BSI2</td>
<td>I</td>
<td>Nch open-drain general-purpose I/O port. It also serves as data input pin 2 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>62</td>
<td>P75/BSO2</td>
<td></td>
<td>Nch open-drain general-purpose I/O port. It also serves as data output pin 2 of serial I/O with one-byte buffer.</td>
</tr>
<tr>
<td>63</td>
<td>P76</td>
<td>I</td>
<td>Nch open-drain general-purpose I/O port.</td>
</tr>
<tr>
<td>64</td>
<td>P77</td>
<td></td>
<td>Nch open-drain general-purpose I/O port.</td>
</tr>
<tr>
<td>65 to 72</td>
<td>P80 to P87</td>
<td>J</td>
<td>Nch open-drain general-purpose output port.</td>
</tr>
<tr>
<td>73 to 80</td>
<td>P90/INT20 to P97/INT27</td>
<td>F</td>
<td>General-purpose I/O ports. They also function as external interrupt 2 input. External interrupt 2 is hysteresis input.</td>
</tr>
<tr>
<td>81</td>
<td>PA0/INT28</td>
<td></td>
<td>General-purpose I/O port. They also function as external interrupt 1 input. External interrupt 1 is hysteresis input.</td>
</tr>
<tr>
<td>82</td>
<td>PA1/INT29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>PA2/INTA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>PA3/INTB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>97 to 100</td>
<td>PA4/INT0 to PA7/INT3</td>
<td>F</td>
<td>General-purpose I/O port. They also function as external interrupt 1 input. External interrupt 1 is hysteresis input.</td>
</tr>
<tr>
<td>95</td>
<td>DTMF</td>
<td>K</td>
<td>DTMF signal output pin</td>
</tr>
<tr>
<td>1, 50</td>
<td>Vcc</td>
<td>-</td>
<td>Power supply pin</td>
</tr>
<tr>
<td>8, 59</td>
<td>Vss</td>
<td>-</td>
<td>Power supply (GND) pin</td>
</tr>
<tr>
<td>93</td>
<td>Vcc (AVcc)</td>
<td>-</td>
<td>A/D converter power supply pin</td>
</tr>
<tr>
<td>94</td>
<td>AVR</td>
<td>-</td>
<td>A/D converter reference voltage input pin</td>
</tr>
<tr>
<td>84</td>
<td>Vss (AVss)</td>
<td>-</td>
<td>A/D converter power supply (GND) pin</td>
</tr>
</tbody>
</table>

*1: FPT-100P-M06
*2: MQP-100C-P01
**Table 1.7-2 External EPROM Pin Functions (MB89PV890 Only)**

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Pin Name</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>A15</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>103</td>
<td>A12</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>104</td>
<td>A7</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>105</td>
<td>A6</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>106</td>
<td>A5</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>107</td>
<td>A4</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>108</td>
<td>A3</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>109</td>
<td>A2</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>110</td>
<td>A1</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>111</td>
<td>A0</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>113</td>
<td>O1</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>114</td>
<td>O2</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>115</td>
<td>O3</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>116</td>
<td>Vss</td>
<td>O</td>
<td>Power supply (GND) pin</td>
</tr>
<tr>
<td>118</td>
<td>O4</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>119</td>
<td>O5</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>120</td>
<td>O6</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>121</td>
<td>O7</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>122</td>
<td>O8</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>123</td>
<td>CE</td>
<td>O</td>
<td>Chip enable pin for ROM Outputs &quot;H&quot; in standby mode.</td>
</tr>
<tr>
<td>124</td>
<td>A10</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>125</td>
<td>OE</td>
<td>O</td>
<td>Output enable pin for ROM Always outputs &quot;L.&quot;</td>
</tr>
<tr>
<td>127</td>
<td>A11</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>128</td>
<td>A9</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>129</td>
<td>A8</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>130</td>
<td>A13</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>131</td>
<td>A14</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>132</td>
<td>Vss</td>
<td>O</td>
<td>Power supply pin for EPROM</td>
</tr>
<tr>
<td>101</td>
<td>N.C.</td>
<td>-</td>
<td>Internal connection pin. Do not connect</td>
</tr>
<tr>
<td>112</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>117</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>126</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1.8 Input Circuit Types

Table 1.8-1 "I/O Circuit Types" shows the I/O circuit types. The letters in the "Type" column in Table 1.8-1 "I/O Circuit Types" correspond to the "I/O Circuit Type" column in Table 1.7-1 "Pin Functions".

<table>
<thead>
<tr>
<th>Type</th>
<th>Circuit</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| A    | ![Main clock control signal](image) | Main clock control signal  
  - Oscillation feedback resistor  Approx. 1 MΩ (5 V) |
| B    | ![Low speed (sub-clock oscillation)](image) | Low speed (sub-clock oscillation)  
  - Oscillation feedback resistor  Approx. 4.5 MΩ (5 V) |
| C    | ![Circuit](image) | — |
| D    | ![Output pull-up resistor (Pch)](image) |  
  - Output pull-up resistor (Pch)  Approx. 50 kΩ (5 V)  
  - Hysteresis input |
### Table 1.8-1 I/O Circuit Types (Continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Circuit</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| **E** | ![Circuit Diagram](image) | • CMOS output  
• CMOS input  
• Pull-up resistor option selectable (excluding MB89PV890)  
Approx. 50 kΩ (5 V) |
| **F** | ![Circuit Diagram](image) | • CMOS output  
• Hysteresis input  
• Pull-up resistor option selectable (excluding MB89PV890)  
Approx. 50 kΩ (5 V) |
| **G** | ![Circuit Diagram](image) | • CMOS output |
| **H** | ![Circuit Diagram](image) | • Nch open-drain output  
• Analog input |
| **I** | ![Circuit Diagram](image) | • Nch open-drain output  
• Hysteresis input  
• Pull-up resistor option selectable (excluding MB89PV890)  
Approx. 50 kΩ (5 V) |
### Table 1.8-1 I/O Circuit Types (Continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Circuit</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| J    | ![Circuit Diagram J](image1) | • Nch open-drain output  
• Pull-up resistor option selectable (excluding MB89PV890)  
Approx. 50 kΩ (5 V) |
| K    | ![Circuit Diagram K](image2) | • DTMF analog output |
This chapter describes the precautions in handling the MB89890 series.

2.1 "Precautions in Handling Electronic Devices"
2.1 Precautions in Handling Electronic Devices

This section describes the precautions on handling of the supply voltage of devices and treatment of pins.

- **Precautions in Handling Electronic Devices**

  - **Take Care to Ensure that the Device does not Exceed the Maximum Rated Voltage (to Prevent Latch-up)**
    
    In CMOS IC devices, latch-up may occur if a voltage higher than Vcc or a voltage lower than Vss is applied to pins that are not designed to carry medium or high voltages or if a voltage exceeding the rated voltage is applied between Vcc and Vss.
    
    Latch-up is characterized by a server rise in supply current and can lead to thermal destruction of semiconductor elements. When using the IC, take care to ensure that the device does not exceed the maximum rated voltage.
    
    When turning on and off analog systems, care must also be taken to ensure that the analog power supply (AVcc, AVR) and analog input voltages do not exceed the digital supply voltage (Vcc).

  - **Stabilize the Supply Voltage as much as Possible.**
    
    Though Vcc supply voltage is warranted to remain within the operating range listed in the specifications, sudden changes in supply voltage within that range can cause abnormal operations, and so precautions should be taken to stabilize the supply voltage to the IC.
    
    As guides for stabilization, it is recommended that Vcc ripple fluctuations (P-P value) in commercial frequencies (50 to 60Hz) be restricted to 10% or less of the typical Vcc and at the momentary fluctuation at power supply switching the coefficient of transient fluctuation be restricted to no more than 0.1 V/ms.

  - **Handling of Unused Input Pins**
    
    Any unused input signal pins that are left open are potential sources of abnormal operating conditions. All such pins should be pulled up or down.

  - **Handling of N.C. Pins**
    
    Be sure to keep N.C. (internal connection) pins open.

  - **Handling of Power Supply Pins in A/D Converter and D/A Converter Mounted Models**
    
    A/D converters and D/A converters not in use should be connected in such a manner AVcc = DAVC = Vcc and AVss = AVR = Vss hold.

  - **Precautions in Using an External Clock Signal**
    
    Even if an external clock signal is used, oscillator stabilization time is required to restore from power-on reset (optional), sub-clock mode, and stop mode.
CHAPTER 3  CPU

This chapter describes the functions and operations of the CPU.

3.1 "Memory"
3.2 "Dedicated Registers"
3.3 "General-purpose Registers"
3.4 "Interrupts"
3.5 "Reset"
3.6 "Clock"
3.7 "Standby Mode"
3.8 "Memory Access Mode"
CHAPTER 3 CPU

3.1 Memory

The MB89890 Series, having a total memory area of 64K bytes, consists of I/O, RAM, ROM, and external areas. In the memory configuration, there are areas used for specific applications, such as general-purpose registers and vector table.

Memory Configuration

- **I/O area (address: 0000\text{H} to 007F\text{H])**
  - Control registers and data registers of the built-in peripheral functions are allocated to the I/O area.
  - Since the I/O area is allocated in a part of the memory, it can be accessed in the same manner as memory. Use of direct addressing allows faster access.

- **RAM area**
  - Static RAM is embedded as an internal data area.
  - The internal RAM capacity is different from model to model.
  - The area from 80\text{H} to FF\text{H} can be addressed at high speed by direct addressing.
  - The area from 100\text{H} to 1FF\text{H} can be used as a general-purpose register area.
  - By resetting, RAM data becomes undefined.

- **ROM area**
  - ROM is embedded as an internal program area.
  - The internal ROM capacity is different from model to model.
  - The area from FFC0\text{H} to FFFF\text{H} is used as a vector table or other applications.
3.1 Memory

Figure 3.1-1 Memory Map

[Diagram of Memory Map showing memory allocation and access inhibition marks.]
3.1.1 Area for Specific Applications

Besides the I/O area, the areas for specific applications include the general-purpose register area and vector table area.

- **General-purpose Register Area (Address: 0100\textsubscript{H} to 01FF\textsubscript{H})**
  
  - In this area, auxiliary registers used for 8-bit computations and transfer are placed.
  - This area is allocated in a part of the RAM area and can also be used as ordinary RAM.
  - When this area is used as general-purpose registers, high-speed access can be made with a short instruction by general-purpose register addressing.

  For details, see Section 3.2.2 "Register Bank Pointer" and Section 3.3 "General-purpose Register."

- **Vector Table Area (Address: FFC0\textsubscript{H} to FFFF\textsubscript{H})**

  - This area is used as a vector table for vector call instructions, interrupts, and resets.
  - This area is allocated in the top portion of the ROM area. In each vector table address, set the start address of the corresponding process routine as data.

  Table 3.1-1 "Vector Table" shows the addresses of the vector table referenced in correspondence with vector call instructions, interrupts, and resets.

  For details, see Section 3.4 "Interrupts," Section 3.5 "Reset," and Item (6), "CALLV \#vct" in Section B.3 "Special Instructions."
### Table 3.1-1 Vector Table

<table>
<thead>
<tr>
<th>Vector call instruction</th>
<th>Vector table address</th>
<th>Interrupt name</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Upper</td>
<td>Lower</td>
<td>Upper</td>
</tr>
<tr>
<td>CALLV #0</td>
<td>FFC0&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFC1&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQB</td>
</tr>
<tr>
<td>CALLV #1</td>
<td>FFC2&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFC3&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQA</td>
</tr>
<tr>
<td>CALLV #2</td>
<td>FFC4&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFC5&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ9</td>
</tr>
<tr>
<td>CALLV #3</td>
<td>FFC6&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFC7&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ8</td>
</tr>
<tr>
<td>CALLV #4</td>
<td>FFC8&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFC9&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ7</td>
</tr>
<tr>
<td>CALLV #5</td>
<td>FFCA&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFCB&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ6</td>
</tr>
<tr>
<td>CALLV #6</td>
<td>FFCC&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFCD&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ5</td>
</tr>
<tr>
<td>CALLV #7</td>
<td>FFCE&lt;sub&gt;H&lt;/sub&gt;</td>
<td>FFCF&lt;sub&gt;H&lt;/sub&gt;</td>
<td>IRQ4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IRQ3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IRQ2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IRQ1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IRQ0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mode data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Reset vector</td>
</tr>
</tbody>
</table>

*: FFFC<sub>H</sub> cannot be used. (Set to FF<sub>H</sub>)
CHAPTER 3  CPU

3.1.2  Allocation of 16-bit Data in Memory

For 16-bit data, high byte data is stored in the lower address number in the memory and stack.

■ 16-bit Data Storage in RAM

When 16-bit data is written to memory, the high-byte data is stored in the lower address and the low-byte data is stored in the next address. When 16-bit data is fetched, it is handled in the same way.

Figure 3.1-2 “Allocation of 16-bit Data in Memory” shows the allocations of 16-bit data in memory.

![Figure 3.1-2 Allocation of 16-bit Data in Memory](image)

■ Storage of 16-bit Operand

Similarly, when an instruction operand refers to 16-bit data, high-byte data is stored in the address closest to the opcode value and low-byte data is stored in the next address.

This allocation holds true irrespective of whether an operand refers to a memory address or 16-bit immediate data.

Figure 3.1-3 "Allocation of 16-bit Data in Instructions" indicates the allocation of 16-bit data in instructions.
### Figure 3.1-3  Allocation of 16-bit Data in Instructions

![Figure 3.1-3 Allocation of 16-bit Data in Instructions](image)

#### Storage of 16-bit Data in Stack

Similarly, when 16-bit register data is saved on the stack, high-byte data is stored in the lower address.
CHAPTER 3 CPU

3.2 Dedicated Registers

The dedicated registers in the CPU consists of a program counter (PC), two operation registers (A, T), three address pointers (IX, EP, SP), and a program status (PS). Each register is 16 bits long.

■ Dedicated Register Configuration

The dedicated registers in the CPU are made of seven 16-bit registers. For some registers, only lower 8-bits can be used.

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

Figure 3.2-1 Dedicated Register Configuration

<table>
<thead>
<tr>
<th>Initial value</th>
<th>16 bits</th>
<th>Program counter: A register indicating the location in which the present instruction is stored.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFDH</td>
<td>PC</td>
<td></td>
</tr>
<tr>
<td>Undefined</td>
<td>A</td>
<td>Accumulator: A temporary memory register for calculation and transfer.</td>
</tr>
<tr>
<td>Undefined</td>
<td>T</td>
<td>Temporary accumulator: Used to perform calculation together with the accumulator.</td>
</tr>
<tr>
<td>Undefined</td>
<td>IX</td>
<td>Index register: A register indicating the index address.</td>
</tr>
<tr>
<td>Undefined</td>
<td>EP</td>
<td>Extra pointer: A pointer indicating the memory address.</td>
</tr>
<tr>
<td>Undefined</td>
<td>SP</td>
<td>Stack pointer: A pointer indicating the present stack location</td>
</tr>
<tr>
<td>!flag = 0, IL1,0=11</td>
<td>R P</td>
<td>Program status: A register that stores register bank pointers or condition codes.</td>
</tr>
<tr>
<td>Other bits are undefined.</td>
<td>CCR</td>
<td></td>
</tr>
<tr>
<td>P S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

■ Functions of Dedicated Registers

- **Program counter (PC)**
  The program counter, a 16-bit counter, indicates the memory address of the instruction the CPU is executing. The contents of the program counter are updated by the execution of an instruction, interrupt, and reset. The initial value at reset operation is the mode data read address (FFFDH).

- **Accumulator (A)**
  The accumulator, a 16-bit register for calculation, performs various types of calculations and transfers with data in memory and data in other registers such as the temporary accumulator (T). The data in the accumulator can be handled as both word length (16-bit) data and byte length (8-bit) data. When byte-length calculation or transfer is performed, only the lower eight bits (AL) of the accumulator are used and the upper eight bits (AH) remain unchanged. The initial value after a reset is undefined.
3.2 Dedicated Registers

- **Temporary accumulator (T)**
  The temporary accumulator, a 16-bit calculation auxiliary register, performs various calculations with data in the accumulator (A). The data in the temporary accumulator handled as the word length (16 bits) when the calculation with the accumulator (A) is the word length and handled as the byte length (eight bits) when the calculation with the accumulator (A) is the byte length. When calculation in the byte length is performed, only the lower eight bits (TL) in the temporary accumulator are used and the upper eight bits (TH) are not used. When data is transferred to the accumulator (A) with a MOV type instruction, the data in the accumulator before transfer is transferred to the accumulator automatically. Also in this case, the upper eight bits (TH) are not changed if byte-length data is transferred. The initial value after a reset is undefined.

- **Index register (IX)**
  The index register, a 16 bit register, retains the index address. The index register is used together with one byte offset (-128 to +127). By adding a sign extended offset value to the index address, the memory address for data access is generated. The initial value after a reset is undefined.

- **Extra pointer (EP)**
  The extra pointer is a 16 bit register. The contents of this register become memory address to access data. The initial value after a reset is undefined.

- **Stack pointer (SP)**
  The stack pointer, a 16 bit register, retains the address referenced by an interrupt, subroutine calling, stack saving/restoration instruction. The value in the stack pointer indicates the address of the stack on which the latest data saved. The initial value after a reset is undefined.

- **Program status (PS)**
  The program status is a 16-bit control register. The upper eight bits are a register bank pointer (RP) used for indicating the address of the general-purpose register bank.

  The lower eight bits are a condition code register (CCR) consisting of various flags indicating the state of the CPU. Each eight-bit register is a part of the program status and cannot be accessed separately. (The instructions to access the program status are MOVW A, PS and MOVW PS, A only.)

  For details of the use of the dedicated registers, see "F²MC-8L MB89600 Series Programming Manual".
3.2.1 Condition Code Register (CCR)

The condition code register (CCR) that is the lower eight bits of the program status (PS) consists of the bits that indicate the contents of the calculation result and transferred data (C, V, Z, N, H) and the bits that control the acceptance of interrupt requests (I, IL1, IL0).

■ Condition Code Register (CCR) Configuration

![Figure 3.2-2 Condition Code Register Configuration](image)

■ Bits Indicating the Calculation Results

- **Half carry flag (H)**
  This flag is set to "1" when the result of a calculation includes carrying from bit 3 to bit 4 or borrowing from bit 4 to bit 3 and cleared to "0" in other cases. This flag is used for decimal adjustment instructions and can only be used in addition and subtraction.

- **Negative flag (N)**
  This flag is set to "1" when the highest bit indicates "1" as the result of a calculation and cleared to "0" when it indicates "0."

- **Zero flag (Z)**
  This flag is set to "1" when the result of a calculation is "0" and cleared to "0" in other cases.

- **Overflow flag (V)**
  This flag is set to "1" when the result of a calculation creates a two’s complement overflow and cleared to "0" when no overflow occurs.
3.2 Dedicated Registers

- **Carry flag (C)**

  This flag is set to "1" when the result of a calculation includes carrying from bit 7 or borrowing to bit 7 and cleared to "0" in other cases. It also takes the shift-out value when the shift instruction is used.

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

  ![Figure 3.2-3 Change of Carry Flag in Shift Instruction](image)

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

  Reference:
  
  When using flags, there is almost no case in which a bit in the flag is removed and used directly. Normally, it is used indirectly in a branch instruction (BNZ, etc.) and a decimal adjustment instruction (DAA, DAS). The initial value after resetting these flags is undefined.

- **Bits Controlling the Acceptance of Interrupts**

  - **Interrupt enable flag (I)**

    When this flag is "1," interrupts are enabled and the CPU accepts interrupts. When this flag is "0," interrupts are disabled and the CPU does not accept interrupts.

    The initial value after a reset is "0."

    Normally, this flag is set to "1" in the SETI instruction and cleared to "0" in the CLRI instruction.

  - **Interrupt level bit (IL1, 0)**

    This bit indicates the level of the interrupt the CPU is currently accepting. It is compared to the value of the interrupt setting register (ILR1 to 3) set in correspondence with the interrupt request (IRQ0 to IRQB) of each peripheral function.

    The CPU accepts the interrupt only when the interrupt enable flag is enabled (I=1) and an interrupt request with an interrupt level smaller than the value indicated by this bit occurs. The priority of interrupt levels is as shown in Table 3.2-1 "Interrupt Level". The initial value after a reset is "11B".
Table 3.2-1  Interrupt Level

<table>
<thead>
<tr>
<th>IL1</th>
<th>IL0</th>
<th>Interrupt level</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Highest</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>Lowest</td>
</tr>
</tbody>
</table>

Reference:

When the CPU is not processing an interrupt (when the CPU is processing the main program), the interrupt level bits (IL1, 0) are usually "11_2".

For details on interrupts, see Section 3.4 "Interrupts."
3.2 Dedicated Registers

### 3.2.2 Register Bank Pointer (RP)

The register bank pointer (RP) that is the upper eight bits of the program status (PS) indicates the current general-purpose register address, which is converted into the real address at general-purpose register addressing.

#### Register Bank Pointer (RP) Configuration

Figure 3.2-4 "Register Bank Pointer (RP) Configuration" shows the configuration of the register bank pointer.

**Figure 3.2-4 Register Bank Pointer (RP) Configuration**

<table>
<thead>
<tr>
<th>PS</th>
<th>R4</th>
<th>R3</th>
<th>R2</th>
<th>R1</th>
<th>R0</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>——</th>
<th>CCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP</td>
<td>bit15 bit14 bit13 bit12 bit11 bit10 bit 9 bit 8 bit 7 bit 6 bit 5 bit 4 bit 3 bit 2 bit 1 bit 0</td>
<td>PR initial value</td>
<td></td>
<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>X</td>
<td>Undefined</td>
<td></td>
<td></td>
<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>

The register bank pointer indicates the address of the register bank currently in use. Figure 3.2-5 "Real Address Conversion Rule in General-purpose Register Area" shows the conversion rule between the contents of the register bank pointer and the real address.

**Figure 3.2-5 Real Address Conversion Rule in General-purpose Register Area**

The register bank pointer specifies the memory block (register bank) used as a general-purpose register in the RAM area. There are 32 register banks in total and one register bank is specified by entering a value between 0 and 31 into the upper five bits of the register bank pointer. In one register bank, one of the eight 8-bit general-purpose registers is selected with the lower three bits of the op code.

With this register bank pointer, the area from \(0100_{H}\) to \(01FF_{H}\) can be used as a general-purpose register area at the maximum. When using only the internal RAM, however, there is a limit in the available area for some models. The initial value after a reset is undefined.

**Note:**

Before using a general-purpose register, be sure to set the register bank pointer (RP).

The register bank pointer is a part of the program status (PS) and only the register bank cannot be accessed independently.
3.3 General-purpose Registers

The general-purpose registers are a memory block consisting of memory banks. Each memory bank consists of eight 8-bit registers.
To specify the register bank, use the register bank pointer (RP).
Though up to 32 banks are available for use in total, there are cases in which all banks cannot be used when the internal RAM capacity is not enough.
The general-purpose registers are effectively used for vector call processing and subroutine calling.

■ General-purpose Register Configuration

- The general-purpose registers, eight-bit registers, are located in register banks in the general-purpose register area (in RAM).
- Each bank has eight registers (R0-R7) and a maximum of 32 banks are available for use. When using only the internal RAM, however, there is limit in the number of banks available for some models.
- The bank currently in use is indicated by the register bank pointer (RP) and the lower three bits in the op code selects one of general-purpose register 0 (R0) to general-purpose register 7 (R7).

Figure 3.3-1 "Register Bank Configuration" shows the configuration of the register banks.

Figure 3.3-1 Register Bank Configuration
3.3 General-purpose Registers

For general-purpose register area available in each model, see Section 3.1.1 "Area for specific applications."

- Features of General-purpose Registers

The general-purpose registers are featured as follows.

- High speed accessing to RAM with a short instruction (general-purpose register addressing)
- Because the registers are blocked by register banks, it is easy to protect data and divide function units.

For general-purpose registers, it is possible to allocate a dedicated register bank to an interrupt processing routine and vector call (CALLV #0-#7) in a fixed manner. For example, the following specification is possible: "Use the fourth register bank for the second interrupt."

In the case of interrupts, if the contents of the dedicated register bank corresponding to one interrupt are not rewritten by another routine accidentally, the general-purpose register before the interrupt is stored simply by specifying the dedicated register bank at the head of the interrupt processing routine. This function eliminates the need for saving the general-purpose register on the stack, making it possible to accept an interrupt at high speed without confusion.

For a subroutine calling, the general-purpose register can be protected and a reentrant program (a program that can be re-entered by not fixing the variable address) normally created using the index register (IX) can be implemented with the register bank.

Note:

When specifying the register bank by rewriting the register bank pointer (RP) during an interrupt processing routine, it is necessary to write a program in such a manner that the values of the interrupt level bits (CCR: IL1, 0) of the condition code register are not changed.
CHAPTER 3  CPU

3.4 Interrupts

The MB89890 Series has 12 interrupt requests corresponding to peripheral functions and their interrupt levels can be set independently. An interrupt request that occurred in a peripheral function is compared to the interrupt level by the interrupt controller when the interrupt request output of peripheral functions is enabled. The CPU performs an interrupt operation in accordance with the interrupt acceptance state. The occurrence of an interrupt request releases standby mode and restores an interrupt or standard operation.

Interrupt Requests from Peripheral Functions

Table 3.4-1 "Interrupt Requests and Interrupt Vectors" shows the interrupt requests corresponding to the peripheral functions. When an interrupt is accepted, the program branches to the interrupt processing routine with the contents of the interrupt vector table address corresponding to the interrupt request as the address of the branch destination.

The priority of each interrupt request is set to one of the three levels by the interrupt level setting registers (ILR1, 2, 3).

If the level of an interrupt request is equal to or lower than the level of the interrupt processing routine during execution, it is processed after the current interrupt processing routine is completed. If interrupt requests set to the same level occur simultaneously, the highest priority is given to IRQ0.

Table 3.4-1 Interrupt Requests and Interrupt Vectors

<table>
<thead>
<tr>
<th>Interrupt request</th>
<th>Vector table address</th>
<th>Bit names of interrupt level setting registers</th>
<th>Priority when the level is the same (at simultaneous occurrence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ0 (External interrupt 1)</td>
<td>FFFA H</td>
<td>FFFB H</td>
<td>L01, L00</td>
</tr>
<tr>
<td>IRQ1 (External interrupt 2)</td>
<td>FFF8 H</td>
<td>FFF9 H</td>
<td>L11, L10</td>
</tr>
<tr>
<td>IRQ2 (Modem signal output)</td>
<td>FFF6 H</td>
<td>FFF7 H</td>
<td>L21, L20</td>
</tr>
<tr>
<td>IRQ3 (Modem timer)</td>
<td>FFF4 H</td>
<td>FFF5 H</td>
<td>L31, L30</td>
</tr>
<tr>
<td>IRQ4 (Serial I/O with one-byte buffer)</td>
<td>FFF2 H</td>
<td>FFF3 H</td>
<td>L41, L40</td>
</tr>
<tr>
<td>IRQ5 (8-bit serial I/O)</td>
<td>FFF0 H</td>
<td>FFF1 H</td>
<td>L51, L50</td>
</tr>
<tr>
<td>IRQ6 (8/16-bit timer/counter)</td>
<td>FFFE H</td>
<td>FFEF H</td>
<td>L61, L60</td>
</tr>
<tr>
<td>IRQ7 (8-bit PWM timer)</td>
<td>FFE8 H</td>
<td>FFED H</td>
<td>L71, L70</td>
</tr>
<tr>
<td>IRQ8 (A/D converter)</td>
<td>FFEA H</td>
<td>FFE8 H</td>
<td>L81, L80</td>
</tr>
<tr>
<td>IRQ9 (Unused)</td>
<td>FFE8 H</td>
<td>FFE9 H</td>
<td>L91, L90</td>
</tr>
<tr>
<td>IRQA (Timebase timer)</td>
<td>FFE6 H</td>
<td>FFE7 H</td>
<td>LA1, LA0</td>
</tr>
<tr>
<td>IRQB (Watch prescaler)</td>
<td>FFE4 H</td>
<td>FFE5 H</td>
<td>LB1, LB0</td>
</tr>
</tbody>
</table>

Interrupt Requests and Interrupt Vectors
3.4 Interrupts

3.4.1 Interrupt Level Setting Registers 1, 2, 3 (ILR1, 2, 3)

12 pairs of 2-bit data corresponding to the interrupt requests from the peripheral functions are assigned to the interrupt level setting registers 1, 2, 3 (ILR1, 2, 3). The interrupt level of the peripheral function can be assigned to these two-bit data items (interrupt level setting bits).

Configuration of Interrupt Level Setting Registers 1, 2, 3 (ILR1, 2, 3)

Figure 3.4-1 Configuration of Interrupt Level Setting Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILR1</td>
<td>0 0 7 C8</td>
<td>L31</td>
<td>L30</td>
<td>L21</td>
<td>L20</td>
<td>L11</td>
<td>L10</td>
<td>L01</td>
<td>L00</td>
<td>111111111e</td>
</tr>
<tr>
<td></td>
<td></td>
<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></td>
</tr>
<tr>
<td>ILR2</td>
<td>0 0 7 D8</td>
<td>L71</td>
<td>L70</td>
<td>L61</td>
<td>L60</td>
<td>L51</td>
<td>L50</td>
<td>L41</td>
<td>L40</td>
<td>111111111e</td>
</tr>
<tr>
<td></td>
<td></td>
<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></td>
</tr>
<tr>
<td>ILR3</td>
<td>0 0 7 E8</td>
<td>L1B</td>
<td>L1A</td>
<td>L10</td>
<td>L91</td>
<td>L90</td>
<td>L81</td>
<td>L80</td>
<td>L70</td>
<td>111111111e</td>
</tr>
<tr>
<td></td>
<td></td>
<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></td>
</tr>
</tbody>
</table>

W: Write only

In the interrupt level setting registers, two bits are assigned to each interrupt request. The value of the interrupt level setting bits set in these registers determines the priority of the interrupt processing (Interrupt levels 1 to 3).

The interrupt level setting bits are compared to the interrupt level bits of the condition code register (CCR: IL1, 0).

When interrupt level 3 is set, the CPU does not accept the interrupt request.

Table 3.4-2 “Relation between Interrupt Setting Bits and Interrupt Levels” shows the relation between interrupt setting bits and interrupt levels.

Table 3.4-2 Relation between Interrupt Setting Bits and Interrupt Levels

<table>
<thead>
<tr>
<th>L01 to LB1</th>
<th>L00 to LB0</th>
<th>Interrupt request level</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Highest</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>Lowest</td>
</tr>
</tbody>
</table>

Reference:

When the main program is being executed, the interrupt level bits (CCR: IL1, 0) of the condition code register are usually "11B".

Note:

Because the ILR1, ILR2, ILR3 registers are write only, bit operation instructions (SETB, CLRb) cannot be used.
3.4.2 Flow of Interrupt Operation

When an interrupt request occurs from a peripheral function, the interrupt controller transfers the interrupt level to the CPU. If the CPU is ready to accept an interrupt, it stops the currently executed program temporarily and executes the interrupt processing routine.

Flow of Interrupt Operation

The interrupt operation is performed in the following order: detecting an interrupt source in a peripheral function, setting the interrupt request flag bit (request FF), evaluating the interrupt request enable bit (enable FF), evaluating the interrupt level (ILR1, 2, 3 and CCR: IL1, 0), evaluating the simultaneous request of the same level, and evaluating the interrupt enable flag (CCR: I).

Figure 3.4-2 "Flow of Interrupt Operation" shows the flow of interrupt operation.
3.4 Interrupts

(1) After a reset, all interrupt requests are disabled.
   Initialize the peripheral functions occurring interrupts with the peripheral function
   initialization program and set the interrupt level in the corresponding interrupt level setting
   registers (ILR1, 2, 3) before operating the peripheral functions.
   As the interrupt level, either 1, 2, or 3 can be set. Level 1 has the highest priority followed
   by Level 2. If Level 3 is set, interrupt of the corresponding peripheral function is disabled.

(2) Execute the main program (interrupt processing routine for multi-interrupt).

(3) If an interrupt source occurs in a peripheral function, the interrupt request enable bit
    (request FF) of the peripheral function is set to "1." When the interrupt request enable bit
    of the peripheral function is enabled (enable FF = 1), an interrupt request is output to the
    interrupt controller.

(4) The interrupt controller is always monitoring interrupt requests from peripheral functions.
    Of the interrupt levels corresponding to the currently occurring interrupts, the interrupt
    controller transfers the highest level to the CPU. When interrupt requests of the same
    interrupt levels occurs, the controller also evaluates their priorities.

(5) When the interrupt level received by the CPU is higher (when the level number is
    smaller) than the level set in the interrupt level bit (CCR: IL1, 0) of the condition code
    register (CCR: I), the CPU checks the contents of the interrupt enable flag (CCR: I) and
    accepts the interrupt if the interrupt is enabled (CCR: I = 1).

(6) The CPU saves the contents of the program counter (PC) and the program status (PS) on
    the stack, fetches the head address of the interrupt processing routine from the
    corresponding interrupt vector table, and changes the values of the interrupt level bit of
    the condition code register (CCR: IL1, 0) into the value of the interrupt level accepted
    before starting the execution of the interrupt processing routine.

(7) Finally, the CPU restores the values of the program counter (PC) and the program status
    (PS) saved on the stack with the RETI instruction and resumes the processing from the
    instruction next to the instruction executed immediately before the interrupt.

Note:
Since the interrupt request flag bit of the peripheral function is not cleared automatically even
after the interrupt request is accepted, it must be cleared with a program in the interrupt
processing routine (usually by writing "0" into the interrupt request flag bit).

Standby (power save ) mode is cleared by the occurrence of an interrupt. For details, see
Section 3.7 "Standby Mode."

Reference:
After the interrupt request flag is cleared at the top of the interrupt processing routine, the
peripheral function causing the interrupt can generate an interrupt again during the interrupt
processing routine (re-setting of the interrupt request flag bit). However, the interrupt is
usually accepted after the current interrupt processing routine is completed.
By setting different interrupt levels in the interrupt level setting registers (ILR1, 2, 3) for two or more interrupt requests from peripheral functions, multi-interrupts can be performed.

### Multi-interrupts

When an interrupt request having a higher level of priority occurs during the execution of an interrupt processing routine, the current interrupt processing is suspended and the interrupt request having the higher level of priority is accepted. For the interrupt level, either Level 1, 2, or 3 can be set. When interrupt level 3 is set, the CPU does not accept the interrupt request.

#### Example of multi-interrupts

As an example of multi-interrupts, the case in which a higher priority is given to the external interrupt than the timer interrupt is assumed. In this example, the timer interrupt level is set to 2 and the external interrupt level is set to 1. When an external interrupt occurs during the processing of a timer interrupt, processing shown in Figure 3.4-3 “Example of Multi-interrupts” is performed.

#### Figure 3.4-3 Example of Multi-interrupts

- During timer interrupt processing, the same value is set to the interrupt level bit of the condition code register (CCR: IL1, 0) and the interrupt level setting register (ILR1, 2, 3) corresponding to the timer interrupt (2 in the above example). When an interrupt having a higher level of priority (1 in the above example) occurs at this time, this interrupt processing routine is processed first.

- To disable multi-interrupts during the timer interrupt temporarily, set the interrupt enable flag of the condition code register to “disable” (CCR: I = 0) or set the interrupt level bits (IL1, 0) to “00B”.

- When the interrupt processing is completed and the interrupt restoration instruction (RETI) is executed, the values of the program counter (PC) and the program status (PS) saved on the stack are restored and the CPU restarts the processing of the interrupted program.

The condition code register (CCR) has the value before the interrupt when the program status (PS) is restored.


3.4 Interrupts

3.4.4 Interrupt Processing Time

From the occurrence of an interrupt request to the transition of control to the interrupt processing routine, the time required to complete the current instruction plus the interrupt handling time (the time required for the preparation for the interrupt processing) is required. This time is a maximum of 30 instruction cycles.

- **Interrupt Processing Time**

  From the occurrence of an interrupt request to the execution of the interrupt processing routine after the acceptance of the interrupt request, the interrupt request sample wait time plus the interrupt handling time is required.

- **Interrupt request sample wait time**

  The occurrence of an interrupt request is determined by sampling an interrupt request at the last cycle of each instruction. Therefore, the CPU cannot recognize interrupt requests during the execution of each instruction. When an interrupt request occurs immediately after the start of the execution of the DIVU instruction having the longest execution cycles (21 instruction cycles), this time becomes the longest.

- **Interrupt handling time**

  Nine instruction cycles are required to complete the following interrupt processing preparations when the CPU accepts an interrupt.
  
  - Saving the program counter (PC) and the program status (PS).
  - Setting the head address (interrupt vector) of the interrupt processing routine to PC.
  - Updating the interrupt level bits (PS: CCR: IL1, 0) of the program status (PS)

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

  ![Figure 3.4-4 Interrupt Processing Time](image)

  When an interrupt request occurs immediately after the start of the execution of the DIVU instruction having the longest execution cycles (21 instruction cycles), an interrupt processing time of 30 (21+9) instruction cycles is required. If the DIVU and MULU instructions are not used in the program, an interrupt processing time of 15 (6+9) instruction cycles is the longest.

  The instruction cycles are changed by switching the clock mode and the main clock speed (gear function). For details, see Section 3.6 "Clock."
### 3.4.5 Stack Operation during Interrupt Processing

This section explains the saving and restoration of the registers during interrupt processing.

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

When an interrupt is accepted, the CPU saves the contents of the current program counter (PC) and the program status (PS) on the stack automatically.

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

![Stack Operation at the Start of Interrupt Processing](image)

#### Stack Operation at the Time of Restoring from Interrupt

When the interrupt restoration instruction (RETI) after the completion of interrupt processing, the contents of the program status (PS) and the program counter (PC) are restored from the stack in this order, which is the reverse of the saving operation. Using this operation, the PS and PC are returned to the states existing immediately before the interrupt.

**Note:**

Since the accumulator (A) and the temporary accumulator (T) are not saved on the stack automatically, save and restore the contents of A and T with the PUSHW and POPW instructions.
3.4.6 Stack Area for Interrupt Processing

To execute interrupt processing, the stack area in RAM is used. The contents in the stack pointer (SP) are the head address of the stack area.

Stack area for Interrupt Processing

The stack area is also used for saving/restoring the program counter (PC) when executing a subroutine call instruction (CALL) or vector call instruction (CALLV) or for saving/restoring registers temporarily by PUSHW or POPW instruction.

- The stack area is reserved in RAM together with the data area.
- It is recommended that the initial setting be made in such a manner the stack pointer (SP) indicates the maximum value of the RAM address and the data area be assigned from the smaller RAM address.

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

Reference:

The stack area is used from larger address values to smaller address values by an interrupt, subroutine call, and PUSHW instruction and the stack area is released from smaller address values to larger address values by a restoration (RETI, RET) or POPW instruction. When the address values of the stack area in use become small due to multi-interrupts and subroutine call, ensure that the area does not overlap the data area retaining other data and the general-purpose register area.
There are four reset sources as shown below.

- External reset
- Software reset
- Watchdog reset
- Power-on reset (optional)

Depending on the operation mode and optional setting at the occurrence of a reset, main clock oscillation stabilization wait time is either required or not required.

### Reset Sources

<table>
<thead>
<tr>
<th>Reset source</th>
<th>Reset condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>External reset</td>
<td>The external reset pin is set to the &quot;L&quot; level.</td>
</tr>
<tr>
<td>Software reset</td>
<td>&quot;0&quot; is written into the software reset bit of the standby control register (STBC: RST).</td>
</tr>
<tr>
<td>Watchdog reset</td>
<td>Overflow of watchdog timer</td>
</tr>
<tr>
<td>Power-on reset</td>
<td>Power-on (models with power-on reset only)</td>
</tr>
</tbody>
</table>

#### External reset

An external reset is caused by setting the external reset pin (RST) to the "L" level. When the reset pin is set to the "H" level, the external reset is released.

The external reset pin also functions as an optional reset output pin.

#### Software reset

A software reset of four instruction cycles is caused by writing "0" into the software reset bit of the standby control register (STBC: RST).

#### Watchdog reset

A watchdog reset of four instruction cycles is effected when no data is written to the watchdog control register (WDTC) within the specified period after the start of the watchdog timer.

#### Power-on reset

Whether to use power-on reset can be specified as an option.

When power-on reset is used, a reset is effected by turning on the power.

When power-on reset is not used, an external reset circuit that effects a reset by turning on the power is required.
### Reset Sources and Main Clock Oscillation Stabilization Wait Time

Operation of the oscillation stabilization wait time differs depending on the operation mode at the occurrence of a reset and the optional setting of power-on reset.

After the completion of a reset, ordinary operation starts in the main clock mode regardless of the operation mode (clock mode or standby mode) before the reset and the reset source. Therefore, if a reset occurs when the oscillation of the main clock is stopped or during the main clock oscillation stabilization wait time, the main clock oscillation stabilization wait reset state occurs. For models with no power-on reset, the main clock oscillation stabilization wait time is not used when the power is turned on and an external reset occurs.

For software or watchdog resets, the main clock oscillation stabilization wait time is not used during the operation in main clock mode but the oscillation stabilization wait time is used during the operation in sub-clock mode because the oscillation of the main clock is stopped.

Table 3.5-2 "Reset Sources and Oscillation Stabilization Wait Time" shows the relation among reset sources, main clock oscillation stabilization wait time, and reset operation (mode fetch).

#### Table 3.5-2 Reset Sources and Oscillation Stabilization Wait Time

<table>
<thead>
<tr>
<th>Reset source</th>
<th>Operation state</th>
<th>Reset operation and main clock oscillation stabilization wait time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>With power-on reset</td>
</tr>
<tr>
<td>External reset*¹</td>
<td>At power-on, in stop or sub-clock mode</td>
<td>Reset operation is performed if the external reset is released after the expiration of the main clock oscillation stabilization wait time.*²</td>
</tr>
<tr>
<td>Software reset and watchdog reset</td>
<td>Main clock mode</td>
<td>Reset operation is performed after generating a reset of four instruction cycles.*³</td>
</tr>
<tr>
<td></td>
<td>Sub-clock mode</td>
<td>Reset operation is performed after the expiration of the main clock oscillation stabilization wait time.*²</td>
</tr>
<tr>
<td>Power-on reset</td>
<td>Reset operation is performed after the power is turned on and the main clock oscillation stabilization wait time has expired.*²</td>
<td></td>
</tr>
</tbody>
</table>

*¹: In the external reset during main clock mode operation, the oscillation stabilization wait time is not used. Reset operation is performed when the external reset is released.

*²: When the "reset output" option is selected, "L" level signals are output to the RST pin during the main clock oscillation stabilization wait time.

*³: When the "reset output" option is selected, "L" level signals are output to the RST pin for four instruction cycles.
The external reset pin generates a reset with the input of an "L" level signal. When the "reset output" option is selected, an "L" level signal is output depending on the internal reset source.

### Block Diagram of the External Reset Pin

The external reset pin (RST) in a model with reset output has hysteresis input and pulled-up Nch open-drain output.

The external reset pin in a model without reset output serves as a reset input pin only.

Figure 3.5-1 "Block Diagram of the External Reset Pin" shows the block diagram of the external reset pin.

![Block Diagram of the External Reset Pin](image)

### Functions of the External Reset Pin

The external reset pin (RST) generates an internal reset signal when an "L" level signal is entered.

For a model with reset output, an internal reset source and the oscillation stabilization wait time due to a reset causes the external reset pin to output an "L" level signal. Internal reset sources include software reset, watchdog reset, and power-on reset.

**Note:**

External reset input is accepted asynchronously regardless of the internal clock.

A clock is required to initialize an internal circuit and must be input when entering a reset, particularly when using an external clock.
3.5.2 Reset Operation

After a reset is released, the CPU fetches the mode data and reset vector from the internal ROM depending on the mode pin setting (mode fetch). When a model with power-on reset is turned on or restored from sub-clock or stop mode following a reset, a mode fetch is performed after the expiration of the oscillation stabilization wait time. If a reset is generated when writing to RAM, the contents of the address cannot be guaranteed.

■ Reset Operation Flow

![Figure 3.5-2 Reset Operation Flow](image)
CHAPTER 3 CPU

■ Mode Pins

The MB89890 Series is for single chip mode only. Be sure to set the mode pins (MOD1, MOD0) to “VSS, VSS.” Select the internal ROM as the location from which mode data and reset vectors are fetched.

Do not change the setting of the mode pins even after the completion of the reset operation.

■ Mode Fetch

When reset is released, mode data and reset vector are fetched from the internal ROM.

❖ Mode data (address: FFFD<sup>H</sup>)

For mode data, be sure to specify the single chip mode (00<sup>H</sup>).

❖ Reset vector (address: upper FFFE<sup>H</sup>/lower FFFF<sup>H</sup>)

Write the execution start address at the completion of the reset operation. Instructions are executed from the address therein.

■ Oscillation Stabilization Wait Time Reset State

When a model with power-on reset is turned on or restored from sub-clock or stop mode (main/sub) due to a reset, a reset operation for the external reset is executed following the expiration of the main clock oscillation stabilization wait time selected in optional setting. If external reset input is not released at this time, a reset operation is performed when the external reset is released.

Because the oscillation stabilization wait time is used even if an external clock is used, input of an internal clock is required at the time of the reset operation.

The main clock oscillation stabilization wait time is generated by the timebase timer.

For a model without power-on reset, because there is no oscillation stabilization wait reset state, inhibit the operation of the CPU by maintaining the external reset pin (RST) at the “L” level until oscillation is stabilized.

■ Influence of Resets on the Contents in RAM

External resets are not synchronized with the internal clock and internal resetting is effected directly. Therefore, the contents in RAM may be changed before and after a reset. Be sure to initialize RAM before using.
3.5.3 States of Pins at the Occurrence of a Reset

The state of each pin is initialized by a reset.

- **State of Pins during Reset Operation**
  When a reset source occurs, almost all I/O pins (resource pins) are set to high impedance and mode data is fetched from the internal ROM. (The pins having pull-up resistors as options are set to "H" level.)

- **State after Mode Data Fetch**
  The state of most of the I/O pins immediately after mode data fetch remain at high impedance. (The pins having pull-up resistors as options are set to "H" level)

  **Note:**
  For pins set to high impedance when a reset source occurs, exercise caution to prevent the equipment connected to the pins from malfunctioning.

  For the status of pins being reset, see Appendix E "Pin State of MB89890 Series."
The clock generator contains two systems of oscillation circuits. By connecting a resonator to each, a high-speed main clock and a low-speed sub clock can be generated independently (oscillation) and an externally generated clock can be input. The speed and supply of the dual clock operation is controlled in correspondence with clock mode and standby mode.

- **Clock Supply Map**

  Because clock oscillation and supply to the CPU and peripheral circuits (peripheral functions) are controlled by the clock controller, the operating clocks in the CPU and the peripheral circuit are influenced by the switching between the main clock and the sub-clock (clock mode), the main clock speed switching (gear function), and standby mode (sleep/stop/watch).

  The divided output of the free run counter operating in the clock for peripheral circuits is supplied to each peripheral function.

  There are also some peripheral circuits that are not influenced by the gear function and to which the divided output of the timebase timer operating in the main clock oscillation frequency divided by 2 or the divided output of the watch prescaler operating in sub-clock are supplied.

  Figure 3.6-1 "Clock Supply Map” shows the clock supply map.
Figure 3.6-1 Clock Supply Map

- **FCH**: Main clock source oscillation
- **FCL**: Sub-clock source oscillation
- **tinst**: Instruction cycle

*1 Not influenced by clock mode or gear function.
*2 The operation speed is influenced by clock mode and gear function.
*3 When the clock serving as oscillation (main clock or sub-clock) stops, operation is stopped.
*4 Selectable by the serial I/O port switching function.
3.6.1 Clock Generator

The permission and stop of the oscillation of the main clock and sub-clock are controlled by clock mode and stop mode.

- Clock Generator

  ○ Crystal or ceramic resonator

  Connect as shown in Figure 3.6-2 "Connection Example of Crystal or Ceramic Resonator".

  **Figure 3.6-2 Connection Example of Crystal or Ceramic Resonator**

  ![Diagram of Crystal or Ceramic Resonator Connection](image)

  **Reference:**

  A piezoelectric resonator containing an external capacitor (FAR series) can also be used. For details, see "Data Sheet."
**External clock**

As shown in Figure 3.6-3 "Connection Example of External Clock", connect the external clock to the X0 pin and keep the X1 pin open. To supply a sub-clock externally, connect the external clock to the X0A pin and keep the X1A pin open.

**Figure 3.6-3 Connection Example of External Clock**

<table>
<thead>
<tr>
<th>Dual clock operation option</th>
<th>Single clock system option</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main clock oscillation circuit</strong></td>
<td><strong>Sub-clock oscillation circuit</strong></td>
</tr>
<tr>
<td>MB89890 series</td>
<td>MB89890 series</td>
</tr>
<tr>
<td>X0</td>
<td>X1</td>
</tr>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>
3.6.2 Clock Controller

The clock controller consists of the following seven blocks.

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

Clock Controller Block Diagram

Figure 3.6-4 "Clock Controller Block Diagram" shows the block diagram of the clock controller.
3.6 Clock

- **Main clock oscillation circuit**
  An oscillation circuit of the main clock. In main stop mode and in sub-clock mode, oscillation is stopped.

- **Sub-clock oscillation circuit**
  An oscillation circuit of the sub-clock. In a mode other than sub-stop mode, it is always oscillating.
  It does not operate in the single clock option.

- **System clock selector**
  The system clock selector selects one from four types of clocks made by dividing the main clock oscillation and sub-clock and supplies it to the clock control circuit.

- **Clock control circuit**
  The clock control circuit controls the supply of the operating clock to the CPU and peripheral circuits depending on whether operation is in standard operation (RUN) or in standby mode (sleep, stop, watch).
  It also stops the supply of clock to the CPU until the clock supply stop signal of the oscillation stabilization wait time selector is released.

- **Oscillation stabilization wait time selector**
  The oscillation stabilization wait time selector selects one from the four types of main clock oscillation stabilization wait time made in the timebase timer and sub-clock oscillation stabilization wait time made in the watch prescaler in correspondence with clock mode, standby mode, and reset and outputs it as a clock supply stop signal to the CPU.

- **System clock control register (SYCC)**
  The system clock control register selects and checks the status of the clock mode, main clock speed, and main clock oscillation stabilization wait time.

- **Standby control register (STBC)**
  The standby control register changes the standard operation (RUN) to standby mode, sets the pin state in stop mode or watch mode, and performs software reset.
3.6.3 System Clock Control Register (SYCC)

The system clock control register (SYCC) switches between the main clock and the sub-clock, selects the main clock speed, and selects the main clock oscillation stabilization wait time.

System Clock Control Register (SYCC) Configuration

![System Clock Control Register (SYCC) Configuration](image)

- **SCM (System clock monitor bit)**: 0: Unused 1: System clock
- **SCS (System clock select bit)**: 0: Selects the sub-clock (32 kHz) mode 1: Selects the main clock mode
- **WT1 (Oscillation stabilization wait time selection bit)**: 0: Approx. 24.FCH (Approx. 0 ms) 1: Approx. 216.FCH (Approx. 0.5 ms)
- **WT0 (Main clock oscillation stabilization wait time by timebase timer output)**: 0: Approx. 216.FCH (Approx. 8.2 ms) 1: Approx. 218.FCH (Approx. 32.8 ms)
- **CS1 and CS0 (Main clock speed selection bit)**: 00: 64/FCH (8.0µs) 01: 16/FCH (2.0µs) 10: 8/FCH (1.0µs) 11: 4/FCH (0.5µs)

**FCH**: Main clock oscillation
### Table 3.6-1 Functions of Each Bit in the System Clock Control Register (SYCC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 7 SCM: System clock monitor bit | • A bit for checking the current clock mode (operating clock)  
• When this bit is set to "0," the system is operating in sub-clock mode. (Main clock is stopped or is in oscillation stabilization wait time to move to main clock mode.)  
• When this bit is set to "1," the system is operating in main clock mode.  
**Reference:**  
This bit is read only. The written value has no significance and does not affect operation. |
| Bit 6 Bit 5 Unused bit     | • The value in read cycle is undefined.  
• In write cycle, this bit does not affect operation.                                                                                                                                                   |
| Bit 4 Bit 3 WT1, WT0: Oscillation stabilization wait time selection bits | • Bits for selecting the main clock oscillation stabilization wait time.  
• When moving from sub-clock mode to main clock mode, when restoring from main stop mode to standard operation by an external interrupt, the oscillation stabilization wait time selected by these bits is used.  
• Because the initial values of these bits are selected by optional setting, when using oscillation stabilization wait time at the time of a reset operation, the oscillation stabilization wait time selected as an option is used.  
**Note:**  
Do not rewrite these bits when switching from the sub-clock to the main clock (SCS = 1 --> 0). Before rewriting, check the SCM bit to confirm that the chip is not in the main clock oscillation stabilization time. |
| Bit 2 SCS: System clock select bit | • A bit for specifying the clock mode  
• Writing "0" into this bit results in a transition from main clock mode to sub-clock mode.  
• Writing "1" into this bit results in a transition from sub-clock mode to main clock mode following the expiration of the oscillation stabilization time set in the WT1 and WT0 bits.  
**Note:**  
When the single clock option is selected, this bit has no significance. Always specify "1". |
| Bit 1 Bit 0 CS1, CS0: Main clock speed selection bit | Bits for selecting the clock speed in main clock mode.  
Four types of operating clock speed for the CPU and peripheral functions can be set (gear function). However, the operating clocks of the timebase timer and the watch prescaler are not affected by these bits. |
CHAPTER 3  CPU

Instruction Cycle ($t_{\text{inst}}$)

An instruction cycle (minimum instruction execution time) can be selected from divide-by-4, -8, -16, and -64 of the main clock and divide-by-2 of the sub-clock (32.768 kHz) by the system clock select bit (SCS) and main clock speed selection bits (CS1, CS0) of the SYCC register.

The instruction cycle at the highest speed in main clock mode (SYCC: SCS = 1, CS1, CS0 = 11B) is $4/F_{CH} = \text{approx. } 0.5 \mu s$ when the main clock oscillation ($F_{CH}$) is 8 MHz.

The instruction cycle in sub-clock mode (SCS = 0) is $2/F_{CL} = \text{approx. } 61.0 \mu s$ when the sub-clock oscillation ($F_{CL}$) is 32.768 kHz.
3.6.4 Clock Mode

There are two types of clock mode: main clock and sub-clock mode. In main clock mode, the main clock is the major operating clock. The speed of the main clock can be switched by selecting a speed from the four clocks by dividing its oscillation (gear function).

In sub-clock mode, the oscillation of the main clock is stopped and only the sub-clock serves as the operating clock.

Clock Mode Operating State

Table 3.6-2 Clock Mode Operating State

<table>
<thead>
<tr>
<th>Clock mode</th>
<th>Main clock speed SYCC registers (CS1, CS0)</th>
<th>Standby mode</th>
<th>Clock generation</th>
<th>Operating clocks</th>
<th>Standby mode release source (other than reset)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Main</td>
<td>Sub</td>
<td>CPU</td>
</tr>
<tr>
<td>(1.1)</td>
<td>Highest</td>
<td>RUN</td>
<td>Oscillation</td>
<td>F_Chi/4</td>
<td>F_Chi/2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sleep</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>(1.0)</td>
<td>(1.0)</td>
<td>RUN</td>
<td>Oscillation</td>
<td>F_Chi/8</td>
<td>F_Chi/2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sleep</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>(0.1)</td>
<td>(0.1)</td>
<td>RUN</td>
<td>Oscillation</td>
<td>F_Chi/16</td>
<td>F_Chi/2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sleep</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>(0.0)</td>
<td>(0.0)</td>
<td>RUN</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sleep</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Sub-clock mode</td>
<td>–</td>
<td>RUN</td>
<td>Oscillation</td>
<td>F_Chi</td>
<td>F_Chi</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sleep</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
</tbody>
</table>

F_Chi: Main clock oscillation
F_CL: Sub-clock oscillation
*: Because the timebase timer operates with the main clock, it stops operation in sub-clock mode.

In each clock mode, it is possible to move to standby mode provided for each clock mode. For standby mode, see Section 3.7 “Standby Mode.”
CHAPTER 3  CPU

- Gear Function (Main Clock Speed Switching Functions)
  
  By writing "00B" to "11B" into the main clock speed selection bits of the system clock control register (SYCC: CS1, CS0), four types of main clock speed can be selected.

  The CPU and peripheral circuits operate with the switched main clock. However, the timebase timer and the watch prescaler are not affected by the gear function.

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

- Operation in Main Clock Mode
  
  In ordinary operation in main clock mode (main RUN mode), both the main clock and sub-clock are oscillating. The watch prescaler operates with the sub-clock, while the CPU, timebase timer, and other peripheral circuits operate with the main clock.

  It is possible to change the main clock speed during operation in main clock mode except the timebase timer (gear function). By specifying standby mode, the mode changes to the main sleep mode or the main stop mode.

  Even if any type of reset occur, operation always start from the RUN mode. (Releasing each operation mode by a reset).

- Transition from main clock mode to sub-clock mode
  
  By writing "0" into the system clock select bit of the system clock control register (SYCC: SCS), transition from main clock mode to sub-clock mode occurs. The current operating clock can be checked by reading the system clock monitor bit of the system clock control register (SYCC: SCM).

  **Note:**

  To move to sub-clock mode immediately after turning on power, take longer than the sub-clock oscillation stabilization wait time generated by the watch prescaler before moving to sub-clock mode.
3.6 Clock

■ Operation in Sub-clock Mode

In ordinary operation in sub-clock mode (sub-RUN mode), the oscillation of the main clock is stopped and the system operates only with the sub-clock. By operating with the low-speed clock, power consumption can be saved.

All functions other than the timebase timer, operation is performed in the same manner as in main clock mode. By specifying standby mode during operation in sub-clock mode, the mode changes to sub-sleep mode, sub-stop mode, or watch mode.

○ Restoration from sub-clock mode to main clock mode

By writing "1" into the system clock select bits of the system clock control register (SYCC: SCS), the chip restores from sub-clock mode to main clock mode.

However, operation with the main clock starts after the expiration of the main clock oscillation stabilization wait time. The oscillation stabilization wait time can be selected from the four types by the oscillation stabilization wait time selection bits of the system clock control register (SYCC: WT1, WT0).

Note:

Do not rewrite the oscillation stabilization wait time selection bits (SYCC: WT1, WT0) during the switching from the sub-clock to the main clock (SYCC: SCS = 1). Also do not rewrite them during the main clock oscillation stabilization wait time. Before rewriting, check the system clock monitor bit to confirm that the operating clock has been switched to the main clock (SYCC: SCM=1).

When restoring from sub-clock mode to main clock mode by a reset, models with power-on reset take main clock oscillation stabilization wait time but models without power-on reset do not take oscillation stabilization wait time when a reset other than software reset and watchdog reset occurs.
3.6.5 Oscillation Stabilization Wait Time

When operating the main clock in main RUN mode from the main clock stop state such as at power on or in main stop or sub-clock mode, the main clock oscillation stabilization wait time must be used. Similarly, oscillation of the sub-clock is stopped in sub-clock mode and sub-clock oscillation stabilization wait time must be used.

Oscillation Stabilization Wait Time

A ceramic or crystal resonator requires a period of several milliseconds to several tens of seconds from the start of oscillation to the stabilization of oscillation in a unique frequency (oscillation frequency).

Therefore, it is necessary to inhibit CPU operation immediately after oscillation and to supply a clock to the CPU after oscillation is fully stabilized following expiration of the oscillation stabilization wait time.

Since the time required to stabilize oscillation differs depending on the type of resonator (crystal, ceramics) connected to the oscillator (clock generator), it is necessary to select an oscillation stabilization wait time that will allow the resonator to be used.

Figure 3.6-6 “Operation of the Oscillator Immediately after the Start of Oscillation” shows the operation of the oscillator immediately after the start of oscillation.

![Figure 3.6-6 Operation of the Oscillator Immediately after the Start of Oscillation](image)

Main Clock Oscillation Stabilization Wait Time

When starting operation in main clock mode from states in which oscillation of the main clock is stopped, the main clock oscillation stabilization wait time must be used.

The main clock oscillation stabilization wait time is a period from the start of counting in states in which the counter of the timebase timer is cleared to the overflow of a specified bit.

Oscillation stabilization wait time during operation

When restoring to main RUN mode from main stop mode by an eternal interrupt and when moving from sub-clock mode to main clock mode, one of the four types of oscillation stabilization wait time can be selected by the oscillation stabilization wait time selection bits of the system clock control register (SYCC: WT1, WT0).
3.6 Clock

- Oscillation stabilization wait time at the time of resetting

The oscillation stabilization wait time at the time of resetting (initial values of WT1, WT0) are selected in optional setting.

For models with power-on reset, oscillation stabilization wait time is used at the time of resets in sub-clock mode, power-on reset, and release of stop mode following an external reset.

For models without power-on reset, oscillation stabilization wait time is used only at the time of software and watchdog resets in sub-clock mode.

Table 3.6-3 "Main Clock Mode Operation Start Conditions and Oscillation Stabilization Wait Time" shows the relation between main clock mode operation start conditions and oscillation stabilization wait time.

Table 3.6-3 Main Clock Mode Operation Start Conditions and Oscillation Stabilization Wait Time

<table>
<thead>
<tr>
<th>Main clock mode operation start condition</th>
<th>At power on</th>
<th>In sub-clock mode</th>
<th>Release of main stop mode</th>
<th>Transition from sub-clock mode to main clock mode (SYCC: SCS'*1=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>External reset</td>
<td>Software reset and watchdog reset</td>
<td></td>
</tr>
<tr>
<td>Selection of oscillation stabilization wait time</td>
<td>Optional setting</td>
<td></td>
<td></td>
<td>SYCC: WT1, WT0*2</td>
</tr>
<tr>
<td>With power-on reset</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Without power-on reset</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>

Y: Oscillation stabilization wait time is used
N: Oscillation stabilization wait time is not used
*1: System clock select bit of system clock control register
*2: Oscillation stabilization wait time selection bits of the system clock control register

- Sub-clock Oscillation Stabilization Wait Time

When restoring to the sub-RUN mode from sub-stop mode (state in which oscillation of the sub-clock is stopped) by an external interrupt (When starting oscillation of the sub-clock), a certain amount of sub-clock oscillation stabilization wait time \(2^{15}/F_{CL}\) \(F_{CL}\): sub-clock oscillation) is used.

The sub-clock oscillation stabilization wait time is a period from the start of operation in states in which the watch prescaler is cleared when an overflow occurs.

To move to sub-clock mode after turning the power on, use more time than the sub-clock oscillation stabilization wait time in software because the sub-clock oscillation stabilization wait time is required when the power is turned on.
3.7 Standby Mode

Standby modes include sleep mode, stop mode, and watch mode. To move to standby mode, make a setting to the standby control register (STBC) in both main clock and sub-clock mode.

In main clock mode, it is possible to move to sleep and stop modes. In sub-clock mode, it is possible to move to sleep, stop, and watch modes.

By stopping the operation of the CPU and peripheral functions by setting to standby mode, power consumption can be reduced.

This section explains the relation between standby mode and clock mode and the operation state of each part in standby mode.

- **Standby Mode**
  
  In clock mode, power consumption is reduced by lowering the speed of the operating clock of the CPU and the peripheral circuits such as switching between the main clock and sub-clock and switching of the main clock speed (gear function). In standby mode, on the other hand, power consumption is reduced by stopping the supply of clock to the CPU with the clock controller (sleep mode), stopping the supply of clock to the CPU and peripheral circuits (watch mode), and stopping the oscillation itself (stop mode).

  - **Main sleep mode**
    
    In main sleep mode, the operation of the CPU and the watchdog timer is stopped. The peripheral functions, excluding the prescaler, operate with the main clock (some functions can operate with the sub-clock.)

  - **Sub-sleep mode**
    
    In sub-sleep mode, the oscillation of the main clock, the operation of the CPU, and the operation of the watchdog timer and timebase timer are stopped. The peripheral functions operate with the sub-clock.

  - **Main stop mode**
    
    In main stop mode, the operation of the CPU and the peripheral functions is stopped. Though the main clocks stops oscillation, the sub-clock continues oscillation. All functions excluding external interrupts, the count operation of the watch prescaler, and a part of functions operating with the sub-clock are stopped.

  - **Sub-stop mode**
    
    In sub-stop mode, all functions, including main clock and sub-clock stop oscillation, except external interrupts are stopped.

  - **Watch mode**
    
    The watch mode can be entered only from sub-clock mode. In this mode, all functions excluding watch prescaler (watch interrupt), external interrupts, and a part of functions operating with the sub-clock are stopped.

    Even in states in which the main clock is stopped such as main stop mode and watch mode, a part of the buzzer output can be operated with the sub-clock provided the sub-clock is
3.7 Standby Mode

oscillating. For details, see Chapter 12 "BUZZER OUTPUT" and "Clock Supply Map" in Section 3.6 "Clock."
This section explains the operating states of the CPU and the peripheral functions in standby mode.

### Operating States in Standby Mode

#### Table 3.7-1 Operating States of the CPU and the Peripheral Functions in Standby Mode

<table>
<thead>
<tr>
<th>Function</th>
<th>Operation mode</th>
<th>Main clock mode</th>
<th>Sub-clock mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>RUN</td>
<td>Sleep</td>
</tr>
<tr>
<td>Main clock</td>
<td>Operation</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Sub-clock</td>
<td>Operation</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td>Operation</td>
<td>Retained</td>
<td>Retained</td>
</tr>
<tr>
<td>CPU</td>
<td>Operation</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Instruction</td>
<td>Operation</td>
<td>Stop</td>
<td>Operation</td>
</tr>
<tr>
<td>ROM</td>
<td>Retained</td>
<td>Retained</td>
<td>Retained</td>
</tr>
<tr>
<td>RAM</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>Peripheral functions</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>I/O port</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Timebase timer</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>Stop</td>
<td>Stop</td>
<td>Operation</td>
</tr>
<tr>
<td>8-bit PWM timer</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>8/16-bit timer/counter</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>8-bit serial I/O</td>
<td>Stop</td>
<td>Stop</td>
<td>Inhibited</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>Stop</td>
<td>Stop</td>
<td>Inhibited</td>
</tr>
<tr>
<td>External interrupts 1, 2</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>A/D converter</td>
<td>Stop</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>Watch prescaler</td>
<td>Operation</td>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>DTMF generator</td>
<td>Stop</td>
<td>Stop</td>
<td>Inhibited</td>
</tr>
<tr>
<td>Serial I/O port switching</td>
<td>Retained</td>
<td>Retained</td>
<td>Retained</td>
</tr>
<tr>
<td>Modern timer</td>
<td>Operation</td>
<td>Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Modern signal output</td>
<td>Retained</td>
<td>Retained</td>
<td>Hi-z</td>
</tr>
<tr>
<td>Pin</td>
<td>Retained</td>
<td>Retained</td>
<td>Hi-z</td>
</tr>
<tr>
<td>Release method</td>
<td>Reset, various types of interrupts</td>
<td>Reset, various types of interrupts</td>
<td>Reset, various types of interrupts</td>
</tr>
</tbody>
</table>

*1: Though the watch prescaler performs counting, it generates no watch interrupt.

*2: It can be operated when the output of the watch prescaler or the frequency-divided output of the sub-clock are selected for the operating clock.

*3: Inhibited. Inhibit use with the program before moving to sub-clock mode.
Pin states in standby mode

The states of almost all I/O pins before moving to stop or watch mode can be retained or high-impedance can be set by the pin state specification bit of the standby control register (STBC: SPL) regardless of the clock mode.

For the states of the pins in standby mode, see Appendix E "Pin states of the MB89990 Series."
3.7.2 Sleep Mode

This section explains sleep mode operation.

- Sleep Mode Operation

  - Transition to sleep mode
    Sleep mode stops the operating clock of the CPU. Though the CPU stops while retaining the contents of the register and RAM immediately before the transition to sleep mode, peripheral functions excluding the watchdog timer continue to operate.

    In sub-clock mode, the oscillation of the main clock is stopped and the timebase timer does not operate because it uses the main clock oscillation frequency divided by 2 as its count clock.

    By writing "1" into the sleep bit of the standby control register (STBC: SLP), the chip enters sleep mode. If an interrupt request is generated when "1" is written into the SLP bit, the writing is ignored and execution of the instruction continues without moving to sleep mode. (Transition to sleep mode does not occur even after interrupt processing is completed.)

  - Releasing sleep mode
    Sleep mode is released by the input of a reset signal or an interrupt from an external function.

    For models with power-on reset, a reset operation is performed following the expiration of the oscillation stabilization wait time when a reset occurs in sub-sleep mode.

    For models without power-on reset, oscillation stabilization wait time is not used when a reset occurs in main sleep mode.

    The state of the pin is initialized by a reset.

    If an interrupt request having a higher priority of interrupt than "11B" is generated from a peripheral function or an external interrupt circuit while the chip is in sleep mode, the chip is released from sleep mode regardless of the values of the interrupt enable flag of the CPU (CCR: I) or the interrupt level bits (CCR: IL1, 0).

    When the chip is released from sleep mode, a standard interrupt operation is performed and interrupt processing is executed if the interrupt request can be accepted. If the interrupt request cannot be accepted, processing restarts with the instruction next to the instruction executed immediately before moving to sleep mode.
3.7.3 Stop Mode

This section explains stop mode operation.

■ Stop Mode Operation

- Transition to stop mode
  Stop mode stops the oscillation. Almost all functions stop while retaining the contents of the register and RAM immediately before moving to sleep mode.

  In main clock mode, the sub-clock continues oscillation though the oscillation of the main clock is stopped. Therefore, the counting operation of the watch prescaler and a part of functions operating with the sub-clock continue operation but other peripheral functions, excepting the external interrupt circuit, and the CPU stop operation.

  In sub-clock mode, both the main clock and sub-clock stop oscillation and all functions except the external interrupt circuit stop. Therefore, data can be retained with a minimal amount of power.

  By writing "1" into the stop bit of the standby control register (STBC: STP), the chip enters stop mode. If the pin state specification bit (STBC: SPL) is set to "0" at this time, the state of the external pin is retained. If it is set to "1," the state of the external pin is set to high-impedance. (The pins having pull-up resistors as options are set to "H" level.)

  If an interrupt request is generated when "1" is written into the STP bit, the writing is ignored and execution of the instruction continues without moving to stop mode. (Transition to stop mode does not occur even after completion of the interrupt processing.)

  To move to stop mode from main clock mode, inhibit the timebase timer interrupt request output (TBTC: TBIE = 0) as required. To move to stop mode from sub-clock mode, inhibit the watch interrupt request output of the watch prescaler (WPCR: WIE = 0) in the same manner.
CHAPTER 3 CPU

- Releasing stop mode

Stop mode is released by input of a reset signal or an interrupt from external functions.

For models with power-on reset, reset operation starts after the expiration of the main clock oscillation stabilization wait time when a reset occurs in stop mode.

For models without power-on reset, the oscillation stabilization wait time is not used when a reset occurs in stop mode. The state of the pin is initialized by a reset.

If an interrupt request having a higher priority of interrupt than "11B" is generated from a peripheral function while the chip is in stop mode, the chip is released from stop mode regardless of the values of the interrupt enable flag of the CPU (CCR: I) or the interrupt level bits (CCR: IL1, 0). Since the peripheral functions are stopped in stop mode, an interrupt request other than an external interrupt does not occur. In main stop mode, the watch prescaler is operating but no watch interrupt occurs.

After the chip is released from stop mode, ordinary interrupt operation is performed after the expiration of the oscillation stabilization wait time and interrupt processing is executed if the interrupt request can be accepted. If the interrupt request cannot be accepted, processing restarts with the instruction next to the instruction executed immediately before moving to stop mode.

When stop mode is released by an external interrupt, a part of peripheral functions resumes operation. Therefore, the initial interval time of the interval timer function becomes undefined. Initialize the peripheral functions after restoring from stop mode.

Reference:

Release of stop mode by an interrupt is performed only by an interrupt request from an external interrupt circuit.
3.7.4 Watch Mode

This section explains the watch mode operation.

## Watch Mode Operation

- **Transition to watch mode**
  
  The watch mode stops the operating clocks of the CPU and the major peripheral circuits. It can only be entered from sub-clock mode (main clock oscillation stop).
  
  While retaining the contents of the register and RAM immediately before moving to watch mode, all functions excluding the watch prescaler (watch interrupt), external interrupt circuit, and a part of functions operating with the sub-clock are stopped. Therefore, data can be retained with very low power consumption.
  
  By writing "1" into the watch bit of the standby control register (STBC: TMD) in states where sub-clock mode is set by the system clock select bit of the system clock control register (SYCC: SCS=0), the chip enters watch mode. If the pin state specification bit (STBC: SPL) of the standby control register is set to "0" when moving to watch mode, the state of the external pin is retained. If it is set to "1," the state of the external pin is set to high-impedance. (The pins having pull-up resistors as options are set to the "H" level.)
  
  If an interrupt request is generated when "1" is written into the TMD bit, the writing is ignored and the execution of the instruction continues without moving to watch mode. (Transition to watch mode does not occur even after interrupt processing is completed.)

- **Releasing watch mode**
  
  Watch mode is released by input of a reset signal or a watch interrupt and external interrupt.
  
  For models with power-on reset, reset operation starts after the expiration of the main clock oscillation stabilization wait time when a reset occurs in watch mode.
  
  For models without power-on reset, the oscillation stabilization wait time is not used when reset occurs in watch mode. The state of the pin is initialized by reset.
  
  If an interrupt request having a higher priority of interrupt than "11" is generated from the watch prescaler or a peripheral function while the chip is in watch mode, the chip is released from watch mode regardless of the values of the interrupt enable flag of the CPU (CCR: I) or the interrupt level bits (CCR: IL1, 0). Since all peripheral functions other than the watch prescaler are stopped in watch mode, an interrupt request other than a watch interrupt and an external interrupt does not occur.
  
  After the chip is released from watch mode, ordinary interrupt operation is performed and interrupt processing is executed if the interrupt request can be accepted. If the interrupt request cannot be accepted, processing restarts with the instruction next to the instruction executed immediately before moving to watch mode.
  
  When watch mode is released, a part of peripheral functions resumes operation. Therefore, the initial interval time of the interval timer function becomes undefined. Initialize the peripheral functions after restoring from watch mode.
### 3.7.5 Standby Control Register (STBC)

The standby control register (STBC) is used to move to sleep/stop/watch mode, set the pin state in stop and watch modes, and perform software reset.

#### Standby Control Register (STBC)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 8</td>
<td>STP</td>
<td>SLP</td>
<td>SPL</td>
<td>RST</td>
<td>TMD</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>00010XX000</td>
</tr>
</tbody>
</table>

#### Bit Descriptions

- **Watch bit (TMD)**
  - Valid only in sub-clock mode (SYCC: SCS = 0)
  - Read cycle: "0" is always read
  - Write cycle: Does not affect operation
  - Valid only in sub-clock mode (SYCC: SCS = 0)

- **Software reset bit (RST)**
  - Read cycle: "1" is always read
  - Write cycle: Reset signal of four instruction cycles is generated

- **Pin state specification bit (SPL)**
  - Read cycle: Retains the external pin to the state immediately entering stop mode
  - Write cycle: Sets the external pin to high impedance in stop mode

- **Sleep bit (SLP)**
  - Read cycle: "0" is always read
  - Write cycle: Moves to sleep bit

- **Stop bit (STP)**
  - Read cycle: "0" is always read
  - Write cycle: Moves to stop bit

---

**Legend**

- **R/W**: Read/write possible
- **W**: Write only
- **—**: Unused
- **X**: Undefined
- **Initial value**: 

---

**Figure 3.7-1 Standby Control Register (STBC)**

![Standby Control Register (STBC)](image-url)
### 3.7 Standby Mode

#### Table 3.7-2 Functions of Each Bit in the Standby Control Register (STBC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>STP:</strong> Stop bit</td>
</tr>
</tbody>
</table>
|                   | • A bit for specifying transition to stop mode  
|                   | • By writing "1" into this bit, the chip enters stop mode.  
|                   | • Writing "0" into this bit does not affect operation.  
|                   | • When this bit is read, "0" is always read.                                                                                                                                                            |
| Bit 6             | **SLP:** Sleep bit                                                                                                                                                                                        |
|                   | • A bit for specifying transition to sleep mode  
|                   | • By writing "1" into this bit, the chip enters sleep mode.  
|                   | • Writing "0" into this bit does not affect operation.  
|                   | • When this bit is read, "0" is always read.                                                                                                                                                            |
| Bit 5             | **SPL:** Pin state specification bit                                                                                                                                                                     |
|                   | • A bit for specifying the state of the external pin in stop mode and watch mode  
|                   | • When this bit is set to "0," the state (level) of the external pin is retained when moving to stop or watch mode  
|                   | • When this bit is set to "1," the state (level) of the external pin is set to high impedance when moving to stop or watch mode.  
|                   | • "H" level if "with pull-up resistor" is specified in optional setting  
|                   | • When reset, this bit is set to "0".                                                                                                                                                                     |
| Bit 4             | **RST:** Software reset bit                                                                                                                                                                              |
|                   | • A bit for specifying software rest.  
|                   | • By writing "1" into this bit, an internal reset source of four instruction cycles is generated.  
|                   | • Writing "0" into this bit does not affect operation.  
|                   | • When this bit is read, "1" is always read.                                                                                                                                                            |
| **Reference:**    | When software reset is performed in sub-clock mode, operation in main clock mode starts after the oscillation stabilization wait time is used. Therefore, when "reset output" is selected in optional setting, reset signal is output during the oscillation stabilization wait time. |
| Bit 3             | **TMD:** Watch bit                                                                                                                                                                                        |
|                   | • A bit for specifying transition to watch mode  
|                   | • Writing into this bit is valid only in sub-clock mode (SYCC: SCS = 0)  
|                   | • By writing "1" into this bit, the chip enters watch mode.  
|                   | • Writing "0" into this bit does not affect operation.  
|                   | • When this bit is read, "0" is always read.                                                                                                                                                            |
| Bit 2, Bit 1, Bit 0| Unused bit                                                                                                                                                                                              |
|                   | • The value in read cycle is undefined.  
|                   | • In write cycle, this bit does not affect operation.                                                                                                                                                  |
3.7.6 Status Transition Chart 1 (with Power-on Reset, Dual clock operation)

This section shows the status transition chart when a model with power-on reset is used in dual clock operation.

Figure 3.7-2 Status Transition Chart 1 (with Power-on Reset, Dual Clock Operation)
3.7 Standby Mode

○ Transition to and release of clock mode (non-standby mode)

Table 3.7-3 Transition and Release of Clock Mode (with Power-on Reset, Dual Clock Operation)

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition to main clock mode ordinary status</td>
<td>[1] Expiration of main clock oscillation stabilization wait time (timebase timer output)</td>
</tr>
<tr>
<td>(main RUN) after power-on reset</td>
<td>[2] Release of reset input</td>
</tr>
<tr>
<td>Reset in main RUN status</td>
<td>[3] External reset, software reset, watchdog reset</td>
</tr>
<tr>
<td>Transition from main RUN status to sub-RUN status</td>
<td>[4] SYCC: SCS=0*</td>
</tr>
<tr>
<td>Restoration from sub-RUN status to main RUN status</td>
<td>[5] SYCC: SCS=1</td>
</tr>
<tr>
<td></td>
<td>[6] Expiration of main clock oscillation stabilization wait time (It can be checked with SYCC: SCM)</td>
</tr>
<tr>
<td></td>
<td>[7] External reset, software reset, watchdog reset</td>
</tr>
<tr>
<td>Reset in sub-RUN status</td>
<td>[8] External reset, software reset, watchdog reset</td>
</tr>
</tbody>
</table>

SYCC: System clock control register
*: Transition to sub-RUN status directly after power on is after the expiration of the sub-clock oscillation stabilization wait time.

○ Transition to and release of standby mode

Table 3.7-4 Transition to and Release of Standby Mode (with Power-on Reset, Dual Clock Operation)

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition to sleep mode</td>
<td>[1] STBC: SLP = 1</td>
</tr>
<tr>
<td>Release of sleep mode</td>
<td>&lt;1&gt; STBC: SLP = 1</td>
</tr>
<tr>
<td></td>
<td>[2] Interrupts (various types)</td>
</tr>
<tr>
<td></td>
<td>[3] External reset</td>
</tr>
<tr>
<td></td>
<td>&lt;2&gt; Interrupts (various types)</td>
</tr>
<tr>
<td></td>
<td>&lt;3&gt; External reset</td>
</tr>
<tr>
<td>Transition to stop mode</td>
<td>[4] STBC: STP = 1</td>
</tr>
<tr>
<td>Release of stop mode</td>
<td>&lt;4&gt; STBC: STP = 1</td>
</tr>
<tr>
<td></td>
<td>[5] External interrupt</td>
</tr>
<tr>
<td></td>
<td>[6] Expiration of main clock oscillation stabilization wait time (timebase timer output)</td>
</tr>
<tr>
<td></td>
<td>[7] External reset</td>
</tr>
<tr>
<td></td>
<td>[8] External reset (waiting for oscillation stabilization)</td>
</tr>
<tr>
<td>Transition to watch mode</td>
<td>&lt;9&gt; STBC: TMD = 1*</td>
</tr>
<tr>
<td>Release of watch mode</td>
<td>&lt;10&gt; External interrupt or watch interrupt</td>
</tr>
<tr>
<td></td>
<td>&lt;11&gt; External reset</td>
</tr>
</tbody>
</table>

STBC: Standby control register
*: Transition to watch mode is possible only from sub-RUN status (SYCC: SCS = 0).

Reference:
Since the CPU and the watchdog timer is stopped in standby mode, software reset and watchdog reset do not occur.
3.7.7 Status Transition Chart 2 (without Power-on Reset, Dual clock operation)

This section shows the status transition chart when a model with no power-on reset is used in dual clock operation.

Figure 3.7-3 Status Transition Chart 2 (without Power-on Reset, Dual Clock Operation)
### Table 3.7-5 Transition and Release of Clock Mode (without Power-on Reset, Dual Clock Operation)

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
</tr>
</thead>
</table>
| Transition to main clock mode ordinary status (main RUN) after external reset | [1] Entry of the external reset signal must continue until oscillation of the main clock has stabilized.  
[2] Release of reset input |
| Reset in main RUN status                                | [3] External reset, software reset, watchdog reset                                                                                                    |
| Transition from main RUN status to sub-RUN status       | [4] SYCC: SCS=0*                                                                                                                                       |
| Restoration from sub-RUN status to main RUN status      | [5] SYCC: SCS=1  
[6] Expiration of main clock oscillation stabilization wait time (It can be checked with SYCC: SCM)  
[7] Software reset, watchdog reset  
[8] Entry of the external reset signal must continue until oscillation of the main clock has stabilized. |
| Reset in sub-RUN status                                 | [9] Software reset, watchdog reset  
[10] Entry of the external reset signal must continue until oscillation of the main clock has stabilized. |

SYCC: System clock control register

*: Transition to sub-RUN status directly after power on follows expiration of the sub-clock oscillation stabilization wait time.
### Transition to and release of standby mode

**Table 3.7-6 Transition to and Release of Clock Mode (without Power-on Reset, Dual Clock Operation)**

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
<th>Main clock mode</th>
<th>Sub-clock mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release of sleep mode</td>
<td>[2] Interrupts (various types)</td>
<td></td>
<td>[2] Interrupts (various types)</td>
</tr>
<tr>
<td>Transition to watch mode</td>
<td>—</td>
<td></td>
<td>[9] STBC: TMD = 1*</td>
</tr>
<tr>
<td>Release of watch mode</td>
<td>—</td>
<td></td>
<td>[10] External interrupt or watch interrupt</td>
</tr>
</tbody>
</table>

STBC: Standby control register

*: Transition to watch mode is possible only from sub-RUN status (SYCC: SCS = 0).

**Note:**

Except in main clock mode, standard operation status (RUN), and in sleep status, entry of an external reset must continue until oscillation of the main clock has stabilized.
3.7.8 Status Transition Chart 3 (Single Clock System)

This section shows the status transition charts when a model with power-on reset and a model with no power-on reset are used in a single clock system. When a single clock system is used, there is no sub-clock mode and watch mode status.

- Status Transition Chart 3 (Single Clock System)

![Figure 3.7-4 Status Transition Chart 3 (Model with Power-on Reset)]
Transition to and release of ordinary status (RUN)

Table 3.7-7 Transition to and Release of Main Clock Mode RUN Status (Single Clock System)

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
<th>Model with power-on reset (Figure 3.7-7)</th>
<th>Model with no power-on reset (Figure 3.7-8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition to ordinary status (RUN)</td>
<td>[1] Expiration of main clock oscillation stabilization wait time (timebase timer output)</td>
<td>[1] Entry of an external reset signal must continue until oscillation of the main clock has stabilized.</td>
<td>[2] Release of reset input</td>
</tr>
<tr>
<td></td>
<td>[2] Release of reset input</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.7-8 Transition to and Release of Standby Mode (Single Clock System)

<table>
<thead>
<tr>
<th>Status transition</th>
<th>Transition conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model with power-on reset (Figure 3.7-7)</td>
</tr>
</tbody>
</table>

**Note:**

When releasing the stop mode of a model with no power-on reset, entry of an external reset must continue until oscillation of the main clock has stabilized.

STBC: Standby control register
# States of Pins in Standby Mode

Table 3.7-9 "States of Pins in Standby Mode" shows the states of pins in standby mode

## States of Pins in Standby Mode

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Sleep mode</th>
<th>Stop mode (SPL=0)</th>
<th>Stop mode (SPL=1)</th>
<th>Watch mode (SPL=0)</th>
<th>Watch mode (SPL=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Main sleep</td>
<td>Sub-sleep</td>
<td>Main stop</td>
<td>Sub-stop</td>
<td>Main stop</td>
</tr>
<tr>
<td>X0</td>
<td>Oscillation input</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
</tr>
<tr>
<td>X1</td>
<td>Oscillation output</td>
<td>&quot;H&quot; output</td>
<td>&quot;H&quot; output</td>
<td>&quot;H&quot; output</td>
<td>&quot;H&quot; output</td>
</tr>
<tr>
<td>X0A</td>
<td>Oscillation input</td>
<td>Oscillation input</td>
<td>Hi-z</td>
<td>Oscillation input</td>
<td>Hi-z</td>
</tr>
<tr>
<td>X1A</td>
<td>Oscillation output</td>
<td>Oscillation output</td>
<td>&quot;H&quot; output</td>
<td>Oscillation output</td>
<td>&quot;H&quot; output</td>
</tr>
<tr>
<td>MOD0/MOD1</td>
<td>Mode input</td>
<td>Mode input</td>
<td>Mode input</td>
<td>Mode input</td>
<td>Mode input</td>
</tr>
<tr>
<td>RST</td>
<td>Reset input</td>
<td>Reset input</td>
<td>Reset input</td>
<td>Reset input</td>
<td>Reset input</td>
</tr>
<tr>
<td>P00 to P07</td>
<td>Retained</td>
<td>Port I/O</td>
<td></td>
<td>Hi-z</td>
<td>Port I/O</td>
</tr>
<tr>
<td>P10 to P17</td>
<td>Port I/O</td>
<td>Port I/O</td>
<td></td>
<td>Hi-z</td>
<td>Port I/O</td>
</tr>
<tr>
<td>P20 to P27</td>
<td>Port output</td>
<td></td>
<td></td>
<td></td>
<td>Port input</td>
</tr>
<tr>
<td>P30/PWM to P37/SO1</td>
<td>Port I/O/resource I/O</td>
<td>Port I/O/resource I/O</td>
<td></td>
<td>Hi-z</td>
<td>Port I/O/resource I/O</td>
</tr>
<tr>
<td>P40 to P44</td>
<td>Port output</td>
<td>Retained</td>
<td></td>
<td>Hi-z</td>
<td>Retained</td>
</tr>
<tr>
<td>P50/AN00 to P57/AN07</td>
<td>Port output/analog input</td>
<td>Port output/analog input</td>
<td></td>
<td>Hi-z</td>
<td>Port output/analog input</td>
</tr>
<tr>
<td>P60/TMO1 to P67/BSO1</td>
<td>Port I/O/resource I/O</td>
<td>Port I/O/resource I/O</td>
<td></td>
<td>Hi-z</td>
<td>Port I/O/resource I/O</td>
</tr>
<tr>
<td>P70/SK2 to P75/BSO2</td>
<td>Port I/O</td>
<td>Port I/O</td>
<td></td>
<td></td>
<td>Port I/O</td>
</tr>
<tr>
<td>P76, P77</td>
<td>Port I/O</td>
<td>Port I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P80 to P87</td>
<td>Port output</td>
<td>Retained</td>
<td>Hi-z</td>
<td>Retained</td>
<td>Hi-z</td>
</tr>
</tbody>
</table>
Table 3.7-9 States of Pins in Standby Mode (Continued)

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Sleep mode</th>
<th>Stop mode (SPL=0)</th>
<th>Stop mode (SPL=1)</th>
<th>Watch mode (SPL=0)</th>
<th>Watch mode (SPL=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Main sleep</td>
<td>Sub-sleep</td>
<td>Main stop</td>
<td>Sub-stop</td>
<td>Main stop</td>
</tr>
<tr>
<td>P90/INT20 to P97/INT27</td>
<td>Port I/O/ resource I/O</td>
<td>Port I/O/ resource I/O</td>
<td></td>
<td>Hi-z*</td>
<td>Port I/O/ resource I/O</td>
</tr>
<tr>
<td>PA0/INT28 to PA7/INT3</td>
<td>Port I/O/ resource I/O</td>
<td>Port I/O/ resource I/O</td>
<td></td>
<td>Hi-z*</td>
<td>Port I/O/ resource I/O</td>
</tr>
</tbody>
</table>

*: To prevent leakage by open input, the input level is fixed.
Hi-z: High-impedance.
SPL: Pin state specification bit of standby control register (STBC)
Retained: The pin set for output retains the state of the pin immediately before mode transition.

For pin states in other than standby mode, see Appendix E "Pin States of MB89890 Series."
3.7.10 Precautions on Use of Standby Mode

Even if standby mode is set in the standby control register (STBC), transition to standby mode does not occur if an interrupt request is generated from a peripheral function. When restoring from standby mode to standard operation mode by an interrupt, operation after restoration differs depending on whether the interrupt request is accepted.

- Transition to Standby Mode and Interrupt

  If an interrupt request having a higher priority of interrupt than "11B" is generated from a peripheral function to the CPU, writing to "1" into the stop bit (STBC: STP), sleep bit (SLP), and watch bit (TMD) of the standby control register respectively is ignored and transition to standby mode does not occur. (Transition to standby mode does not occur even after interrupt processing is completed.)

  This is not relevant to the CPU accepting the interrupt.

  Even if the CPU is processing an interrupt, transition to standby mode is possible if its interrupt request flag is cleared and no other interrupt request exists.

- Release of Standby Mode by an Interrupt

  If an interrupt request having a higher priority of interrupt than "11B" is generated while the chip is in sleep or stop mode, standby mode is released. This is not relevant to the CPU accepting the interrupt.

  After release, if the priority of the interrupt level setting registers (ILR1 to ILR3) corresponding to the interrupt request is higher than the interrupt level bit of the condition registers (CCR: IL1, 0) and if the interrupt enable flag is enabled (CCR: I = 1) as a standard interrupt operation, the CPU branches to the interrupt processing routine. If the interrupt is not accepted, processing restarts with the instruction next to the instruction starting the standby mode.

  If the CPU is not to be branched to the interrupt processing routine immediately after restoration, countermeasures, such as inhibiting interrupt before setting standby mode, are required.

- Precautions on Standby Mode Setting

  To set standby mode with the standby control register (STBC), see Table 3.7-10 "Setting of Power Save Mode with the Standby Control Register (STBC)". Though the highest priority is given to stop mode, set only one bit to "1", followed by watch mode when "1" is written into all of the bits.

  Do not move to stop mode, sleep mode, and watch mode immediately after switching from sub-clock mode to main clock mode (SYCC: SCS=0 --> 1). Before moving to these modes, check that the clock monitor bit of the system control register (SYCC: SCM) has been changed to "1".

  During operation in main clock mode, writing into the watch bit (TMD) is ignored.
3.7 Standby Mode

Table 3.7-10 Setting of Power Save Mode with the Standby Control Register (STBC)

<table>
<thead>
<tr>
<th>STBC register</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>STP (bit7)</td>
<td>SLP (bit6)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>

Oscillation Stabilization Wait Time

In both main clock mode and sub-clock mode, the oscillator oscillation is stopped in stop mode. Therefore, it is necessary to take oscillation stabilization wait time after these oscillators start operation.

In main clock mode, the main clock oscillation stabilization wait time (select one from four types) created in the timebase timer is used. In sub-clock mode, the sub-clock oscillation stabilization wait time created in the watch prescaler is used.

In main clock mode, an interval timer interrupt request occurs during the oscillation stabilization wait time if the interval timer of the timebase timer selected is shorter than the oscillation stabilization wait time. To move to stop mode from main clock mode, inhibit the timebase timer interrupt request output (TBTC: TBIE = 0) as required.

In the same manner, a watch interrupt request occurs by the selection of the interrupt interval time of the watch prescaler. To move to stop mode from sub-clock mode, inhibit the watch interrupt request output of the watch prescaler (WPCR: WIE = 0) as required.
3.8 Memory Access Mode

The operation mode related to memory access of the MB89890 Series is only in single chip mode.

■ Single Chip Mode

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

■ Mode Pins (MOD1, 0)

Be sure to set the mode pins to "VSS, VSS."

At the time of reset, mode data and reset vector are read from the internal ROM.

Do not change the value of the mode pins even after completion of the reset operation (during reset operation).

Table 3.8-1 "Mode Pin Settings" shows the mode pin settings.

Table 3.8-1 Mode Pin Settings

<table>
<thead>
<tr>
<th>Pin states</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD1 MOD0</td>
<td></td>
</tr>
<tr>
<td>VSS VSS</td>
<td>Reads mode data and reset vector from the internal ROM.</td>
</tr>
<tr>
<td>VSS VCC</td>
<td>Inhibited</td>
</tr>
<tr>
<td>VCC VSS</td>
<td></td>
</tr>
<tr>
<td>VCC VCC</td>
<td></td>
</tr>
</tbody>
</table>

■ Mode Data

Be sure to set the internal ROM mode data to "00H" to select single chip mode.

Figure 3.8-1 Mode Data Configuration

<table>
<thead>
<tr>
<th>Address</th>
<th>bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF FF FF DH</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>Select single chip mode</td>
</tr>
<tr>
<td>Other than 00H</td>
<td>Reserved. Do not set.</td>
</tr>
</tbody>
</table>
### Memory Access Mode Selective Operation

Only a selective single chip mode operation can be performed.

Table 3.8-2 "Mode Pins and Mode Data" shows mode pins and mode data.

**Table 3.8-2 Mode Pins and Mode Data**

<table>
<thead>
<tr>
<th>Memory access mode</th>
<th>Mode pins (MOD1, 0)</th>
<th>Mode data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single chip mode</td>
<td>V&lt;sub&gt;SS&lt;/sub&gt;, V&lt;sub&gt;SS&lt;/sub&gt;</td>
<td>00&lt;sub&gt;H&lt;/sub&gt;</td>
</tr>
<tr>
<td>Other modes</td>
<td>Setting inhibited</td>
<td>Setting inhibited</td>
</tr>
</tbody>
</table>

Figure 3.8-2 "Memory Access Selective Operation" shows the flow of memory access selective operation.

**Figure 3.8-2 Memory Access Selective Operation**

- Check the mode pins
- Wait for release of reset source (external reset or oscillation stabilization wait time)
- Mode fetch
- Check mode data
- Set I/O pin function in program execution (RUN) status

- Reset source occurrence
- Mode pins (MOD1, 0)
- Single chip mode
- Mode data is read from internal ROM
- I/O pin High impedance
- Resetting
- Yes
- Fetch mode data and reset vector from internal ROM
- No
- Setting inhibited
- Other
- Mode data
- Single chip mode (00<sub>H</sub>)
- Setting inhibited
- Other
- Mode data
- Single chip mode
- Set I/O of each I/O pin with port direction register (DDR)
- I/O pins can be used as ports.
This chapter describes the functions and operations of the I/O ports.

4.1 "Overview of the I/O Ports"
4.2 "Port 0, 1"
4.3 "Port 2"
4.4 "Port 3"
4.5 "Port 4, 8"
4.6 "Port 5"
4.7 "Port 6"
4.8 "Port 7"
4.9 "Port 9, A"
4.10 "Program Example of I/O Ports"
4.1 Overview of the I/O Ports

Eleven ports (85 pins) can be used as an output-only port or general-purpose I/O port (parallel I/O port).
Each port also functions as resources (I/O pins for various peripherals).

Functions of the I/O Ports

The I/O ports have functions to output data from the CPU to the I/O pins using the port data register (PDR) or send signals entered in the I/O pins to the CPU. Also, depending on the port, the direction of input-output of the I/O pins can be set optionally for each bit using the port direction register (DDR).

The following lists the functions of each port and shared resources:
- Port 0: General-purpose I/O port
- Port 1: General-purpose I/O port
- Port 2: Output-specific port
- Port 3: General-purpose I/O port/resource (PWM, buzzer, modem, serial I/O-1 pin) shared
- Port 4: Output-specific port
- Port 5: General-purpose output port/resource (analog input pin) shared
- Port 6: General-purpose I/O port/resource (timer/counter, modem, BSIO-1 pin) shared
- Port 7: General-purpose I/O port/resource (serial I/O-2, BSIO-2 pin) shared
- Port 8: Output-specific port
- Port 9: General-purpose I/O port/resource (external interrupt-2 input pin) shared
- Port A: General-purpose I/O port/resource (external interrupt-1/2 input pin) shared

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

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Input type</th>
<th>Output type</th>
<th>Function</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>Port 0</td>
<td>P00 to P07</td>
<td>CMOS</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P07</td>
<td>P06</td>
<td>P05</td>
<td>P04</td>
<td>P03</td>
<td>P02</td>
<td>P01</td>
<td>P00</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>
</tr>
<tr>
<td>Port 1</td>
<td>P10 to P17</td>
<td>CMOS</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P17</td>
<td>P16</td>
<td>P15</td>
<td>P14</td>
<td>P13</td>
<td>P12</td>
<td>P11</td>
<td>P10</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>
</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>
</tr>
<tr>
<td>Port 3</td>
<td>P30/PWM to P37/SO1</td>
<td>CMOS (hysteresis)</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P37</td>
<td>P36</td>
<td>P35</td>
<td>P34</td>
<td>P33</td>
<td>P32</td>
<td>P31</td>
<td>P30</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>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resource</td>
<td>SO1</td>
<td>SI1</td>
<td>SK1</td>
<td>-</td>
<td>-</td>
<td>MSK1</td>
<td>BZ</td>
<td>PWM</td>
</tr>
<tr>
<td>Port 4</td>
<td>P40 to P44</td>
<td>-</td>
<td>CMOS push-pull</td>
<td>Output-specific port</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>P44</td>
<td>P43</td>
<td>P42</td>
<td>P41</td>
<td>P40</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>
</tr>
<tr>
<td>Port 5</td>
<td>P50/AN0 to P57/AN07</td>
<td>Analog channel select</td>
<td>CMOS push-pull</td>
<td>General-purpose output port</td>
<td>P57</td>
<td>P56</td>
<td>P55</td>
<td>P54</td>
<td>P53</td>
<td>P52</td>
<td>P51</td>
<td>P50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Analog input</td>
<td>AN07</td>
<td>AN06</td>
<td>AN05</td>
<td>AN04</td>
<td>AN03</td>
<td>AN02</td>
<td>AN01</td>
<td>AN00</td>
</tr>
<tr>
<td>Port 6</td>
<td>P60/TMO1 to P67/BSO1</td>
<td>CMOS (hysteresis)</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P67</td>
<td>P66</td>
<td>P65</td>
<td>P64</td>
<td>P63</td>
<td>P62</td>
<td>P61</td>
<td>P60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resource</td>
<td>BSO1</td>
<td>BSI1</td>
<td>BSK1</td>
<td>-</td>
<td>-</td>
<td>MSKO</td>
<td>TCLK</td>
<td>TMO2</td>
</tr>
<tr>
<td>Port 7</td>
<td>P70/SK2 to P77</td>
<td>CMOS (hysteresis)</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P77</td>
<td>P76</td>
<td>P75</td>
<td>P74</td>
<td>P73</td>
<td>P72</td>
<td>P71</td>
<td>P70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resource</td>
<td>-</td>
<td>-</td>
<td>BSO2</td>
<td>BSI2</td>
<td>BSK2</td>
<td>SO2</td>
<td>SI2</td>
<td>SK2</td>
</tr>
<tr>
<td>Port 8</td>
<td>P80 to P87</td>
<td>-</td>
<td>CMOS push-pull</td>
<td>Output-specific port</td>
<td>P87</td>
<td>P86</td>
<td>P85</td>
<td>P84</td>
<td>P83</td>
<td>P82</td>
<td>P81</td>
<td>P80</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>
</tr>
</tbody>
</table>
### CHAPTER 4 I/O PORTS

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Input type</th>
<th>Output type</th>
<th>Function</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>Port 9</td>
<td>P90/INT20 to P97/INT27</td>
<td>CMOS (hysteresis)</td>
<td>CMOS push-pull</td>
<td>General-purpose I/O port</td>
<td>P97</td>
<td>P66</td>
<td>P95</td>
<td>P94</td>
<td>P93</td>
<td>P92</td>
<td>P91</td>
<td>P90</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>External interrupt 2</td>
<td>INT27</td>
<td>INT26</td>
<td>INT25</td>
<td>INT24</td>
<td>INT23</td>
<td>INT22</td>
<td>INT21</td>
<td>INT20</td>
</tr>
<tr>
<td>Port A</td>
<td>PA0/INT28 to PA7/INT3</td>
<td></td>
<td></td>
<td>General-purpose I/O port</td>
<td>PA7</td>
<td>PA6</td>
<td>PA5</td>
<td>PA4</td>
<td>PA3</td>
<td>PA2</td>
<td>PA1</td>
<td>PA0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>External interrupt 1</td>
<td>INT3</td>
<td>INT2</td>
<td>INT1</td>
<td>INT0</td>
<td>INT9</td>
<td>INT8</td>
<td>INT7</td>
<td>INT6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT5</td>
<td>INT4</td>
<td>INT3</td>
<td>INT2</td>
<td>INT1</td>
<td>INT0</td>
<td>INT9</td>
<td>INT8</td>
</tr>
</tbody>
</table>

- Unused

---

### Table 4.1-2 Registers of Each Port

<table>
<thead>
<tr>
<th>Register name</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0 data register (PDR0)</td>
<td>R/W</td>
<td>0000H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 0 direction register (DDR0)</td>
<td>W*</td>
<td>0001H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>R/W</td>
<td>0002H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 1 direction register (DDR1)</td>
<td>W*</td>
<td>0003H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 2 data register (PDR2)</td>
<td>R/W</td>
<td>0004H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>R/W</td>
<td>000CH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 3 direction register (DDR3)</td>
<td>R/W</td>
<td>000DH</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>R/W</td>
<td>000EH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 5 data register (PDR5)</td>
<td>R/W</td>
<td>0010H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>R/W</td>
<td>0012H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 6 direction register (DDR6)</td>
<td>R/W</td>
<td>0013H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port 7 data register (PDR7)</td>
<td>R/W</td>
<td>0014H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 8 data register (PDR8)</td>
<td>R/W</td>
<td>0016H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 9 data register (PDR9)</td>
<td>R/W</td>
<td>0018H</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port 9 direction register (DDR9)</td>
<td>R/W</td>
<td>0019H</td>
<td>00000000B</td>
</tr>
<tr>
<td>Port A data register (PDRA)</td>
<td>R/W</td>
<td>001AH</td>
<td>XXXXXXXXXB</td>
</tr>
<tr>
<td>Port A direction register (DDRA)</td>
<td>R/W</td>
<td>001BH</td>
<td>00000000B</td>
</tr>
</tbody>
</table>

*: Bit processing commands cannot be used for DDR0/1.
R/W: Read/Write enabled
R: Read-only
W: Write-only
X: Undefined
4.2 Port 0, 1

Port 0 and Port 1 are general-purpose I/O ports. This section describes the configuration, pins, a pin block diagram, and related registers of Ports 0 and Port 1.

- Configuration of Port 0, 1
  Port 0 and Port 1 each consist of the following three parts:

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

- **Port 1**
  - General-purpose I/O pins (P10 to P17)
  - Port 1 data register (PDR1)
  - Port 1 direction register (DDR1)
CHAPTER 4 I/O PORTS

Pins of Port 0, 1

Port 0 and Port 1 have eight I/O pins for CMOS input/CMOS output, respectively. Table 4.2-1 "Pins of Port 0, 1" lists the pins of Port 0, 1.

Table 4.2-1 Pins of Port 0, 1

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0</td>
<td>P00</td>
<td>P00 general-purpose I/O</td>
<td>—</td>
<td>CMOS</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>P01</td>
<td>P01 general-purpose I/O</td>
<td></td>
<td>CMOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P02</td>
<td>P02 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P03</td>
<td>P03 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P04</td>
<td>P04 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P05</td>
<td>P05 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P06</td>
<td>P06 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P07</td>
<td>P07 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1</td>
<td>P10</td>
<td>P10 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P11</td>
<td>P11 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P12</td>
<td>P12 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P13</td>
<td>P13 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P14</td>
<td>P14 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P15</td>
<td>P15 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P16</td>
<td>P16 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P17</td>
<td>P17 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the circuit types, see Section 1.7 "Pin Functions."
4.2 Port 0, 1

**Block Diagram of Port 0, 1**

**Figure 4.2-1 Block Diagram of Pins of Port 0, 1**

![Block Diagram of Port 0, 1](image)

SPL: Pin status designation bit of the standby control register (STBC)

**Registers of Port 0, 1**

There are four registers that are related to Port 0, 1; PDR0, DDR0, PDR1, and DDR1. Bits configuring each register are in a one-to-one correspondence with the pins of each port. Table 4.2-2 "Correspondence between the Registers and Pins of Port 0, 1" lists the correspondence between the registers and the pins of Port 0, 1.

**Table 4.2-2 Correspondence between the Registers and Pins of Port 0, 1**

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related registers and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0</td>
<td>PDR0, DDR0</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin</td>
</tr>
<tr>
<td>Port 1</td>
<td>PDR1, DDR1</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin</td>
</tr>
</tbody>
</table>
4.2.1 Registers of Port 0, 1 (PDR0, DDR0, PDR1, and DDR1)

The following explains the registers related to Port 0, 1.

### Functions of the Registers of Port 0, 1

- **Port 0, 1 data register (PDR0, PDR1)**
  
The PDR0 and PDR1 registers indicate the status of each pin. Thus, a pin set as an output port can read the same value ("0" or "1") as that of the output latch, but the pin cannot read the output latch value when the port is set for input and reads the pin status instead.

  **Reference:**
  
  When a bit processing command (SETB, CLRb) is applied, the value of the output latch is read instead of the pin. Thus, the values of the output latch other than those of bits to be manipulated do not change.

- **Port 0, 1 direction register (DDR0, DDR1)**
  
The DDR0 and DDR1 registers set the input-output direction of pins for each bit.

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

  **Note:**
  
  Because the DDR0 and DDR1 registers are write-only, bit processing commands (SETB, CLRb) cannot be used.

  Table 4.2-3 "Functions of Registers of Port 0, 1" lists the functions of registers of Port 0, 1.

### Table 4.2-3 Functions of Registers of Port 0, 1

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0 data register (PDR0)</td>
<td>0</td>
<td>Pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>0000H</td>
<td>XXXXXXXX&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 0 direction register (DDR0)</td>
<td>0</td>
<td>Cannot read (write-only)</td>
<td>Enables output transistor operations to set the pin for output.</td>
<td>W</td>
<td>0001&lt;sub&gt;H&lt;/sub&gt;</td>
<td>00000000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>Disables output transistor operations to set the pin for input.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 4.2-3 Functions of Registers of Port 0, 1 (Continued)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>0</td>
<td>Pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>0002&lt;sub&gt;H&lt;/sub&gt;</td>
<td>XXXXXXXXX&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1 direction register (DDR1)</td>
<td>0</td>
<td>Cannot read (write-only)</td>
<td>Disables output transistor operations to set the pin for input.</td>
<td>W</td>
<td>0003&lt;sub&gt;H&lt;/sub&gt;</td>
<td>00000000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>Enables output transistor operations to set the pin for output.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled  
W: Write-only  
X: Undefined
4.2.2  **Explanation of Operations of Port 0, 1**

The following explains the operations of Port 0, 1.

### Operations of Port 0, 1

- **Operations when the port is set for output**
  - If the corresponding bit of the DDR0 or DDR1 register is set to "1", the port functions as an output port.
  - When the port is set for output, operations of the output transistor are enabled and data in the output latch is output to the pin.
  - If data is written to the PDR0 or PDR1 register, the data is retained in the output latch and then output to the pin directly.
  - If the PDR0 or PDR1 register is read, the value of the pin is read.

- **Operations when the port is set for input**
  - If the corresponding bit of the DDR0 or DDR1 register is set to "0", the port functions as an input port.
  - When the port is set for input, the output transistor is turned off and the pin is placed in high impedance state.
  - If data is written to the PDR0 or PDR1 register, it is retained in the output latch but not output to the pin.
  - If the PDR0 or PDR1 register is read, the value of the pin is read.

- **Operations after reset**
  - If the CPU is reset, the values of the DDR0 and DDR1 registers are initialized to "0". Thus, the output transistors are turned off (input port) and the pins are placed in high impedance state.
  - The PDR0 and PDR1 registers are not initialized by a reset. Thus, to use the port as an output port, it is necessary to set output data to the PDR0 or PDR1 register and then set the corresponding DDR0 or DDR1 register for output.

- **Operations in stop mode or watch mode**
  If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the pin is placed in high impedance state because the output transistor is forced to off regardless of the value of the DDR0 or DDR1 register. On the other hand, the input is fixed to prevent leakage due to input release.
Table 4.2-4 "Pin Status of Port 0, 1" lists the pin status of Port 0, 1.

Table 4.2-4 Pin Status of Port 0, 1

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=0)</th>
<th>Sub-sleep</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>Main stop (SPL=1)</th>
<th>Sub-stop (SPL=1)</th>
<th>Watch mode (SPL=1)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00 to P07</td>
<td>General-purpose I/O port</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>After reset</td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance

Reference:
If "Pull-up resistor present" is selected in the option settings, the pin status when the output buffer becomes off is at "H" level (pull-up state) instead of high impedance.
4.3 Port 2

Port 2 is an output-only port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 2.

### Configuration of Port 2

Port 2 consists of the following two parts:
- Output-only pins (P20 to P27)
- Port 2 data register (PDR2)

### Pins of Port 2

Port 2 has eight output-only pins for CMOS output. Table 4.3-1 "Pins of Port 2" lists the pins of Port 2.

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output 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>Port 2</td>
<td>P20</td>
<td>P20 output-only</td>
<td>-</td>
<td></td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>P21</td>
<td>P21 output-only</td>
<td>-</td>
<td></td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>P22</td>
<td>P22 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P23</td>
<td>P23 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P24</td>
<td>P24 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P25</td>
<td>P25 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P26</td>
<td>P26 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P27</td>
<td>P27 output-only</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For circuit types, see Section 1.7 "Pin Functions."
4.3 Port 2

### Block Diagram of Port 2

#### Figure 4.3-1 Block Diagram of Pins of Port 2

![Block Diagram of Port 2](image)

SPL: Pin status designation bit of the standby control register (STBC)

### Register of Port 2

There is one register that is related to Port 2: PDR2. Bits configuring the PDR2 register are in a one-to-one correspondence with the pins of Port 2. Table 4.3-2 "Correspondence between the Register and Pins of Port 2" lists the correspondence between the register and pins of Port 2.

#### Table 4.3-2 Correspondence between the Register and Pins of Port 2

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related register and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2</td>
<td>PDR2: 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>
CHAPTER 4 I/O PORTS

4.3.1 Register of Port 2 (PDR2)

The following explains the register related to Port 2.

■ Functions of the Register of Port 2

○ Port 2 data register (PDR2)

The PDR2 register indicates the status of the output latch, so that the pin status cannot be read. Table 4.3-3 “Functions of the Register of Port 2” lists the functions of the register of Port 2.

Table 4.3-3 Functions of the Register of Port 2

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2 data register (PDR2)</td>
<td>0</td>
<td>The output latch value is &quot;0&quot;.</td>
<td>Outputs the &quot;L&quot; level to the pin (Sets the output latch to &quot;0&quot;).</td>
<td>R/W</td>
<td>0004_B</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The output latch value is &quot;1&quot;.</td>
<td>Outputs the &quot;H&quot; level to the pin (Sets the output latch to &quot;1&quot;).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
W: Write-only
X: Undefined
4.3.2 Explanation of Operations of Port 2

The following explains the operations of Port 2.

- **Operations of Port 2**

  - **Operations when the port is set for output**
    If data is written to the PDR2 register, it is retained in the output latch and then output to the pin through the output buffer.

  - **Operations after reset**
    If the CPU is reset, the value of the PDR2 register is initialized to "0" and the pin is placed in "L" level output.

  - **Operations in stop mode or watch mode**
    If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the output buffer is forced to off and the pin is placed in high impedance state. On the other hand, the input is fixed to prevent leakage due to input release.

    Table 4.3-4 "Pin Status of Port 2" lists the pin status of Port 2.

**Table 4.3-4 Pin Status of Port 2**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=0)</th>
<th>Sub-sleep</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>Main stop (SPL=1)</th>
<th>Sub-stop (SPL=1)</th>
<th>Watch mode (SPL=1)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P20 to P27</td>
<td>Output-only port</td>
<td>Hi-z</td>
<td></td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance
4.4 Port 3

Port 3 is a general-purpose I/O port sharing resource I/O. Each pin can be used by switching the resource and port for each bit. The explanation in this section focuses on the functions as a general-purpose I/O port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 3.

### Configuration of Port 3

Port 3 consists of the following three parts:
- General-purpose I/O pins/resource I/O pins (P30/PWM to P37/SO1)
- Port 3 data register (PDR3)
- Port 3 direction register (DDR3)

### Pins of Port 3

Port 3 has eight I/O pins for hysteresis input/CMOS output. Among the pins, those pins that also function for resource I/O cannot be used as a general-purpose I/O port when a resource is used.

Table 4.4-1 "Pins of Port 3" lists the pins of Port 3.

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P30/PWM</td>
<td>P30 general-purpose I/O</td>
<td>PWM: 8-bit PWM timer, PWM output</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P31/BZ</td>
<td>P31 general-purpose I/O</td>
<td>BZ: buzzer output</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P32/MSK1</td>
<td>P32 general-purpose I/O</td>
<td>MSKI: modem timer, clock input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P33</td>
<td>P33 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P34</td>
<td>P34 general-purpose I/O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P35/SK1</td>
<td>P35 general-purpose I/O</td>
<td>SK1: 8-bit serial I/O, clock I/O 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P36/SI1</td>
<td>P36 general-purpose I/O</td>
<td>SI1: 8-bit serial I/O, data input 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P37/SO1</td>
<td>P37 general-purpose I/O</td>
<td>SO1: 8-bit serial I/O, data output 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the circuit types, see Section 1.7 "Pin Functions."


### Registers of Port 3

The two registers that are related to Port 3 are PDR3 and DDR3. Bits configuring the PDR3 and DDR3 registers are in a one-to-one correspondence with the pins of Port 3. Table 4.4-2 "Correspondence between the Registers and Pins of Port 3" lists the correspondence between the registers and pins of Port 3.

#### Table 4.4-2 Correspondence between the Registers and Pins of Port 3

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related register and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3</td>
<td>bit7</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P37</td>
</tr>
</tbody>
</table>

---

[Diagram of Port 3, Block Diagram of Pins of Port 3]
4.4.1 Registers of Port 3 (PDR3 and DDR3)

The following explains the registers related to Port 3.

### Functions of the registers of Port 3

- **Port 3 data register (PDR3)**
  
  The PDR3 register indicates the pin status. Thus, a pin set as an output port can read the same value ("0" or "1") as that of the output latch, but the pin cannot read the output latch value when the port is set for input and instead reads the pin status.

  **Reference:**
  
  When a bit processing command (SETB, CLRb) is applied, the value of the output latch rather than the pin is read. So the values of the output latch other than those of the bits to be manipulated do not change.

- **Port 3 direction register (DDR3)**
  
  The DDR3 register sets the input-output direction of pins for each bit.

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

- **Settings for resource output**
  
  When a resource with the output pin is to be used, enable the output enable bit of the resource.

  Since priority is given to the resource output, values set to the PDR3 and DDR3 corresponding to the resource output pin are not related to the output value and output enable/disable of the resource, and thus have no significance.
4.4 Port 3

Settings for resource input

When a resource with the input pin is to be used, set the pin corresponding to input of the resource as an input port. At this time, the value of the corresponding output latch has no significance.

Table 4.4-3 "Functions of the Registers of Port 3" lists the functions of the registers of Port 3.

Table 4.4-3 Functions of the Registers of Port 3

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>0</td>
<td>The pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>000CH</td>
<td>XXXXXXXXxB</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 3 direction register (DDR3)</td>
<td>0</td>
<td>The pin is set for input.</td>
<td>Disables output transistor operations to set the pin for input.</td>
<td>R/W</td>
<td>000DH</td>
<td>00000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin is set for output.</td>
<td>Enables output transistor operations to set the pin for output.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
W: Write-only
X: Undefined
4.4.2 Explanation of Operations of Port 3

The following explains the operations of Port 3.

- Operations of Port 3

  - **Operations when the port is set for output**
    - If the corresponding bit of the DDR3 register is set to "1", the port functions as an output port.
    - When the port is set for output, operations of the output transistor are enabled and data in the output latch is output to the pin.
    - If data is written to the PDR3 register, it is retained in the output latch and then output to the pin directly.
    - If the PDR3 register is read, the value of the pin is read.

  - **Operations when the port is set for input**
    - If the corresponding bit of the DDR3 register is set to "0", the port functions as an input port.
    - When the port is set for input, the output transistor is turned off and the pin is placed in high impedance state.
    - If data is written to the PDR3 register, it is retained in the output latch but is not output to the pin.
    - If the PDR3 register is read, the value of the pin is read.

  - **Operations when the port is set for resource output**
    - If the output enable bit of a resource is enabled, the corresponding pin is set for resource output.
    - It is possible to read the resource output value if the resource output is enabled, because the pin value can be read by the PDR3 register.

  - **Operations when the port is set for resource input**
    - If the bit of the DDR3 register corresponding to the resource input pin is set to "0", the port is set for input.
    - The pin value is always entered as resource input (When the chip is neither in stop mode nor watch mode).
    - The pin value can be read by reading the PDR3 register regardless of whether an input pin is used by the resource.

  - **Operations after reset**
    - If the CPU is reset, the value of the DDR3 register is initialized to "0". Thus, the output transistors are turned off (input port) and the pins are placed in high impedance state.
    - The PDR3 register is not initialized by a reset. Thus, to use the port as an output port, it is necessary to set output data to the PDR3 register and then set the corresponding DDR3 register for output.
4.4 Port 3

- **Operations in stop mode or watch mode**

  If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the pin is placed in high impedance state because the output transistor is forced to off regardless of the value of the DDR3 register. On the other hand, the input is fixed to prevent leakage due to input release.

  Table 4.4-4 "Pin Status of Port 3" lists the pin status of Port 3.

**Table 4.4-4 Pin Status of Port 3**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=0)</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P30/PWM to P37/SO1</td>
<td>General-purpose I/O, resource I/O</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- SPL: Pin status designation bit of the standby control register (STBC: SPL)
- Hi-z: High impedance

**Reference:**

If "Pull-up resistor present" is selected in the option settings, the pin status when the output buffer becomes off is the "H" level (pull-up state) instead of high impedance.
CHAPTER 4 I/O PORTS

4.5 Port 4, 8

Port 4 and Port 8 are output-only ports. This section explains the configuration, pins, a pin block diagram, and related registers of Port 4, 8.

- Configuration of Port 4, 8
  Port 4 and Port 8 each consist of the following two parts:

  - **Port 4**
    - Output-only pins (P40 to P47)
    - Port 4 data register (PDR4)

  - **Port 8**
    - Output-only pins (P80 to P87)
    - Port 8 data register (PDR8)
## Pins of Port 4, 8

Port 4 has five output-only pins for N-channel open-drain output and Port 8 has eight output-only pins for N-channel open-drain output.

Table 4.5-1 "Pins of Port 4, 8" lists the pins of Port 4, 8.

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4</td>
<td>P40</td>
<td>P40 output-only</td>
<td>-</td>
<td>-</td>
<td>N-channel open-drain</td>
</tr>
<tr>
<td></td>
<td>P41</td>
<td>P41 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P42</td>
<td>P42 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P43</td>
<td>P43 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P44</td>
<td>P44 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Port 8</td>
<td>P80</td>
<td>P80 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P81</td>
<td>P81 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P82</td>
<td>P82 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P83</td>
<td>P83 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P84</td>
<td>P84 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P85</td>
<td>P85 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P86</td>
<td>P86 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P87</td>
<td>P87 output-only</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

For circuit types, see Section 1.7 "Pin Functions."
CHAPTER 4 I/O PORTS

Block Diagram of Port 4, 8

Figure 4.5-1 Block Diagram of Pin of Port 4, 8

The two registers related to Port 4, 8 are PDR4 and PDR8. Bits configuring each register are in a one-to-one correspondence with the pins of each port. Table 4.5-2 "Correspondence between the Registers and Pins of Port 4, 8" lists the correspondence between the registers and pins of Port 4, 8.

Table 4.5-2 Correspondence between the Registers and Pins of Port 4, 8

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related register and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4</td>
<td>PDR4</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>-</td>
</tr>
<tr>
<td>Port 8</td>
<td>PDR8</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P87</td>
</tr>
</tbody>
</table>
4.5.1 Registers of Port 4, 8 (PDR4 and PDR8)

The following explains the registers related to Port 4, 8.

- Functions of the Registers of Port 4, 8

  - Port 4, 8 data register (PDR4 and PDR8)

    The PDR4 and PDR8 registers indicate the output latch status. Thus, pin status cannot be read.

    Table 4.5-3 "Functions of the Registers of Port 4, 8" lists the functions of the registers of Port 4, 8.

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>0</td>
<td>The output latch value is &quot;0&quot;.</td>
<td>Outputs the &quot;L&quot; level to the pin (Sets the output latch to &quot;0&quot; and turns on the output transistor).</td>
<td>R/W</td>
<td>000E&lt;sub&gt;H&lt;/sub&gt;</td>
<td>XXXXXXXX&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The output latch value is &quot;1&quot;.</td>
<td>Places the pin in high impedance state (Sets the output latch to &quot;1&quot; and turns off the output transistor).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 8 data register (PDR8)</td>
<td>0</td>
<td>The output latch value is &quot;0&quot;.</td>
<td>Outputs the &quot;L&quot; level to the pin (Sets the output latch to &quot;0&quot; and turns on the output transistor).</td>
<td>R/W</td>
<td>0016&lt;sub&gt;H&lt;/sub&gt;</td>
<td>XXXXXXXX&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The output latch value is &quot;1&quot;.</td>
<td>Places the pin in high impedance state (Sets the output latch to &quot;1&quot; and turns off the output transistor).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
CHAPTER 4 I/O PORTS

4.5.2 Explanation of Operations of Port 4, 8

The following explains the operations of Port 4, 8.

Operations of Port 4, 8

- **Operations as an output port**
  - If data is written to the PDR4, 8 register, it is retained in the output latch. If the value of the output latch is "0", the output transistor is turned on and the "L" level is then output to the pin. If the value of the output latch is "1", the output transistor is turned off and the pin is then placed in high impedance state. If the output pin is pulled up and the output latch value is "1", the port is pulled up.
  - When the PDR4 register is read, the output latch value is always read.

- **Operations after reset**
  If the CPU is reset, the values of the PDR4, 8 registers are initialized to "1". Thus, all output transistors are turned off (input port) and the pins are placed in high impedance state.

- **Operations in stop mode or watch mode**
  If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the output transistor is forced to be turned off and the pin is placed in high impedance state. On the other hand, the input is fixed to prevent leakage due to input release.
  Table 4.5-4 "Pin Status of Port 4, 8" lists the pin status of Port 4, 8.

**Table 4.5-4 Pin Status of Port 4, 8**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=1)</th>
<th>Sub-stop (SPL=1)</th>
<th>Watch mode (SPL=1)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P40 to P44</td>
<td>Output-only port</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P80 to P87</td>
<td>Output-only port</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance

**Reference:**
If "Pull-up resistor present" is selected in the option settings, pin status when the output buffer becomes off is the "H" level (pull-up state) instead of high impedance.
4.6 Port 5

Port 5 is an output-only port that also functions as analog input. Each pin can be used by switching between the analog input and port for each bit. The explanation in this section focuses on the functions as an output-only port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 5.

■ Configuration of Port 5

Port 5 consists of the following two parts:
- Output-only pins/analog input pins (P50/AN00 to P57/AN07)
- Port 5 data register (PDR5)

■ Pins of Port 5

Port 5 has eight output-only pins for N-channel open-drain output. If any of these pins are used as an analog input pin using the A/D converter, do not use as an output port.

Table 4.6-1 “Pins of Port 5” lists the pins of Port 5.

Table 4.6-1 Pins of Port 5

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5</td>
<td>P50/AN00</td>
<td>P50 output-only</td>
<td>AN0 analog input 0</td>
<td>Analog</td>
<td>N-channel open-drain H</td>
</tr>
<tr>
<td></td>
<td>P51/AN01</td>
<td>P51 output-only</td>
<td>AN1 analog input 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P52/AN02</td>
<td>P52 output-only</td>
<td>AN2 analog input 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P53/AN03</td>
<td>P53 output-only</td>
<td>AN3 analog input 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P54/AN04</td>
<td>P54 output-only</td>
<td>AN4 analog input 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P55/AN05</td>
<td>P55 output-only</td>
<td>AN5 analog input 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P56/AN06</td>
<td>P56 output-only</td>
<td>AN6 analog input 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P57/AN07</td>
<td>P57 output-only</td>
<td>AN7 analog input 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For circuit types, see Section 1.7 “Pin Functions.”
CHAPTER 4 I/O PORTS

■ Block Diagram of Port 5

**Figure 4.6-1 Block Diagram of the Pins of Port 5**

![Block Diagram of the Pins of Port 5](image)

**Note:**

When using the A/D converter, do not select “Pull-up resistor present” in the option settings for P57/AN07 to P50/AN00.

Do not use a pin as a general-purpose port that is to be used as an analog input pin.

■ Register of Port 5

There is one register related to Port 5; PDR5. Bits configuring the PDR5 register are in a one-to-one correspondence with the pins of Port 5. Table 4.6-2 "Correspondence between the Register and Pins of Port 5" lists the correspondence between the register and pins of Port 5.

<table>
<thead>
<tr>
<th>Table 4.6-2 Correspondence between the Register and Pins of Port 5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port name</strong></td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>Port 5</td>
</tr>
<tr>
<td>Corresponding pin</td>
</tr>
</tbody>
</table>
4.6.1 Register of Port 5 (PDR5)

The following explains the register related to Port 5.

- **Functions of the Register of Port 5**
  
  - **Port 5 data register (PDR5)**
    
    The PDR5 register indicates the output latch status. Thus, pin status cannot be read.
  
  - **Setting for analog input**
    
    Set to off the output transistor corresponding to the pin to be used as analog input and place the pin in high impedance state.
    
    Table 4.6-3 "Functions of the Register of Port 5" lists the functions of the register of Port 5.

**Table 4.6-3 Functions of the Register of Port 5**

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5 data register (PDR5)</td>
<td>0</td>
<td>The output latch value is &quot;0&quot;.</td>
<td>Outputs the &quot;L&quot; level to the pin (Sets the output latch to &quot;0&quot; and turns on the output transistor).</td>
<td>R/W</td>
<td>0010H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The output latch value is &quot;1&quot;.</td>
<td>Places the pin in high impedance state (Sets the output latch to &quot;1&quot; and turns off the output transistor).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled

---
4.6.2 Explanation of Operations of Port 5

The following explains the operations of Port 5.

- **Operations of Port 5**

  - **Operations as an output port**
    - If data is written to the PDR5 register, the data is retained in the output latch. If the value of the output latch is "0", the output transistor is turned on and the "L" level is then output to the pin. If the value of the output latch is "1", the output transistor is turned off and the pin is then placed in high impedance state. If the output pin is pulled up and the output latch value is "1", the port is pulled up.
    - When the PDR5 register is read, the output latch value is always read.
  
  - **Operations when the port is set for analog input**
    - Set the bit of the PDR5 register corresponding to the analog input pin to "1" and turn off the output transistor.
    - When the PDR5 register is read, the output latch value is read.
  
  - **Operations after reset**
    - If the CPU is reset, the value of the PDR5 register is initialized to "1". Thus, all output transistors are turned off (input port) and the pins are placed in high impedance state.
  
  - **Operations in stop mode or watch mode**
    - If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the output transistor is forced to be turned off and the pin is placed in high impedance state. On the other hand, the input is fixed to prevent leakage due to input release.

Table 4.6-4 "Pin Status of Port 5" lists the pin status of Port 5.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=0)</th>
<th>Sub-sleep (SPL=0)</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P50/AN00 to P57/AN07</td>
<td>Output-only port/analog input</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance

*Reference:* If "Pull-up resistor present" is selected in the option settings, the pin status when the output buffer becomes off is the "H" level (pull-up state) instead of high impedance.
4.7 Port 6

Port 6 is a general-purpose I/O port that also functions as a resource I/O port. Each pin can be used by switching between a resource and a port for each bit. Explanations in this section focus on the functions as a general-purpose I/O port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 6.

- Configuration of Port 6
  Port 6 consists of the following three parts:
  - Output-only pins/analog input pins (P60/TMO1 to P67/BSO1)
  - Port 6 data register (PDR6)
  - Port 6 direction register (DDR6)

- Pins of Port 6
  Port 6 has eight I/O pins for CMOS input and CMOS output.
  If any of these pins that also function for resource I/O ports are used for resource I/O, the pin cannot be used as a general-purpose I/O port.
  Table 4.7-1 "Pins of Port 6" lists the pins of Port 6.

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 6</td>
<td>P60/TMO1</td>
<td>P60 general-purpose I/O</td>
<td>TMO1: 8/16-bit timer/counter, timer output 1</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P61/TMO2</td>
<td>P61 general-purpose I/O</td>
<td>TMO2: 8/16-bit timer/counter, timer output 2</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P62/TCLK</td>
<td>P62 general-purpose I/O</td>
<td>TCLK: 8/16-bit timer/counter, clock input</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P63/MSKO</td>
<td>P63 general-purpose I/O</td>
<td>MSKO: modem signal output</td>
<td>Hysteresis</td>
<td>CMOS</td>
</tr>
<tr>
<td></td>
<td>P64</td>
<td>P64 general-purpose I/O</td>
<td>—</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P65/BSK1</td>
<td>P65 general-purpose I/O</td>
<td>BSK1: serial I/O-1 with 1-byte buffer, clock I/O</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P66/BSI1</td>
<td>P66 general-purpose I/O</td>
<td>BSI1: serial I/O-1 with 1-byte buffer, data input</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td>P67/BSO1</td>
<td>P67 general-purpose I/O</td>
<td>BSO1: serial I/O-1 with 1-byte buffer, data output</td>
<td>Input</td>
<td>Output</td>
</tr>
</tbody>
</table>

For circuit types, see Section 1.7 "Pin Functions."
CHAPTER 4 I/O PORTS

■ Block Diagram of Port 6

Figure 4.7-1 Block Diagram of Pins of Port 6

■ Registers of Port 6

The two registers that are related to Port 6 are PDR6 and DDR6. Bits configuring each register are in a one-to-one correspondence with the pins of Port 6. Table 4.7-2 "Correspondence between the Registers and Pins of Port 6" lists the correspondence between the registers and the pins of Port 6.

Table 4.7-2 Correspondence between the Registers and Pins of Port 6

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related register and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 6</td>
<td>bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P67 P66 P65 P64 P63 P62 P61 P60</td>
</tr>
</tbody>
</table>
4.7.1 Registers of Port 6 (PDR6 and DDR6)

The following explains the registers related to Port 6.

Functions of the Registers of Port 6

- **Port 6 data register (PDR6)**
  The PDR6 register indicates pin status. Thus, a pin set as an output port can read the same value ("0" or "1") as that of the output latch, but the pin cannot read the output latch value when the port is set for input and instead reads the pin status.

  **Reference:**
  When a bit processing command (SETB, CLRB) is applied, the value of the output latch instead of the pin is read, and so values of the output latch other than those of bits to be manipulated do not change.

- **Port 6 direction register (DDR6)**
  The DDR6 register sets the input-output direction of pins for each bit.

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

- **Settings for resource output**
  When a resource with the output pin is to be used, enable the output enable bit of the resource.

  Since priority is given to the resource output, values set to the PDR6 and DDR6 registers corresponding to the resource output pin are not related to the output value and output enable/disable of the resource, and thus have no significance.
Settings for resource input

When a resource with the input pin is to be used, set the pin corresponding to input of the resource as an input port. At this time, the value of the corresponding output latch has no significance.

Table 4.7-3 "Functions of the Registers of Port 6" lists the functions of the registers of Port 6.

Table 4.7-3 Functions of the Registers of Port 6

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>0</td>
<td>The pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>0012H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 6 direction register (DDR6)</td>
<td>0</td>
<td>The pin is set for input.</td>
<td>Disables output transistor operations to set the pin for input.</td>
<td>R/W</td>
<td>0013H</td>
<td>0000000B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin is set for output.</td>
<td>Enables output transistor operations to set the pin for output.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
W: Write-only
X: Undefined
4.7.2  Explanation of Operations of Port 6

The following explains the operations of Port 6.

■ Operations of Port 6

- **Operations when the port is set for output**
  - If the corresponding bit of the DDR6 register is set to "1", the port functions as an output port.
  - When the port is set for output, operations of the output transistor are enabled and data in the output latch is output to the pin.
  - If data are written to the PDR6 register, the data are retained in the output latch and then output to the pin directly.
  - If the PDR6 register is read, the value of the pin is read.

- **Operations when the port is set for input**
  - If the corresponding bit of the DDR6 register is set to "0", the port functions as an input port.
  - When the port is set for input, the output transistor is turned off and the pin is placed in high impedance state.
  - If data are written to the PDR6 register, the data is retained in the output latch but is not output to the pin.
  - If the PDR6 register is read, the value of the pin is read.

- **Operations when the port is set for resource output**
  - If the output enable bit of a resource is enabled, the corresponding pin is set for resource output.
  - It is possible to read the resource output value if the resource output is enabled, because the pin value can be read by the PDR6 register.

- **Operations when the port is set for resource input**
  - If the bit of the DDR6 register corresponding to the resource input pin is set to "0", the port is set for input.
  - The pin value is always entered as resource input (When the chip is neither in stop mode nor watch mode).
  - The pin value can be read by reading the PDR6 register regardless of whether an input pin is used by the resource.

- **Operations after reset**
  - If the CPU is reset, the value of the DDR6 register is initialized to "0". Thus, the output transistors are turned off (input port) and the pins are placed in high impedance state.
  - The PDR6 register is not initialized by a reset. Thus, to use the port as an output port, it is necessary to set output data to the PDR6 register and set the corresponding DDR6 register for output.
Operations in stop mode or watch mode

If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the pin is placed in high impedance state, because the output transistor is forced to be turned off regardless of the value of the DDR6 register. On the other hand, the input is fixed to prevent leakage due to input release.

Table 4.7-4 "Pin Status of Port 6" lists the pin status of Port 6.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main sleep</th>
<th>Main stop (SPL=0)</th>
<th>Sub-sleep</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>Main stop (SPL=1)</th>
<th>Sub-stop (SPL=1)</th>
<th>Watch mode (SPL=1)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P60/TMO1 to P67/BSO1</td>
<td>General-purpose I/O port/ resource I/O port</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td></td>
<td></td>
<td></td>
<td>Hi-z</td>
<td></td>
<td></td>
<td>Hi-z</td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance

Reference:

If "Pull-up resistor present" is selected in the option settings, the pin status when the output buffer becomes "OFF" is the "H" level (pull-up state) instead of high impedance.
4.8 Port 7

Port 7 is a general-purpose I/O port that also functions as a resource I/O port. Each pin can be used by switching between a resource and port for each bit. Explanations in this section focus on the functions as a general-purpose I/O port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 7.

■ Configuration of Port 7

Port 7 consists of the following two parts:

- Output-only pins/analog input pins (P70/SK2 to P77)
- Port 7 data register (PDR7)

■ Pins of Port 7

Port 7 has eight I/O pins for hysteresis input/N-channel open-drain output. If any of these pins that also function as resource I/O ports are used for resource I/O, they cannot be used as a general-purpose I/O port.

Table 4.8-1 "Pins of Port 7" lists the pins of Port 7.

Table 4.8-1 Pins of Port 7

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input (Hysteresis)</td>
<td>N-channel open-drain</td>
</tr>
<tr>
<td>P70/SK2</td>
<td>P70</td>
<td>general-purpose I/O</td>
<td>SK2: 8-bit serial I/O-2, clock I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P71/SI1</td>
<td>P71</td>
<td>general-purpose I/O</td>
<td>SI2: 8-bit serial I/O-2, data input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P72/SO2</td>
<td>P72</td>
<td>general-purpose I/O</td>
<td>SO2: 8-bit serial I/O-2, data output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P73/BSK2</td>
<td>P73</td>
<td>general-purpose I/O</td>
<td>BSK2: serial I/O-2 with 1-byte buffer, clock I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P74/BSI2</td>
<td>P74</td>
<td>general-purpose I/O</td>
<td>BSI2: serial I/O-2 with 1-byte buffer, data input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P75/BSO2</td>
<td>P75</td>
<td>general-purpose I/O</td>
<td>BSO2: serial I/O-2 with 1-byte buffer, data output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P76</td>
<td>P76</td>
<td>general-purpose I/O</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P77</td>
<td>P77</td>
<td>general-purpose I/O</td>
<td>—</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For circuit types, see Section 1.7 "Pin Functions."
CHAPTER 4 I/O PORTS

- Block diagram of Port 7

**Figure 4.8-1** Block Diagram of Pins of Port 7

- Registers of Port 7

  The one register related to Port 7 is PDR7. Bits configuring the PDR7 register are in one-to-one correspondence with the pins of Port 7. Table 4.8-2 "Correspondence between the Registers and Pins of Port 7" lists the correspondence between the registers and pins of Port 7.

**Table 4.8-2** Correspondence between the Registers and Pins of Port 7

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the related register and corresponding pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 7</td>
<td>bit7  bit6  bit5  bit4  bit3  bit2  bit1  bit0</td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P77  P76  P75  P74  P73  P72  P71  P70</td>
</tr>
</tbody>
</table>
4.8.1 Register of Port 7 (PDR7)

The following explains the register related to Port 7.

- **Functions of the Register of Port 7**

  - **Port 7 data register (PDR7)**
    
    The PDR7 register indicates the pin status. Thus, a pin set as an output port can read the same value ("0" or "1") as that of the output latch, but the pin cannot read the output latch value when the port is set for input and reads the pin status instead.

    **Reference:**
    
    When a bit processing command (SETB, CLRZ) is applied, the value of the output latch instead of the pin is read, and so values of the output latch other than those of the bits to be manipulated do not change.

  - **Settings for resource output**
    
    When a resource with the output pin is to be used, enable the output enable bit of the resource. Since priority is given to the resource output, the value set to the PDR7 register corresponding to the resource output pin is not related to the output value and output enable/disable of the resource, and thus has no significance.

  - **Settings for resource input**
    
    When a resource with the input pin is to be used, set the pin corresponding to input of the resource as an input port. At this time, the value of the corresponding output latch has no significance.

    Table 4.8-3 "Functions of the Register of Port 7" lists the functions of the register of Port 7.

**Table 4.8-3 Functions of the Register of Port 7**

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 7 data register (PDR7)</td>
<td>0</td>
<td>The output latch value is &quot;0&quot;.</td>
<td>Outputs the &quot;L&quot; level to the pin (Sets the output latch to &quot;0&quot; and turns on the output transistor).</td>
<td>R/W</td>
<td>0014\textsubscript{H}</td>
<td>XXXXXXXXXX\textsubscript{B}</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The output latch value is &quot;1&quot;.</td>
<td>Places the pin in high impedance state (Sets the output latch to &quot;1&quot; and turns off the output transistor).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
4.8.2 Explanation of Operations of Port 7

The following explains the operations of Port 7.

- **Operations of Port 7**

  - **Operations when the port is set for output**
    - If data is written to the PDR7 register, the data is retained in the output latch. If the value of the output latch is "0", the output transistor is turned on and the "L" level is then output to the pin. If the value of the output latch is "1", the output transistor is turned off and the pin is then placed in high impedance state. If the output pin is pulled up and the output latch value is "1", the port is pulled up.
    - If the PDR7 register is read, the value of the output latch is always read.

  - **Operations when the port is set for input**
    - If the corresponding bit of the PDR7 register is set to "1", the output transistor is turned off and the pin is placed in high impedance state.
    - If the PDR7 register is read, the value of the pin is read.

  - **Operations when the port is set for resource output**
    - If the output enable bit of a resource is enabled, the corresponding pin is set for resource output.
    - It is possible to read the resource output value if the resource output is enabled, because the pin value can be read by the PDR7 register.

  - **Operations when the port is set for resource input**
    - If the bit of the PDR7 register corresponding to the resource input pin is set to "1", the port is set for input.
    - The pin value is always entered as resource input (When the chip is neither in stop mode nor watch mode).
    - The pin value can be read by reading the PDR7 register regardless of whether an input pin is used by the resource.

  - **Operations after reset**
    - If the CPU is reset, the value of the PDR7 register is initialized to "1". Thus, all output transistors are turned off (input port) and the pins are placed in high impedance state.
Operations in stop mode or watch mode

If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the output transistor is forced to be turned off and the pin is placed in high impedance state. On the other hand, the input is fixed to prevent leakage due to input release.

Table 4.8-4 "Pin Status of Port 7" lists the pin status of Port 7.

Table 4.8-4 Pin Status of Port 7

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=1)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Main sleep</td>
<td>Sub-stop (SPL=1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Main stop (SPL=0)</td>
<td>Sub-stop (SPL=0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sub-sleep</td>
<td>Watch mode (SPL=0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Main stop (SPL=1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sub-stop (SPL=1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Watch mode (SPL=1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>After reset</td>
<td></td>
</tr>
<tr>
<td>P70/SCK2 to P77</td>
<td>General-purpose I/O port/ resource I/O port</td>
<td>Hi-z</td>
<td>Hi-z</td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance

Reference:

If "Pull-up resistor present" is selected in the option settings, the pin status when the output buffer becomes "OFF" is the "H" level (pull-up state) instead of high impedance.
Port 9 and Port A are general-purpose I/O ports that also function as external interrupt input pins. Explanations in this section focus on the functions as a general-purpose I/O port. This section explains the configuration, pins, a pin block diagram, and related registers of Port 9, A.

### Configuration of Port 9, A

Port 9 and Port A each consist of the following three parts:

- **Port 9**
  - General-purpose I/O pins/external interrupt-2 input pins (P90/INT20 to P97/INT27)
  - Port 9 data register (PDR9)
  - Port 9 direction register (DDR9)

- **Port A**
  - General-purpose I/O pins/external interrupt-2 input pins (PA0/INT28 to PA3/INTB) and general-purpose I/O pins/general-purpose interrupt-1 input pins (PA4/INT0 to PA7/INT3)
  - Port A data register (PDRA)
  - Port A direction register (DDRA)
4.9 Port 9, A

Pins of Port 9, A

Port 9 and Port A each have eight I/O pins for CMOS input and CMOS output. If these pins are used as input pins, they can also be used as external interrupt input pins. Table 4.9-1 "Pins of Port 9, A" lists the pins of Port 9, A.

Table 4.9-1 Pins of Port 9, A

<table>
<thead>
<tr>
<th>Port name</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared resources</th>
<th>Input-output type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9</td>
<td>P90/INT20</td>
<td>P90 general-purpose I/O</td>
<td>INT20 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P91/INT21</td>
<td>P91 general-purpose I/O</td>
<td>INT21 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P92/INT22</td>
<td>P92 general-purpose I/O</td>
<td>INT22 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P93/INT23</td>
<td>P93 general-purpose I/O</td>
<td>INT23 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P94/INT24</td>
<td>P94 general-purpose I/O</td>
<td>INT24 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P95/INT25</td>
<td>P95 general-purpose I/O</td>
<td>INT25 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P96/INT26</td>
<td>P96 general-purpose I/O</td>
<td>INT26 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P97/INT27</td>
<td>P97 general-purpose I/O</td>
<td>INT27 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port A</td>
<td>PA0/INT28</td>
<td>PA0 general-purpose I/O</td>
<td>INT28 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA1/INT29</td>
<td>PA1 general-purpose I/O</td>
<td>INT29 external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA2/INTA</td>
<td>PA2 general-purpose I/O</td>
<td>INTA external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA3/INTB</td>
<td>PA3 general-purpose I/O</td>
<td>INTB external interrupt-2 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA4/INT0</td>
<td>PA4 general-purpose I/O</td>
<td>INT0 external interrupt-1 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA5/INT1</td>
<td>PA5 general-purpose I/O</td>
<td>INT1 external interrupt-1 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA6/INT2</td>
<td>PA6 general-purpose I/O</td>
<td>INT2 external interrupt-1 input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PA7/INT3</td>
<td>PA7 general-purpose I/O</td>
<td>INT3 external interrupt-1 input</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the circuit types, see Section 1.7 "Pin Functions."
CHAPTER 4 I/O PORTS

Block Diagram of Port 9, A

Figure 4.9-1 Block Diagram of Pins of Port 9, A

Note:
It is necessary to disable operations of the external interrupt circuit corresponding to the pin if the pin is to be used as a normal I/O port, because the pin value is always entered in the external interrupt circuit. For more details, see Chapter 13 "EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)," and Chapter 14 "EXTERNAL INTERRUPT CIRCUIT 2 (LEVEL)."

Registers of Port 9, A

The four registers that are related to Port 9, A are PDR9, DDR9, PDRA, and DDRA. Bits configuring each register are in a one-to-one correspondence with the pins of each port. Table 4.9-2 "Correspondence between the Registers and Pins of Port 9, A" lists the correspondence between the registers and pins of Port 9, A.

Table 4.9-2 Correspondence between the Registers and Pins of Port 9, A

<table>
<thead>
<tr>
<th>Port name</th>
<th>Bits of the 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>Corresponding pin P97 P96 P95 P94 P93 P92 P91 P90</td>
</tr>
<tr>
<td>Port A</td>
<td>PDRA, DDRA bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0</td>
</tr>
</tbody>
</table>
### 4.9.1 Registers of Port 9, A (PDR9, DDR9, PDRA, and DDRA)

The following explains the registers related to Port 9, A.

#### Functions of the Registers of Port 9, A

- **Port 9, A data register (PDR9/PDRA)**
  
The PDR9 and PDRA registers indicate the pin status. Thus, a pin set as an output port can read the same value ("0" or "1") as that of the output latch, but the pin cannot read the output latch value when the port is set for input and instead reads the pin status.

  **Reference:**
  
  When a bit processing command (SETB, CLRB) is applied, the value of the output latch instead of the pin is read, and so values of the output latch other than those of bits to be manipulated do not change.

- **Port 9, A direction register (DDR9/DDRA)**
  
The DDR9 and DDRA registers set the input-output direction of pins for each bit.

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

- **Settings for external interrupt input**
  
  To use the pin as an external interrupt input pin, enable operations of the external interrupt circuits (1 and 2) and set the corresponding pins as an input port. At this time, the values of the corresponding output latches have no significance.

  Table 4.9-3 "Functions of the Registers of Port 9, A" lists the functions of the registers of Port 9, A.

**Table 4.9-3 Functions of the Registers of Port 9, A**

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9 data register (PDR9)</td>
<td>0</td>
<td>The pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>0018\text{H}</td>
<td>XXXXXXXXX\text{B}</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 4.9-3 Functions of the Registers of Port 9, A (Continued)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Data</th>
<th>When reading</th>
<th>When writing</th>
<th>R/W</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 9 direction register (DDR9)</td>
<td>0</td>
<td>The pin is set for input.</td>
<td>Disables output transistor operations to set the pin for input.</td>
<td>R/W</td>
<td>0019_H</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin is set for output.</td>
<td>Enables output transistor operations to set the pin for output.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port A data register (PDRA)</td>
<td>0</td>
<td>The pin status is the &quot;L&quot; level.</td>
<td>Sets the output latch to &quot;0&quot;. When the port is set for output, the &quot;L&quot; level is output to the pin.</td>
<td>R/W</td>
<td>001A_H</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin status is the &quot;H&quot; level.</td>
<td>Sets the output latch to &quot;1&quot;. When the port is set for output, the &quot;H&quot; level is output to the pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port A direction register (DDRA)</td>
<td>0</td>
<td>The pin is set for input.</td>
<td>Disables output transistor operations to set the pin for input.</td>
<td>R/W</td>
<td>001B_H</td>
<td>00000000_B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>The pin is set for output.</td>
<td>Enables output transistor operations to set the pin for output.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled  
W: Write-only  
X: Undefined
4.9.2 Explanation of Operations of Port 9, A

The following explains the operations of Port 9, A.

- **Operations of Port 9, A**

  - **Operations when the port is set for output**
    - If the corresponding bit of the DDR9 or DDRA register is set to "1", the port functions as an output port.
    - When the port is set for output, operations of the output transistor are enabled and data in the output latch is output to the pin.
    - If data is written to the PDR9, PDRA register, it is retained in the output latch and then output to the pin directly.
    - If the PDR9, PDRA register is read, the value of the pin is read.

  - **Operations when the port is set for input**
    - If the corresponding bit of the DDR9, DDRA register is set to "0", the port functions as an input port.
    - When the port is set for input, the output transistor is turned off and the pin is placed in high impedance state.
    - If data is written to the PDR9, DDRA register, it is retained in the output latch but is not output to the pin.
    - If the PDR9, PDRA register is read, the value of the pin is read.

  - **Operations when the port is set for external interrupt input**
    - If the bit of the DDR9, DDRA register corresponding to the external interrupt input pin is set to "0", the port is set for input.
    - The pin value can be read by reading the PDR9, PDRA register regardless of the external interrupt input and enable/disable of the interrupt request output.

  - **Operations after reset**
    - If the CPU is reset, the values of the DDR9 and DDRA registers are initialized to "0". Thus, the output transistors are turned off (input port) and the pins are placed in high impedance state.
    - The PDR9, PDRA register is not initialized by a reset. Thus, to use the port as an output port, it is necessary to set output data to the PDR9 or PDRA register and then set the corresponding DDR9 or PDRA register for output.
CHAPTER 4 I/O PORTS

- Operations in stop mode or watch mode

If the pin status designation bit (STBC: SPL) of the standby control register is set to "1" when the chip is placed in stop mode or watch mode, the pin is placed in high impedance state, because the output transistor is forced to be turned off regardless of the value of the DDR9, DDRA register.

Table 4.9-4 "Pin Status of Port 9, A" lists the pin status of Port 9, A.

Table 4.9-4 Pin Status of Port 9, A

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main stop (SPL=0)</th>
<th>Sub-stop (SPL=0)</th>
<th>Watch mode (SPL=0)</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P90/INT20 to P97/INT27</td>
<td>General-purpose I/O port/external interrupt I/O port</td>
<td>Hi-z (external interrupt input)</td>
<td></td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>PA0/INT25 to PA7/INT3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SPL: Pin status designation bit of the standby control register (STBC: SPL)
Hi-z: High impedance
4.10 Program Example of I/O Ports

The following shows a program example in which I/O ports are used.

Program Example of the I/O ports

Processing specifications

- All 7-segment (8-segment if Dp is included) LEDs are turned on by Port 0, 1.
- The P00 pin corresponds to the anode common pin of LED and the P10 to P17 pins correspond to each segment pin.

Figure 4.10-1 Example of 8-segment LED Connection shows an example of 8-segment LED connection.

Example of coding

CSEG ; [CODE SEGMENT]

CLRB PDR0:0 ; Set the "L" level to P00.
MOV PDR1, #11111111B ; Set the "H" level to all bits of Port 1.
MOV DDR0, #11111111B ; Set output to P00, enabled by #xxxxxxx1B.
MOV DDR1, #11111111B ; Set output for all bits of Port 1.

ENDS

END
CHAPTER 5  TIMEBASE TIMER

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

5.1 "Overview of the Timebase Timer"
5.2 "Configuration of the Timebase Timer"
5.3 "Timebase Timer Control Register (TBTC)"
5.4 "Timebase Timer Interrupts"
5.5 "Explanation of Operations of the Timebase Timer"
5.6 "Precautions when Using the Timebase Timer"
5.7 "Program Example of the Timebase Timer"
5.1 Overview of the Timebase Timer

The timebase timer is a 21-bit free-running counter that counts up in synchronization with the internal count clock (main clock source oscillation divided by 2). It provides the interval timer function that enables one to select from four types of interval time. The timebase timer also supplies the timer output of oscillation stabilization wait time and the operating clock of the watchdog timer. The timebase timer stops operation in the mode in which the main clock source oscillation stops.

■ Interval Timer Function

The interval timer function is a function for generating an interrupt repeatedly at regular intervals.

- An interrupt occurs when an overflow error of the bit for the interval timer in the counter of the timebase timer occurs.
- The bit for the interval timer (interval time) can be selected from four types of interval time.

Table 5.1-1 “Interval Time of the Timebase Timer” lists the four types of interval time of the timebase timer.

Table 5.1-1 Interval Time of the Timebase Timer

<table>
<thead>
<tr>
<th>Internal count clock cycle</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2/F_{CH}$ (0.25 μs)</td>
<td>$2^{13}/F_{CH}$ (About 1.02 ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{15}/F_{CH}$ (About 4.10 ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{18}/F_{CH}$ (About 32.8 ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{22}/F_{CH}$ (About 524.3 ms)</td>
</tr>
</tbody>
</table>

$F_{CH}$: Main clock source oscillation
Values in ( ) are the interval time when the main clock source oscillation frequency is 8 MHz.
5.1 Overview of the Timebase Timer

Clock Supply Function

The clock supply function is a function to supply the timer output (four types) for oscillation stabilization wait time of the main clock and operating clock to part of the peripheral functions.

Table 5.1-2 "Clock Supplied from the Timebase Timer" lists the clock frequencies supplied to each peripheral function from the timebase timer.

Table 5.1-2 Clock Supplied from the Timebase Timer

<table>
<thead>
<tr>
<th>Clock receiver</th>
<th>Clock cycle</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillation stabilization wait time of the main clock</td>
<td>(2^4/F_{CH}) (About 0.0 ms)</td>
<td>Selected by the oscillation stabilization wait time selection bit (SYCC: WT1, WT0) of the system clock control register in the clock controller</td>
</tr>
<tr>
<td></td>
<td>(2^{12}/F_{CH}) (About 0.512 ms)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2^{16}/F_{CH}) (About 8.19 ms)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2^{18}/F_{CH}) (About 32.77 ms)</td>
<td></td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>(2^{22}/F_{CH}) (About 524.3 ms)</td>
<td>Count-up clock of the watchdog timer</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>(2^{10}/F_{CH}) to (2^{13}/F_{CH}) (About 0.13 to 1.02 ms)</td>
<td>See Chapter 12 &quot;BUZZER OUTPUT.&quot;</td>
</tr>
</tbody>
</table>

\(F_{CH}\): Main clock source oscillation

Values in ( ) are the interval time when the main clock source oscillation frequency is 8 MHz.

Reference:

The oscillation cycle is unstable just after the start of oscillation, and the oscillation stabilization wait time serves as a guideline.
5.2 Configuration of the Timebase Timer

The timebase timer consists of the following four blocks:
- Timebase timer counter
- Counter clear circuit
- Interval time selector
- Timebase timer control register (TBTC)

Block Diagram of the Timebase Timer

- **Timebase timer counter**
  A 21-bit up-counter with the main clock source oscillation divided by 2 as its count clock. The timebase timer counter stops its operation when the main clock source oscillation stops.

- **Counter clear circuit**
  Clears the counter when the transition to the main stop mode (STBC: STP=1) or sub-clock mode (SYCC: SCS=0) occurs or a power-on reset (optional) occurs in addition to the setting of the TBTC register (TBR=0).
5.2 Configuration of the Timebase Timer

- **Interval timer selector**
  Circuit to select one bit for the interval timer from four bits in the timebase timer counter. The overflow of the selected bit becomes an interrupt source.

- **Timebase timer control register (TBTC)**
  Used to select the interval time, clear the counter, control the interrupts, and check the status.
CHAPTER 5  TIMEBASE TIMER

5.3  Timebase Timer Control Register (TBTC)

The timebase timer control register (TBTC) is used to select the interval time, clear the counter, control the interrupts, and check the status.

- Timebase Timer Control Register (TBTC)

![Figure 5.3-1  Timebase Timer Control Register (TBTC)]

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 Ah</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00XXX0000b</td>
</tr>
</tbody>
</table>

- **R/W**: Read/write enabled
- **W**: Write-only
- **X**: Undefined
- **/: Initial value

**Timebase timer initialize bit (TBR)**
- Read cycle: No change.
- Write cycle: Clears the counter of the timebase timer.

**Interval time selection bit (TBC1, TBC0)**
- 00: $2^{15}$/$F_{CH}$
- 01: $2^{14}$/$F_{CH}$
- 10: $2^{13}$/$F_{CH}$
- 11: $2^{12}$/$F_{CH}$

- $F_{CH}$: Main clock source oscillation

**Interrupt request enable bit (TBIE)**
- 0: Enables interrupt request output
- 1: Disables interrupt request output

**Overflow interrupt request flag bit (TBOF)**
- 0: No overflow of the specified bit
- 1: Overflow of the specified bit
### 5.3 Timebase Timer Control Register (TBTC)

Table 5.3-1 Functional Explanation of Each Bit of the Timebase Timer Control Register (TBTC)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit name</th>
<th>Function</th>
</tr>
</thead>
</table>
| Bit 7 | TBOF: Overflow interrupt request flag bit | • This bit is set to "1" when the overflow of the specified bit in the counter of the timebase timer.  
• If this bit and the interrupt request enable bit (TBIE) are "1", an interrupt request is output.  
• This bit is cleared by writing "0" to the bit. If "1" is written, no change occurs and operations are not affected. |
| Bit 6 | TBIE: Interrupt request enable bit     | Bit to enable/disable interrupt request output to the CPU. If this bit and the overflow interrupt request flag bit (TBOF) are "1", an interrupt request is output. |
| Bit 5 | Unused                                | • Value when these bits are read is not defined.  
• Writing to these bits does not affect operations. |
| Bit 4 | Bit 3                                 |                                                                                                                                         |
| Bit 3 | TBC1, TBC0: Interval time selection bit | Bit to select the cycle of the interval timer. The bit for interval timer in the counter of the timebase timer is specified. Four types of interval time can be selected. |
| Bit 2 | Bit 1                                 |                                                                                                                                         |
| Bit 1 | TBR: Timebase timer initialize bit    | Bit to initialize the counter of the timebase timer. Writing "0" to this bit clears the counter to "000000H". If "1" is written, operations are not affected.  
**Reference:**  
"1" is always read. |
| Bit 0 |                                       |                                                                                                                                         |
5.4 Timebase Timer Interrupts

As an interrupt source of the timebase timer, overflow of the specified bit of the timebase timer counter is available (interval timer function).

- **Interrupt When the Interval Timer Function is ON**
  
  The overflow interrupt request flag bit (TBTC: TBOF) is set to "1" when the counter is counted up by the internal count clock and thus an overflow error of the selected bit for interval timer occurs. If, at this point, the interrupt request enable bit is enabled (TBTC: TBIE=1), an interrupt request (IRQA) to the CPU occurs. Write "0" to the TBOF bit using an interrupt processing routine to clear the interrupt request. The TBOF bit is set when an overflow of the specified bit occurs regardless of the value of the TBIE bit.

  **Note:**
  
  To enable (TBIE=1) interrupt request output after releasing a reset, clear the TBOF bit simultaneously.

  **Reference:**
  
  If the TBIE bit is changed from disable to enable (0 --> 1) when the TBOF bit is "1", an interrupt request is generated immediately.

  If the selection of counter clear (TBTC: TBR=0) and an overflow of the selected bit occur simultaneously, the TBOF bit is not set.

- **Oscillation Stabilization Wait Time and Timebase Timer Interrupts**
  
  If an interval time shorter than the oscillation stabilization wait time of the main clock is set, an interval interrupt request (TBTC: TBOF=1) of the timebase timer is generated when the operation in main clock mode is started. In this case, disable (TBTC: TBIE=0) the interrupts of the timebase timer when transition to a mode (main stop mode or sub-clock mode) in which the main clock stops occurs.

- **Register and Vector Table Related to the Timebase Timer Interrupt**
  
  **Table 5.4-1 Register and Vector Table Related to the Timebase Timer Interrupt**

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Register to set the interrupt level</th>
<th>Address of the vector table</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQA</td>
<td>ILR3 (007EH)</td>
<td>FFE6H</td>
</tr>
<tr>
<td></td>
<td>LA1 (bit 5)</td>
<td>FFE7H</td>
</tr>
<tr>
<td></td>
<td>LA0 (bit 4)</td>
<td></td>
</tr>
</tbody>
</table>

For the interrupt operations, see Section 3.4.2 "Flow of Interrupt Operation."
5.5 Explanation of Operations of the Timebase Timer

The timebase timer operates and provides the interval timer function and the clock supply function to part of the peripherals.

■ Operation of the Interval Timer Function (Timebase Timer)

The setting in Figure 5.5-1 is required so that the interval timer function can operate correctly.

![Figure 5.5-1 Setting for the Interval Timer Function](image)

As long as the main clock oscillates, the counter of the timebase timer continues to count up in synchronization with the internal count clock (main clock source oscillation divided by 2).

If the counter is cleared (TBR=0), the count-up starts with "0". If an overflow error of the bit for the interval timer occurs, the overflow interrupt request flag bit (TBOF) is set. That is, an interrupt request is generated at intervals of the selected interval time with reference to the time at which the counter was cleared.

■ Operations of the Clock Supply Function

The timebase timer is also used as a timer to create the oscillation stabilization wait time of the main clock. The oscillation stabilization wait time is the time interval from the count-up in a state in which the counter of the timebase timer is cleared to an overflow of the bit for the oscillation stabilization wait time. Select one type from four types of oscillation stabilization wait time using the oscillation stabilization wait time selection bit (SYCC: WT1, WT0) of the system clock control register.

The timebase timer supplies the clock to the watchdog timer and buzzer output. Clearing the timebase timer counter affects the operations of the buzzer output. When the timebase timer output is selected (WDTC: CS=0), the counter of the watchdog timer is cleared simultaneously.

■ Operations of the Timebase Timer

Figure 5.5-2 "Operations of the Timebase Timer" shows the operation in the following conditions:

- When a power-on reset occurs.
- When the sleep mode is entered during operation of the interval timer function in main clock mode
- When the main stop mode is entered.
- When the counter clear is requested.

In sub-clock mode and main stop mode, the timebase timer is cleared and its operation is stopped. When returning from the sub-clock mode or main stop mode, the oscillation stabilization wait time is counted by the timebase timer.
Figure 5.5-2 Operations of the Timebase Timer

If "11b" is set to the interval time selection bit (TBTC: TBC1, TBC0) of the timebase timer control register (22/FCH)

[Diagram showing operations of the Timebase Timer]

Indicates the oscillation stabilization wait time.
5.6 Precautions when Using the Timebase Timer

The following shows the precautions to be taken when using the timebase timer.

- Precautions When Using the Timebase Timer

  - Precautions when the timebase timer is set in programs
    
    It is not possible to return from interrupt processing if the interrupt request flag bit (TBTC: TBOF) is set to "1" and the interrupt request enable bit is enabled (TBTC: TBIE=1), therefore, the TBOF bit must be cleared.

  - Clearing the timebase timer
    
    The timebase timer can be cleared, in addition to clearing by the timebase timer initialize bit (TBTC: TBR=0), when the oscillation stabilization wait time of the main clock is needed. When the timebase timer is selected (WDTC: CS=0) as the count clock of the watchdog timer, the watchdog timer is cleared when the timebase timer is cleared.

  - When the timebase timer is used as the timer for oscillation stabilization wait time
    
    Since the main clock source oscillation is stopped during power on or in main stop mode or sub-clock mode, it is necessary to take the oscillation stabilization wait time for the main clock using the timebase timer after oscillation starts.
    
    Depending on the type of vibrator connected to the oscillator (clock generator) of the main clock, it is necessary to select the appropriate oscillation stabilization wait time.
    
    For more details, see Section 3.6.5 "Oscillation Stabilization Wait Time."

  - Precautions for the peripheral functions to which the clock is supplied from the timebase timer
    
    In a mode in which the main clock source oscillation is stopped, the counter is cleared and the operation of the timebase timer is stopped. When the timebase timer counter is cleared, the clock supplied from the timebase timer is output from the initial state. Thus, the "H" level may be shorter or the "L" level may be longer by a maximum of 1/2 cycle. The clock for the watchdog timer is also output from the initial state, but the watchdog timer operates in normal cycles because the counter of the watchdog timer is cleared simultaneously.
    
    Figure 5.6-1 "Influence on Buzzer Output by Clearing of the Timebase Timer" shows the influence on buzzer output by clearing of the timebase timer.
Figure 5.6-1 Influence on Buzzer Output by Clearing of the Timebase Timer

X: Arbitrary value, but cleared to "0"
When "011s" is set to the buzzer selection bit (BZCR: BZ2, BZ1, BZ0) of the buzzer register
(main clock source oscillation divided by 2048, about 3.91 kHz is output when operating at 8 MHz)
5.7 Program Example of the Timebase Timer

The following shows a program example of the timebase timer.

- Program Example of the Timebase Timer

  - Processing specifications
  
  An interval timer interrupt of $2^{18}/F_{CH}$ ($F_{CH}$ main clock source oscillation) is generated repeatedly. The interval time in this case is about 32.8 ms (when operating at 8 MHz).

  - Example of coding

```assembly
TBTC EQU 0000AH ; Address of the timebase timer control register
TBOF EQU TBTC:7 ; Interrupt request flag bit definition
ILR3 EQU 007EH ; Address of the interrupt level setting register
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFE6H
IRQA DW WARI ; Interrupt vector setting
INT_V ENDS
;-----Main program-------------------------------------------------------------------------------------------------------------------
CSEG ; [CODE SEGMENT]
; The stack pointer (SP) and others are assumed to have been initialized.

: CLRI ; Interrupt disable
MOV ILR3,#11011111B ; Interrupt level setting (level 1)
MOV TBTC,#01000100B ; Interrupt request flag clear, interrupt request output enable, $2^{18}/F_{CH}$ selection, and timebase timer clear
SETI ; Interrupt enable
:
;-----Interrupt program------------------------------------------------------------------------------------------------------------------
WARI CLRIB TBOF ; Interrupt request flag clear
PUSHW A
XCHW A,T
PUSHW A
:
User processing
:
POPW A
XCHW A,T
POPW A
RETI
ENDS
;--------------------------------------------------------------------------------------
END
```
CHAPTER 6  WATCHDOG TIMER

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

6.1  "Overview of the Watchdog Timer"
6.2  "Configuration of the Watchdog Timer"
6.3  "Watchdog Control Register (WDTC)"
6.4  "Explanation of Operations of the Watchdog Timer"
6.5  "Precautions when Using the Watchdog Timer"
6.6  "Program Example of the Watchdog Timer"
CHAPTER 6 WATCHDOG TIMER

6.1 Overview of the Watchdog Timer

The watchdog timer is a 1-bit counter that takes the output of either the timebase timer operating with the main clock or the clock prescaler operating with the sub-clock as its count clock. The CPU is reset if the watchdog timer is not cleared within a specified period of time.

■ Watchdog Timer Function

The watchdog timer is a counter to prevent programs from running away. Once activated, it is necessary to clear the watchdog timer periodically within a specified period of time. If the watchdog timer is not cleared within a specified period of time, for example, because the program falls into an infinite loop, a watchdog reset of 4 instruction cycles is generated to the CPU.

The timebase timer output or clock prescaler output can be selected as the count clock of the watchdog timer.

Table 6.1-1 "Watchdog Timer Interval Time" lists the interval time of the watchdog timer. Provided the watchdog timer is not cleared, a watchdog reset is generated between the minimum and maximum times. Clear the counter within the minimum time in this table.

Table 6.1-1 Watchdog Timer Interval Time

<table>
<thead>
<tr>
<th>Count clock</th>
<th>Timebase timer output (when the main clock source oscillation frequency is 8 MHz)</th>
<th>Clock prescaler output (when the Sub-clock source oscillation frequency is 32.768 KHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum time</td>
<td>About 524.3 ms&lt;sup&gt;1&lt;/sup&gt;</td>
<td>500 ms&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>Maximum time</td>
<td>About 1048.6 ms</td>
<td>1000 ms</td>
</tr>
</tbody>
</table>

*1: Main clock source oscillation (F<sub>CH</sub>) divided by 2 x count of the timebase timer (2<sup>21</sup>)

*2: Sub-clock source oscillation (F<sub>CL</sub>) cycle x count of the clock prescaler (2<sup>14</sup>)

For minimum and maximum times of the watchdog timer interval time, see Section 6.4 "Explanation of Operations of the Watchdog Timer."

Note:

If the timebase timer output is selected as the count clock, the counter of the watchdog timer is cleared simultaneously when the timebase timer is cleared (TBTC: TBR=0). If the clock prescaler is selected as the count clock, the counter of the watchdog timer is cleared simultaneously when the clock prescaler is cleared (WPCR: WCLR=0). Thus, the watchdog timer cannot function if the counter (timebase timer or clock prescaler) used as the count clock is repeatedly cleared within the interval time of the watchdog timer.

Reference:

When the sleep mode, stop mode, or clock mode is entered, the counter of the watchdog timer is cleared and will not operate before returning to normal operation (running state).
6.2 Configuration of the Watchdog Timer

The watchdog timer consists of the following six parts:
- Count clock selector
- Watchdog timer counter
- Reset control circuit
- Watchdog timer clear selector
- Counter clear control circuit
- Watchdog control register (WDTC)

Block Diagram of the Watchdog Timer

![Block Diagram of the Watchdog Timer](image)

- **Count clock selector**
  Selects the count clock of the watchdog timer counter. The timebase timer output or clock prescaler output can be selected as the count clock.

- **Watchdog timer counter**
  1-bit counter with the timebase timer output or clock prescaler output as its count clock.

- **Reset control circuit**
  A reset signal to the CPU is generated by an overflow of the watchdog timer counter.
CHAPTER 6 WATCHDOG TIMER

- **Watchdog timer clear selector**
  Selects the watchdog timer clear signal from the timebase timer or clock prescaler simultaneously with the count clock selector.

- **Counter clear control circuit**
  Controls the clearing of the watchdog timer counter and the termination of its operation.

- **Watchdog control register (WDTC)**
  Performs the count clock selection and activation/clearing of the watchdog timer counter. Since this register is write-only, bit processing commands cannot be used.
6.3 Watchdog Control Register (WDTC)

The watchdog control register (WDTC) is a register to activate/clear the watchdog timer.

- **Watchdog Control Register (WDTC)**

![Watchdog Control Register (WDTC) Diagram]

**Figure 6.3-1** Watchdog Control Register (WDTC)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 9h</td>
<td>CS</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WTE3</td>
<td>WTE2</td>
<td>WTE1</td>
<td>WTE0</td>
</tr>
</tbody>
</table>

| CS | Output cycle of the timebase timer (2^{22}/F_CH) |
| WTE3 | - Activates the watchdog timer (when writing for the first time after a reset) |
| WTE2 | - Clears the watchdog timer (when writing for the second time or after a reset) |
| WTE1 | Otherwise |
| WTE0 | No operation |

<table>
<thead>
<tr>
<th>WTE3</th>
<th>WTE2</th>
<th>WTE1</th>
<th>WTE0</th>
<th>Watchdog control bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>- Activates the watchdog timer (when writing for the first time after a reset)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Clears the watchdog timer (when writing for the second time or after a reset)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CS</th>
<th>Counter clock switch bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Output cycle of the timebase timer (2^{14}/F_CL)</td>
</tr>
<tr>
<td>1</td>
<td>Output cycle of the clock prescaler (2^{14}/F_CL)</td>
</tr>
</tbody>
</table>

- R/W: Read/write enabled
- W: Write-only
- Unused
- Undefined
- Initial value

*1: Since this register is write-only, bit processing commands cannot be used.

F_CH: Main clock source oscillation
F_CL: Sub-clock source oscillation
# Watchdog Timer

## Table 6.3-1 Functional Explanation of Each Bit of the Watchdog Control Register (WDTC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Function</th>
</tr>
</thead>
</table>
| Bit 7 | **CS:**  
Count clock switch bit  
 Selects the count clock of the watchdog timer when the watchdog timer is activated. The timebase timer output or clock prescaler output can be selected as the count clock.  
**Note:**  
When using the sub-clock mode, be sure to select the clock prescaler output. Select the count clock simultaneously with activation of the watchdog timer and do not change the selection after the activation. Bit processing commands cannot be used. |
| Bit 6, Bit 5, Bit 4 | Unused  
• Value when these bits are read is not defined.  
• Writing to these bits does not affect operations. |
| Bit 3, Bit 2, Bit 1, Bit 0 | **WTE3, WTE2, WTE1, WTE0:**  
Watchdog control bit  
• Writing "0101B" activates the watchdog timer (first writing after a reset). Or, the watchdog timer is cleared (second or subsequent writing after a reset).  
• Operations are not affected if anything other than "0101B" is written.  
**Note:**  
"1111B" is read. Bit processing commands cannot be used. |
6.4 Explanation of Operations of the Watchdog Timer

A watchdog reset is generated by an overflow of the watchdog timer counter in the watchdog timer.

Operations of the Watchdog Timer

- **Activating the watchdog timer**
  - The watchdog timer can be activated by writing the first "0101B" to the watchdog control bit (WDTC: WTE3 to 0) of the watchdog control register after a reset. Specify, at this time, the count clock switch bit (WDTC: CS) simultaneously.
  - Once activated, the watchdog timer cannot be stopped except by a reset.

- **Clearing the watchdog timer**
  - The watchdog timer counter can be cleared by writing the second or subsequent "0101B" to the watchdog control bits (WDTC: WTE3 to 0) of the watchdog control register after a reset.
  - If the counter is not cleared within the interval time of the watchdog timer, an overflow of the counter occurs, thereby generating an internal reset signal of 4 instruction cycles.

- **Watchdog timer interval time**
  The interval time changes depending on the timing of clearing the watchdog timer. Figure 6.4-1 "Watchdog Timer Clearing and Interval Time" shows the relationship between the timing of clearing the watchdog timer and the interval time when the timebase timer (with the 8 MHz main clock source oscillation) output is selected as the count clock.
Figure 6.4-1  Watchdog Timer Clearing and Interval Time
6.5 Precautions when Using the Watchdog Timer

The following explains the precautions to be taken when using the watchdog timer.

- **Precautions when Using the Watchdog Timer**

  - **Stopping the watchdog timer**
    Once activated, the watchdog timer cannot be stopped until a reset occurs.

  - **Selecting the count clock**
    The count clock switch bit (WDTC: CS) can be rewritten only if "0101B" is set to the watchdog control bits (WDTC: WTE3 to 0) when the watchdog timer is activated. Thus, bit processing commands cannot be used for writing. Do not change the setting after the activation.
    The timebase timer does not operate in sub-click mode because the main clock source oscillation is stopped.
    It is necessary to select (WDTC: CS=1) the clock prescaler as the count clock so that the watchdog timer can operate in sub-clock mode.

  - **Clearing the watchdog timer**
    - If the counter (timebase timer or clock prescaler) used as the count clock of the watchdog timer is cleared, the counter of the watchdog timer is cleared simultaneously.
    - When the sleep mode, stop mode, or clock mode is entered, the counter of the watchdog timer is cleared.

  - **Precautions when creating a program**
    When creating a program in which the watchdog timer is cleared repeatedly in the main loop, the processing time of the main loop including interrupt processing must be equal to or shorter than the minimum time of the watchdog timer interval time.

  - **Operations in sub-clock mode**
    If a watchdog reset occurs in sub-clock mode, the operation is started in main clock mode after taking the oscillation stabilization wait time. Thus, if "Reset output" is selected in the option settings, the reset signal is output during the oscillation stabilization wait time.
CHAPTER 6 WATCHDOG TIMER

6.6 Program Example of the Watchdog Timer

The following shows a program example in which the watchdog timer is used.

- Program Example of the Watchdog Timer

  o Processing specifications

  - Select the clock prescaler as the count clock just after starting the program and then activate the watchdog timer.
  - Clear the watchdog timer each time in a loop of the main program.
  - The main loop must make a round including interrupt processing shorter than or equal to the minimum interval time (about 524.3 ms/operating at 8 MHz) of the watchdog timer.

  o Example of coding

```assembly
WDTC EQU 00009H ; Address of the watchdog control register
WDT_CLR EQU 10000101B

VECT DSEG ABS ; [DATA SEGMENT]
ORG 0FFFEH
RST_V DW PROG ; Reset vector setting

;-----Main program-------------------------------------------------------------------------------------------------------------
CSEG ; [CODE SEGMENT]

PROG ; Initialization routine for a reset
MOVW SP, #0280H ; Initialize the stack pointer (for interrupt processing)

; Initialize peripheral functions (interrupts)

INIT MOV WDTC, #WDT_CLR ; Activate the watchdog timer

; Select the clock prescaler as the count clock

MAIN MOV WDTC, #WDT_CLR ; Clear the watchdog timer

; User processing (Interrupt processing may occur in between)

JMP MAIN ; Loop processing must be shorter than the minimum interval time of the watchdog timer

ENDS
```

END
CHAPTER 7  8-BIT PWM TIMER

This chapter describes the functions and operations of the 8-bit PWM timer.

7.1 "Overview of 8-bit PWM Timer"
7.2 "8-bit PWM Timer Configuration"
7.3 "Pin of 8-bit PWM Timer"
7.4 "Registers of the 8-bit PWM Timer"
7.5 "8-bit PWM Timer Interrupts"
7.6 "Operation of Interval Timer Function"
7.7 "Operation of PWM Timer Function"
7.8 "State in Each Mode during 8-bit PWM Timer Operation"
7.9 "Precaution on Use of 8-bit PWM Timer"
7.10 "8-bit PWM Timer Program Examples"
7.1 Overview of 8-bit PWM Timer

The 8-bit PWM timer has two functions: an interval timer function that performs count operation in synchronization with four types of internal count clock and a PWM timer function of eight-bit resolution. Either can be selected. With these functions, the 8-bit interval timer time can be set. In addition, square waves of any frequency can be output by using its output. It can also be used as a D/A converter by connecting the PWM output to a low-pass filter.

■ Interval Timer Function (Square Wave Output Function)

The interval timer function generates interrupts repeatedly at arbitrary time intervals. Since the output level of the pin (PWM pin) can be inverted each time an interrupt occurs, square waves of any frequency can be output.

- Interval timer operation ranging from the count clock cycle to a cycle $2^8$ times greater can be performed.
- There are options of four types of count clock.

Table 7.1-1 "Interval Time and Square Wave Output Range" shows the interval time and square wave output range.

### Table 7.1-1 Interval Time and Square Wave Output Range

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>Interval time</th>
<th>Square wave output (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal count</td>
<td>$t_{\text{inst}}$</td>
<td>$t_{\text{inst}}$ to $2^8 t_{\text{inst}}$</td>
</tr>
<tr>
<td>16 $t_{\text{inst}}$</td>
<td>$2^4 t_{\text{inst}}$ to $2^{12} t_{\text{inst}}$</td>
<td>$1/(2^5 t_{\text{inst}})$ to $1/(2^{13} t_{\text{inst}})$</td>
</tr>
<tr>
<td>64 $t_{\text{inst}}$</td>
<td>$2^8 t_{\text{inst}}$ to $2^{14} t_{\text{inst}}$</td>
<td>$1/(2^7 t_{\text{inst}})$ to $1/(2^{15} t_{\text{inst}})$</td>
</tr>
<tr>
<td>256 $t_{\text{inst}}$</td>
<td>$2^8 t_{\text{inst}}$ to $2^{16} t_{\text{inst}}$</td>
<td>$1/(2^9 t_{\text{inst}})$ to $1/(2^{17} t_{\text{inst}})$</td>
</tr>
</tbody>
</table>

$t_{\text{inst}}$: Instruction cycle (influenced by clock mode)

Reference:

Calculation example of interval time and square wave frequency

The interval time obtained when the count clock cycle is set to $1 t_{\text{inst}}$ ($4/F_{\text{CH}}$) at a main clock source oscillation frequency ($F_{\text{CH}}$) of 8 MHz and a PWM compare register (COMR) value of “$\text{DDH (221)}$” is calculated as shown below. The frequency of the square wave output from the PWM pin when continuous operation is performed without changing the COMR register value is also calculated as shown below.

Note that it is the value when the highest speed ($\text{CS1, CS0 = 11B, 1 instruction cycle = } 4/F_{\text{CH}}$) in main clock mode ($\text{SCS = 1}$) is selected by the system clock control register (SYCC).
7.1 Overview of 8-bit PWM Timer

The PWM timer function has a resolution of eight bits and can control the "H" and "L" widths in one cycle.

- Since the resolution is 1/256, pulses can be output at a duty ratio of 0 to 99.6%.
- Four PWM wave cycle times can be selected.
- It can also be used as a D/A converter by connecting the PWM output to a low-pass filter.

Table 7.1-2 "PWM Wave Cycles That Can Set with the PWM Timer Function" shows the PWM wave cycles that can be set using the PWM timer function. Figure 7.1-1 "Example of D/A Converter Configuration with PWM Output and Low-pass Filter" shows an example of the D/A converter configuration.

Table 7.1-2 PWM Wave Cycles That Can Set with the PWM Timer Function

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>Internal count clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ( t_{\text{inst}} )</td>
<td>16 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td>PWM wave frequency</td>
<td>( 2^8 t_{\text{inst}} )</td>
</tr>
</tbody>
</table>

\( t_{\text{inst}} \): Instruction cycle (influenced by clock mode)
Figure 7.1-1 Example of D/A Converter Configuration with PWM Output and Low-pass Filter

Reference:

In PWM function operation, no interrupt request occurs.
7.2 8-bit PWM Timer Configuration

The 8-bit PWM timer consists of the following six blocks:
- Count clock selector
- 8-bit counter
- Compare circuit
- PWM generator circuit and output control circuit
- PWM compare register (COMR)
- PWM control register (CNTR)

---

Block Diagram of the 8-bit PWM Timer

Figure 7.2-1 Block Diagram of the 8-bit PWM Timer
CHAPTER 7 8-BIT PWM TIMER

- **Count clock selector**
  A circuit for selecting the count-up clock of the 8-bit counter.

- **8-bit counter**
  The 8-bit counter counts up with the count clock selected by the count clock selector.

- **Compare circuit**
  The compare circuit has a latch for holding the COMR register value. The latching the COMR register value is performed when the 8-bit counter value is "00H". The compare circuit also compares the value in the 8-bit counter to the value in the latched COMR register to detect a match.

- **PWM generator circuit and output control circuit**
  In interval timer operation, an interrupt request is generated when a match is detected. When the output pin control bit (CNRT: OE) is "1", the output control circuit inverts the output level of the PWM pin. At this time, the 8-bit counter is cleared.

  In PWM timer operation, the PWM generator circuit changes the output level of the PWM pin from "H" to "L" when a match is detected, thereafter the output level of the PWM pin returns to "H" when the eight-bit counter overflows.

- **PWM compare register (COMR)**
  The PWM compare register has the value to be compared to the counter value in the 8-bit counter.

- **PWM control register (CNTR)**
  The PWM control register selects the operation mode, enables/disables operation, selects the count clock, controls interrupts, and checks the state.

  When the operation mode is set to PWM timer mode (P/TX = 0), clearing the 8-bit counter and interrupt request (IRQ7) because of a match detection signal from the compare circuit are prohibited.

- **PWM-related interrupts**
  **IRQ7:**
  An interrupt request is generated if the interrupt request output is enabled (CNTR:TIE = 1) when the counter value matches the value set in the COMR register in interval timer function mode. (No interrupt request is generated in PWM function mode.)
This section describes the 8-bit PWM timer related pin and provides a block diagram of the pin.

■ Pin Related to the 8-bit PWM Timer

The pin related to the 8-bit PWM timer is the P30/PWM pin.

○ P30/PWM pin

This pin serves as a general-purpose I/O port (P30) and interval timer or PWM timer output (PWM).

PWM:

In interval timer function mode, a square wave is output to this pin.

In PWM timer function mode, a PWM wave is output to this pin.

When the output pin control bit is set to the dedicated pin (CNTR: OE = 1), the P30/PWM pin is set to an output pin automatically regardless of the value of the port direction register (DDR3: bit0) and functions as a PWM pin.
Block Diagram of the Pin Related to the 8-bit PWM Timer

Figure 7.3-1  Block Diagram of the Pin Related to the 8-bit PWM Timer

Reference:
When "With pull-up resistor" is selected in optional setting, the pin state at the time of reset and in stop or watch mode (SPL = 1) is "H."
7.4 Registers of the 8-bit PWM Timer

This section describes the registers related to the 8-bit PWM timer.

- Registers Related to the 8-bit PWM Timer

**Figure 7.4-1 Registers Related to the 8-bit PWM Timer**

<table>
<thead>
<tr>
<th>CNTR (PWM control register)</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 E_H</td>
<td></td>
</tr>
<tr>
<td>R/W bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
<td>Initial value</td>
</tr>
<tr>
<td>P/TX P1 P0 TPE TIR OE TIE</td>
<td>0X000000B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMR (PWM compare register)</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 F_H</td>
<td></td>
</tr>
<tr>
<td>W W W W W W W W</td>
<td>Initial value</td>
</tr>
<tr>
<td>XXXXXXXXXb</td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

Since the PWM compare register (COMR) is write only, the bit manipulation instruction cannot be used.

CHAPTER 7  8-BIT PWM TIMER

7.4.1 PWM Control Register (CNTR)

The PWM control register (CNTR) selects the operation mode of the 8-bit PWM timer (interval timer and PWM timer operations and enables/disables operation, selects the count clock, controls interrupts, and checks the state.

- PWM Control Register

![PWM Control Register Diagram](image)
### Table 7.4-1 Functions of Each Bit in the PWM Control Register (CNTR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 7 P/TX: Operation mode selection bit | A bit for switching between the interval timer function (P/TX = 0) and the PWM timer function (P/TX = 1).  
**Note:** Before writing into this bit, stop the counter operation (TPE = 0), disable interrupts (TIE = 0), and clear the interrupt request flag bit (TIR = 0). |
| Bit 6 Unused bit          | • The value in read cycle is undefined.  
• In write cycle, this bit has no influence on operation. |
| Bit 5 P1, P0: Clock selection bits | • Bits for selecting the count clock for the interval timer function or the PWM timer function.  
• There are options of four types of internal count clocks.  
**Note:** Do not switch P1 and P0 when the counter is operating (TPE=1). |
| Bit 4 TPE: Counter operation enable bit | • A bit for starting and stopping the interval timer function or the PWM timer function  
• Writing "1" into this bit starts a count operation, while writing "0" into this bit clears the counter to "00H" and stops the count operation. |
| Bit 3 TIR: Interrupt request flag bit | • In interval timer function mode  
This bit is set to "1" when the counter value matches the PWM compare register (COMR) value.  
When this bit or the interrupt request enable bit (TIE) is set to "1", an interrupt request to the CPU is output.  
• In PWM timer function mode, no interrupt request occurs.  
• In a write cycle, this bit is cleared when "0" is written and does not affect operation when "1" is written. |
| Bit 2 OE: Output pin control bit | • The P30/PWM pin serves as a general-purpose port (P30) when this bit is "0" and serves as a dedicated pin (PWM) when the bit is "1".  
• To the PWM pin, a square wave is output in interval timer function mode and a PWM wave is output in PWM timer function mode. |
| Bit 0 TIE: Interrupt request enable bit | A bit for enabling/disabling interrupt request output to the CPU. When this bit and the interrupt request flag bit (TIR) are set to "1", an interrupt request is output to the CPU. |
7.4.2  PWM Compare Register (COMR)

The PWM compare register (COMR) sets the interval time in interval timer function mode. In PWM timer function mode, this register value becomes the "H" width of the pulse.

- **PWM Compare Register (COMR)**

  Figure 7.4-3 "PWM Compare Register (COMR)" shows the bit configuration of the PWM compare register.

  Since this register is write only, bit manipulation instructions cannot be used.

- **Interval timer mode**

  The PWM compare register sets the value to be compared to the counter value. Specify the interval time therein.

  When the value written into the PWM compare register matches the counter value, the counter is cleared and the interrupt request flag bit is set to "1" (CNTR: TIR = 1).

  If a value is written into the COMR register during the counter operation, the value is effective from the next cycle (after a match is detected).

  **Reference:**

  The value set in the COMR register in interval timer operation mode can be calculated using the following expression. Note that the instruction cycle is influenced by the clock mode and gear function.

  \[
  \text{COMR register value} = \frac{\text{interval time}}{(\text{count clock cycle} \times \text{instruction cycle}) - 1}
  \]

- **PWM timer mode**

  The PWM compare register sets the value to be compared to the counter value. Specify the "H" width of the pulse.

  The PWM pin continues to output "H" until the value written into the PWM compare register matches the counter value. After the matching, the PWM pin continues to output "L" until the counter value overflows.

  When a value is written into the COMR register when the counter is operating, the value is effective from the next cycle (after overflow).
Reference:

The value set in the COMR register in PWM timer operation mode can be calculated using the following expression. Note that the instruction cycle is influenced by the clock mode and gear function.

\[
\text{COMR register value} = \text{duty ratio (\%)} \times 256
\]

\[
\text{PWM wave cycle} = \text{count clock cycle} \times \text{instruction cycle} \times 256
\]
7.5 8-bit PWM Timer Interrupts

The interrupt source of the 8-bit PWM timer is a match of the counter value and the PWM compare register value in interval timer function mode. In PWM timer function mode, no interrupt request occurs.

Interrupts in Interval Timer Function Mode

The selected count clock counts up from the counter value "00H" until the count value matches the PWM compare register (COMR) value. When a match occurs, the interrupt request flag bit (CNRT: TIR) is set to "1".

If the interrupt request enable bit is set to "1" (CNTR: TIE = 1) at this time, an interrupt request (IRQ7) to the CPU occurs. Write "0" into the TIR bit in the interrupt processing routine to clear the interrupt request.

The TIR bit is set to "1" when the counter value matches the set value, regardless of the value of the TIE bit.

Reference:

A match of the counter value and the COMR register value occurs concurrently with the stop of the counter (CNRT: TPR = 0), the TIR bit is not set.

If the TIE bit is changed from "0" to "1" ("disable" --> "enable") when the TIR bit is "1", an interrupt request occurs immediately.

Register and Vector Table Related to 8-bit PWM Timer Interrupts

Table 7.5-1 Register and Vector Table Related to 8-bit PWM Timer Interrupts

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ7</td>
<td>ILR2 (007DH)</td>
<td>L71 (bit7)</td>
</tr>
</tbody>
</table>

For interrupt operation, see Section 3.4.2 "Flow of Interrupt Operation."
7.6 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the 8-bit PWM timer.

- Operation of the Interval Timer Function

  To operate the interval timer function, the setting shown in Figure 7.6-1 "Interval Timer Function Setting is required."

  \[
  \text{Figure 7.6-1 Interval Timer Function Setting}
  \]

  When the counter is started, the counter counts up from "00 H" at the rising edge of a clock pulse of the selected count clock. When the counter value matches the value set in the COMR register (comparison value), the level of the output pin (PWM pin) is inverted at the next rising edge of the count clock to clear the counter and the interrupt request flag bit is set to "1" (CNTR: TIR = 1). The counter then counts up from "00H" again.

  Figure 7.6-2 "Operation of the 8-bit PWM Timer" shows the operation of the 8-bit PWM timer.
**Figure 7.6-2 Operation of the 8-bit PWM Timer**

**Note:**
During the operation of the interval timer function (CNTR: TPE = 1), do not change the count clock cycle (CNTR: P1, P0).

**Reference:**
When the COMR register is set to "00_{H}\), the output of the 8-bit PWM timer is inverted in the count clock cycle selected.

In interval timer function mode, the output of the 8-bit PWM timer in the counter stop state (CNTR: TPE = 0) is "L" level.

*If the value in the COMR register changes when the counter is operating, the value is effective from the next cycle.*
7.7 Operation of PWM Timer Function

This section describes the operation of the PWM timer function of the 8-bit PWM timer.

### Operation of the PWM Timer Function

To operate the PWM timer function, the setting shown in Figure 7.7-1 "PWM Timer Function Setting" is required.

![PWM Timer Function Setting](image)

When the counter is started, the counter counts up from "00 H" at the rising edge of a clock pulse of the selected count clock. The PWM pin output (PWM waveform) is kept at "H" level until the counter value matches the value set in the COMR register and is kept at "L" level until the counter value overflows (FFH → 00H).

Figure 7.7-2 "Example of PWM Waveform (PWM Pin) Output" shows the PWM waveform output to the PWM pin.

![Example of PWM Waveform (PWM Pin) Output](image)

**Note:**

During the operation of the PWM timer function (CNTR: TPE = 1), do not change the count clock cycle (CNTR: P1, P0).
Reference:

In PWM timer function mode, the output of the PWM pin in the counter stop state (CNRT: TPE=0) maintains the level immediately before the stop.
7.8 State in Each Mode during 8-bit PWM Timer Operation

This section describes the operation when transition to sleep or stop mode occurs and a stop request is issued during operation of the 8-bit PWM timer.

■ Operation in Standby Mode and Stop during Operation

When transition to sleep mode occurs, transition to stop mode occurs and a stop request is issued during the operation in interval timer function mode and PWM timer function mode. The states of the counter values are as shown in Figure 7.8-1 "Operation of Counter in Standby Mode and Interruption Mode (Interval Functioning Mode)" and Figure 7.8-2 "Operation of Counter in Standby Mode and Interruption Mode (PWM Timer Functioning Mode)"

When transition to stop mode occurs, the counter stops with its value retained. When stop mode is released by an external interrupt, the counter restarts operation from the value at the time of stop. Therefore, the initial interval time and the initial cycle of the PWM wave are not the set value. Initialize the 8-bit PWM timer again when the stop mode is released.

Transition to watch mode (STBC: TMD = 1) and release of watch mode operate in the same manner as transition to stop mode and release of stop mode. Watch mode is released by a watch interrupt and an external interrupt.
CHAPTER 7  8-BIT PWM TIMER

- Interval timer functioning mode

Figure 7.8-1  Operation of Counter in Standby Mode and Interruption Mode (Interval Functioning Mode)

* If the pin state specification bit (STBC: SPL) of the standby control register is set to "1" and the PWM pin is not pulled up (optional), the PWM pin in stop and watch modes is set to high-impedance. If the SPL bit is "0", the value immediately before transition to stop or watch mode is retained.
○ PWM timer functioning mode

Figure 7.8-2 Operation of Counter in Standby Mode and Interruption Mode (PWM Timer Functioning Mode)

* If the pin state specification bit (STBC: SPL) of the standby control register is set to "1" and the PWM pin is not pulled up (optional), the PWM pin in stop and watch modes is set to high-impedance. If the SPL bit is set to "0", the value immediately before transition to stop or watch mode is retained.
7.9 Precaution on Use of 8-bit PWM Timer

This section describes precautions when using the 8-bit PWM timer.

- Precaution on Use of 8-bit PWM Timer

  ○ Error
  
  Since the start of the counter by the program and that of count operation with the selected count clock are asynchronous, there may be an error before the detection of a match of the counter value and the PWM compare register (COMR) by up to one count clock cycle. Figure 7.9-1 "Error before the start of a count operation" shows the error before the start of the count operation.

  Figure 7.9-1  Error before the start of a count operation

  ○ Precautions on setting in a program

  - During the operation of the interval timer function and the PWM timer function (CNTR: TPE = 1), do not change the count clock cycle (CNTR: P1, P0).
  
  - Switching between the interval timer function and the PWM timer function (CNTR: P/TX) can only be made in counter stop (CNTR: TPE = 0), interrupt disabled (TIE = 0), and interrupt request clear (TIR = 0) states.
  
  - Restoration from interrupt processing cannot be made in a state in which the interrupt request flag bit (CNTR: TIR) is set to "1" and the interrupt request enable bit is set to "1" (CNTR: TIE = 1). The TIR bit must be cleared.
  
  - A match of the counter value and the COMR register value occurs concurrently with the stop of the counter (TPE = 0), and the TIR bit is not set.

  ○ Depending on the setting method of TPE, P/TX, and OE, the PWM output waveform differs as shown below. When making a setting in a program, note the following.

  (1) When TPE, P/TX, and OE are set simultaneously
When OE is set after TPE and P/TX are set:

```assembly
mov CNTR, #11001010B ; PWM operation, internal clock, start of count operation
; Enable PWM output

mov CNTR, #11001000B ; PWM operation, internal clock, start of count operation
Check
; Use a general-purpose port
mov CNTR, #11001010B ; Enable PWM output
Check
```

1/4 instruction cycle

PWM output enable instruction executed
7.10 8-bit PWM Timer Program Examples

Program examples of the 8-bit PWM timer are shown below.

- A Program Example in Interval Timer Function Mode

  - **Processing specifications**
    - Interval timer interrupts of 1 ms are generated repeatedly.
    - The square wave output inverted at the interval time is output to the PWM pin.
    - The COMR register value in which the interval time becomes about 1 ms when the main clock source oscillation frequency is 8 MHz is shown below. The 16 $t_{\text{inst}}$ ($t_{\text{inst}}$ : the main clock source oscillation frequency at the gear highest speed divided by four) internal count clock is used in the following example.

    \[
    \text{COMR register value} = \frac{1 \text{ ms}}{16 \times 4/8 \text{ MHz}} - 1 = 77.1 \ (04DH)
    \]
Coding example

CNTR EQU 001EH ; Address of PWM control register
COMR EQU 001FH ; Address of PWM compare register
TPE EQU CNTR:3 ; Definition of counter operation enable bit
TIR EQU CNTR:2 ; Definition of interrupt request flag bit
ILR2 EQU 007DH ; Address of interrupt level setting register 2
INT_V DSEG ABS ; [DATA SEGMENT]
ORG FFECH
IRQ7 DW WARI ; Set interrupt vector
INT_V ENDS

;-----Main program-------------------------------------------------------------

CSEG ; [CODE SEGMENT]
; Assuming that the stack pointer (SP) has been initialized

CLRI ; Interrupt disable
CLRB TPE ; Stop counter operation
MOV ILR2,#01111111B ; Set interrupt level (level 1)
MOV COMR,#04DH ; Value compared with counter value (interval time)
MOV CNTR,#00011011B ; Interval timer operation, Select 16 tInst,
; Start counter operation, clear interrupt request flag, enable PWM pin output, and enable interrupt request output

SETI ; Interrupt enable

;-----Interrupt program-------------------------------------------------------

WARI CLRB TIR ; Clear interrupt request flag
PUSHW A
XCHW A,T ; Save A and T
PUSHW A

User processing

; POPW A
XCHW A,T ; Restore A and T
POPW A
RETI
ENDS

;-----------------------------------------------------
END
A Program Example in PWM Timer Function Mode

○ Processing specifications

- A PWM wave with a duty ratio of 50% is generated and changed to 25%.
- No interrupt occurs.
- When the 16 t_{inst} (t_{inst}: the main clock source oscillation frequency at the gear highest speed divided by four) internal count clock is used at a main clock source oscillation frequency of 8 MHz, the cycle of the PWM wave is 16 x 4/8 MHz x 256 = 2.048 ms.
- The COMR register value with a duty ratio of 50% is shown below.
  COMR register value = 50/100 x 256 = 128 (080H)

○ Coding example

```assembly
CNTR EQU 001EH ; Address of PWM control register
COMR EQU 001FH ; Address of PWM compare register
TPE EQU CNTR:3 ; Definition of counter operation enable bit
CSEG ; [CODE SEGMENT]

CLRB TPE ; Stop counter operation
MOV COMR,#80H ; Specify the "H" width of the pulse. Duty ratio: 50%
MOV CNTR,#10011010B ; PWM timer operation, select 16 t_{inst},
                      ; start counter operation, clear interrupt request
                      ; flag, enable PWM pin output, disable interrupt
                      ; request output

MOV COMR,#40H ; Change duty ratio to 25% (effective from
               ; the next cycle of PWM wave)

ENDS
```

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

8.1 "Overview of 8/16-bit Timer/Counter"
8.2 "8/16-bit Timer/Counter Configuration"
8.3 "Pins of 8/16-bit Timer/Counter"
8.4 "Registers of 8/16-bit Timer/Counter"
8.5 "8/16-bit Timer/Counter Interrupts"
8.6 "Operation of Interval Timer Function"
8.7 "Operation of Counter Function"
8.8 "Operation of Square Wave Output Initial Setting Function"
8.9 "Stop and Restart Operation of 8/16-bit Timer/Counter"
8.10 "State in Each Mode during 8/16-bit Timer/Counter Operation"
8.11 "Precautions on Use of 8/16-bit Timer/Counter"
8.12 "8/16-bit Timer/Counter Program Examples"
8.1 Overview of 8/16-bit Timer/Counter

The 8/16-bit timer/counter consists of two 8-bit counters (Timer 1, Timer 2) that can be used independently (8-bit clock mode) or combined (16-bit mode). For Timer 1, either the interval timer function that counts up in synchronization with three types of internal count clock pulse or the counter function that counts up by externally input clock pulses are selectable. By using its output, square waves of any frequency can be output. Timer 2 has only an interval timer function that counts up 0 synchronously with three types of internal clock pulse. Though any frequency of square waves can be output, it is connected to Timer 1 in 16-bit mode.

- **Interval Timer Function**

  The interval timer function generates interrupt requests repeatedly at arbitrary time intervals.

  In addition, by inverting the output level of the pins (TMO1, TMO2 pins) at each time interval, square waves of any frequency can be output.

  - In 8-bit mode, Timer 1 and Timer 2 operate independently and can both operate in the range from the count clock cycle to a cycle $2^8$ times greater. In addition, square waves can be output using the TMO1 pin as Timer 1 output and the TMO2 pin as Timer 2 output.

  - In 16-bit mode, Timer 1 as a low timer and Timer 2 as a high timer connected each other and operate as an interval timer whose operation ranges from the count clock cycle to a cycle $2^{16}$ times greater. In addition, square waves can be output using the TMO1 pin as timer output.

  - There are three optional types of internal count clock. (An external clock selected in Timer 1 serves as a counter function.)

  Table 8.1-1 "Timer 1 Interval Time and Square Wave Output Range in 8-bit Mode" to Table 8.1-3 "Interval Time and Square Wave Output Range in 16-bit Mode" show the interval Time and square wave output range in each mode.

<table>
<thead>
<tr>
<th></th>
<th>Count clock cycle</th>
<th>Interval time</th>
<th>Square wave output range (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal count clock</td>
<td>$2 \ t_{inst}$</td>
<td>$2 \ t_{inst}$ to $2^8 \ t_{inst}$</td>
<td>$1/(2^2 \ t_{inst})$ to $1/(2^{10} \ t_{inst})$</td>
</tr>
<tr>
<td></td>
<td>$32 \ t_{inst}$</td>
<td>$2^5 \ t_{inst}$ to $2^{13} \ t_{inst}$</td>
<td>$1/(2^6 \ t_{inst})$ to $1/(2^{14} \ t_{inst})$</td>
</tr>
<tr>
<td></td>
<td>$128 \ t_{inst}$</td>
<td>$2^7 \ t_{inst}$ to $2^{15} \ t_{inst}$</td>
<td>$1/(2^8 \ t_{inst})$ to $1/(2^{16} \ t_{inst})$</td>
</tr>
<tr>
<td>External clock</td>
<td>$1 \ t_{ext}$</td>
<td>$1 \ t_{ext}$ to $2^8 \ t_{ext}$</td>
<td>$1/(2 \ t_{ext})$ to $1/(2^8 \ t_{ext})$</td>
</tr>
</tbody>
</table>
8.1 Overview of 8/16-bit Timer/Counter

Reference:
Calculation example of interval time and square wave output frequency

When the count clock cycle is set to 2 \( t_{\text{inst}} \) 8-bit mode operation with a main clock source oscillation frequency \( (F_{CH}) \) of 8 MHz and a timer 1 register (T1DR) value of "DDH (221)," the frequency of a square wave output from the TMO1 pin can be calculated as shown below when the timer is operated continuously without changing the interval time of Timer 1 and the T1DR register value.

Note that the following is the value when the clock of the highest speed in main clock mode \( (SCS = 1) \) is selected \( (CS1, CS0 = 11_B, 1 \text{ instruction cycle} = 4/F_{CH}) \) by the system clock control register (SYCC).

\[
\text{Interval time} \quad = (2 \times 4/F_{CH}) \times (T1DR \text{ register value} + 1) \\
\quad = (8/8 \text{ MHz}) \times (221 + 1) \\
\quad \approx 222 \mu s
\]

\[
\text{Output frequency} \quad = F_{CH}/(2 \times 8 \times (T1DR \text{ register value} + 1)) \\
\quad = 8 \text{ MHz}/(16 \times (221 + 1)) \\
\quad \approx 2.25 \text{ kHz}
\]

Table 8.1-2  Timer 2 Interval Time and Square Wave Output Range in 8-bit Mode

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>Interval time</th>
<th>Square wave output range (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal count clock</td>
<td>( 2 t_{\text{inst}} )</td>
<td>( 2 t_{\text{inst}} ) to ( 2^{9} t_{\text{inst}} )</td>
</tr>
<tr>
<td>32 ( t_{\text{inst}} )</td>
<td>( 2^{5} t_{\text{inst}} ) to ( 2^{13} t_{\text{inst}} )</td>
<td>( 1/(2^{6} t_{\text{inst}}) ) to ( 1/(2^{14} t_{\text{inst}}) )</td>
</tr>
<tr>
<td>128 ( t_{\text{inst}} )</td>
<td>( 2^{7} t_{\text{inst}} ) to ( 2^{15} t_{\text{inst}} )</td>
<td>( 1/(2^{8} t_{\text{inst}}) ) to ( 1/(2^{16} t_{\text{inst}}) )</td>
</tr>
</tbody>
</table>

Table 8.1-3  Interval Time and Square Wave Output Range in 16-bit Mode

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>Interval time</th>
<th>Square wave output range (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal count clock</td>
<td>( 2 t_{\text{inst}} )</td>
<td>( 2 t_{\text{inst}} ) to ( 2^{17} t_{\text{inst}} )</td>
</tr>
<tr>
<td>32 ( t_{\text{inst}} )</td>
<td>( 2^{5} t_{\text{inst}} ) to ( 2^{21} t_{\text{inst}} )</td>
<td>( 1/(2^{6} t_{\text{inst}}) ) to ( 1/(2^{22} t_{\text{inst}}) )</td>
</tr>
<tr>
<td>128 ( t_{\text{inst}} )</td>
<td>( 2^{7} t_{\text{inst}} ) to ( 2^{23} t_{\text{inst}} )</td>
<td>( 1/(2^{8} t_{\text{inst}}) ) to ( 1/(2^{24} t_{\text{inst}}) )</td>
</tr>
<tr>
<td>External clock</td>
<td>( 1 t_{\text{ext}} )</td>
<td>( 1 t_{\text{ext}} ) to ( 2^{16} t_{\text{ext}} )</td>
</tr>
</tbody>
</table>

\( t_{\text{inst}} \): Instruction cycle (influenced by clock mode)
\( t_{\text{ext}} \): External clock cycle \( (1 t_{\text{ext}} \) greater than or equal to \( 2 t_{\text{inst}}) \)
CHAPTER 8 8/16-BIT TIMER/COUNTER

■ Counter Function

The counter function counts the number of falling edges of external clock pulses input to the external pin (TCLK pin). Since only Timer 1 can select the external clock, the counter function operates with Timer 1 in 8-bit mode or with the 16-bit timer in 16-bit mode.

- The counter counts up using the external clock. When the counter value reaches the set value, an interrupt request is generated and the output levels of the TMO1 and TMO2 pins are inverted.

- Timer 1 in 8-bit mode can count up to \(2^8\).

- The 16-bit timer can count up to \(2^{16}\).

- By inputting external clock pulses of a constant cycle, it can be used in the same manner as the interval timer function.
8.2 8/16-bit Timer/Counter Configuration

The 8/16-bit timer/counter consists of the following five blocks:
- Count clock selectors 1, 2
- Counter circuits 1, 2
- Square wave output control circuits 1, 2
- Timer 1, 2 data registers (T1DR, T2DR)
- Timer 1, 2 control registers (T1CR, T2CR)

**Figure 8.2-1  Block Diagram of the 8/16-bit Timer/Counter**
CHAPTER 8  8/16-BIT TIMER/COUNTER

- **Count clock selectors 1, 2**
  
  A circuit for selecting the input clock. For Timer 1 in 8-bit mode and the 16-bit timer in 16-bit mode, one clock can be selected from three types of internal clock and one external clock. For Timer 2 in 8-bit mode, one clock can be selected from three types of internal clock.

- **Counter circuits 1, 2**

  Counter circuits 1 and 2 each consist of an 8-bit counter, comparator, comparator data latch, and data register (T1DR1 or T2DR).

  The 8-bit counter counts up with clock pulses of the selected count clock. This counter value and the value in the comparator data latch are compared by the comparator. When a match is detected, the counter is cleared and the value in the data register is set (loaded) to the comparator data latch.

  In 8-bit mode, counter circuits 1 and 2 operate independently as Timers 1 and 2. In 16-bit mode, the circuits are combined and operate as a 16-bit counter with counter circuit 1 as the lower eight bits and counter circuit 2 as the higher eight bits.

- **Square wave output control circuits 1, 2**

  For Timers 1 and 2 in 8-bit mode and the 16-bit timer in 16-bit mode, square wave output control circuits 1 and 2 generate an interrupt request when a match is detected by the comparator. If the square wave output is enabled at this time, the output of TMO1 and TMO2 pins are inverted by the output control circuit.

  The square wave output can be initialized to "L" or "H" level.

- **Timer 1, 2 data registers (T1DR, T2DR)**

  Timer 1, 2 data registers are used to set the data to be compared to each 8-bit counter in the write cycle. In the read cycle, the present counter value in each counter is read.

- **Timer 1, 2 control registers (T1CR, T2CR)**

  Timer 1, 2 control registers select the function, enable/disable operation, control interrupts, and check the state.

- **Interrupts of the 8/16-bit timer/counter**

  **IRQ6:**

  An interrupt request of IRQ6 occurs when the interrupt request output is enabled (T1CR: T1IE = 1 for Timer 1 in 8-bit mode or the 16-bit timer in 16-bit mode, T2CR: T21E = 1 for Timer 2 in 8-bit mode) when the counter value matches the value set in the data register in the internal timer function or counter function mode.
This section describes the 8/16-bit timer/counter related pins and provides a block diagram of these pins.

- **Pins Related to the 8/16-bit Timer/Counter**

  The pins related to the 8/16-bit timer/counter are the P62/TCLK pin and the P60/TMO1 and P61/TMO2 pins.

- **P62/TCLK pin**

  This pin serves as a general-purpose I/O port (P62) and an external clock pulse input pin of the timer (TCLK).

  **TCLK:**

  When the external clock pulse input (counter function) is selected for Timer 1 in 8-bit mode or for the 16-bit timer in 16-bit mode (T1CR: T1CS1, T1CS0 = 11B), the clock pulses input in this pin are counted. To use the P62/TCLK pin as the TCLK pin, set the input port in the port direction register (DDR6: bit 2 = 0).

- **P60/TMO1, P61/TMO2 pins**

  The P60/TMO1 and P61/TMO2 pins provide a general-purpose I/O port (P60, P61) and the square wave output pin of the timer (TMO1, TMO2).

  **TMO1:**

  This pin outputs a square wave when Timer 1 is used in 8-bit mode or the 16-bit timer is used in 16-bit mode. The P60/TMO1 pin functions as the TM01 pin and is automatically set to an output pin regardless of the value of the port direction register (DDR6: bit0) when square wave output is enabled (T1CR: T1OS1, T1OS0 = other than 00B).

  **TMO2:**

  This pin outputs a square wave when Timer 2 is used in 8-bit mode. The P61/TMO2 pin functions as the TM02 pin and is automatically set to an output pin regardless of the value of the port direction register (DDR6: bit1) when square wave output is enabled (T2CR: T2OS1, T2OS0 = other than 00B).
CHAPTER 8 8/16-BIT TIMER/COUNTER

Block Diagram of the Pins Related to the 8/16-bit Timer/Counter

Figure 8.3-1 Block Diagram of the Pins Related to the 8/16-bit Timer/Counter (P60, P61)

- To external clock input (P60 only)
- Stop, watch mode (SPL=1)
- Pull-up resistor (optional) Approx. 50 kΩ (5 V)

- Pch
- Nch

- P60/TMO1
- P61/TMO2

- Latch

- PDR (port data register)
  - PDR read
  - PDR read (at bit manipulation instruction)
  - PDR write

- DDR (Port direction register)
  - DDR read
  - DDR write

- Internal data bus

- DDR read (at bit manipulation instruction)

SPL: Pin state specification bit of standby control register (STBC)
Reference:

When "With pull-up resistor" is set in an optional setting, the pin state at the time of reset and in stop or watch mode (SPL = 1) is "H."
8.4 Registers of 8/16-bit Timer/Counter

This section describes the registers related to the 8/16-bit timer/counter.

### Registers Related to the 8/16-bit Timer/Counter

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 7H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X0000000b</td>
<td>T1CR (Timer 1 control register)</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>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 6H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X0000000b</td>
<td>T2CR (Timer 2 control register)</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>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 9H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXb</td>
<td>T1DR (Timer 1 data register)</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>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 8H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXb</td>
<td>T2DR (Timer 2 data register)</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>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/write enabled
X: Undefined
8.4 Registers of 8/16-bit Timer/Counter

8.4.1 Timer 1 Control Register (T1CR)

The timer 1 control register (T1CR) selects functions, enables/disables operation, controls interrupts, and checks the state when Timer 1 is used in 8-bit mode and the 16-bit timer is used in 16-bit mode. Even if only Timer 1 is used in 8-bit mode, it is necessary to initialize the timer 2 control register (T2CR).

- Timer 1 Control Register (T1CR)

Figure 8.4-2 Timer 1 Control Register (T1CR)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 7h</td>
<td>T1IE</td>
<td>T1OS1</td>
<td>T1OS0</td>
<td>T1CS1</td>
<td>T1CS0</td>
<td>T1STP</td>
<td>T1STR</td>
<td>X00000000e</td>
<td></td>
</tr>
</tbody>
</table>

- **T1STR**: Timer start bit
  - 0: Stop the counter operation
  - 1: Clear the counter and start operation

- **T1STP**: Timer stop bit
  - 0: Continue operation without clearing counter
  - 1: Stop the counter operation temporarily

- **T1CS1** and **T1CS0**: Count clock selection bit
  - 00: $2^{2\text{tinst}}$
  - 01: $32^{2\text{tinst}}$
  - 10: $128^{2\text{tinst}}$
  - 11: External clock

- **T1OS1** and **T1OS0**: Square wave output control bit
  - 00: This pin is used as a general-purpose port (P60)
  - 01: Set data to set the square wave output to "L"
  - 10: Set data to set the square wave output to "H"
  - 11: Output the level corresponding to the set level to the square wave output pin (TMO1)*

- **T1IE**: Interrupt request enable bit
  - 0: Disable output request output
  - 1: Enable output request output

- **T1IF**: Interrupt request flag bit
  - Read cycle: No counter matching
  - Write cycle: Clear this bit
  - 1: Counter matching: No change, does not affect other operations

---

* The square wave output pin outputs the level corresponding to the data set when T1STR is "0".
<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>T1IF:</strong> Interrupt request flag bit</td>
</tr>
</tbody>
</table>
|                   | • In 8-bit mode  
|                   |   This bit is set to "1" when the counter value of Timer 1 matches the value set in the timer 1 data register (T1DR) (comparator data latch).                                                                 |
|                   | • In 16-bit mode  
|                   |   This bit is set to "1" when the counter values of Timers 1 and 2 match the values set in the T1DR and T2DR registers, respectively.                                                                      |
|                   | • When this bit and the interrupt request enable bit (T1IE) are set to "1", an interrupt request is output.                                                                                               |
|                   | • In a write cycle, this bit is cleared when "0" is written. Writing "1" into this bit does not affect this bit.                                                                                             |
| Bit 6             | **T1IE:** Interrupt request enable bit                                                                                                                                                                     |
|                   | • A bit for enabling/disabling interrupt request output to the CPU.                                                                                                                                       |
|                   | • When this bit and the interrupt request flag bit (T1IF) are set to "1", an interrupt request is output.                                                                                                  |
| Bit 5, Bit 4      | **T1OS1, T1OS0:** Square wave output control bits                                                                                                                                                         |
|                   | • The P60/TMO1 pin serves as a general-purpose port (P60) when these bits are set to a value other than "11B" and serves as a square wave output pin (TMO1) when these bits are set to "11B".                                 |
|                   | • When "01B" or "10B" is written, the initialized data is set to the square wave output control circuit.                                                                                                    |
|                   | • When these bits are set to "11B" and the timer is in stop state (STR1 = 0), the TMO1 pin is set to a level corresponding to the initialized data.                                                          |
| Bit 3, Bit 2      | **T1CS1, T1CS0:** Clock source selection bits                                                                                                                                                             |
|                   | • Bits for selecting the count clock to be supplied to the counter.                                                                                                                                      |
|                   | • Select one count clock from three types of internal clock and one external clock.                                                                                                                      |
|                   | • When these bits are set to "11B", external clock pulses are input and the 8/16-bit timer/counter can operate as a counter function.                                                                    |
|                   | **Note:** When the external clock input is selected (T1CS1, T1CS0 = 11B), it is necessary to set the P62/TCLK pin to the input port.                                                                      |
| Bit 1             | **T1STP:** Timer stop bit                                                                                                                                                                                 |
|                   | • A bit for stopping the counter temporarily.                                                                                                                                                             |
|                   | • When "1" is written into this bit, the counter stops operation temporarily. When "0" is written in timer start state (T1STR = 1), the counter continues operation.                                             |
| Bit 0             | **T1STR:** Timer start bit                                                                                                                                                                                |
|                   | • A bit for starting and stopping the counter.                                                                                                                                                            |
|                   | • When this bit is changed from "0" to "1", the counter is cleared. If the timer is in the timer operation continuing state (T1STR = 0) at this time, the counter starts operation and counts up with the selected count clock. When "0" is written, the counter stops operation. |
|                   | • In 16-bit mode, both Timers 1 and 2 are cleared by starting the timer (T1STP = 0 --> 1).                                                                                                                |
Note:

When using only Timer 1 of the 8/16-bit timer/counter in 8-bit mode, set the count clock selection bit of the timer 2 control register (T2CR: T2CS1, T2CS0) to a value other than "11B" before use. Use of Timer 1 of the 8/16-bit timer/counter without setting the register may result in a malfunction.

To use the square wave output control bit, see Section 8.8 "Operation of Square wave Output Initial Setting Function."
8.4.2 Timer 2 Control Register (T2CR)

The timer 2 control register (T2CR) selects functions, enables/disables operation, controls interrupts, and checks the state of Timer 2 in 8-bit mode. When in 16-bit mode, control is made by the timer 1 control register (T1CR) but setting of the timer 2 control register (T2CR) is also required.

Timer 2 Control Register (T2CR)

Figure 8.4-3 Timer 2 Control Register (T2CR)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 6h</td>
<td>T2IF</td>
<td>T2IE</td>
<td>T2OS1</td>
<td>T2OS0</td>
<td>T2CS1</td>
<td>T2CS0</td>
<td>T2STP</td>
<td>T2STR</td>
<td>X00000000h</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
</tr>
</thead>
</table>

- **T2STR** Timer start bit
  - 0: Stop the counter operation
  - 1: Clear the counter and start operation

- **T2STP** Timer stop bit
  - 0: Continue operation without clearing the counter.
  - 1: Stop the counter operation temporarily

- **T2CS1** Count clock selection bits
  - 0 0: 2inst
  - 0 1: 32inst
  - 1 0: 128inst
  - 1 1: 16-bit mode

- **T2OS1** Unused bit
  - 0 0: This pin is used as a general-purpose port (P61)
  - 0 1: Set data to set the square wave output to "L"
  - 1 0: Set data to set the square wave output to "H"
  - 1 1: Output the level corresponding to the set level to the square wave output pin (TMO2)*.

- **T2IE** Interrupt request enable bit
  - 0: Disable output request output
  - 1: Enable output request output

- **T2IF** Interrupt request flag bit
  - 0: No counter matching
  - 1: Counter matching

R/W: Read/write enabled
X: Undefined
: Initial value

*: The square wave output pin is the level corresponding to the data set when T2STR is "0".
### Table 8.4-2 Functions of Each Bit in the Timer 2 Control Register (T2CR)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>T2IF: Interrupt request flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 7</strong></td>
<td>When the counter value of Timer 2 matches the value set in the timer 2 data register (T2DR) (comparator latch), this bit is set to &quot;1&quot;.</td>
</tr>
<tr>
<td></td>
<td>When this bit and the interrupt request enable bit (T2IE) are set to &quot;1&quot;, an interrupt request is output.</td>
</tr>
<tr>
<td></td>
<td>In write cycle, this bit is cleared when &quot;0&quot; is written. Writing &quot;1&quot; into this bit does not affect this bit.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>In 16-bit mode, the T1IF bit is valid and the T2IF bit does not affect the operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 6</th>
<th>T2IE: Interrupt request enable bit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 6</strong></td>
<td>A bit for enabling/disabling interrupt request output to the CPU.</td>
</tr>
<tr>
<td></td>
<td>When this bit and the interrupt request flag bit (T2IF) are set to &quot;1&quot;, an interrupt request is output.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>In 16-bit mode, the T1IF bit is valid and the T2IF bit does not affect the operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 5, Bit 4</th>
<th>T2OS1, T2OS0: Square wave output control bits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 5</strong></td>
<td>The P61/TMO2 pin serves as a general-purpose port (P61) when these bits are set to a value other than &quot;11B&quot; and serves as a square wave output pin (TMO2) when these bits are set to &quot;11B&quot;.</td>
</tr>
<tr>
<td><strong>Bit 4</strong></td>
<td>When &quot;01B&quot; or &quot;10B&quot; is written, the initialized data is set to the square wave output control circuit.</td>
</tr>
<tr>
<td></td>
<td>When these bits are set to &quot;11B&quot; and the timer is in stop state (STR1 = 0), the TMO2 pin is set to a level corresponding to the initialized data.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 3, Bit 2</th>
<th>T2CS1, T2CS0: Clock source selection bits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 3</strong></td>
<td>Bits for selecting the count clock to be supplied to the counter.</td>
</tr>
<tr>
<td><strong>Bit 2</strong></td>
<td>Select one count clock from three types of internal clock.</td>
</tr>
<tr>
<td></td>
<td>Setting to &quot;11B&quot; selects the 16-bit mode.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>In 16-bit mode, the T1CS1 and T1CS0 bits are valid and the T2CS1 and T2CS0 bits select 16-bit mode only.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 1</th>
<th>T2STP: Timer stop bit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 1</strong></td>
<td>A bit for stopping the counter temporarily.</td>
</tr>
<tr>
<td></td>
<td>When &quot;1&quot; is written into this bit, the counter stops operation temporarily. When &quot;0&quot; is written in timer start state (T2STR = 0), the counter continues operation.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>In 16-bit mode, the T1STP bit is valid and the T2STP bit does not affect the operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>T2STR: Timer start bit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 0</strong></td>
<td>A bit for starting and stopping the counter.</td>
</tr>
<tr>
<td></td>
<td>When this bit is changed from &quot;0&quot; to &quot;1&quot;, the counter is cleared. If the timer is in the timer operation continuing state (T2STR = 0) at this time, the counter starts operation and counts up with the selected count clock. When &quot;0&quot; is written, the counter stops operation.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>In 16-bit mode, the T1STR bit is valid and the T2STR bit does not affect the operation.</td>
</tr>
</tbody>
</table>

**Note:**

When using in 16-bit mode, write "11B" into the T2CS1 and T2CS0 bits before controlling with the T1CR register.
8.4.3 Timer 1 Data Register (T1DR)

The timer 1 data register (T1DR) sets the counter value (in counter function mode) and the interval timer value of Timer 1 in 8-bit mode and the lower eight bits in 16-bit mode (in interval timer function mode). This register is also used to read the counter value.

### Timer 1 Data Register (T1DR)

The value set in this register is compared to the counter value. When this register is read, the present counter value is read. The value set in the register cannot be read.

Figure 8.4-4 "Timer 1 Data Register (T1DR)" shows the bit configuration of the timer 1 data register.

![Figure 8.4-4 Timer 1 Data Register (T1DR)]

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 9H</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>
<td>XxxxxxxH</td>
</tr>
</tbody>
</table>

R/W: Read/write enabled
X: Undefined

### 8-bit mode (Timer 1)

The value set in this register is compared to the counter value of Timer 1. In interval timer function mode, this register is used to set the interval time and to set the count number to be detected in counter function mode. When the count operation is enabled (T1CR: T1STR = 0 --> 1, T1STP = 0), the value in the T1DR register is set (loaded) in the comparator data latch and the counter starts a count operation.

When the value in the comparator data latch and the counter value match as the result of a count operation, Timer 1 re-sets the value in the T1DR register in the comparator data latch, clears the counter, and continues the count operation.

The comparator data latch is re-set by the detection of a match. Therefore, if a value is written into the T1DR register when a counter is operating, the value is effective from the next cycle (when a match is detected).

**Reference:**

The value set in the T1DR register during interval timer operation can be calculated with the following expression. Note that the instruction cycle is affected by the clock mode and gear function.

\[ \text{T1DR register value} = \frac{\text{interval time}}{\text{(count clock cycle} \times \text{instruction cycle})} - 1 \]
16-bit mode

The value set in this register is compared to the counter value of the lower eight bits of the 16-bit timer. In interval timer function mode, this register is used to set the lower eight bits of the interval time and to set the lower eight bits of the count number to be detected in counter function mode. The T1DR register is loaded into the lower eight bits of the comparator data latch when a count operation is started and a match with the 16-bit counter value is detected.

If a value is written into the T1DR register when the 16-bit counter is operating, the value is effective when a match is detected.

For the value set in the T1DR register in interval timer function mode, see Section 8.4.4 "Timer 2 Data Register (T2DR)."
8.4.4 Timer 2 Data Register (T2DR)

The timer 2 data register (T2DR) sets the interval timer value of Timer 2 in 8-bit mode and the upper eight bits in 16-bit mode (in interval timer function mode) and also sets the counter value (in counter function mode). This register is also used to read the counter value.

- **Timer 2 Data Register (T2DR)**
  
  The value set in this register is compared to the counter value. When this register is read, the present counter value is read. The value set in the register cannot be read.

  Figure 8.4-5 "Timer 2 Data Register (T2DR)" shows the bit configuration of the timer 2 data register.

  **Figure 8.4-5 Timer 2 Data Register (T2DR)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 8H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXb</td>
</tr>
</tbody>
</table>

  R/W: Read/write enabled

  X: Undefined

- **8-bit mode (Timer 2)**
  
  The value set in this register is compared to the counter value of Timer 2. In interval timer function mode, this register is used to set the interval time and to set the count number to be detected in counter function mode. When the count operation is started and a match with the counter value is detected, the value in the T2DR register is re-set (loaded) in the comparator data latch.

  If a value is written into the T2DR register when the counter is operating, the value is effective from the next cycle (when a match is detected).

  **Reference:**

  The value set in the T2DR register during interval timer operation can be calculated with the following expression. Note that the instruction cycle is affected by the clock mode and gear function.

  \[ T2DR \text{ register value} = \frac{\text{interval time}}{\text{count clock cycle} \times \text{instruction cycle}} - 1 \]
16-bit mode

The value set in this register is compared to the counter value of the upper eight bits of the 16-bit timer. In interval timer function mode, this register is used to set the upper eight bits of the interval time and to set the upper eight bits of the count number to be detected in counter function mode. The T2DR register is loaded into the upper eight bits of the comparator data latch when a count operation is started and a match with the 16-bit counter value is detected.

If a value is written into the T2DR register when the 16-bit counter is operating, the value is effective when a match is detected. In 16-bit mode, the count operation is controlled by the timer 1 control register (T1CR).

Reference:

The values set in the T1DR and T2DR registers in interval function mode can be calculated with the following expression. Note that the instruction cycle is affected by the clock mode and gear function.

$$16 \text{ bit data value} = \frac{\text{interval time}}{\text{(count clock cycle} \times \text{instruction cycle})} - 1$$

The upper eight bits of the 16-bit data value are set in the T2DR register, while the lower eight bits of the 16-bit data value are set in the T1DR register.
8.5 8/16-bit Timer/Counter Interrupts

The interrupt source of the 8/16-bit timer/counter is a match of the value set in the data register and the counter value in both interval timer function mode and counter operation mode.

Interrupts of the 8/16-bit Timer/Counter

Table 8.5-1 "8/16-bit Timer/Counter Interrupt Control Bits and Interrupt Sources" shows the relation among the 8/16-bit timer/counter interrupt request flag bit, the interrupt request output enable bit, and the interrupt occurrence source.

Table 8.5-1 8/16-bit Timer/Counter Interrupt Control Bits and Interrupt Sources

<table>
<thead>
<tr>
<th>Interrupt request flag bit</th>
<th>8-bit mode</th>
<th>16-bit mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer 1</td>
<td>T1CR:T1IF</td>
<td>T1CR:T1IF</td>
</tr>
<tr>
<td>Timer 2</td>
<td>T2CR:T2IF</td>
<td>T1CR:T1IF</td>
</tr>
<tr>
<td>Timer 1 + Timer 2</td>
<td>T1CR:T1IF</td>
<td>T1CR:T1IF</td>
</tr>
<tr>
<td>Interrupt request enable bit</td>
<td>T1CR:T1IE</td>
<td>T2CR:T2IE</td>
</tr>
<tr>
<td>Timer 1</td>
<td>T1CR:T1IE</td>
<td>T1CR:T1IE</td>
</tr>
<tr>
<td>Timer 2</td>
<td>T2CR:T2IE</td>
<td>T1CR:T1IE</td>
</tr>
<tr>
<td>Timer 1 + Timer 2</td>
<td>T1CR:T1IE</td>
<td>T1CR:T1IE</td>
</tr>
<tr>
<td>Interrupt source</td>
<td>A match of value set in T1DR and 8-bit counter value</td>
<td>A match of value set in T2DR and 8-bit counter value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A match of values set in T1DR and T2DR and 16-bit counter value</td>
</tr>
<tr>
<td>Interrupt request</td>
<td>IRQ6</td>
<td></td>
</tr>
</tbody>
</table>

Though 8/16-bit timer/counter interrupt requests are generated by Timers 1 and 2 independently in 8-bit mode and generated by Timer 1 in 16-bit mode, the basic operation is the same. Only the interrupt operation of Timer 1 in 8-bit mode is explained below.

Interrupt operation of Timer 1 in 8-bit mode

The selected count clock counts up from the counter value "00H" until the count value matches the value set in the comparator data latch corresponding to the timer data register (T1DR). When a match occurs, the interrupt request flag bit (T1CR:T1IF) is set to "1".

When the interrupt request enable bit is set to "1" (T1CR: T1IE = 1) at this time, an interrupt request to the CPU (IRQ6) occurs. Write "0" into the T1IF bit in the interrupt processing routine to clear the interrupt request.

The T1IF bit is set to "1" when the counter value matches the set value regardless of the value of the T1IF bit.

In 8-bit mode, Timers 1 and 2 operate independently and generate the same interrupt request (IRQ6). Therefore, evaluation of the interrupt request flag bit by software may be required.

Reference:

A match of the counter value and the T1DR register value occurs concurrently with the stop of the counter (T1CRT: T1STR = 0), and the T1IF bit is not set.

If the T1IE bit is changed from "0" to "1" ("disable" --> "enable") when the T1IF bit is "1", an interrupt request occurs immediately.
Table 8.5-2 Register and Vector Table Related to 8/16-bit Timer/Counter Interrupts

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ6</td>
<td>ILR2 (007DH)</td>
<td>L61 (bit 5)</td>
</tr>
</tbody>
</table>

For interrupt operation, see Section 3.4.2 “Flow of Interrupt Operation.”
8.6 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the 8/16-bit timer/counter.

- **Operation of the Interval Timer Function**

- **8-bit mode**

  To operate Timer 1 as the interval timer function in 8-bit mode, a setting as shown in Figure 8.6-1 "Interval Timer Function (Timer 1) Setting" is required.

  ![Figure 8.6-1 Interval Timer Function (Timer 1) Setting](image)

  To operate Timer 2 as the interval timer function in 8-bit mode, a setting as shown in Figure 8.6-2 "Interval Timer Function (Timer 2) Setting" is required.

  ![Figure 8.6-2 Interval Timer Function (Timer 2) Setting](image)

  When the counter is started in 8-bit mode, the timer counts up from “00H” at the rising edge of a clock pulse of the selected count clock. When the counter value matches the value set in the data register (comparator data latch), the timer sets the interrupt request flag bit of the timer control register (T1IF or T2IF) to “1” and counts up from “00H” again. When Timer 1 is used, the output of the square wave output control circuit is inverted when a match is detected and a square wave is output from the TMO1 pin when square wave output is enabled (T1CR: T1OS1, T1OS0 = other than 11B).

  When Timer 2 is used, the output of the square wave output control circuit is inverted when a match is detected and a square wave is output from the TMO2 pin when square wave output is enabled (T2CR: T2OS1, T2OS0 = other than 11B).
8.6 Operation of Interval Timer Function

Figure 8.6-3 "Operation of the Interval Timer Function in 8 bit Mode (Timer 1)" shows the operation of the interval timer function in 8-bit mode.

**Figure 8.6-3 Operation of the Interval Timer Function in 8 bit Mode (Timer 1)**

To operate the 16-bit timer as the interval timer function in 16-bit mode, a setting as shown in Figure 8.6-4 "Interval Timer Function Setting (in 16-bit Mode)" is required.

**Figure 8.6-4 Interval Timer Function Setting (in 16-bit Mode)**

Though timer control in 16-bit mode is performed by the timer 1 control register (T1CR), an initial setting of the timer 2 control register (T2CR) is also required. The value set in the data register is 16-bit data with the T2DR register as the high-order byte and the T1DR register as the low-order byte and is compared to the 16-bit counter value. The 16 bits are cleared simultaneously when the counter is cleared. For operations other than the above, operation in 16-bit mode is the same as that in 8-bit mode.
8.7 Operation of Counter Function

This section describes the operation of the counter function of the 8/16-bit timer/counter.

- **Operation of the Counter Function**

  - **8-bit mode**

    To operate Timer 1 as the counter function in 8-bit mode, a setting as shown in Figure 8.7-1 "Counter Function Setting (in 8-bit Mode)" is required.

    **Figure 8.7-1 Counter Function Setting (in 8-bit Mode)**

    | DDR6   | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
    |--------|------|------|------|------|------|------|------|------|
    | ☒      | ☒    | ☒    | ☒    | ☒    | 0    | ☒    | ☒    | ☒    |

    The operation of the counter function in 8-bit mode is the same as the interval timer function (Timer 1 in 8-bit mode) except that an external clock is used instead of an internal clock.
16-bit mode

To operate the counter function in 16-bit mode, a setting as shown in Figure 8.7-2 "Counter Function Setting (in 16-bit Mode)" is required.

**Figure 8.7-2 Counter Function Setting (in 16-bit Mode)**

- **DDR6**: Used bit
- **T1CR**: T1IF, T1IE, T1OS1, T1OS0, T1CS1, T1CS0, T1STP, T1STR
- **T2CR**: T2IF, T2IE, T2OS1, T2OS0, T2CS1, T2CS0, T2STP, T2STR
- **T1DR**: Set the lower eight-bits of the counter value to be compared
- **T2DR**: Set the upper eight-bits of the counter value to be compared

The operation of the counter function in 16-bit mode is the same as the interval timer function (in 16-bit mode) except that an external clock is used instead of an internal clock.

Figure 8.7-3 "Operation of the Counter Function in 16-bit Mode" shows the operation of the counter function in 16-bit mode.
Note:

When reading the counter value in operation in 16-bit mode, confirm that the value is acceptable.
8.8 Operation of Square Wave Output Initial Setting Function

By setting the timer control registers (T1CR, T2CR), square wave output (TMO1, TMO2) can be set to an initial value.

Operation of Square Wave Output Initial Setting Function (Timer 1)

Square wave output can be set to an initial value by a program only when the timer operation is stopped (T1CR: T1STR = 0)

Figure 8.8-1 "Initial Setting Equivalent Circuit of the Square Wave Output Control Circuit" shows the initial setting equivalent circuit of the square wave output control circuit. For initial setting, follow the steps shown in Table 8.8-1 "Steps for Initial Setting of Square Wave Output (T1CR Register)". The square wave output operation at this time is shown in Figure 8.8-2 "Initial Setting Operation of Square Wave Output".

Table 8.8-1 Steps for Initial Setting of Square Wave Output (T1CR Register)

<table>
<thead>
<tr>
<th>Step</th>
<th>Setting and operation</th>
</tr>
</thead>
</table>
| (1)  | To output "L" level signals from the square wave output pin (TMO1), write "01B" to the square wave output control bit (T1CR: T1OS1, T1OS0), and to output "H" level signals from the TMO1 pin, write "10B" to the bit.  
  **Note:** Before "11B" is written, a value is retained in the latch and the level of the TMO1 pin maintains the present or previous value. |
| (2)  | When "11B" is written to the square wave output control bits (T1OS1, T1OS0), the level corresponding to the value of the level latch written in Step (1) (initial value) is output to the TMO1 pin. Do not make an initial setting of the pins (T1OS1, T1OS0=11) and start the counter (T1STR = 0 --> 1) simultaneously. When the timer start bit is set (T1STR = 1), the counter starts operation. |
| (3)  | Whenever the counter value and the data register set value match, the square wave output is inverted. |
Operation of Square Wave Output Initial Setting Function (Timer 2)

Square wave output can be set to any initial value by a program only when the timer operation is stopped (T2CR: T2STR = 0).

Figure 8.8-3 "Initial Setting Equivalent Circuit of the Square Wave Output Control Circuit" shows the initial setting equivalent circuit of the square wave output control circuit. For initial setting, follow the steps shown in Table 8.8-2 "Steps for Initial Setting of Square Wave Output (T2CR Register)". The square wave output operation at this time is shown in Figure 8.8-4 "Initial Setting Operation of Square Wave Output".

Figure 8.8-3 Initial Setting Equivalent Circuit of the Square Wave Output Control Circuit
6.8 Operation of Square Wave Output Initial Setting Function

<table>
<thead>
<tr>
<th>Step</th>
<th>Setting and operation</th>
</tr>
</thead>
</table>
| **(1)** | To output "L" level signals from the square wave output pin (TMO2), write "01B" to the square wave output control bits (T2CR: T2OS1, T2OS0), and to output "H" level signals from the TMO2 pin, write "10B" to the bit.  
**Note:** Before "11B" is written, a value is retained in the latch and the level of the TMO2 pin maintains the present or previous value. |
| **(2)** | When "11B" is written to the square wave output control bits (T2OS1, T2OS0), the level corresponding to the value of the level latch written in Step (1) (initial value) is output to the TMO2 pin. Do not make an initial setting of the pins (T2OS1, T2OS0=11) and start the counter (T2STR = 0 --> 1) simultaneously. When the timer start bit is set (T2STR = 1), the counter starts operation. |
| **(3)** | Whenever the counter value and the data register set value match, the square wave output is inverted. |

Table 8.8-2  Steps for Initial Setting of Square Wave Output (T2CR Register)

---

**Figure 8.8-4 Initial Setting Operation of Square Wave Output**

*1 When the T2OS1 and T2OS0 bits of the T2CR register are something other than "11", the P61/TMO2 pin serves as a general-purpose port (P61).  
*2 When "11" is set to the T2OS1 and T2OS0 bits, the P61/TMO2 pin serves as a square wave output pin (TMO2).
8.9 Stop and Restart Operation of 8/16-bit Timer/Counter

This section describes the stop and restart operation of the 8/16-bit timer/counter.

- Stop and Restart of the Timer

Since the operation of Timers 1 and 2 are the same, this section describes Timer 1 only.

For the stop and restart of Timer 1, use the timer stop bit (T1STP) and the timer start bit (T1STR) of the timer 1 control register (T1CR).

- When the counter is cleared before starting a count operation

Set the T1STP and T1STR bits to "01B" when the T1STR bit is "0". The timer is cleared at the rising edge of the T1STR bit, and the timer starts a count operation.

- When the timer is stopped temporarily and the count operation is restarted without clearing the counter

To stop the count operation temporarily, set the T1STP and T1STR bits to "11B". To restart the count operation without clearing the counter from the temporary stop state, set the T1STP and T1STR bits to "01B".

Table 8.9-1 "Stop and Restart of the Timer" shows the states of the timer indicated with T1STP and T1STR bits and the operation when the timer is started from this state (T1STP, T1STR = 01B).

<table>
<thead>
<tr>
<th>T1STP (T2STP)</th>
<th>T1STR (T2STR)</th>
<th>Timer state</th>
<th>Timer operation when the timer is started (T1STP, T1STR = 01B) from the condition at left</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Count operation stop</td>
<td>Clears the counter and starts count operation.</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Count operation is in progress</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Count operation stop</td>
<td>Clears the counter and starts count operation.</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Temporary stop of count operation</td>
<td>Starts count operation without clearing the counter</td>
</tr>
</tbody>
</table>
This section describes the operation when transition to sleep or stop mode occurs and a stop request is issued during the operation of the 8/16-bit timer/counter.

### Operation in Sub-clock Mode, Standby Mode, and Stop during Operation

When transition to sleep or stop mode occurs and a stop request is issued during the operation in interval timer function mode and counter function mode, the states of the counter values are as shown in Figure 8.10-1 "Operation of Counter in Sub-clock Mode, Standby Mode, and Stop during Operation" (when Timer 1 is operating).

When transition to stop mode occurs, the counter stops with its value retained. When stop mode is released by an external interrupt, the counter restarts operation from the value at the time of the stop. Therefore, the initial interval time and the initial cycle of the PWM signals are not the set value. After stop mode is released, initialize the 8/16-bit timer/counter again.

Transition to watch mode (STBC: TMD = 1) and release of watch mode operate in the same manner as transition to stop mode and release of stop mode. Watch mode is released by a watch interrupt and an external interrupt.

When the counter is stopped temporarily (T1STP = 1), the counter stops with its value retained. When the operation is continued later (T1STP = 0), the counter restarts the count operation.

*If the pin state specification bit (STBC: SPL) of the standby control register is set to “1” and the TMO1 pin is not pulled up (optional), the TMO1 pin in stop mode is set to high-impedance.

If the SPL bit is set to “0”, the value immediately before transition to stop mode is maintained.
8.11 Precautions on Use of 8/16-bit Timer/Counter

This section describes precautions when using the 8/16-bit timer/counter.

- Precautions on Use of 8/16-bit Timer/Counter

  - Caution on Timer Stop

    This section provides an explanation of Timer 1 that also applies to Timer 2.

    When the timer is stopped temporarily with T1STP bit, the counter value is incremented if the input clock pulse is "L" level. When "00B" is written into T1STP and T1STR bits simultaneously after a temporary stop, the counter value may be incremented if the input clock pulse is "L" level.

    When the timer is stopped temporarily with T1STP bit, read the counter value before writing "0" to the T1STR bit.

    **Figure 8.11-1 Operation When the Timer Stop Bit is Used**

    ![Operation Diagram]

  - Error

    Since the start of the counter by a program and the start of a count operation with the selected count clock are asynchronous, an error may occur before the detection of a match of the counter value and the set data by a maximum of one cycle of the count clock cycle. Figure 8.11-2 "Error before the Start of the Count Operation" shows the error before the start of the count operation.
oca Precautions on Use of 8/16-bit Timer/Counter

Figure 8.11-2 Error before the Start of the Count Operation

- **Use on one channel of eight bits**
  When using only Timer 1 of the 8/16-bit timer/counter in 8-bit mode, set the count clock selection bit of the timer 2 control register (T2CR: T2CS1, T2CS0) to a value other than "11B" before use. Use of Timer 1 of the 8/16-bit timer/counter without setting the register may result in a malfunction.

- **Precautions on setting by a program**
  - When using the 8/16-bit timer/counter in 16-bit mode, write "11B" into the timer count clock selection bits of the timer 2 control register (T2CR: T2CS1, T2CS0) and write "00B" into the square wave output control bits (T2CR: T2OS1 T2OS0).
  - When reading the counter value in operation in 16-bit mode, confirm that the value is acceptable.
  - The output value of the square wave output is not changed even if the initial setting is made during the timer operation (T1CR: T1STR = 1). It is initialized when the timer operation is stopped.
  - Restoration from interrupt processing cannot be made when the interrupt request flag bit (T1CR: T1IF, T2CR: T2IF) is set to “1” and the interrupt request enable bit is set to “1” (T1CR: T1IE = 1, T2CR: T2IE = 1). The interrupt request flag bit must be cleared.
  - When the counter operation stop due to timer start bits (T1CR: T1STR = 0, T2CR: T2STR=0) and the occurrence of an interrupt source occur simultaneously, the interrupt request flag bit is not set (T1CR: T1IF, T2CR: T2IF).
8.12 8/16-bit Timer/Counter Program Examples

Program examples of the 8/16-bit timer/counter are shown below

A Program Example of the Interval Timer Function

- **Processing specifications**
  - Only Timer 1 is used in 8-bit mode and interval timer interrupts of 2.62 ms are generated repeatedly.
  - A square wave that is inverted at the interval time is output to the TMO1 pin.
  - The T1DR register value in which the interval time becomes about 2.62 ms when the main clock speed (gear) is highest (1 instruction cycle = 4/F<sub>CH</sub>) at a main clock source oscillation frequency (F<sub>CH</sub>) of 8 MHz is shown below. The internal count clock 128 t<sub>inst</sub> is used in the following example.

\[
\text{T1DR register value} = \frac{2.62 \text{ ms}}{(128 \times 4/8 \text{ MHz})} - 1 = 40.0 \text{ (28H)}
\]
Coding example

\[\text{T2CR} \text{ EQU} \ 0026\text{H} \quad ; \text{Address of timer 2 control register}\]
\[\text{T1CR} \text{ EQU} \ 0027\text{H} \quad ; \text{Address of timer 1 control register}\]
\[\text{T2DR} \text{ EQU} \ 0028\text{H} \quad ; \text{Address of timer 2 data register}\]
\[\text{T1DR} \text{ EQU} \ 0029\text{H} \quad ; \text{Address of timer 1 data register}\]
\[\text{T1IF} \text{ EQU} \ \text{T1CR}:7 \quad ; \text{Definition of timer 1 interrupt request flag bit}\]
\[\text{ILR2} \text{ EQU} \ 007\text{DH} \quad ; \text{Address of interrupt level setting register}\]
\[\text{INT_V} \text{ DSEG} \text{ ABS}\]
\[\text{ORG} \ \text{FFEEH}\]
\[\text{IRQ6} \text{ DW} \ \text{WARI} \quad ; \text{Interrupt vector setting}\]

;----------Main program------------------------------------------------------------
CSEG ; [CODE SEGMENT]
  ; Assuming that stack pointer (SP) has been initialized
  CLRI ; Interrupt disable
  MOV ILR2,#11101111B ; Set interrupt level to 2
  MOV T2CR,#00000010B ; Clear timer 2 interrupt request flag, disable interrupt
                    ; request output, set to something other than 16-bit
                    ; mode, stop operation
  MOV T1CR,#00011000B ; Clear timer 1 interrupt request flag, square wave
                      ; output initial value "L", select 128 \text{Inst}, stop operation
  MOV T1DR,#28H ; Set value to be compared to counter value (interval time).
  MOV T1CR,#00111000B ; Output "L" level to square wave output pin (TMO1).
  MOV T1CR,#11111001B ; Enable timer 1 interrupt request output, clear counter,
                      ; and start timer
  SETI ; CPU interrupt enable
  CLRB T1IF ; Clear interrupt request flag
  PUSHW A
  XCHW A,T
  POPW A
  User processing
  POPW A
  XCHW A,T
  POPW A
  RETI
ENDS
;------------------------------------------------------------------------------------------------------------------------------
END

;----------Interrupt Program---------------------------------------------------------
WARI CLRB T1IF ; Clear interrupt request flag
  PUSHW A
  XCHW A,T
  PUSHW A
  User processing
  POPW A
  XCHW A,T
  POPW A
  RETI
ENDS
;-------------------------------------------------------------------------------------------------------------------------------

END
A Program Example of Pulse Counter Function

- **Processing specifications**
  - Timers 1 and 2 are used in 16-bit mode and an interrupt is generated when the input of an external clock pulse to the TCLK pin is counted 5000 times ($1\text{388}_{10}$).
  - A sample program for reading the 16-bit counter value during a counter operation (READ16) is shown below.
Coding example

<table>
<thead>
<tr>
<th>DDR6</th>
<th>EQU 0013H ; Address of port direction register</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2CR</td>
<td>EQU 0026H ; Address of timer 2 control register</td>
</tr>
<tr>
<td>T1CR</td>
<td>EQU 0027H ; Address of timer 1 control register</td>
</tr>
<tr>
<td>T2DR</td>
<td>EQU 0028H ; Address of timer 2 data register</td>
</tr>
<tr>
<td>T1DR</td>
<td>EQU 0029H ; Address of timer 1 data register</td>
</tr>
<tr>
<td>T1IF</td>
<td>EQU T1CR:7 ; Definition of timer 1 interrupt request flag bit</td>
</tr>
<tr>
<td>ILR2</td>
<td>EQU 007DH ; Address of interrupt level setting register 2</td>
</tr>
<tr>
<td>NT_V</td>
<td>DSEG ABS ORG FFEEH</td>
</tr>
<tr>
<td>IRQ6</td>
<td>DW WARI ; Interrupt vector setting</td>
</tr>
</tbody>
</table>

;----------Main program-------------------------------------------------------------------------------------------------------- -----
CSEG ; [CODE SEGMENT] ; Assuming that stack pointer (SP) has been initialized
MOV DDR2,#00000000B ; Set P62/TCLK pin to input
CLR1 ; Interrupt disable
MOV ILR2,#11101111B ; Set interrupt level to 2
MOV T1DR,#088H ; Set lower 8 bits of comparison value to be compared to counter value
MOV T2DR,#013H ; Set upper 8 bits of comparison value to be compared to counter value
MOV T2CR,#00001100B ; Set Timer 2 to 16-bit mode
MOV T1CR,#01001101B ; Clear timer 1 interrupt request flag, enable interrupt request output, set to general-purpose port (P60), select external clock, clear counter to start operation
SETI ; CPU interrupt enable

;----------Data read sub-routine----------------------------------------------------------------------------------------------- ---
READ16 MOVW A,T2DR ; Read 16 bits of T1DR + T2DR
MOVW A,T2DR ; Read 16 bits of T1DR + T2DR and store the old value to T register
CMPW A ; Check by reading twice and compare A and T
BEQ RET16 ; Match, Return
XCHW A,T
INCW A ; Old value + 1
CMPW A
BNE READ16 ; To re-read when no match

RET16 RET

;----------Interrupt program--------------------------------------------------------------------------------------------------- -----
WARI CLR1 T1IF ; Clear interrupt request flag
PUSHW A
XCHW A,T
PUSHW A

; User process

POPW A
XCHW A,T
POPW A
RET1 RET
ENDS

END
CHAPTER 9  8-BIT SERIAL I/O

This chapter describes the functions and operations of the 8-bit serial I/O.

9.1  "Overview of 8-Bit Serial I/O"
9.2  "8-Bit Serial I/O Configuration"
9.3  "Pins of 8-bit Serial I/O"
9.4  "Registers of the 8-bit Serial I/O"
9.5  "8-Bit Serial I/O Interrupts"
9.6  "Operation of Serial Output"
9.7  "Operation of Serial Input"
9.8  "State in Each Mode during 8-bit Serial I/O Operation"
9.9  "Precaution on Use of 8-Bit Serial I/O"
9.10 "8-bit Serial I/O Connection Example"
9.11 "8-Bit Serial I/O Program Examples"
9.1 Overview of 8-Bit Serial I/O

The 8-bit serial I/O has the function of transferring 8-bit data serially in synchronization with shift clock pulses. For the shift clock, one shift clock can be selected from three types of internal shift clock and one external shift clock. For the data shift direction, either MSB first or LSB first can be selected.

### Serial I/O Function

The 8-bit serial I/O has the function of transferring 8-bit data serially in synchronization with shift clock pulses.

- Eight-bit parallel data is converted into serial data and output. Or serial data input is converted into parallel data and stored.
- For the shift clock, one shift clock can be selected from three types of internal shift clock and one external shift clock.
- I/O of the shift clock can be controlled and the internal shift clock can be output.
- For the data shift direction (transfer direction), either MSB first or LSB first can be selected.

#### Table 9.1-1 Cycle and Transfer Speed of the Shift Clock

<table>
<thead>
<tr>
<th>Shift clock</th>
<th>Clock cycle</th>
<th>Frequency (Hz)</th>
<th>Transfer speed (F_CH = 8 MHz, at the highest speed)*1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal shift clock (output)</td>
<td>2 t_inst</td>
<td>1/(2 t_inst)</td>
<td>1.0 Mbps</td>
</tr>
<tr>
<td></td>
<td>8 t_inst</td>
<td>1/(8 t_inst)</td>
<td>250.0 kbps</td>
</tr>
<tr>
<td></td>
<td>32 t_inst</td>
<td>1/(32 t_inst)</td>
<td>62.5 kbps</td>
</tr>
<tr>
<td>External shift clock (input)</td>
<td>2 t_inst or more</td>
<td>1/(2 t_inst) or less</td>
<td>DC to 1.0 Mbps</td>
</tr>
</tbody>
</table>

F_CH: Main clock source oscillation  

`t_inst`: Instruction cycle (influenced by clock mode)  

*1: When the highest speed (CS1, CS0 = 11B, 1 instruction cycle = 4/F_CH) in main clock mode (SCS = 1) is selected by the system clock control register (SYCC)
9.2 8-Bit Serial I/O Configuration

Each channel of the 8-bit serial I/O consists of the following four blocks:
- Shift clock control circuit
- Shift clock counter
- Serial data register (SDR)
- Serial mode register (SMR)

Block Diagram of the 8-bit Serial I/O

Figure 9.2-1 Block Diagram of the 8-bit Serial I/O

| tinst: Instruction cycle |
CHAPTER 9 8-BIT SERIAL I/O

- **Shift clock control circuit**

  There are four optional types of shift clock: three types of internal clock and one external clock.

  When an internal clock is selected, shift clock pulses can be output to the SK1 and SK2 pins. When the external clock is selected, clock pulses input from the SK1 and SK2 pins serve as shift clock pulses. By this shift clock, the SDR register is shifted and the shift-out value is output to the SO1 and SO2 pins. The inputs of the SI1 and SI2 pins are fetched while shifting to the SDR register.

- **Shift clock counter**

  The shift clock counter counts the number of SDR register and counter overflows when the shift of eight bits is completed.

  By this overflow, the serial I/O transfer start bit of the SMR register is cleared (SST = 0) and the interrupt request flag is set (SIOF = 1). The shift clock counter stops the count operation by the stop of serial transfer (SST = 0). The counter is cleared by starting the count operation (SST = 1).

- **Serial data register (SDR)**

  A register for retaining transfer data. The data written into this register is output after being converted into serial data, and serial input is stored after being converted into parallel data.

- **Serial mode register (SMR)**

  The serial mode register is a control register of serial I/O. This register enables/disables serial I/O operation, selects the shift clock, sets the transfer (shift) direction, controls interrupts, and checks the states.

- **Serial I/O-related interrupts**

  **IRQ5:**

  An interrupt request (IRQ5) occurs if the interrupt request output is enabled (SMR: SIOE = 1) when eight bits of serial data are input or output in the input/output function of the serial I/O.
9.3 Pins of 8-bit Serial I/O

This section describes the 8-bit serial-I/O related pins and provides block diagrams of the pins.

- Pins Related to the 8-bit Serial I/O

The pins related to the 8-bit serial I/O include P36/SI1, P37/SO1, P35/SCK1, P71/SI2, P72/SO2, and P70/SCK2 pins. These pins can be switched by the serial I/O port switching register (SSEL), as shown in Table 9.3-1 "Pins Related to the 8-bit Serial I/O".

Table 9.3-1  Pins Related to the 8-bit Serial I/O

<table>
<thead>
<tr>
<th>I/O pins</th>
<th>Shift clock I/O</th>
<th>Data input</th>
<th>Data output</th>
</tr>
</thead>
<tbody>
<tr>
<td>P35/SK1&lt;sup&gt;1&lt;/sup&gt;</td>
<td>P36/SI1</td>
<td>P37/SO1&lt;sup&gt;1&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>P70/SK&lt;sup&gt;2&lt;/sup&gt;</td>
<td>P71/SI2</td>
<td>P72/SO&lt;sup&gt;2&lt;/sup&gt;</td>
<td></td>
</tr>
</tbody>
</table>

*1: Output is CMOS format
*2: Output is Nch open-drain format

: Initial setting

- P36/SI1 pin

This pin serves as a general-purpose I/O port (P36) and a serial data input pin (hysteresis input) of the serial I/O (SI1).

When using this pin as a SI1 pin, set it to the input port (DDR3: bit6 = 0) by the port direction register.

- P71/SI2 pin

This pin serves as a general-purpose I/O port (P71) and a serial data input pin (hysteresis input) of the serial I/O (SI2).

When using this pin as a SI2 pin, set it to the input port by setting the port data register (PDR7: bit1) to "1" to turn off the output transistor.

- P37/SO1 pin

This pin serves as a general-purpose I/O port (P37) and a serial data output pin (CMOS output) of the serial I/O (SO1).

When the output of serial data is enabled (SMR: SOE = 1), this pin is set to an output pin automatically regardless of the value of the port direction register (DDR3: bit7) and also functions as the SO1 pin.
CHAPTER 9 8-BIT SERIAL I/O

○ P72/SO2 pin

This pin serves as a general-purpose I/O port (P72) and a serial data output pin (Nch open-drain output) of the serial I/O (SO2).

When the output of serial data is enabled (SMR: SOE = 1), this pin is set to an output pin automatically regardless of the value of the output latch (PDR7: bit2) and also functions as the SO2 pin.

○ P35/SK1 pin

This pin provides a general-purpose I/O port (P35) and a shift clock pulse input/output pin (hysteresis input, CMOS output) of the serial I/O (SK1).

- When this pin is used as a shift clock input pin

To use SK1 as an input pin, set it to the input port (DDR3: bit5 = 0) by the port direction register and disable the output of shift clock pulses (SMR: SCKE =0). At this time, be sure to select the external shift clock (SMR: CKS1, CKS0 = 11B).

- When this pin is used as a shift clock pulse output pin

When the output of the shift clock is enabled (SMR: SCKE = 1), this pin is set to an output pin automatically regardless of the value of the port direction register (DDR3: bit5) and also functions as the SK1 pin. At this time, be sure to select an internal shift clock (SMR: CKS1, CKS0 = Other than 11B).

○ P70/SK2 pin

This pin provides a general-purpose I/O port (P70) and a shift clock pulse input/output pin (hysteresis input, Nch open-drain output) of the serial I/O (SK2).

- When this pin is used as a shift clock input pin

When using SK2 as an input pin, set it to the input port (PDR7: bit0 = 1) by the port direction register and disable the output of shift clock pulses (SMR: SCKE =0). At this time, be sure to select the external shift clock (SMR: CKS1, CKS0 = 11B).

- When this pin is used as a shift clock pulse output pin

When the output of the shift clock is enabled (SMR: SCKE = 1), this pin is set to an output pin automatically regardless of the value of the output latch (PDR7: bit0) and also functions as the SK2 pin. At this time, be sure to select an internal shift clock (SMR: CKS1, CKS0 = other than 11B).
9.3 Pins of 8-bit Serial I/O

Block Diagram of the Pins Related to the 8-bit Serial I/O

**Figure 9.3-1** Block Diagram of the Pins Related to the 8-bit Serial I/O (P35 to P37)

![Block Diagram of the Pins Related to the 8-bit Serial I/O (P35 to P37)](image1)

**Figure 9.3-2** Block Diagram of the Pins Related to the 8-bit Serial I/O (P70 to P72)

![Block Diagram of the Pins Related to the 8-bit Serial I/O (P70 to P72)](image2)

**Reference:****

When "With pull-up resistor" is set in optional setting, the pin state at the time of reset and in stop or watch mode (SPL = 1) is "H."

---

SPL: Pin state specification bit of standby control register (STBC)
### Registers of the 8-bit Serial I/O

This section describes the registers related to the 8-bit serial I/O.

#### Registers Related to the 8-bit Serial I/O

**SMR (Serial mode register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 CH</td>
<td>SIOF</td>
<td>SIOE</td>
<td>SCKE</td>
<td>SOE</td>
<td>CKS1</td>
<td>CKS0</td>
<td>BDS</td>
<td>SST</td>
<td>00000000h</td>
</tr>
<tr>
<td></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>
<td></td>
</tr>
</tbody>
</table>

**SDR (Serial data register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 DH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXX0h</td>
</tr>
<tr>
<td></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>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/write enabled  
X: Undefined
### 9.4.1 Serial Mode Register (SMR)

The serial mode register (SMR) enables/disables the operation of 8-bit serial I/O, selects a shift clock, sets the transfer direction, controls interrupts, and checks the state.

![Serial Mode Register (SMR)](image-url)
### Table 9.4-1 Functions of Each Bit in the Serial Mode Register (SMR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 7 | SIOF: Interrupt request flag bit | • This bit is set to "1" when eight bits of serial data are input or output in serial I/O operation. When this bit and the interrupt request enable bit (SIOE) are set to "1", an interrupt request is output.  
• In write cycle, this bit is cleared when "0" is written. Writing "1" into this bit does not affect the bit. |
| Bit 6 | SIOE: Interrupt request enable bit | A bit for enabling/disabling interrupt request output to the CPU. When this bit and the interrupt request flag bit (SIOF) are set to "1", an interrupt request is output. |
| Bit 5 | SCKE: Shift clock pulse output enable bit | • A bit for controlling I/O of shift clock pulses.  
• When this bit is "0", the P35/SK1 and P70/SK2 pins serve as shift clock input pins; when this bit is "1", they serve as shift clock pulse output pins.  
**Note:**  
• To use the P35/SK1 and P70/SK2 pins as shift clock input, they must be set to input ports. Select the external shift clock (CKS1, CKS0 = 11B) by the shift clock selection bits.  
• To use them as shift clock pulse output (SCKE = 1), select an internal shift clock (CKS1, CKS0 = other than 11B).  
**Reference:**  
• When the shift clock pulse output is enabled (SCKE = 1), the P35/SK1 pins functions as the SK1 pin regardless of the states of the general-purpose port (P35)  
• To use the P35/SK1 and P70/SK2 pins as general-purpose ports (P35, P70), select shift clock input (SCKE = 0). |
| Bit 4 | SOE: Serial data output enable bit | When this bit is "0", the P37/SO1 and P72/SO2 pins serve as general-purpose ports (P37, P72); when this bit is "1", the P37/SO1 pin serves as a serial data output pin (SO1).  
**Reference:** When the serial data output is enabled (SOE = 1), the P37/SO1 pin functions as the SO1 pin regardless of the state of the general-purpose port (P35). |
| Bit 3 | CKS1, CKS0: Shift clock selection bits | • Bits for selecting a shift clock from three types of shift clock and one external shift clock.  
• When these bits are set to something other than "11B", an internal shift clock is selected. If the shift count pulse output enable bit (SCKE) is "1", shift clock pulses are output from the SK1 and SK2 pins.  
When these bits are set to "11B", the external shift clock is selected. If shift clock input is set, shift clock pulses are input from the SK1 pin. |
| Bit 2 | BDS: Transfer direction selection bit | A bit for selecting whether serial data is transferred from the least significant bit (LSB first, BDS = 0) or from the most significant bit (MSB first, BDS = 1).  
**Note:** If this bit is rewritten after data is written to the serial data register (SDR), the data becomes invalid because the upper bits and lower bits of the data are exchanged when reading and writing data from/to the SDR. |
### Table 9.4-1 Functions of Each Bit in the Serial Mode Register (SMR) (Continued)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 0: SST: Serial I/O transfer start bit | • A bit for controlling the start of transfer and enabling/disabling transfer of the serial I/O. The completion of transfer can be checked with this bit.  
• When "1" is written into this bit when an internal shift clock is selected (CKS1, CKS0 = other than 11B), the shift clock counter is cleared and transfer starts.  
• When "1" is written into this bit when an external shift clock is selected (CKS1, CKS0 = 11B), transfer is enabled and the shift clock counter is cleared to wait for input of external shift clock pulses.  
• After transfer is completed, this bit is cleared to "0" and the SIOF bit is set to "1".  
• When "0" is written into this bit during transfer (SST = 1), transfer is suspended. When transfer is suspended, re-setting of the SDR register on the data output side and re-starting of transfer on the data input side (clearing the shift clock counter) is required. |

---

237
9.4.2 Serial Data Register (SDR)

The serial data register (SDR) retains transfer data of 8-bit serial I/O. In serial output operation mode, it functions as a transmission data register. In serial input operation mode, it functions as a reception data register.

Serial Data Register (SDR)

Figure 9.4-3 "Serial Data Register (SDR)" shows the bit configuration of the serial data register.

### Figure 9.4-3 Serial Data Register (SDR)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 Dn</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>
<td>XXXXXXXXa</td>
</tr>
</tbody>
</table>

- **R/W**: Read/write enabled
- **X**: Undefined

- **Serial output operation mode**
  This register functions as a transmission data register. When transfer of serial I/O starts (SMR: SST = 1), the data written in this register is transmitted serially.
  The transmitted data is shifted out by transfer and does not remain in the SDR register.

- **Serial input operation mode**
  This register functions as a reception data register. When transfer of serial I/O starts (SMR: SST = 1), the serially transferred data is stored in this register.

- **Precaution on serial I/O transfer**
  While the serial I/O is transferring data, do not write data into the SDR register. The read data has no significance.

**Reference:**
When serial output and serial input are enabled simultaneously, a serial I/O operation is performed.
9.5  8-Bit Serial I/O Interrupts

The interrupt source of the 8-bit serial I/O is the completion of serial I/O of eight-bit data.

- **Interrupts during Serial I/O Operation**

  In the 8-bit serial I/O, a serial output operation and serial input operation are performed simultaneously. When serial transfer starts, the value in the serial data register (SDR) is input/output bit by bit in synchronization with the set shift clock cycle. The interrupt request flag bit (SMR: SIOF) is set to "1" following a rise of the shift clock of the eighth bit.

  When the interrupt request enable bit is set to "1" (SMR: SIOE = 1) at this time, an interrupt request to the CPU (IRQ5) occurs.

  Write "0" into the SIOF bit in the interrupt processing routine to clear the interrupt request.

  The SIOF bit is set after the completion of 8-bit serial output regardless of the value of the SIOE bit.

  **Reference:**

  When the stop of serial transfer (SMR: SST = 0) and the completion of serial data transfer occur concurrently, the interrupt request flag bit is not set (SMR: SIOF = 1).

  If the SIOE bit is changed from "0" to "1" ("disable" --> "enable") when the SIOF bit is "1", an interrupt request occurs immediately.

- **Register and Vector Table Related to 8-bit Serial I/O Interrupts**

  **Table 9.5-1  Register and Vector Table Related to 8-bit serial I/O Interrupts**

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ5</td>
<td>ILR2 (007D&lt;sub&gt;H&lt;/sub&gt;)</td>
<td>L51 (bit 3) L50 (bit 2) FFF0&lt;sub&gt;H&lt;/sub&gt; FFF1&lt;sub&gt;H&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

  For interrupt operation, see Section 3.4.2 “Flow of Interrupt Operation.”
9.6 Operation of Serial Output

The 8-bit serial I/O can perform a serial output operation of 8-bit data in synchronization with the shift clock cycle.

Serial Output Operation

Serial output is operated by an internal or external shift clock. When serial I/O operation is enabled, the contents of the SDR register are output to the serial data output pins (SO1, SO2) concurrently with serial input.

Operation by an internal shift clock

To operate serial output with an internal shift clock, the setting in Figure 9.6-1 "Serial Output Setting (Internal Shift Clock)" is required.

Figure 9.6-1 Serial Output Setting (Internal Shift Clock)

When serial output operation is started, the contents of the SDR register are output to SO1 and SO2 pins in synchronization with the falling edge of a clock pulse of the selected internal shift clock. At this time, the transfer destination (serial input) must be in the external shift clock pulse input wait state.

Operation by an external shift clock

To operate serial output with an external shift clock, the setting in Figure 9.6-2 "Serial Output Setting (External Shift Clock)" is required.

Figure 9.6-2 Serial Output Setting (External Shift Clock)

When serial output operation is enabled, the contents of the SDR register are output to SO1 and SO2 pins in synchronization with the falling edge of a clock pulse of the external shift clock.
Upon completion of serial output, it is necessary to reset the SDR register and enable operation (SMR: SST = 1) immediately in preparation for the next data output.

When waiting for the output of the next data (idle state) following completion of the serial input operation by the receiving end (rising edge), set the external shift clock to “H” level.

Figure 9.6-3 "Operation of 8-bit Serial Output" shows the operation of 8-bit serial output

Figure 9.6-3 Operation of 8-bit Serial Output

#### Operation at the Completion of Serial Output

At the rising edge of a shift clock pulse at which the eighth bit of data is input/output, the interrupt request flag bit is set (SMR: SIOF = 1) and the serial I/O start bit is cleared (SMR: SST = 0).
The 8-bit serial I/O can perform a serial input operation of 8-bit data in synchronization with the shift clock cycle.

### Serial Input Operation

Serial input is operated by an internal shift clock or an external shift clock. When serial I/O operation is enabled, the contents of the SDR register are output to the serial data output pins (SO1, SO2) concurrently with serial input.

#### Operation by an internal shift clock

To operate serial input with an internal shift clock, the setting in Figure 9.7-1 “Serial Input Setting (Internal Shift Clock)” is required.

<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>×</td>
<td>0</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</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>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>1</td>
<td>×</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>0x10</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIOF</td>
<td>SIOE</td>
<td>SCKE</td>
<td>SOE</td>
<td>CKS1</td>
<td>CKS0</td>
<td>BDS</td>
<td>SST</td>
</tr>
</tbody>
</table>

When serial input operation is started, the contents of the serial data input pins (SI1, SI2) are taken into the SDR register in synchronization with the rising edge of a clock pulse of the selected internal shift clock. At this time, the SDR register in the transfer destination (serial output) must be set and the transfer destination must be in the external shift clock pulse input wait state.
9.7 Operation of Serial Input

- **Operation by an external shift clock**

  To operate serial input with an external shift clock, the setting in Figure 9.7-2 "Serial Input Setting (External Shift Clock)" is required.

  **Figure 9.7-2 Serial Input Setting (External Shift Clock)**

  When serial input operation is enabled, the contents of the serial data input pins (SI1, SI2) are taken into the SDR register in synchronization with the rising edge of an external shift clock pulse. Upon completion of serial input, it is necessary to read the SDR register and enable operation (SMR: SST = 1) immediately in preparation for the next data input.

  When waiting for the output of the next data (idle state), set the external shift clock to "H" level.

  Figure 9.7-3 "Operation of 8-bit Serial Input" shows the operation of 8-bit serial input.

- **Operation at the Completion of Serial Output**

  At the rising edge of a shift clock pulse at which the eighth bit of serial data is input/output, the interrupt request flag bit is set (SMR: SIOF = 1) and the serial I/O start bit is cleared (SMR: SST = 0).
9.8 State in Each Mode during 8-bit Serial I/O Operation

This section describes the operation when transition to sleep, stop, or watch mode occurs and a stop request is issued during the operation of the 8-bit serial I/O.

When an Internal Shift Clock Is Used

- **Operation in sleep mode**
  In sleep mode, serial I/O operation is not stopped and transfer is continued as shown in Figure 9.8-1 "Operation in Sleep Mode (Internal Shift Clock)".

  ![Figure 9.8-1 Operation in Sleep Mode (Internal Shift Clock)](image)

- **Operation in stop and watch modes**
  In stop and watch modes, serial I/O operation is stopped and transfer is suspended as shown in Figure 9.8-2 "Operation in Stop or Watch Mode (Internal Shift Clock)". After stop or watch mode is released and operation is restarted from the middle of a cycle, initialize again in accordance with the state of the transfer destination.

  ![Figure 9.8-2 Operation in Stop or Watch Mode (Internal Shift Clock)](image)
9.8 State in Each Mode during 8-bit Serial I/O Operation

- **Operation when transfer in progress is stopped**

  When operation is stopped in the middle of transfer operation (SMR: SST = 0), transfer is stopped and the shift clock counter is cleared as shown in Figure 9.8-3 "Operation Performed When Transfer Is Stopped in the Middle of Transfer (Internal Shift Clock)". Therefore, it is also necessary to initialize the transfer destination. At the time of serial output operation, reset the SDR register before restart.

  ![Figure 9.8-3 Operation Performed When Transfer Is Stopped in the Middle of Transfer (Internal Shift Clock)](image)

- **When an External Shift Clock Is Used**

- **Operation in sleep mode**

  In sleep mode, serial I/O operation is not stopped and transfer is continued as shown in Figure 9.8-4 "Operation in Sleep Mode (External Shift Clock)".

  ![Figure 9.8-4 Operation in Sleep Mode (External Shift Clock)](image)
Operation in stop and watch modes
In stop and watch modes, serial I/O operation is stopped and transfer is suspended as shown in Figure 9.8-5 "Operation in Stop or Watch Mode (External shift Clock)". Since operation is restarted from the middle of a cycle after stop or watch mode is released, a transfer error occurs. Initialize again.

Figure 9.8-5 Operation in Stop or Watch Mode (External shift Clock)

Operation when transfer is stopped in the middle of operation
When operation is stopped in the middle of transfer (SMR: SST = 0), transfer is stopped and the shift clock counter is cleared as shown in Figure 9.8-6 "Operation Performed When Transfer is Stopped in the Middle of Transfer (External Shift Clock)"). Therefore, it is also necessary to initialize the transfer destination. At the time of serial output operation, reset the SDR register before restart. When external clock pulses are input at this time, the output of SO1 and SO2 pins is changed.

Figure 9.8-6 Operation Performed When Transfer is Stopped in the Middle of Transfer (External Shift Clock)
9.9 Precaution on Use of 8-Bit Serial I/O

This section describes the precautions when using the 8-bit serial I/O.

- Precaution on Use of the 8-bit Serial I/O

  - Error at the time of serial transfer start
    Since the start of serial transfer by a program (SMR: SST = 1) and the time up to the falling edge (output) or the rising edge (input) of a shift clock pulse are asynchronous, the time up to the input/output of the first serial data is retarded by the set shift clock cycle at the maximum.

  - Malfunction due to noises
    If excess pulses (pulses exceeding the hysteresis width) are carried on the shift clock by external noise in serial data transfer, the serial I/O may malfunction.

  - Precautions on setting in a program
    - Writing into the serial mode register (SMR) and serial data register (SDR) is allowed only when the serial I/O is stopped (SMR: SST = 0).
    - When starting/enabling serial I/O transfer (SMR: SST = 1), do not change the other bits of the SMR register.
    - When using the externally input shift clock, the output levels of the SO1 and SO2 pins are the levels of the most significant bit in MSB first mode and the levels of the least significant bit in LSB first mode even during the serial I/O transfer (SMR: SST = 0) if the serial data output is enabled (SMR: SOE = 1).
    - If the serial I/O transfer stop (SMR: SST = 0) and completion of serial data transfer occur simultaneously, the interrupt request flag bit (SMR: SIOF) is not set.
    - Restoration from interrupt processing cannot be made when the SIOF bit is set to “1” and the interrupt request output is enabled (SIOE = 1). The SIOF bit must be cleared.

  - Serial I/O transfer speed
    The serial data output pin (SO2) of the serial I/O is not suited for high-speed transfer because it is Nch open-drain output. Caution is required when using it with a high-speed shift clock.
Shift clock idle state

During the wait time (idle state) between a transfer of 8-bit shift data and another transfer, set the external shift clock to "H." When an internal shift clock (SMR: CKS1, 0 = other than \(11_B\)) is specified as the shift clock pulse output (SMR: SCKE = 1), it outputs a "H" level signal in idle states.

Figure 9.9-1 "Shift Clock Idle State" shows the idle state of the shift clock.
9.10 8-bit Serial I/O Connection Example

This section shows a connection example in which the 8-bit serial I/O in MB89890 Series models are connected and bidirectional serial I/O is performed.

■ When Bidirectional Serial I/O Is Performed

Figure 9.10-1 8-bit Serial I/O Connection Example (Interface between MB89890 Series Models)
**Figure 9.10-2 Operation of Bidirectional Serial I/O**

- **SIO-A**
  - START
  - Stop operation of SIO-A (SST = 0)
  - Set S11 and S12 pins to serial data input (input ports)
  - Set SK1 and SK2 pins to shift count pulse output
  - Set SO1 and SO2 pins to serial data output
  - Select internal shift clock
  - Set transfer (shift) direction of data
  - 
  - Is SIO-B in serial transfer enabled state? *1
    - NO
    - Transfer enabled state
    - Transfer enabled state
    - Set output data
    - Start serial transfer (SST = 1)*2
      - SIO-A outputs serial data
      - Is 8-bit transfer completed? *3
        - NO
        - SIO-B is input simultaneously
        - SIO-A outputs serial data
        - Is there next data? *1
          - NO
          - END
        - YES (SST=0)
          - Read input data
    - YES
      - Read input data
      - Is there next data? *1
        - NO
        - END
  - YES
    - Set output data
    - Enable serial transfer (SST = 1)

- **SIO-B**
  - START
  - Stop SIO-B operation (SST = 0)
  - Set S11 and S12 pins to serial data input (input ports)
  - Set SK1 and SK2 pins to shift count pulse output
  - Set SO1 and SO2 pins to serial data output
  - Select external shift clock
  - Select same data transfer (shift) direction as SIO-A.
  - 
  - Is SIO-B in serial transfer enabled state? *1
    - NO
    - Transfer enabled state
    - Transfer enabled state
    - Set output data
    - Enable serial transfer (SST = 1)
      - SIO-B data is input simultaneously
      - Transferring serial data
      - 8-bit transfer completed? *3
        - NO
        - YES (SST=0)
          - Read input data
    - YES
      - Read input data
      - Is there next data? *1
        - NO
        - END

*SST: The SST bit is the serial I/O transfer start bit of the serial mode register (SMR).

*1: When connecting SO1, SO2, S11, S12, SK1, and SK2 pins only, a determination of whether SIO-B has entered the serial transfer enabled state is not possible. Therefore, it is necessary to wait until SIO-B enters the serial transfer enabled state by a software timer.

*2: Even if SIO-A starts transfer in states where SIO-B has not entered the serial transfer enable state, correct data transfer cannot be performed.

*3: After 8-bit data transfer is completed, an interrupt request occurs.
9.11 8-Bit Serial I/O Program Examples

Program examples of the 8-bit serial I/O are shown below.

A Program Example of Serial Output

- Processing specifications
  - 8-bit serial data (55H) is output from the SO1 pin of the serial I/O and an interrupt is generated when transfer is completed.
  - Transfer data is reset by the interrupt processing routine and is output continuously.
  - It operates with an internal shift clock, and a shift clock pulse is output from the SK1 pin.
  - The transfer speed and interrupt occurrence cycle in which the main clock speed (gear) is the highest (1 instruction cycle = 4/F_{CH}) at a main clock source oscillation frequency (F_{CH}) of 8 MHz are shown below. The shift clock 32 t_{inst} is used in the following example.

  Transfer speed = 8 MHz/4/32 = 62.5 kbps, Interrupt cycle = 4 x 32 x 8/8 MHz = 128 µs.
CHAPTER 9  8-BIT SERIAL I/O

○ Coding example

SMR EQU 001CH ; Address of serial mode register
SDR EQU 001DH ; Address of serial data register
SIOF EQU SMR:7 ; Definition of interrupt request flag bit
SST EQU SMR:0 ; Definition of serial I/O transfer start bit
ILR2 EQU 007DH ; Address of interrupt level setting register
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFF0H
IRQ5 DW WARI ; Interrupt vector setting
INT_V ENDS

;;---------------Main program-----------------------------------------------------------------------------------

CSEG ; [CODE SEGMENT]

; Assuming that stack pointer (SP) has been initialized

CLRI ; Interrupt disable
CLRB SST ; Stop serial I/O transfer
MOV ILR2,#11110111B ; Set interrupt level (level 1)
MOV SDR,#55H ; Set transfer data (55H)
MOV SMR,#01111000B ; Clear interrupt request flag, enable interrupt request
; output, enable shift clock pulse output (SK1), enable
; serial data output (SO1), select 32 t_{inst}, LSB first

SETB SST ; Start serial I/O transfer
SETI ; Interrupt enable

;;----------------Interrupt processing routine----------------------------------------------------------------------------------

WARI CLRB SIOF ; Clear interrupt request flag
PUSHW A
XCHW A,T ; Save A, T
PUSHW A
MOV SDR,#55H ; Reset transfer data (55H)
SETB SST ; Start serial I/O transfer

; User processing

: POPW A
XCHW A,T ; Restore A, T
POPW A
RETI
ENDS

;------------------------------------------------------------------------------------------------------------------------------

END
9.11 8-Bit Serial I/O Program Examples

■ A Program Example of Serial Input

○ Processing specifications
  - 8-bit serial data (55H) is input from the SI1 pin of the serial I/O and an interrupt is generated when transfer is completed.
  - Transferred data is read by the interrupt processing routine and is input continuously.
  - It operates with an external shift clock, and a shift clock pulse is input from the SK1 pin.

○ Coding example

```
; Program Example of Serial Input

; Processing specifications
; - 8-bit serial data (55H) is input from the SI1 pin of the serial I/O and an interrupt is generated when transfer is completed.
; - Transferred data is read by the interrupt processing routine and is input continuously.
; - It operates with an external shift clock, and a shift clock pulse is input from the SK1 pin.

; Coding example

DDR3 EQU 000DH ; Address of serial mode register
SMR EQU 001CH ; Address of serial data register
SDR EQU 001DH ; Address of serial data register
SIOF EQU SMR:7 ; Definition of interrupt request flag bit
SST EQU SMR:0 ; Definition of serial I/O transfer start bit
ILR2 EQU 007DH ; Address of interrupt level setting register
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFF0H

IRQ5 DW WARI ; Set interrupt vector
INT_V ENDS

;-----------------Main program---------------------------------------------------------------

CSEG ; [CODE SEGMENT]
; Assuming that stack pointer (SP) has been initialized

MOV DDR3,#00000000B ; Set P35/SK1 and P36/SI1 pins to input
CLRI ; Interrupt disable
CLRB SST ; Stop serial I/O transfer
MOV ILR2,#11110111B ; Set interrupt level (level 1)
MOV SMR,#01001100B ; Clear interrupt request flag, enable interrupt request
; output, set shift clock input (SK1), disable serial data
; output (SO1), select external clock, LSB first
SETB SST ; Enable serial I/O transfer
SETI ; Interrupt enable

;-----------------Interrupt processing routine-----------------------------------------------

WARI CLRB SIOF ; Clear interrupt request flag
PUSHW A
XCHW A,T
PUSHW A
MOV A,SDR ; Read transferred data
SETB SST ; Enable serial I/O transfer

; User processing

: POPW A
XCHW A,T
POPW A
RETI

END
```
This chapter describes the functions and operations of the serial I/O with 1-byte buffer.

10.1 "Overview of the Serial I/O with 1-byte Buffer"
10.2 "Configuration of the Serial I/O with 1-byte Buffer"
10.3 "Pins of the Serial I/O with 1-byte Buffer"
10.4 "Registers of the Serial I/O with 1-byte Buffer"
10.5 "Interrupts of the Serial I/O with 1-byte Buffer"
10.6 "Explanation of Operations of the Serial I/O with 1-byte Buffer"
10.7 "Connection Example of the Serial I/O with 1-byte Buffer"
10.8 "State Transition Diagram of the Serial I/O with 1-byte Buffer"
10.9 "Precautions when Using the Serial I/O with 1-byte Buffer"
10.10 "Program Example of the Serial I/O with 1-byte Buffer"
10.1 Overview of the Serial I/O with 1-byte Buffer

The serial I/O with 1-byte buffer provides a function to serial-transfer 8-bit data synchronized with the shift clock. A 1-byte transmitting buffer and a 1-byte receiving buffer are available so that data can be transferred continuously. One shift clock mode can be selected from three internal clock modes and one external clock mode. As the shift direction of data, LSB-first or MSB-first can be selected. Furthermore, an interrupt is generated by one of three types of interrupt source.

Serial I/O Function

The serial I/O function with a 1-byte buffer is a function for the serial input/output of 8-bit data synchronized with the shift clock.

- 8-bit parallel data is output after parallel-to-serial conversion, and serial data input is stored after serial-to-parallel conversion.
- One shift clock mode can be selected from three internal clock modes and one external clock mode.
- Input/output of the shift clock can be controlled so as to output the internal shift clock.
- As the shift direction (transfer direction) of data, LSB-first or MSB-first can be selected.
- A 1-byte serial buffer is available for writing and for reading so as to transfer data continuously.
- Three types of interrupt source can be used to generate an interrupt; overrun error, transmitting data empty, and receiving data full.
- By linking to the modem signal output circuit, transmitting data can be output as a modem signal (before modulation) of about 1208 bps or 2415 bps.

<table>
<thead>
<tr>
<th>Shift clock</th>
<th>Clock cycle</th>
<th>Frequency (Hz)</th>
<th>Transfer rate (when $F_{CH}=8\text{ MHz}$ and $t_{inst}=0.5\text{ μs}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal shift clock (output)</td>
<td>$2\ t_{inst}$</td>
<td>$1/(2\ t_{inst})$</td>
<td>1.0 Mbps</td>
</tr>
<tr>
<td></td>
<td>$8\ t_{inst}$</td>
<td>$1/(8\ t_{inst})$</td>
<td>250.0 kbps</td>
</tr>
<tr>
<td></td>
<td>$32\ t_{inst}$</td>
<td>$1/(32\ t_{inst})$</td>
<td>62.5 kbps</td>
</tr>
<tr>
<td>External shift clock (input)</td>
<td>$2\ t_{inst}$ or more</td>
<td>$1/(2\ t_{inst})$ or less</td>
<td>DC to 1.0 Mbps</td>
</tr>
</tbody>
</table>

$F_{CH}$: Main clock source oscillation
$t_{inst}$: Instruction cycle (CPU operating clock divided by 2)
10.1 Overview of the Serial I/O with 1-byte Buffer

Note:

The serial I/O with 1-byte buffer is a serial I/O that can send and receive data simultaneously, but because the transmission and reception of data cannot be selected, procedures for sending and receiving data are required if data is transferred continuously.

If data is received only, dummy transmitting data is written to the transmitting data buffer and the output of the transmitting buffer is disabled.

If data is sent only, empty receiving data is read.
10.2 Configuration of the Serial I/O with 1-byte Buffer

Each channel of the serial I/O with 1-byte buffer consists of the following seven blocks:
- Shift clock control circuit
- Shift clock counter
- Flag evaluation circuit
- Serial data register with 1-byte buffer (SBDR)
- Serial buffer register (SBUFW/R)
- Serial mode register with 1-byte buffer (SBMR)
- Serial flag register with 1-byte buffer (SBFR)

Block Diagram of the Serial I/O with 1-byte Buffer

Figure 10.2-1 Block Diagram of the Serial I/O with 1-byte Buffer
10.2 Configuration of the Serial I/O with 1-byte Buffer

- **Shift clock control circuit**
  One shift clock mode can be selected from three internal clock modes and one external clock mode.
  After the internal clock mode is selected, the shift clock can be output to the BSK1 (BSK2) pin. If the external clock mode is selected, the clock input pin through the BSK1 (BSK2) pin becomes the shift clock. The SBDR register is shifted using this shift clock, and the shifted out value is then output to the BSO1 (BSO2) pin. Also, input to the BSI1 (BSI2) pin is sent to the SBDR register by shifting it.

- **Shift clock counter**
  Counts the shift count of the SBDR register by the shift clock. When the 8-bit shift is completed, the counter overflows.
  This overflow is used as a timing signal for transmitting data load, receiving data load, and flag evaluation.

- **Flag evaluation circuit**
  Controls the output of the interrupt flag of an overrun error of the SBUFR register, data empty of the SBUFW register, and data full of the SBUFR register.

- **Serial data register with 1-byte buffer (SBDR)**
  A register to retain transfer data. Data written into this register is output after serial conversion. Furthermore, serial data input is stored after serial-to-parallel conversion.

- **Serial buffer register (SBUFW/R)**
  The SBUFW is provided as a buffer for 1-byte transmitting data write and the SBUFR register is provided as a buffer for 1-byte receiving data read. Both registers are located at the same address and function as an SBDR register. Since these registers enable the user to delay transfer data read/write by the transfer time of up to eight bits, data can be transferred continuously.

- **Serial mode register with 1-byte buffer (SBMR)**
  Control register of the serial I/O. This register is used to enable/disable serial I/O operations, select the shift clock, set the transfer (shift) direction, control interrupts, and check the status.

- **Serial flag register with 1-byte buffer (SBFR)**
  Register to retain the three interrupt flags of overrun error, data empty, and data full.
CHAPTER 10 SERIAL I/O WITH 1-BYTE BUFFER

- Interrupt sources related to the serial I/O with 1-byte buffer

  **IRQ4:**
  
  Input/output function of the serial I/O with 1-byte buffer. If any of the following conditions is satisfied, an interrupt request (IRQ4) is generated:
  
  - **Data full interrupt**
    
    If receiving data that has not been read is in the SBURF register (SBFR: SFLF=1), the data full interrupt output is enabled (SBMR: SFLE=1).
  
  - **Data empty interrupt**
    
    If no transmitting data of the SBDR register before transfer is in the SBUFW register (SBFR: SEMF=1), the data empty interrupt output is enabled (SBMR: SFME=1).
  
  - **Overrun error interrupt**
    
    If an overrun error occurs (SBFR: SOVF=1) after new receiving data is loaded into the SBUFR register when there is still receiving data that has not been read in the SBUFR register, the overrun error interrupt output is enabled (SBMR: SOVE=1).
10.3 Pins of the Serial I/O with 1-byte Buffer

This section explains the pins related to the serial I/O with 1-byte buffer and provides block diagrams of the pins. For the registers, see Section 10.4 "Registers of the Serial I/O with 1-byte Buffer".

## Pins Related to Serial I/O with 1-byte Buffer

The following pins are provided as those related to the serial I/O with 1-byte buffer; P65/BSK1, P66/BSI1, P67/BSO1, P73/BSK2, P74/BSI2, and P75/BSO2. As shown in Table 10.3-1 "Pins Related to the Serial I/O with 1-byte Buffer", these pins can be switched by the serial I/O port switch register (SSEL).

### Table 10.3-1  Pins Related to the Serial I/O with 1-byte Buffer

<table>
<thead>
<tr>
<th>I/O pins</th>
<th>Shift clock I/O</th>
<th>Data input</th>
<th>Data output</th>
</tr>
</thead>
<tbody>
<tr>
<td>P65/BSK1</td>
<td>*1</td>
<td>P66/BSI1</td>
<td>P67/BSO1*1</td>
</tr>
<tr>
<td>P73/BSK2</td>
<td>*2</td>
<td>P74/BSI2</td>
<td>P75/BSO2*2</td>
</tr>
</tbody>
</table>

*1: Output is in CMOS format.
*2: Output is in N-channel open-drain format.

: Initial setting

- **P66/BSI1 pin**
  The P66/BSI1 pin functions as a general-purpose I/O port (P66) and also as a serial data input pin (hysteresis input) of the serial I/O (BSI1).
  
  To use the pin as the BSI1 pin, set the pin as an input port (DDR6: bit6=0) using the port direction register.

- **P74/BSI2 pin**
  The P74/BSI2 pin functions as a general-purpose I/O port (P74) and also as a serial data input pin (hysteresis input) of the serial I/O (BSI2). To use the pin as the BSI2 pin, set the pin as an input port by setting the port data register (PDR7: bit4) to “1” and turning off the output transistor.

- **P67/BSO1 pin**
  The P67/BSO1 pin functions as a general-purpose I/O port (P67) and also as a serial data output pin (CMOS output) of the serial I/O (BSO1).

  If the output of serial data is enabled (SBMR: SOE=1), the BSO1 pin becomes an output pin automatically regardless of the value of the port direction register (DDR6: bit7) and thus functions as the BSO1 pin.
CHAPTER 10 SERIAL I/O WITH 1-BYTE BUFFER

○ P75/BSO2 pin

The P75/BSO2 pin functions as a general-purpose I/O port (P75) and also as a serial data output pin (N-channel open-drain output) of the serial I/O (BSO2).

If the output of serial data is enabled (SBMR: SOE=1), the BSO2 pin becomes an output pin automatically regardless of the value of the output latch register (PDR7: bit5) and thus functions as the BSO2 pin.

○ P65/BSK1 pin

The P65/BSK1 pin functions as a general-purpose I/O port (P65) and also as a shift clock I/O pin (hysteresis input/CMOS output) of the serial I/O (BSK1).

• When used as a shift clock input pin

To use the BSK1 pin as an input pin, set the pin as an input port using port direction register (DDR6: bit5=0) and then select the external shift clock mode (SBMR: CKS1, CKS0=00B).

• When used as a shift clock output pin

If the external shift clock mode is selected (SBMR: CKS1, CKS0=00B), the P65/BSK1 pin becomes an output pin automatically regardless of the value of the port direction register (DDR6=bit5) and thus functions as the BSK1 pin.

○ P73/BSK2 pin

The P73/BSK2 pin functions as a general-purpose I/O port (P73) and also as a shift clock I/O pin (hysteresis input/N-channel open-drain output) of the serial I/O (BSK2).

• When used as a shift clock input pin

To use the BSK2 pin as an input pin, set the pin as an input port using the port data register (PDR7: bit3=1) and then select the external shift clock mode (SBMR: CKS1, CKS0=00B).

• When used as a shift clock output pin

If the external shift clock mode is selected (SBMR=CKS1, CKS0=00B), the P73/BSK2 pin becomes an output pin automatically regardless of the value of the output latch (PDR7=bit3) and thus functions as the BSK2 pin.

For pin settings of the serial I/O, see Section 11.5 “I/O Pin Setting of the Serial I/O Ports.”
10.3 Pins of the Serial I/O with 1-byte Buffer

### Block Diagrams of the Serial I/O with 1-byte Buffer

#### Figure 10.3-1 Block Diagram of Pins Related to the Serial I/O with 1-byte Buffer (P65-P67)

![Block Diagram of Pins Related to the Serial I/O with 1-byte Buffer (P65-P67)](image)

- **PDR (port data register)**
  - **PDR read**
  - **PDR read (for bit processing commands)**
  - **PDR write**
  - **DDR read**
  - **DDR write**

- **DDR (port direction register)**
  - **Output latch**

**SPL**: Pin status designation bit of the standby control register (STBC)

- **To each input**
  - [only for the BSI1 and BSK1 pins]

- **From each output**
  - [only for the BSO1 and BSK1 pins]

- **From each output enable**
  - [only for the BSO1 and BSK1 pins]

- **Pin**
  - **P65/BSK1**
  - **P66/BSI1**
  - **P67/BSO1**

- **Pull-up resistor (optional)**
  - **about 50 kΩ (5 V)**

- **Output latch Pin**

- **Stop, watch mode (SPL=1)**

- **Pch**

**Reference:**

If "Pull-up resistor present" is selected in option settings, the pin status during reset or in stop or watch mode (SPL = 1) becomes "H."

---

#### Figure 10.3-2 Block Diagram of Pins Related to the Serial I/O with 1-byte Buffer (P73-P75)

![Block Diagram of Pins Related to the Serial I/O with 1-byte Buffer (P73-P75)](image)

- **PDR (port data register)**
  - **PDR read**
  - **PDR read (for bit processing commands)**
  - **PDR write**

- **DDR (port direction register)**
  - **Output latch**

**SPL**: Pin status designation bit of the standby control register (STBC)

- **To each input**
  - [only for the BSI2 and BSK2 pins]

- **From each output**
  - [only for the BSO2 and BSK2 pins]

- **From each output enable**
  - [only for the BSO2 and BSK2 pins]

- **Pin**
  - **P73/BSK2**
  - **P74/BSI2**
  - **P75/BSO2**

- **Pull-up resistor (optional)**
  - **about 50 kΩ (5 V)**

- **Output latch Pin**

- **Stop or watch mode (SPL=1)**

- **Pch**

**Reference:**

If "Pull-up resistor present" is selected in option settings, the pin status during reset or in stop or watch mode (SPL = 1) becomes "H."
10.4 Registers of the Serial I/O with 1-byte Buffer

This section explains the registers related to the serial I/O with 1-byte buffer.

- Registers Related to the Serial I/O with 1-byte Buffer

**Figure 10.4-1  Registers Related to the Serial I/O with 1-byte Buffer**

<table>
<thead>
<tr>
<th>SBMR (serial mode register with 1-byte buffer)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 2 2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>00000000</td>
</tr>
<tr>
<td></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>
<td>R/W</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBDR (serial data register with 1-byte buffer)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 2 5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>XX...XX...</td>
</tr>
<tr>
<td></td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td>R/R</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBFR (serial flag register with 1-byte buffer)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 2 3</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>XX...00...</td>
</tr>
<tr>
<td></td>
<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></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBUFW (serial buffer write register)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 2 4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>XX...XX...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBUFR (serial buffer read register)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 2 4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>XX...XX...</td>
</tr>
<tr>
<td></td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td>R/R/R/R</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
R: Read-only
W: Write-only
—: Unused
X: Undefined
### 10.4.1 Serial Mode Register with 1-byte Buffer (SBMR)

The serial mode register with 1-byte buffer (SBMR) is a register used to enable/disable operations of the serial I/O with 1-byte buffer, select the shift clock mode, set the transfer direction, control interrupts, and enable/disable interrupts.
### Table 10.4-1 Functions of Each Bit of the Serial Mode Register with 1-byte Buffer (SBMR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 7</strong></td>
<td>SOVE: overrun error interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td>SEME: data empty interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 5</td>
<td>SFLE: data full interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 4</td>
<td>SOE: serial data output enable bit</td>
</tr>
<tr>
<td>Bit 3</td>
<td>SEN: serial I/O shift operation enable bit</td>
</tr>
<tr>
<td>Bit 2</td>
<td>CKS1, CKS0: shift clock selection bit</td>
</tr>
<tr>
<td>Bit 1</td>
<td>BDS: transfer direction selection bit</td>
</tr>
</tbody>
</table>

**Bit 7**
This is a bit to enable/disable interrupt request output to the CPU if an overrun error occurs during receiving serial data or sending/receiving data.
If this bit and the overrun error interrupt request flag bit (SBFR: SOVF) of the serial flag register with 1-byte buffer are "1", an interrupt request is output.

**Bit 6**
This is a bit to enable/disable interrupt request output to the CPU if there is no transmitting data (data empty) in the serial buffer write register (SBUFW) when receiving serial data.
If this bit and the data empty interrupt request flag bit (SBFR: SEMF) of the serial flag register with 1-byte buffer are "1", an interrupt request is output.

**Bit 5**
This is a bit to enable/disable interrupt request output to the CPU if there is receiving data (data full) in the serial buffer read register (SBUFUR) when receiving serial data or sending/receiving serial data.
If this bit and the data full interrupt request flag bit (SBFR: SFLF) of the serial flag register with 1-byte buffer are "1", an interrupt request is output.

**Bit 4**
If this bit is "0", the P67/BSO1 pin becomes a general-purpose I/O port (P67). If this bit is "1", the P67/BSO1 pin becomes a serial data output pin (BSO1 or BSO2).

**Reference:**
The P67/BSO1 and P73/BSK2 output pin can be switched by setting the serial port switch register (SSEL).

**Bit 3**
- Bit to enable/disable the shift operation of the serial I/O.
- If transfer data is written to the SBUFW register when this bit is "1", the shift operation starts to transfer data. Shift operations can be stopped by changing this bit to "0". At this point, each flag of the SBFR register is set to its initial value.

**Bit 2**
- Select the shift clock from three internal shift clock modes or one external shift clock mode.
- If these bits accept a value other than "00B", one of the internal shift clock modes is selected and the shift clock is output from the BSK1 or BSK2 pin.
- If these bits accept the value of "00B", the external shift clock mode is selected. If a pin is set to the shift clock input(*1), the shift clock is input from the BSK1 or BSK2 pin.

**Reference:**
The P65/BSK1 and P73/BSK2 output pins can be switched by setting the serial port switch register (SSEL)(*1).

**Bit 1**
Bit to select whether serial data should be sent starting with the lowest-order bit (LSB-first, BDS=0) or highest-order bit (MSB-first, BDS=1).

**Note:**
If this bit is rewritten after data is written to the serial buffer write register (SBUFW), the data becomes invalid.

---

*1: For the serial I/O port switching and pin settings, see Section 11.5 "I/O Pin Setting of the Serial I/O Ports."
10.4.2 Serial Data Register with 1-byte Buffer (SBDR)

The serial data register with 1-byte buffer (SBDR) is a register used to retain transfer data of the serial I/O with 1-byte buffer. This register functions as a transmitting data register during serial output operation and as a receiving data register during serial input operation.

Serial Data Register with 1-byte Buffer (SBDR)

Figure 10.4-3 "Serial Data Register with 1-byte Buffer (SBDR)" shows the bit configuration of the serial data register with 1-byte buffer.

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 5h</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>XXXXXXXXh</td>
</tr>
</tbody>
</table>

R: Read-only
X: Undefined
10.4.3 Serial Flag Register with 1-byte Buffer (SBFR)

The serial flag register with 1-byte buffer (SBFR) is used to retain the interrupt source flags of data full, data empty, and overrun error that may occur when sending or receiving data with the serial I/O with 1-byte buffer.

Serial Flag Register with 1-byte Buffer (SBFR)

```
<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXXX</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
R: Read-only
- : Unused
- : Initial value
```

- **SEMF**: Data empty interrupt request flag bit
  - 0: There is data waiting for transfer in the serial buffer write register (SBUFW).
  - 1: There is no data waiting for transfer in the serial buffer write register (SBUFW).

- **SOVF**: Overrun error interrupt request flag bit
  - 0: No overrun error occurred. Clears this bit.
  - 1: An overrun error occurred. No change. No affect on others.
## 10.4 Registers of the Serial I/O with 1-byte Buffer

### Table 10.4-2 Functions of Each Bit of the Serial Flag Register with 1-byte Buffer (SBFR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 7, Bit 6, Bit 5, Bit 4, Bit 3 | Unused bit  
- Values are not defined when these bits are read.  
- Writing these bits does not affect operations. |
| Bit 2             | SOVF:  
overrun error interrupt request flag bit  
- If the data received last has been loaded in the serial buffer read register (SBUFR) but has not been read and if subsequent data received is loaded in SBUFR when receiving serial data or sending/receiving serial data, this bit is set to "1". If this bit and the overrun error interrupt request enable bit (SBMR: SOVE) of the serial mode register with 1-byte buffer are "1", an interrupt request is output.  
- In a write cycle, this bit is cleared if "0" is written. If "1" is written, this bit has no influence on operation.  
**Note:**  
- "1" is always read when the read-modify-write type commands are read.  
- Shift operations will continue if this bit is set to "1".  
- This bit is cleared to "0" when the transfer is stopped (SBMR: SEN=0). |
| Bit 1             | SEMF:  
data empty interrupt request flag bit  
- If there is no data waiting for transfer to the serial data register with 1-byte buffer (SBDR) in the serial buffer write register (SBUFWR) when sending/receiving serial data, this bit is set to "1". This bit can be cleared to "0" by writing transmitting data to SBUFWR. If this bit and the data empty interrupt enable bit (SBMR: SEME) of the serial mode register with 1-byte buffer are "1", an interrupt request is output.  
**Note:**  
- "1" is always read when the read-modify-write type commands are read.  
- Since this bit is "1" before starting data transfer, an interrupt request is generated immediately after enabling the data empty interrupt (SBMR: SEME=1). This bit is set to "1" when the data transfer is stopped (SBMR: SEN=0). |
| Bit 0             | SFLF:  
data full interrupt request flag bit  
- If there is data in the serial buffer read register (SBUFR) and the data has not been read when receiving serial data or sending/receiving serial data, this bit is set to "1". This bit can be cleared to "0" by reading the data in SBUFR. If this bit and the data full interrupt enable bit (SBMR: SFLE) of the serial mode register with 1-byte buffer are "1", an interrupt request is output.  
**Note:**  
- "1" is always read when the read-modify-write type commands are read.  
- This bit is cleared to "0" when the transfer is stopped (SBMR: SEN=0). |
10.4.4 Serial Buffer Write Register (SBUFW)/Serial Buffer Read Register (SBUFR)

The serial buffer register (SBUFW/R) is a buffer register for the serial data register (SBDR) with 1-byte buffer to read/write serial data. When writing transfer data, it is written to the serial buffer write register (SBUFW). When reading transfer data, it is read from the serial buffer read register (SBUFR). The SBUFW register and SBUFR register are located at the same address.

Serial Buffer Write Register (SBUFW)/Serial Buffer Read Register (SBUFR)

Figure 10.4-5 "Serial Buffer Write Register (SBUFW)/Serial Buffer Read Register (SBUFR)" shows the bit configuration of the serial buffer register.

The serial buffer register (SBUFW/R) is composed of the serial buffer write register (SBUFW) and serial buffer read register (SBUFR) located at the same address. When writing data, the SBUFW register is accessed; when reading data, the SBUFR register is accessed.
10.5 Interrupts of the Serial I/O with 1-byte Buffer

The following interrupt sources are available for the serial I/O with 1-byte buffer:
- If a receipt overrun occurs when receiving data or sending/receiving data.
- If the serial buffer write register (SBUFW) becomes empty when sending/receiving data.
- If receiving data is stored in the serial buffer read register when receiving data or sending/receiving data.

IRQ4 is generated by the serial I/O with 1-byte buffer as an interrupt request.

Table 10.5-1 "Interrupt Control Bits and Interrupt Sources of the Serial I/O with 1-byte Buffer" lists the interrupt request flag bits, interrupt request output enable bits, and interrupt sources of the serial I/O with 1-byte buffer.

### Table 10.5-1 Interrupt Control Bits and Interrupt Sources of the Serial I/O with 1-byte Buffer

<table>
<thead>
<tr>
<th>Interrupt generation condition (flag)</th>
<th>Receipt overrun</th>
<th>Write data empty</th>
<th>Read data full</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt request flag bit</td>
<td>SBFR: SOVF</td>
<td>SBFR: SEMF</td>
<td>SBFR: SFLF</td>
</tr>
<tr>
<td>Interrupt request enable bit</td>
<td>SBMR: SOVE</td>
<td>SBMR: SEME</td>
<td>SBMR: SFLE</td>
</tr>
<tr>
<td>Interrupt source</td>
<td>When there is receiving data that has not been read in the serial buffer read register (SBUFR), new receiving data is overwritten.</td>
<td>No transmitting data waiting for transfer in the serial buffer write register (SBUFW)</td>
<td>There is receiving data that has not been read in the serial buffer read register (SBUFR).</td>
</tr>
</tbody>
</table>

Figure 10.5-1 "Timing of Interrupt Request Output (for 8-bit Transfer)" to Figure 10.5-3 "Timing of Interrupt Request Output (Overrun Error)" show the timing of interrupt request output.
If an interrupt by the SFLF flag is enabled

If an interrupt by the SFLF/SEMF flag is enabled

SEMF interrupt enabled
SFLF interrupt enabled

Figure 10.5-1  Timing of Interrupt Request Output (for 8-bit Transfer)

Figure 10.5-2  Timing of Interrupt Request Output (for Continuous Transfer)
If an interrupt by the SOVF flag is enabled

<table>
<thead>
<tr>
<th>Register</th>
<th>Interrupt Request Output (Overrun Error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSK</td>
<td></td>
</tr>
<tr>
<td>SBUFR read</td>
<td></td>
</tr>
<tr>
<td>SBUFW write</td>
<td></td>
</tr>
<tr>
<td>SBUFW load</td>
<td></td>
</tr>
<tr>
<td>SBUFR load</td>
<td></td>
</tr>
<tr>
<td>SEMF</td>
<td></td>
</tr>
<tr>
<td>SFLF</td>
<td></td>
</tr>
<tr>
<td>SOVF</td>
<td></td>
</tr>
<tr>
<td>IRQ</td>
<td></td>
</tr>
<tr>
<td>BSO/BSI</td>
<td></td>
</tr>
</tbody>
</table>

Table 10.5-2  Register and Vector Table Related to the Serial I/O with 1-byte Buffer

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ4</td>
<td>ILR2 (007D)</td>
<td>L41 (bit 1)</td>
</tr>
</tbody>
</table>

For interrupt operations, see Section 3.4.2 "Flow of Interrupt Operation"
10.6 Explanation of Operations of the Serial I/O with 1-byte Buffer

This section explains the operations of the serial I/O with 1-byte buffer.

- Operations of the Serial I/O with 1-byte Buffer

  - Selection of the sending/receiving mode
    The serial I/O with 1-byte buffer sends and receives 8-bit data as internal operations simultaneously. To send or receive data cannot be selected, but by stopping (turning off the output buffer) external output of transmitting data or by retaining (not reading) receiving data, send or read operations only can be enabled. If only receive operations are enabled, dummy transmitting data must be written.

  - Operating clock mode
    Serial operations are driven by an internal or an external shift clock. To connect the serial I/O with 1-byte buffer mutually, select the internal shift clock operation for one side and the external shift clock operation for the other side.

  - Start/Stop of serial operations
    If data is written to the serial buffer register (SBUFW/R) after enabling serial operations (SBMR: SEN=1), transmitting data is immediately loaded from the SBUFW register to the SBDR register and shift clock output is enabled if data is not transferred.

    In the SBDR register, 1-bit data is sent and received simultaneously by shifting data one bit at a time synchronized with the falling edge of the shift clock. If data sending is enabled (SBMR: SOE=1), transmitting data is output to the outside from the pin. When the last bit is sent/received at the rise time of the 8th clock, data of the SBDR register is loaded to the SBUFR register as receiving data.

    At this point, the following operations are performed:
    - If new transmitting data is not in the SBUFW register (SBFR: SFLF=0), transmitting data is not loaded and shift clock output is stopped to terminate transfer operations (for 8-bit transfer).
    - If new transmitting data has been written to the SBUFW register (SBFR: SFLF=1), receiving data is loaded to the SBUFR register and transmitting data is then loaded from the SBUFW register to the SBDR register to continue the shift operation (for continuous transfer).

    If the serial operation is stopped (SBMR: SEN=0) during transfer, the shift operation is stopped. At this point, the values of the SBDR register and SBUFR register are retained and so can be used after reading out. However, flags have been cleared and so the serial operation cannot be resumed by setting SEN=1 again.
10.6 Explanation of Operations of the Serial I/O with 1-byte Buffer

10.6.1 Timing for the Shift/Load Operation of the Serial I/O with 1-byte Buffer

The timing for the I/O shift operation of the serial I/O with 1-byte buffer and the timing for the data load operation between the buffer register and data register are explained below.

- **Shift/Load Operations**
  The serial I/O with 1-byte buffer sends one bit and receives one bit almost simultaneously by bit-shifting serial data register with a 1-byte buffer (SBDR) synchronized with the shift clock. Data can be sent and received continuously by a series of operations at every 8th clock of loading data in the serial buffer write register (SBUFW) to the SBDR register after loading data in the SBDR register to the serial buffer read register (SBUFR).

  Figure 10.6-1 “Shift/Load Operation of the Serial I/O 1-byte Buffer” shows the shift/load operation of the serial I/O 1-byte buffer.

- **I/O Shift Timing**
  Data is output from the serial output pin (BSO1 or BSO2) synchronized with the falling edge of the shift clock. Data of the serial input pin (BSI1 or BSI2) is input synchronously with the rising edge of the shift clock.
CHAPTER 10 SERIAL I/O WITH 1-BYTE BUFFER

Figure 10.6-2 Shift Timing Diagram

<table>
<thead>
<tr>
<th>LSB-first (when BDS=&quot;0&quot;)</th>
<th>MSB-first (when BDS=&quot;1&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSK</td>
<td>BSK</td>
</tr>
<tr>
<td>BSI</td>
<td>BSI</td>
</tr>
<tr>
<td>BSO</td>
<td>BSO</td>
</tr>
<tr>
<td>#0 #1 #2 #3 #4 #5 #6 #7</td>
<td>#7 #6 #5 #4 #3 #2 #1 #0</td>
</tr>
</tbody>
</table>

Data Load Timing

If there is data in the SBUFW register at the rise time of the 8th clock of the shift clock when writing data to the SBUFW register for the first time after the shift operation is enabled (SBMR: SEN=1), the SBUFW load signal is issued. With this load signal, data in the SBUFW register is loaded to the SBDR register.

Also, at the rise time of every 8th clock, the SBUFR signal is always issued so that data in the SBDR register is loaded to the SBUFR register.

Figure 10.6-3 Load Timing Diagram (for 8-bit Transfer)

Figure 10.6-4 Load Timing Diagram (for Continuous Transfer)
10.6.2 Transfer Operation Flow of the Serial I/O with 1-byte Buffer

The following explains the transfer operation flow of the serial I/O with 1-byte buffer.

- **Transfer Operation Flow of the Serial I/O with 1-byte Buffer**
  
  Figure 10.6-5 “Transfer Operation Flow of the Serial I/O with 1-byte Buffer (When No Interrupt Is Used)” shows the transfer operation flow when no interrupt of the serial I/O with 1-byte buffer is used.

  The operation flow is one of the following:
  
  - One-byte transmission/receipt (Dummy transmitting data is written for one-byte receipt)
  - Continuous transmission/receipt (Dummy transmitting data is written for continuous receipt)
  - Continuous receipt (Receiving data is left unattended)
Figure 10.6-5 Transfer Operation Flow of the Serial I/O with 1-byte Buffer (when No Interrupt is Used)

Note:
When a high-speed shift clock is used, data cannot be transferred continuously using the flag check. Use the interrupts of data empty and data full.
10.7 Connection Example of the Serial I/O with 1-byte Buffer

The serial I/O with 1-byte buffer can be connected mutually so that serial input/output can be carried out in both directions. Furthermore, by connecting a modem signal output circuit to the serial I/O with 1-byte buffer, serial transmitting data can be output as a modem signal (before modulation).

- When Serial Input/Output Is Carried Out in Both Directions

  To wire the serial I/O with 1-byte buffer (BSIO) mutually, select the internal shift clock operation for one side and the external shift clock operation for the other side.

**Figure 10.7-1  Connection Example of the Serial I/O with 1-byte Buffer**

![Diagram of BSIO connection example]
CHARTER 10 SERIAL I/O WITH 1-BYTE BUFFER

Figure 10.7-2 Bi-directional I/O Operation of the Serial I/O with 1-byte Buffer

*1: Because BSIO-B is operated by an external clock, write first output data (dummy data if no output data is available) to BSIO-B prior to BSIO-A to make it wait for the arrival of a shift clock.

*2: If receiving data is required, read receiving data after checking the data full interrupt request flag (SFLF) during transfer or a data full interrupt occurs.
When the Modem Signal Is Output by Connecting a Modem Signal Output Circuit

Figure 10.7-3 Connection Image of the Serial I/O with 1-byte Buffer and Modem Signal Output Circuit

The serial I/O with 1-byte buffer and modem signal output circuit are connected inside a chip. Transmitting data can be converted into modem signal (before modulation) before outputting it from the pin only by setting the modem signal control register (MODC).

The pin that actually outputs the signal can be switched by the serial I/O port switch register (SSEL).

To use a model signal output circuit, the setting in Figure 10.7-4 "Setting for Connection of the Serial I/O with 1-byte Buffer and Modem Signal Output Circuit" is required.

Figure 10.7-4 Setting for Connection of the Serial I/O with 1-byte Buffer and Modem Signal Output Circuit

For the serial I/O port switching and pin settings, see Section 11.5 "I/O Pin Setting of the Serial I/O Ports"
10.8 State Transition Diagram of the Serial I/O with 1-byte Buffer

This section explains the state transition diagram of the serial I/O with 1-byte buffer.

- State Transition Diagram of the Serial I/O with 1-byte Buffer

---

**Figure 10.8-1 State Transition Diagram of the Serial I/O with 1-byte Buffer**

- (1): Power-on reset or system reset
- (2): Shift operation enabled (SEN=1)
- (3): Shift operation start (SBUF written)
- (4): Data write during continuous transfer
- (5): Operation end
- (6): Retransfer
- (7): Overrun error
- (8): Operation continued by clearing the overrun error flag
- (9): Operation end with the overrun error flag set
- (10): Shift operation stop (SEN=0)
10.9 Precautions when Using the Serial I/O with 1-byte Buffer

This section explains precautions when using the serial I/O with a 1-byte buffer.

- Precautions When Using the Serial I/O with 1-byte Buffer

  For general precautions, see Section 9.9 "Precaution on Use of 8-Bit Serial I/O". The precautions specific to the serial I/O with 1-byte buffer are as follows.

  - **Precautions when making settings in a program**
    - When operating sending data only, receiving data can be left unattended. However, since the interrupt request flag bits of data full and overrun are set to "1", it is necessary to set the interrupt request enable bits (SBMR: SOVE, SFLE) corresponding to these bits to "0". Also, clear the data full interrupt request flag bit after reading receiving data or clear the overrun error interrupt request flag bit by writing to the SBFR register, if necessary.
    - Also when operating receiving data only, it is necessary to write dummy transmitting data. If transmitting data is loaded (SBFR: SFLF=1) to the serial buffer read register (SBUFR) when the serial buffer write register (SBUFW) is empty (SBFR: SEMF=1), a no receive operation is performed hereafter even if the shift clock is input.
    - When using a high-speed shift clock, data cannot be transferred continuously using the flag check. Use an interrupt, such as data empty or data full. The shift clock speed may be restricted depending on the programs.
10.10 Program Example of the Serial I/O with 1-byte Buffer

The following explains a program example of the serial I/O with 1-byte buffer.

Program Example of the Serial I/O with 1-byte Buffer

- **Processing specifications**
  - 8-bit serial data (55H) will be output from the BSO pin (P67/BSO1) of the serial I/O with 1-byte buffer. When the transfer ends, an interrupt occurs.
  - Transfer data will be set again by an interrupt processing routine for continuous output.
  - The serial I/O with 1-byte buffer shall be operated by an internal shift clock, and the shift clock will be output from the BSK pin.
  - The following shows the transfer rate and interrupt generation cycle if the shift clock is 32 \( t_{\text{inst}} \) (\( t_{\text{inst}} \): CPU operating clock divided by 2) and the main clock source oscillation frequency is 8 MHz:
    
    Transfer rate=8 MHz/4/32=62.5 bps, Interrupt cycle=4 x 32 x 8/8 MHz=128 \( \mu \)s
10.10 Program Example of the Serial I/O with 1-byte Buffer

- **Coding example**

```assembly
SBMR EQU 0022H ; Address of the BSIO mode register
SBDR EQU 0025H ; Address of the BSIO data register
SBFR EQU 0023H ; Address of the BSIO flag register
SIOF EQU SBMR:7 ; Definition of the interrupt request flag bit
SEN EQU SBMR:3 ; Definition of the shift operation enable bit
SOVF EQU SBFR:2 ; Definition of the overrun error interrupt request flag bit
ILR2 EQU 007DH ; Address of the interrupt level setting register 2
INT_V DSEG ABS ; [DATA SEGMENT]
ORG OFFF2H
IRQ4 DW WARI ; Interrupt vector setting
INT_V ENDS

;-----------Main program--------------------------------------------------------------------------------------- ----

CSEG ; [CODE SEGMENT]
; Assuming that the stack pointer (SP) and others have been initialized.

CLRI ; Disable the interrupts
CLRB SEN ; Stop the shift operation
CLRB SOVF ; Clear the overrun error
MOV ILR2,#11111101B ; Set the interrupt level (level 1)
MOV SBDR,#55H ; Set transfer data (55H)
MOV SBMR,#01011110B ; Enable the serial data output (SO1) Enable the shift operation Select 32 tinst and LSB-first
SETI ; Enable the interrupts

;-----------Interrupt processing routine--------------------------------------------------------------------------------------- --

WARI CLRB SIOF ; Clear the interrupt request flag
PUSHW A
XCHW A,T ; Save A and T
PUSHW A
MOV SBDRI,#55H ; Set transfer data (55H) again

User processing

CLRB SIOF ; Clear the interrupt request flag
POPW A
XCHW A,T ; Restore A and T
POPW A
RETI

END
```
CHAPTER 10 SERIAL I/O WITH 1-BYTE BUFFER
CHAPTER 11  SERIAL I/O PORT SWITCH CIRCUIT

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

11.1 "Overview of the Serial I/O Port Switch Circuit"
11.2 "Configuration of the Serial I/O Port Switching Circuit"
11.3 "Pins of the Serial I/O Switching Circuit"
11.4 "Register of the Serial I/O Port Switching Circuit"
11.5 "I/O Pin Setting of the Serial I/O Ports"
11.6 "Precautions when Using the Serial I/O Port Switching Circuit"
11.7 "Program Example of the Serial I/O Port Switching Circuit"
CHAPTER 11 SERIAL I/O PORT SWITCH CIRCUIT

11.1 Overview of the Serial I/O Port Switch Circuit

The serial I/O port switch circuit provides a function to switch the I/O ports of the 8-bit serial I/O (SSI0) and serial I/O with 1-byte buffer (BSIO). By switching the output port, either the CMOS output or N-channel open-drain output can be selected as an output form.

■ Serial I/O Port Switching Function

The serial I/O port switching function is a function to switch the I/O ports of the 8-bit serial I/O and serial I/O with 1-byte buffer.

- Three ports for the shift clock I/O, data input, and data output are switched together. It is not possible to switch only one port.
- In the 8-bit serial I/O, the I/O ports are switched between port3 (CMOS output) and port7 (Nch open-drain output).
- In the serial I/O with 1-byte buffer, the I/O ports are switched between port6 (CMOS output) and port7 (Nch open-drain output).

Table 11.1-1 "I/O Port Switching of the Serial I/O" lists information about I/O port switching of the serial I/O.

Table 11.1-1 I/O Port Switching of the Serial I/O

<table>
<thead>
<tr>
<th>Serial I/O</th>
<th>Shift clock I/O</th>
<th>Data input</th>
<th>Data output</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-bit serial I/O</td>
<td>P35/SK1 *1</td>
<td>P36/SI1</td>
<td>P37/SO1 *1</td>
</tr>
<tr>
<td></td>
<td>P70/SK2 *2</td>
<td>P71/SI2</td>
<td>P72/SO2 *2</td>
</tr>
<tr>
<td>Serial I/O with 1-byte buffer</td>
<td>P65/BSK1 *1</td>
<td>P66/BSI1</td>
<td>P67/BSO1 *1</td>
</tr>
<tr>
<td></td>
<td>P73/BSK2 *2</td>
<td>P74/BSI2</td>
<td>P75/BSO2 *2</td>
</tr>
</tbody>
</table>

*1: Output is in the CMOS form.
*2: Output is in the N-channel open-drain form.

: Initial setting
11.2 Configuration of the Serial I/O Port Switching Circuit

The serial I/O port switching circuit consists of the following three parts:
- I/O port selector of the 8-bit serial I/O
- I/O port selector of the serial I/O with 1-byte buffer
- Serial I/O port switching register (SSEL)

Block Diagram of the Serial I/O Port Switching Circuit

Figure 11.2-1 Block Diagram of the Serial I/O Port Switching Circuit
CHAPTER 11 SERIAL I/O PORT SWITCH CIRCUIT

- **I/O port selector of the 8-bit serial I/O**
  Switches the I/O ports of the input data, output data, and shift clock of the 8-bit serial I/O between port3 (initial state) and port7 by setting the SSEL register.

- **I/O port selector of the serial I/O with 1-byte buffer**
  Switches the I/O ports of the input data, output data, and shift clock of the 8-bit serial I/O between port6 (initial state) and port7 by setting the SSEL register.

- **Serial I/O switching register (SSEL)**
  Selects the I/O ports. This register also enables/disables the serial data output and shift clock output when port7 is selected as the I/O port of the serial I/O.

  **Note:**
  If the data input source of the modem output module is a serial I/O with 1-byte buffer, the shift clock of the serial I/O with 1-byte buffer is supplied from the modem output module and data output is input to the modem output module, in which case the shift clock and data cannot be extracted from outside through the pins.
11.3 Pins of the Serial I/O Switching Circuit

This section explains the pins related to the serial I/O port switching circuit and block diagrams of the pins.

- Pins Related to the Serial I/O Port Switching Circuit

  The following are the pins related to the serial I/O port switching circuit: P35/SK1-P37/SO1, P65/BSK1-P67/BSO1, P70/SK2-P75/BSO2.

- P35/SK1-P37/SO1, P65/BSK1-P67/BSO1
  P35/SK1-P37/SO1, P65/BSK1-P67/BSO1 are serial I/O pins that also function as a general-purpose I/O port (CMOS output).
  In the initial setting, these pins function as a serial I/O pin.

- P70/SK2-P75/BSO2
  P70/SK2-P75/BSO2 are serial I/O pins that also function as a general-purpose I/O port (N-channel open-drain output).
  In the initial setting, these pins function as a general-purpose I/O port.
  To use these pins as an I/O pin of the serial I/O, settings for the 8-bit serial I/O or serial I/O with 1-byte buffer and as well as for the I/O pins are required. For more details, see Section 11.5 "I/O Pin Settings of the Serial I/O Ports."
CHAPTER 11 SERIAL I/O PORT SWITCH CIRCUIT

Block Diagram of Pins Related to the Serial I/O Port Switching Circuit

Figure 11.3-1 Block Diagram of Pins Related to the Serial I/O Port Switching Circuit (P35-P37, P65-P67)

Figure 11.3-2 Block Diagram of Pins Related to the Serial I/O Port Switching Circuit (P70-P75)

Reference:

If "Pull-up resistor present" is selected in option settings, the pin status during reset or in stop or watch mode (SPL = 1) is "H."
11.4 Register of the Serial I/O Port Switching Circuit

This section explains the register related to the serial I/O port switching circuit

- Register Related to the Serial I/O Port Switching Circuit

![Figure 11.4-1 Register Related to the Serial I/O Port Switching Circuit](image)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 Dₜ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XX000000₀</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled

- : Unused
X: Undefined
11.4.1 Serial I/O Port Switching Register (SSEL)

The serial I/O port switching register (SSEL) is a register used to switch the I/O ports of the 8-bit serial I/O (SSIO) and serial I/O with 1-byte buffer (BSIO) and control external output pins.

Serial I/O Port Switching Register (SSEL)

**Figure 11.4-2 Serial I/O Port Switching Register (SSEL)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 D1</td>
<td>—</td>
<td>—</td>
<td>BSKE</td>
<td>BSOE</td>
<td>BPSL</td>
<td>SSKE</td>
<td>SSOE</td>
<td>SPSL</td>
<td>XX000000n</td>
</tr>
</tbody>
</table>

- **SPSL**: SSIO I/O port selection bit
  - 0: Use P35/SK1, P36/SI1, and P37/SO1 as ports for the 8-bit serial I/O
  - 1: Use P70/SK2, P71/SI2, and P72/SO2 as ports for the 8-bit serial I/O

- **SSOE**: SSIO serial data output enable bit
  - 0: Use P72/SO2 as a general-purpose port
  - 1: Use P72/SO2 as a serial data output pin

- **SSKE**: SSIO shift clock output enable bit
  - 0: Use P70/SK2 as a general-purpose port or shift clock input pin
  - 1: Use P70/SK2 as a shift clock output pin

- **BPSL**: BSIO I/O port selection bit
  - 0: Use P65/BSK1, P66/BSI1, and P67/BSO1 as a serial I/O port with 1-byte buffer
  - 1: Use P73/BSK2, P74/BSI2, and P75/BSO2 as a serial I/O port with 1-byte buffer

- **BSOE**: BSIO serial data output enable bit
  - 0: Use P75/BSO2 as a general-purpose port
  - 1: Use P75/BSO2 as a serial data output pin

- **BSKE**: BSIO shift clock output enable bit
  - 0: Use P73/BSK2 as a general-purpose port or shift clock input pin
  - 1: Use P73/BSK2 as a shift clock output pin

R/W: Read/Write enabled
- : Unused
X: Undefined
!: Initial value
# 11.4 Register of the Serial I/O Port Switching Circuit

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 7</strong></td>
<td>Unused bit</td>
</tr>
<tr>
<td><strong>Bit 6</strong></td>
<td>Unused bit: • Values when these bits are read are not defined. • Writing to these bits does not affect operations.</td>
</tr>
<tr>
<td><strong>Bit 5</strong></td>
<td>BSKE: BSIO shift clock output enable bit • Bit to control input/output of the shift clock for the serial I/O with 1-byte buffer • If this bit is &quot;0&quot;, the P73/BSK2 pin functions as a shift clock input pin. If this bit is &quot;1&quot;, the pin functions as a shift clock output pin. <strong>Note:</strong> • If the BSK2 pin is a shift clock input pin (BSKE=0), set the P73/BSK2 pin as an input port (PDR7: bit3=0) using the port data register. • If output is enabled when the BSK2 pin is a shift clock output pin (BSKE=1), the P73/BSK2 pin functions as a shift clock output pin regardless of the value of the port data register.</td>
</tr>
<tr>
<td><strong>Bit 4</strong></td>
<td>BSOE: BSIO serial data output enable bit If this bit is &quot;0&quot;, the P75/BSO2 pin functions as a general-purpose port (P75). If this bit is &quot;1&quot;, the pin functions as a serial data output pin (SO) for the serial I/O with 1-byte buffer. <strong>Reference:</strong> To make the P75/BSO2 pin a serial data output pin, write &quot;1&quot; to the BPSL bit. If the pin functions as a serial data output pin (BSOE=1), it functions as a serial data output pin regardless of the value of the data register for the general-purpose port (P75).</td>
</tr>
<tr>
<td><strong>Bit 3</strong></td>
<td>BPSL: BSIO I/O port selection bit If this bit is &quot;0&quot;, the I/O port for the serial I/O with 1-byte buffer becomes port6 (P65/BSK1, P66/BSI1, and P67/BSO1). If this bit is &quot;1&quot;, the I/O port becomes port7 (P73/BSK2, P74/BSI2, and P75/BSO2).</td>
</tr>
<tr>
<td><strong>Bit 2</strong></td>
<td>SSKE: SSIO shift clock output bit • Bit to control input/output of the shift clock for the 8-bit serial I/O • If this bit is &quot;0&quot;, the P70/SK2 pin functions as a shift clock input pin. If this bit is &quot;1&quot;, the pin functions as a shift clock output pin. <strong>Note:</strong> • If the SK2 pin is a shift clock input pin (SSKE=0), set the P70/SK2 pin as an input port (PDR7: bit0=0) using the port data register. • If output is enabled when the SK2 pin is a shift clock output pin (SSKE=1), the P70/SK2 pin functions as a shift clock output pin regardless of the value of the port data register.</td>
</tr>
<tr>
<td><strong>Bit 1</strong></td>
<td>SSOE: SSIO serial data output enable bit If this bit is &quot;0&quot;, the P72/SO2 pin functions as a general-purpose port (P72). If this bit is &quot;1&quot;, the pin functions as a serial data output pin (SO2) for the 8-bit serial I/O. <strong>Reference:</strong> To make the P72/SO2 pin a serial data output pin, write &quot;1&quot; to the SPSL bit.</td>
</tr>
<tr>
<td><strong>Bit 0</strong></td>
<td>SPSL: SSIO I/O port selection bit If this bit is &quot;0&quot;, the I/O port for the 8-bit serial I/O becomes port3 (P35/SK1, P36/SI1, and P37/SO1). If this bit is &quot;1&quot;, the I/O port becomes port7 (P70/SK2, P71/SI2, and P72/SO2).</td>
</tr>
</tbody>
</table>
11.5 I/O Pin Setting of the Serial I/O Ports

The I/O pins of the 8-bit serial I/O and serial I/O with 1-byte buffer also function as a general-purpose I/O port. To use the pins as an I/O pin of the serial I/O, the pins must be controlled by using the serial I/O port switching register, each serial I/O control register, or the direction register or data register of a general-purpose I/O port.

I/O Pin Setting of the Serial I/O Ports

To use an 8-bit serial I/O (SSIO) or serial I/O with 1-byte buffer (BSIO), settings for each serial I/O and for the I/O pins are required. First, set the serial I/O as an I/O port by setting the serial I/O port switching register (SSEL), then, for the serial data input (SI) or shift clock input (SK), make a setting as an input pin. For serial data output (SO) or SK output, make a setting as an output pin.

Switching of the I/O ports

The I/O ports of SI, SO, and SK of each serial I/O are switched together by the serial I/O port switching register (SSEL).

- In the 8-bit serial I/O, if the SSIO I/O port selection bit (SSEL: SPSL) is set to "0", port3 is selected as the I/O port. If the bit is set to "1", port7 is selected as the I/O port.
- In the serial I/O with 1-byte buffer, if the BSIO I/O port selection bit (SSEL: BPSL) is set to "0", port6 is selected as the I/O port. If the bit is set to "1", port7 is selected as the I/O port.

Port3 and port6 are a CMOS I/O port and port7 is a Nch open-drain I/O port.

Settings of the input pins (SI, SK)

The input pins of each serial I/O function as a general-purpose I/O port and also as an SI input or SK input pin. To use the pin as an input pin, disable the operation as an output transistor of the general-purpose port to set the pin as an input port. Set the relevant bit of the port direction register (DDR3, DDR6) to "0" for port3 and port6, and set the relevant bit of the port direction register (DDR7) to "1" for port7 so that the pin functions as an input port.

Settings of the output pins (SO, SK)

The output pins of each serial I/O function not only as a general-purpose I/O port, but also as an SO output or SK output pin. These pins function as an output pin regardless of the general-purpose port setting if the SO/SK output is enabled in each serial I/O (if port3/port6 is selected) or the SO/SK output is enabled by the serial I/O port switching register (if port7 is selected).

Table 11.5-1 "I/O Pin Settings for the 8-bit Serial I/O Port" and Table 11.5-2 "I/O Pin Settings for the Serial I/O Port with 1-byte Buffer" list the I/O pin settings for each serial I/O port.
### Table 11.5-1 I/O Pin Settings for the 8-bit Serial I/O Port

<table>
<thead>
<tr>
<th>I/O port switching</th>
<th>SI</th>
<th>SO</th>
<th>SK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input</td>
<td>Output</td>
<td>Input</td>
</tr>
<tr>
<td>Port3 selection</td>
<td>I/O pin</td>
<td>P36/SI1</td>
<td>P37/SO1</td>
</tr>
<tr>
<td>Register setting</td>
<td>Serial I/O port switching</td>
<td>SSEL: SPSL=0 (port3 selected)</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>8-bit serial I/O</td>
<td>—</td>
<td>SSEL: SSOE=1 *1</td>
</tr>
<tr>
<td></td>
<td>I/O port3</td>
<td>DDR3: bit6=0 *2</td>
<td>—</td>
</tr>
<tr>
<td>Port7 selection</td>
<td>I/O pin</td>
<td>P71/SI2</td>
<td>P72/SO2</td>
</tr>
<tr>
<td>Register setting</td>
<td>Serial I/O port switching</td>
<td>SSEL: SPSL=1 (port7 selected)</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>8-bit serial I/O</td>
<td>—</td>
<td>SMR: SOE=1 *1</td>
</tr>
<tr>
<td></td>
<td>I/O port7</td>
<td>PDR7: bit1=1 *2</td>
<td>—</td>
</tr>
</tbody>
</table>

*1: Output pin control bit. Data output or shift clock output of the serial I/O is permitted regardless of the I/O port setting. Also, an output buffer in the serial I/O is permitted.

*2: Set as an input port by turning off the output transistor.

### Table 11.5-2 I/O Pin Settings for the Serial I/O Port with 1-byte Buffer

<table>
<thead>
<tr>
<th>I/O port switching</th>
<th>SI</th>
<th>SO</th>
<th>SCK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input</td>
<td>Output</td>
<td>Input</td>
</tr>
<tr>
<td>Port6 selection</td>
<td>I/O pin</td>
<td>P66/BSI1</td>
<td>P67/BSO1</td>
</tr>
<tr>
<td>Register setting</td>
<td>Serial I/O port switching</td>
<td>SSEL: BPSL=0 (port6 selected)</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Serial I/O with 1-byte buffer</td>
<td>—</td>
<td>SBMR: SSOE=1 *1</td>
</tr>
<tr>
<td></td>
<td>I/O port6</td>
<td>DDR6: bit6=0 *2</td>
<td>—</td>
</tr>
<tr>
<td>Port7 selection</td>
<td>I/O pin</td>
<td>P74/BSI2</td>
<td>P75/BSO2</td>
</tr>
<tr>
<td>Register setting</td>
<td>Serial I/O port switching</td>
<td>SSEL: BPSL=1 (port7 selected)</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Serial I/O with 1-byte buffer</td>
<td>—</td>
<td>SBMR: SOE=1 *1</td>
</tr>
<tr>
<td></td>
<td>I/O port7</td>
<td>PDR7: bit4=1 *2</td>
<td>—</td>
</tr>
</tbody>
</table>

*1: Output pin control bit. Data output or shift clock output of the serial I/O is permitted regardless of the I/O port setting. Also, an output buffer in the serial I/O is permitted.

*2: Set as an input port by turning off the output transistor.
11.6 Precautions when Using the Serial I/O Port Switching Circuit

This section explains the precautions to be taken when using the serial I/O port switching circuit.

■ Precautions When Using the Serial I/O Port Switching Circuit

○ Precautions when making settings in a program

• If port3 is selected as an I/O port of the 8-bit serial I/O, set P71 for "L" output (PDR7: bit1=0) or set up a pull-up resistor externally so that the serial data input pin (P71/SI2) of port7 is not in high impedance state. Conversely, if port7 is selected as an I/O port, set P36 for output (DDR3: bit6=1), so that the serial data input pin (P36/SI1) of port3 is not in high impedance state.

• If port6 is selected as an I/O port of the serial I/O with 1-byte buffer, set P74 for "L" output (PDR7: bit4=0) or set up a pull-up resistor externally so that the serial data input pin (P74/BSI2) of port7 is not in high impedance state. Conversely, if port7 is selected as an I/O port, set P66 for output (DDR6: bit6=1) so that the serial data input pin (P66/BSI1) of port6 is not in high impedance state.
11.7 Program Example of the Serial I/O Port Switching Circuit

The following explains a program example of the serial I/O port switching circuit.

Program Example of the Serial I/O Port Switching Circuit

- **Processing specifications**
  - The I/O port of the 8-bit serial I/O shall be port3, and the I/O port of the serial I/O with 1-byte buffer shall be port7.
  - Settings of each serial I/O are omitted.

- **Coding example**

```assembly
DDR3 EQU 000DH ; Address of the port direction register
PDR7 EQU 0014H ; Address of the port data register
SMR EQU 001CH ; Address of the 8-bit serial I/O mode register
SBMR EQU 0022H ; Address of the serial mode register with 1-byte buffer
SSEL EQU 003DH ; Address of the serial I/O port switching register
ILR2 EQU 007DH ; Setting of the interrupt level setting register 2
SPSL EQU SSEL:0 ; Definition of the SSIO I/O port selection bit
BPSL EQU SSEL:3 ; Definition of the BSIO I/O port selection bit
BSOE EQU SSEL:4 ; Definition of the BSIO serial data output enable bit
SSOE EQU SSEL:1 ; Definition of the SSIO serial data output enable bit

;--------------Main program -----------------------------------------------------------------------------------------------
CSEG ; [CODE SEGMENT]
; Assuming that the stack pointer (SP) and others have been initialized.
; Interrupts are also assumed to be initialized
MOV DDR3,#10111111B ; Set bit6 as input for SI input (DDR: bit6=0)
MOV PDR7,#11111111B ; Set bit4 as input for SI input (DDR: bit4=1)
CLR SPSL ; Set port3 as an 8-bit serial port with SSEL: SPSL=0
SET BPSL ; Set port7 as an I/O port with SSEL: BPSL=1
SET BSOE ; Enable the data output pin with SSEL: BSOE=1
SET SSOE ; Enable the data output pin with SSEL: SSOE=1

; User processing
;
ENDS
;--------------------------------------------------------------------------------------------------------------------------
END
```

CHAPTER 12  BUZZER OUTPUT

This chapter describes the functions and operations of the buzzer output.

12.1  "Overview of the Buzzer Output"
12.2  "Configuration of the Buzzer Output"
12.3  "Pins of the Buzzer Output"
12.4  "Register of the Buzzer Output"
12.5  "Program Example of the Buzzer Output"
12.1 Overview of the Buzzer Output

Seven types of output frequency (square waves) can be selected as buzzer output. The buzzer output can be used, for example, as a confirmation tone for key input.

Buzzer Output Function

This is a function to output the signal (square waves) as a confirmation tone.

- The output of seven types of output frequency can be selected or the output can be disabled.
- As buzzer output, four types of divided output are supplied from the time base timer and three types from the clock prescaler.

Reference:

If the supply source (time base timer or clock prescaler) of the clock selected for buzzer output is cleared, the buzzer output is affected because the divided output of the time base timer or clock prescaler is used directly as buzzer output.

Note:

Since the time base timer does not operate when the main clock oscillation is stopped (in sub-clock mode), do not select the divided output of the time base timer for buzzer output. If a one-system clock is selected in option settings, do not select the divided output of a clock prescaler.

Table 12.1-1 "Output Frequency" lists the seven types of output frequency that can be set by the buzzer output function.

<table>
<thead>
<tr>
<th>Clock supply source</th>
<th>Buzzer output frequency</th>
<th>Square wave output (kHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time base timer</td>
<td>$2^{13}/F_{CH}$</td>
<td>$F_{CH}/2^{13}$ (0.977 kHz)</td>
</tr>
<tr>
<td></td>
<td>$2^{12}/F_{CH}$</td>
<td>$F_{CH}/2^{12}$ (1.953 kHz)</td>
</tr>
<tr>
<td></td>
<td>$2^{11}/F_{CH}$</td>
<td>$F_{CH}/2^{11}$ (3.906 kHz)</td>
</tr>
<tr>
<td></td>
<td>$2^{10}/F_{CH}$</td>
<td>$F_{CH}/2^{10}$ (7.813 kHz)</td>
</tr>
<tr>
<td>Clock prescaler</td>
<td>$2^5/F_{CL}$</td>
<td>$F_{CL}/2^5$ (1.024 kHz)</td>
</tr>
<tr>
<td></td>
<td>$2^4/F_{CL}$</td>
<td>$F_{CL}/2^4$ (2.048 kHz)</td>
</tr>
<tr>
<td></td>
<td>$2^3/F_{CL}$</td>
<td>$F_{CL}/2^3$ (4.096 kHz)</td>
</tr>
</tbody>
</table>

$F_{CH}$: Main clock source oscillation frequency
$F_{CL}$: Sub-clock signal source oscillation frequency
Frequencies in the parentheses those when $F_{CH}$=8 MHz and $F_{CL}$= 32.768 kHz
12.1 Overview of the Buzzer Output

Reference:

Example of calculating the output frequency

If the main clock source oscillation frequency ($F_{CH}$) is 8 MHz and $F_{CH}/2^{11}$ is selected (BZ2, BZ1, BZ0=011B) as divide-by-the time base timer output by the buzzer register (BZCR), the output frequency output from the BZ pin is as follows:

$$\text{Output frequency} = \frac{F_{CL}}{2^{11}}$$
$$= \frac{8 \text{ MHz}}{2048}$$
$$\approx 3.906 \text{ kHz}$$
12.2 Configuration of the Buzzer Output

The buzzer output consists of the following two parts:
- Buzzer output selector
- Buzzer register (BZCR)

Block Diagram of the Buzzer Output

Figure 12.2-1 Block Diagram of the Buzzer Output

- **Buzzer output selector**
  Circuit to select one type of frequency from four types of frequency (square waves) output from the time base timer and three types of frequency output from the clock prescaler.

- **Buzzer register (BZCR)**
  Register to set the buzzer output frequency and enable the buzzer output.
  If the output frequency (except 000B) is set by the BZCR register, the buzzer output is enabled.
12.3 Pins of the Buzzer Output

This section explains the pin related to the buzzer output and provides a block diagram of the pin.

■ Pin Related to the Buzzer Output

The pin related to the buzzer output is the P31/BZ pin.

○ P31/BZ pin

This pin functions as a general-purpose I/O port (P31) and also as a buzzer output pin (BZ).

BZ:

Square waves for the buzzer of the specified frequency are output to this pin. If the buzzer output frequency is set (BZCR: other than BZ2, BZ1, BZ0=000b), the P31/BZ pin functions automatically as the BZ pin regardless of the output latch value and settings of the remote control sending frequency generation circuit.

■ Block Diagram Related to the Buzzer Output

Figure 12.3-1 Block Diagram of the P31/BZ Pin

SPL: Pin status designation bit of the standby control register (STBC)
12.4 Register of the Buzzer Output

This section explains the register related to the buzzer output.

Register Related to the Buzzer Output

![Figure 12.4-1 Register Related to the Buzzer Output]

BZCR (buzzer register)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 F</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>BZ2</td>
<td>BZ1</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
—: Unused
X: Undefined
# 12.4.1 Buzzer Register

The buzzer register (BZCR) is a register to select the output frequency of the buzzer and is also used to enable the buzzer output.

---

**Buzzer Register (BZCR)**

---

**Figure 12.4-2 Buzzer Register (BZCR)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 Fx</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>XXXXX000B</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled  
—: Unused  
X: Undefined  
Initial value

<table>
<thead>
<tr>
<th>BZ2</th>
<th>BZ1</th>
<th>BZ0</th>
<th>Buzzer selection bit (when FCH=8 MHz and FCL=32 kHz)</th>
<th>General-purpose output port (P31)</th>
<th>Time base timer output</th>
<th>Clock prescaler output</th>
<th>Frequency of the main clock source oscillation</th>
<th>Frequency of the sub-clock source oscillation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>FCH/2^12</td>
<td>FCL/2^4</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FCH/2^11</td>
<td>FCL/2^3</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>FCH/2^10</td>
<td>FCL/2^2</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FCH/2^9</td>
<td>FCL/2^1</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>FCH/2^8</td>
<td>FCL/2</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FCH/2^7</td>
<td>FCL/1</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>FCH/2^6</td>
<td>FCL</td>
<td>FCH</td>
<td>FCL</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FCH/2^5</td>
<td>FCL/2</td>
<td>FCH</td>
<td>FCL</td>
</tr>
</tbody>
</table>

FCH: Frequency of the main clock source oscillation  
FCL: Frequency of the sub-clock source oscillation
### Table 12.4-1 Functions of Each Bit of the Buzzer Register (BZCR)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7, Bit 6, Bit 5, Bit 4, Bit 3</td>
<td>Unused bit • Values when these bits are read are not defined. • Writing to these bits does not affect operations.</td>
</tr>
<tr>
<td>Bit 2, Bit 1, Bit 0</td>
<td>BZ2, BZ1, BZ0: buzzer selection bit • These bits are used to select the buzzer output and enable the output. • If &quot;000B&quot; is set to these bits, the buzzer output is disabled and the pin functions as a general-purpose port (P31). If a value other than &quot;000B&quot; is set, the pin functions as a buzzer output pin (BZ) and outputs square waves of the selected frequency. • The frequency of the buzzer output can be selected from four types of time base timer divided output and three types of clock prescaler divided output. <strong>Note:</strong> Do not select divide-by-the time base timer output in sub-clock mode. <strong>Reference:</strong> The sub-clock oscillates in main clock mode. Thus, by selecting divide-by-the clock prescaler output as buzzer output, the buzzer (BZ2, BZ1, BZ0=101B-111B) can be output also in main stop mode if the pin status designation bit (STBC: SPL) is set to &quot;0&quot;.</td>
</tr>
</tbody>
</table>
12.5 Program Example of the Buzzer Output

The following explains a program example of the buzzer output.

- Program Example of the Buzzer Output

  - **Processing specifications**
    - Buzzer output of about 0.977 kHz is output to the BZ pin and is then turned OFF.
    - The following shows the buzzer output frequency if \( \frac{2^{13}}{F_{CH}} \) (\( F_{CH} \) : main clock source oscillation frequency) is selected when the main clock source oscillation frequency is 8 MHz.

    \[
    \text{Buzzer output frequency} = \frac{8 \text{ MHz}}{2^{13}} = \frac{8 \text{ MHz}}{8192} = 0.977 \text{ kHz}
    \]

  - **Example of coding**

    ```
    BZCR EQU 000FH ; Address of the buzzer register
    CSEG ; [CODE SEGMENT]
    BUZON MOV BZCR,#00000001B ; Buzzer output ON
    (approx. 0.977 kHz/main clock frequency = 8 MHz)
    BUZOFF MOV BZCR,#00000000B ; Buzzer output OFF
    ENDS
    ```
CHAPTER 13  EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)

This chapter describes the functions and operations of the external interrupt circuit 1 (edge).

13.1  "Overview of the External Interrupt Circuit 1 (Edge)"
13.2  "Configuration of the External Interrupt Circuit 1"
13.3  "Pins of the External Interrupt Circuit 1"
13.4  "Registers of the External Interrupt Circuit 1"
13.5  "Interrupt of the External Interrupt Circuit 1"
13.6  "Explanation of Operations of the External Interrupt Circuit 1"
13.7  "Program Example of the External Interrupt Circuit 1"
CHAPTER 13  EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)

13.1 Overview of the External Interrupt Circuit 1 (Edge)

The external interrupt circuit 1 detects edges of the signal input to the four external interrupt pins and then generates an interrupt request to the CPU.

- **Function of the External Interrupt Circuit 1 (Edge Detection)**

  The external interrupt circuit 1 has a function for detecting edges of the signal input to the external interrupt pins and then generating an interrupt request to the CPU. This interrupt causes a return from the standby mode to resume normal operation (main RUN or sub-RUN state).

  - **External interrupt pin**: 4 pins (PA4/INT0-PA7/INT3)
  - **External interrupt source**: Signal input to an arbitrary edge (rising, falling) of the external interrupt pin
  - **Interrupt control**: Enable/Disable of the external interrupt input and interrupt request output by the external interrupt 1 control register (EIE1)
  - **Interrupt flag**: Detection of the specified edge by the external interrupt request flag bit of the external interrupt 1 flag register (EIF1)
  - **Interrupt request**: Generated by ORing each external interrupt source (IRQ0)
13.2 Configuration of the External Interrupt Circuit 1

The external interrupt circuit 1 consists of the following three parts:
- Edge detector
- External interrupt 1 control register (EIE1)
- External interrupt 1 flag register (EIF1)

Block Diagram of the External Interrupt Circuit 1

Figure 13.2-1 Block Diagram of the External Interrupt Circuit 1

- **Edge detector**
  The signal input to the external interrupt pin (INT0-INT3) is inverted or not inverted depending on the value of the SIV bit of the EIE1 register. If the external interrupt input is enabled, the fall of the signal processed there is detected and the external interrupt request bit (IF10-IF13) of the corresponding external interrupt pin is set to "1".

- **External interrupt 1 control register (EIE1)**
  Inversion/Non-inversion of the input signal is determined by the external interrupt input inverting bit (SIV0-SIV3). Using the external interrupt enable bit (IE10-IE13), the interrupt input and interrupt request output are enabled/disabled simultaneously.
CHAPTER 13 EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)

- **External interrupt 1 flag register (EIF1)**
  
  Using the external interrupt request flag bit (IF10-IF13), generated interrupt requests are checked and cleared.

- **Interrupt request of the external interrupt circuit 1**
  
  **IRQ0:**

  When an edge of the selected polarity enters any external interrupt pin of INT0-INT3, an interrupt request is generated if the corresponding external interrupt is enabled (EIE1: IE10-IE13=1).
13.3 Pins of the External Interrupt Circuit 1

This section explains the pins related to the external interrupt circuit 1 and provides a block diagram of the pins.

- **Pins Related to the External Interrupt Circuit 1**

  The pins related to the external interrupt circuit 1 are PA4/INT0-PA7/INT3.

  - **PA4/INT0-PA7/INT3 pin**

    These pins function as a general-purpose I/O port (PA4-PA7) and also as an external interrupt input (hysteresis input) pin (INT0-INT3).

    These pins function as an external interrupt input pin (INT0-INT3) by setting the relevant pin as an input port using the port A direction register (DDRA) and enabling the external interrupt using the external interrupt 1 control register (EIE1). The pin status can always be read by the Port A data register (PDRA).

    Table 13.3-1 "Pins Related to the External Interrupt Circuit 1" lists the pins related to the external interrupt circuit 1.

### Table 13.3-1 Pins Related to the External Interrupt Circuit 1

<table>
<thead>
<tr>
<th>External interrupt pin</th>
<th>Used as an external interrupt input pin (interrupt input enabled)</th>
<th>Used as a general-purpose I/O port (interrupt input disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA4/INT0</td>
<td>INT0 (EIE1:IE10=1, DDRA:bit4=0)</td>
<td>PA4 (EIE1:IE10=0)</td>
</tr>
<tr>
<td>PA5/INT1</td>
<td>INT1 (EIE1:IE11=1, DDRA:bit5=0)</td>
<td>PA5 (EIE1:IE11=0)</td>
</tr>
<tr>
<td>PA6/INT2</td>
<td>INT2 (EIE1:IE12=1, DDRA:bit6=0)</td>
<td>PA6 (EIE1:IE12=0)</td>
</tr>
<tr>
<td>PA7/INT3</td>
<td>INT3 (EIE1:IE13=1, DDRA:bit7=0)</td>
<td>PA7 (EIE1:IE13=0)</td>
</tr>
</tbody>
</table>

INT0 - INT3: If an edge of the selected polarity enters these pins, an interrupt request corresponding to the pin1 is generated.
Block Diagram of Pins Related to the External Interrupt Circuit 1

Figure 13.3-1 Block Diagram of Pins Related to the External Interrupt Circuit 1

Reference:

If "Pull-up resistor present" is selected in option settings, the pin status during reset or in stop or watch mode (SPL = 1) becomes "H."
# 13.4 Registers of the External Interrupt Circuit 1

This section explains the registers related to the external interrupt circuit 1.

- **Registers Related to the External Interrupt Circuit 1**

![Figure 13.4-1 Registers Related to the External Interrupt Circuit 1](image)

<table>
<thead>
<tr>
<th>EIE1 (External interrupt 1 control register)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 0\text{H}</td>
<td>SIV3</td>
<td>SIV2</td>
<td>SIV1</td>
<td>SIV0</td>
<td>IE13</td>
<td>IE12</td>
<td>IE11</td>
<td>IE10</td>
<td></td>
<td>00000000\text{B}</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>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EIF1 (External interrupt 1 flag register)</th>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 1\text{H}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IF13</td>
<td>IF12</td>
<td>IF11</td>
<td>IF10</td>
<td>XXXX0000\text{B}</td>
</tr>
<tr>
<td>R/W: Read/Write enabled</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X: Unused</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The external interrupt 1 control register (EIE1) is used to determine inversion/non-inversion of the interrupt input signal to the external interrupt pins INT0 - INT3 and enable/disable of the interrupt.
Figure 13.4-2 External Interrupt 1 Control Register (EIE1)

Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | Initial value
---|---|---|---|---|---|---|---|---|---
0 0 3 0 H | SIV3 | SIV2 | SIV1 | SIV0 | IE13 | IE12 | IE11 | IE10 | 00000000B

IE10 | IE13
---|---
0 | Disable the external interrupt input and interrupt request output
1 | Enable the external interrupt input and interrupt request output

SIV0 | SIV3
---|---
0 | Do not invert the external interrupt input signal (Detect the falling edge of pin input)
1 | Invert the external interrupt input signal (Detect the rising edge of pin input)

R/W: Read/Write enabled

: Initial value
CHAPTER 13  EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)

Table 13.4-1  Correspondence Between Each Bit of the External Interrupt 1 Control Register (EIE1) and the Pin

<table>
<thead>
<tr>
<th>Bit name</th>
<th>External interrupt pin</th>
<th>Interrupt name</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 7</td>
<td>SIV3</td>
<td>INT3</td>
</tr>
<tr>
<td>bit 3</td>
<td>IE13</td>
<td></td>
</tr>
<tr>
<td>bit 6</td>
<td>SIV2</td>
<td></td>
</tr>
<tr>
<td>bit 2</td>
<td>IE12</td>
<td>INT2</td>
</tr>
<tr>
<td>bit 5</td>
<td>SIV1</td>
<td></td>
</tr>
<tr>
<td>bit 1</td>
<td>IE11</td>
<td>INT1</td>
</tr>
<tr>
<td>bit 4</td>
<td>SIV0</td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td>IE10</td>
<td>INT0</td>
</tr>
</tbody>
</table>

Table 13.4-2  Functions of Each Bit of the External Interrupt 1 Control Register (EIE1)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>SIV3-SIV0: external interrupt input inverting bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td></td>
</tr>
<tr>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td>Bit 3</td>
<td>IE13-IE10: interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 2</td>
<td></td>
</tr>
<tr>
<td>Bit 1</td>
<td></td>
</tr>
<tr>
<td>Bit 0</td>
<td></td>
</tr>
</tbody>
</table>

- Bit to determine whether the signal input to the external interrupt pin is inverted
- If this bit is "0", falling edges of the pin input signal are detected. If this bit is "1", rising edges of the pin input signal are detected.

Bit that enables/disables the interrupt request output to the CPU and external interrupt input simultaneously. If this bit and the corresponding external interrupt request flag bit (IF13-IF10) are "1", an interrupt request is output.

Reference:
- To use an external interrupt pin, set the pin for input by writing "0" to the corresponding bit of the port A direction register (DDRA).
- The external interrupt pin status can be read directly by the port A data register (PDRA) regardless of the status of the external interrupt enable bit.
13.4.2 External Interrupt 1 Flag Register (EIF1)

The external interrupt 1 flag register (EIF1) is used to detect interrupt edges and clear the interrupt request flag.

- External Interrupt 1 Flag Register (EIF1)

![Figure 13.4-3 External Interrupt 1 Flag Register (EIF1)](image)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 1h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IF13</td>
<td>IF12</td>
<td>IF11</td>
<td>IF10</td>
<td>XXXX0000h</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/W</th>
<th>bit name</th>
<th>External interrupt pin</th>
<th>Interrupt name</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W</td>
<td>bit 3</td>
<td>IF13</td>
<td>INT3</td>
</tr>
<tr>
<td>R/W</td>
<td>bit 2</td>
<td>IF12</td>
<td>INT2</td>
</tr>
<tr>
<td>R/W</td>
<td>bit 1</td>
<td>IF11</td>
<td>INT1</td>
</tr>
<tr>
<td>R/W</td>
<td>bit 0</td>
<td>IF10</td>
<td>INT0</td>
</tr>
</tbody>
</table>

Table 13.4-3 Correspondence between Each Bit of the External Interrupt 1 Flag Register (EIF1) and the Pin

<table>
<thead>
<tr>
<th>Bit name</th>
<th>External interrupt pin</th>
<th>Interrupt name</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 3</td>
<td>IF13</td>
<td>INT3</td>
</tr>
<tr>
<td>bit 2</td>
<td>IF12</td>
<td>INT2</td>
</tr>
<tr>
<td>bit 1</td>
<td>IF11</td>
<td>INT1</td>
</tr>
<tr>
<td>bit 0</td>
<td>IF10</td>
<td>INT0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit name</th>
<th>External interrupt pin</th>
<th>Interrupt name</th>
</tr>
</thead>
</table>
### Table 13.4-4 Functions of Each Bit of the External Interrupt 1 Flag Register (EIF1)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7, Bit 6, Bit 5, Bit 4</td>
<td>Unused bit</td>
</tr>
</tbody>
</table>
|                                 | • Values when these bits are read are not defined.  
|                                 | • Writing of these bits does not affect operations.                                                                                     |
| Bit 3, Bit 2, Bit 1, Bit 0      | IF13-IF10: external interrupt request flag bit                                                                                         |
|                                 | • If the input edge specified by the external interrupt input inverting bit (SIV3-SIV0) is detected, the corresponding external interrupt  
|                                 |     request flag bit is set to "1".                                                                                                       |
|                                 | • If this bit and the corresponding interrupt request enable bit (IE13-IE10) are "1", an interrupt request is output.                      |
|                                 | • If "0" is written, these bits are cleared. If "1" is written, these bits are not affected.                                               |
|                                 | **Reference:**                                                                                                                           |
|                                 | Because the external interrupt input is also disabled when the external interrupt input inverting bit (EIE1: IE13-IE10) is "0", this bit  |
|                                 |     does not change when the specified edge is input.                                                                                     |
13.5 Interrupt of the External Interrupt Circuit 1

As an interrupt source of the external interrupt circuit 1, the detection of the specified edge of the signal input to the external interrupt pin is available.

- **Interrupt When the External Interrupt Circuit 1 Is Operating**

  If the specified edge of external interrupt input is detected when the external interrupt is enabled (EIE1: IE10-IE13=1), the corresponding external interrupt request flag bit (EIF1: IF10-IF13) is set to "1" and an interrupt request (IRQ0) to the CPU is generated. Because the same interrupt request (IRQ0) is generated by input of the specified edge to the external interrupt pin (INT0-INT3), software is responsible for determining to which pin the set external interrupt request flag bit corresponds. To clear the interrupt requests, use an interrupt processing routine to write "0" to the corresponding external interrupt request flag bit.

  **Note:**

  To enable (EIE1: IE10-IE13=1) the external interrupt after releasing a reset, clear (EIF1: IF10-IF13=0) the external interrupt request flag bit first.

  If the external interrupt request flag bit is "1" and the external interrupt enable bit is ON, it is not possible to return from an interrupt. Be sure to clear the external interrupt request flag bit in an interrupt processing routine.

  **Reference:**

  If the interrupt input is switched from non-inversion to inversion when the external interrupt input pin is in a "H" state, or if the interrupt input is switched from inversion to non-inversion when the external interrupt input pin is in a "L" state, the external interrupt request flag bit (EIF1: IF10-IF13) is set just after switching. Also, if the external interrupt input is switched from disable to enable (EIE1: IE10-IE13: 0 --> 1), the external interrupt request flag bit may be set. Disable the interrupt before manipulating the external interrupt input inverting bit and external interrupt enable bit. Enable the interrupt after clearing the interrupt request flag.

  If the external interrupt enable bit is switched from disable to enable (0 --> 1) when the external interrupt request flag bit is set to "1", an interrupt request is generated immediately.

  The stop mode release by an interrupt is possible only in the external interrupt circuit 1 or 2.

- **Register and Vector Table Related to the Interrupt of the External Interrupt Circuit 1**

  **Table 13.5-1  Register and Vector Table Related to the Interrupt of the External Interrupt Circuit 1**

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ0</td>
<td>ILR1 (007CH)</td>
<td>L01 (bit 1)</td>
</tr>
</tbody>
</table>

For the interrupt operation, see Section 3.4.2 "Flow of Interrupt Operation".
13.6 Explanation of Operations of the External Interrupt Circuit 1

The external interrupt circuit 1 detects the specified edge of the external interrupt pin and generates an interrupt request to the CPU.

Operations of the External Interrupt Circuit 1

For operation of the external interrupt circuit 1, the setting in Figure 13.6-1 "Setting of the External Interrupt Circuit 1" is required.

**Figure 13.6-1 Setting of the External Interrupt Circuit 1**

The signal input from the external interrupt pin (INT0-INT3) is inverted or not inverted depending on the setting of the external interrupt input inverting bit (EIE1: SIV0-SIV3). Only when the external interrupt enable bit (EIE1: IE10-IE13) is "1" is the corresponding external interrupt request flag bit (EIF1: IF10-IF13) set to "1" by the falling edge of the processed signal.

Figure 13.6-2 "Operation of the External Interrupt Circuit 1 (INT0)" shows the operation of the external interrupt circuit 1 (when the INT0 pin is used).
Reference:

The pin status can be read directly by the port data register (PDRA) if the port is used as an external interrupt pin.
13.7 Program Example of the External Interrupt Circuit 1

The following explains a program example of the external interrupt circuit 1.

Program Example of the External Interrupt Circuit 1

- **Processing specifications**
  
  An interrupt shall be generated after detecting a falling edge input to the INT0 pin.

- **Example of coding**

```
; Program Example of the External Interrupt Circuit 1

; Processing specifications
; An interrupt shall be generated after detecting a falling edge input to the INT0 pin.

; Example of coding

DDRA EQU 001BH ; Address of the port A direction register
EIE1 EQU 0030H ; Address of the external interrupt 1 control register
EIF1 EQU 0031H ; Address of the external interrupt 1 flag register
IE10 EQU EIE1:0 ; Definition of the external interrupt enable bit
SIV0 EQU EIE1:4 ; Definition of the external interrupt input inverting bit
IF10 EQU EIF1:0 ; Definition of the external interrupt request flag bit
ILR1 EQU 007CH ; Address of the interrupt level setting register

; ------------Main program-----------------------------------------------
CSEG ; [CODE SEGMENT]
        ; Assuming that the stack pointer (SP) and others have
        ; been initialized.
CLR1 ; Disable the interrupts
        
MOV ILR1,#11111110B ; Set the interrupt level to 2
MOV DDRA,#00000000B ; Set the PA4/INT0 pin for input
        
CLR1 ; Disable the interrupts
        
SETB IE10 ; Enable the INT0 interrupt input
        
CLR1 ; Disable the interrupts
        
SET1 ; Enable the interrupts
        
: ; ------------Interrupt processing routine-------------------------------
WARI CLRB IF10 ; Clear the external interrupt request flag
        
PUSHW A
XCHW A,T
        
User processing
        
: ; ------------Interrupt processing routine-------------------------------
POPW A
XCHW A,T
POPW A
RETI
ENDS
;-----------------------------------------------
END

```
CHAPTER 14 EXTERNAL INTERRUPT CIRCUIT 2 (LEVEL)

This chapter describes the functions and operations of external interrupt circuit 2 (level).

14.1 "Overview of External Interrupt Circuit 2"
14.2 "External Interrupt Circuit 2 Configuration"
14.3 "Pins of External Interrupt Circuit 2"
14.4 "Registers of External Interrupt Circuit 2"
14.5 "External Interrupt Circuit 2 Interrupts"
14.6 "Operation of External Interrupt Circuit 2"
14.7 "External Interrupt Circuit 2 Programming example"
14.1 Overview of External Interrupt Circuit 2

External interrupt circuit 2 detects the levels of the signals entered to the 12 external interrupt pins and generates one interrupt to the CPU.

- **Function of External Interrupt Circuit 2 (Level Detection)**

  External interrupt circuit 2 has a function for detecting the "L" level signal entered to the external output pins and generating an interrupt to the CPU. This interrupt results in the restoration from standby mode and transition to the ordinary operating state (main RUN or sub-RUN state).

  - **External interrupt pins**: 12 (P90/INT20 to P97/INT27, PA0/INT28 to PA3/INTB)
  - **External interrupt source**: Input of "L" level signals to the external interrupt pins
  - **Interrupt control**: The external interrupt 2 control register (EIE2) enables/disables external interrupt input.
  - **Interrupt flag**: The external interrupt request flag bit of the external interrupt 2 control register (EIF2) detects an "L" level signal.
  - **Interrupt request**: Generated by OR'ing individual external interrupt sources (IRQ1)
14.2 External Interrupt Circuit 2 Configuration

External interrupt circuit 2 consists of the following three blocks.
- Interrupt request generator circuit
- External interrupt 2 control register (EIE2)
- External interrupt 2 flag register (EIF2)

Block Diagram of External Interrupt Circuit 2

![Block Diagram of External Interrupt Circuit 2](image)

- **Interrupt request generator circuit**
  The interrupt request generator circuit generates an interrupt request signal by the signals input to the external interrupt pins (INT20 to INTB) and the external interrupt input enable bits of the external interrupt 2 control register (EIE2) and the external interrupt 2 flag register (EIF2: IE28).
CHAPTER 14  EXTERNAL INTERRUPT CIRCUIT 2 (LEVEL)

- **External interrupt 2 control register (EIE2)**
  The external interrupt input enable bits (IE20 to IE27) enable or disable the input of "L" level signals from the corresponding external interrupt pins.

- **External interrupt 2 flag register (EIF2)**
  The external interrupt enable bit (IE28) enables or disables the input of "L" level signals from the external interrupt pins INT28 to INTB.
  The external interrupt request flag bit (IF20) retains or clears the interrupt request signal generated.

- **External interrupt circuit 2 related interrupts**
  **IRQ1:**
  An interrupt request occurs if an "L" level signal is input to any of the external interrupt pins INT20 to INTB and the external interrupt request input enable bit corresponding to the pin is "1".
14.3 Pins of External Interrupt Circuit 2

This section describes the external interrupt circuit 2 related pins and provides a block diagram of these pins.

- Pins Related to External Interrupt Circuit 2

  The pins related to external interrupt circuit 2 are 12 external interrupt pins.

- P90/INT20 to P97/INT27, PA0/INT28 to PA3/INTB pins

  These pins function as both external interrupt input (hysteresis) pins and general I/O ports.

  The P90/INT20 to P97/INT27, PA0/INT28 to PA3/INTB pins set the corresponding pins to the input port by the port direction register (DDR9, DDRA) and function as external interrupt input pins (INT20 to INTB) in states where external interrupt input is enabled (EIE2: IE20 to IE27 = 1, EIF2: IE28 = 1). The states of the pins can be read from the port data register (PDR9, PDRA) at any time.

  Table 14.3-1 "Pins Related to External Interrupt Circuit 2" shows the pins related to external interrupt circuit 2.

<table>
<thead>
<tr>
<th>External interrupt pin</th>
<th>Used as external interrupt input (interrupt input enabled)</th>
<th>Used as general-purpose I/O port (interrupt input disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P90/INT20</td>
<td>INT20 (EIE2:IE20=1, DDR9:bit0=0)</td>
<td>P90 (EIE2:IE20=0)</td>
</tr>
<tr>
<td>P91/INT21</td>
<td>INT21 (EIE2:IE21=1, DDR9:bit1=0)</td>
<td>P91 (EIE2:IE21=0)</td>
</tr>
<tr>
<td>P92/INT22</td>
<td>INT22 (EIE2:IE22=1, DDR9:bit2=0)</td>
<td>P92 (EIE2:IE22=0)</td>
</tr>
<tr>
<td>P93/INT23</td>
<td>INT23 (EIE2:IE23=1, DDR9:bit3=0)</td>
<td>P93 (EIE2:IE23=0)</td>
</tr>
<tr>
<td>P94/INT24</td>
<td>INT24 (EIE2:IE24=1, DDR9:bit4=0)</td>
<td>P94 (EIE2:IE24=0)</td>
</tr>
<tr>
<td>P95/INT25</td>
<td>INT25 (EIE2:IE25=1, DDR9:bit5=0)</td>
<td>P95 (EIE2:IE25=0)</td>
</tr>
<tr>
<td>P96/INT26</td>
<td>INT26 (EIE2:IE26=1, DDR9:bit6=0)</td>
<td>P96 (EIE2:IE26=0)</td>
</tr>
<tr>
<td>P97/INT27</td>
<td>INT27 (EIE2:IE27=1, DDR9:bit7=0)</td>
<td>P97 (EIE2:IE27=0)</td>
</tr>
<tr>
<td>PA0/INT28</td>
<td>INT28 (EIE2:IE28=1, DDRA:bit0=0)</td>
<td>PA0 (EIE2:IE28=0)</td>
</tr>
<tr>
<td>PA1/INT29</td>
<td>INT29 (EIE2:IE28=1, DDRA:bit1=0)</td>
<td>PA1 (EIE2:IE28=0)</td>
</tr>
<tr>
<td>PA2/INTA</td>
<td>INTA (EIE2:IE28=1, DDRA:bit2=0)</td>
<td>PA2 (EIE2:IE28=0)</td>
</tr>
<tr>
<td>PA3/INTB</td>
<td>INTB (EIE2:IE28=1, DDRA:bit3=0)</td>
<td>PA3 (EIE2:IE28=0)</td>
</tr>
</tbody>
</table>
Block Diagram of the Pins Related to External Interrupt Circuit 2

Figure 14.3-1 Block diagram of the Pins Related to External Interrupt Circuit 2

Reference:

When "With pull-up resistor" is selected in optional setting, the pin state at the time of reset and in stop or watch mode (SPL = 1) is "H."
### Relationship between Interrupt Enable Bit and External Interrupt Pins in External Interrupt Circuit 2

Table 14.3-2 "Correspondence between Interrupt Enable Bits and External Interrupt Pins" shows the correspondence between interrupt enable bits and external interrupt pins.

**Table 14.3-2  Correspondence between Interrupt Enable Bits and External Interrupt Pins**

<table>
<thead>
<tr>
<th>Register</th>
<th>Bit name</th>
<th>External interrupt pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIE2</td>
<td>bit0</td>
<td>IE20</td>
</tr>
<tr>
<td></td>
<td>bit1</td>
<td>IE21</td>
</tr>
<tr>
<td></td>
<td>bit2</td>
<td>IE22</td>
</tr>
<tr>
<td></td>
<td>bit3</td>
<td>IE23</td>
</tr>
<tr>
<td></td>
<td>bit4</td>
<td>IE24</td>
</tr>
<tr>
<td></td>
<td>bit5</td>
<td>IE25</td>
</tr>
<tr>
<td></td>
<td>bit6</td>
<td>IE26</td>
</tr>
<tr>
<td></td>
<td>bit7</td>
<td>IE27</td>
</tr>
<tr>
<td>EIF2</td>
<td>bit1</td>
<td>IE28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INT28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INT29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INTA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INTB</td>
</tr>
</tbody>
</table>
This section describes the registers related to external interrupt circuit 2.

### Registers Related to External Interrupt Circuit 2

#### Figure 14.4-1 Registers Related to External Interrupt Circuit 2

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 2H</td>
<td>IE27</td>
<td>IE26</td>
<td>IE25</td>
<td>IE24</td>
<td>IE23</td>
<td>IE22</td>
<td>IE21</td>
<td>IE20</td>
<td>00000000b</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>EIF2 0 0 3 3H</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>IE28</td>
<td>IF20</td>
<td>XXXXX00b</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled

- : Unused
- X: Undefined
### 14.4.1 External Interrupt 2 Control Register (EIE2)

External interrupt 2 control register (EIE2) enables and disables interrupt input to the external interrupt pins INT20 to INT27.

- **External Interrupt 2 Control Register (EIE2)**

#### Figure 14.4-2 External Interrupt 2 Control Register (EIE2)

![External Interrupt 2 Control Register (EIE2)](image)

R/W: Read/Write enabled

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 2</td>
<td>IE27</td>
<td>IE26</td>
<td>IE25</td>
<td>IE24</td>
<td>IE23</td>
<td>IE22</td>
<td>IE21</td>
<td>IE20</td>
<td>00000000 B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IE20</th>
<th>IE27</th>
<th>Interrupt request enable bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Disables output request input</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Enables output request input</td>
</tr>
</tbody>
</table>

#### Table 14.4-1 Correspondence between Bits and External Interrupt Pins in the External Interrupt 2 Control Register (EIE2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>External interrupt pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 7</td>
<td>IE27</td>
</tr>
<tr>
<td>bit 6</td>
<td>IE26</td>
</tr>
<tr>
<td>bit 5</td>
<td>IE25</td>
</tr>
<tr>
<td>bit 4</td>
<td>IE24</td>
</tr>
<tr>
<td>bit 3</td>
<td>IE23</td>
</tr>
<tr>
<td>bit 2</td>
<td>IE22</td>
</tr>
<tr>
<td>bit 1</td>
<td>IE21</td>
</tr>
<tr>
<td>bit 0</td>
<td>IE20</td>
</tr>
<tr>
<td></td>
<td>INT27</td>
</tr>
<tr>
<td></td>
<td>INT26</td>
</tr>
<tr>
<td></td>
<td>INT25</td>
</tr>
<tr>
<td></td>
<td>INT24</td>
</tr>
<tr>
<td></td>
<td>INT23</td>
</tr>
<tr>
<td></td>
<td>INT22</td>
</tr>
<tr>
<td></td>
<td>INT21</td>
</tr>
<tr>
<td></td>
<td>INT20</td>
</tr>
</tbody>
</table>
### Table 14.4-2 Functions of Each Bit in the External Interrupt 2 Control Register (EIE2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>IE27 to IE20:</strong> Interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td>- These bits enable and disable interrupt input to the external interrupt input pins INT20 to INT27.</td>
</tr>
<tr>
<td>Bit 5</td>
<td>- When one of these bits is set to &quot;1&quot;, the corresponding external interrupt pin functions as an external interrupt input pin and accepts external interrupt input.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>- When one of these bits is set to &quot;0&quot;, the corresponding external interrupt pin functions as a general-purpose port and does not accept external interrupt input.</td>
</tr>
</tbody>
</table>

**Reference:**
- When using an external interrupt pin, write "0" into the corresponding bit of the port direction register (DDR9) to set the pin to input.
- The states of the external interrupt pins can be read directly from the port data register (PDR9) regardless of the state of the external interrupt input enable bit.
14.4.2 External Interrupt 2 Flag Register (EIF2)

The external interrupt 2 flag register (EIF2) detects level interrupts, clears the interrupt request flag, and enables and disables the interrupt input to the external interrupt pins INT28 to INTB.

![Diagram of the External Interrupt 2 Flag Register (EIF2)]

**External Interrupt 2 Flag Register (EIF2)**

**Figure 14.4-3 External Interrupt 2 Flag Register (EIF2)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 3h</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>IE2</td>
<td>XXXXX00h</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IF20</th>
<th>External interrupt request flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read cycle</td>
<td>Write cycle</td>
</tr>
<tr>
<td>0</td>
<td>No interrupt request (no &quot;L&quot; level detection)</td>
</tr>
<tr>
<td>1</td>
<td>Interrupt request detected (&quot;L&quot; level detected)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IE28</th>
<th>External interrupt input enable bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disable external interrupt input</td>
</tr>
<tr>
<td>1</td>
<td>Enable external interrupt input</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
—: Unused
X: Undefined
: Initial value
### CHAPTER 14  EXTERNAL INTERRUPT CIRCUIT 2 (LEVEL)

**Table 14.4-3  Functions of Each Bit in the External Interrupt 2 Flag Register (EIF2)**

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>Unused bit • The value in read cycle is undefined. • In write cycle, this bit has no influence on operation.</td>
</tr>
<tr>
<td>Bit 6</td>
<td></td>
</tr>
<tr>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td>Bit 3</td>
<td></td>
</tr>
<tr>
<td>Bit 2</td>
<td></td>
</tr>
<tr>
<td>Bit 1</td>
<td>IE28: External interrupt input enable bit • This bit enables and disables the interrupt input to the four external interrupt pins INT28 to INTB correctly. • When this bit is set to &quot;1&quot;, the external interrupt pins function as external interrupt input pins and accept external interrupt input. <strong>Note:</strong> When there is a pin not used as an external interrupt pin, fix the input level of the unused pin to &quot;H.&quot; <strong>Reference:</strong> • When using an external interrupt pin, write &quot;0&quot; into the corresponding bit of the port direction register (DDRA) to set the pin to input. • The states of the external interrupt pins can be read from the port data register (PDRA) directly regardless of the state of the external interrupt input enable bit.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>IF20: External interrupt request flag bit • When an &quot;L&quot; level signal is input to any of the external interrupt pins INT20 to INTB in states where external interrupt input is enabled for the pin, this bit is set to &quot;1&quot;. • In write cycle, this bit is cleared when &quot;0&quot; is written. It does not affect operation when &quot;1&quot; is written. <strong>Note:</strong> The external interrupt input enable bits of the external interrupt 2 control register (E1E2: IE20 to IE27, EIF2: IE28) disable the input of external interrupts. Interrupt requests continue to be generated until the IF20 bit is cleared to &quot;0&quot;.</td>
</tr>
</tbody>
</table>
14.5 External Interrupt Circuit 2 Interrupts

The interrupt source of external interrupt circuit 2 is an "L" level input signal entered to an external interrupt pin.

■ Interrupts during the Operation of External Interrupt Circuit 2

When an "L" level signal is input to an external interrupt pin when its interrupt input (IRQ1) is enabled, the external interrupt request flag bit (EIF2: IF20) is set to "1" and an interrupt request to the CPU is generated. Write "0" into the IF20 bit in the interrupt processing routine to clear the interrupt request.

When the external interrupt request flag bit (IF20) is set to "1", interrupt requests continue to be generated until the IF20 bit is cleared to "0" even if external interrupt input is disabled by the interrupt request enable bits (EIE2: IE20 to IE27, EIF2: IE28). Therefore, be sure to clear the IF20 bit.

Even if the IF20 bit is cleared in states where external interrupt input is not disabled, the IF20 bit is set immediately if the external interrupt pin remains low. Disable external interrupt input or eliminate the external interrupt source as required.

**Note:**

To enable interrupts to the CPU after reset, clear the IF20 bit in advance.

**Reference:**

Input of "L" level signals to the external interrupt pins (INT20 to INTB) generates the same interrupt request (IRQ1). Therefore, it is necessary to determine to which pin the interrupt input corresponds by reading the port data register (PDR9, PDRA) before the input changes to "H."

Release by an interrupt of stop mode is possible only in external interrupt circuits 1 and 2.

■ Register and Vector Table Related to External Interrupt Circuit 2 Interrupts

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ1</td>
<td>ILR1 (007Ch)</td>
<td>L11 (bit 3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L10 (bit 2)</td>
</tr>
</tbody>
</table>

For interrupt operation, see Section 3.4.2 "Flow of Interrupt Operation."
External interrupt circuit 2 detects the "L" level signal entered to an external interrupt pin and generates an interrupt request to the CPU.

### Operation of External Interrupt Circuit 2

To operate external interrupt circuit 2, the setting shown in Figure 14.6-1 "External Interrupt Circuit 2 Setting" is required.

**Figure 14.6-1 External Interrupt Circuit 2 Setting**

When an "L" level signal is input to an external interrupt pin of INT20 to INTB in a state in which external interrupt input is enabled in the corresponding bit of IE20 to IE28, an IRQ1 interrupt request to the CPU is generated.

Figure 14.6-2 "Operation of External Interrupt Circuit 2 (INT20)" shows the operation of external interrupt circuit 2 (when INT20 pin is used).
Figure 14.6-2 Operation of External Interrupt Circuit 2 (INT20)

Input waveform to INT20 pin ("L" level is detected)

EIE2:IE20

EIF2:IF20
(State in IRQ1 is the same)

Operation of interrupt processing routine corresponding to IRQ1

PDR9:bit0

External interrupt input enable state

Cleared in interrupt processing routine

Interrupt processing

Interrupt process

Readable at any point

RETI

Reference:
The states of the external interrupt pins can be read from the port data registers (PDR9, PDRA) directly even if they are used as external interrupt input.
14.7 External Interrupt Circuit 2 Programming example

A programming example of external interrupt circuit 2 is shown in the following.

A Programming Example of External Interrupt Circuit 2

- **Processing specifications**
  
  Detect the "L" level signal input to the INT20 pin to generate an interrupt.

- **Coding example**

```
; Processing specifications
Detect the "L" level signal input to the INT20 pin to generate an interrupt.

; Coding example

DDR9 EQU 0019H ; Address of port direction register
EIE2 EQU 0032H ; Address of external interrupt 2 control register
EIF2 EQU 0033H ; Address of external interrupt 2 flag register
IF20 EQU EIF2:0 ; Definition of interrupt request flag bit
ILR1 EQU 007CH ; Address of interrupt level setting register

INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFF8H
IRQ1 DW WARI ; Set interrupt vector
INT_V ENDS

;------------Main program-----------------------------------------------
CSEG ; [CODE SEGMENT]
; Assuming that stack pointer (SP) has been initialized
CLR1 ; Disable interrupt
CLRB IF20 ; Clear external interrupt request flag
MOV ILR1,#11111110B ; Set interrupt level (level 2)
MOV DDR9,#00000000B ; Set the INT20 pin to input
MOV EIF2,#00000001B ; Enable external interrupt input to INT20 pin
SET1 ; Interrupt enable

;------------Interrupt processing routine-------------------------------------
WARI MOV EIE2,#00000000B ; Disable external interrupt input to INT20 pin
CLRB IF20 ; Clear external interrupt request flag
PUSHW A
XCHW A,T
PUSHW A
; User processing
POPW A
XCHW A,T
POPW A
RETI
ENDS

;-------------------------------------------------------------------------
END
```
This chapter describes the functions and operations of the A/D converter.

15.1 "Overview of A/D Converter"
15.2 "A/D Converter Configuration"
15.3 "Pins of A/D Converter"
15.4 "Registers of A/D Converter"
15.5 "A/D Converter Interrupts"
15.6 "Operation of A/D Converter"
15.7 "Caution on Use of A/D Converter"
15.8 "Programming Examples of A/D Converter"
15.1 Overview of A/D Converter

As for the A/D converter, there are options of two functions: the 8-bit successive approximation type A/D converting function and the sensing function that compares the set voltage and input voltage at high speed. These functions can be started by software by selecting one input signal from 8-channel analog input pins. The sensing function can be used to check battery charge status and temperature fluctuation.

■ A/D Converting Function

The A/D converting function converts an analog voltage applied to analog input pins (input voltage) into an 8-bit digital value (A/D conversion).

- From the eight analog input pins, one pin can be selected.
- The conversion speed is 44 instruction cycles (22 µs when the highest clock speed of 8 MHz is selected for the main clock source oscillation).
- When A/D conversion is completed, an interrupt occurs.
- The completion of conversion can also be determined by software.

This function is started by starting the A/D conversion function.

■ Sensing Function

The sensing function compares an analog voltage applied to analog input pins (input voltage) to the voltage corresponding to the value set in the A/D data register (ADCD) (comparison voltage) and determines which is higher.

- From the eight analog input pins, one pin can be selected.
- The comparison speed is 12 instruction cycles (6.3 µs when the highest clock speed of 8 MHz is selected for the main clock source oscillation).
- When the comparison condition is met, an interrupt occurs.
15.2 A/D Converter Configuration

The A/D converter consists of the following eight blocks:

- Analog channel selector
- Sample hold circuit
- D/A converter
- Comparator
- Control circuit
- A/D data register (ADCD)
- A/D control register 1 (ADC1)
- A/D control register 2 (ADC2)

Block Diagram of the A/D Converter

Figure 15.2-1  Block Diagram of the A/D Converter

- Analog channel selector

The analog channel selector is a circuit for selecting one pin from the eight analog input pins.
CHAPTER 15 A/D CONVERTER

❖ Sample hold circuit
The sample hold circuit holds the input voltage selected with the analog channel selector. By sample-holding the input voltage immediately after the start of A/D conversion or the sensing function, conversion can be performed without being affected by the input voltage during A/D conversion (comparison).

❖ D/A converter
The D/A converter generates a voltage corresponding to the value set in the ADCD register.

❖ Comparator
The comparator compares the sample-held input voltage to the output voltage of the D/A converter and determines which is higher.

❖ Control circuit
The control circuit has two functions.
- In A/D conversion mode, the values of the ADCD register are determined from the most significant bit to the least significant bit separately based on the signals sent from the comparator indicating which is higher. When conversion is completed, the interrupt request flag bit (ADC1: ADI) is set.
- In sensing mode, when the signal from the comparator indicating which is higher matches the condition of the comparison condition setting bit of the ADC1 register (SIFM), the interrupt request flag bit (ADI) is set.

❖ A/D data register (ADCD)
The ADCD register has two functions.
- In A/D conversion mode, the result of A/D conversion is stored in this register.
- In sensing mode, the data of the voltage compared to the input voltage is written into this register.

❖ A/D control register 1 (ADC1)
A/D control register 1 (ADC1) enables/disables functions, selects an analog input pin, checks the state, and controls interrupts.

❖ A/D control register 2 (ADC2)
A/D control register 2 (ADC2) selects the input clock, enables/disables interrupts, and selects functions.

❖ A/D converter related interrupt
IRQ8:
An interrupt request occurs when the interrupt request output is enabled (ADC2: ADIE = 1) when A/D conversion is completed and when the condition set in the sensing function is met.
Supply Voltage of the A/D Converter

- **AVcc**
  A power supply pin of the A/D converter. Use it in the same potential as Vcc. When the high precision of A/D conversion is required, respond so that the Vcc noise is not put on AVcc or use another power supply. Even if the A/D converter is not used, connect this pin to the power supply.

- **AVss**
  A ground pin of the A/D converter. Use it in the same potential as Vss. When the high precision of A/D conversion is required, respond so that noise of Vss is not put on AVss.
  Even if the A/D converter is not used, connect this pin to ground (GND).

- **AVR**
  A pin to input the reference voltage of the A/D converter. 8-bit A/D conversion is performed between AVR and AVss.
  When the A/D converter is not used, connect it to AVss.
15.3 Pins of A/D Converter

This section describes the A/D converter related pins and provides a block diagram of these pins.

- **Pins Related to the A/D Converter**
  
  The pins related to the A/D converter are the P50/AN00 to P57/AN07 pins.
  
  **P50/AN00 to P57/AN07 pins**
  
  These pins serve as output ports (P50 to P57) and as analog input pins (AN00 to AN07).

  **AN00 to AN07:**
  
  When the A/D conversion function or sensing function is used, an analog voltage to be converted or compared is entered into these pins.

  To make one of these pins function as an analog input pin, write "1" into the corresponding bit of the port data register (PDR5) to turn off the output transistor and select the pin with the analog input channel select bits (ADC1: ANS0 to ANS3). Even if the A/D converter is used, the pins not used as analog input pins can be used as output ports.

- **Block Diagram of Pins Related to the A/D Converter**

  ![Block Diagram of P50/AN00 to P57/AN07 Pins](image)

  **Reference:**
  
  When "With pull-up resistor" is set in optional setting, the pin state at the time of reset and in stop or watch mode (SPL = 1) is "H."
15.4 Registers of A/D Converter

This section describes the registers related to the A/D converter.

- Registers Related to the A/D Converter

![Figure 15.4-1 Registers Related to the A/D Converter](image)

ADC1 (A/D control register 1)

- Address: 0 0 2 D_H
- Initial value: 00000000b
- Bit definitions:
  - bit7: ANS3
  - bit6: ANS2
  - bit5: ANS1
  - bit4: ANS0
  - bit3: ADI
  - bit2: ADMV
  - bit1: SIFM
  - bit0: AD

ADC2 (A/D control register 2)

- Address: 0 0 2 E_H
- Initial value: XXX00001b
- Bit definitions:
  - bit7: ADIE
  - bit6: EXT
  - bit5: RESV
  - bit4: ADMD
  - bit3: EXT
  - bit2: RESV
  - bit1: ADMD
  - bit0: ADIE

ADCD (A/D data register)

- Address: 0 0 2 F_H
- Initial value: XXXXXXXXb
- Bit definitions:
  - bit7: 
  - bit6: 
  - bit5: 
  - bit4: 
  - bit3: 
  - bit2: 
  - bit1: 
  - bit0: 

R/W: Read/Write enabled
R: Read only
_:_ Unused
X: Undefined
15.4.1 A/D Control Register 1 (ADC1)

The A/D control register 1 (ADC1) enables/disables the functions of the A/D converter, selects an analog input pin, and checks the state.

### A/D Control Register 1 (ADC1)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 Dh&lt;sub&gt;H&lt;/sub&gt;</td>
<td>ANS3</td>
<td>ANS2</td>
<td>ANS1</td>
<td>ANS0</td>
<td>ADi</td>
<td>ADMV</td>
<td>SIFM</td>
<td>AD</td>
<td>00000000&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

#### AD: A/D conversion start bit
- Valid only when software is started (ADC2: EXT = 0)
- 0: A/D conversion function or sensing function is not started
- 1: A/D conversion function or sensing function is started

#### SIFM: Comparison condition setting bit
- Valid only when sensing function is used (ADC2: ADMD = 1)
- 0: The interrupt request flag bit is set when the input voltage is lower than the comparison voltage.
- 1: The interrupt request flag bit is set when the input voltage is higher than the comparison voltage.

#### ADMV: Conversion flag bit
- 0: Conversion or comparison is not in progress
- 1: Conversion or comparison is in progress

#### ADi: Interrupt request flag bit
- **Read cycle**
  - A/D conversion mode
    - 0: Conversion is not completed
    - 1: Conversion is completed
  - Sensing mode
    - Set condition is not met
    - Set condition is met
- **Write cycle**
  - 0: This bit is cleared
  - 1: No change, no affect on others

#### ANS0-ANS3: Analog input channel select bit
- 0 0 0 0: AN0 pin
- 0 0 0 1: AN1 pin
- 0 0 1 0: AN2 pin
- 0 0 1 1: AN3 pin
- 0 1 0 0: AN4 pin
- 0 1 0 1: AN5 pin
- 0 1 1 0: AN6 pin
- 0 1 1 1: AN7 pin
- 1 0 0 0: Prohibited
- 1 1 1 1: Prohibited

---

**Figure 15.4-2 A/D Control Register 1 (ADC1)**
### Table 15.4-1 Functions of Each Bit in A/D Control Register 1 (ADC1)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7 ANS3, ANS2, ANS1, ANS0: Analog input channel select bits</td>
<td>From AN0 to AN07, these bits select a pin used as an analog input pin. When starting with software (ADC2: EXT = 0), these bits can be rewritten simultaneously with the start of the A/D conversion function or sensing function (AD = 1).</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Be sure to set ANS3 to &quot;0&quot;. When ANS3 is set to &quot;1&quot;, an analog input pin is not selected. Disable the output of the general-purpose port that corresponds to the analog input pin. Do not rewrite these bits when the ADMV bit is &quot;1&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Reference:</strong> The pins not used as analog input pins can be used as general-purpose ports.</td>
</tr>
<tr>
<td>Bit 6</td>
<td>ADI: Interrupt request flag bit</td>
</tr>
<tr>
<td></td>
<td>• In A/D conversion mode, this bit is set to &quot;1&quot; when A/D conversion is completed.</td>
</tr>
<tr>
<td></td>
<td>• In sensing mode, this bit is set to &quot;1&quot; when the input voltage meets the condition set in the comparison condition setting bit (SIFM).</td>
</tr>
<tr>
<td></td>
<td>• In each function, an interrupt request is output when this bit and the interrupt request enable bit (ADC2: ADIE) are set to &quot;1&quot;.</td>
</tr>
<tr>
<td></td>
<td>• In write cycle, this bit is cleared when &quot;0&quot; is written and this bit does not affect operation when &quot;1&quot; is written.</td>
</tr>
<tr>
<td>Bit 5</td>
<td>ADMV: Conversion flag bit</td>
</tr>
<tr>
<td></td>
<td>In A/D conversion mode, this bit indicates that conversion is in progress. During conversion (comparison), this bit is set to &quot;1&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Reference:</strong> This bit is read only. The written value has no significance and does not affect operation.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>SIFM: Comparison condition setting bit</td>
</tr>
<tr>
<td></td>
<td>• In A/D conversion mode, this bit has no significance.</td>
</tr>
<tr>
<td></td>
<td>• In sensing mode, the comparison condition between the input voltage and comparison voltage is set as an interrupt source.</td>
</tr>
<tr>
<td></td>
<td>• If the input voltage is lower than the comparison voltage when this bit is &quot;0&quot;, an interrupt request occurs (ADI = 1). If the input voltage is higher than the comparison voltage when this bit is &quot;1&quot;, an interrupt request occurs (ADI = 1).</td>
</tr>
<tr>
<td></td>
<td>• When the input voltage and comparison voltage are equal, no interrupt request occurs.</td>
</tr>
<tr>
<td></td>
<td>• When starting with software (ADC2: EXT = 0), this bit can be rewritten simultaneously with the start of the sensing function (AD = 1).</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Do not rewrite these bits when the ADMV bit is &quot;1&quot;.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>AD: A/D conversion start bit</td>
</tr>
<tr>
<td></td>
<td>• This bit starts the A/D conversion function or sensing function with software.</td>
</tr>
<tr>
<td></td>
<td>• When writing &quot;1&quot; into this bit in states where continuous start is not performed (ADC2: EXT = 0), the A/D conversion function or sensing function is started.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Even if &quot;0&quot; is written to this bit, the operation of the A/D conversion function or sensing function cannot be stopped. The read value is always &quot;0&quot;.</td>
</tr>
<tr>
<td></td>
<td>• In continuous start, this bit has no significance.</td>
</tr>
</tbody>
</table>
15.4.2 A/D Control Register 2 (ADC2)

The A/D control register 2 (ADC 2) selects the functions of the A/D converter, selects the input clock, enables/disables interrupts, and checks the state.

A/D Control Register 2 (ADC2)

Figure 15.4-3 A/D Control Register 2 (ADC2)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 Eh</td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
<td>RESV</td>
<td>XXXX0001h</td>
</tr>
</tbody>
</table>

- **R/W**: Read/Write enabled
- **X**: Undefined
- **—**: Unused

### Bit Descriptions
- **ADIE**: Interrupt request enable bit
  - 0: Disable interrupt request output
  - 1: Enable interrupt request output
- **ADMD**: Function selection bit
  - 0: A/D conversion function
  - 1: Sensing function
- **RESV**: Reserved bit
  - Be sure to write "1" into this bit.
- **EXT**: Reserved bit
  - Be sure to write "0" into this bit.
Table 15.4-2 Functions of Each Bit in A/D Control Register 2 (ADC2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7, Bit 6, Bit 5, Bit 4</td>
<td>Unused bit</td>
</tr>
<tr>
<td>Bit 3</td>
<td>ADIE: Interrupt request enable bit</td>
</tr>
<tr>
<td>Bit 2</td>
<td>ADMD: Function selection bit</td>
</tr>
<tr>
<td>Bit 1</td>
<td>EXT: Reserved bit</td>
</tr>
<tr>
<td>Bit 0</td>
<td>RESV: Reserved bit</td>
</tr>
</tbody>
</table>

- **Bit 7**: Unused bit. The value in read cycle is undefined. In write cycle, this bit does not affect operation.
- **Bit 3**: ADIE: Interrupt request enable bit. This bit enables/disables interrupt request output to the CPU. When this bit and the interrupt request flag bit (ADC1: ADI) are "1", an interrupt request is output.
- **Bit 2**: ADMD: Function selection bit. This bit switches between the A/D conversion mode and the sensing mode. When this bit is "0", the A/D converter operates as the A/D conversion function. When this bit is "1", the A/D converter operates as the sensing function.
- **Bit 1**: EXT: Reserved bit. Be sure to write "0" into this bit.
- **Bit 0**: RESV: Reserved bit. Be sure to write "1" into this bit. The read value is always "1".

**Note:**
15.4.3 A/D Data Register (ADCD)

In A/D conversion mode, the result of A/D conversion is stored in this register. In sensing mode, the data of comparison voltage is written into this register.

### A/D Data Registers (ADCD)

Figure 15.4-4 "A/D Data Register (ADCD)" shows the bit configuration of the A/D data register.

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 Fh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXh</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled  
: Unused  
X: Undefined  

- **A/D Conversion mode**

  When A/D conversion is started, the conversion result data is determined after about 44 instruction cycles and it is stored in this register. During the period from the completion of A/D conversion to the start of the next A/D conversion cycle, read the contents of this register (conversion result), write "0" into the ADI bit (bit 3) of the ADC1 register to clear the conversion completion flag.

- **Sensing function mode**

  Before starting the sensing function, set the data corresponding to the voltage to be compared (comparison voltage).

  Check that operation is stopped (ADC2: EXT = 0, ADC1, ADMV = 0) before writing into this register.
## ADCD Register Setting Example in Sensing Mode

Table 15.4-3  ADCD Register Setting Example in Sensing Mode

<table>
<thead>
<tr>
<th>Comparison voltage (V)</th>
<th>FF&lt;sub&gt;H&lt;/sub&gt;</th>
<th>CD&lt;sub&gt;H&lt;/sub&gt;</th>
<th>9A&lt;sub&gt;H&lt;/sub&gt;</th>
<th>66&lt;sub&gt;H&lt;/sub&gt;</th>
<th>33&lt;sub&gt;H&lt;/sub&gt;</th>
<th>00&lt;sub&gt;H&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Condition: $AV_{CC} = AVR = 5.0 \text{ V}, AV_{SS} = 0 \text{ V}$
15.5 A/D Converter Interrupts

The interrupt sources of the A/D converter include:
• Completion of conversion in A/D conversion mode
• A match between the input voltage and the comparison condition in sensing mode

Interrupts in A/D Conversion Mode
At the completion of A/D conversion, the interrupt request flag bit (ADC1: ADI) is set to “1”. When the interrupt request enable bit is enabled (ADC2: ADIE = 1) at this time, an interrupt request to the CPU (IRQ8) occurs. Write “0” into the ADI bit in the interrupt processing routine to clear the interrupt request.

The ADI bit is set to “1” when A/D conversion is completed regardless of the value of the ADIE bit.

Reference:
If the ADIE bit is changed from "disable" to "enable" (0 --> 1) when the ADI bit is "1", an interrupt request occurs immediately.

Interrupts in Sensing Mode
When the comparison of input voltage and comparison voltage is completed in the sensing mode and the input voltage meets the comparison condition, the interrupt request flag bit (ADC1: ADI) is set to “1”. When the interrupt request enable bit is enabled (ADC2: ADIE = 1) at this time, an interrupt request to the CPU (IRQ8) occurs. Write “0” into the ADI bit in the interrupt processing routine to clear the interrupt request.

The ADI bit is set to "1" when the comparison condition is met regardless of the value of the ADIE bit.

Reference:
If the ADIE bit is changed from "disable" to "enable" (0 --> 1) when the ADI bit is "1", an interrupt request occurs immediately.

Register and Vector Table Related to A/D Converter Interrupts

Table 15.5-1 Register and Vector Table Related to A/D Converter Interrupts

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ8</td>
<td>ILR3 (007E&lt;sub&gt;H&lt;/sub&gt;)</td>
<td>FFEA&lt;sub&gt;H&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>L81 (bit 1)</td>
<td>FFEB&lt;sub&gt;H&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>L80 (bit 0)</td>
<td></td>
</tr>
</tbody>
</table>

For interrupt operation, see 3.4.2 “Flow of Interrupt Operation.”
15.6 Operation of A/D Converter

The A/D converter operates in A/D conversion mode and sensing mode. Each function can be operated by software start or continuous start.

### Starting the A/D Conversion Function

#### Software Start

To start the A/D conversion function by software, the setting shown in Figure 15.6-1 “A/D Conversion Function Setting (Software Start)” is required.

![Figure 15.6-1 A/D Conversion Function Setting (Software Start)](image)

When A/D conversion is started, the A/D conversion function starts operation. Even during conversion, the A/D conversion function can be restarted.

### Operation of A/D Conversion Function

The operation of the A/D converter is explained in the following. The time period from the start to the end of A/D conversion is about 44 instruction cycles.

1. When A/D conversion is started, the conversion flag bit is set (ADC1: ADMV = 1) and the set analog input pin is connected to the sample hold circuit.

2. The voltage of the analog input pin is incorporated into the internal sample hold capacitor for about 8 instruction cycles. This voltage is held until A/D conversion is completed.

3. The voltage incorporated into the sample hold capacitor is compared with the reference voltage for A/D conversion from the most significant bit (MSB) to the least significant bit (LSB) in turn with the comparator and the results are transferred to the ADCD register individually.

4. When the transfer of the results is completed, the conversion flag bit is cleared (ADC1: ADMV = 0) and the interrupt request flag bit is set (ADC1: ADI = 1).
Starting the Sensing Function

Software Start

To start the sensing function by software, the setting shown in Figure 15.6-2 “Sensing Function Setting” is required.

**Figure 15.6-2 Sensing Function Setting**

Starting the sensing function starts the sensing operation.

Operation of the Sensing Function

The operation of the sensing function is explained in the following. The time period from the start to the end of the sensing function is about 12 instruction cycles.

1. When the sensing function is started, the conversion flag bit is set (ADC1: ADMV = 1) and the set analog input pin is connected to the sample hold circuit.

2. The voltage of the analog input pin is incorporated into the internal sample hold capacitor for about 8 instruction cycles. This voltage is held until comparison is completed.

3. The voltage incorporated into the sample hold capacitor is compared to the voltage corresponding to the value set in the ADCD register.

4. When the input voltage meets the condition set in the comparison condition setting bit (ADC1: SIFM) as the result of voltage comparison, the interrupt request flag bit is set (ADC1: ADI = 1). When the set condition is not met or when the input voltage equals the set voltage, the ADI bit is not changed.

Reference:

In sensing mode, an interrupt does not occur if the comparison condition is not met, and even at the end of the comparison, no interrupt request occurs. The end of comparison is determined by the conversion flag bit (ADC1: ADMV). When this bit is "0", the comparison is ended.
This section describes precautions when using the A/D converter.

### Caution on Use of the A/D Converter

- **Input impedance of analog input pins**
  
  This A/D converter has a sample hold circuit in it as shown in Figure 15.7-1 "Equivalent Circuit to Analog Input" and incorporates the voltage of the analog input pin into the sample hold capacitor about 8 instruction cycles after the start of A/D conversion (sensing). Therefore, if the output impedance of the external circuit for analog input is high, the analog input voltage may not be stabilized within the analog input sampling period. To avoid this, keep the output impedance of the external circuit low enough to stabilize the voltage (10 kΩ or less). If the output impedance of the external circuit cannot be kept low, it is recommended that a capacitor of around 0.1 μF be added to the analog input pin externally.

![Figure 15.7-1 Equivalent Circuit to Analog Input](image)

- **Precautions when setting with a program**
  
  - In A/D conversion mode, the ADCD register holds the previous values until A/D conversion is started. Immediately after A/D conversion is started, the contents of the ADCD register become undefined.
  
  - During A/D conversion function operation and sensing function operation, do not re-select an analog input channel (ADC1: ANS3 to ANS0) and switch between A/D conversion mode and sensing mode (ADC2: ADMD). Particularly during continuous start, disable the continuous start (ADC2: EXT = 0) and check that the conversion flag bit (ADC1: ADMV) is set to "0" before re-selecting an analog input channel or switching the mode. Similarly, before rewriting the comparison condition setting bit (ADC1: SIFM) in sensing mode, stop the operation of the A/D converter.
  
  - Before writing to the ADCD register in sensing mode, stop the operation of the A/D converter.
  
  - Before switching between A/D conversion mode and sensing mode, clear the interrupt request flag bit (ADC1: ADI = 0).
  
  - Reset operation and the start of stop or watch mode stops the A/D converter.
  
  - If the interrupt request flag bit (ADC1: ADI) is set to "1" and the interrupt request is enabled (ADC2: ADIE = 1), the A/D converter cannot be restored from an interrupt. Be sure to clear the ADI bit.
CHAPTER 15 A/D CONVERTER

<table>
<thead>
<tr>
<th>i</th>
<th>Caution on interrupt request</th>
</tr>
</thead>
<tbody>
<tr>
<td>When restart (ADC1: AD = 1) and completion of A/D conversion occur simultaneously or when restart (ADC1: AD = 1) of the sense function and fulfillment of the comparison condition occur simultaneously, the interrupt request flag bit (ADC1: ADI) is not set.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>On Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>As (</td>
<td>\text{AVR-AVss}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>Temporal Order of turning on A/D converter power supply and applying analog input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrently with or before turning on the power supply of the A/D converter (AVcc, AVss) and applying the analog input (AN00 to AN07), turn on the digital power supply (Vcc).</td>
<td></td>
</tr>
<tr>
<td>Concurrently with or after turning off the A/D converter power supply (AVcc, AVss) and disconnecting analog input (AN00 to AN07), turn off the digital power supply (Vcc).</td>
<td></td>
</tr>
<tr>
<td>When turning on and off the power of the A/D converter, AVcc, AVss, and analog input should not exceed the voltage of the digital power supply.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>Conversion time</th>
</tr>
</thead>
<tbody>
<tr>
<td>The conversion speed in A/D conversion mode and the comparison speed in sensing mode are affected by the clock mode and the main clock speed switching (gear function).</td>
<td></td>
</tr>
</tbody>
</table>
This section shows programming examples of the 8-bit A/D converter in A/D conversion mode and sensing mode.

### A Programming Example of A/D Conversion Function

#### Processing specification
- The analog voltage entered into the AN00 pin is converted into a digital voltage with the AD bit of the ADC1 register. An interrupt is not used and the end of conversion is detected in the loop of the program.

#### Coding example

```assembly
CSEG ; [CODE SEGMENT]

SETB AN00 ; Set the P50/AN00 pin to analog input pin
CLRI ; Interrupt disable

AD_WAIT
BBS ADMV,AD_WAIT ; A/D converter stop checking loop
MOV ADC1,#00000000B ; Select analog input channel 0 (AN00) clear interrupt request flag, do not start software
MOV ADC2,#00000001B ; Disable interrupt request output, select A/D conversion function, select software start with A/D bit
SETI ; Interrupt enable

AD_CONV
SETB AD ; Start A/D conversion

BBS ADMV,AD_CONV ; A/D conversion end wait loop (at about 22 μs /8 MHz)
CLRB ADI ; Clear interrupt request flag
MOV A,ADCD ; Read A/D conversion data

ENDS

END
```

CHAPTER 15 A/D CONVERTER

A Programming Example of Sensing Function

Processing specification

- An interrupt occurs when the analog voltage input to the AN00 pin is lower than 3.0 V.
- Convert and start CH0 with the AD bit of the ADC1 register and sense the voltage.
- Start the sensing function continuously by the timebase timer output (oscillation divided by $2^8$)
- When analog supply voltage (AVcc) = reference voltage (AVR) = 5.0 V, the ADCD register value that serves as the comparison voltage of 3.0 V is 0A9H.
15.8 Programming Examples of A/D Converter

Coding example

PDR5 EQU 0010H ; Address of port register
ADC1 EQU 002DH ; Address of A/D control register 1
ADC2 EQU 002EH ; Address of A/D control register 2
ADCD EQU 002FH ; Address of A/D data register
AN00 EQU PDR5:0 ; Definition of AN00 analog input pin
ADI EQU ADC1:3 ; Definition of interrupt request flag bit
ADMV EQU ADC1:2 ; Definition of conversion flag bit
AD EQU ADC1:0 ; Definition of A/D conversion start bit (software start)
ILR3 EQU 007EH ; Set interrupt level setting register 3
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFEAH
IRQ8 DW WARI
INT_V ENDS

;---------Main program---------------------------------------------------------------
CSEG ; [CODE SEGMENT]
; Assuming that stack pointer (SP) has been initialized

; SETB AN00 ; Set P50/AN00 pin to analog input pin
; CLRI ; Interrupt disable
; MOV ILR3,#11111101B ; Set interrupt level (level 1)
AD_WAIT
BBS ADMV,AD_WAIT ; A/D converter stop checking loop
MOV ADCD,#9AH ; Set comparison voltage data (3.0 V)
MOV ADC1,#00000000B ; Select analog input channel 0 (AN00),
; clear interrupt request flag,
; set comparison condition (interrupt occurs
; when voltage is lower), do not start software
MOV ADC2,#00001111B ; Enable interrupt, select sensing function
; SETI ; Interrupt enable

;---------Interrupt processing routine--------------------------------------------------
WARI CLRB ADI ; Clear interrupt request flag
PUSHW A
XCHW A,T
PUSHW A

; User processing
;
POPW A
XCHW A,T
POPW A
RETI
ENDS

;-----------------------------------------------------------------------------------------------
END
CHAPTER 16  WATCH PRESCALER

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

16.1  "Overview of Watch Prescaler"
16.2  "Watch Prescaler Configuration"
16.3  "Watch Prescaler Control Register (WPCR)"
16.4  "Watch Prescaler Interrupts"
16.5  "Operation of Watch Prescaler"
16.6  "Precautions on Use of Watch Prescaler"
16.7  "Watch Prescaler Programming Example"
16.1 Overview of Watch Prescaler

The watch prescaler is a 15-bit free run counter that counts up in synchronization with the sub-clock pulse generated in the clock generator. In its interval timer function, there are four optional types of interval time. The watch prescaler also provides operation clock pulses such as the timer output of the sub-clock oscillation stabilization wait time and the watchdog timer.

### Interval Timer Function (Watch Interrupts)

The interval timer function generates interrupts repeatedly at constant time intervals with the sub-clock as a count clock.

- The watch prescaler generates interrupts with the frequency divided output for the interval timer.
- There are four optional types of frequency divided output for the interval timer (interval time).
- The counter of the watch prescaler can be cleared.

Table 16.1-1 "Interval Time of the Watch Prescaler" shows the interval time of the watch prescaler.

<table>
<thead>
<tr>
<th>Sub-clock cycle</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1/F_{CL}$ (Approx. 30.5 μs)</td>
<td>$2^{10}/F_{CL}$ (31.25 ms)</td>
</tr>
<tr>
<td></td>
<td>$2^{13}/F_{CL}$ (0.25 s)</td>
</tr>
<tr>
<td></td>
<td>$2^{14}/F_{CL}$ (0.50 s)</td>
</tr>
<tr>
<td></td>
<td>$2^{15}/F_{CL}$ (1.00 s)</td>
</tr>
</tbody>
</table>

$F_{CL}$: Sub-clock source oscillation

The values in parentheses are those at a sub-clock source oscillation of 32.768 kHz.

**Note:**

The watch prescaler cannot be used when the one clock system is selected in optional setting.
16.1 Overview of Watch Prescaler

### Clock Pulse Supply Function

The clock pulse supply function of the watch prescaler supplies one type of timer output for the sub-clock oscillation stabilization wait time, as well as three types of clock pulse for the watchdog timer and for buzzer output.

Table 16.1-2 "Clock Cycles Supplied from the Watch Prescaler" shows the clock cycles supplied from the watch prescaler to peripheral functions.

<table>
<thead>
<tr>
<th>Sub-clock pulse supply destination</th>
<th>Sub-clock cycle</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-clock oscillation stabilization wait time</td>
<td>$2^{15}/F_{CL}$ (1.00 s)</td>
<td>Do not move to sub-clock mode during the oscillation stabilization wait time.</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>$2^{14}/F_{CL}$ (0.50 s)</td>
<td>Watchdog timer count-up clock</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>$2^{3}/F_{CL}$ to $2^{5}/F_{CL}$ (Approx. 0.24 ms to approx. 0.98 ms)</td>
<td>See Chapter 12 &quot;BUZZER OUTPUT.&quot;</td>
</tr>
</tbody>
</table>

$F_{CL}$: Sub-clock source oscillation
The values in parentheses are those at a sub-clock source oscillation of 32.768 kHz.

Reference:

Since the oscillation cycle is unstable immediately after the start of oscillation, the oscillation stabilization wait time is only a guide.
16.2 Watch Prescaler Configuration

The watch prescaler consists of the following four blocks

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

The values in parentheses are those at a sub-clock source oscillation of 32.768 kHz.

- **Counter for watch prescaler**
  The counter for watch prescaler is a 15-bit up counter with sub-clock source oscillation as the count clock.

- **Counter clear circuit**
  The counter clear circuit clears the counter when setting is made to clear the counter in the WPCR register (WCLR = "0") and also when the transition to sub-stop mode occurs (STBC: STP = 1). A power-on reset (option) also occurs.
Interval timer selector
The interval timer selector is a circuit that selects one interval time from the four types of frequency divided output of the counter for watch prescaler. The falling edge of the selected frequency divided output becomes an interrupt source.

Watch prescaler control register (WPCR)
The watch prescaler control register (WPCR) selects interval time, clears the counter, controls interrupts, and checks the state.
CHAPTER 16 WATCH PRESCALER

16.3 Watch Prescaler Control Register (WPCR)

The watch prescaler control register (WPCR) selects interval time, clears the counter, controls interrupts, and checks the state.

Watch Prescaler Control Register (WPCR)

Figure 16.3-1 Watch Prescaler Control Register (WPCR)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 B5</td>
<td>WIF</td>
<td>WIE</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WS1</td>
<td>WS0</td>
<td>WCLR</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
—: Unused
X: Undefined

<table>
<thead>
<tr>
<th>WCLR</th>
<th>Watch prescaler clear bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Read cycle</td>
</tr>
<tr>
<td>1</td>
<td>Write cycle</td>
</tr>
<tr>
<td>1<em>1</em> is always read.</td>
<td>Clear watch prescaler</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WS1</th>
<th>WS0</th>
<th>Watch interrupt interval time selection bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>2^0/FCL</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2^1/FCL</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2^2/FCL</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2^3/FCL</td>
</tr>
</tbody>
</table>

FCL: Sub-clock source oscillation

<table>
<thead>
<tr>
<th>WIE</th>
<th>Interrupt request enable bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disable interrupt request output</td>
</tr>
<tr>
<td>1</td>
<td>Enable interrupt request output</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WIF</th>
<th>Watch interrupt request flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Read cycle</td>
</tr>
<tr>
<td>1</td>
<td>Write cycle</td>
</tr>
<tr>
<td>1</td>
<td>Interval interrupt occurred Clear this bit</td>
</tr>
</tbody>
</table>

No change, no influence on other operation.

Initial value

No change, no influence on other operation.
### Table 16.3-1 Functions of Each Bit in the Watch Prescaler Control Register (WPCR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| Bit 7 | WIF: Watch interrupt request flag bit | - This bit is set to "1" at the falling edge of the selected frequency divided output for interval timer.  
- When this bit and the interrupt request enable bit (WIE) are set to "1", an interrupt request is output.  
- In write cycle, this bit is cleared when "0" is written and does not affect operation when "1" is written. |
| Bit 6 | WIE: Interrupt request enable bit             | A bit for enabling/disabling interrupt request output to the CPU. When this bit and the watch interrupt request flag bit (WIF) are set to "1", an interrupt request is output. |
| Bit 5 | Unused bit                      | - The value in read cycle is undefined.  
- In write cycle, this bit does not affect operation. |
| Bit 4 | Unused bit                      |                                                                                                                                          |
| Bit 3 | Unused bit                      |                                                                                                                                          |
| Bit 2 | WS1, WS0: Watch interrupt interval time selection bit | - Bits for selecting the cycle of the interval timer  
- A bit for interval timer (frequency divided output) of the watch prescaler counter is specified.  
- There are four optional types of interval time |
| Bit 1 |                                    |                                                                                                                                          |
| Bit 0 | WCLR: Watch prescaler clear bit    | - A bit for clearing the watch prescaler counter  
- Writing "0" into this bit clears the counter to "0000_H", while writing "1" into this bit does not affect the bit or other operations. |

**Reference:**  
Read value is always "1".
16.4 Watch Prescaler Interrupts

The watch prescaler generates an interrupt request at the falling edge of the selected frequency divided output (interval timer function).

- **Interrupts in Interval Timer Mode (Watch Interrupt)**
  The watch prescaler counter counts up with the sub-clock source oscillation. When the set interval timer time expires, the watch interrupt request flag bit is set to "1" (WPCR: WIF = 1) in a mode other than the main stop mode. When the interrupt request enable bit is enabled (WPCR: WIE = 1) at this time, an interrupt request to the CPU (IRQB) occurs. Write "0" into the WIF bit in the interrupt processing routine to clear the interrupt request. The WIF bit is set to "1" when the specified frequency divided output falls regardless of the value of the WIE bit.

  **Note:**
  To enable the interrupt request output (WIE = 1) after reset, be sure to clear the WIF bit at the same time (WIF = 0).

  **Reference:**
  If the WIE bit is changed from "disable" to "enable" (0 → 1) when the WIF bit is "1", an interrupt request occurs immediately.
  When clearing of the counter (WPCR: WCLR = 0) and an overflow of the selected bit occur simultaneously, the WIF bit is not set.

- **Oscillation Stabilization Wait Time and Watch Interrupt**
  If an interval time shorter than the sub-clock oscillation stabilization wait time is set, a watch interrupt request of the watch prescaler (WPCR: WIF = 1) occurs when restoring from sub-stop mode due to external interrupt. In this case, disable the watch prescaler interrupt (WPCR: WIE = 0) when moving to sub-stop mode.

- **Register and Vector Table Related to Watch Prescaler Interrupts**
  Table 16.4-1 "Register and Vector Table Related to Watch Prescaler Interrupts" shows the register and vector table related to watch prescaler interrupts.

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQB</td>
<td>ILR3 (007E_H)</td>
<td>FFE4_H FFE5_H</td>
</tr>
</tbody>
</table>

For interrupt operation, see Section 3.4.2 "Flow of Interrupt Operation."
16.5 Operation of Watch Prescaler

The watch prescaler operates as an interval timer and as a clock pulse supply.

- **Operation in Interval Timer Mode**
  
  To operate as an interval timer, the setting shown in Figure 16.5-1 "Interval Timer Function Setting" is required.

  ![Figure 16.5-1 Interval Timer Function Setting](image)

  The 15-bit counter of the watch prescaler continues to count up with the sub-clock as the count clock as long as the sub-clock continues oscillation.

  When the counter is cleared (WCLR = 0), the counter counts up from "0000H" until it reaches "7FFFH". When it reaches "7FFFH", it returns to "0000H" and continues to count up. If a falling edge occurs in the selected frequency divided output for interval timer during a count operation, the watch interrupt request flag bit (WIF) is set to "1" in a mode other than the main stop mode. That is, a watch interrupt request is generated in every selected interval time with the cleared time as a reference.

- **Operation in Clock Supply Mode**

  The watch prescaler is also used as a timer for making the sub-clock oscillation stabilization wait time. The timer counts up from the state where the watch prescaler is cleared and the period up to the falling of the most significant bit is the sub-clock oscillation stabilization wait time ($2^{15}/F_{CL}$, $F_{CL}$: sub-clock source oscillation).

  The watch prescaler supplies clock pulses to the watchdog timer and the buzzer output. Clearance of the watch prescaler counter affects the operation of the buzzer output. The counter of the watchdog timer is cleared at the same time if the watch prescaler output is selected (WDTC: CS = 1).
Operation of the Watch Prescaler

Figure 16.5-2 "Operation of the Watch Prescaler" shows the state of the counter when transition to sleep or stop mode occurs during the operation of the interval timer function and when a counter clear request is issued.

Transition to watch mode is the same as the transition to sub-sleep mode.

Figure 16.5-2 Operation of the Watch Prescaler

When the interval time selection bits of the watch prescaler control register (WPCR: WS1, SW0) is set to “11b” (2^{15}/FCL)
16.6 Precautions on Use of Watch Prescaler

This section describes precautions on use of the watch prescaler. The watch prescaler cannot be used when the single clock system is selected in optional setting.

- Precautions on Use of the Watch Prescaler

  - **Precautions on setting in a program**
    Restoration from interrupt processing cannot be made in states where the interrupt request flag bit (WPCR: WIF) is set to "1" and the interrupt request enable bit is enabled (WPCR: WIE = 1). Be sure to clear the WIF bit.

  - **Clearing the watch prescaler**
    The watch prescaler is cleared when the watch prescaler clear bit is set to "0" (WPCR: WCLR = 0) and also when the sub-clock oscillation stabilization wait time is required.
    When the watch prescaler is selected for the count clock of the watchdog timer (WDTC: CS = 1), the watchdog timer is cleared when the watch prescaler is cleared.

  - **Precautions on use of a timer for oscillation stabilization wait time**
    Since the oscillation of the sub-clock is stopped after turning on the power and in sub-stop mode, the oscillation stabilization wait time is taken by the watch prescaler when the oscillator starts operation.
    Do not move from main clock mode to sub-clock mode during a sub-clock oscillation stabilization wait time, such as immediately after turning on power.
    The sub-clock oscillation stabilization wait time is fixed.
    For details, see Section 3.6.5 "Oscillation Stabilization Wait Time."

  - **Precaution on watch interrupt**
    In main stop mode, the watch prescaler performs a count operation but a watch interrupt (IRQB) does not occur.
Precautions when using peripheral functions to which clock pulses are supplied from the watch prescaler

When the watch prescaler counter is cleared, a clock pulse supplied from the watch prescaler is output from the initial state. Therefore, the "H" level pulse may become shorter or the "L" level pulse may become longer up to 1/2 cycles.

Though the clock for the watchdog timer also outputs a pulse from the initial state, the watchdog timer operates in normal cycle because the counter of the watchdog timer is cleared at the same time.

Figure 16.6-1 "Effect of the Cleared Watch Prescaler on the Buzzer Output" shows the effect of the cleared watch prescaler on the buzzer output.

**Figure 16.6-1 Effect of the Cleared Watch Prescaler on the Buzzer Output**

Counter value

<table>
<thead>
<tr>
<th>Counter value</th>
<th>Clock pulse supplied to buzzer output</th>
</tr>
</thead>
<tbody>
<tr>
<td>001FH</td>
<td></td>
</tr>
<tr>
<td>0010H</td>
<td></td>
</tr>
<tr>
<td>0000H</td>
<td></td>
</tr>
</tbody>
</table>

The counter is cleared by a program. (WPCR:WCLR=0)

When the buzzer selection bits of the buzzer register (BZCR: BZ2, BZ1, BZ0) are set to "101" (1024 Hz is output when operating in 32.768 kHz, the sub-clock source oscillation divided by 32)
A programming example of the watch prescaler is shown in the following.

**A Programming Example of the Watch Prescaler**

- **Processing specifications**
  Watch interrupts of \(2^{15}/F_{CL}\) (\(F_{CL}\) sub-clock source oscillation) are generated repeatedly. The interval time is one second (at 32.768 kHz operation).

- **Coding example**

```assembly
WPCR EQU 000BH ; Address of watch prescaler control register
WIF EQU WPCR:7 ; Definition of watch interrupt request flag bit
ILR3 EQU 007EH ; Address of interrupt level setting register
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFEAH ; Set interrupt vector
IRQB DW WARI
INT_V ENDS

;--------------Main program----------------------------------------------------------------------------------------------- ----------
CSEG ; [CODE SEGMENT]
; Assuming that stack pointer (SP) has been initialized
CLRI ; Interrupt disable
MOV ILR3,#10111111B ; Set interrupt level (level 2)
MOV WPCR,#01000110B ; Clear interrupt request flag, enable interrupt request output, select \(2^{15}/F_{CL}\), clear watch prescaler
SETI ; Interrupt enable

;--------------Interrupt program--------------------------------------------------------------------------------------------------
WARI CLRIB WIF ; Clear interrupt request flag
PUSHW A
XCHW A,T
PUSHW A
; User processing
; POPW A
XCHW A,T
POPW A
RETI
ENDS

;------------------------------------------------------------------------------------------------------------------------------
END
```
CHAPTER 17  DTMF GENERATOR

This chapter describes the functions and operations of the DTMF generator.

17.1 "Overview of DTMF Generator"
17.2 "DTMF Generator Configuration"
17.3 "Pin of DTMF Generator"
17.4 "Registers of DTMF Generator"
17.5 "Operation of DTMF Generator"
17.6 "DTMF Generator Programming Example"
17.1 Overview of DTMF Generator

The DTMF generator can continuously output all the ITU-T (old CCITT) recommended tones that can be used in telephones, including "0" through "9", "+", ",", and "A" through "D".

Functions of the DTMF Generator

- The DTMF generator generates dial tone signals that can be used in telephones and outputs the signals from the DTMF pin.
- DTMF signals can be output continuously. (Single tones can also be output.)
- All the ITU-T (old CCITT) recommended tones can be output.
- At a clock oscillation other than a main clock source oscillation of 4 MHz or 8 MHz, accurate DTMF signals cannot be generated. When 1200 bps or 2400 bps modem output is used, however, use 3.9744 MHz or 7.9488 MHz.
- Table 17.1-1 "Correspondence between Set Values and Output Frequencies (Main Clock Source Oscillation: 4 MHz)" shows the correspondence table of set values and output frequencies. Table 17.1-2 "Reference Frequencies and Frequency Deviations" shows frequency deviations and reference frequencies.
### Table 17.1-1 Correspondence between Set Values and Output Frequencies (Main Clock Source Oscillation: 4 MHz)

<table>
<thead>
<tr>
<th>Dial number</th>
<th>Low frequency tone</th>
<th>High frequency tone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ROW</td>
<td>COL</td>
</tr>
<tr>
<td></td>
<td>Frequency (Hz)</td>
<td>Frequency (Hz)</td>
</tr>
<tr>
<td>1</td>
<td>ROW1</td>
<td>COL1</td>
</tr>
<tr>
<td>2</td>
<td>ROW1</td>
<td>COL2</td>
</tr>
<tr>
<td>3</td>
<td>ROW1</td>
<td>COL3</td>
</tr>
<tr>
<td>4</td>
<td>ROW2</td>
<td>COL1</td>
</tr>
<tr>
<td>5</td>
<td>ROW2</td>
<td>COL2</td>
</tr>
<tr>
<td>6</td>
<td>ROW2</td>
<td>COL3</td>
</tr>
<tr>
<td>7</td>
<td>ROW3</td>
<td>COL1</td>
</tr>
<tr>
<td>8</td>
<td>ROW3</td>
<td>COL2</td>
</tr>
<tr>
<td>9</td>
<td>ROW3</td>
<td>COL3</td>
</tr>
<tr>
<td>0</td>
<td>ROW4</td>
<td>COL2</td>
</tr>
<tr>
<td>*</td>
<td>ROW4</td>
<td>COL1</td>
</tr>
<tr>
<td>#</td>
<td>ROW4</td>
<td>COL3</td>
</tr>
<tr>
<td>A</td>
<td>ROW1</td>
<td>COL4</td>
</tr>
<tr>
<td>B</td>
<td>ROW2</td>
<td>COL4</td>
</tr>
<tr>
<td>C</td>
<td>ROW3</td>
<td>COL4</td>
</tr>
<tr>
<td>D</td>
<td>ROW4</td>
<td>COL4</td>
</tr>
</tbody>
</table>

### Table 17.1-2 Reference Frequencies and Frequency Deviations

<table>
<thead>
<tr>
<th>Reference frequency (ITU-T recommendation)</th>
<th>DTMF output frequency*1</th>
<th>Frequency deviation*1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROW1</td>
<td>692.89 Hz</td>
<td>-0.59%</td>
</tr>
<tr>
<td>ROW2</td>
<td>765.24 Hz</td>
<td>-0.62%</td>
</tr>
<tr>
<td>ROW3</td>
<td>846.34 Hz</td>
<td>-0.66%</td>
</tr>
<tr>
<td>ROW4</td>
<td>934.71 Hz</td>
<td>-0.67%</td>
</tr>
<tr>
<td>COL1</td>
<td>1200.00 Hz</td>
<td>-0.75%</td>
</tr>
<tr>
<td>COL2</td>
<td>1328.34 Hz</td>
<td>-0.57%</td>
</tr>
<tr>
<td>COL3</td>
<td>1465.49 Hz</td>
<td>-0.78%</td>
</tr>
<tr>
<td>COL4</td>
<td>1623.53 Hz</td>
<td>-0.58%</td>
</tr>
</tbody>
</table>

*1: Main clock source oscillation: 7.9488 MHz
17.2 DTMF Generator Configuration

The DTMF generator consists of the following nine blocks
- Voltage data adder
- COL staircase wave generator circuit
- ROW staircase wave generator circuit
- ROW/COL decoder
- Frequency divider circuit
- Selector
- Control signal generator
- DTMF control register (DTMC)
- DTMF data register (DTMD)

Block Diagram of the DTMF Generator

Figure 17.2-1 Block Diagram of the DTMF Generator

- **Voltage data adder**
  The voltage data adder adds and synthesizes high frequency tone signals and low frequency tone signals and outputs the dial tone signals.
17.2 DTMF Generator Configuration

- **COL staircase wave generator circuit**
  The COL staircase wave generator circuit generates high frequency tone signals.

- **ROW staircase wave generator circuit**
  The ROW staircase wave generator circuit generates low frequency tone signals.

- **ROW/COL decoder**
  The ROW/COL decoder selects the frequency to be generated with DTMD data.

- **Frequency divider circuit**
  The frequency divider circuit divides the main clock and generates clock pulses for operating the DTMF generator.

- **Selector**
  The selector selects the operation clock of the DTMF generator.

- **Control signal generator**
  The control signal generator controls the output of high group and low frequency tones.

- **DTMF control register (DTMC)**
  The DTMF control register (DTMC) enables/disables the output of high frequency tones and low frequency tones, enables/disables the output of DTMF signals, and selects the main clock source oscillation frequency used.

- **DTMF data register (DTMD)**
  The DTMF data register (DTMD) sets the dial data to be output. With this data, the frequency to generate high and low groups and the DTMF signal to be output is determined.
17.3 Pin of DTMF Generator

This section describes the DTMF generator related pin and the block diagram of the pin.

- **Pin Related to the DTMF Generator**
  
  The pin related to the DTMF generator is the DTMF pin.
  
  This pin is used only for outputting DTMF signals.

- **Block Diagram of the Pin Related to the DTMF Generator**

  ![Block Diagram of the DTMF Pin](image-url)

  Figure 17.3-1  Block Diagram of the DTMF Pin
17.4 Registers of DTMF Generator

This section describes the registers related to the DTMF generator

- Registers related to the DTMF Generator

**Figure 17.4-1 Registers Related to the DTMF Pin**

<table>
<thead>
<tr>
<th>DTMC (DTMF control register)</th>
<th>Address</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>Initial value</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 0</td>
<td></td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>CSEL</td>
<td>CDIS</td>
<td>RDIS</td>
<td>OUTE</td>
<td>XXX000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- DTMC (DTMF control register)

<table>
<thead>
<tr>
<th>DTMD (DTMF data register)</th>
<th>Address</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>Initial value</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 1</td>
<td>RESV</td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>——</td>
<td>DDAT3</td>
<td>DDAT2</td>
<td>DDAT1</td>
<td>DDAT0</td>
<td>XXX000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- DTMD (DTMF data register)

R/W: Read/Write enabled

- Unused

X: Undefined
17.4.1 DTMF Control Register (DTMC)

The DTMF control register (DTMC) enables/disables the output of high frequency tones and low frequency tones, enables/disables the output of DTMF signals, and selects the main clock source oscillation frequency to be used.

The diagram illustrates the DTMF Control Register (DTMC) with its bit assignments and initial values.

- **OUTE**: DTMF output control bit
  - 0: Disable DTMF signal output
  - 1: Enable DTMF signal output

- **RDIS**: Low frequency tone generation control bit
  - 0: Enable generation of low frequency tones
  - 1: Disable generation of low frequency tones

- **CDIS**: High frequency tone generation control bit
  - 0: Enable generation of high frequency tones
  - 1: Disable generation of high frequency tones

- **CSEL**: Frequency selection bit
  - 0: Main clock source oscillation 4 MHz
  - 1: Main clock source oscillation 8 MHz

**Address**: XXXX 0000

**Initial value**: XXXX0000h

R/W: Read/Write enabled

- : Unused

X: Undefined
### 17.4 Registers of DTMF Generator

#### Table 17.4-1 Functions of Each Bit in the DTMF Control Register (DTMC)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>Unused bit</td>
</tr>
</tbody>
</table>
| Bit 6    | • The value in read cycle is undefined.  
| Bit 5    | • In write cycle, this bit has no influence on operation.  
| Bit 4    | |
| Bit 3    | CSEL: Frequency selection bit  
|          | • A bit for selecting the operation clock of the DTMF generator. Set the frequency of the main clock source oscillation used.  
|          | • Note: When the main clock source oscillation is something other than 4 MHz or 8 MHz, accurate DTMF signals cannot be generated.  
| Bit 2    | CDIS: High frequency tone generation control bit  
|          | • A bit for enabling/disabling the generation of high frequency tone signals.  
|          | • When this bit is "0", high frequency tones are generated; when this bit is "1", they are not generated.  
| Bit 1    | RDIS: Low frequency tone generation control bit  
|          | • A bit for enabling/disabling the generation of low frequency tone signals.  
|          | • When this bit is "0", low frequency tones are generated; when this bit is "1", they are not generated.  
| Bit 0    | OUTE: DTMF output control bit  
|          | • This bit enables/disables DTMF signal output.  
|          | • When this bit is "1", DTMF signals are output; when this bit is "0", they are not output.  

17.4.2 DTMF Data Register (DTMD)

The DTMF data register (DTMD) sets the dial data to be output.

Figure 17.4-3 DTMF Data Register (DTMD)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 1</td>
<td>RESV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0XXX0000b</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
---: Unused
X: Undefined

DTMF data register (DTMD) | Low frequency tone | High frequency tone | Dial number
---|---|---|---
ROW1 | 692.89 | 1200.00 | 1
ROW1 | 692.89 | 1328.34 | 2
ROW1 | 692.89 | 1465.49 | 3
ROW2 | 765.24 | 1200.00 | 4
ROW2 | 765.24 | 1328.34 | 5
ROW2 | 765.24 | 1465.49 | 6
ROW3 | 846.34 | 1200.00 | 7
ROW3 | 846.34 | 1328.34 | 8
ROW3 | 846.34 | 1465.49 | 9
ROW4 | 934.71 | 1200.00 | *
ROW4 | 934.71 | 1328.34 | #
ROW4 | 934.71 | 1465.49 | A
ROW2 | 765.24 | 1623.53 | B
ROW3 | 846.34 | 1623.53 | C
ROW4 | 934.71 | 1623.53 | D

RESV: Reserved bit
0: Be sure to write 0 into this bit.
The DTMF generator can generate dial tone signals.

### Operation of the DTMF Generator

To operate the DTMF generator, the setting shown in Figure 17.5-1 "DTMF Generator Setting" is required.

![Figure 17.5-1 DTMF Generator Setting](image)

When the output of DTMF signals is enabled, the dial tone signals set in the DTMF data register (DTMD) are output from the DTMF pin.
This section gives a programming example of the GTMF generator.

A Programming Example of the DTMF Generator

- Processing specifications
  Dial tone signals (low frequency tone: ROW3 851.79 MHz, high frequency tone: COL1 1207.73 MHz) of dial number “7” are generated when the main clock source oscillation is 4 MHz.

- Coding example

```assembly
DTMC EQU 0020H ; Address of DTMF control register
DTMD EQU 0021H ; Address of DTMF data register
CSEL EQU DTMC:3 ; Definition of frequency selection bit
CDIS EQU DTMC:2 ; Definition of high frequency tone generation control bit
RDIS EQU DTMC:1 ; Definition of low frequency tone generation control bit
OUTE EQU DTMC:0 ; Definition of DTMF output control bit

;------------Main program-----------------------------------------------------------------------------------------------------------
CSEG ; [CODE SEGMENT]
: MOV DTMD,#00000111B ; Set dial number "7" to DTMF data register
CLR CSEL ; Set frequency selection bit to main clock source oscillation: 4 MHz
CLR CDIS ; Enable high frequency tone generation
CLR RDIS ; Enable low frequency tone generation
SET OUTE ; Enable DTMF output
:
ENDS
:------------------------------------------------------------------------------------------------------------------------------
END
```

END
CHAPTER 18  MODEM TIMER

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

18.1  "Overview of Modem Timer"
18.2  "Modem Timer Configuration"
18.3  "Pin of Modem Timer"
18.4  "Registers of Modem Timer"
18.5  "Modem Timer Interrupts"
18.6  "Operation of Low-Pass Filter of Modem Timer"
18.7  "Operation of Modem Timer Function (Rising Edge Detection)"
18.8  "Operation of Modem Timer Function (Falling Edge Detection)"
18.9  "Operation of Modem Timer Function (Both Edge Detection)"
18.10 "Precaution on Use of Modem Timer"
18.11 "Modem Timer Programming Example"
18.1 Overview of Modem Timer

The modem timer functions to measure the pulse width of the external clock entered from the pin. For pulse widths, the "H + L" width, "L + H" width, and "H" and "L" widths can be measured. With an internal digital filter and the function for handling edge detection and counter overflow, the modem timer can take measurements in a stable manner.

■ Modem Timer Function

The modem timer function measures the pulse cycle entered to the external pin (MSKI) as the "H + L" width (the width from the rising edge to the next rising edge), "L + H" width (the width from the falling edge to the next falling edge), and "H" and "L" widths.

○ Measurement function

- The pulse width can be measured continuously.
- With the "H" width data storage register and "L" width data storage register, the pulse width and also the values of the "H" and "L" widths can be measured.
- There are four optional types of internal count clock for measurement.
- The pulse width that can be measured with a counter ranges from a cycle of the internal count clock to a cycle $2^8$ times greater than the cycle of the count clock. However, longer pulse widths can be measured using counter overflow interrupts.

Table 18.1-1 "Counter Interval Time of the Modem Timer" shows the counter interval time of the modem timer.

**Table 18.1-1 Counter Interval Time of the Modem Timer**

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 $t_{inst}$</td>
<td>2 $t_{inst}$ to $2^9$ $t_{inst}$</td>
</tr>
<tr>
<td>4 $t_{inst}$</td>
<td>$2^2$ $t_{inst}$ to $2^{10}$ $t_{inst}$</td>
</tr>
<tr>
<td>8 $t_{inst}$</td>
<td>$2^3$ $t_{inst}$ to $2^{11}$ $t_{inst}$</td>
</tr>
<tr>
<td>16 $t_{inst}$</td>
<td>$2^4$ $t_{inst}$ to $2^{12}$ $t_{inst}$</td>
</tr>
</tbody>
</table>

$t_{inst}$: Instruction cycle (influenced by divide-by-two CPU operation clock and clock mode)
18.1  Overview of Modem Timer

Noise removal function

- A 5-bit digital low-pass filter is built in.
- As a clock source for noise canceller, one internal count clock can be selected from four types.
- The removable noise width is a maximum of five times the cycle of the noise canceller clock. Table 18.1-2 "Noise Removing Capability" shows the noise removing capability.

Table 18.1-2  Noise Removing Capability

<table>
<thead>
<tr>
<th>Internal count clock</th>
<th>Noise canceller clock cycle</th>
<th>Maximum noise width</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 ( t_{\text{inst}} )</td>
<td>5 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td></td>
<td>2 ( t_{\text{inst}} )</td>
<td>10 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td></td>
<td>4 ( t_{\text{inst}} )</td>
<td>20 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td></td>
<td>8 ( t_{\text{inst}} )</td>
<td>40 ( t_{\text{inst}} )</td>
</tr>
</tbody>
</table>

\( t_{\text{inst}} \): Instruction cycle (influenced by CPU operation clock divided by two and clock mode)

Note:
The input pulse having a cycle equal to or less than half the maximum removable noise width cannot be measured.
18.2 Modem Timer Configuration

The modem timer consists of the following seven blocks
- Count clock selector
- Noise canceller clock selector
- Digital low-pass filter
- Edge detector
- Counter circuit
- Modem timer control register 1 (MDC1)
- Modem timer control register 2 (MDC2)

**Figure 18.2-1 Block Diagram of the Modem Timer**

- tinst: instruction cycle (influenced by divide-by-two CPU operation clock and clock mode)
- *1: A pulse is output when a rising edge is detected
- *2: A pulse is output when a falling edge is detected
18.2 Modem Timer Configuration

- **Count clock selector**
  The count clock selector selects a measurement clock from the four types of internal clock.

- **Noise canceller clock selector**
  The noise canceller clock selector selects a sampling clock for noise removal from the four types of internal clock.

- **Digital low-pass filter**
  The digital low-pass filter removes noise from an input pulse (MSKI) with the noise canceller clock and outputs a noise-removed pulse (MSIG).

- **Edge detector**
  The edge detector detects a rising or falling edge of the MSIG pulse and generates a pulse in synchron with its timing.

- **Counter circuit**
  The counter circuit consists of an 8-bit counter, "H" level data register (MLDH), and "L" level register (MLDL).
  The 8-bit counter counts up with the selected count clock. When a rising or falling edge of the selected edge is detected, the value in the 8-bit counter is stored in the MLDL register (when a rising edge is detected) or the MLDH register (when a falling edge is detected) and the 8-bit counter is then cleared.

- **Modem timer control register 1 (MDC1)**
  The modem timer control register 1 (MDC1) selects functions, controls interrupts, and checks the state.

- **Modem timer control register 2 (MDC2)**
  The modem timer control register 2 (MDC2) selects a count clock and a noise canceller clock.

- **Modem timer-related interrupt factors**
  **IRQ3:**
  An interrupt request (IRQ3) occurs if either of the following two conditions is met in the modem timer function.
  - **Edge detection interrupt**
    The edge selected in the edge selection/interrupt enable bit of the modem timer control register 1 (MDC1: ESL1, ESL0) is detected in the input pulse.
  - **Overflow interrupt**
    The overflow interrupt is enabled (MDC1: MOIE = 1) at the time of counter overflow.
18.3 Pin of Modem Timer

This section describes the modem timer related pin and provides a block diagram of the pin.

- Pin Related to the Modem Timer
  
  The pin related to the modem timer is the P32/MSKI pin.

- P32/MSKI pin
  
  This pin serves as a general-purpose I/O port (P32) and an external clock pulse input pin of the modem timer (MSKI).

  MSKI: The cycle or the "H" and "L" widths of the clock pulse entered into this pin are measured.

- Block Diagram of the Pin Related to the Modem Timer

  ![Block Diagram of the Pin Related to the Modem Timer]

  **Figure 18.3-1 Block Diagram of the Pin Related to the Modem Timer**

  - PDR (port data register)
  - DDR (Port direction register)
  - Internal data bus
  - Output latch
  - Latch
  - Pch
  - Nch
  - SPL: Pin state specification bit of standby control register (STBC)
  - Pull-up resistor (optional) Approx. 50 kΩ (5 V)
18.4 Registers of Modem Timer

This section describes the registers related to the modem timer.

- Registers Related to the Modem Timer

![Figure 18.4-1 Registers Related to the Modem Timer]

**MDC1 (Modem timer control register 1)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 4H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XX0XX00XB</td>
</tr>
<tr>
<td></td>
<td>R</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>

**MDC2 (Modem timer control register 2)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 5H</td>
<td>RESV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0XXX0000B</td>
</tr>
<tr>
<td></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>
<td></td>
</tr>
</tbody>
</table>

**MLDH (Modem "H" level data register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 6H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXB</td>
</tr>
<tr>
<td></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></td>
</tr>
</tbody>
</table>

**MLDL (Modem "L" level data register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 3 7H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXA</td>
</tr>
<tr>
<td></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></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
R: Read only
—: Unused
X: Undefined
Modem timer control register 1 (MDC1) selects the modem timer function, controls interrupts, and checks the state.

### Modem Timer Control Register 1 (MDC1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>R/W</th>
<th>Description</th>
<th>Value</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>R</td>
<td>Modem Overflow Flag (HOF)</td>
<td>0</td>
<td>Clear</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>No change, no influence on other operation</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>Modem Overflow Flag (LOF)</td>
<td>0</td>
<td>Clear</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>No change, no influence on other operation</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>W</td>
<td>Edge selection/interrupt enable (ESL1)</td>
<td>0</td>
<td>Disable interrupts</td>
</tr>
<tr>
<td>2</td>
<td>W</td>
<td>Disabling edge detection</td>
<td>1</td>
<td>Enable interrupts</td>
</tr>
<tr>
<td>1</td>
<td>W</td>
<td>Both edge detection</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>W</td>
<td>No edge detected</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Edge detected</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>Overflow interrupt request enable (MOIE)</td>
<td>0</td>
<td>Disable request</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabling overflow interrupt request output</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>Level sense bit of noise-removed pulses (MSIG)</td>
<td>0</td>
<td>“L” level</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>“H” level</td>
</tr>
</tbody>
</table>

**Figure 18.4-2 Modem Timer Control Register 1 (MDC1)**
### Table 18.4-1 Functions of Each Bit in Modem Timer Control Register 1 (MDC1)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>Unused bit • The value in read cycle is undefined. • In write cycle, this bit has does not affect operation.</td>
</tr>
<tr>
<td>Bit 6</td>
<td>MSIG: Level sense bit of noise-removed pulses • A bit for indicating the level of the input pulse from which noise is removed by the digital low-pass filter. • When this bit is &quot;0&quot;, the pulse is &quot;L&quot; level; when this bit is &quot;1&quot;, the pulse is &quot;H&quot; level. <strong>Reference:</strong> This bit is read only. The written value has no significance. For the pulse before noise removal, read the value of the port (P32).</td>
</tr>
<tr>
<td>Bit 5</td>
<td>MOIE: Overflow interrupt request enable bit • A bit for enabling/disabling the CPU interrupt request output • An interrupt request is output if either the &quot;H&quot; level overflow bit (HOF) or the &quot;L&quot; level overflow bit (LOF) is &quot;1&quot; when this bit is set.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>HOF: Overflow flag bit for &quot;H&quot; level measurement (interrupt request flag bit) • This bit is set to &quot;1&quot; if the MSIG pulse is at the &quot;H&quot; level when the counter overflows. • When this bit and the interrupt request enable bit (MOIE) are set to &quot;1&quot;, an interrupt request is output. • In write cycle, this bit is cleared when &quot;0&quot; is written and does not affect operation when &quot;1&quot; is written.</td>
</tr>
<tr>
<td>Bit 3</td>
<td>LOF: Overflow flag bit for &quot;L&quot; level measurement (interrupt request flag bit) • This bit is set to &quot;1&quot; if the MSIG pulse is at the &quot;L&quot; level when the counter overflows. • When this bit and the interrupt request enable bit (MOIE) are set to &quot;1&quot;, an interrupt request is output. • In write cycle, this bit is cleared when &quot;0&quot; is written and does not affect operation when &quot;1&quot; is written.</td>
</tr>
<tr>
<td>Bit 2, Bit 1</td>
<td>ESL1, ESL0: Edge selection/interrupt request enable bit Bits for selecting the detection edge. When either edge is selected, interrupt request output is enabled. • When these bits are set to &quot;01B&quot;, one cycle of the pulse (&quot;H&quot; level width + &quot;L&quot; level width) is measured by detecting only rising edges and its count value is stored in the &quot;L&quot; level register. An interrupt request is then generated. • When these bits are set to &quot;10B&quot;, one cycle of the pulse (&quot;L&quot; level width + &quot;H&quot; level width) is measured by detecting only falling edges and its count value is stored in the &quot;H&quot; level register. An interrupt request is then generated. • When these bits are set to &quot;11B&quot;, both edges are detected. For a falling pulse, the &quot;H&quot; level width measurement value is stored in the &quot;H&quot; level register. For a rising pulse, the &quot;L&quot; level width measurement value is stored in the &quot;L&quot; level register. An interrupt request is generated in each edge. • When these bits are set to &quot;00B&quot;, edge detection and interrupt request output are disabled.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>MDIF: Edge detection flag bit (interrupt request flag bit) • This bit is set to &quot;1&quot; when the edge specified in the ESL1 and ESL0 bits is detected. • In write cycle, this bit is cleared when &quot;0&quot; is written and does not affect operation when &quot;1&quot; is written.</td>
</tr>
</tbody>
</table>
18.4.2 Modem Timer Control Register 2 (MDC2)

Modem timer control register 2 (MDC2) selects a count clock for measuring the input clock to the modem timer and a clock source of the sampling clock for digital filter.

Figure 18.4-3 Modem Timer Control Register 2 (MDC2)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 5h</td>
<td>RESV</td>
<td>—</td>
<td>—</td>
<td>MCS1</td>
<td>MCS0</td>
<td>NCS1</td>
<td>NCS0</td>
<td></td>
<td>0XXX0000b</td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled
—: Unused
X: Undefined
: Initial value
Table 18.4-2 Functions of Each Bit in Modem Timer Control Register 2 (MDC2)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>RESV: Reserved bit</td>
</tr>
<tr>
<td></td>
<td>• Be sure to write &quot;0&quot; into this bit.</td>
</tr>
<tr>
<td></td>
<td>• The read value is always &quot;0&quot;</td>
</tr>
<tr>
<td>Bit 6</td>
<td>Unused bit</td>
</tr>
<tr>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• The value in read cycle is undefined.</td>
</tr>
<tr>
<td></td>
<td>• In write cycle, this bit does not affect operation.</td>
</tr>
<tr>
<td>Bit 3</td>
<td>MCS1, MCS0: Count clock selection bits</td>
</tr>
<tr>
<td>Bit 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Bits for selecting a count clock used for measuring the pulse width.</td>
</tr>
<tr>
<td></td>
<td>• Select one from the four types of internal clock.</td>
</tr>
<tr>
<td>Bit 1</td>
<td>NCS1, NCS0: Clock selection bits for noise canceller</td>
</tr>
<tr>
<td>Bit 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Bits for selecting a reference clock for noise canceller used by the</td>
</tr>
<tr>
<td></td>
<td>• Select one from the four types of internal clock.</td>
</tr>
</tbody>
</table>
18.4.3 Modem Timer "H" Level Data Register (MLDH)

The modem timer "H" level data register (MLDH) is a read-only 8-bit data register in which the counter value is latched at each falling edge of the noise-removed pulse (MSIG).

Figure 18.4-4 "Modem Timer "H" Level Data Register (MLDH)" shows the bit configuration of the modem timer "H" level data register.
18.4.4 Modem Timer "L" Level Data Register (MLDL)

The modem timer "L" level data register (MLDL) is a read-only 8-bit data register in which the counter value is latched at each rising edge of the noise-removed pulse (MSIG).

Figure 18.4-5 "Modem Timer "L" Level Data Register (MLDL)" shows the bit configuration of the modem timer "L" level data register.
18.5 Modem Timer Interrupts

The interrupt sources of the modem timer are the edge detection of a noise-removed pulse (MSIG) during a measurement and counter overflow.

## Interrupts of the Modem Timer

Table 18.5-1 "Relation between Modem Timer Interrupt Control Bits and Interrupt Sources" shows the relation between modem timer interrupt control bits and interrupt sources.

### Table 18.5-1 Relation between Modem Timer Interrupt Control Bits and Interrupt Sources

<table>
<thead>
<tr>
<th>Interrupt occurrence condition</th>
<th>Detection edge selection</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Rising edge</td>
<td>Falling edge</td>
<td>Both edges</td>
</tr>
<tr>
<td>1 MSIG edge detection</td>
<td>MDC1: MDIF (Edge detection flag bit)</td>
<td>None (Enabled unconditionally when the detection edge is selected)</td>
<td>Rising edge detection</td>
<td>Falling edge detection</td>
</tr>
<tr>
<td>2 Counter overflow</td>
<td>MDC1: HOF, LOF (Overflow flag bit)</td>
<td>MDC1: MOIE (Overflow interrupt request enable bit)</td>
<td>Counter overflow</td>
<td></td>
</tr>
</tbody>
</table>

When the detection edge is selected with the edge selection/interrupt enable bits of modem timer control register 1 (MDC1: ESL1, ESL0), interrupts are enabled automatically, the edge detection flag bit (MDC1: MDIF) is set to "1" at the rising edge (when the rising edge detection is selected or when both edge detection is selected) or at the falling edge (when the falling edge detection is selected or when both edge detection is selected), and an interrupt request to the CPU (IRQ3) occurs. Write "0" into the MDIF bit in the interrupt processing routine to clear the interrupt request.

This interrupt request is generated repeatedly until edge detection is disabled (MDC1: ESL1, ESL0 = "00"B) or the input pulse is fixed to "L" or "H."

If the noise-removed input pulse (MSIG) is at the "L" level when the counter overflows, the overflow flag bit for "L" level measurement (LOF) is set to "1"; if it is at the "H" level, the overflow flag bit for "H" level measurement (HOF) is set to "1".

When the overflow interrupt request is enabled (MDC1: MOIE = 1), an interrupt request to the CPU (IRQ3) occurs. Clear the LOF flag and HOF flag in the interrupt processing routine.

As described above, if the overflow interrupt request is enabled, the edge detection interrupt and overflow interrupt generate the same interrupt request. Therefore, it is necessary to determine which interrupt request occurs with each flag bit.
Register and Vector Table Related to Modem Timer Interrupts

Table 18.5-2  Register and Vector Table Related to Modem Timer Interrupts

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ3</td>
<td>ILR1 (007C\text{H})</td>
<td>L31 (bit 7)</td>
</tr>
</tbody>
</table>

Reference

For interrupt operation, see Section 3.4.2 “Flow of Interrupt Operation.”
18.6 Operation of Low-Pass Filter of Modem Timer

This section describes the operation of the digital low-pass filter of the modem timer.

■ Operation of the Digital Low-pass Filter

When an input pulse (MSKI) passes through the low-pass filter, it is turned into a noise-removed pulse (MSIG).

To operate the digital low-pass filter, the setting shown in Figure 18.6-1 "Digital Low-pass Filter Setting" is required.

Figure 18.6-1 Digital Low-pass Filter Setting

![Digital Low-pass Filter Setting Diagram]

An input pulse (MSKI) is always sampled by the sampling clock (T_SMP) selected with the clock selection bits for the noise canceller (NCS1, NCS2) and converted into Levels 0 to 6 by integration. When this level reaches 0, the level of the output pulse (MSIG) is cleared to "L." When this level reaches 6, it is set to "H." In the intermediate levels between 1 and 5, the output is not changed. By this operation, a noise of up to five sampling clocks in width can be removed.

Note:

Maximum removable noise width (N_W) = sampling clock cycle (T_SMP) x 5
Example: When $F_{CH} = 8 \text{ MHz}$, NCS1, NCS0 = "11 B", $N_W$ is calculated in the following expression.

$$N_W = 4.0 \ \mu\text{s} \times 5 = 20 \ \mu\text{s}$$
CHAPTER 18 MODEM TIMER

18.7 Operation of Modem Timer Function (Rising Edge Detection)

This section describes the operation when the rising edge detection is selected for the modem timer.

Operation of the Modem Timer Function (Rising Edge Detection)

To operate as a modem timer that detects rising edges, the setting shown in Figure 18.7-1 "Modem Timer Function Setting (at Rising Edge Detection)" is required.

Figure 18.7-1 Modem Timer Function Setting (at Rising Edge Detection)

Whenever a rising edge of the input pulse (MSIG) from which noise is removed by the digital filter is detected, the counter value is latched in the "L" level data register (MLDL) and the counter is cleared to "00H". The latched value is equivalent to one cycle of the "H" and "L" level widths.

The moment the rising edge detection is selected by setting the detection edge selection bits (ESL1, ESL0) to "01B", interrupts at edge detection are also enabled. Therefore, when a rising edge is detected, the edge detection flag bit (MDIF) is set to "1" and an interrupt to the CPU occurs.

When the counter overflows during measurement, either LOF or HOF bit is set to "1" depending on the level of the MSIG pulse. If the overflow interrupt request enable bit (MOIE) is set to "1" at this time, an interrupt request occurs.

Figure 18.7-2 "Operation of the Modem Timer (Rising Edge Detection)" shows the operation of the modem timer when the rising edge detection is selected.
18.7 Operation of Modem Timer Function (Rising Edge Detection)

Figure 18.7-2 Operation of the Modem Timer (Rising Edge Detection)
18.8 Operation of Modem Timer Function (Falling Edge Detection)

This section describes the operation when the falling edge detection is selected for the modem timer.

■ Operation of the Modem Timer Function (Falling Edge Detection)

To operate as a modem timer that detects falling edges, the setting shown in Figure 18.8-1 "Modem Timer Function Setting (at Falling Edge Detection)" is required.

Figure 18.8-1 Modem Timer Function Setting (at Falling Edge Detection)

Whenever a falling edge of the input pulse (MSIG) from which noise is removed by the digital filter is detected, the counter value is latched in the "H" level data register (MLDH) and the counter is cleared to "00H". The latched value is equivalent to one cycle of the "L" and "H" level widths.

The moment the falling edge detection is selected by setting the detection edge selection bits (ESL1, ESL0) to "10B", interrupts at edge detection are also enabled. Therefore, when a falling edge is detected, the edge detection flag bit (MDIF) is set to "1" and an interrupt to the CPU occurs.

When the counter overflows during measurement, either LOF or HOF bit is set to "1" depending on the level of the MSIG pulse. If the overflow interrupt request enable bit (MOIE) is set to "1" at this time, an interrupt request occurs.

Figure 18.8-2 "Operation of the Modem Timer (Falling Edge Detection)" shows the operation of the modem timer when the falling edge detection is selected.
Figure 18.8-2 Operation of the Modem Timer (Falling Edge Detection)
18.9 Operation of Modem Timer Function (Both Edge Detection)

This section describes the operation when both (rising and falling) edge detections are selected for the modem timer.

Operation of the Modem Timer Function (Both Edge Detection)

To operate as a modem timer that detects both edges, the setting shown in Figure 18.9-1 "Modem Timer Function Setting (at Both Edge Detection)" is required.

Figure 18.9-1 Modem Timer Function Setting (at Both Edge Detection)

```
<table>
<thead>
<tr>
<th>MDC1</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>MSIG</td>
<td>MOIE</td>
<td>HOF</td>
<td>LOF</td>
<td>ESL1</td>
<td>ESL0</td>
<td>MDIF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>MDC2</th>
<th>RESV</th>
<th>---</th>
<th>---</th>
<th>MCS1</th>
<th>MCS0</th>
<th>NCS1</th>
<th>NCS0</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>
<td>0</td>
</tr>
</tbody>
</table>
```

Each time a rising or falling edge of the input pulse (MSIG) from which noise is removed by the digital filter is detected, the counter value is latched in the "L" level data register (MLDL, for the rising edge) and the "H" level data register (MLDH, for the falling edge), respectively, and the counter is cleared to "00H". The latched value is equivalent to the "L" level width (for the rising edge) or the "H" level width (for the falling edge) of the pulse, respectively.

The moment that both edge detections are selected by setting the detection edge selection bits (ESL1, ESL0) to "11B", interrupts at edge detection are also enabled. When a rising edge is detected, the edge detection flag bit (MDIF) is set to "1" and an interrupt to the CPU occurs. When a falling edge is detected, the same operation is performed.

When the counter overflows during measurement, either LOF or HOF bit is set to "1" depending on the level of the MSIG pulse. If the overflow interrupt request enable bit (MOIE) is set to "1" at this time, an interrupt request occurs.

Figure 18.9-2 "Operation of the Modem Timer (Both Edge Detection)" shows the operation of the modem timer when both edge detections are selected.
Figure 18.9-2 Operation of the Modem Timer (Both Edge Detection)
18.10 Precaution on Use of Modem Timer

This section describes the precautions when using the modem timer.

- Precaution When Using the Modem Timer

- Precautions when setting in a program
  - The 8-bit counter cannot be cleared by a program.
  - When the detection edge is selected, the counter is latched and cleared when the specified edge is detected and an interrupt is generated. However, the value in the counter latch at the detection of the first specified edge has no significance. Do not use the counter latch value at the first edge detection interrupt.

  When overflow interrupts are enabled, an overflow interrupt may occur before the first edge detection interrupt but this overflow also has no significance.

  - Even if the count clock is selected again by the count clock selector, the 8-bit counter is not cleared.

    The counter latch value becomes valid from the second edge detection interrupt.

  - In overflow interrupts, restoration from interrupt cannot be made in states where either of the overflow flag bits (MDC1: HOF, LOF) is set to "1" and the overflow interrupt is enabled (MDC1: MOIE = 1). Be sure to clear the overflow flag bit in the interrupt routine.

  - In edge detection interrupts, restoration from an interrupt cannot be made in states where the edge detection flag bit (MDC1: MDIF) is set to "1" and the edge detection interrupt is enabled. Be sure to clear the edge detection flag bit even during the edge detection interrupt routine.

  - Since the count clock and the noise canceller clock in the modem timer supply pulses of the frequency divided main clock, they do not work accurately when the main clock is in a mode other than ordinary operation mode or sleep mode. Particularly in sub-clock ordinary operation mode or sleep mode, do not use the modem timer.
18.11 Modem Timer Programming Example

A programming example of the modem timer is shown in the following.

- A Programming Example of the Modem Timer

  - Processing specifications
    - The count clock is $4 \, t_{\text{inst}}$ and the noise canceller clock is $2 \, t_{\text{inst}}$.
    - The rising edge detection and overflow interrupt are enabled.
    - The second and later rising edge interrupts are valid. For interrupts before the second rising edge interrupt, the corresponding flag is cleared.
CHAPTER 18 MODEM TIMER

Coding example

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR3</td>
<td>EQU</td>
<td>000DH</td>
</tr>
<tr>
<td>MDC1</td>
<td>EQU</td>
<td>0034H</td>
</tr>
<tr>
<td>MDC2</td>
<td>EQU</td>
<td>0035H</td>
</tr>
<tr>
<td>MLDH</td>
<td>EQU</td>
<td>0036H</td>
</tr>
<tr>
<td>MLDL</td>
<td>EQU</td>
<td>0037H</td>
</tr>
<tr>
<td>ILR1</td>
<td>EQU</td>
<td>007CH</td>
</tr>
<tr>
<td>MSK1</td>
<td>EQU</td>
<td>DDR3:2</td>
</tr>
<tr>
<td>MDIF</td>
<td>EQU</td>
<td>MDC1:0</td>
</tr>
<tr>
<td>LOF</td>
<td>EQU</td>
<td>MDC1:3</td>
</tr>
<tr>
<td>HOF</td>
<td>EQU</td>
<td>MDC1:4</td>
</tr>
</tbody>
</table>

DSEG
WARI_CNT
RB 1 ; Interrupt flag to detect the first interrupt.
WARI_F EQU WARI_CNT:0 ; Only one bit is used
LOFCNT RB 1 ; How many times did "H" level overflow occur
HOFCNT RB 1 ; How many times did "L" level overflow occur
ENDS

INT_V DSEG ABS ; Set interrupt vector, specify absolute address
ORG 0FFF4H
IRQ3 DW WARI ; Set interrupt vector
ENDS

;-----------Main program-----------------------------------------------------------------------------------------------
CSEG ; [CODE SEGMENT]
; Assuming that stack pointer (SP) has been initialized
; Interrupt disable
CLRI
CLRB WARI_F ; Clear interrupt flag
MOV ILR1,#10111111B ; Set interrupt level (level 2)
MOV DDR3,#11111011B ; Set bit 2 to input
MOV MDC1,#00100010B ; Enable overflow interrupt, clear HOF/LOF, detect rising edge, clear MDIF
MOV MDC2,#00000101B ; Count clock 4_tinst
    Noise canceller clock 2 tinst
SETI ; CPU interrupt enable
interrupt program

WARI PUSHW A
XCHW A,T ; Save A and T
PUSHW A
BBC MDIF,OF_WARI ; Another processing if it is overflow interrupt

CLR B MDIF ; [Edge detection interrupt]
BBS WARI_F,ED_WARI ; Clear interrupt request

SETB WARI_F ; If it is first interrupt, validate subsequent interrupts
JMP WARI_RET ; [Edge interrupt]

ED_WARI

User processing

JMP WARI_RET ; [Overflow interrupt]

OF_WARI
BBC WARI_F,OF_NO ; If valid edge interrupt has not been generated, clear the flag.

OF_YES

User processing

OF_NO
CLR B LOF ; Perform overflow processing here.
CLR B HOF ; For example, the number of overflows is counted and the pulse width is corrected.

WARI_RET
POPW A ; Clear overflow
XCHW A,T ; Restore A and T
POPW A
RETI

END
This chapter describes the functions and operations of the modem signal output circuit.

19.1 "Overview of the Modem Signal Output Circuit"
19.2 "Configuration of the Modem Signal Output Circuit"
19.3 "Pin of the Modem Signal Output Circuit"
19.4 "Registers of the Modem Signal Output Circuit"
19.5 "Interrupt of the Modem Signal Output Circuit"
19.6 "Description of Operations when the Modem Signal Output Circuit is Used Alone"
19.7 "Description of Operations when Output Data of the Serial I/O with 1-Byte Buffer is Used"
19.8 "Precautions When Using the Modem Signal Output Circuit"
19.9 "Program Example of the Modem Signal"
19.1 Overview of the Modem Signal Output Circuit

The modem signal output circuit has a function to output transmitting data after converting it into a modem signal (before modulation). The baud rate and the source oscillation each can be selected from the two types. Output data from the serial I/O with 1-byte buffer may be used as transmitting data.

■ Modem Signal Output Function

The modem signal output function is a function to output transmitting data after converting it into a modem signal (digital signal before modulation).

- It is possible to select either of two square wave output frequencies; 1208 bps or 2415 bps.
- 4 MHz or 8 MHz can be selected as the source oscillation frequency.
- To make the output frequency 1200 bps or 2400 bps, select 3.9744 MHz or 7.9488 MHz as the source oscillation frequency.
- Transmitting data is written to the data register in 1 bit units, or output data of the serial I/O with 1-byte buffer is used so that data is automatically sent to the data register.
- By supplying the clock generated by the built-in serial I/O clock generator as a shift clock of the serial I/O with 1-byte buffer (The serial I/O with 1-byte buffer operates in external clock mode), data receiving from the serial I/O and modem signal output can be linked.
- The output pin from which no data is being output can be brought to a high impedance output state.

Figure 19.1-1 "Output Waveform Diagram of the Modem Signal Output Circuit" shows an output waveform diagram of the modem signal output circuit.
19.1 Overview of the Modem Signal Output Circuit

Figure 19.1-1 Output Waveform Diagram of the Modem Signal Output Circuit

(For 1208 bps transfer)
Transmitting data
(MODA register value)

Pin output signal
(MSKO pin)

Modulation filter output signal
(for reference, outside chip)

(For 2415 bps transfer)
Transmitting data
(MODA register value)

Pin output signal
(MSKO pin)

Modulation filter output signal
(for reference, outside chip)
19.2 Configuration of the Modem Signal Output Circuit

The modem signal output circuit consists of the following seven blocks:
- Baud rate counter circuit
- Cycle counter circuit
- Output control circuit
- Serial clock generator
- Shift control circuit
- Modem output data register (MODA)
- Modem output control register (MODC)

![Block Diagram of the Modem Signal Output Circuit](image)

*1: Serial I/O port switching circuit
*2: Serial I/O with 1-byte buffer
*3: Output data
19.2 Configuration of the Modem Signal Output Circuit

- **Baud rate counter circuit**
  One cycle is determined by generating the standard clock by dividing the main clock signal and then counting the clock. Each time one cycle is counted, the counter is reset and the trigger signal for modem output reversal is sent to the output control circuit.

- **Cycle counter circuit**
  The 1-bit sending cycle when the modem output bit is "0" and "1" is defined for each of 1208 bps and 2415 bps. By counting the clock for this cycle, the 1-bit send time is adjusted.

- **Output control circuit**
  Each time the output reverse signal of the baud rate counter circuit is input, the output of the modem output pin (MSKO) is reversed. By settings of the MODC register, the MSKO pin can be brought to high impedance state after all transmitting data bits are sent.

- **Serial I/O clock generator**
  The serial I/O clock generator generates a shift clock and supplies it to the serial I/O with 1-byte buffer when data of the serial I/O with 1-byte buffer is output as a modem signal. This shift clock is also supplied to the data latch circuit so that data is sent to the MDA0 bit of the MODA register in each shift clock.

- **Shift control circuit**
  The circuit to determine the shift timing of the MODA register. When transmitting data is written to the MODA register bit by bit by a program, the shift timing is determined by the cycle counter. When output data of the serial I/O with 1-byte buffer is used, the shift timing is determined by the shift clock generated by the serial I/O clock generator.

- **Modem output data register (MODA)**
  Data register composed of the 2-bit data buffer to hold transfer data and the 1-bit empty flag.
  Data is written bit by bit. If data is shifted in the register by the shift control circuit, it is output and the empty flag bit is set.

- **Modem output control register (MODC)**
  Control register of the modem signal output circuit. This register is used to enable/disable the operation of modem signal output, select the baud rate, control the interrupts, control the output pins, and specify how to input transmitting data.

- **Interrupt of the modem signal output circuit**
  **IRQ2:**
  An interrupt request (IRQ2) is generated if the data empty interrupt request output is enabled (MODC: DEIE=1) when there is no 1-bit data (MODA: DEMP=1) that has not been transferred in the modem output data register.
19.3 Pin of the Modem Signal Output Circuit

This section describes the pin related to the modem signal output circuit and a block diagram of the pin.

- **Pin Related to the Modem Signal Output Circuit**

  The pin related to the modem signal output circuit is the P63/MSKO pin.

  - **P63/MSKO pin**

    This pin functions both as a general-purpose I/O port (P63) and as a modem signal output pin (MSKO).

    **MSKO:**

    To use the pin as a modem signal output pin, set it as an input port (DDR6: bit=0) by the port direction register.

- **Block Diagram of the Modem Signal Output Circuit**

![Block Diagram of the Pin Related to the Modem Signal Output Circuit](image-url)
19.4 Registers of the Modem Signal Output Circuit

This section describes the registers related to the modem signal output circuit.

### Registers Related to the Modem Signal Output Circuit

![Figure 19.4-1 Registers Related to the Modem Signal Output Circuit](image)

**MODC (modem output control register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 A</td>
<td>HIZE</td>
<td>DEIE</td>
<td>SIOE</td>
<td>MOEN</td>
<td>RESV</td>
<td>CSL1</td>
<td>CSL0</td>
<td>STAT</td>
<td>00000000b</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>
<td></td>
</tr>
</tbody>
</table>

**MODA (modem output data register)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 B</td>
<td>DEMP</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>MDA1</td>
<td>MDA0</td>
</tr>
<tr>
<td>R</td>
<td>R</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Read/Write enabled  
R: Read-only
CHAPTER 19 MODEM SIGNAL OUTPUT CIRCUIT

19.4.1 Modem Output Control Register (MODC)

The modem output control register (MODC) is a register used to select various functions of the modem signal output, enable/disable the operations, and control the interrupt.

- Modem Output Control Register (MODC)

![Figure 19.4-2 Modem Output Control Register (MODC)](image)

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 A</td>
<td>HIZE</td>
<td>DEIE</td>
<td>SIOE</td>
<td>MOEN</td>
<td>RESV</td>
<td>CSL1</td>
<td>CSL0</td>
<td>STAT</td>
<td>0000000b</td>
</tr>
</tbody>
</table>

- **R/W**: Read/Write enabled
- **Initial value**: Indicates the initial state of the bit

### Field Descriptions
- **HIZE**: Hi-z output enable bit
  - 0: Enable high impedance output
  - 1: Disable high impedance output
  - The current output level is retained.
- **DEIE**: Data empty interrupt request enable bit
  - 0: Disable interrupt request output
  - 1: Enable interrupt request output
- **SIOE**: Serial I/O with 1-byte buffer enable bit
  - 0: Enable the serial I/O with 1-byte buffer output
  - 1: Disable the serial I/O with 1-byte buffer output
- **MOEN**: Modem output enable bit
  - 0: Use the P63/MSKO pin as a general-purpose port
  - 1: Use the P63/MSKO pin as a modem output pin
- **RESV**: Reserved bit
  - Be sure to write "0" to this bit.
- **STAT**: Modem output start bit
  - 0: Stop modem output
  - 1: Disable modem output
- **CSL1**: Baud rate selection bit
  - 0 0: 1208 bps output (source oscillation 4 MHz)
  - 0 1: 2415 bps output (source oscillation 4 MHz)
  - 1 0: 1208 bps output (source oscillation 8 MHz)
  - 1 1: 2415 bps output (source oscillation 8 MHz)
### Table 19.4-1 Modem Output Control Register (MODC)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>HIZE: Hi-z output enable bit</td>
</tr>
<tr>
<td></td>
<td>- Bit to enable/disable high impedance output of the MSKO pin while the pin is not in modem output</td>
</tr>
<tr>
<td></td>
<td>- If this bit is &quot;0&quot;, the high impedance output is enabled. If this bit is &quot;1&quot;, the high impedance output is disabled and the current state is retained.</td>
</tr>
<tr>
<td>Bit 6</td>
<td>DEIE: data empty interrupt request enable bit</td>
</tr>
<tr>
<td></td>
<td>- Bit to enable/disable interrupt request output to the CPU. If this bit and the data empty flag bit (MODA: DEMP) of the modem output data register are &quot;1&quot;, an interrupt request is output.</td>
</tr>
<tr>
<td>Bit 5</td>
<td>SIOE: Serial I/O with 1-byte buffer enable bit</td>
</tr>
<tr>
<td></td>
<td>- Bit to enable/disable the use of output of the serial I/O with 1-byte buffer as modem data.</td>
</tr>
<tr>
<td></td>
<td>- If this bit is &quot;0&quot;, the use of output of the serial I/O with 1-byte buffer is enabled. If this bit is &quot;1&quot;, the use of output of the serial I/O with 1-byte buffer is disabled.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>MOEN: modem output enable bit</td>
</tr>
<tr>
<td></td>
<td>- If this bit is &quot;0&quot;, the P63/MSKO pin functions as a general-purpose port (P63). If this bit is &quot;1&quot;, the P63/MSKO pin functions as a modem output pin (MSKO).</td>
</tr>
<tr>
<td></td>
<td>- Reference: If the modem output (MSKO) is enabled, the pin functions as an output pin even if the port direction register corresponding to the general-purpose port (P63) is set for input (DDR6: bit3=0) and thus functions as the MSKO pin.</td>
</tr>
<tr>
<td>Bit 3</td>
<td>RESV: reserved bit</td>
</tr>
<tr>
<td></td>
<td>- Note: Be sure to write &quot;0&quot; to this bit. &quot;0&quot; is always read from this bit.</td>
</tr>
<tr>
<td>Bit 2</td>
<td>CSL1, CSL0: baud rate selection bit</td>
</tr>
<tr>
<td>Bit 1</td>
<td>- Bits to specify the main clock signal frequency and baud rate</td>
</tr>
<tr>
<td></td>
<td>- Select 4 MHz or 8 MHz for the signal frequency, then select 1208 bps or 2415 bps for each.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>STAT: modem output start bit</td>
</tr>
<tr>
<td></td>
<td>- Bit to enable/disable the modem output.</td>
</tr>
<tr>
<td></td>
<td>- If this bit is &quot;0&quot;, the modem output stops.</td>
</tr>
<tr>
<td></td>
<td>- Data transfer is started if &quot;1&quot; is written to this bit, and data is then written to the modem output data register (MODA). To use the output of the serial I/O with 1-byte buffer as modem data, write &quot;1&quot; to this bit and then start the serial transfer.</td>
</tr>
</tbody>
</table>
19.4.2 Modem Output Data Register (MODA)

The modem output data register (MODA) is that composed of the modem output data write bit, its shift bit, and the data empty flag bit.

![Modem Output Data Register (MODA)](image)

- **Modem Output Data Register (MODA)**

---

**Figure 19.4-3 Modem Output Data Register (MODA)**

<table>
<thead>
<tr>
<th>Address</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>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 2 B₄</td>
<td>DEMP</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>XXXXXXXₓ</td>
</tr>
</tbody>
</table>

- **R**: Read
- **R/W**: Read/Write enabled
- **X**: Undefined
- **—**: Unused

**Modem data write bit**

<table>
<thead>
<tr>
<th>When reading</th>
<th>When writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&quot;0&quot; is transferred as bit data.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;1&quot; is transferred as bit data.</td>
</tr>
</tbody>
</table>

**Modem data shift bit**

| 0 | Bit data shifted last is "0". |
| 1 | Bit data shifted last is "1". |

**Data empty flag bit**

| 0 | Modem data write bit data are valid (before shifting). |
| 1 | Modem data write bit data are invalid (shifted). |

---
### Table 19.4-2  Modem Output Data Register (MODA)

<table>
<thead>
<tr>
<th>Bit name</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>DEMP:</strong> data empty flag bit</td>
</tr>
<tr>
<td></td>
<td>• &quot;0&quot; in this bit indicates that there are data in the modem data buffer bit0 (MDA0). &quot;1&quot; in this bit indicates that MDA0 is empty.</td>
</tr>
<tr>
<td></td>
<td>• The DEMP bit is cleared when data are written to the MDA0 bit from a program or output data of the serial I/O with 1-byte buffer is latched and then sent to the MDA0 bit.</td>
</tr>
<tr>
<td>Bit 6</td>
<td><strong>Unused bit</strong></td>
</tr>
<tr>
<td>Bit 5</td>
<td>• Values when these bits are read are not defined.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>• Writing to these bits does not affect operations.</td>
</tr>
<tr>
<td>Bit 3</td>
<td></td>
</tr>
<tr>
<td>Bit 2</td>
<td></td>
</tr>
<tr>
<td>Bit 1</td>
<td><strong>MDA1:</strong> modem data shift bit</td>
</tr>
<tr>
<td></td>
<td>• Buffer bit for shifting the modem data write bit (MDA0). The value of MDA0 is shifted and then stored. This bit is not cleared if the modem data transfer is completed.</td>
</tr>
<tr>
<td></td>
<td>• Read-only bit. The value shifted last is read out when this bit is read.</td>
</tr>
<tr>
<td>Bit 0</td>
<td><strong>MDA0:</strong> modem data write bit</td>
</tr>
<tr>
<td></td>
<td>• Bit to write transfer data to. This bit is not cleared if this bit is shifted to the MDA1 bit.</td>
</tr>
<tr>
<td></td>
<td>• The value written last is read out when this bit is read.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td></td>
<td>• Bit manipulation instructions (SETB, CLRb) can also be used for writing data. However, since data can be written only to this bit in this register, the MOV instruction can also be used.</td>
</tr>
<tr>
<td></td>
<td>• When using output data of the serial I/O with 1-byte buffer, transfer data is automatically written. In this case, do not write data to this bit during transfer from a program.</td>
</tr>
</tbody>
</table>
19.5 Interrupt of the Modem Signal Output Circuit

In the modem signal output circuit, data empty becomes an interrupt source.

■ Interrupt of the Modem Signal Output Circuit

In the modem signal output circuit, an interrupt request (IRQ2) is generated if the data empty flag bit (MODA: DEMP) is set to "1" when the data empty interrupt request enable bit is enabled (MODC: DEIE=1).

Since the initial value of the DEMP bit is "1", an interrupt request is immediately generated if the interrupt request output is enabled when data is not being transferred.

Each time the data shift occurs in the modem output data register, the DEMP bit is set to "1", thereby generating an interrupt request. The interrupt source is cleared by writing data to the data buffer.

Figure 19.5-1 "Operation of the Modem Signal Output Circuit" shows an interrupt operation of the modem signal output circuit.

Figure 19.5-1 Operation of the Modem Signal Output Circuit

Reference:

In a sending program using the interrupts, transmitting data is provided in the main routine. An interrupt request is generated if the data empty interrupt is enabled (MODC: DEIE=1). Thus, by creating an interrupt routine in such a way that an interrupt occurs after writing 1-bit data before returning, data transmission is completed after as many interrupts as there are required bits. When the last bit is written, it is necessary to disable the data empty interrupt (MODC: DEIE=0).
Table 19.5-1 Register and Vector Table Related to the Interrupt of the Modem Signal Output Circuit

<table>
<thead>
<tr>
<th>Interrupt name</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Set bit</td>
</tr>
<tr>
<td>IRQ2</td>
<td>ILR1 (007CH)</td>
<td>L21 (bit 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L20 (bit 4)</td>
</tr>
</tbody>
</table>

For interrupt operations, see Section 3.4.2 "Flow of Interrupt Operation."
19.6 Description of Operations when the Modem Signal Output Circuit is Used Alone

This section describes the operations when the modem signal output circuit is used alone.

Operations of the Modem Signal Output Circuit

When using the modem signal output circuit alone, the modem signal (before modulation) can be output by writing transmitting data bit by bit.

Figure 19.6-1 "Setting of the Modem Signal Output Circuit (when Used Alone)" shows the setting required to use the modem signal output circuit alone.

![Figure 19.6-1 Setting of the Modem Signal Output Circuit (when Used Alone)](image)

If data is written to the modem data write bit (MODA: MDA0) after enabling the modem output (MODC: STAT=1), it is automatically shifted to the modem data shift bit to start modem signal output. By writing the next data before the 1-bit data transmission is completed, data can be output as a continuous modem signal. If there is no data when the 1-bit data transmission is completed and the data empty flag bit (MODA: DEMP) is set to "1", data transmission is completed. If, at this time, the Hi-z output enable bit (MODC: HIZE) is "0", the modem output (MSKO) pin is set in high impedance state. If the Hi-z output enable bit is "1", the last output is retained.

Since an interrupt request is generated when the data empty interrupt request enable bit (MODC: DEIE) is set to "1" while the DEMP bit is "1", transmission using the interrupts can be carried out.

Figure 19.6-2 "Transmission Flow of the Modem Signal Output Circuit (when Used Alone)" and Figure 19.6-3 "Operation of the Modem Signal Output Circuit (when Used Alone)" show transmission flow and operation when the modem signal output circuit is used alone.
19.6 Description of Operations when the Modem Signal Output Circuit is Used Alone

Figure 19.6-2 Transmission Flow of the Modem Signal Output Circuit (when Used Alone)

- Settings of the signal, baud rate, and interrupt enable/disable
- Write “1” to the STAT bit
- Set data to MDA0
- MDA0 → MDA1, cycle counter/baud rate counter start, MSKO output
  - Baud rate counter matched?
    - NO  Baud rate counter + 1
    - YES
  - Cycle counter matched?
    - NO  Cycle counter + 1
    - YES
  - DEMP flag = 1?
    - NO
    - YES  Transfer end, MSKO pin Hi-z
Figure 19.6-3  Operation of the Modem Signal Output Circuit (when Used Alone)

(Transfer start timing when the frequency is 2415 bps)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAT</td>
<td></td>
</tr>
<tr>
<td>MODA write</td>
<td></td>
</tr>
<tr>
<td>MODA load</td>
<td></td>
</tr>
<tr>
<td>DEMP</td>
<td></td>
</tr>
<tr>
<td>Baud rate clock</td>
<td></td>
</tr>
<tr>
<td>Baud rate counter</td>
<td></td>
</tr>
<tr>
<td>MDA0</td>
<td>0, 1</td>
</tr>
<tr>
<td>MDA1</td>
<td>0, 1</td>
</tr>
<tr>
<td>MSKO</td>
<td>Hi-z</td>
</tr>
</tbody>
</table>

(Transfer end timing when the frequency is 2415 bps)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAT</td>
<td></td>
</tr>
<tr>
<td>MODA write</td>
<td></td>
</tr>
<tr>
<td>MODA load</td>
<td></td>
</tr>
<tr>
<td>DEMP</td>
<td></td>
</tr>
<tr>
<td>Baud rate clock</td>
<td></td>
</tr>
<tr>
<td>Baud rate counter</td>
<td></td>
</tr>
<tr>
<td>MDA1</td>
<td>1, X</td>
</tr>
<tr>
<td>MSKO</td>
<td></td>
</tr>
</tbody>
</table>
19.7 Description of Operations when Output Data of the Serial I/O with 1-Byte Buffer is Used

This section describes the operations when output data of the serial I/O with 1-byte buffer is output as a modem signal.

**Operations When Output Data of the Serial I/O with 1-byte Buffer is Used**

To output data of the serial I/O with 1-byte buffer as a modem signal, the shift clock generated by the serial I/O clock generator in the modem signal output circuit is supplied as the shift clock of the serial I/O with 1-byte buffer. By writing output data of the serial I/O with 1-byte buffer automatically to the modem data write bit (MODA: MDA0), the data can be output efficiently as a modem signal.

Figure 19.7-1 “Setting of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer is Used)” shows the setting required to use output data of the serial I/O with 1-byte buffer.

**Figure 19.7-1 Setting of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer is Used)**

Set the external shift clock mode for the serial I/O with 1-byte buffer after enabling (MODC: SIOE=1) the use of output data of the serial I/O with 1-byte buffer, then set transmitting data and enable the transfer.

If the modem signal output is enabled (MODC: STAT=1) after the above operation, transmitting data is written bit by bit to the modem data write bit (MODA: MDA0) synchronized with the shift clock by supplying the shift clock from the modem signal output circuit to the serial I/O with 1-byte buffer so that the data can be output continuously as a modem signal.

The end of transmission can be detected from the serial I/O with 1-byte buffer. Write "0" to both the SIOE bit and STAT bit of the MODC register when the transmission is completed.

The supply of the shift clock from the modem signal output circuit will be stopped, terminating the transmission. If the Hi-z output enable bit (MODC: HIZE) is "0" at this time, the modem output (MSKO) pin is set in high impedance state. If the bit is "1", the current state is retained.

Figure 19.7-2 "Transmission Flow of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer)” and Figure 19.7-3 “Operation of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer)” show the transmission flow and operation when the modem signal output circuit is used alone.
Figure 19.7-2 Transmission Flow of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer)

- Set "1" to SIOE
- Set data to BSIO to start data transfer (SEN=1)
- If "1" is set to STAT, SCK output starts (H → L)
  - SCK latches SO to MDA0 and reset DEMP
  - SCK loads SO to MDA1 and output MSKO
- Do Baud rate counters match?
  - YES
  - Do cycle counters match?
    - YES
    - DEMP flag = 1?
      - YES
      - SIOE=0?
        - YES
        - Transfer end, BSK1 "H" SIOE=STAT=0
        - NO
        - NO
        - Baud rate counter + 1
      - NO
      - NO
      - Baud rate counter + 1
    - NO
    - Baud rate counter + 1
  - NO
  - Baud rate counter + 1
Figure 19.7-3  Operation of the Modem Signal Output Circuit (when Output Data of the Serial I/O with 1-byte Buffer)
19.8 Precautions When Using the Modem Signal Output Circuit

This section describes precautions to be taken when using the modem signal output circuit.

- **Precautions When Using the Modem Signal Output Circuit**

  - **Precautions when using the data empty interrupt**
    
    Since data empty (state in which there is no 1-bit data that has not been transferred) is an interrupt source, an interrupt request (IRQ2) is immediately generated if the data empty interrupt request is enabled (MODC: DEIE=1).

    To send data using the interrupt, enable the data empty interrupt request after transmitting data is prepared. Also, be sure to disable (DEIE=0) the data empty interrupt request when sending the last bit of the transmitting data.

  - **Precautions when making settings in a program**
    
    - Write data to the modem output control register (MODC) when the modem signal output stops (MODC: STAT=0).
    - When enabling (MODC: STAT=1) the modem signal output, do not change other bits of the MODC register.

  - **When using output data of the serial I/O with 1-byte buffer**
    
    - Do not enable (MODC: STAT=1) the modem signal output before transmitting data is prepared in the serial I/O with 1-byte buffer.
    - The end of transmission cannot be determined from the modem signal output circuit. Determine the end of transmission from the serial I/O with 1-byte buffer and then stop the modem signal output (MODC: STAT=0 and MODC: SIOE=0).
19.9 Program Example of the Modem Signal

The following describes a program example of the modem signal output circuit.

- **Program Example When the Modem Signal Output Circuit Is Used Alone**

  - **Processing specifications**
    - 1-byte data is to be sent using the data empty interrupt.
    - By writing data bit by bit to the modem output data register (MODA) eight times clockwise (shifting to the lower-order), data is output from the modem signal output pin from the highest order bit to the lowest order bit.
Program example

DDR6 EQU 0013H ; Address of the port direction register
MODC EQU 002AH ; Address of the modem output control register
MODA EQU 002BH ; Address of the modem output data register
ILR1 EQU 007CH ; Address of the interrupt level setting register
DEIE EQU MODC:6 ; Definition of the data empty interrupt request enable bit
STAT EQU MODC:0 ; Definition of the modem output start bit
DEMP EQU MODA:7 ; Definition of the data empty flag bit
MDA0 EQU MODA:0 ; Definition of the modem data write bit

DSEG
SDATA RB 1 ; Transmitting data
CNT RW 1 ; Number counter

ENDS

INT_V DSEG ABS ; Interrupt vector setting and absolute addressing
IRQ2 DW WARI ; Interrupt vector setting

;-----------Main program-----------------------------------------------------------------------------------------------
CSEG
; [CODE SEGMENT]
; Assuming that the stack pointer (SP) and others have been initialized.
CLS_I ; Disable the interrupt
MOV ILR1,#11101111B ; Set the interrupt level to 2
MOV DDR6,#11110111B ; Set bit3 for input
MOV MODC,#00010100B ; Hi-z enabled, interrupt disabled, no BSIO data use
MOV MODC,#00010100B ; Modem output pin, 1200 bps/8 MHz, Modem output stopped
SET1 ; Enable the CPU interrupt
MOV A,#83H ; Set transmitting data
MOV SDATA,A
MOV A,#08H ; Down-counter for 8-time transmission
MOV CNT,A ; Send 8 bits
SETB STAT ; Declare the transmission start
SETB DEIE ; Enable the buffer empty interrupt request output
; ; ;
; User processing
; ; ;
---Interrupt program---

WARI
PUSHW A
SCHW A,T
PUSHW A

MOV A,SDATA ; Fetch transmitting data from memory
MOV MODA,A ; Write the transmitting data directly to the register as modem data
; Actually, only the lowest-order is sent
RORC A ; Prepare the next transmission by rotating clockwise by one bit
MOV SDATA,A ; Return the result to memory

MOVW A,CNT ; Fetch CNT and decrement it by 1
DECW A
MOVW CNT,A
CMP A,0 ; Determine the end of transmission
BNZ WARI_RET

SND_END ; [End of transmission/8th transmission]
CLRB DEIE ; Disable the data empty interrupt

WARI_RET
POPW A
XCHW A,T
POPW A
RETI
ENDS

END
APPENDIX

The appendixes provide the I/O map, instruction lists, and other information.

APPENDIX A  "I/O Maps"
APPENDIX B  "Overview of Instructions"
APPENDIX C  "Mask Options"
APPENDIX D  "Write Operation of PROM"
APPENDIX E  "Pin Status of the MB89890 Series"
Addresses listed in Table A are allocated to each register of the peripheral functions contained in the MB89890 series.

### I/O Maps

#### Table A-1 I/O Map

<table>
<thead>
<tr>
<th>Address</th>
<th>Register code</th>
<th>Register name</th>
<th>Write/Read</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00_H</td>
<td>PDR0</td>
<td>Port 0 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>01_H</td>
<td>DDR0</td>
<td>Port 0 direction register</td>
<td>W</td>
<td>000000000_B</td>
</tr>
<tr>
<td>02_H</td>
<td>PDR1</td>
<td>Port 1 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>03_H</td>
<td>DDR1</td>
<td>Port 1 direction register</td>
<td>W</td>
<td>000000000_B</td>
</tr>
<tr>
<td>04_H</td>
<td>PDR2</td>
<td>Port 2 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>05_H</td>
<td></td>
<td>(Free space)</td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>06_H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07_H</td>
<td>SYCC</td>
<td>System clock control register</td>
<td>R/W</td>
<td>X--MM100_B</td>
</tr>
<tr>
<td>08_H</td>
<td>STBC</td>
<td>Standby control register</td>
<td>R/W</td>
<td>00010XXX_B</td>
</tr>
<tr>
<td>09_H</td>
<td>WDTC</td>
<td>Watchdog control register</td>
<td>R/W</td>
<td>0XXXXXXX_B</td>
</tr>
<tr>
<td>0A_H</td>
<td>TBTC</td>
<td>Timebase timer control register</td>
<td>R/W</td>
<td>00XXX000_B</td>
</tr>
<tr>
<td>0B_H</td>
<td>WPCR</td>
<td>Clock prescaler control register</td>
<td>R/W</td>
<td>00XXX000_B</td>
</tr>
<tr>
<td>0C_H</td>
<td>PDR3</td>
<td>Port 3 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>0D_H</td>
<td>DDR3</td>
<td>Port 3 direction register</td>
<td>R/W</td>
<td>000000000_B</td>
</tr>
<tr>
<td>0E_H</td>
<td>PDR4</td>
<td>Port 4 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>0F_H</td>
<td>BZCR</td>
<td>Buzzer register</td>
<td>R/W</td>
<td>XXXXXX000_B</td>
</tr>
<tr>
<td>10_H</td>
<td>PDR5</td>
<td>Port 5 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>11_H</td>
<td></td>
<td>(Free space)</td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>12_H</td>
<td>PDR6</td>
<td>Port 6 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>13_H</td>
<td>DDR6</td>
<td>Port 6 direction register</td>
<td>R/W</td>
<td>000000000_B</td>
</tr>
<tr>
<td>14_H</td>
<td>PDR7</td>
<td>Port 7 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>15_H</td>
<td></td>
<td>(Free space)</td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>16_H</td>
<td>PDR8</td>
<td>Port 8 data register</td>
<td>R/W</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>17_H</td>
<td></td>
<td>(Free space)</td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>Address</td>
<td>Register code</td>
<td>Register name</td>
<td>Write/Read</td>
<td>Initial value</td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>------------------------------------</td>
<td>------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>18H</td>
<td>PDR9</td>
<td>Port 9 data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>19H</td>
<td>DDR9</td>
<td>Port 9 direction register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>1AH</td>
<td>PDRA</td>
<td>Port A data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>1BH</td>
<td>DDRA</td>
<td>Port A direction register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>1CH</td>
<td>SMR</td>
<td>Serial mode register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>1DH</td>
<td>SDR</td>
<td>Serial data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>1EH</td>
<td>CNTR</td>
<td>PWM control register</td>
<td>R/W</td>
<td>0X000000_B</td>
</tr>
<tr>
<td>1FH</td>
<td>COMR</td>
<td>PWM compare register</td>
<td>W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>20H</td>
<td>DTMC</td>
<td>DTMF control register</td>
<td>R/W</td>
<td>XXXX0000_B</td>
</tr>
<tr>
<td>21H</td>
<td>DTMD</td>
<td>DTMF data register</td>
<td>R/W</td>
<td>0XXX0000_B</td>
</tr>
<tr>
<td>22H</td>
<td>SBMR</td>
<td>Serial mode register with 1-byte buffer</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>23H</td>
<td>SBFR</td>
<td>Serial flag register with 1-byte buffer</td>
<td>R/W</td>
<td>XXXX010_B</td>
</tr>
<tr>
<td>24H</td>
<td>SBUFW</td>
<td>Serial buffer write register</td>
<td>W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td></td>
<td>SBUFR</td>
<td>Serial buffer read register</td>
<td>R</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>25H</td>
<td>SBDR</td>
<td>Serial data register with 1-byte buffer</td>
<td>R</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>26H</td>
<td>T2CR</td>
<td>Timer 2 control register</td>
<td>R/W</td>
<td>X00000X0_B</td>
</tr>
<tr>
<td>27H</td>
<td>T1CR</td>
<td>Timer 1 control register</td>
<td>R/W</td>
<td>X00000X0_B</td>
</tr>
<tr>
<td>28H</td>
<td>T2DR</td>
<td>Timer 2 data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>29H</td>
<td>T1DR</td>
<td>Timer 1 data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>2AH</td>
<td>MODC</td>
<td>Modem output control register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>2BH</td>
<td>MODA</td>
<td>Modem output data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>2CH</td>
<td></td>
<td>(Free space)</td>
<td></td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>2DH</td>
<td>ADC1</td>
<td>A/D control register 1</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>2EH</td>
<td>ADC2</td>
<td>A/D control register 2</td>
<td>R/W</td>
<td>XXXX0001_B</td>
</tr>
<tr>
<td>2FH</td>
<td>ADCD</td>
<td>A/D data register</td>
<td>R/W</td>
<td>XXXXXXXX_B</td>
</tr>
<tr>
<td>30H</td>
<td>EIE1</td>
<td>External interrupt 1 control register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>31H</td>
<td>EIF1</td>
<td>External interrupt 1 flag register</td>
<td>R/W</td>
<td>XXXX0000_B</td>
</tr>
<tr>
<td>32H</td>
<td>EIE2</td>
<td>External interrupt 2 control register</td>
<td>R/W</td>
<td>00000000_B</td>
</tr>
<tr>
<td>33H</td>
<td>EIF2</td>
<td>External interrupt 2 flag register</td>
<td>R/W</td>
<td>XXXXXXX00_B</td>
</tr>
<tr>
<td>34H</td>
<td>MDC1</td>
<td>Modem timer control register 1</td>
<td>R/W</td>
<td>XX0XX0X_B</td>
</tr>
<tr>
<td>35H</td>
<td>MDC2</td>
<td>Modem timer control register 2</td>
<td>R/W</td>
<td>0XXX0000_B</td>
</tr>
<tr>
<td>36H</td>
<td>MLDH</td>
<td>Modem timer H level data register</td>
<td>R</td>
<td>XXXXXXXX_B</td>
</tr>
</tbody>
</table>
### Table A-1 I/O Map (Continued)

<table>
<thead>
<tr>
<th>Address</th>
<th>Register code</th>
<th>Register name</th>
<th>Write/Read</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>37H</td>
<td>MLDL</td>
<td>Modem timer L level data register</td>
<td>R</td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>38H to 3CH</td>
<td>(Free space)</td>
<td></td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>3DH</td>
<td>SSEL</td>
<td>Serial I/O port switching register</td>
<td>R/W</td>
<td>XX000000_B</td>
</tr>
<tr>
<td>3EH to 7BH</td>
<td>(Free space)</td>
<td></td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
<tr>
<td>7CH</td>
<td>ILR1</td>
<td>Interrupt level register 1</td>
<td>W</td>
<td>11111111_B</td>
</tr>
<tr>
<td>7DH</td>
<td>ILR2</td>
<td>Interrupt level register 2</td>
<td>W</td>
<td>11111111_B</td>
</tr>
<tr>
<td>7EH</td>
<td>ILR3</td>
<td>Interrupt level register 3</td>
<td>W</td>
<td>11111111_B</td>
</tr>
<tr>
<td>7FH</td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXX_B</td>
</tr>
</tbody>
</table>

- **Abbreviations for the write/read operations**
  - R/W: Read/Write enabled
  - R: Read-only
  - W: Write-only

- **Abbreviations for the initial values**
  - 0: The initial value of this bit is "0".
  - 1: The initial value of this bit is "1".
  - X: The initial value of this bit is not defined.
  - M: The initial value of this bit is determined by the mask option.

**Note:**
Do not use the free space.
Appendix B describes the instructions used by the F^2MC-8L.

B.1 "Overview of F^2MC-8L instructions"
B.2 "Addressing"
B.3 "Special Instructions"
B.4 "Bit Manipulation Instructions (SETB, CLRb)"
B.5 "F^2MC-8L Instructions"
B.6 "Instruction Map"
APPENDIX B  Overview of Instructions

B.1  Overview of $F^2MC$-8L instructions

The $F^2MC$-8L supports 140 types of instructions.

- **Overview of $F^2MC$-8L instructions**
  
  The $F^2MC$-8L has 140 1-byte machine instructions (256-byte instruction map). An instruction code consists of an instruction and zero or more operands that follow.

  Figure B.1-1 "Relationship between the instruction codes and the instruction map" shows the relationship between the instruction codes and the instruction map.

  **Figure B.1-1  Relationship between the instruction codes and the instruction map**

- The instructions are classified into four types: transfer, arithmetic, branch, and other.
- A variety of addressing methods is available. One of ten addressing modes can be selected depending on the selected instruction and specified operand(s).
- Bit manipulation instructions are provided. They can be used for read-modify-write operations.
- Some instructions are used for special operations.
Symbols used with instructions

Table B.1-1 "Symbols in the instruction list" lists the symbols used in the instruction code descriptions in Appendix B.

Table B.1-1 Symbols in the instruction list

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>dir</td>
<td>Direct address (8 bits)</td>
</tr>
<tr>
<td>off</td>
<td>Offset (8 bits)</td>
</tr>
<tr>
<td>ext</td>
<td>Extended address (16 bits)</td>
</tr>
<tr>
<td>#vct</td>
<td>Vector table number (3 bits)</td>
</tr>
<tr>
<td>#d8</td>
<td>Immediate data (8 bits)</td>
</tr>
<tr>
<td>#d16</td>
<td>Immediate data (16 bits)</td>
</tr>
<tr>
<td>dir:16</td>
<td>Bit direct address (8 bits:3 bits)</td>
</tr>
<tr>
<td>rel</td>
<td>Branch relative address (8 bits)</td>
</tr>
<tr>
<td>@</td>
<td>Register indirect addressing (examples: @A, @IX, @EP)</td>
</tr>
<tr>
<td>A</td>
<td>Accumulator (8 or 16 bits, which are determined depending on the instruction being used)</td>
</tr>
<tr>
<td>AH</td>
<td>Higher 8 bits of the accumulator (8 bits)</td>
</tr>
<tr>
<td>AL</td>
<td>Lower 8 bits of the accumulator (8 bits)</td>
</tr>
<tr>
<td>T</td>
<td>Temporary accumulator (8 or 16 bits, which are determined depending on the instruction being used)</td>
</tr>
<tr>
<td>TH</td>
<td>Higher 8 bits of the temporary accumulator (8 bits)</td>
</tr>
<tr>
<td>TL</td>
<td>Lower 8 bits of the temporary accumulator (8 bits)</td>
</tr>
<tr>
<td>IX</td>
<td>Index register (16 bits)</td>
</tr>
<tr>
<td>EP</td>
<td>Extra pointer (16 bits)</td>
</tr>
<tr>
<td>PC</td>
<td>Program counter (16 bits)</td>
</tr>
<tr>
<td>SP</td>
<td>Stack pointer (16 bits)</td>
</tr>
<tr>
<td>PS</td>
<td>Program status (16 bits)</td>
</tr>
<tr>
<td>dr</td>
<td>Either accumulator or index register (16 bits)</td>
</tr>
<tr>
<td>CCR</td>
<td>Condition code register (8 bits)</td>
</tr>
<tr>
<td>RP</td>
<td>Register bank pointer (5 bits)</td>
</tr>
<tr>
<td>Ri</td>
<td>General-purpose register (8 bits, i = 0 to 7)</td>
</tr>
<tr>
<td>X</td>
<td>X is immediate data (8 or 16 bits, which are determined depending on the instruction being used).</td>
</tr>
<tr>
<td>(X)</td>
<td>The content of X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).</td>
</tr>
<tr>
<td>((X))</td>
<td>The address indicated by the X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).</td>
</tr>
</tbody>
</table>
### APPENDIX B  Overview of Instructions

#### Items in the instruction list

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MNEMONIC</strong></td>
<td>This column shows the instruction in assembly language.</td>
</tr>
<tr>
<td>~</td>
<td>This column shows the number of cycles required by the instruction (instruction cycle count).</td>
</tr>
<tr>
<td>#</td>
<td>This column shows the number of bytes for the instruction.</td>
</tr>
<tr>
<td>Operation</td>
<td>This column shows the operation performed by the instruction.</td>
</tr>
</tbody>
</table>
| TL, TH, AH | These columns indicate a change in the contents of TL, TH, and AH (automatic transfer from A to T) upon the execution of the instruction. The meanings of symbols in each column are as follows:  
• "-" indicates that no change is made.  
• "dH" indicates the higher 8 bits of data in the operation column.  
• "AL" and "AH" indicate that the contents of AL and AH immediately before the execution of the instruction are set.  
• "00" indicates that 00 is set. |
| N, Z, V, C | These columns indicate whether their respective flags are changed upon the execution of the instruction. A plus (+) sign indicates that the instruction changes the corresponding flag. |
| **OP CODE** | This column shows the operation code(s) of the instruction. When the instruction uses two or more operation codes, the following notation is used:  
[Example] 48 to 4F: This represents from 48 to 4F. |
B.2 Addressing

The F\textsuperscript{2}MC-8L has the following ten addressing modes:
- Direct addressing
- Extended addressing
- Bit direct addressing
- Index addressing
- Pointer addressing
- General-purpose register addressing
- Immediate addressing
- Vector addressing
- Relative addressing
- Inherent addressing

Explanation of addressing

- **Direct addressing**
  
  Direct addressing is indicated by dir in the instruction list. This addressing is used to access the area between 0000\text{H} and 00FF\text{H}. In this addressing mode, the higher byte of the address is 00\text{H} and the lower byte is specified by the operand. Figure B.2-1 "Example of direct addressing" shows an example.

  **Figure B.2-1 Example of direct addressing**
  
  MOV 12H, A
  
  \[\begin{array}{c}
  00 \ 12H \\
  45H \\
  \end{array}\]

  \[A \ 45H\]

- **Extended addressing**
  
  Extended addressing is indicated by ext in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the first operand specifies the higher byte of the address, and the second operand specifies the lower byte. Figure B.2-2 "Example of extended addressing" shows an example.

  **Figure B.2-2 Example of extended addressing**
  
  MOVW A, 1234H
  
  \[\begin{array}{c}
  12 \ 3 4H \\
  12 \ 3 5H \\
  56H \\
  78H \\
  \end{array}\]

  \[A \ 5678H\]
APPENDIX B  Overview of Instructions

○ Bit direct addressing

Bit direct addressing is indicated by dir:b in the instruction list. This addressing is used to access a particular bit in the area between 0000H and 00FFH. In this addressing mode, the higher byte of the address is 00H and the lower byte is specified by the operand. The bit position at the address is specified by the lower three bits of the operation code. Figure B.2-3 "Example of bit direct addressing" shows an example.

![Figure B.2-3 Example of bit direct addressing](image)

○ Index addressing

Index addressing is indicated by @IX+off in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is the value resulting from sign-extending the contents of the first operand and adding them to IX (index register). Figure B.2-4 "Example of index addressing" shows an example.

![Figure B.2-4 Example of index addressing](image)

○ Pointer addressing

Pointer addressing is indicated by @EP in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is contained in EP (extra pointer). Figure B.2-5 "Example of pointer addressing" shows an example.

![Figure B.2-5 Example of pointer addressing](image)

○ General-purpose register addressing

General-purpose register addressing is indicated by Ri in the instruction list. This addressing is used to access a register bank in the general-purpose register area. In this addressing mode, the higher byte of the address is always 01 and the lower byte is specified based on the contents of RP (register bank pointer) and the lower three bits of the operation code. Figure B.2-6 "Example of general-purpose register addressing" shows an example.

![Figure B.2-6 Example of general-purpose register addressing](image)
**Immediate addressing**

Immediate addressing is indicated by #d8 in the instruction list. This addressing is used when immediate data is required. In this addressing mode, the operand is used as immediate data. Whether the data is specified in bytes or words is determined by the operation code. Figure B.2-7 "Example of immediate addressing" shows an example.

![Figure B.2-7 Example of immediate addressing](image)

**Vector addressing**

Vector addressing is indicated by vct in the instruction list. This addressing is used to branch to a subroutine address stored in the vector table. In this addressing mode, vct information is contained in the operation codes, and the corresponding table addresses are created as shown in Table B.2-1 "Vector table addresses corresponding to vct".

<table>
<thead>
<tr>
<th>#vct</th>
<th>Vector table address (higher address:lower address of branch destination)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FFC0_H : FFC1_H</td>
</tr>
<tr>
<td>1</td>
<td>FFC2_H : FFC3_H</td>
</tr>
<tr>
<td>2</td>
<td>FFC4_H : FFC5_H</td>
</tr>
<tr>
<td>3</td>
<td>FFC6_H : FFC7_H</td>
</tr>
<tr>
<td>4</td>
<td>FFC8_H : FFC9_H</td>
</tr>
<tr>
<td>5</td>
<td>FFCA_H : FFCB_H</td>
</tr>
<tr>
<td>6</td>
<td>FFCC_H : FFCD_H</td>
</tr>
<tr>
<td>7</td>
<td>FFCE_H : FFCF_H</td>
</tr>
</tbody>
</table>

Figure B.2-8 "Example of vector addressing" shows an example.

![Figure B.2-8 Example of vector addressing](image)
Relative addressing

Relative addressing is indicated by rel in the instruction list. This addressing is used to branch to within the area between the address 128 bytes higher and that 128 bytes lower relative to the address contained in the PC (program counter). In this addressing mode, the result of a signed addition of the contents of the operand to the PC is stored in the PC. Figure B.2-9 "Example of relative addressing" shows an example.

![Figure B.2-9 Example of relative addressing](image)

In this example, a branch to the address of the BNE operation code occurs, thus resulting in an infinite loop.

Inherent addressing

Inherent addressing is indicated as the addressing without operands in the instruction list. This addressing is used to perform the operation determined by the operation code. In this addressing mode, different operations are performed via different instructions. Figure B.2-10 "Example of inherent addressing" shows an example.

![Figure B.2-10 Example of inherent addressing](image)
B.3 Special Instructions

This section describes the special instructions used for other than addressing.

- Special instructions

  - **JMP @A**
    
    This instruction sets the contents of A (accumulator) to PC (program counter) as the address, and causes a branch to that address. One of the N branch destination addresses is selected from a table, and then transferred to A. The instruction can be executed to perform N-branch processing.
    
    Figure B.3-1 "JMP @A" shows a summary of the instruction.

    ![Figure B.3-1 JMP @A]

    After the MOVW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of MOVW A, PC. Accordingly, Figure B.3-2 "MOVW A, PC" shows that A contains 1234H, which is the address of the operation code of the instruction that follows MOVW A, PC.

  - **MOVW A, PC**
    
    This instruction performs the operation which is the reverse of that performed by JMP @A. That is, the instruction stores the contents of PC in A. When the instruction is executed in the main routine, so that a specific subroutine is called, whether A contains a predetermined value can be checked by the subroutine. This can be used to determine that the branch source is not any unexpected section of the program and to check for program runaway.

    Figure B.3-2 "MOVW A, PC" shows a summary of the instruction.
APPENDIX B  Overview of Instructions

○ MULU A

This instruction performs an unsigned multiplication of AL (lower eight bits of the accumulator) and TL (lower eight bits of the temporary accumulator), and stores the 16-bit result in A. The contents of T (temporary accumulator) do not change. The contents of AH (higher eight bits of the accumulator) and TH (higher eight bits of the temporary accumulator) before execution of the instruction are not used for the operation. The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a multiplication.

Figure B.3-3 "MULU" shows a summary of the instruction.

![Figure B.3-3 MULU](image)

○ DIVU A

This instruction divides the 16-bit value in T by the unsigned 8-bit value in AL, and stores the 8-bit result and the 8-bit remainder in AL and TL, respectively. A value of 0 is set to both AH and TH. The contents of AH before execution of the instruction are not used for the operation. An unpredictable result is produced from data that results in more than eight bits. In addition, there is no indication of the result having more than eight bits. Therefore, if it is likely that data will cause a result of more than eight bits, the data must be checked to ensure that the result will not have more than eight bits before it is used.

The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a division.

Figure B.3-4 "DIVU A" shows a summary of the instruction.

![Figure B.3-4 DIVU A](image)

○ XCHW A, PC

This instruction swaps the contents of A and PC, resulting in a branch to the address contained in A before execution of the instruction. After the instruction is executed, A contains the address that follows the address of the operation code of MOVW A, PC. This instruction is effective especially when it is used in the main routine to specify a table for use in a subroutine.

Figure B.3-5 “XCHW A, PC” shows a summary of the instruction.

![Figure B.3-5 XCHW A, PC](image)
After the XCHW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of XCHW A, PC. Accordingly, Figure B.3-5 "XCHW A, PC" shows that A contains 1235\text{H}, which is the address of the operation code of the instruction that follows XCHW A, PC. This is why 1235\text{H} is stored instead of 1234\text{H}.

Figure B.3-6 "Example of using XCHW A, PC" shows an assembly language example.

![Figure B.3-6 Example of using XCHW A, PC](image)

\textbf{CALLV \#vct}

This instruction is used to branch to a subroutine address stored in the vector table. The instruction saves the return address (contents of PC) in the location at the address contained in SP (stack pointer), and uses vector addressing to cause a branch to the address stored in the vector table. Because CALLV \#vct is a 1-byte instruction, the use of this instruction for frequently used subroutines can reduce the entire program size.

Figure B.3-7 "Example of executing CALLV \#3" shows a summary of the instruction.

![Figure B.3-7 Example of executing CALLV \#3](image)

After the CALLV \#vct instruction is executed, the contents of PC saved on the stack area are the address of the operation code of the next instruction, rather than the address of the operation code of CALLV \#vct. Accordingly, Figure B.3-7 "Example of executing CALLV \#3" shows that the value saved in the stack (1232\text{H} and 1233\text{H}) is 5679\text{H}, which is the address of the operation code of the instruction that follows CALLV \#vct (return address).
B.4 Bit Manipulation Instructions (SETB, CLRB)

Some bits of peripheral function registers include bits that are read by a bit manipulation instruction differently than usual.

Read-modify-write operation

By using these bit manipulation instructions, only the specified bit in a register or RAM location can be set to 1 (SETB) or cleared to 0 (CLRB). However, as the CPU operates on data in 8-bit units, the actual operation (read-modify-write operation) involves a sequence of steps: 8-bit data is read, the specified bit is changed, and the data is written back to the location at the original address.

Table B.4-1 “Bus operation for bit manipulation instructions” shows bus operation for bit manipulation instructions.

Table B.4-1 Bus operation for bit manipulation instructions

<table>
<thead>
<tr>
<th>CODE</th>
<th>MNEMONIC</th>
<th>TO</th>
<th>Cycle</th>
<th>Address bus</th>
<th>Data bus</th>
<th>RD</th>
<th>WR</th>
<th>RMW</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0 to A7</td>
<td>CLR B dir:b</td>
<td>4</td>
<td>1</td>
<td>N+1</td>
<td>Dir</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>A8 to AF</td>
<td>SET B dir:b</td>
<td>2</td>
<td>dir address</td>
<td>Data</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>dir address</td>
<td>Data</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>N+2</td>
<td>Next instruction</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Read operation upon the execution of bit manipulation instructions

For some I/O ports and for the interrupt request flag bits, the value to be read differs between a normal read operation and a read-modify-write operation.

- **I/O ports (during a bit manipulation)**
  
  From some I/O ports, an I/O pin value is read during a normal read operation, while an output latch value is read during a bit manipulation. This prevents the other output latch bits from being changed accidentally, regardless of the I/O directions and states of the pins.

- **Interrupt request flag bits (during a bit manipulation)**
  
  An interrupt request flag bit functions as a flag bit indicating whether an interrupt request exists during a normal read operation. However, 1 is always read from this bit during a bit manipulation. This prevents the flag from being cleared accidentally by a value of 0 which would otherwise be written to the interrupt request flag bit when another bit is manipulated.
B.5 F²MC-8L Instructions

Table B.5-1 "Transfer Instructions" to Table B.5-4 "Other instructions" list the instructions used with the F²MC-8L.

- Transfer instructions

Table B.5-1 Transfer Instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MOV dir, A</td>
<td>3</td>
<td>2</td>
<td>(dir)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>2</td>
<td>MOV @IX+off, A</td>
<td>4</td>
<td>2</td>
<td>((IX)+off)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>MOV ext, A</td>
<td>4</td>
<td>3</td>
<td>(ext)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>61</td>
</tr>
<tr>
<td>4</td>
<td>MOV @EP, A</td>
<td>3</td>
<td>1</td>
<td>((EP))&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>MOV Ri, A</td>
<td>3</td>
<td>1</td>
<td>(Ri)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>48 to 4F</td>
</tr>
<tr>
<td>6</td>
<td>MOV A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;--d8</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>04</td>
</tr>
<tr>
<td>7</td>
<td>MOV A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;--(dir)</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>05</td>
</tr>
<tr>
<td>8</td>
<td>MOV A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;--((IX)+off)</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>06</td>
</tr>
<tr>
<td>9</td>
<td>MOV A, ext</td>
<td>4</td>
<td>3</td>
<td>(A)&lt;--(ext)</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>60</td>
</tr>
<tr>
<td>10</td>
<td>MOV A, @A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;--((A))</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>92</td>
</tr>
<tr>
<td>11</td>
<td>MOV A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;--((EP))</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>07</td>
</tr>
<tr>
<td>12</td>
<td>MOV A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;--(Ri)</td>
<td>AL</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>08 to 0F</td>
</tr>
<tr>
<td>13</td>
<td>MOV dir, #d8</td>
<td>4</td>
<td>3</td>
<td>(dir)&lt;--d8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>85</td>
</tr>
<tr>
<td>14</td>
<td>MOV @IX+off, #d8</td>
<td>5</td>
<td>3</td>
<td>((IX)+off)&lt;--d8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>86</td>
</tr>
<tr>
<td>15</td>
<td>MOV @EP, #d8</td>
<td>4</td>
<td>2</td>
<td>((EP))&lt;--d8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>87</td>
</tr>
<tr>
<td>16</td>
<td>MOV Ri, #d8</td>
<td>4</td>
<td>2</td>
<td>(Ri)&lt;--d8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>88 to 8F</td>
</tr>
<tr>
<td>17</td>
<td>MOVW dir, A</td>
<td>4</td>
<td>2</td>
<td>(dir)&lt;--(AH), (dir+1)&lt;--(AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D5</td>
</tr>
<tr>
<td>18</td>
<td>MOVW @IX+off, A</td>
<td>5</td>
<td>2</td>
<td>((IX)+off)&lt;--(AH), ((IX)+off+1)&lt;--(AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D6</td>
</tr>
<tr>
<td>19</td>
<td>MOVW ext, A</td>
<td>5</td>
<td>3</td>
<td>(ext)&lt;--(AH), (ext+1)&lt;--(AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D4</td>
</tr>
<tr>
<td>20</td>
<td>MOVW @EP, A</td>
<td>4</td>
<td>1</td>
<td>((EP))&lt;--(AH), ((EP)+1)&lt;--(AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D7</td>
</tr>
<tr>
<td>21</td>
<td>MOVW EP, A</td>
<td>2</td>
<td>1</td>
<td>(EP)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E3</td>
</tr>
<tr>
<td>22</td>
<td>MOVW A, #d16</td>
<td>3</td>
<td>3</td>
<td>(A)&lt;--d16</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>E4</td>
</tr>
<tr>
<td>23</td>
<td>MOVW A, dir</td>
<td>4</td>
<td>2</td>
<td>(AH)&lt;--(dir), (AL)&lt;--(dir+1)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>C5</td>
</tr>
</tbody>
</table>
APPENDIX B  Overview of Instructions

### Table B.5-1  Transfer Instructions (Continued)

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>MOVW A, @IX+offset</td>
<td>5</td>
<td>2</td>
<td>(AH)&lt;--((IX)+off), (AL)&lt;--((IX)+off+1)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>C6</td>
</tr>
<tr>
<td>25</td>
<td>MOVW A, ext</td>
<td>5</td>
<td>3</td>
<td>(AH)&lt;--(ext), (AL)&lt;--(ext+1)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>C4</td>
</tr>
<tr>
<td>26</td>
<td>MOVW A, @A</td>
<td>4</td>
<td>1</td>
<td>(AH)&lt;--((A)), (AL)&lt;--((A)+1)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>93</td>
</tr>
<tr>
<td>27</td>
<td>MOVW A, @EP</td>
<td>4</td>
<td>1</td>
<td>(AH)&lt;--((EP)), (AL)&lt;--((EP)+1)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>C7</td>
</tr>
<tr>
<td>28</td>
<td>MOVW A, EP</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;--(EP)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F3</td>
</tr>
<tr>
<td>29</td>
<td>MOVW EP, #d16</td>
<td>3</td>
<td>3</td>
<td>(EP)&lt;--#d16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E7</td>
</tr>
<tr>
<td>30</td>
<td>MOVW IX, A</td>
<td>2</td>
<td>1</td>
<td>(IX)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E2</td>
</tr>
<tr>
<td>31</td>
<td>MOVW A, IX</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;--(IX)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F2</td>
</tr>
<tr>
<td>32</td>
<td>MOVW SP, A</td>
<td>2</td>
<td>1</td>
<td>(SP)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E1</td>
</tr>
<tr>
<td>33</td>
<td>MOVW A, SP</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;--(SP)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F1</td>
</tr>
<tr>
<td>34</td>
<td>MOV @A, T</td>
<td>3</td>
<td>1</td>
<td>((A))&lt;--(T)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>82</td>
</tr>
<tr>
<td>35</td>
<td>MOVW @A, T</td>
<td>4</td>
<td>1</td>
<td>((A))&lt;--(TH), ((A)+1)&lt;--(TL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>83</td>
</tr>
<tr>
<td>36</td>
<td>MOVW IX, #d16</td>
<td>3</td>
<td>3</td>
<td>(IX)&lt;--#d16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E6</td>
</tr>
<tr>
<td>37</td>
<td>MOVW A, PS</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;--(PS)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>70</td>
</tr>
<tr>
<td>38</td>
<td>MOVW PS, A</td>
<td>3</td>
<td>1</td>
<td>(PS)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>71</td>
</tr>
<tr>
<td>39</td>
<td>MOVW SP, #d16</td>
<td>3</td>
<td>3</td>
<td>(SP)&lt;--#d16</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E5</td>
</tr>
<tr>
<td>40</td>
<td>SWAP</td>
<td>2</td>
<td>1</td>
<td>(AH)&lt;-- --&gt; (AL)</td>
<td>-</td>
<td>-</td>
<td>AL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>10</td>
</tr>
<tr>
<td>41</td>
<td>SETB dir:b</td>
<td>4</td>
<td>2</td>
<td>(dir):b &lt;--1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>A8 to AF</td>
</tr>
<tr>
<td>42</td>
<td>CLRb dir:b</td>
<td>4</td>
<td>2</td>
<td>(dir):b &lt;--0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>A0 to A7</td>
</tr>
<tr>
<td>43</td>
<td>XCH A, T</td>
<td>2</td>
<td>1</td>
<td>(AL)&lt;-- --&gt; (TL)</td>
<td>AL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>42</td>
</tr>
<tr>
<td>44</td>
<td>XCHW A, T</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-- --&gt; (T)</td>
<td>AL</td>
<td>AH</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>43</td>
</tr>
<tr>
<td>45</td>
<td>XCHW A, EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-- --&gt; (EP)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F7</td>
</tr>
<tr>
<td>46</td>
<td>XCHW A, IX</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-- --&gt; (IX)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F6</td>
</tr>
<tr>
<td>47</td>
<td>XCHW A, SP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-- --&gt; (SP)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F5</td>
</tr>
<tr>
<td>48</td>
<td>MOVW A, PC</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;-- (PC)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F0</td>
</tr>
</tbody>
</table>

**Note:**

In automatic transfer to T during byte transfer to A, AL is transferred to TL.

If an instruction has two or more operands, they are assumed to be saved in the order indicated by MNEMONIC.
### Arithmetic instructions

#### Table B.5-2 Arithmetic Operation Instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADDC A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)+(Ri)+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>28 to 2F</td>
</tr>
<tr>
<td>2</td>
<td>ADDC A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;-(A)+d8+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>24</td>
</tr>
<tr>
<td>3</td>
<td>ADDC A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;-(A)+(dir)+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>ADDC A, @IX+offset</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;-(A)+((IX)+offset)+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>26</td>
</tr>
<tr>
<td>5</td>
<td>ADDC A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)+((EP)))+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>27</td>
</tr>
<tr>
<td>6</td>
<td>ADDCW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)+(T)+C</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>23</td>
</tr>
<tr>
<td>7</td>
<td>ADDC A</td>
<td>2</td>
<td>1</td>
<td>(AL)&lt;-(AL)+(TL)+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>22</td>
</tr>
<tr>
<td>8</td>
<td>SUBC A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)-(Ri)-C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>38 to 3F</td>
</tr>
<tr>
<td>9</td>
<td>SUBC A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;-(A)-d8-C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>SUBC A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;-(A)-(dir)-C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>35</td>
</tr>
<tr>
<td>11</td>
<td>SUBC A, @IX+offset</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;-(A)-((IX)+offset)-C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>36</td>
</tr>
<tr>
<td>12</td>
<td>SUBC A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)-((EP))+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>37</td>
</tr>
<tr>
<td>13</td>
<td>SUBCW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(T)-(A)-C</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>33</td>
</tr>
<tr>
<td>14</td>
<td>SUBC A</td>
<td>2</td>
<td>1</td>
<td>(AL)&lt;-(TL)-(AL)+C</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>32</td>
</tr>
<tr>
<td>15</td>
<td>INC Ri</td>
<td>4</td>
<td>1</td>
<td>(Ri)&lt;-(Ri)+1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>C8 to CF</td>
</tr>
<tr>
<td>16</td>
<td>INCW EP</td>
<td>3</td>
<td>1</td>
<td>(EP)&lt;-(EP)+1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>C3</td>
</tr>
<tr>
<td>17</td>
<td>INCW IX</td>
<td>3</td>
<td>1</td>
<td>(IX)&lt;-(IX)+1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>C2</td>
</tr>
<tr>
<td>18</td>
<td>INCW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)+1</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>C0</td>
</tr>
<tr>
<td>19</td>
<td>DEC Ri</td>
<td>4</td>
<td>1</td>
<td>(Ri)&lt;-(Ri)-1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>D8 to DF</td>
</tr>
<tr>
<td>20</td>
<td>DECW EP</td>
<td>3</td>
<td>1</td>
<td>(EP)&lt;-(EP)-1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D3</td>
</tr>
<tr>
<td>21</td>
<td>DECW IX</td>
<td>3</td>
<td>1</td>
<td>(IX)&lt;-(IX)-1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>D2</td>
</tr>
<tr>
<td>22</td>
<td>DECW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A)-1</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>D0</td>
</tr>
<tr>
<td>23</td>
<td>MULU A</td>
<td>19</td>
<td>1</td>
<td>(A)&lt;-(AL)x(TL)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>01</td>
</tr>
<tr>
<td>24</td>
<td>DIVU A</td>
<td>21</td>
<td>1</td>
<td>(A)&lt;-(T)/(AL), MOD --&gt; (T)</td>
<td>dL</td>
<td>00</td>
<td>00</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>11</td>
</tr>
<tr>
<td>25</td>
<td>ANDW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A) ∧ (T)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>63</td>
</tr>
<tr>
<td>26</td>
<td>ORW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A) ∨ (T)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>73</td>
</tr>
<tr>
<td>27</td>
<td>XORW A</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(A) ⊕ (T)</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>53</td>
</tr>
<tr>
<td>28</td>
<td>CMP A</td>
<td>2</td>
<td>1</td>
<td>(TL)-(AL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>12</td>
</tr>
<tr>
<td>29</td>
<td>CMPW A</td>
<td>3</td>
<td>1</td>
<td>(T)-(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>13</td>
</tr>
<tr>
<td>30</td>
<td>RORC A</td>
<td>2</td>
<td>1</td>
<td>C --&gt; A</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>03</td>
</tr>
</tbody>
</table>
### APPENDIX B  Overview of Instructions

Table B.5-2  Arithmetic Operation Instructions  (Continued)

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>ROLC A</td>
<td>2</td>
<td>1</td>
<td>C &lt;-- A</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>02</td>
</tr>
<tr>
<td>32</td>
<td>CMP A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)-d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>14</td>
</tr>
<tr>
<td>33</td>
<td>CMP A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)-(dir)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>15</td>
</tr>
<tr>
<td>34</td>
<td>CMP A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)-((EP))</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>17</td>
</tr>
<tr>
<td>35</td>
<td>CMP A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)-((IX)+off)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>16</td>
</tr>
<tr>
<td>36</td>
<td>CMP A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)-((Ri))</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>18 to 1F</td>
</tr>
<tr>
<td>37</td>
<td>DAA</td>
<td>2</td>
<td>1</td>
<td>decimal adjust for addition</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>84</td>
</tr>
<tr>
<td>38</td>
<td>DAS</td>
<td>2</td>
<td>1</td>
<td>decimal adjust for subtraction</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>94</td>
</tr>
<tr>
<td>39</td>
<td>XOR A</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;-(AL) (TL)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>52</td>
</tr>
<tr>
<td>40</td>
<td>XOR A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;-(AL) d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>54</td>
</tr>
<tr>
<td>41</td>
<td>XOR A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;-(AL) (dir)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>55</td>
</tr>
<tr>
<td>42</td>
<td>XOR A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) ((EP))</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>57</td>
</tr>
<tr>
<td>43</td>
<td>XOR A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;-(AL) ((IX)+off)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>56</td>
</tr>
<tr>
<td>44</td>
<td>XOR A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) (Ri)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>58 to 5F</td>
</tr>
<tr>
<td>45</td>
<td>AND A</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;-(AL) (TL)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>62</td>
</tr>
<tr>
<td>46</td>
<td>AND A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;-(AL) d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>64</td>
</tr>
<tr>
<td>47</td>
<td>AND A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;-(AL) (dir)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>65</td>
</tr>
<tr>
<td>48</td>
<td>AND A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) ((EP))</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>67</td>
</tr>
<tr>
<td>49</td>
<td>AND A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;-(AL) ((IX)+off)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>66</td>
</tr>
<tr>
<td>50</td>
<td>AND A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) (Ri)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>68 to 6F</td>
</tr>
<tr>
<td>51</td>
<td>OR A</td>
<td>2</td>
<td>1</td>
<td>(A)&lt;-(AL) (TL)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>72</td>
</tr>
<tr>
<td>52</td>
<td>OR A, #d8</td>
<td>2</td>
<td>2</td>
<td>(A)&lt;-(AL) d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>74</td>
</tr>
<tr>
<td>53</td>
<td>OR A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)&lt;-(AL) (dir)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>75</td>
</tr>
<tr>
<td>54</td>
<td>OR A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) ((EP))</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>77</td>
</tr>
<tr>
<td>55</td>
<td>OR A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)&lt;-(AL) ((IX)+off)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>76</td>
</tr>
<tr>
<td>56</td>
<td>OR A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)&lt;-(AL) (Ri)</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>R</td>
<td>-</td>
<td>78 to 7F</td>
</tr>
<tr>
<td>57</td>
<td>CMP dir, #d8</td>
<td>5</td>
<td>3</td>
<td>(dir)-d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>95</td>
</tr>
<tr>
<td>58</td>
<td>CMP @EP, #d8</td>
<td>4</td>
<td>2</td>
<td>((EP))-d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>97</td>
</tr>
<tr>
<td>59</td>
<td>CMP @IX+off, #d8</td>
<td>5</td>
<td>3</td>
<td>((IX)+off)-d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>96</td>
</tr>
<tr>
<td>60</td>
<td>CMP Ri, #d8</td>
<td>4</td>
<td>2</td>
<td>(Ri)-d8</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>98 to 9F</td>
</tr>
<tr>
<td>61</td>
<td>INCW SP</td>
<td>3</td>
<td>1</td>
<td>(SP)&lt;-(SP)+1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C1</td>
</tr>
<tr>
<td>62</td>
<td>DECW SP</td>
<td>3</td>
<td>1</td>
<td>(SP)&lt;-(SP)-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D1</td>
</tr>
</tbody>
</table>
### Branch instructions

**Table B.5-3  Branch instructions**

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BZ/BEQ rel</td>
<td>3</td>
<td>2</td>
<td>if Z=1 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FD</td>
</tr>
<tr>
<td>2</td>
<td>BNZ/BNE rel</td>
<td>3</td>
<td>2</td>
<td>if Z=0 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FC</td>
</tr>
<tr>
<td>3</td>
<td>BC/BLO rel</td>
<td>3</td>
<td>2</td>
<td>if C=1 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F9</td>
</tr>
<tr>
<td>4</td>
<td>BNC/BHS rel</td>
<td>3</td>
<td>2</td>
<td>if C=0 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F8</td>
</tr>
<tr>
<td>5</td>
<td>BN rel</td>
<td>3</td>
<td>2</td>
<td>if N=1 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FB</td>
</tr>
<tr>
<td>6</td>
<td>BP rel</td>
<td>3</td>
<td>2</td>
<td>if N=0 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FA</td>
</tr>
<tr>
<td>7</td>
<td>BLT rel</td>
<td>3</td>
<td>2</td>
<td>if V ( \vee ) N=1 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FF</td>
</tr>
<tr>
<td>8</td>
<td>BGE rel</td>
<td>3</td>
<td>2</td>
<td>if V ( \vee ) N=0 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>FE</td>
</tr>
<tr>
<td>9</td>
<td>BBC dir:b, rel</td>
<td>5</td>
<td>3</td>
<td>if (dir:b)=0 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>B0 to B7</td>
</tr>
<tr>
<td>10</td>
<td>BBS dir:b, rel</td>
<td>5</td>
<td>3</td>
<td>if (dir:b)=1 then PC&lt;--PC+rel</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>B8 to BF</td>
</tr>
<tr>
<td>11</td>
<td>JMP @A</td>
<td>2</td>
<td>1</td>
<td>(PC)&lt;--(A)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E0</td>
</tr>
<tr>
<td>12</td>
<td>JMP ext</td>
<td>3</td>
<td>3</td>
<td>(PC)&lt;--ext</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>21</td>
</tr>
<tr>
<td>13</td>
<td>CALLV #vct</td>
<td>6</td>
<td>1</td>
<td>vector call</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E8 to EF</td>
</tr>
<tr>
<td>14</td>
<td>CALL ext</td>
<td>6</td>
<td>3</td>
<td>subroutine call</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>31</td>
</tr>
<tr>
<td>15</td>
<td>XCHW A, PC</td>
<td>3</td>
<td>1</td>
<td>(PC)&lt;--(A), (A)&lt;--(PC)+1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F4</td>
</tr>
<tr>
<td>16</td>
<td>RET</td>
<td>4</td>
<td>1</td>
<td>return from subroutine</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
</tr>
<tr>
<td>17</td>
<td>RETI</td>
<td>6</td>
<td>1</td>
<td>return from interrupt</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>restore 30</td>
</tr>
</tbody>
</table>
Other instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>MNEMONIC</th>
<th>~</th>
<th>#</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>N</th>
<th>Z</th>
<th>V</th>
<th>C</th>
<th>OP CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PUSHW A</td>
<td>4</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>POPW A</td>
<td>4</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>dH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>PUSHW IX</td>
<td>4</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>41</td>
</tr>
<tr>
<td>4</td>
<td>POPW IX</td>
<td>4</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>NOP</td>
<td>1</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>00</td>
</tr>
<tr>
<td>6</td>
<td>CLRC</td>
<td>1</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>R</td>
<td>81</td>
</tr>
<tr>
<td>7</td>
<td>SETC</td>
<td>1</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>91</td>
</tr>
<tr>
<td>8</td>
<td>CLRi</td>
<td>1</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>80</td>
</tr>
<tr>
<td>9</td>
<td>SETI</td>
<td>1</td>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>90</td>
</tr>
</tbody>
</table>
## Table B.6-1: F₂MC-8L Instruction Map

<table>
<thead>
<tr>
<th>H</th>
<th>L</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOP</td>
<td>SWAP</td>
<td>RET</td>
<td>RETI</td>
<td>PUSHW</td>
<td>POPW</td>
<td>MOV</td>
<td>MOVW</td>
<td>MOV</td>
<td>CPI</td>
<td>CLI</td>
<td>SETI</td>
<td>CLR</td>
<td>BBC</td>
<td>INW</td>
<td>DCS</td>
<td>JMP</td>
<td>MOV</td>
</tr>
<tr>
<td>MUL</td>
<td>DIV</td>
<td>JMP</td>
<td>CALL</td>
<td>PUSHW</td>
<td>POPW</td>
<td>MOV</td>
<td>MOVW</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td></td>
</tr>
<tr>
<td>ROLC</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>ROIC</td>
<td>CMP</td>
<td>ADDCW</td>
<td>SUBCW</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</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>MOV</td>
<td>CMP</td>
<td>ADDC</td>
<td>SUBC</td>
<td>XOR</td>
<td>XOR</td>
<td>AND</td>
<td>OR</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
</tr>
</tbody>
</table>
APPENDIX C  Mask Options

The following table lists the mask options of the MB89890 series.

### List of Mask Options

Table C-1  List of Mask Options

<table>
<thead>
<tr>
<th>No.</th>
<th>Product type</th>
<th>Setting</th>
<th>MB89898</th>
<th>MB89899</th>
<th>MB89P899</th>
<th>MB89VP890</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Setting</td>
<td>MB89898</td>
<td>MB89899</td>
<td>MB89P899</td>
<td>MB89VP890</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Specified to order mask</td>
<td>Can be set for every two pins P00 to P07, P10 to P17 P30 to P37, P60 to P67 P90 to P97 PA0 to PA7</td>
<td>Can be set for each pin P40 to P44, P70 to P77 P80 to P87</td>
<td>Set in units of the above combinations</td>
<td>No pull-up resistor, fixed</td>
</tr>
<tr>
<td>2</td>
<td>Power-on reset</td>
<td>• Power-on reset available • Power-on reset unavailable</td>
<td>Can be set</td>
<td>Can be set</td>
<td>Power-on reset available, fixed</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Oscillation stabilization wait time selection</td>
<td>Initial value of the oscillation stabilization wait time of the main clock can be selected by values of the following WTM1 and WTM0 bits.</td>
<td>Can be selected</td>
<td>Can be set</td>
<td>Oscillation stabilization $2^{18}$/ $F_{CH}$, fixed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WTM1 WTM0</td>
<td>MB89898</td>
<td>MB89899</td>
<td>MB89P899</td>
<td>MB89VP890</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 0</td>
<td>$2^3$/ $F_{CH}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 1</td>
<td>$2^{12}$/ $F_{CH}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0</td>
<td>$2^{16}$/ $F_{CH}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1</td>
<td>$2^{18}$/ $F_{CH}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reset pin output</td>
<td>• Reset output available • Reset output unavailable</td>
<td>Can be set</td>
<td>Can be set</td>
<td>Reset output available, fixed</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Clock mode selection</td>
<td>• Dual clock operation mode • Single clock operation mode fixed</td>
<td>Can be set</td>
<td>Can be set</td>
<td>Dual clock operation mode</td>
<td></td>
</tr>
</tbody>
</table>
Table C-2  Order Model

<table>
<thead>
<tr>
<th>Model</th>
<th>Package</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB89898PF, MB89899PF, MB89P899PF</td>
<td>Plastic/QFP, 100-pin (FPT-100P-M06)</td>
<td></td>
</tr>
<tr>
<td>MB89PV890CF</td>
<td>Ceramic/MQFP, 100-pin (MQP-100C-P01)</td>
<td></td>
</tr>
</tbody>
</table>
Appendix D describes the PROM programming.

D.1 "PROM Programming"
D.2 "Programming to the One-time PROM"
D.3 "Programming to the EPROM with Piggyback/Evaluation Chip"
D.1 PROM Programming

MB89P899 has the PROM mode, which is a function equivalent to MBM27C1001A. By using a special adapter, data can be written by a general-purpose ROM programmer. Note, however, that the electronic signature mode cannot be used.

- **One-time PROM Product**
  MB89P899 is available as a one-time PROM product.
  
  This product type contains PROM inside the CPU. By using a special adapter, data can be written by a general-purpose ROM programmer, but because an all-bit write test cannot be performed, a 100% write yield is not always possible. For more details, see Section D.2 "Programming to the One-time PROM".

- **Piggyback/Evaluation Product**
  MB89PV890 is available as a piggyback/evaluation product.
  
  This product type does not contain PROM inside the CPU.
  
  By using a special adapter after removing the external ROM (MBM27C512-20TV), data can be written by a general-purpose ROM programmer.
  
  For more details, see Section D.3 "Programming to the EPROM with the Piggyback/Evaluation Chip".
D.2 Programming to the One-time PROM

This section describes the programming to the PROM with MB89P899.

- ROM Programmer Adapter and Recommended Programmer

  Connect the jumper pin of the adapter on the VSS side.

  Depending on the ROM programmer and to ensure efficient data writing, a capacitor of about 0.1 μF should be inserted between VPP and VSS or between VCC and VSS.

  Table D.2-1 "ROM Programmer Adapter and Recommended ROM Programmer" lists the ROM programmer adapter and recommended ROM programmer.

**Table D.2-1 ROM Programmer Adapter and Recommended ROM Programmer**

<table>
<thead>
<tr>
<th>Package name</th>
<th>Conforming adapter type, Sunhayato Corporation</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-100P-M06</td>
<td>ROM-100QF-32DP-8LA</td>
</tr>
</tbody>
</table>

Contact phone No.: Sunhayato Corp., TEL: (81) 3-3986-0403

- Memory Map in PROM Mode

  Figure D.2-1 "Memory Map in PROM Mode" shows the memory map in PROM mode. Write option data to the option setting area, referring to "Bit Map of the PROM Options."

**Figure D.2-1 Memory Map in PROM Mode**
Programming to the PROM

1. Set the EPROM programmer for MBM27C1001.

2. Load program data to 1000H to FFFFH of the EPROM programmer. Load option information to 0FE4H to 0FFCH (For the correspondence to each option, see "Bit Map of the PROM Options").

3. Write data of 0FE4H to 0FFCH and 1000H to FFFFH using the EPROM programmer.

Bit Map of the PROM Options

Table D.2-2 Bit Map of the PROM Options (MB89P899)

<table>
<thead>
<tr>
<th>Address</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>00FE4H</td>
<td>Empty</td>
<td>Empty</td>
<td>Empty</td>
<td>Clock mode</td>
<td>Reset pin</td>
<td>Power-on</td>
<td>Oscillation stabiliz</td>
<td>11:2^16/\text{F}_{CH}</td>
</tr>
<tr>
<td></td>
<td>Read/Write enabled</td>
<td>Read/Write enabled</td>
<td>Read/Write enabled</td>
<td>selection</td>
<td>output</td>
<td>reset</td>
<td>wait time</td>
<td>10:2^15/\text{F}_{CH}</td>
</tr>
<tr>
<td>00FE8H</td>
<td>P17 to P16 Pull-up</td>
<td>P15 to P14 Pull-up</td>
<td>P13 to P12 Pull-up</td>
<td>P11 to P10 Pull-up</td>
<td>P07 to P06 Pull-up</td>
<td>P05 to P04 Pull-up</td>
<td>P03 to P02 Pull-up</td>
<td>P01 to P00 Pull-up</td>
</tr>
<tr>
<td></td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
<tr>
<td>00FEC,H</td>
<td>P67 to P66 Pull-up</td>
<td>P65 to P64 Pull-up</td>
<td>P63 to P62 Pull-up</td>
<td>P61 to P60 Pull-up</td>
<td>P37 to P36 Pull-up</td>
<td>P35 to P34 Pull-up</td>
<td>P33 to P32 Pull-up</td>
<td>P31 to P30 Pull-up</td>
</tr>
<tr>
<td></td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
<tr>
<td>00FF0H</td>
<td>PA7 to PA6 Pull-up</td>
<td>PA5 to PA4 Pull-up</td>
<td>PA3 to PA2 Pull-up</td>
<td>PA1 to PA0 Pull-up</td>
<td>P97 to P96 Pull-up</td>
<td>P95 to P94 Pull-up</td>
<td>P93 to P92 Pull-up</td>
<td>P91 to P90 Pull-up</td>
</tr>
<tr>
<td></td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
<tr>
<td>00FF4H</td>
<td>Empty</td>
<td>Empty</td>
<td>Empty</td>
<td>P44 Pull-up</td>
<td>P43 Pull-up</td>
<td>P42 Pull-up</td>
<td>P41 Pull-up</td>
<td>P40 Pull-up</td>
</tr>
<tr>
<td></td>
<td>Read/Write enabled</td>
<td>Read/Write enabled</td>
<td>Read/Write enabled</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
<tr>
<td>00FF8H</td>
<td>P77 Pull-up</td>
<td>P76 Pull-up</td>
<td>P75 Pull-up</td>
<td>P74 Pull-up</td>
<td>P73 Pull-up</td>
<td>P72 Pull-up</td>
<td>P71 Pull-up</td>
<td>P70 Pull-up</td>
</tr>
<tr>
<td></td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
<tr>
<td>00FFC,H</td>
<td>P87 Pull-up</td>
<td>P86 Pull-up</td>
<td>P85 Pull-up</td>
<td>P84 Pull-up</td>
<td>P83 Pull-up</td>
<td>P82 Pull-up</td>
<td>P81 Pull-up</td>
<td>P80 Pull-up</td>
</tr>
<tr>
<td></td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
<td>1: No</td>
</tr>
<tr>
<td></td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
<td>0: Yes</td>
</tr>
</tbody>
</table>

- Each bit is set to "1" when deleted.
- Note that the address in the option section can accept every four bytes to support the 4-byte write mode.
- \text{F}_{CH}: Source oscillation frequency
Recommended Screening Conditions

We recommend high-temperature aging as a screening method before mounting for the one-time PROM to which no microcomputer program has been written. Figure D.2-2 "Flow of Screening" shows the flow of screening.

Yield of the Write Operation

Due to the features of the one-time PROM to which no microcomputer program has been written, an all-bit write test cannot be performed, and so a 100% write yield is not always possible.
D.3 Programming to the EPROM with Piggyback/Evaluation Chip

This section describes the programming to the EPROM with piggyback/evaluation chip.

- **EPROM Used**
  MBM27C512-20TV

- **Programming Socket Adapter**
  To program to the EPROM using an ROM programmer, use the socket adapter (manufactured by Sunhayato Corp.) listed below.

  **Table D.3-1 Programming Socket Adapter**

<table>
<thead>
<tr>
<th>Package</th>
<th>Adapter socket part number</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCC-32 (rectangle)</td>
<td>ROM-32LC-28DP-YG</td>
</tr>
</tbody>
</table>

  Contact phone No.: Sunhayato Corp., TEL: (81) 3-3986-0403

- **Memory Space**

  **Figure D.3-1 Memory Map of the Piggy/Ever-chip**

- **Write Data to the EPROM**
  1. Set the EPROM programmer for MBM27C512.
  2. Load program data to 1000H to FFFFH of the EPROM programmer.
  3. Write the data of 1000H to FFFFH using the EPROM programmer.
The following table lists the pin status in each operation of the MB89890 series.

### Pin Status in Each Operation

**Table E-1 Pin Status in Each Operation**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>During sleep</th>
<th>During stop (SPL=0)</th>
<th>During stop (SPL=1)</th>
<th>Watch mode (SPL=0)</th>
<th>Watch mode (SPL=1)</th>
<th>During reset</th>
</tr>
</thead>
</table>
|          | Main RUN | Sub-RUN | Main sleep | Sub-sleep | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-stop | Main stop | Sub-
Table E-1 Pin Status in Each Operation (Continued)

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>During sleep</th>
<th>During stop (SPL=0)</th>
<th>During stop (SPL=1)</th>
<th>Watch mode (SPL=0)</th>
<th>Watch mode (SPL=1)</th>
<th>During reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Main RUN</td>
<td>Sub- RUN</td>
<td>Main sleep</td>
<td>Sub- sleep</td>
<td>Main stop</td>
<td>Sub- stop</td>
<td>Main stop</td>
</tr>
<tr>
<td>P30/PWM to P37/SO1</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td></td>
</tr>
<tr>
<td>P40 to P44</td>
<td>Port output</td>
<td>Port output</td>
<td>Retaining</td>
<td>Hi-z</td>
<td>Retaining</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P50/AN00 to P57/AN07</td>
<td>Port output/analag input</td>
<td>Port output/analag input</td>
<td>Port output/analag input</td>
<td>Hi-z</td>
<td>Port output/analag input</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P60/TMO1 to P67/BSO1</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td></td>
</tr>
<tr>
<td>P70/SK2 to P75/BSO2</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td></td>
</tr>
<tr>
<td>P76 P77</td>
<td>Port I/O</td>
<td>Port I/O</td>
<td>Port I/O</td>
<td>Hi-z</td>
<td>Retaining</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P80 to P87</td>
<td>Port output</td>
<td>Port output</td>
<td>Retaining</td>
<td>Hi-z</td>
<td>Retaining</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P90/INT20 to P97/INT27</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *3</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *3</td>
<td></td>
</tr>
<tr>
<td>PA0/INT28 to PA3/INTB</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td></td>
</tr>
<tr>
<td>PA4/INT0 to PA7/INT3</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td>Port I/O, resource I/O</td>
<td>Hi-z *1</td>
<td></td>
</tr>
</tbody>
</table>

*1: The input level is fixed to prevent leakage due to input release.
*2: The reset pin may be set for output depending on the option settings.
*3: To use the external interrupt, the input level is not fixed.
Hi-z: Indicates high impedance.
SPL: Pin status designation bit of the standby control register (STBC)
Retaining: The pin set for output retains the pin status (level) just before mode change.
INDEX

The index follows on the next page.
This is listed in alphabetic order.
## Index

### Numerics

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-bit data in stack, storage of</td>
<td>27</td>
</tr>
<tr>
<td>16-bit data storage in RAM</td>
<td>26</td>
</tr>
<tr>
<td>16-bit operand, storage of</td>
<td>26</td>
</tr>
<tr>
<td>8/16-bit timer/counter interrupt, register and vector table related to</td>
<td>209</td>
</tr>
<tr>
<td>8/16-bit timer/counter, block diagram of</td>
<td>193</td>
</tr>
<tr>
<td>8/16-bit timer/counter, block diagram of pin related to</td>
<td>196</td>
</tr>
<tr>
<td>8/16-bit timer/counter, interrupt of</td>
<td>208</td>
</tr>
<tr>
<td>8/16-bit timer/counter, pin related to</td>
<td>195</td>
</tr>
<tr>
<td>8/16-bit timer/counter, precaution on use of</td>
<td>220</td>
</tr>
<tr>
<td>8/16-bit timer/counter, register related to</td>
<td>198</td>
</tr>
<tr>
<td>8-bit PWM timer interrupt, register and vector table related to</td>
<td>176</td>
</tr>
<tr>
<td>8-bit PWM timer, block diagram of</td>
<td>167</td>
</tr>
<tr>
<td>8-bit PWM timer, block diagram of pin related to</td>
<td>170</td>
</tr>
<tr>
<td>8-bit PWM timer, pin related to</td>
<td>169</td>
</tr>
<tr>
<td>8-bit PWM timer, precaution on use of</td>
<td>184</td>
</tr>
<tr>
<td>8-bit PWM timer, register related to</td>
<td>171</td>
</tr>
<tr>
<td>8-bit serial I/O interrupt, register and vector table related to</td>
<td>239</td>
</tr>
<tr>
<td>8-bit serial I/O, block diagram of</td>
<td>229</td>
</tr>
<tr>
<td>8-bit serial I/O, block diagram of pin related to</td>
<td>233</td>
</tr>
<tr>
<td>8-bit serial I/O, pin related to</td>
<td>231</td>
</tr>
<tr>
<td>8-bit serial I/O, precaution on use of</td>
<td>247</td>
</tr>
<tr>
<td>8-bit serial I/O, register related to</td>
<td>234</td>
</tr>
</tbody>
</table>

### A

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D control register 1 (ADC1)</td>
<td>350</td>
</tr>
<tr>
<td>A/D control register 2 (ADC2)</td>
<td>352</td>
</tr>
<tr>
<td>A/D conversion function, operation of</td>
<td>357</td>
</tr>
<tr>
<td>A/D conversion function, programming example of</td>
<td>361</td>
</tr>
<tr>
<td>A/D conversion function, starting</td>
<td>357</td>
</tr>
<tr>
<td>A/D conversion mode, interrupt in</td>
<td>356</td>
</tr>
<tr>
<td>A/D converter interrupt, register and vector table related to</td>
<td>356</td>
</tr>
<tr>
<td>A/D converter, block diagram of</td>
<td>345</td>
</tr>
<tr>
<td>A/D converter, block diagram of pin related to</td>
<td>348</td>
</tr>
<tr>
<td>A/D converter, caution on use of</td>
<td>359</td>
</tr>
<tr>
<td>A/D converter, pin related to</td>
<td>348</td>
</tr>
<tr>
<td>A/D converter, register related to</td>
<td>349</td>
</tr>
<tr>
<td>A/D converter, supply voltage of</td>
<td>347</td>
</tr>
<tr>
<td>A/D converting function</td>
<td>344</td>
</tr>
<tr>
<td>A/D data register (ADCD)</td>
<td>354</td>
</tr>
<tr>
<td>ADCD register setting example in sensing mode</td>
<td>355</td>
</tr>
<tr>
<td>arithmetic instruction</td>
<td>461</td>
</tr>
</tbody>
</table>

### B

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bidirectional serial I/O performing</td>
<td>249</td>
</tr>
<tr>
<td>bit controlling acceptance of interrupt</td>
<td>31</td>
</tr>
<tr>
<td>bit indicating calculation result</td>
<td>30</td>
</tr>
<tr>
<td>bit manipulation instruction, read operation upon the execution of</td>
<td>458</td>
</tr>
<tr>
<td>branch instruction</td>
<td>463</td>
</tr>
<tr>
<td>buzzer output function</td>
<td>302</td>
</tr>
<tr>
<td>buzzer output, block diagram of</td>
<td>304</td>
</tr>
<tr>
<td>buzzer output, block diagram related to</td>
<td>305</td>
</tr>
<tr>
<td>buzzer output, pin related to</td>
<td>305</td>
</tr>
<tr>
<td>buzzer output, program example of</td>
<td>309</td>
</tr>
<tr>
<td>buzzer output, register related to</td>
<td>306</td>
</tr>
<tr>
<td>buzzer register (BZCR)</td>
<td>307</td>
</tr>
</tbody>
</table>

### C

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock controller block diagram</td>
<td>54</td>
</tr>
<tr>
<td>clock generator</td>
<td>52</td>
</tr>
<tr>
<td>clock mode operating state</td>
<td>59</td>
</tr>
<tr>
<td>clock pulse supply function</td>
<td>367</td>
</tr>
<tr>
<td>clock supply function</td>
<td>141</td>
</tr>
<tr>
<td>clock supply function, operation of</td>
<td>147</td>
</tr>
<tr>
<td>clock supply map</td>
<td>50</td>
</tr>
<tr>
<td>clock supply mode, operation in</td>
<td>373</td>
</tr>
<tr>
<td>condition code register (CCR) configuration</td>
<td>30</td>
</tr>
<tr>
<td>counter function</td>
<td>192</td>
</tr>
<tr>
<td>counter function, operation of</td>
<td>212</td>
</tr>
</tbody>
</table>

### D

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>data load timing</td>
<td>276</td>
</tr>
<tr>
<td>dedicated register configuration</td>
<td>28</td>
</tr>
<tr>
<td>dedicated register, function of</td>
<td>28</td>
</tr>
<tr>
<td>difference among model and precaution in selecting model</td>
<td>6</td>
</tr>
<tr>
<td>digital low-pass filter, operation of</td>
<td>406</td>
</tr>
<tr>
<td>DTMF control register (DTMC)</td>
<td>386</td>
</tr>
<tr>
<td>DTMF data register (DTMD)</td>
<td>388</td>
</tr>
<tr>
<td>DTMF generator, block diagram of</td>
<td>382</td>
</tr>
</tbody>
</table>
INDEX

G
gear function (main clock speed switching function) ........................................ 60
general-purpose register configuration .................................................. 34
general-purpose register, feature of .................................................. 35
general-purpose register area
   (address 0100H to 01FFH) .................................................. 24
GTMF generator, programming example of ........................................ 390

I
I/O circuit type ................................................................. 15
I/O map .............................................................................. 444
I/O pin setting of serial I/O port ............................................... 296
I/O port, function of ................................................................. 90
I/O port, program example of .................................................. 137
I/O shift timing ................................................................. 275
instruction cycle (tinst) ................................................................. 58
instruction map ......................................................................... 465
internal shift clock useing .............................................................. 244
interrupt at interval timer function being ON ........................................ 146
interrupt at operating of external interrupt circuit 1 ................................ 323
interrupt level setting register 1, 2, 3 (ILR1, 2, 3),
   configuration of ......................................................................... 37
interrupt of external interrupt circuit 1, register
   and vector table related to .......................................................... 323
interrupt of modem signal output circuit, register
   and vector table related to .......................................................... 431
interrupt operation, flow of .............................................................. 38
interrupt processing time ................................................................. 41
interrupt request from peripheral function ........................................ 36
interval timer function ................................................................. 140, 190
interval timer function (square wave output function).......................... 164
interval timer function (timebase timer),
   operation of ........................................................................ 147
interval timer function (watch interrupt) ........................................ 366
interval timer function mode, interrupt in ........................................ 176
interval timer function mode, program
   example in ........................................................................ 186
interval timer function, operation of ........................................ 177, 210
interval timer function, program example of ........................................ 222
interval timer mode (watch interrupt),
   interrupt in ........................................................................ 372
items in the instruction list ................................................................. 450

M
main clock mode, operation in ......................................................... 60
main clock oscillation stabilization wait time ........................................ 62
mask option, list of ..................................................................... 466
<table>
<thead>
<tr>
<th>Page Numbers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>344</td>
<td>sensing function</td>
</tr>
<tr>
<td>358</td>
<td>sensing function, operation of</td>
</tr>
<tr>
<td>362</td>
<td>sensing function, programming example of</td>
</tr>
<tr>
<td>358</td>
<td>sensing function, operation of</td>
</tr>
<tr>
<td>353</td>
<td>product feature</td>
</tr>
<tr>
<td>472</td>
<td>recommended screening condition</td>
</tr>
<tr>
<td>127</td>
<td>port 7, function of register of</td>
</tr>
<tr>
<td>128</td>
<td>port 7, operation of</td>
</tr>
<tr>
<td>125</td>
<td>port 7, pin of</td>
</tr>
<tr>
<td>126</td>
<td>port 7, register of</td>
</tr>
<tr>
<td>130</td>
<td>port 9, A, configuration of</td>
</tr>
<tr>
<td>133</td>
<td>port 9, A, function of register of</td>
</tr>
<tr>
<td>135</td>
<td>port 9, A, operation of</td>
</tr>
<tr>
<td>131</td>
<td>port 9, A, pin of</td>
</tr>
<tr>
<td>132</td>
<td>port 9, A, register of</td>
</tr>
<tr>
<td>283</td>
<td>precaution of using serial I/O with 1-byte buffer</td>
</tr>
<tr>
<td>473</td>
<td>programming socket adapter</td>
</tr>
<tr>
<td>470</td>
<td>PROM mode, memory map in</td>
</tr>
<tr>
<td>471</td>
<td>PROM option, bit map of</td>
</tr>
<tr>
<td>471</td>
<td>PROM, programming to</td>
</tr>
<tr>
<td>224</td>
<td>pulse counter function, program example of</td>
</tr>
<tr>
<td>174</td>
<td>PWM compare register (COMR)</td>
</tr>
<tr>
<td>172</td>
<td>PWM control register</td>
</tr>
<tr>
<td>165</td>
<td>PWM timer function</td>
</tr>
<tr>
<td>188</td>
<td>PWM timer function, program example in</td>
</tr>
<tr>
<td>179</td>
<td>PWM timer function, operation of</td>
</tr>
<tr>
<td>458</td>
<td>read-modify-write operation</td>
</tr>
<tr>
<td>472</td>
<td>register and vector table related to timebase</td>
</tr>
<tr>
<td>146</td>
<td>timer interrupt</td>
</tr>
<tr>
<td>33</td>
<td>register bank pointer (RP) configuration</td>
</tr>
<tr>
<td>48</td>
<td>reset operation flow</td>
</tr>
<tr>
<td>44</td>
<td>reset source</td>
</tr>
<tr>
<td>45</td>
<td>reset source and main clock oscillation stabilization wait time</td>
</tr>
<tr>
<td>470</td>
<td>ROM programmer adapter and recommended programmer</td>
</tr>
<tr>
<td>238</td>
<td>serial data register (SDR)</td>
</tr>
<tr>
<td>267</td>
<td>serial data register with 1-byte buffer (SBDR)</td>
</tr>
<tr>
<td>268</td>
<td>serial flag register with 1-byte buffer (SBFR)</td>
</tr>
<tr>
<td>228, 256</td>
<td>serial I/O function</td>
</tr>
<tr>
<td>239</td>
<td>serial I/O operation, interrupt during</td>
</tr>
<tr>
<td>289</td>
<td>serial I/O port switching circuit, block diagram of</td>
</tr>
<tr>
<td>292</td>
<td>serial I/O port switching circuit, pin related to</td>
</tr>
<tr>
<td>291</td>
<td>serial I/O port switching circuit, precaution of using</td>
</tr>
<tr>
<td>298</td>
<td>serial I/O port switching circuit, program example of</td>
</tr>
<tr>
<td>293</td>
<td>serial I/O port switching circuit, register related to</td>
</tr>
<tr>
<td>288</td>
<td>serial I/O port switching function</td>
</tr>
<tr>
<td>294</td>
<td>serial I/O port switching register (SSEL)</td>
</tr>
<tr>
<td>258, 263</td>
<td>serial I/O with 1-byte buffer, block diagram of</td>
</tr>
<tr>
<td>271</td>
<td>serial I/O with 1-byte buffer, interrupt of</td>
</tr>
<tr>
<td>274</td>
<td>serial I/O with 1-byte buffer, operation of</td>
</tr>
<tr>
<td>435</td>
<td>serial I/O with 1-byte buffer, operation of using of output data of</td>
</tr>
<tr>
<td>261</td>
<td>serial I/O with 1-byte buffer, pin related to</td>
</tr>
<tr>
<td>284</td>
<td>serial I/O with 1-byte buffer, program example of</td>
</tr>
<tr>
<td>273</td>
<td>serial I/O with 1-byte buffer, register and vector table related to</td>
</tr>
<tr>
<td>264</td>
<td>serial I/O with 1-byte buffer, register related to</td>
</tr>
<tr>
<td>282</td>
<td>serial I/O with 1-byte buffer, state transition diagram of</td>
</tr>
<tr>
<td>277</td>
<td>serial I/O with 1-byte buffer, transfer operation flow of</td>
</tr>
<tr>
<td>242</td>
<td>serial input operation</td>
</tr>
<tr>
<td>253</td>
<td>serial input, program example of</td>
</tr>
<tr>
<td>279</td>
<td>serial input/output carrying out in both direction</td>
</tr>
<tr>
<td>235</td>
<td>serial mode register</td>
</tr>
<tr>
<td>265</td>
<td>serial mode register with 1-byte buffer (SBMR)</td>
</tr>
<tr>
<td>240</td>
<td>serial output operation</td>
</tr>
<tr>
<td>241, 243</td>
<td>serial output, operation at completion of</td>
</tr>
<tr>
<td>251</td>
<td>serial output, program example of</td>
</tr>
<tr>
<td>275</td>
<td>shift/load operation</td>
</tr>
<tr>
<td>86</td>
<td>single chip mode</td>
</tr>
<tr>
<td>68</td>
<td>sleep mode operation</td>
</tr>
<tr>
<td>455</td>
<td>special instruction</td>
</tr>
<tr>
<td>215</td>
<td>square wave output initial setting function (timer 1), operation of</td>
</tr>
<tr>
<td>216</td>
<td>square wave output initial setting function (timer 2), operation of</td>
</tr>
</tbody>
</table>
INDEX

stack area for interrupt processing......................... 43
stack operation at start of interrupt processing ...... 42
stack operation at time of restoring from
interrupt.......................................................... 42
standby control register ...................................... 72
standby mode ....................................................... 64
standby mode and interrupt, transition to ............... 84
standby mode and stop during operation,
operation in .......................................................... 181
standby mode by interrupt, release of ...................... 84
standby mode setting, precaution on ...................... 84
standby mode, operating states in ......................... 66
state after mode data fetch .................................... 49
state of pin during reset operation.......................... 49
state of pin in standby mode .................................. 82
status transition chart 1 (with power-on reset,
dual clock operation) ............................................. 74
status transition chart 2 (without power-on reset,
dual clock operation) ............................................. 76
status transition chart 3 (single clock system) ......... 79
stop and restart of timer ....................................... 218
stop mode operation ............................................. 69
sub-clock mode, operation in ................................ 61
sub-clock mode, standby mode, and stop
during operation, operation in ............................. 219
sub-clock oscillation stabilization wait time .......... 63
symbols used with instruction ................................. 449
system clock control register (SYCC)
configuration .......................................................... 56

T
timebase timer control register (TBTC) ................. 144
timebase timer, block diagram of ......................... 142
timebase timer, operation of ................................. 147
timebase timer, precaution at using ....................... 149
timebase timer, program example of ..................... 151
timer 1 control register (T1CR) .............................. 199
timer 1 data register (T1DR) ................................. 204
timer 2 control register (T2CR) .............................. 202
timer 2 data register (T2DR) ................................. 206
Interval timer mode, operation in ......................... 373
transfer instruction ............................................... 459

U
using modem timer, precaution of ......................... 414

V
vector table area (address FFC0H to FFFFH) ......... 24

W
watch mode operation ............................................ 71
watch prescaler control register (WPCR) ............... 370
watch prescaler interrupt, register and vector
table related to .................................................. 372
watch prescaler, block diagram of ......................... 368
watch prescaler, operation of ................................. 374
watch prescaler, precaution on use of ..................... 375
watch prescaler, programming example of .............. 377
watchdog control register (WDTC) ......................... 157
watchdog timer function ....................................... 154
watchdog timer, block diagram of ......................... 155
watchdog timer, operation of ................................. 159
watchdog timer, precaution at using ...................... 161
watchdog timer, program example of ..................... 162
write operation, yield of ...................................... 472